Bu kitaba sığmayan daha neler var!

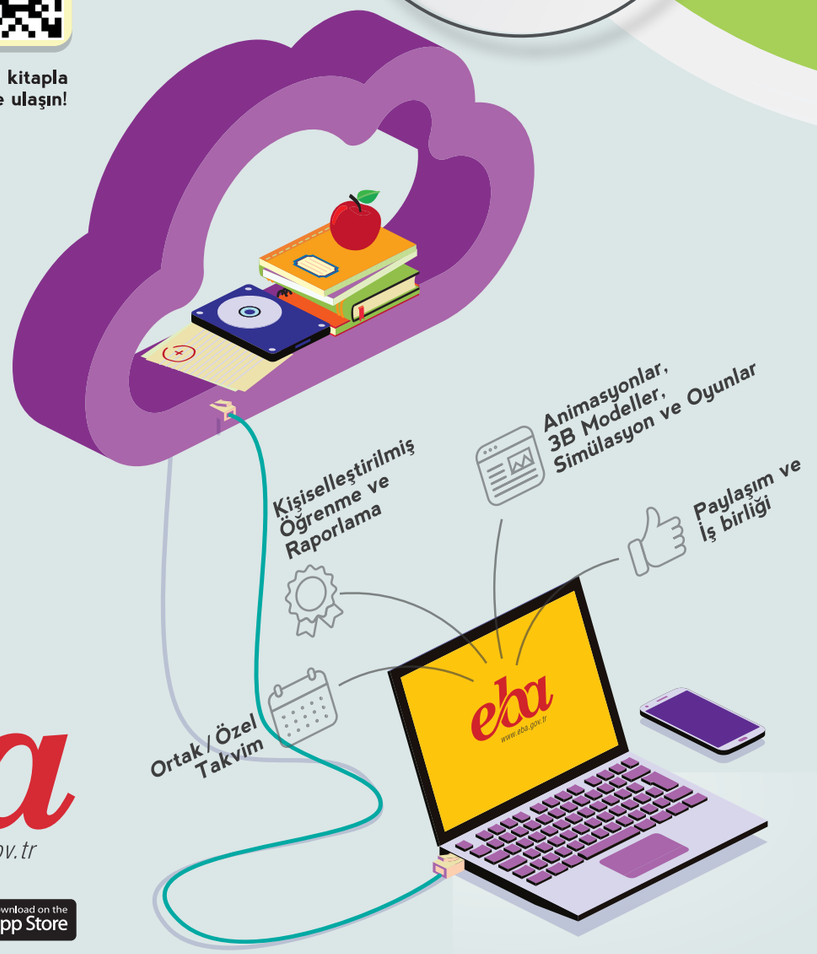
Karekodu okutun, bu kitapla ilgili EBA içeriklerine ulaşın!

ÖDS

ÖĞRENCİ/ÖĞRETMEN DESTEK SİSTEMİ

https://ods.eba.gov.tr

- Konu Anlatımlı Ders Videoları
- Soru Çözüm Videoları
- Ders Anlatım Videoları
- Çoktan Seçmeli Sorular



eba
www.eba.gov.tr

Google play | App Store

40181 700982

BU DERS KİTABI MİLLÎ EĞİTİM BAKANLIĞINCA ÜCRETSİZ OLARAK VERİLMİŞTİR.
PARA İLE SATILAMAZ.

ISBN: 978-975-11-7999-9

Bandrol Uygulamasına İlişkin Usul ve Esaslar Hakkında Yönetmelik'in 5'inci Maddesinin İkinci Fıkrası Çerçevesinde Bandrol Taşıması Zorunlu Değildir.

SİBER GÜVENLİK ALANI SİBER GÜVENLİK ATÖLYESİ 10 DERS MATERYALİ

MESLEKİ VE TEKNİK ANADOLU LİSESİ

SİBER GÜVENLİK ALANI

SİBER GÜVENLİK ATÖLYESİ

10

DERS MATERYALİ

MESLEKİ VE TEKNİK ANADOLU LİSESİ

SİBER GÜVENLİK ALANI

SİBER GÜVENLİK ATÖLYESİ

10

DERS MATERYALİ

YAZARLAR
Dr. Ahmet Nusret ÖZALP
Dr. Erdal ÖZDOĞAN
Atılım ÇİFTÇİ
Cahide ÜNAL
Murat KARATAŞ
Mustafa ALGÜL
Tarık BAĞRIYANIK

MİLLÎ EĞİTİM BAKANLIĞI YAYINLARI 9456
YARDIMCI VE KAYNAK KİTAPLAR DİZİSİ 3116

Bu kitap, siber güvenlik eğitimlerinde kullanılmak üzere bir kaynak olması amacıyla hazırlanmıştır. Siber Güvenlik Atölyesi kitabındaki konular, açıklamalar, kodlar ve araçlar tamamen eğitim amaçlıdır. Bu kitapta yer alan bilgilerin etik ve yasal amaçlar doğrultusunda kullanılması gerekir. Amacı dışında kullanımlardan kitaptaki yazarlar ve editörler sorumlu değildir.

HAZIRLAYANLAR

Dil Uzmanı
Melek DEMİR

Program Geliştirme Uzmanı
Ergül SIRKINTI

Rehberlik Uzmanı
Gülşen YALIN

Ölçme ve Değerlendirme Uzmanı
Günay DURUCAN

Görsel Tasarım Uzmanı
Özden ALTUN

ISBN: 978-975-11-7999-9

Millî Eğitim Bakanlığının 24.12.2020 gün ve 18433886 sayılı oluru ile Meslekî ve Teknik Eğitim Genel Müdürlüğünce ders materyali olarak hazırlanmıştır.

İSTİKLÂL MARŞI

Korkma, sönmez bu şafaklarda yüzen al sancak;
Sönmeden yurdumun üstünde tüten en son ocak.
O benim milletimin yıldızıdır, parlayacak;
O benimdir, o benim milletimindir ancak.

Çatma, kurban olayım, çehreni ey nazlı hilâl!
Kahraman ırkıma bir gül! Ne bu şiddet, bu celâl?
Sana olmaz dökülen kanlarımız sonra helâl.
Hakkıdır Hakk'a tapan milletimin istiklâl.

Ben ezelden beridir hür yaşadım, hür yaşarım.
Hangi çılgın bana zincir vuracakmış? Şaşarım!
Kükremiş sel gibiyim, bendimi çiğner, aşarım.
Yırtarım dağları, enginlere sığmam, taşarım.

Garbın âfâkını sarmışsa çelik zırhlı duvar,
Benim iman dolu göğsüm gibi serhaddim var.
Ulusun, korkma! Nasıl böyle bir imanı boğar,
Medeniyyet dediğin tek dişi kalmış canavar?

Arkadaş, yurduma alçakları uğratma sakın;
Siper et gövdeni, dursun bu hayâsızca akın.
Doğacaktır sana va'dettiği günler Hakk'ın;
Kim bilir, belki yarın, belki yarından da yakın.

Bastığın yerleri toprak diyerek geçme, tanı:
Düşün altındaki binlerce kefensiz yatanı.
Sen şehit oğlusun, incitme, yazıktır, atanı:
Verme, dünyaları alsan da bu cennet vatanı.

Kim bu cennet vatanın uğruna olmaz ki feda?
Şüheda fışkıracak toprağı sıksan, şüheda!
Cânı, cânânı, bütün varımı alsın da Huda,
Etmesin tek vatanımdan beni dünyada cüda.

Ruhumun senden İlâhî, şudur ancak emeli:
Değmesin mabedimin göğsüne nâmahrem eli.
Bu ezanlar -ki şehadetleri dinin temeli-
Ebedî yurdumun üstünde benim inlemeli.

O zaman vecd ile bin secde eder -varsa- taşım,
Her cerîhamdan İlâhî, boşanıp kanlı yaşım,
Fışkırır ruh-ı mücerret gibi yerden na'şım;
O zaman yükselerek arşa değer belki başım.

Dalgalan sen de şafaklar gibi ey şanlı hilâl!
Olsun artık dökülen kanlarımın hepsi helâl.
Ebediyyen sana yok, ırkıma yok izmihlâl;
Hakkıdır hür yaşamış bayrağımın hürriyyet;
Hakkıdır Hakk'a tapan milletimin istiklâl!

Mehmet Âkif Ersoy

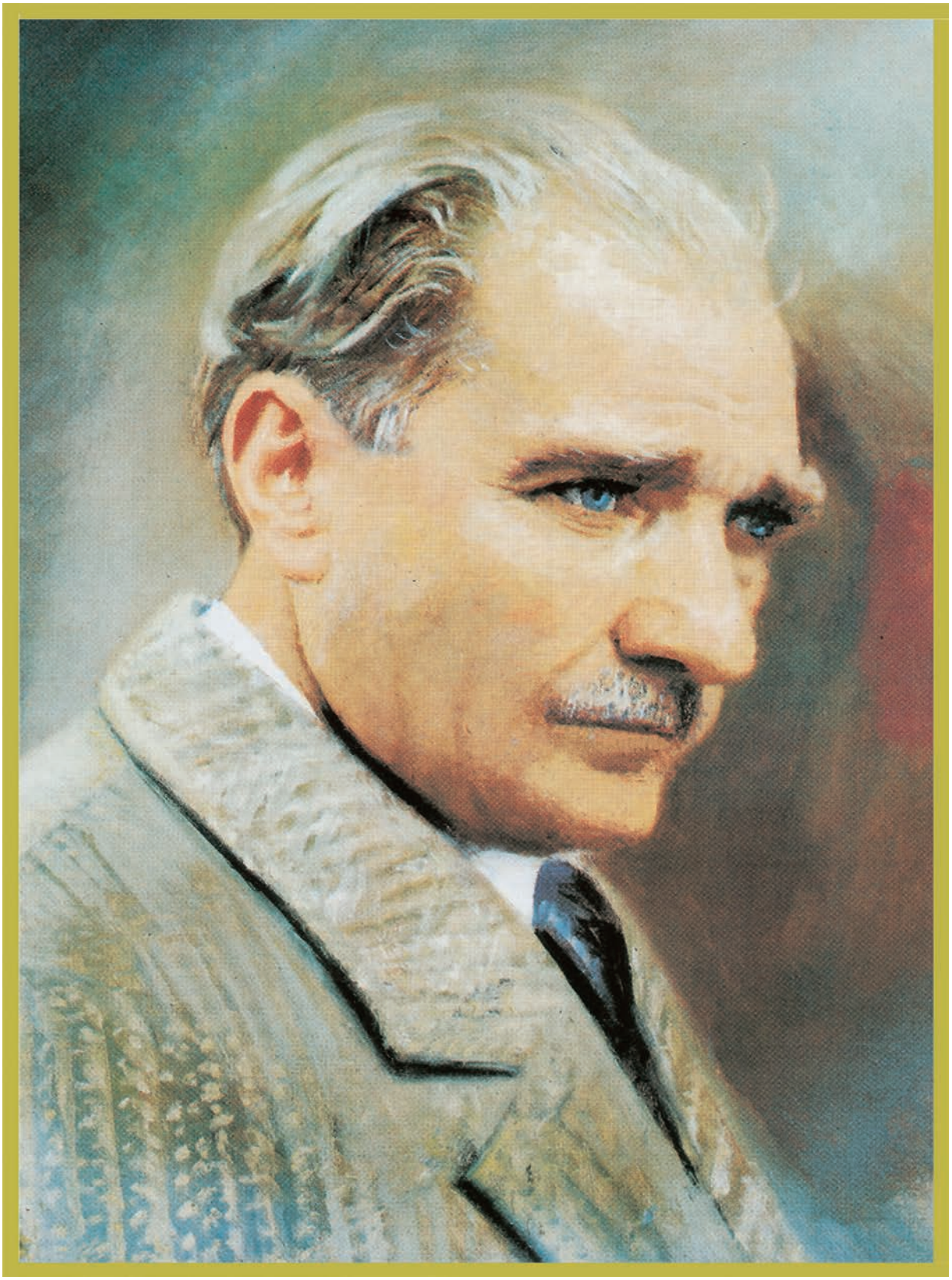
GENÇLİĞE HİTABE

Ey Türk gençliği! Birinci vazifen, Türk istiklâlini, Türk Cumhuriyetini, ilelebet muhafaza ve müdafaa etmektir.

Mevcudiyetinin ve istikbalinin yegâne temeli budur. Bu temel, senin en kıymetli hazinendir. İstikbalde dahi, seni bu hazineden mahrum etmek isteyecek dâhilî ve hâricî bedhahların olacaktır. Bir gün, istiklâl ve cumhuriyeti müdafaa mecburiyetine düşersen, vazifeye atılmak için, içinde bulunacağın vaziyetin imkân ve şeraitini düşünmeyeceksin! Bu imkân ve şerait, çok namüsait bir mahiyette tezahür edebilir. İstiklâl ve cumhuriyetine kastedecek düşmanlar, bütün dünyada emsali görülmemiş bir galibiyetin mümessili olabilirler. Cebren ve hile ile aziz vatanın bütün kaleleri zapt edilmiş, bütün tersanelerine girilmiş, bütün orduları dağıtılmış ve memleketin her köşesi bilfiil işgal edilmiş olabilir. Bütün bu şeraitten daha elîm ve daha vahim olmak üzere, memleketin dâhilinde iktidara sahip olanlar gaflet ve dalâlet ve hattâ hıyanet içinde bulunabilirler. Hattâ bu iktidar sahipleri şahsî menfaatlerini, müstevlîlerin siyasî emelleriyle tevhit edebilirler. Millet, fakr u zaruret içinde harap ve bîtap düşmüş olabilir.

Ey Türk istikbalinin evlâdı! İşte, bu ahval ve şerait içinde dahi vazifen, Türk istiklâl ve cumhuriyetini kurtarmaktır. Muhtaç olduğun kudret, damarlarındaki asil kanda mevcuttur.

Mustafa Kemal Atatürk



MUSTAFA KEMAL ATATÜRK

İÇİNDEKİLER

DERS MATERYALİNİN TANITIMI

HAZIRLIK ÇALIŞMALARI

1. Kişiler, verilerini neden saklar ve onları korumak için hangi güvenlik tedbirlerini alır? Düşüncelerinizi arkadaşlarınızla paylaşınız?
2. Veri tabanı güvenliği ile ilgili güncel tehditler nelerdir? Düşüncelerinizi sınıfta arkadaşlarınızla tartışınız.

Konulara başlanmadan önce yapılacak hazırlık çalışmalarını gösterir.

Araştırma

Veri tabanı şifrelerinin gücünü denetleyen şifre kırıcı araçları araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

Konuları pekiştirici ve destekleyici araştırmaları gösterir.

Sıra Sizde

Laboratuvardaki Kali işletim sistemi kurulu sanal makineye bettercap aracını yükleyiniz. LAN içindeki IP-MAC adresi eşleştirmelerini keşfediniz. Kendinize bir hedef IP adresi seçerek ARP önbellek zehirlemesi atağını gerçekleştiriniz.

Konuları destekleyici ve pekiştirici sıra sizde çalışmalarını gösterir.

8. Uygulama

İşlem adımlarına göre Tablo 8.10'daki normalizasyon kurallarını uygulayınız.

Konuları destekleyici ve pekiştirici uygulama çalışmalarını gösterir.

Tablo 8.10: Normalizasyon Kuralları Uygulanacak Tablo

Konuları destekleyici tabloları gösterir.

Proje No.	Proje Adı	Proje Yöneticisi	Öğrenci No.	Öğrenci Adı	Sınıfı
P1	Kütüphane sistemi	Erdal U.	125	Eda K.	12
P2	Kütüphane sistemi	Erdal U.	135	Canan E.	12
P3	Otel sistemi	Emel B.	118	Fatma T.	11
P4	Otel sistemi	Emel B.	352	Nur G.	11

3. Uygulama

İşlem adımlarına göre tablolar arasında ilişki oluşturunuz.

1. Adım: SQL kodunu çalıştırarak iki tablo ve bu tablolar içinde veriler olmasını sağlayınız.

```
CREATE DATABASE veritabani;
GO
USE veritabani;
GO
CREATE TABLE kategoriler(
        kimlik int PRIMARY KEY IDENTITY(1,1) NOT NULL,
        kategoriAdi nvarchar(50)
```

Örnek kod panelini gösterir.

İÇİNDEKİLER

Konuları ve konuların bulunduğu sayfa numaralarını gösterir.

ÖLÇME VE DEĞERLENDİRME

Öğrenme birimi ile ilgili ölçme ve değerlendirme çalışmalarını gösterir.

A. Aşağıdaki parantezlerin içine cümlelerde verilen bilgiler doğru ise "D", yanlış ise "Y" yazınız.

1. ( ) Veri tabanı tablolarında bir alana birden fazla veri girişi olabilir.

2. ( ) Herhangi bir tablonun tekrarlı bilgiler içerdiği duruma 2NF denir.

3. ( ) Tablodaki benzersiz değerler içeren her bir sütun veya sütunlar grubu aday anahtar ola
adlandırılır.

CEVAP ANAHTARI

Ders materyali içeriğinde yer verilen soruların cevaplarını gösterir.

1. ÖĞRENME BİRİMİ

A	**1.** Yanlış	**2.** Doğru	**3.** Doğru	**4.** Yanlış	**5.** Doğru
B	**6.** beyaz şapkalı	**7.** 6698	**8.** delil	**9.** imajını	**10.** fiziksel
C	**11.** C	**12.** A	**13.** E	**14.** B	**15.** D

GENEL AĞ KAYNAKÇASI VE GÖRSEL KAYNAKÇASI

Ders materyalindeki konuların genel ağ ve görsel kaynakçasını gösterir.

http://kitap.eba.gov.tr/karekod/Kaynak.php?KOD=3436

Kontrol Listesi

Konuları destekleyici ve pekiştirici sıra sizde çalışmalarının kontrol listesini gösterir.

Değerlendirme Ölçütleri	Evet	Hayır
1. Kurulum dosyasını indirdi.		
2. Kurulum türünü belirledi.		
3. Kurulumu başlattı.		
4. Kullanıcı hesabını oluşturdu.		
5. Yapılandırma işlemini gerçekleştirdi.		
6. Veri tabanı yönetim sistemi üzerinde veri tabanı oluşturdu.		
7. Veri tabanı tabloları oluşturdu.		
8. Oluşturduğu veri tabanı tablolarına veri girişi yaptı.		

"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.

SİBER GÜVENLİĞE GİRİŞ

1. ÖĞRENME BİRİMİ

KONULAR

1.1. SİBER GÜVENLİK ETİK İLKELERİ

1.2. SİBER GÜVENLİK KAVRAMLARI

1.3. FİZİKSEL GÜVENLİK

1.4. BİLGİ GÜVENLİĞİ STANDARTLARI

1.5. DİJİTAL ARŞİVLEME

1.6. ADLİ BİLİŞİM KAVRAMLARI

1.7. SİBER GÜVENLİK SERTİFİKA PROGRAMLARI

NELER ÖĞRENECEKSİNİZ?

- Siber güvenlik etik ilkelerini açıklama
- Siber güvenlik kavramlarını açıklama
- Ağ cihazlarının fiziksel güvenliğinin sağlanması ile ilgili uygulamalar yapma
- Bilgi güvenliği standartlarını açıklama
- Dijital arşivlemeyi açıklama
- Adli bilişim ile ilgili kavramları açıklama
- Siber güvenlik sertifika programlarını açıklama

ANAHTAR KELİMELER

Adli bilişim, arka kapı, beyaz şapka, bütünlük, dijital arşiv, erişilebilirlik, gizlilik, gri şapka, IoT, kişisel veri, kriptografi, saldırgan, saldırı, siber güvenlik, siyah şapka

HAZIRLIK ÇALIŞMALARI

1. Siber saldırılara karşı kişisel bilgisayar ve mobil telefonlarda ne tür önlemler alınabilir? Düşüncelerinizi arkadaşlarınızla paylaşınız.
2. Kişisel verilerin izinsiz bir şekilde kullanılması ne tür problemlere sebep olabilir? Arkadaşlarınızla değerlendiriniz.

1.1. SİBER GÜVENLİK ETİK İLKELERİ

Siber güvenlik etik ilkeleri, siber güvenlik alanında çalışan ve teknolojiyi kullanan kişilerin veya kuruluşların uyması gereken temel değerler ve davranış kurallarıdır. Bu ilkeler, siber güvenlik topluluğunun etik davranışı ve profesyonellik düzeyinin çerçevesini belirlemeyi amaçlar. Bu sayede bireylerin ve kuruluşların dijital dünyada dürüst, adil ve sorumluluk sahibi bir şekilde hareket etmeleri sağlanır.

1.1.1. Etik ve Bilgi Gizliliği

Etik, temel değerler ve normlar aracılığıyla bireylerin ve toplumların davranışlarını düzenleyen bir kavramdır. İnsanların birlikte yaşama biçimini şekillendiren kültürel, felsefi ve toplumsal faktörlerden etkilenen etik, doğru ile yanlışı, iyi ile kötüyü, haklı ile haksızı belirlemeye ve değerlendirmeye yardımcı olur. Adalet, sorumluluk gibi önemli prensipleri içererek insanların diğer insanlarla ve çevreleriyle olan ilişkilerini düzenler.

Etik, kişisel yaşamlardan iş dünyasına, sanattan bilimsel araştırmalara kadar her alanda büyük bir öneme sahiptir. Etik ilkesine uymayan eylemler, toplumda güven eksikliğine ve sorunlara neden olarak uzun vadede olumsuz sonuçlara yol açabilir. Bu nedenle etik düşünce ve davranış biçimleri, insanların birlikte yaşamaları ve karşılaştıkları zorlukları aşmaları için hayati öneme sahiptir.

Bilgi ve iletişim teknolojilerinin hızlı gelişimi, yaygınlaşması ve teknolojinin toplum ve bireyler üzerinde davranış biçimini değiştirmesi gibi etkenler, etik kavramının bu alanda bilişim etiği olarak kabul görmesini sağlamıştır (Görsel 1.1).

Görsel 1.1: Bilişim etiği

İnternet, sosyal medya gibi platformların yaygınlaşmasıyla ortaya çıkan bilgi kirliliği, siber zorbalık, veri ihlalleri gibi etik dışı uygulamalar, bilişim etiğinin önemini vurgular. Bilişim etiği, teknoloji geliştiricilerinin, karar alıcıların ve kullanıcıların bilinçli bir şekilde hareket etmesini teşvik eder. Bilişim etiği, bilgi gizliliğine önem verir. Günümüzde teknolojik gelişmelerin etkisiyle internet ortamında bulunan veri boyutu sürekli artış gösterir. Bu veriler de kötü niyetli kullanıcıların hedefi olabilir. Bu yüzden bilgi teknolojileri ve bilişim araçları kullanıcılarının bilişim etiği çerçevesinde davranmalarının sağlanması için kişisel, kurumsal ve hassas verilere yetkisiz erişim, izinsiz paylaşım, bilgi gizliliğinin ihlali gibi kötü niyetli davranışlardan uzak durması konusunda eğitilmeleri gerekir. Bilişim etiği, teknoloji kullanımının insan haklarına ve değerlerine saygılı olmasını sağlar.

1.1.2. Bilgi Güvenliğiyle İlgili Yasal Düzenlemeler, Problemler ve Telif Hakları

Bilgi teknolojilerinin hızlı gelişimi, dijital dünyanın yaygınlaşması, internet ortamında verilerin çoğalmasıyla birlikte bilgi sistemlerine yönelik siber saldırılar artış göstermiştir. Bilgi güvenliğini sağlayabilmek ve saldırganları caydırmak için yasal düzenlemelerin yapılması zorunlu hâle gelmiştir (Görsel 1.2).

Görsel 1.2: Bilgi güvenliğini sağlayan araçlar

Ülkemizde bilgi güvenliği ile ilgili yürürlükte olan yasal düzenlemelerden bazıları şunlardır:

- Türkiye Cumhuriyeti Anayasası
- 5237 sayılı Türk Ceza Kanunu
- 6698 sayılı Kişisel Verilerin Korunması Kanunu
- 5846 sayılı Fikir ve Sanat Eserleri Kanunu
- 5809 sayılı Elektronik Haberleşme Kanunu
- 5070 sayılı Elektronik İmza Kanunu
- 5651 sayılı İnternet Ortamında Yapılan Yayınların Düzenlenmesi ve Bu Yayınlar Yoluyla İşlenen Suçlarla Mücadele Edilmesi Hakkında Kanun
- Elektronik Haberleşme Sektöründe Şebeke ve Bilgi Güvenliği Yönetmeliği
- Elektronik Haberleşme Sektöründe Kişisel Verilerin İşlenmesi ve Gizliliğin Korunmasına İlişkin Yönetmelik
- Avrupa Konseyi bünyesinde hazırlanan ve ülkemizin de taraf olduğu Sanal Ortamda İşlenen Suçlar Sözleşmesi (Council of Europe: Convention on Cybercrime)
- 06.07.2019 tarihli ve 30823 sayılı Bilgi ve İletişim Güvenliği Tedbirleri konulu Cumhurbaşkanlığı Genelgesi

- 10.07.2020 tarihli Türkiye Cumhuriyeti Cumhurbaşkanlığı Dijital Dönüşüm Ofisi tarafından yayımlanan Bilgi ve İletişim Güvenliği Rehberi
- 10.10.2021 tarihli Türkiye Cumhuriyeti Cumhurbaşkanlığı Dijital Dönüşü Ofisi tarafından yayımlanan Bilgi ve İletişim Güvenliği Denetim Rehberi

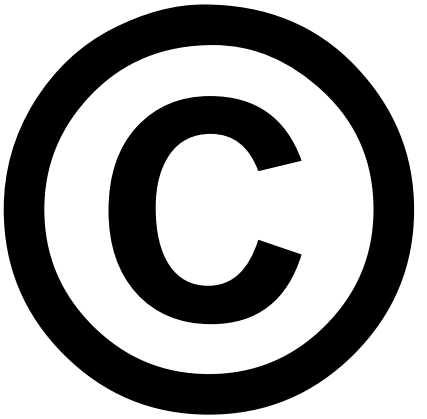
Teknolojinin hızla gelişmesi ve değişmesiyle birlikte bilgi güvenliği ile ilgili yasal düzenlemelerin güncelliğini koruması zorlaşır. Yeni tehditler ve saldırı yöntemleri ortaya çıktıkça mevcut yasaların yeni tehditlere uyum sağlaması gerekir. Ayrıca farklı ülkelerin farklı yasal düzenlemelerinin olması, uluslararası iş birliği ve veri paylaşımı konusunda bazı zorluklar ortaya çıkarır.

Bilgi güvenliği uygulamalarında karşılaşılan başlıca problemler şunlardır:

- Fikri mülkiyet hırsızlığı
- Telif hakları ihlalleri
- Veri ihlalleri
- Elektronik iletişim güvenliği ihlalleri
- Bilgi sistemlerine yetkisiz erişim ihlalleri
- Bilgi sistemlerine erişimin engellenmesi
- Tıbbi kayıtlara erişim ihlali ve tıbbi kayıtların ifşası
- Hassas verilere yetkisiz erişim
- Tersine mühendislik ihlalleri
- Personel bilgilerinin istismarı
- Müşteri bilgilerinin istismarı
- İnternet bankacılığı erişim bilgilerine yetkisiz sahip olunması
- Telif hakkıyla korunan bilgi, ürün ve esere yasa dışı erişim

Telif hakkı; bir kişinin veya kişilerin kendi düşünceleriyle ortaya çıkardığı, bilgi, düşünce, sanat eseri ve ürünün, kullanımı ve kopyalanması üzerinde sahip olduğu yasal haklardır. Bu haklar, ürün sahibinin eserlerini koruyarak adil rekabeti sağlar ve onlara eserlerini ticari olarak değerlendirme imkânı verir. Ülkemizde telif hakları, 5846 sayılı Fikir ve Sanat Eserleri Kanunu ile güvence altına alınmıştır.

Telif hakkı (Görsel 1.3) bulunan bilginin, ürünün, sanat eserinin izinsiz bir şekilde kullanılması, dijital ortamlarda yayımlanması, kopyalanması 5846 sayılı Fikir ve Sanat Eserleri Kanunu ile yasaklanmıştır. Ayrıca telif hakkı ihlali durumunda cezai sorumluluk ortaya çıkar.



Görsel 1.3: Telif hakkı sembolü

Araştırma

https://telifhaklari.ktb.gov.tr/ web sitesindeki "Telif Hakkı İhlali Halinde Ne Yapılabilir?" sekmesini ziyaret ederek telif hakkı ihlali hâlinde neler yapılabileceğini araştırınız.

1.1.3. Siber Güvenlik Uzmanı

Siber güvenlik uzmanı, bilişim sistemlerini ve bilgi teknolojileri altyapısını siber saldırılara ve tehditlere karşı korumakla görevli kişidir. Bilgisayar korsanlarının, kötü niyetli yazılımların ve diğer siber tehditlerin neden olduğu güvenlik açıklarını tespit etmek, önlemek ve zafiyetlere karşı çözüm üretmek için çalışır. Siber güvenlik uzmanının faaliyetleri; kurum çalışanlarını, müşterilerini ve paydaşlarını ahlaki olarak zararlardan korumalıdır (Görsel 1.4).

Görsel 1.4: Siber güvenlik uzmanı

Siber güvenlik uzmanının başlıca sorumlulukları şunlardır:

- Siber güvenlik faaliyetlerini yasalara ve etik ilkelere uygun bir şekilde gerçekleştirmek.
- Kurum ve müşteriye ait fikri mülkiyet haklarının ve ticari sırların ihlalinden kaçınmak.
- Kurum ve kuruluşların güncel yazılım ve uygulamaları kullanmalarını teşvik etmek.
- Tespit ettiği güvenlik açıklarını ve ihlalleri, uygun kanallar aracılığıyla yetkililere bildirmek.
- Sızma testi için izin ve yetki alarak yalnızca verilen izin ve yetki dâhilindeki işlemleri gerçekleştirmek.
- Mesleki faaliyet sırasında ortaya çıkan bilgilerin gizliliğini korumak.
- Yaptığı işlemlerin sonuçlarından sorumlu olmak.
- İşi gereği edindiği verileri menfaati için kullanmamak.
- Verileri saldırılardan korumak için güvenlik önlemi almak.
- Bilgi güvenliği ekibiyle koordineli çalışmak.

1.2. SİBER GÜVENLİK KAVRAMLARI

Siber güvenlik; bilgi teknolojilerinin kullanıldığı dijital ortamlarda bilişim sistemlerini, ağları ve verileri koruma, tehditlere karşı güvenliği sağlama sürecini içerir. Süreç içinde birçok kavram ortaya çıkmıştır. Bu kavramlar, dijital dünyada bilinçli ve güvende olmayı hedefler.

1.2.1. Güvenli Dijital Vatandaşlık

Güvenli dijital vatandaşlık, bireylerin bilişim teknolojilerini ve interneti güvenli bir şekilde kullanmasını, etik ve sorumlu bir şekilde davranmasını, dijital dünyadaki tehlikelerden korunmasını sağlayan bir kavramdır. Siber vatandaşlık veya e-vatandaşlık olarak da adlandırılır. Günümüzde dijital teknolojilerin yaygın kullanımıyla birlikte bireylerin dijital ortamlarda bilinçli, dikkatli olması ve dijital güvenlik önlemlerini alması çok önemlidir. Dijital araçları doğru ve sorumlu bir şekilde kullanabilmesi için eğitim ve farkındalık çalışmaları ile desteklenmesi gerekir. Görsel 1.5'te güvenli dijital vatandaşlıkla ilgili kavramlar verilmiştir.

Görsel 1.5: Güvenli dijital vatandaşlıkla ilgili kavramlar

Güvenli dijital vatandaşlıkla ilgili kavramların açıklamaları şunlardır:

Dijital İletişim: Bilgi ve iletişim teknolojilerinin kullanıldığı elektronik ortamlar aracılığıyla gerçekleştirilen iletişim sürecidir. İnternet, e-posta, sosyal ağlar, mesajlaşma, video konferans gibi dijital araçlar ve platformlar üzerinden veri iletilmesi, paylaşılması ve alınması sürecidir.

Dijital Erişim: Bireylerin bilgisayar, akıllı telefon, tablet vb. teknolojik cihazları kullanabilme ve dijital ortamlara erişebilme becerisidir.

Dijital Okuryazarlık: Bireylerin dijital ortamlardan elde ettiği, okuduğu her bilginin doğru olmayabileceğini değerlendirmesi ve bilgi kirliliğine dikkat ederek doğru, güvenilir kaynaklardan bilgiye ulaşabilmesidir.

Dijital Güvenlik: Dijital vatandaşların güvenliklerine dikkat etmesini ve gerekli önlemleri almasını gerektirir. Örneğin şifre ve parolaların güçlü bir şekilde belirlenmesi, aynı şifre ve parolanın farklı platformlarda kullanılmaması, antivirüs programı kullanılması gibi tedbirler dijital güvenlik için dikkat edilmesi gereken hususlardan bazılarıdır.

Dijital Etik: Dijital ortamlarda kullanıcılara saygılı olmayı ve empatik davranmayı gerekli kılan bir kavramdır. İnternet ortamında düşünce ve ifade özgürlüğüne saygı göstermek, başkalarını rahatsız etmemek, hakaret etmemek, zarar verici içerikler paylaşmamak vb. dijital etik davranışlara örnek gösterilebilir.

Dijital Haklar ve Sorumluluklar: Dijital dünyada bireylerin ve kullanıcıların sahip olduğu hakları ve bu hakları kullanırken uyması gereken sorumlulukları içeren kavramlardır.

Dijital Kanun: İnternet kullanımı esnasında suç olabilecek ihlallerin vatandaşlar tarafından bilinmesi ve bu ihlallere dikkat edilmesidir. Dijital vatandaşların görevi; sosyal medya ortamlarında zorbalık, şantaj, itibar zedeleme gibi siber suçların kanun önünde cezai yaptırımlarının bulunduğunun bilincinde olmak ve bu konuda diğer kullanıcıları bilinçlendirmektir.

Dijital Sağlık: Bilgisayar, akıllı telefon, tablet gibi internete erişim sağlayan cihazların uzun süreli kullanımından dolayı fiziksel ve ruhsal sorunlar ortaya çıkabilir. Bu nedenle cihazların uzun süreli kullanımından kaçınmak gerekir.

Dijital Ticaret: Ticari işlemlerin dijital ortamlara taşınmasıyla birlikte bankacılık, alışveriş gibi işlemler bu dijital ortamlardan gerçekleştirilmeye başlanmıştır. Dijital vatandaş, ticari işlemler gerçekleştirirken doğru ve güvenilir kaynakları tercih etmeli, dolandırıcılık ve fidye isteme gibi kötü niyetli tuzaklardan kendini koruyabilmelidir.

1.2.2. Siber Zorbalık

Siber zorbalık; dijital ortamlarda kişilere karşı saldırganlık içeren, tehdit edici, rahatsız edici veya zarar verici davranışları ifade eder. İnternet, sosyal ağ ortamları, e-posta, anlık mesajlaşma gibi dijital iletişim araçları üzerinden gerçekleştirilen siber zorbalık, fiziksel temas olmaksızın yapılan bir dijital saldırı türüdür. Bu saldırıyı gerçekleştirene **zorba**, saldırıya maruz kalan bireye ise **kurban** denir. Siber zorbalığa maruz kalan birey (Görsel 1.6), duygusal ve fiziksel olarak etkilenebilir ve bu etkiler uzun süre devam edebilir.

Görsel 1.6: Siber zorbalığa maruz kalmış birey

Siber zorbalık davranışları şunlardır:

- Sosyal ağ ortamlarından kötü niyetli mesajlar yazmak.
- Tehdit etmek ve korkutmak.
- Birey hakkında gerçek olmayan bilgiler yaymak.
- Utanç verici içerik ve resimlerle kişiyi hedef almak.
- Kişiye özel bilgileri izinsiz paylaşmak.
- Sosyal ağ ortamlarına erişim sağlamak için kullanıcı adı ve şifresini paylaşmaya zorlamak.
- Ele geçirilmiş sosyal ağ hesabı ile başkalarını hedef almak.

Siber zorbalığa karşı kurbanlar tarafından alınacak önlemler şunlardır:

- Öncelikle sakin olunmalı ve zorba ile **kesinlikle** temas kurulmamalıdır.
- Zorba, dijital ve sosyal ağ ortamlarından engellenmelidir.
- Siber zorbalığa maruz kalan birey, bu durumu bir aile büyüğüne, öğretmenine veya yardım edebilecek bir yakınına mutlaka bildirmelidir.
- Siber zorbalık ile ilgili kanıtlar (mesajlar, yorumlar, resimler vb.) ekran görüntüsü alınarak saklanmalıdır.
- Kanıtlarla birlikte yetkili birimlere müracaat edilmelidir.

Kurbanlar tarafından alınacak önlemlerin haricinde toplumsal farkındalık oluşturmak için ailelere, çocuklara ve öğrencilere siber zorbalığa karşı farkındalık eğitimleri düzenlenmelidir. Bu sayede mağduriyetlerin oluşmasının engellenmesi veya azaltılması sağlanmaya çalışılmalıdır.

1.2.3. Bilgi Güvenliği Unsurları

Bilgi, verilerin anlamlı ve kullanılabilir hâle getirilmiş biçimidir. Bilgi teknolojilerinin yaygınlaşmasıyla birlikte bilginin güvenliğinin sağlanması önemli bir husus hâline gelmiştir. **Bilgi güvenliği**; kişisel veya kurumsal her türlü bilginin yetkisiz şekilde ele geçirilmesi, ifşa edilmesi, yok edilmesi, değiştirilmesi gibi tehditlere karşı korunması için alınan tedbirleri içeren bir disiplindir.

Görsel 1.7: CIA üçlüsü

Gizlilik (confidentiality), bütünlük (integrity) ve **erişilebilirlik (availability)**; bilgi güvenliğinin üç temel unsurudur. Bilgi güvenliği unsurlarına kısaca **CIA üçlüsü** denir (Görsel 1.7). Bilgi güvenliğinde bu üç temel unsur eksiksiz bir şekilde sağlanırsa bilgi güvenliği de sağlanır. Bu unsurlardan biri ihlal edildiğinde güvenlik zafiyeti meydana gelir.

Gizlilik: Bilginin yetkisiz kişiler tarafından elde edilmesinin engellenmesidir. Bilgi, yetkisiz kişiler tarafından elde edilirse gizlilik unsuru ihlal edilmiştir. İnternet bankacılığı erişim bilgileri, e-posta şifreleri vb. bilgilerin yetkisiz kişilerce ele geçirilmesi, gizlilik unsurunun ihlal edilmesine örnek gösterilebilir.

Bütünlük: Bilginin yetkisiz kişiler tarafından değiştirilerek veya silinerek bütünlüğünün bozulmasıdır. Saldırganın, bankacılık işlemlerine ait kayıtları değiştirmesi, silmesi veya yeni bilgiler eklemesi, bütünlük unsurunun ihlal edilmesine neden olur.

Erişilebilirlik: Bilginin yetkili kişiler tarafından belirlenen zaman dilimlerinde sürekli erişilebilir olması durumudur. Erişilebilirlik ihlali, bu sürecin aksamasına veya engellenmesine neden olabilir. Saldırgan tarafından bir web sitesine gerçekleştirilen hizmet reddi saldırısı ile kullanıcıların web sitesine erişimi engellendiğinde erişilebilirlik unsuru ihlal edilir.

1.2.4. Hacker Kavramı

Hacker terimi; başlangıçta bilişim sistemleri, ağ ve yazılım alanlarında üst düzey teknik bilgi, beceri ve yeteneğe sahip kişilere verilen isimdi ancak günümüzde bu tanım değişmiştir. Hackerlara **bilgisayar korsanı** da denir. **Hackerlar** genellikle bilişim sistemlerindeki zafiyetleri tespit edip, bu zafiyetleri kullanarak sistemlere yetkisiz ve izinsiz giriş yapan, verileri ele geçiren, silen, değiştiren, menfaat sağlayan kişiler olarak tanımlanır (Görsel 1.8).

Görsel 1.8: Hacker

Hackerlar; amaçlarına göre siyah şapkalı, beyaz şapkalı, gri şapkalı olmak üzere üçe ayrılır.

Siyah Şapkalı Hacker: Zarar vermek, maddi kazanç sağlamak için sistemlere izinsiz giriş yapan, verileri ele geçiren, silen, değiştiren kötü niyetli bilgisayar korsanlarına verilen isimdir.

Beyaz Şapkalı Hacker: Etik hackerlar olarak bilinirler. Bilişim sistemlerinin güvenliğini artırmak, zafiyetleri tespit etmek ve zafiyetleri ortadan kaldırmak, saldırıları tespit etmek ve önlemek için çalışan, sızma testi yapan iyi niyetli hackerlardır. Beyaz şapkalı hackerlar, faaliyetlerini izin alarak gerçekleştirirler. Ayrıca kurumlarda ve şirketlerde siber güvenlik uzmanı olarak görev yaparlar.

Gri Şapkalı Hacker: Siyah şapka ve beyaz şapka hackerların arasında bir yerde duran hacker türüdür. Siyah şapka hacker gibi sistemlere izinsiz giriş yapabilir. Sistemlere zarar verebileceği gibi zarar vermeden zafiyetin giderilmesi için kuruma bilgi de aktarabilir.

Araştırma

Dünyada bilinen ilk hackerı ve yaptığı hack işlemini araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

1.2.5. Siber Saldırı ve Siber Saldırı Türleri

Siber saldırılar; siber alanlarda var olan zayıflıklardan yararlanarak bilişim sistemlerine, ağlara ve dijital altyapılara siber saldırganlar tarafından gerçekleştirilen kötü niyetli eylemlerdir. Saldırganlar sisteme erişim elde ettiğinde veri kopyalama, veri silme, veri değiştirme, servis kesintisi sağlama veya haksız kazanç elde etme gibi zararlı faaliyetlerde bulunabilirler. Bu tür saldırıların sonucunda, hedef alınan kişi, kurum ve kuruluşların itibarı zedelenir. Siber saldırı gerçekleştirilirken izlenen planlı aşamalara **Siber Saldırı Yaşam Döngüsü (Cyber Kill Chain)** denir. Siber Saldırı Yaşam Döngüsü aşamaları şunlardır:

Keşif (Reconnaissance): Saldırganın, hedef sistemde istismar edebileceği güvenlik açıklarını bulmak için gerçekleştirdiği keşif faaliyetleridir.

Silahlanma (Weaponization): Saldırganın, keşif aşamasında belirlenen güvenlik açıklarına yönelik hangi atak vektörünü kullanacağına karar verdiği aşamadır. Zararlı yazılım veya saldırı araçları bu aşamada geliştirilir.

İletme (Delivery): Hazırlanan zararlı yazılımın veya saldırı araçlarının hedef sisteme iletilme aşamasıdır. Saldırgan, zararlı yazılım veya saldırı araçlarını e-posta, USB bellek vb. yöntemleri kullanarak hedef sisteme iletir.

Sömürme (Exploitation): Saldırgan tarafından hedef sisteme iletilen zararlı yazılım kodları çalıştırılarak ve belirlenen atak vektörü kullanılarak zafiyetler sömürülür.

Yükleme (Installation): Zararlı yazılımın kalıcı bir tehdit hâline gelmesi için hedef sisteme kendisini yükleme aşamasıdır.

Komuta Kontrol [Command and Control (C2)]: Hedef sisteme yüklenen zararlı yazılımın, uzaktan kontrol edilerek sistemin ele geçirildiği aşamadır.

Hedefe Yönelik Eylemler (Actions On Objectives): Saldırganın, ele geçirdiği sisteme yönelik verileri elde etme, değiştirme, silme, şifreleme vb. zararlı faaliyetleri gerçekleştirdiği aşamadır.

Görsel 1.9'da siber saldırı türleri verilmiştir.

Görsel 1.9: Siber saldırı türleri

Sosyal Mühendislik Saldırıları (Social Engineering Attacks): Sosyal mühendislik; bilişim sistemlerine, bilgisayar ağlarına saldırmak yerine insanları yanıltarak bilgi toplamak, manipüle etmek, duygularını istismar etmek, zafiyetlerinden faydalanmak vb. amacıyla psikolojik ve sosyal tekniklerin kullanıldığı bir saldırı türüdür. Saldırgan, kurbanlarına kendini yetkili biri gibi tanıtır ve onların güvenini kazanarak kişisel bilgileri, kredi kartı bilgileri gibi önemli bilgileri ele geçirerek haksız kazanç elde etmeye çalışır. Omuz sörfü, oltalama ve kapıdan sızma, yaygın olarak kullanılan sosyal mühendislik saldırılarıdır. Sosyal mühendislik saldırıları, yetkili bir kurumdan aranıyormuş gibi yapılan bir telefon görüşmesi, sahte web sitelerine yönlendirme veya sahte e-posta şeklinde gerçekleştirilebilir. Sosyal mühendislik saldırılarına karşı koymanın en etkili yolu, insanların farkındalığını artırmaktır.

Araştırma

Omuz sörfü saldırısı olarak değerlendirilebilecek durumları araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

Servis Dışı Bırakma Saldırıları (Denial of Service Attack-DoS Attack): İnternet üzerinde bulunan bir sunucunun geçici veya sürekli servis dışı bırakılarak hizmet veremez hâle getirilmesini hedefleyen ve günümüzde çok sık gerçekleştirilen bir saldırı türüdür. Servis dışı bırakma saldırıları çok yüksek bant genişliğine sahip trafikler oluşturularak gerçekleştirilir ve hedef sunucu, gereksiz talepler ile aşırı yüklenerek taleplere cevap veremez hâle getirilir. Sunucu bu durumda meşru taleplere de cevap veremeyerek hizmet dışı kalır. DoS saldırılarının, zararlı yazılımlar aracılığıyla zombi bilgisayarlar hâline getirilerek çok sayıda bilgisayarın kullanılmasıyla gerçekleştirilen hâline **(Distributed Denial of Service - Dağıtık Servis Dışı Bırakma) saldırısı** denir.

Kaba Kuvvet Saldırıları (Brute Force Attacks): Bir kullanıcıya ait, olası kullanıcı adı ve şifre kombinasyonlarını deneyerek bulup ele geçirmeye yönelik saldırı türüdür. Bu saldırı genellikle basit, zayıf şifre ve parolaları hedef alır.

Kaba kuvvet saldırılarına karşı alınabilecek önlemlerden bazıları şunlardır:

- Güçlü parola ve şifre kullanmak (En az 12 karakter, büyük-küçük harf, sayı ve özel karakterden oluşacak şekilde şifre ve parola belirlemek.).
- İki faktörlü kimlik doğrulama kullanmak.
- Belirli bir sayıdan fazla başarısız deneme sonucunda hesaba erişimi otomatik olarak durdurmak.
- Belirli bir sayıdan fazla başarısız deneme yapılan kaynak IP adresini filtrelemek.

Ortadaki Adam Saldırısı (Man In The Middle Attack): Aynı ağ içinde haberleşen iki bilgisayarın arasına girilerek ağ trafiğinin saldırgan bilgisayar üzerinden gerçekleştirildiği saldırı türüdür. Bu saldırı türünde saldırgan, iki bilgisayar arasındaki ağ trafiğini dinleyebilir. Gerekli güvenlik önlemleri alınmadığında çoğu zaman saldırıya uğrayan bilgisayarlar, saldırıya maruz kaldıklarını anlayamaz. Sahte erişim noktası, DHCP sahtekârlığı, DNS sahtekârlığı ve ARP sahtekârlığı en yaygın ortadaki adam saldırılarıdır.

1. Uygulama

İşlem adımlarına göre **bettercap** aracını kullanarak ARP önbellek zehirlenmesi atağını gerçekleştiriniz.

1. Adım: Görsel 1.10'da görüldüğü gibi bettercap aracını yükleyiniz.

```
┌──(kali㉿kali)-[~]
└─$ sudo apt-get install bettercap
```

Görsel 1.10: Kali işletim sisteminde bettercap aracının yüklenmesi

2. Adım: Görsel 1.11'de görüldüğü gibi bettercap aracının çalıştırılmasını gerçekleştiriniz.

```
┌──(kali㉿kali)-[~]
└─$ sudo su
[sudo] password for kali:
┌──(root㉿kali)-[/home/kali]
└─# bettercap -iface eth0
bettercap v2.32.0 (built for linux amd64 with go1.18) [type 'help' for a list of commands]

192.168.1.0/24 > 192.168.1.59 » [08:53:49] [sys.log] [war] Could not find mac for 192.168.1.1
192.168.1.0/24 > 192.168.1.59 »
```

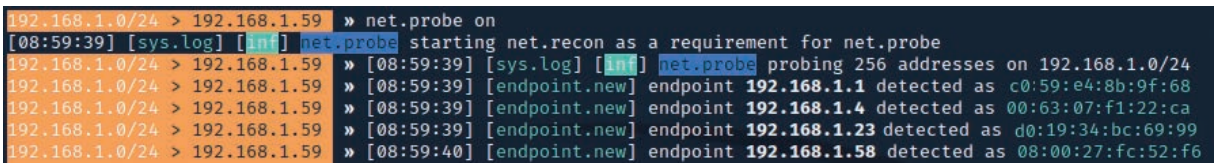
Görsel 1.11: Kali işletim sisteminde bettercap aracının çalıştırılması

3. Adım: Çalışan, çalışmayan servisleri ve servislerle birlikte kullanılacak parametreleri görüntülemek için **help** komutunu kullanınız (Görsel 1.12).

```
192.168.1.0/24 > 192.168.1.59 » help

          help MODULE : List available commands or show module specific help if no module name is provided.
               active : Show information about active modules.
                 quit : Close the session and exit.
        sleep SECONDS : Sleep for the given amount of seconds.
             get NAME : Get the value of variable NAME, use * alone for all, or NAME* as a wildcard.
       set NAME VALUE : Set the VALUE of variable NAME.
  read VARIABLE PROMPT : Show a PROMPT to ask the user for input that will be saved inside VARIABLE.
                clear : Clear the screen.
       include CAPLET : Load and run this caplet in the current session.
            ! COMMAND : Execute a shell command and print its output.
       alias MAC NAME : Assign an alias to a given endpoint given its MAC address.

Modules

     any.proxy > not running
      api.rest > not running
     arp.spoof > not running
            c2 > not running
       caplets > not running
    dhcp6.spoof > not running
      dns.spoof > not running
  events.stream > running
           hid > not running
    http.proxy > not running
   http.server > not running
   https.proxy > not running
  https.server > not running
   mac.changer > not running
   mdns.server > not running
  mysql.server > not running
     ndp.spoof > not running
     net.probe > not running
     net.recon > not running
     net.sniff > not running
  packet.proxy > not running
      syn.scan > not running
     tcp.proxy > not running
```

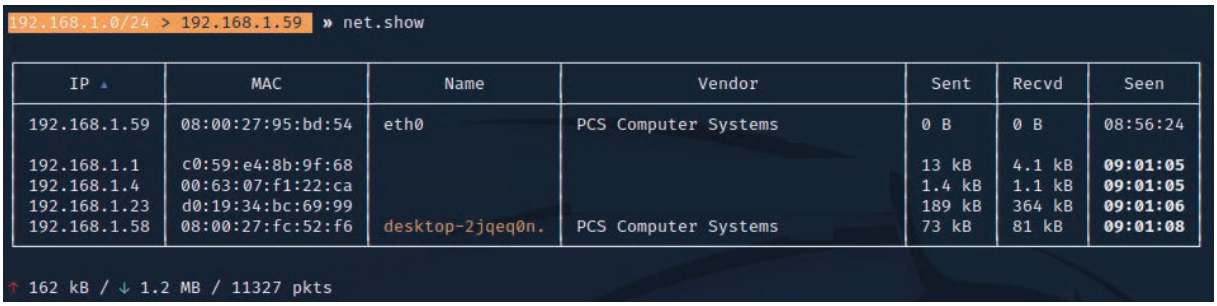
Görsel 1.12: Çalışan ve çalışmayan servislerin görüntülenmesi için help komutunun kullanımı

4. Adım: Aynı ağ içindeki cihazların bulunmasını ve IP-MAC adreslerinin eşleştirilmesini **net.probe on** komutuyla sağlayınız (Görsel 1.13).



Görsel 1.13: Aynı ağ içindeki cihazların net.probe on komutuyla taranması

5. Adım: **net.probe** servisinin çalışıp çalışmadığını **net.show** komutu ile kontrol ediniz (Görsel 1.14).

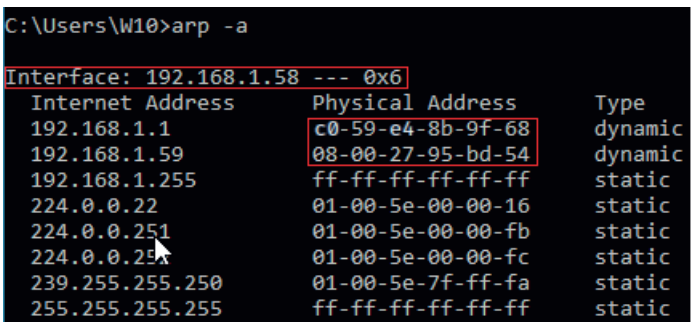


Görsel 1.14: net.probe servisinin net.show komutuyla kontrol edilmesi

6. Adım: ARP önbellek zehirlenmesi işlemini gerçekleştirecek komutları çalıştırınız (Görsel 1.15).

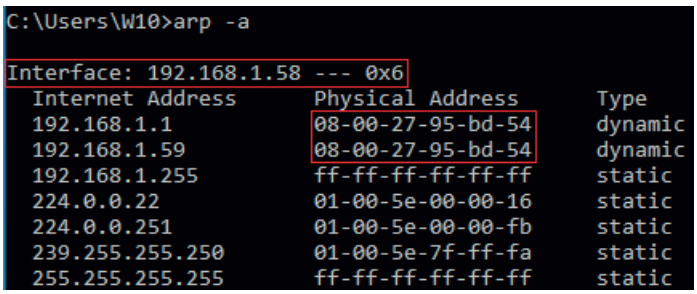
Görsel 1.15: ARP önbellek zehirlenmesinin gerçekleştirilmesi

Birinci uygulama sonucunda hedef bilgisayarın atak gerçekleştirilmeden önceki ARP tablosu Görsel 1.16'da verilmiştir.



Görsel 1.16: Hedef bilgisayarın atak gerçekleştirilmeden önceki ARP tablosu

Hedef bilgisayarın atak gerçekleştirildikten sonra oluşan ARP tablosu ise Görsel 1.17'de verilmiştir. Saldırı sonucunda 192.168.1.1 IP adresine sahip ağ geçidinin MAC adresi, 192.168.1.59 IP adresine sahip saldırgan bilgisayarın MAC adresi ile değiştirilmiştir. Bunun sonucu olarak saldırgan, hedef bilgisayarın ağ trafiğini dinleyebilir.



Görsel 1.17: Saldırı sonrası hedef bilgisayarın ARP tablosu

Sıra Sizde

Laboratuvardaki Kali işletim sistemi kurulu sanal makineye bettercap aracını yükleyiniz. LAN içindeki IP-MAC adresi eşleştirmelerini keşfediniz. Kendinize bir hedef IP adresi seçerek ARP önbellek zehirlemesi atağını gerçekleştiriniz.

SQL Enjeksiyon Saldırıları (SQL Injection Attacks): Web uygulamalarındaki güvenlik açıkları kullanılarak gerçekleştirilen bir saldırı türüdür. Saldırgan, SQL sorgularına müdahale ederek hedef veri tabanındaki SQL kodlarını değiştirebilir. Bu tür saldırılar sonucunda saldırgan, veri tabanı yöneticisinin şifre ve parolalarını ele geçirebilir, veri tabanındaki verilere erişip onları çalabilir, yok edebilir ve değiştirebilir.

Sıfırıncı Gün Saldırıları (Zero Day Attacks): Güvenlik açığı bulunan yazılımların, sistemlerin, henüz açıklarının üreticisi tarafından fark edilmediği ve güvenlik açığının ortadan kaldırılmadığı durumlar, **sıfırıncı gün açığı** olarak tanımlanır. Bu açıklıktan yararlanılarak gerçekleştirilen siber saldırılara ise **sıfırıncı gün saldırısı** denir. Sıfırıncı gün saldırısına karşı alınabilecek önlem, güvenlik açığı fark edildiğinde çok hızlı bir şekilde **yamalama** işleminin gerçekleştirilmesidir.

Gelişmiş Sürekli Tehdit (Advanced Persistent Threat): Saldırganların bir ağa gizlice sızarak ve uzun bir süre fark edilmeden kalarak verilerin ele geçirilmesini hedefleyen, karmaşık, sofistike ve sürekli saldırılardır. APT saldırıları, geleneksel siber saldırılardan farklıdır. Genellikle birbirine düşman devletler veya organizasyonlar tarafından gizlice yürütülür. Çok yüksek düzeyde uzmanlık ve kaynak gerektirir. Stuxnet, APT saldırılarına bir örnektir.

Zararlı Yazılım Saldırıları (Malware Attack): Kötü niyetli amaçlar için geliştirilen zararlı yazılımların, bilgisayar ağlarına, bilgisayarlara bulaştırılması şeklinde gerçekleştirilen saldırı türüdür.

Zararlı yazılım saldırılarına karşı alınabilecek önlemlerden bazıları şunlardır:

- İşletim sisteminin lisanslı ve güncel olması
- Güncel bir virüs tarama programının kullanılması
- Güvenilir olmayan web sitelerinden dosya indirilmemesi
- Kaynağı bilinmeyen e-posta ve eklentilerinin açılmaması
- Taşınabilir ve USB belleklerin kullanılmadan önce virüs taramasından geçirilmesi

1.2.6. Zararlı Yazılım Türleri

Zararlı yazılımlar; kötü amaçlarla oluşturulmuş ve genellikle kullanıcıların cihazlarına veya sistemlerine yetkisiz erişim sağlamak, verilerini ele geçirmek, zarar vermek için tasarlanmıştır. Sayısı her geçen gün artan zararlı yazılımlar, sistemlere ciddi zararlar verebilir. Zararlı yazılım türleri Görsel 1.18'de verilmiştir.

Görsel 1.18: Zararlı yazılım türleri

Virüs: Bir programa, belgeye veya yürütülebilir dosyaya kendi kopyasını ekleyerek yayılan kötü amaçlı bir yazılımdır. Virüsler kolaylıkla bir bilgisayardan başka bir bilgisayara bulaşabilir. Kullanıcı etkileşimi ile içerdikleri zararlı kodlar çalışır. Virüslerin etkileri, barındırdığı zararlı kodlara bağlı olarak farklılık gösterebilir. Bu zararlı kodlar; kişisel verilerin ele geçirilmesi, değiştirilmesi, silinmesi veya işletim sisteminin işlevsiz hâle gelmesi gibi zararlı eylemleri gerçekleştirebilir. Virüs çoğunlukla USB bellekler, web sayfası ve e-posta eklentileri ile bulaşır. Kullanıcılar, bu yollarla bilgisayarlarına zararlı yazılımların bulaşmasını önlemek için dikkatli olmalı ve güvenilir kaynaklardan yazılım veya dosya indirmelidir. Güncel antivirüs yazılımının kullanılması, virüslere karşı önemli bir savunma sağlar.

Solucan (Worm): Bir ağ içindeki bilgisayarlara bulaştığında kendi kendine yayılan zararlı bir yazılımdır. Solucan, kullanıcı etkileşimi ile çalıştırılması gerekmediği için virüsten farklıdır. Bir bilgisayara bulaştıktan sonra sistemdeki diğer cihazlara otomatik yayılır. Bu şekilde hızla birçok bilgisayarı etkileyebilir. Solucanlar; işletim sistemlerinde olabilecek zafiyetler hedeflenerek e-posta eklentileri, USB bellekler, FTP, HTTP ve P2P ağ paylaşımları ile sistemlere bulaşabilir. Solucan, ağın ve işletim sistemlerinin çalışmasını olumsuz yönde etkiler.

Araştırma

Stuxnet solucanı ile ilgili araştırma yapınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

Fidye Yazılımı (Ransomware): Kullanıcının bilgisayarındaki bütün sistemi ve dosyaları şifreleyerek sisteme erişimi engelleyen ve erişim engelinin kaldırılması için kullanıcıdan fidye talep eden bir zararlı yazılımdır. Fidye yazılımı genellikle e-posta eklentileriyle ve işletim sistemlerindeki oluşabilecek zafiyetlerden faydalanılarak kullanıcılara bulaştırılır. Sistem ve dosyaların şifrelerinin çözülebilmesi için özel bir şifre kullanılır. Bu şifreyi iletmek için saldırgan, kurbandan **kripto para** olarak fidye talep eder. Kripto para talebi, saldırganın gizliliğini sağlamak için tercih ettiği bir yöntemdir. Fidye ödemesi yapıldığında saldırgandan alınan şifre ile sistem ve dosyaların şifreleri açılabilir. Fidye ödemesinin yapılması, şifreyi elde etmeyi ve elde edilen şifrenin verilere erişimi sağlayacağını garanti etmez. Fidye yazılımı, saldırganların en çok haksız kazanç elde ettikleri saldırı yöntemidir. Fidye yazılımı saldırısından korunmak için düzenli olarak işletim sistemlerinin güncellemesi ve verilerin yedeğinin alınması çok önemlidir.

Araştırma

WannaCry fidye yazılımı hakkında araştırma yapınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

Arka Kapı (Backdoor): Saldırganlara bir bilgisayar sistemine izinsiz ve gizli erişim sağlama olanağı sağlayan bir yöntemdir. Bu yöntem, kimlik doğrulamasını atlamak suretiyle sisteme giriş yapar ve saldırganlara uzaktan erişim sağlar. Arka kapılar, Truva atları ile oluşturulabilir. Yazılımlarda XSS ve SQL enjeksiyon saldırılarına uygun zafiyetlerin bulunması, sistemde açılan yeni port üzerinden veri çıkışlarının olması arka kapı şeklinde kullanılabilir.

Tuş Kaydedici (Keylogger): Kullanıcının klavyeden bastığı her tuşu yakalayarak kaydeden ve bu kayıtları saldırganlara gönderebilen bir zararlı yazılımdır. Bu yazılımların hedefi, kullanıcılara ait kişisel verileri, kredi kartı bilgilerini, parolaları ve diğer önemli bilgileri elde etmektir. Tuş kaydedici yazılımlar; güvenilir olmayan web sitelerinden indirilen dosyalar, e-posta eklentileri, USB bellekler ve ağ paylaşımları ile bulaşabilir.

Botnet Yazılımları: Saldırganların birçok bilgisayarı uzaktan kontrol etmesini sağlayan zararlı yazılımlardır. Botnetler genellikle dağıtılmış servis dışı bırakma saldırılarında (DDoS) kullanılır.

Korku Yazılımı (Scareware): Kullanıcılarda korku ve endişe oluşturmak için kullanılan zararlı yazılımdır. Sahte güvenlik uyarıları ile tehlikeli bir durumun içinde olunduğu izlenimi oluşturularak kullanıcılar yanıltılır. Sahte çözümler sunularak veya ücretli hizmetlerin satın alınması teşvik edilerek kazanç sağlanması hedeflenir.

Truva Atı (Trojan): Adını Antik Yunan efsanesindeki Truva atından alan bu zararlı yazılım genellikle zararsız bir dosya gibi görünür. Çoğunlukla güvenli olmayan web sitelerinin açılması ve kaynağı bilinmeyen e-posta eklentilerinin bilgisayara indirilip çalıştırılması ile bulaşır. Truva atı, bir bilgisayar sistemine uzaktan erişim sağlamak amacıyla kullanılır. Uzaktan erişim sağlayan saldırgan; kullanıcıya ait kişisel bilgileri, kredi kartı bilgilerini, kullanıcı adı ve şifreleri gibi birçok önemli veriyi elde edebilecek duruma gelir.

Reklam Yazılımı (Adware): Son kullanıcı cihazında ziyaret edilen web sayfalarını izleyerek kullanıcıya uygun reklamları göstermek amacıyla tasarlanmıştır. Web tarayıcı çalıştırıldığında açılır pencereler

şeklinde görüntülenen reklamlar, kullanıcı deneyimini olumsuz etkiler. Reklamlar, kullanıcıların dikkatini çekmeye çalışarak reklamlara tıklamasını, belirli ürünleri ve web sitelerini ziyaret etmesini sağlar. Kullanıcı bu reklamlara tıkladığında zararlı yazılım geliştiricisi maddi kazanç elde edebilir.

Casus Yazılımı (Spyware): Kullanıcıların bilgilerini izinsiz bir şekilde toplayan yazılımlardır. Casus yazılımlar; sistem üzerinde gizli kalarak, kullanıcıların ziyaret ettiği web sayfalarını izleyebilir, kullanıcı adı ve şifreleri, kredi kartı bilgilerini vb. elde edip kötü niyetli kişilere gönderebilir.

Kök Kullanıcı Takımı (Rootkit): Saldırganlara bir sisteme uzaktan erişim ve kontrol izni vermek için kullanılan zararlı yazılımlardır. Rootkitler, işletim sisteminin çekirdeğinde gizlenir. Zararlı yazılım tespit yazılımları ile rootkitlerin tespit edilmesi zordur. Gizli kaldıkları sürece saldırganlar, sistemlere erişim sağlayarak zarar verebilirler.

Araştırma

Zombi ve Command And Controller kavramlarını araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

1.3. FİZİKSEL GÜVENLİK

Fiziksel güvenlik; bilişim sistemleri, ağlar, donanımlar ve verilerin bulunduğu fiziksel ortamların, fiziksel tehditlere ve saldırılara karşı korunması için alınan önlemleri ifade eder. Siber güvenlik farkındalığının oluştuğu ve önlemlerin alınmaya çalışıldığı günümüzde fiziksel güvenlik zaman zaman göz ardı edilir. Fiziksel güvenlik önlemleri; bilişim altyapısının, ağ cihazlarının, ağ kablolarının ve bilgisayarların hırsızlık, sabotaj, doğal afetler gibi fiziksel tehditlere karşı korunmasını sağlar.

Bilişim sistemlerinin fiziksel güvenliğini sağlamak için alınacak tedbirlerden bazıları şunlardır:

- Bina girişinde ve bina içinde fiziksel her ortama erişim için yetkilendirme yapılmalıdır.
  - Kartlı sistemlerle yetkisiz erişim engelleme
  - Biyometrik verilerle yetkisiz erişim engelleme
- Sunucu, ağ cihazları gibi bilişim altyapısıyla ilgili cihazlar bir sistem odasında bulunmalıdır.
- Sistem odaları fiziksel olarak kilitli tutulmalı ve yetkisiz erişim engellenmelidir.
- Güvenlik kamerası sistemleri ile fiziksel güvenlik kontrol edilmelidir.
- Yangın söndürme sistemleri kullanılarak olası bir yangında bilişim altyapısının zarar görmemesi sağlanmalıdır.

Araştırma

Fiziksel güvenlik ihlali olabilecek durumlarla ilgili örnek olay araştırması yapınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

1.4. BİLGİ GÜVENLİĞİ STANDARTLARI

Bilgi güvenliği standartları; kurum ve kuruluşların bilgi varlıklarını korumak, güvenlik risklerini yönetmek ve bilgi güvenliği politikalarını uygulamak için kabul edilen kanun ve standartlardan oluşur. Bunlar, veri güvenliği ve bilginin korunması konusunda etkin ve uygun güvenlik önlemleri alınmasına katkı sağlar.

1.4.1. 6698 Sayılı Kişisel Verilerin Korunması Kanunu

İletişim teknolojilerinin gelişmesi ve internet kullanımının her geçen gün artması ile kullanıcılar alışveriş yapmak, oyun oynamak, iş başvurusunda bulunmak, bankacılık işlemleri yapmak, sosyal ağ platformlarında bulunmak gibi çok çeşitli amaçlarla interneti kullanır. Bu gelişmelerin sonucu olarak kişisel veriler, kolayca paylaşılabilir ve geniş kitlelere yayılabilir hâle gelmiştir. Veri güvenliğinin çok önemli olduğu günümüzde kişisel verilerin korunması bir zorunluluktur.

Dünyada ve ülkemizde kişisel verilerin korunması ile ilgili mevzuat çalışmaları yapılmıştır ve ihtiyaca göre bu çalışmalar devam ettirilmiştir. 1948 yılında imzalanan İnsan Hakları Evrensel Beyannamesi ve 1950 yılında imzalanan Avrupa İnsan Hakları Sözleşmesi, kişisel verilerin korunması ile ilgili uluslararası sözleşmelerin ilk örneklerini oluşturur. Ülkemizde de kişisel verilerin korunması amacıyla çıkartılan 6698 sayılı Kişisel Verilerin Korunması Kanunu, 7 Nisan 2016 tarihinde yürürlüğe girmiştir.

Kişisel Verilerin Korunması Kanunu'nun amacı, "kişisel verilerin işlenmesinde başta özel hayatın gizliliği olmak üzere kişilerin temel hak ve özgürlüklerini korumak ve kişisel verileri işleyen gerçek ve tüzel kişilerin yükümlülükleri ile uyacakları usul ve esasları düzenlemektir." şeklinde belirtilir. Kişisel Verilerin Korunması Kanunu'nun yüklediği görevleri yerine getirmek üzere **Kişisel Verileri Koruma Kurumu** kurulmuştur. Kişisel Verileri Koruma Kurumu, KVKK olarak da bilinir (Görsel 1.19).

KVKK
KİŞİSEL VERİLERİ KORUMA KURUMU

Görsel 1.19: Kişisel Verileri Koruma Kurumu

6698 sayılı Kişisel Verilerin Korunması Kanunu'nda bulunan bazı tanımlar;

- **Kişisel veri:** Kimliği belirli veya belirlenebilir gerçek kişiye ilişkin her türlü bilgiyi,
- **Açık rıza:** Belirli bir konuya ilişkin, bilgilendirmeye dayanan ve özgür iradeyle açıklanan rızayı,
- **İlgili kişi:** Kişisel verisi işlenen gerçek kişiyi,
- **Veri işleyen:** Veri sorumlusunun verdiği yetkiye dayanarak onun adına kişisel verileri işleyen gerçek veya tüzel kişiyi,
- **Veri sorumlusu:** Kişisel verilerin işleme amaçlarını ve vasıtalarını belirleyen, veri kayıt sisteminin kurulmasından ve yönetilmesinden sorumlu olan gerçek veya tüzel kişiyi,
- **Kişisel verilerin işlenmesi:** Kişisel verilerin tamamen veya kısmen otomatik olan ya da herhangi bir veri kayıt sisteminin parçası olmak kaydıyla otomatik olmayan yollarla elde edilmesi, kaydedilmesi,

depolanması, muhafaza edilmesi, değiştirilmesi, yeniden düzenlenmesi, açıklanması, aktarılması, devralınması, elde edilebilir hâle getirilmesi, sınıflandırılması ya da kullanılmasının engellenmesi gibi veriler üzerinde gerçekleştirilen her türlü işlemi ifade eder.

6698 sayılı Kişisel Verilerin Korunması Kanunu'na göre kişisel verilerin işlenmesi sürecinde uygulanan hususlardan bazıları şunlardır:

- Kişisel verilerin işlenme şartları
- Özel nitelikli verilerin işlenme şartları
- Kişisel verilerin silinmesi, yok edilmesi veya anonim hâle getirilmesi
- Kişisel verilerin aktarılması
- Kişisel verilerin yurt dışına aktarılması

Araştırma

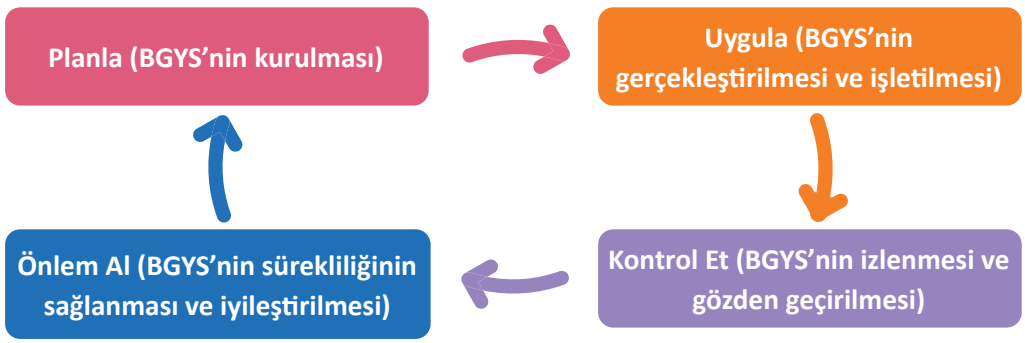
Kişisel verilerin korunmasında ortaya çıkacak suç ve kabahatleri www.kvkk.gov.tr web sitesine girerek 6698 sayılı Kişisel Verilerin Korunması Kanunu'nda araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

1.4.2. ISO 27001 Bilgi Güvenliği Yönetim Sistemi

Bilgi, verilerin anlam ve değer kazanmış hâli olarak tanımlanır. Bilginin güvenliği için gizlilik, bütünlük ve erişilebilirlik unsurlarının sağlanması hayati öneme sahiptir. Bu nedenle kurum ve kuruluşlar bilgi ve bilgi varlıklarını korumak için daha çok kaynak ayırmak ve çaba harcamak zorundadır. Bu çabalara destek olmak, bilgi güvenliğini bir plan ve program çerçevesinde sağlamak için uluslararası ISO 27001 Bilgi Güvenliği Yönetim Sistemi standart hâline gelmiştir. ISO 27001 standardının son güncellemesi 2022 yılında yayımlanmıştır.

ISO 27001 Bilgi Güvenliği Yönetim Sistemi (BGYS); kurum ve kuruluşların bilgi güvenliği politikalarını, prosedürlerini, süreçlerini belirlemelerine, uygulamalarına ve sürekli olarak iyileştirmelerine rehberlik ederek bilginin ve bilgi varlıklarının yönetilmesini ve korunmasını sağlar.

Bilgi Güvenliği Yönetim Sistemi, bilgi üzerindeki riskleri ortadan kaldırmayı veya etkilerini azaltmayı temel alan bir yönetimsel süreçtir. Bu süreçte Görsel 1.20'deki gibi Planla-Uygula-Kontrol Et-Önlem Al (PUKÖ) döngüsü kabul edilmiş ve benimsenmiştir.



Görsel 1.20: PUKÖ döngüsü

- **Planla:** Bilgi güvenliği ve yönetim sistemi politikası, amaçları, hedefleri, prosedürleri ve süreci planla aşamasında belirlenir.
- **Uygula:** Bilgi güvenliği ve yönetim sistemi politikası, kontrolleri, süreci uygulama aşamasında gerçekleştirilir ve işletilir.
- **Kontrol Et:** BGYS politikası, amaçlar ve süreç performansının değerlendirilmesi, uygulanabilen yerlerde ölçülmesi ve sonuçların rapor edilmesidir.
- **Önlem Al:** Yönetimin gözden geçirme sonuçlarına dayalı olarak düzeltici ve önleyici faaliyetlerin gerçekleştirilmesidir.

ISO 27001 standardının maddeleri şunlardır:

1. Giriş
2. Kapsam
3. Atıf yapılan standartlar ve/veya dokümanlar
4. Terimler ve tarifler
5. Kuruluşun içeriği
6. Liderlik
7. Planlama
8. Destek
9. İşletim
10. Performans değerlendirme
11. İyileştirme

Ek A – Referans kontrol hedefleri ve kontroller

ISO 27001 Bilgi Güvenliği Yönetim Sistemi, bir kurumun bilgi güvenliği sürecini daha etkin, güvenilir hâle getirir. Bu durum, kurumun güvenilirliğini artırır ve rekabet avantajı sağlar.

Araştırma

ISO 27001 Bilgi Güvenliği Yönetim Sistemi sertifikasını hangi kurum ve kuruluşların alabileceğini araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

1.4.3. ISO 27002 Bilgi Güvenliği Kontrolleri

ISO 27002, ISO 27000 ailesinin üyesi olan uluslararası bir standarttır ve bilgi güvenliği yönetimi için bilgi güvenliği kontrollerini ve en iyi uygulamaları içerir. ISO 27002 standardı, ISO 27001 standardıyla birlikte kullanılır. ISO 27001 standardında, Ek A-Referans Kontrol Hedefleri ve Kontroller bölümünde belirtilen güvenlik kontrollerinin sadece listesi paylaşılmıştır fakat ISO 27002, güvenlik kontrollerinin listesiyle birlikte bu güvenlik kontrollerinin nasıl uygulanacağını da belirterek kapsamlı bir rehber sunar. ISO 27001 standardı, kurum ve kuruluşlara bilgi güvenliği yönetim sisteminin olmazsa olmaz şartlarını belirtirken ISO 27002, bilgi güvenliği yönetim sistemi kapsamında uygulanabilecek bilgi güvenliği kontrollerini ve bunların nasıl uygulanabileceğini açıklayan bir rehber niteliği taşır.

2022 yılında güncellenen ISO 27002 standardındaki kontroller, dört başlık altında doksan üç kontrolden oluşur. Görsel 1.21'de ISO 27002 standardının gruplara ayrılmış hâli ve bazı kontroller verilmiştir.

Görsel 1.21: ISO 27002 standardının gruplara ayrılmış kontrol sayıları ve bazı kontroller

Araştırma

ISO 27002 standardında 2022 yılındaki güncellemeyle birlikte gelen yenilikleri araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

1.4.4. ISO 27032 Siber Güvenlik ve İnternet Güvenliği Yönergeleri

ISO 27032, ISO 27000 ailesinin üyesi olan uluslararası bir standarttır. ISO 27001 ile birlikte kullanılan bu standart, kurum ve kuruluşların siber güvenliğini iyileştirmek için rehberlik eder. ISO 27032 standardı; bilgi güvenliği, ağ güvenliği, internet güvenliği ve kritik bilgi altyapılarının korunması konularına odaklanır.

1.5. DİJİTAL ARŞİVLEME

Belgelerin fiziksel nüshalarının raf, dolap gibi fiziksel ortamlarda düzenli bir şekilde saklanması işlemine **fiziksel arşivleme** denir. Belgelerin tarayıcılar veya optik karakter tanıma (Optical Character Recognition-OCR) teknolojileriyle dijital ortamlara aktarılması ve .jpeg, .gif, .tif, .pdf gibi değişik dosya formatlarında belirli algoritmalara göre saklanması sürecine ise **dijital arşivleme** denir.

Dijital arşivleme; bilişim sistemleri, sunucular, bulut tabanlı depolama alanları gibi dijital arşivleme altyapıları kullanılarak gerçekleştirilir. Dijital arşivleme çözümleri, bilgi teknolojilerinin hızla gelişmesiyle birlikte kurum ve kuruluşlar tarafından daha yaygın olarak kullanılmaya başlanmıştır.

Görsel 1.22'de dijital arşivleme aşamaları verilmiştir.

Görsel 1.22: Dijital arşivleme aşamaları

1. **Belgelerin Tasnif Edilmesi:** Belgelerin belirli kriterlere göre düzenlendiği ve sınıflandırıldığı aşamadır. Belgeler; türüne, içeriğine veya diğer özelliklere göre kategorilere ayrılır. Kategorilere ayırma işlemi, belgelerin daha sonra bulunmasını kolaylaştırır ve belgelere erişilmesini sağlar.
2. **Belgelerin Taranması:** Fiziksel belgelerin tarayıcılar veya OCR teknolojileri kullanılarak dijital formata dönüştürüldüğü aşamadır.
3. **Görüntü İyileştirme Uygulaması:** Tarama işlemi sonrasında elde edilen dijital görüntüler, kalite düşüklüğü veya görüntü bozuklukları içerebilir. Bu aşamada, görüntü iyileştirme teknikleri kullanılarak görüntüdeki gürültüler giderilir ve görüntü kalitesi artırılır.
4. **İndeks Oluşturulması:** Dijital arşivdeki belgelere hızlı ve etkili bir şekilde erişmek için indeks oluşturulur. İndeksleme işlemi, belge numarası ve türü gibi kriterlere göre yapılabilir veya belgelerin metadata bilgileri ve anahtar kelimelerle belge içeriği tanımlanarak da gerçekleştirilebilir.
5. **Aktarma İşlemi:** Tarama, görüntü iyileştirme ve indeksleme işlemleri tamamlanan belgelerin dijital arşivleme veri tabanına kaydedilmesi ve dijital arşivleme sürecinin tamamlanması aşamasıdır.

Dijital arşivlemenin avantajları şunlardır:

- Belgeler sunucular üzerinde saklandığı için fiziksel alan ihtiyacı daha azdır.
- Belgelere erişim hızlı ve kolaydır.
- Belgeler dijital olarak saklandığı için fiziksel olarak yıpranmaz.
- Dijital belgelere çoklu erişimin sağlanması mümkündür.
- Teknoloji destekli olduğu için sürdürülebilir ve yönetilebilirdir.
- Dijital arşivdeki belgelere yetkilendirme temelli erişim sağlanabilir.
- Kurum ve kuruluşların katma değerini artırır.

Dijital arşivlemenin dezavantajları şunlardır:

- Siber saldırılara karşı uygun güvenlik önlemleri alınmazsa güvenlik riskleri ortaya çıkabilir.
- Herhangi bir veri kaybı durumunda veri kurtarma süreci karmaşık olabilir.
- Düzenli olarak teknolojik altyapı yatırımı gerekebilir.

Sıra Sizde

Dijital arşivleme çözümlerini anlatan bir sunu hazırlayınız. Hazırladığınız sunuyu sınıfta arkadaşlarınızla paylaşınız.

1.6. ADLİ BİLİŞİM KAVRAMLARI

Bilişim alanındaki gelişmelerle birlikte internetin kullanımı artmış ve bu alanda suç olarak tanımlanan çeşitli unsurlar ortaya çıkmaya başlamıştır. Bu unsurlarla mücadele etmek için çeşitli yasal düzenlemeler yapılmıştır. Yasal düzenlemelerle birlikte **adli bilişim (dijital forensic)** kavramı hayata girmiştir.

Bilişim alanında işlenen suçlarla ilgili dijital ortamlarda bulunan ve bu ortamlar aracılığıyla iletilen ses, görüntü, veri, bilgi veya her türlü bilişim verisinin mahkemede sayısal delil niteliği taşıyacak şekilde elde edilmesi, muhafaza edilmesi, çeşitli yazılım ve donanımlarla analiz edilmesi ve rapor oluşturularak mahkemeye sunulması çalışmalarına **adli bilişim** denir.

Ülkemizde adli bilişim sürecinde iş ve işlemler; 5271 sayılı Ceza Muhakemesi Kanunu'nun (CMK) 134. maddesine, Adlî ve Önleme Aramaları Yönetmeliği'nin 17. maddesine ve Suç Eşyası Yönetmeliği'nin 8. maddesine göre yürütülür. Ayrıca bilişim alanında işlenen suçlarla ilgili ceza düzenlemeleri, 5237 sayılı Türk Ceza Kanunu'nun 243, 244, 245, 245/A ve 246. maddelerinde yer alır. Türk Ceza Kanunu'na göre bilişim alanında işlenen suçlar şunlardır:

- Bilişim sistemine girme
- Sistemi engelleme, bozma, verileri yok etme veya değiştirme
- Banka veya kredi kartlarının kötüye kullanılması
- Yasak cihaz veya programlar

Araştırma

Ceza Muhakemesi Kanunu'nun 134. maddesini araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

1.6.1. Adli Bilişimde Delil

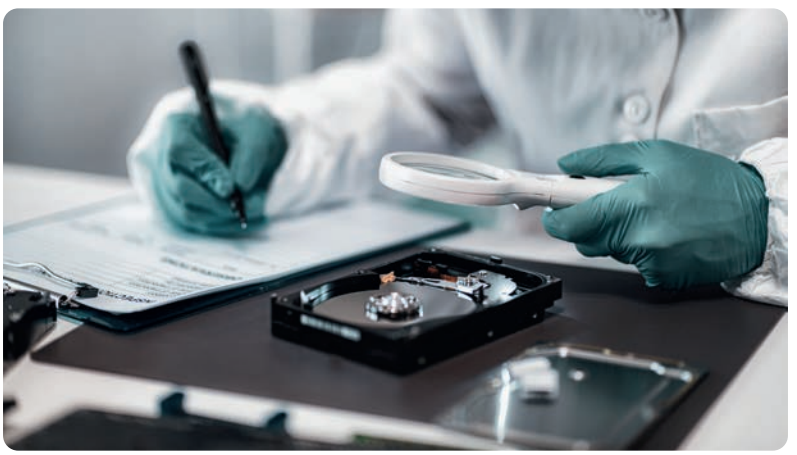
Bir iddianın doğruluğunu veya yanlışlığını kanıtlamak için sunulan kanıt veya kanıtlara **delil** denir. Delil, bir davanın mahkemede çözülmesine yardımcı olmak için sunulan bilgi ve belgelerden oluşur. Adli bilişimde delil, dijital ortamlardan elde edilir. Dijital ortamlardan elde edilen delillere **dijital delil** denir. Görsel 1.23'te dijital delil elde edilebilecek bir sabit disk sürücüsü verilmiştir.

Görsel 1.23: Dijital delil elde edilebilecek bir sabit disk sürücüsü

Dijital delillerin bulunabileceği ortamlar şunlardır:

- Elektronik devreler
- Cep telefonları
- Bilgisayarlar
- Tabletler ve PDA cihazları
- SIM kartlar
- USB bellekler
- Dâhilî ve haricî sabit diskler
- CD ve DVD'ler
- Ses ve kamera kayıt cihazları

Görsel 1.24'te adli bilişim uzmanı tarafından yapılan dijital delil incelemesi verilmiştir.



Görsel 1.24: Dijital delil incelemesi

Dijital delillerin toplanmasında ve muhafazasında dikkat edilecek hususlardan bazıları şunlardır:

- Kişiler, bilgisayar vb. elektronik cihazların başından uzaklaştırılmalıdır.
- Olay yerinin fotoğrafları çekilmelidir.
- Olay yerinde bilgi niteliği taşıyabilecek belgeler toplanmalıdır.
- Açık olan bilgisayarlarda RAM bellek üzerindeki bilgiler kaydedilmeli ve bilgisayar kapatılmalıdır.
- Bilgisayarların elektrik bağlantısı kesilmelidir.
- Bilgisayarların ağ kabloları çıkartılmalıdır.
- Bilgisayarlardaki disklerin imajları alınmalıdır.
- Bütünlüğünün bozulması ihtimali düşünülerek veriler özetleme (hash) fonksiyonuna tabi tutulmalı ve verilerin orijinal hâllerinin özetleme kodları elde edilmelidir.
- Deliller numaralandırılmalıdır.
- Manyetik deliller antistatik ambalajlarla muhafaza edilmelidir.
- CD, DVD gibi araçlar kırılabileceği için eğilip bükülmemelidir.
- Deliller uygun ambalajlarla paketlenmeli ve taşınmalıdır.

Araştırma

RAM bellek üzerindeki bilgilerin kopyasını almak için kullanılan yazılımları araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

1.6.2. Adli Bilişimde Delil Toplama ve Karartma Teknikleri

Adli bilişim sürecinde delil elde etmek için yararlanılan tekniklerden bazıları şunlardır:

- Bilgisayarların disklerinin imajının alınması ve incelenmesi
- RAM belleğin imajının alınması ve incelenmesi
- Silinen dosyaların geri kurtarılması
- Metadata analizinin gerçekleştirilmesi
- Belgelerin ve dosyaların incelenmesi
- Mesajlaşma uygulamalarıyla alınan ve gönderilen mesajların incelenmesi
- Elektronik postaların incelenmesi
- Resim ve videoların incelenmesi
- İnternet tarayıcısı geçmişinin incelenmesi
- Ağ trafiğinin incelenmesi ve trafik oluşturulan IP adreslerinin tespit edilmesi

Suçluların yakalanmamak ve adli bilişim uzmanlarının işlerini zorlaştırmak için gerçekleştirdikleri delil karartma tekniklerinden bazıları şunlardır:

- Delil edilebilmesinin önüne geçmek için verileri geri döndürülemez şekilde silmesi
- Faaliyetlerinin izlerine erişilememesi için RAM belleği temizlemesi
- Delil olabilecek belgelerin zaman damgalarını değiştirmesi
- Delil olabilecek verileri farklı fiziksel ortamlara taşıması
- Diskleri şifreleyerek verilere erişimi engellemesi
- Değiştirerek ve gizleyerek verilere erişimi engellemesi

Araştırma

Adli bilişim incelemesinde kullanılan yazılım ve donanımları araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

2. Uygulama

İşlem adımlarına göre **dd** aracını kullanarak disk imajını almayı ve **bulk_extrcactor** aracı ile imaj incelemeyi gerçekleştiriniz.

1. Adım: Terminal ekranına girdikten sonra yetkili moda geçiş yapınız (Görsel 1.25).

```
┌──(kali㉿kali)-[~]
└─$ sudo su
[sudo] password for kali:
┌──(root㉿kali)-[/home/kali]
└─#
```

Görsel 1.25: Yetkili moda geçiş yapılması

2. Adım: Görsel 1.26'da görüldüğü gibi **fdisk -l** komutunu kullanarak disk sürücülerini görüntüleyiniz.

```
┌──(root㉿kali)-[/home/kali]
└─# fdisk -l
Disk /dev/sda: 80.09 GiB, 86000000000 bytes, 167968750 sectors
Disk model: VBOX HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x4b5f1860

Device     Boot Start       End   Sectors  Size Id Type
/dev/sda1  *     2048 167968749 167966702 80.1G 83 Linux


Disk /dev/sdb: 28.64 GiB, 30752000000 bytes, 60062500 sectors
Disk model: Ultra
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 94AE3ABE-33F4-461A-96B9-F9376C347B13

Device       Start      End  Sectors  Size Type
/dev/sdb1     2048 60060325 60058278 28.6G Microsoft basic data
/dev/sdb2 60060326 60061349     1024  512K Microsoft basic data
```

Görsel 1.26: Disk sürücülerinin fdisk -l komutu kullanılarak görüntülenmesi

3. Adım: Görsel 1.27'de görüldüğü gibi **dd** aracını kullanarak disk imajı alma işlemini gerçekleştiriniz.

```
┌──(root㉿kali)-[/home/kali]
└─# dd if=/dev/sdb1 of=/home/kali/Desktop/test.img
```

Görsel 1.27: İmaj alma işleminde dd aracının kullanılması

Disk imajı alma işlemi tamamlandığında ekran görüntüsü Görsel 1.28'deki gibi olacaktır.

```
┌──(root㉿kali)-[/home/kali]
└─# dd if=/dev/sdb1 of=/home/kali/Desktop/test.img
44513760+0 records in
44513760+0 records out
22791045120 bytes (23 GB, 21 GiB) copied, 1104.51 s, 20.6 MB/s
```

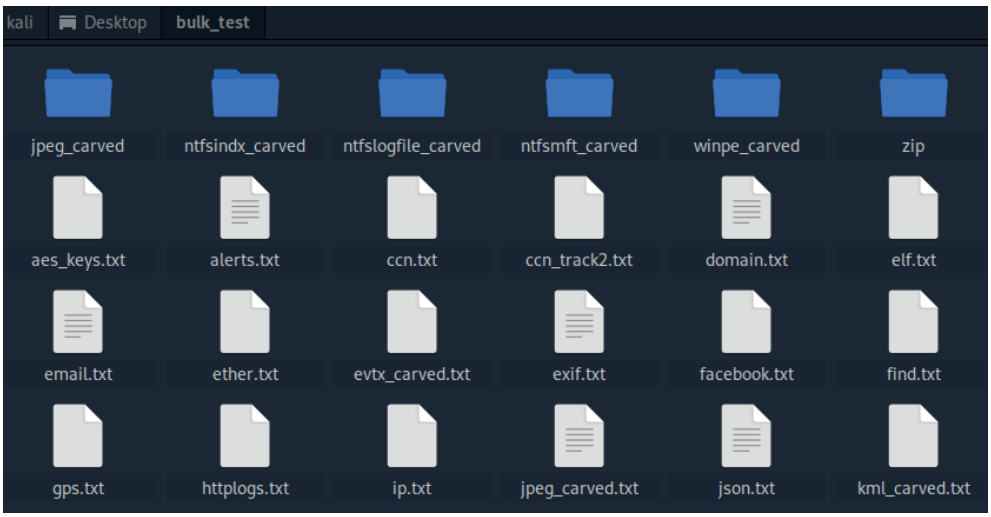
Görsel 1.28: Disk imajı alma işleminin tamamlanması

4. Adım: Görsel 1.29'da görüldüğü gibi disk imajını incelemek için **bulk_extractor** aracını kullanınız.

```
┌──(root㉿kali)-[/home/kali/Desktop]
└─# bulk_extractor -o bulk_test test.img
mkdir "bulk_test"
bulk_extractor version: 2.0.0
Input file: "test.img"
Output directory: "bulk_test"
Disk Size: 22791045120
Scanners: aes base64 elf evtx exif facebook find gzip httplogs json kml_carved msxml net ntfsindx ntfslogfile ntfsmft ntfsu
sn pdf rar sqlite utmp vcard_carved windirs winlnk winpe winprefetch zip accts email gps
Threads: 2
going multi-threaded ... ( 2 )
bulk_extractor      Tue Aug  8 12:15:01 2023

available_memory: 1174159360
bytes_queued: 0
depth0_bytes_queued: 0
depth0_sbufs_queued: 0
elapsed_time:  0:00:00
estimated_date_completion: 2023-08-08 12:15:00
estimated_time_remaining: n/a
fraction_read: 0.000000 %
max_offset: 0
sbufs_created: 0
sbufs_queued: 0
sbufs_remaining: 0
tasks_queued: 0
thread_count: 2
>.....|
```

Görsel 1.29: Disk imajının bulk_extractor aracıyla incelenmesi

5. Adım: Görsel 1.30'da görüldüğü gibi **bulk_extractor** incelemesi bittikten sonra sonuçları **bulk_test** klasörünün içine girerek görüntüleyiniz.



Görsel 1.30: Sonuçların bulk_test klasörü içinde görüntülenmesi

Sıra Sizde

Laboratuvardaki Kali işletim sistemi yüklü sanal makine aracılığıyla **dd** aracını kullanarak disk imajını alınız. Masaüstüne oluşturacağınız **uygulama klasörü** içine imaj dosyasını kaydediniz. İmaj inceleme işlemini, **bulk_extractor** aracını kullanarak gerçekleştiriniz. İnceleme işlemi bittikten sonra sonuç klasöründe **IP adreslerinin** yazıldığı dosya içeriğini **cat** komutu ile görüntüleyiniz.

1.7. SİBER GÜVENLİK SERTİFİKA PROGRAMLARI

Siber güvenlik sertifika programları, siber güvenlik alanında uzmanlaşmak isteyen kişilerin bilgi ve becerilerini geliştirmelerine yardımcı olan eğitim programlarıdır. Bu programlar; bilgisayar sistemlerini, ağları, güvenlik tehditlerini tanımak, güvenlik açıklarını tespit etmek ve siber saldırılara karşı korumak için gerekli yetkinlikleri bireylere kazandırma amacı taşır. Eğitim sonrasında aday, sertifika sınavına girip başarı ölçütlerini sağladığı takdirde sertifikaya sahip olur. Bu sertifikalar, iş başvuruları yapmak ve iş yerinde terfi almak için kullanılır.

Siber güvenlik alanında sürekli yeni sertifika programları geliştirilir. Sertifika seçiminde kişinin ilgi alanlarına, beceri seviyesine ve kariyer hedeflerine uygun sertifika programı önerilir. Siber güvenlik alanında mevcut olan bazı sertifika programları şunlardır:

- Bilgisayar Ağ Temelleri Sertifika Programı
- Siber Güvenlik Temelleri Sertifika Programı
- Beyaz Şapkalı Hacker Sertifika Programı
- Siber Olaylara Müdahale Ekibi Sertifika Programı
- Ofansif Güvenlik Sertifika Programı
- Adli Bilişim Uzmanlığı Sertifika Programı

Bilgisayar Ağ Temelleri Sertifika Programı: Ağ teknolojileri ve ağ yönetimi konularında kariyer planlayan veya mevcut ağ becerilerini geliştirmek isteyen kişiler için başlangıç seviyesi sertifika programıdır. Ağ temelleri sertifikası, siber güvenlik alanında uzmanlaşmak isteyenlere temel oluşturur. Ağ temelleri sertifika programının eğitim içeriğindeki konular şunlardır:

- OSI Referans Modeli
- TCP/IP Protokolleri
  - HTTP, HTTPS, FTP, TFTP, DHCP, DNS, SMTP, TCP, UDP, IPv4, IPv6, ETHERNET, WIFI, PPP
- Anahtarlama Temelleri
- VLAN Konfigürasyonu
- Ağ Güvenliği
  - Ağ Saldırı Türleri
  - Anahtar Port Güvenliği
  - Dinamik ARP Denetimi
  - DHCP İzleme

- Statik Yönlendirme
  - Statik Yönlendirme (Static Routing)
  - Varsayılan Rota (Default Route)
  - Kayan Statik Rota (Floating Static Route)
- Dinamik Yönlendirme
  - OSPF Yönlendirmesi
- Erişim Kontrol Uygulamaları
  - Standart Erişim Kontrol Listeleri
  - Extended Erişim Kontrol Listeleri
- NAT/PAT, DHCP, DNS Uygulamaları
- WLAN Yapılandırması
- Ağ Yedeklilik Sistemleri
- Yazılım Tabanlı Ağlar

Bilgisayar ağ temelleri sertifika eğitimleri; üniversiteler, sertifika veren kuruluşlar ve özel eğitim kurumları tarafından yapılır.

Siber Güvenlik Temelleri Sertifika Programı: Siber güvenlik alanında temel bilgi ve becerileri öğrenmek isteyenlere uygun bir sertifika programıdır. Bu sertifika programı, temel düzeyde siber güvenlik uzmanlarının sahip olduğu kazanımlara yönelik bir program içerir. Siber güvenlik temelleri sertifika programının eğitim içeriğindeki konular şunlardır:

- Ağ Güvenliği
- Bilgi Güvenliği
- Kimlik Doğrulama ve Erişim Kontrolü
- Kriptografi
- Olay İzleme ve Yönetimi
- Saldırılar ve Tehditler
- Sanallaştırma Uygulamaları
- Mobil ve Bulut Güvenliği
- Zararlı Yazılımlar
- Güvenlik Uygulamaları
- Saldırı Tespit Sistemleri
- Sanal Özel Ağ (VPN) Teknolojileri
- Kablosuz Ağlar

Siber güvenlik temelleri sertifika eğitimleri; üniversiteler, sertifika veren kuruluşlar ve özel eğitim kurumları tarafından yapılır.

Beyaz Şapkalı Hacker Sertifika Programı: Yasalara ve bilişim etiği kurallarına uygun şekilde, bilgisayar ağlarında, yazılımlarda ve diğer bilişim sistemlerinde güvenlik açıklarını tespit eden ve bu açıklara karşı güvenlik önlemleri alabilen etik hacker yetiştirmek amaçlanır. Beyaz şapkalı hacker sertifika programının eğitim içeriğindeki konular şunlardır:

- Etik Hacker Özellikleri
- Keşif İşlemi
- Ağ Tarama İşlemi
- Güvenlik Açığı Analizi
- Sistem Hackleme İşlemi
- Zararlı Yazılım Tehditleri
- Koklama Saldırıları
- Sosyal Mühendislik Saldırıları
- Servis Dışı Bırakma Saldırısı
- Oturum Ele Geçirme Saldırısı
- IDS, Güvenlik Duvarları ve Bal Küpü Tuzaklarından Kaçmak
- Web Sunucusu Hacklemek
- Web Uygulamalarını Hacklemek
- SQL Enjeksiyonu Atağı
- Kablosuz Ağları Hacklemek
- Mobil Platformları Hacklemek
- IoT Sistemlerini Hacklemek
- Bulut Bilişim
- Kriptografi

Beyaz şapkalı hacker sertifika eğitimleri; üniversiteler, sertifika veren kuruluşlar ve özel eğitim kurumları tarafından yapılır.

Siber Olaylara Müdahale Ekibi Sertifika Programı: Kamu kurumları ve kurumsal işletmelerde meydana gelebilecek farklı türde siber güvenlik olaylarını planlı bir şekilde ele alıp değerlendirebilen, hızlı ve etkin bir şekilde tepki verme yeteneğine sahip uzmanları yetiştirmeyi amaçlayan bir eğitim programıdır.

Siber olaylara müdahale ekibi sertifika programının eğitim içeriğindeki konular şunlardır:

- Olay Yönetimi ve Müdahale Süreci
- Adli Bilişim Uygunluğu ve İlk Yanıt
- Ağ Güvenliği Olaylarını Ele Alma ve Müdahale Etme
- Zararlı Yazılım Olaylarını Ele Alma ve Müdahale Etme
- E-Posta Güvenliği Olaylarını El Alma ve Müdahale Etme
- Web Uygulaması Güvenlik Olaylarını Ele Alma ve Müdahale Etme
- Bulut Güvenliği Olaylarını Ele Alma ve Müdahale Etme

- İç Ağdan Gelen Tehditleri Ele Alma ve Müdahale Etme

Siber olaylara müdahale ekibi sertifika eğitimleri, sertifika veren kuruluşlar ve özel eğitim kurumları tarafından yapılır.

Ofansif Güvenlik Sertifika Programı: Siber güvenlik alanında yetenekli profesyonellerin yetiştirilmesini amaçlayan bir eğitim programıdır. Bu sertifikayı alanlar, bilişim sistemlerine sızma testi yapacak düzeyde bilgi ve yetkinliğe sahip olur. Ofansif güvenlik sertifika programının eğitim içeriğindeki konular şunlardır:

- Sızma Testi
- Kali Linux İşletim Sistemini Tanıma
- Temel Araçlar
- Pasif Bilgi Toplama
- Aktif Bilgi Toplama
- Zafiyet Tarama
- Exploitlerle Çalışma
- Bellek Taşırma Atakları
- Dosya Transferleri
- Web Uygulama Atakları
- İstemci Tarafı Atakları
- Parola Atakları
- Port Yönlendirme ve Tünel Oluşturma
- Metasploit Framework
- Antivirüs Yazılımı Atlama

Ofansif güvenlik sertifika eğitimleri, sertifika veren kuruluşlar ve özel eğitim kurumları tarafından yapılır.

Adli Bilişim Uzmanlığı Sertifika Programı: Dijital ortamlarda gerçekleştirilen siber saldırılar sonucunda, çeşitli donanım ve yazılımlar kullanılarak dijital delillerin nasıl toplanacağı, analiz edileceği, çözümleneceği ve raporlanacağı konuları üzerine odaklanan eğitim sürecidir. Adli bilişim uzmanlığı sertifika programının eğitim içeriğindeki konular şunlardır:

- Adli Bilişim ve Bilişim Hukukuyla İlgili Temel Kavramlar
- Siber Güvenlikle İlgili Kavramlar
- Bilişim Etiği
- Ülkemizde Adli Bilişim ve Bilişim Hukukuyla İlgili Yasal Düzenlemeler
- Zararlı Yazılımlar
- Hacker Kavramı ve Hacker Türleri
- Siber Saldırı Türleri
- Kimlik Doğrulama Sistemleri
- İşletim Sistemlerinin Özellikleri

- Şifreleme ve Şifre Çözme
- Veri Kurtarma, Veri Saklama, Veri Silme
- Disk İmajlarının Alınması
- Delil Toplama, Analiz Etme, Çözümleme, Raporlama
- Adli Bilişim Süreci
- Adli Bilişim Sürecinde Kullanılan Donanım ve Yazılımlar
- Bilirkişi Raporu Hazırlama

Adli bilişim sertifika eğitimleri; üniversiteler, sertifika veren kuruluşlar ve özel eğitim kurumları tarafından yapılır. Eğitim sonu sertifika sınavında başarılı olan adaylar, **Adli Bilişim Uzmanlığı Sertifikası** almaya hak kazanır. Bu sertifikaya sahip kişiler, mahkemelerce bilirkişi olarak görevlendirilir.

Sıra Sizde

Aşağıda verilen sertifika programının içeriği ve bunların sertifika programı adlarını uygun olan harfleri kullanarak içeriklerin yanındaki parantezlere yazınız ve eşleştirmeyi tamamlayınız.

Sertifika Programının İçeriği	Sertifika Programının Adı
1. ( ) Yasalara ve bilişim etiği kurallarına uygun şekilde, bilgisayar ağlarında, yazılımlarda ve diğer bilişim sistemlerinde güvenlik açıklarını tespit eden ve bu açıklara karşı güvenlik önlemleri alabilen etik hacker yetiştirme programıdır.	a. Bilgisayar Ağ Temelleri
2. ( ) Dijital ortamlarda gerçekleştirilen saldırılar sonucunda dijital delillerin nasıl toplanacağına, analiz edileceğine, çözümleneceğine ve raporlanacağına odaklanır.	b. Siber Güvenlik Temelleri
3. ( ) Bilişim sistemlerine sızma testi yapacak düzeyde bilgi ve yetkinliğe sahip profesyonellerin yetiştirilmesini amaçlayan programdır.	c. Beyaz Şapkalı Hacker
4. ( ) Mevcut ağ becerilerini geliştirmek isteyen kişiler için başlangıç seviyesi sertifika programıdır.	d. Siber Olaylara Müdahale Ekibi
5. ( ) Kamu kurumları ve kurumsal işletmelerde meydana gelebilecek farklı türde siber güvenlik olaylarını planlı bir şekilde ele alıp değerlendirebilen, hızlı ve etkin bir şekilde tepki verme yeteneğine sahip uzmanları yetiştirmeyi amaçlayan bir eğitim programıdır.	e. Siyah Şapkalı Hacker
6. ( ) Temel düzeyde siber güvenlik uzmanlarının sahip olduğu kazanımlara yönelik bir programdır.	f. Adli Bilişim Uzmanlığı
	g. Ofansif Güvenlik

ÖLÇME VE DEĞERLENDİRME

A. Aşağıdaki parantezlerin içine cümlelerde verilen bilgiler doğru ise "D", yanlış ise "Y" yazınız.

1. ( ) Virüsler, bir belgeye veya yürütülebilir dosyaya ihtiyaç duymadan kendini kopyalar.

2. ( ) Stuxnet saldırısı, bir APT saldırısıdır.

3. ( ) ISO 27001, Bilgi Güvenliği Yönetim Sistemi için oluşturulmuş uluslararası bir standarttır.

4. ( ) Siyah şapkalı hackerlar, etik hacker olarak tanımlanır.

5. ( ) Bir kullanıcıya ait şifreyi, olası kombinasyonları deneyerek elde etmeye yönelik siber saldırılara kaba kuvvet saldırıları denir.

B. Aşağıdaki cümlelerde boş bırakılan yerlere doğru sözcükleri yazınız.

6. Bilişim sistemlerinin güvenliğini sağlamak, zafiyetleri tespit etmek ve ortadan kaldırmak için etik olarak çalışan hackerlara ..... hacker denir.

7. Kişisel Verileri Koruma Kurumu ..... sayılı Kişisel Verilerin Korunması Kanunu'nun yüklediği görevleri yerine getirir.

8. Bir iddianın doğruluğunu veya yanlışlığını ispat etmek için mahkemeye sunulan kanıtlara ..... denir.

9. Bir diskin ..... almak için dd aracı kullanılır.

10. Ağ cihazlarına biyometrik veriler kullanılarak erişim sağlanması ..... güvenlik önlemidir.

C. Aşağıdaki soruları dikkatlice okuyarak doğru cevabı işaretleyiniz.

11. Aşağıdakilerden hangisi kullanıcılarda korku ve endişe oluşturmak amacıyla geliştirilen zararlı yazılımdır?

A) Adware B) Ransomware C) Scareware
D) Solucan E) Virüs

12. Aşağıdakilerden hangisi bilgi güvenliği temel unsurlarındandır?

A) Erişilebilirlik B) Keşfedilebilirlik C) Kullanılabilirlik
D) Verimlilik E) Yüklenebilirlik

13. Aşağıdakilerden hangisi PUKÖ döngüsü unsurlarından değildir?

A) Önlem Al B) Kontrol Et C) Uygula
D) Planla E) Gözden Geçir

14. Aşağıdaki araçlardan hangisi ile imaj dosyası üzerinde inceleme gerçekleştirilir?

A) bettercap B) bulk_extractor C) cat D) dd E) fdisk

15. Aşağıdaki siber güvenlik sertifikalarından hangisi bilişim sistemlerine sızma testi yapma yeterliliği kazandırır?

A) Adli bilişim uzmanlığı
B) Bilgisayar ağ temelleri
C) Siber güvenlik temelleri
D) Ofansif güvenlik
E) Siber olaylara müdahale ekibi

D. Aşağıdaki soruların cevabını ilgili alana yazınız.

16. Kişilerin veya kurum ve kuruluşların siber güvenlik etik ilkelerini benimsemesinin önemini açıklayınız.

17. Siber zorbalık olarak değerlendirilebilecek durumların neler olduğunu yazınız.

18. Bir bilişim sisteminde sunucu odasının fiziksel güvenliğinin nasıl sağlanacağını yazınız.

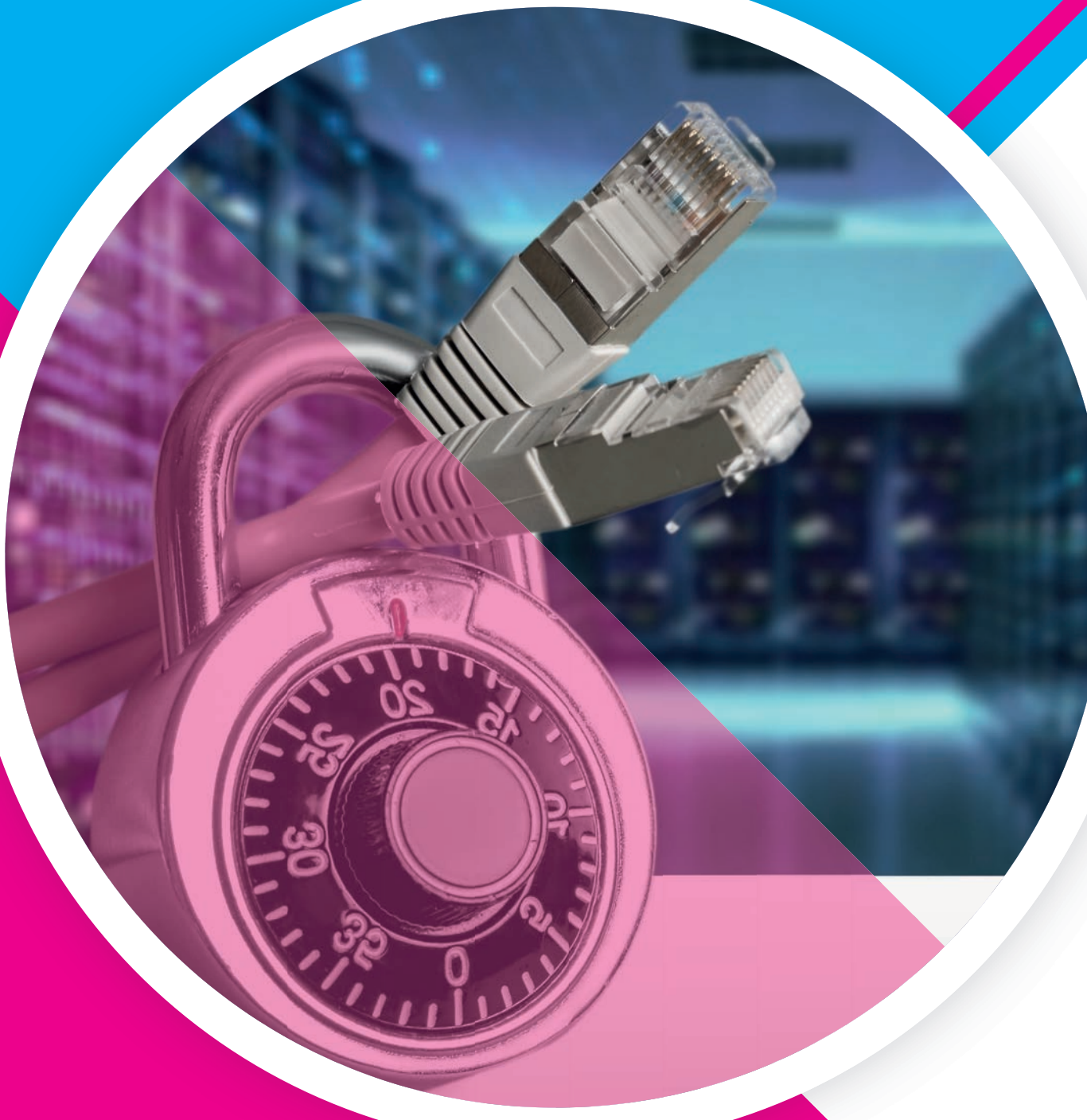
19. Bilgi güvenliği standartlarının kurum ve kuruluşlar için önemini açıklayınız.

20. Dijital arşivleme avantaj ve dezavantajlarını yazınız.

21. Ülkemizde adli bilişim konusundaki yasal düzenlemeleri yazınız.

22. Siber güvenlik sertifikalarından hangilerine sahip olan kişilerin sızma testi gerçekleştirebileceğini yazınız.

AĞ VE SİSTEM GÜVENLİĞİ



2. ÖĞRENME BİRİMİ

KONULAR

2.1. AĞ GÜVENLİĞİ İLKELERİ
2.2. AĞ VE SİSTEM GÜVENLİĞİ CİHAZLARI
2.3. AĞ GÜVENLİK MİMARİSİ
2.4. AĞ SALDIRI TÜRLERİ
2.5. İŞLETİM SİSTEMLERİ GÜVENLİK İLKELERİ
2.6. SUNUCU SİSTEMLERİ GÜVENLİK İLKELERİ
2.7. AĞ CİHAZLARININ SIKILAŞTIRILMASI
2.8. DONANIM GÜVENLİĞİ İLKELERİ

ANAHTAR KELİMELER

Brute Force, DMZ, DLP, firewall, güvenlik ilkeleri, Netbios, NTP, PowerShell, Proxy, SMTP, SQL, SQL injection, siber, sunucu, ortadaki adam saldırısı, Telnet, VPN

NELER ÖĞRENECEKSİNİZ?

- Ağ güvenliği ilkelerini kavrama
- Ağ ve sistem güvenliği için kullanılacak cihazları tanıma
- Ağ güvenlik mimarilerini açıklama
- Ağ saldırı türlerini kavrama
- İşletim sistemi güvenlik ilkelerini kavrama
- Sunucu sistemleri güvenlik ilkelerini kavrama
- Ağ cihazlarının sıkılaştırılması ile ilgili kavramları açıklama
- Donanım güvenliği ilkelerini kavrama

HAZIRLIK ÇALIŞMALARI

1. Bir sistem odasının fiziksel güvenliği nasıl sağlanabilir? Düşüncelerinizi sınıfta arkadaşlarınızla değerlendiriniz.
2. Yerel ağlarda (LAN) hangi saldırılar düzenlenebilir? Düşüncelerinizi sınıfta arkadaşlarınızla tartışınız.

2.1. AĞ GÜVENLİĞİ İLKELERİ

Ağ güvenliği; ağda bulunan verilerin, bilgilerin, ağ trafiğine dâhil olan bütün cihazların ve dijital ögelerin izinsiz kullanımı, yetkisiz erişimi, imha edilmesi, değiştirilmesi gibi zarar verici işlemlerden korunmayı ifade eder.

Her geçen yıl dijital sistemlerin sayısında ve kullanıcılarında artış gerçekleşir. Bu artışla beraber yapılan siber saldırılar da her geçen gün çoğalmıştır. Yapılan siber saldırıların bir kısmı dijital ögeleri kullanan kişilerin bilgi eksikliğinden bir kısmı da teknik kavramların ve güvenlik ilkelerinin eksikliğinden kaynaklanır.

Ağ güvenliği, siber güvenliğin önemli bir unsurudur. Ağ güvenliği ilkelerini belirlemek ve uygulamak ise kişi veya kurumların bilgilerinin bütünlüğünün, gizliliğinin ve kullanılabilirliğinin korunması açısından büyük bir önem arz eder. Ağ güvenliği ilkeleri; ağa erişebilirliği belirleyen kuralları ve prosedürleri, ağa giriş yapabilecek yetkili ve yetkisiz kişilerin rolleri ve etki alanları, ağ cihazlarının güvenlik politikaları ve web erişimi üzerinden yönetimi belirleyen güvenlik politikalarının bir parçasıdır.

2.1.1. Ağ Güvenliğinin Önemi

Bilgisayar ağlarını kullanan bireysel ve kurumsal kullanıcıların sayısındaki artış, güvensiz bir ortamda çok bilginin de depolanmasına yol açar. Hayatın her alanında teknolojik ögelerin kullanılmaya başlaması, beraberinde güvenlik önlemleri alınmasını bir zorunluluk hâline getirmiştir. Fiziki güvenlik ihtiyaçları kişiler için ne kadar önemliyse günümüz dünyasında dijital güvenlik de bir o kadar önemli hâle gelmiştir.

Küçük, orta veya büyük çaplı işletmeler; verilerin güvenliğinin, kaynaklarına erişimin ve bütünlüğünün sağlanması için çeşitli yatırımlar yapar. Ağ güvenliği; kişileri ve kurumları zararlı yazılımlardan, çeşitli saldırı yöntemlerinden, sistem kesintilerinden koruyarak maddi ve manevi kayıplarının önüne geçer.

Günümüzün siber dünyasında her kuruluş çevrimiçi kaynaklarını korumak için ağ güvenlik süreçleri ve çözümleri uygulamalıdır. Tüm ağ güvenliği çözümleri, ağ güvenliğinin temel ilkelerine uygun olmalıdır.

Ağ güvenlik ilkeleri şu yapıları koruyacak şekilde planlanmalıdır:

- **Erişilebilirlik:** Ağ ortamındaki veriler, yetkisi bulunan kişiler tarafından ulaşılabilir olmalıdır. Verilere yetkisiz kişiler tarafından ulaşılması veya yetkili kişilerin erişememesi, ağ güvenliği açısından sorun teşkil eder.
- **Gizlilik:** Ağ ortamında bulunan veriler, yetkisiz kişilere karşı korunmalıdır. Verinin gizliliğinin çeşitli zararlı yazılımlar veya saldırılar sonucu ortadan kaybolması, ağ güvenliğini tehdit eden önemli unsurlardan biridir.

- **Bütünlük:** Ağ üzerinde bulunan verilerin isteyerek veya yanlışlıkla silinmesi, değiştirilmesi, verinin bütünlüğü açısından problem oluşturur. Verinin bütünlüğünün kaybolması, verinin doğruluk ve güvenliğinin de kaybolması demektir.

2.1.2. Ağ Güvenliğini Sağlama

Ağ güvenliği ile siber güvenlik kavramları arasında farklılıklar bulunur. Ağ güvenliği, bilgisayar ağını donanımsal ve yazılımsal yöntemler kullanarak korumayı amaç edinir. Bu durum, veri ve ağın gizliliğini, erişilebilirliğini sağlar. Büyük miktarda veri işleyen her işletme, birçok siber tehdide karşı bir dereceye kadar çözüme sahiptir. Siber güvenlik ise sistemi siber ve kötü niyetli saldırılardan korumak için alınan önlemlerin bütününü ifade eder. Siber güvenlik, bilgisayar korsanının sisteme yetkisiz erişimini önleyebilmek için sistemin güvenliğini artırmayı amaçlar. Siber alanı, saldırılara ve kayıplara karşı korur.

Sistemi daha güvenli hâle getirmek için ağ güvenlik planında gerekli adımlar bulunur. Bu adımlar şunlardır:

- **Güçlü Parolalar Kullanmak:** Ağ güvenliğini korumak için öncelikli olarak güçlü parolalar kullanılmalıdır. Parolalar ne kadar karmaşık olursa sisteme sızmak isteyenlerin işi bir o kadar zor olacaktır.
- **Saldırı Tespit ve Önleme Cihazları Kullanmak (IDS-IPS):** Güvenlik duvarlarından farklı olarak bir saldırı tespit ve önleme sistemi (IDPS), ağı kötü amaçlı etkinliklere karşı izler, ağ güvenliği olaylarını ve olası tehditleri ağ yöneticisine bildirir ve bunlara müdahale eder. Saldırı tespit cihazları (Instrusion Detection System-IDS), normal olmayan trafiği tespit eder. Siber korsanlardan veya başka kötü niyetli kişilerden gelen trafiği bulmaya çalışır. Saldırı önleme sistemi (Instrusion Prevention System-IPS) ise üzerinden geçen trafiğin kötü niyetli kişiler tarafından gerçekleştirildiğini anladığında o trafiği engeller.
- **Sistemin Güncel Kullanılmasını Sağlamak:** Kullanılan sistemlerin güncel kalmasını sağlamak, önemli bir güvenlik önlemidir. Sistemi güncel tutmak, birçok güvenlik açığının kapatılması ve zararlı yazılımlara karşı da önlem alınması demektir.
- **Antivirüs Kullanmak:** Zararlı yazılımlardan korunmanın bir yolu da antivirüs kullanımıdır. Ev kullanıcıları ve kurumsal kullanıcılar mutlaka antivirüs çözümlerini kullanmalıdırlar. Gerek sunucu gerekse istemci cihazlarda mutlaka antivirüs kurulumu yapılmalıdır. Yeni nesil antivirüs sistemler; güvenlik duvarları, IPS ve IDS sistemlerle entegre çalışır ve daha etkili koruma sağlar.
- **Güvenlik Duvarı Oluşturmak:** Güvenlik duvarları, ağdan dışarıya çıkacak veya içeriye alınacak paketleri inceleyen sistemlerdir. Güvenlik duvarlarını doğru ve etkili bir şekilde yapılandırmak, ağ saldırılarını önlemede büyük önem arz eder.
- **Yedek Almak:** Belli zaman dilimlerinde verilerin yedeğini almak, güvenlik politikaları için önemlidir. Alınan yedeklerin farklı lokasyonlarda tutulması, verilerin erişilebilirliği ve güvenliği için temel ilkelerden biridir.
- **Yapılandırma Güvenliğini Sağlamak:** Ağ cihazlarının doğru yapılandırılması gerekir. Ağ ortamında doğru yapılandırılmamış bir cihaz kullanılırsa yanlış kullanımdan kaynaklı veya saldırı sonucu cihazın güvenliği ortadan kalkabilir.
- **Log Yönetimi Sağlamak:** Verilerin log yönetim sistemleri ile takip edilerek verilere kimlerin, ne zaman erişim sağladığının kayıt altına alınması ve bu verilerin anlık veya düzenli olarak raporlanması, yaşanacak veri kayıplarının önüne geçer.

- **Güvenli Bağlantılar:** İnternette güvenilir olmayan bir bağlantıya tıklamanın kullanılan ağa birçok zararlı yazılımı davet edebileceğinin unutulmaması gerekir. Bu nedenle veriler yalnızca güvenli bağlantılardan indirilmeli, bilinmeyen bağlantılarda ve web sitelerinde gezinmekten kaçınılmalıdır. Bu durum için alınması gereken en büyük önlemlerden biri, çalışanlara eğitim verilerek onların bilinçlendirilmesidir.
- **Sanal Özel Ağlar (VPN) Kullanmak:** İnternet gibi daha az güvenli bir ağ üzerinden güvenli ve şifreli bağlantı oluşturan bir teknolojidir. VPN, internet gibi ortak bir ağı kullanarak özel bir ağı genişletmenin yoludur. VPN'lerde bağlantı şifreli olduğu için gönderilen ve alınan tüm veriler şifrelenir, üçüncü taraf kişilerce ağ trafiği izlenemez.
- **Kablosuz Ağ Güvenliği:** Ağı kablosuz bağlantılara özgü güvenlik açıklarına karşı korur. Wi-Fi ağlarının sinyallerinin çevredeki cihazlara açık bir şekilde yayınlanması, tehdit aktörlerinin ağa erişmeye çalışması için ek fırsatlar yaratır. Kablosuz ağlar üzerinden iletilen verilerin şifrelenmesi, erişimi kısıtlamak için MAC adreslerinin filtrelenmesi, ağ adının yayınlanmasını önlemek için ağ SSID'sinin özelleştirilmesi gibi yöntemlerle kablosuz güvenlik artırılır.
- **Siber İstihbarat:** Bir kurum için risk hâline gelebilecek siber tehdit faktörlerini önceden analiz eden, kuruma bu yönde uyarı ve önlem alma fırsatı sağlayan farklı çözüm ve araçlardır. Kurumun bu alanda bir politikasının olması, kurumu birçok saldırı ve zararlı yazılımdan korur.

2.2. AĞ VE SİSTEM GÜVENLİĞİ CİHAZLARI

Dijital ögeler hayatın bir parçası olmaya başladığı andan itibaren ağ ve sistem güvenliği önem arz etmiştir. Sisteme yetkisiz girişlerin ve bilgi hırsızlığının önüne geçmek için kimlik doğrulama ve erişim kontrolü sağlayan cihazlar üretilmiştir. Saldırı tespit cihazları, bu alanda hizmete sunulan ilk güvenlik araçlarından biridir. Bu araçlar sayesinde ağ güvenlik uzmanları çok hızlı bir şekilde saldırıları tespit edip gerekli önlemleri almışlardır. 1984 yılında üretilen IDS cihazlarının ardından 1990'lı yılların sonlarına doğru saldırı önleme cihazları üretilmiştir. IPS'nin IDS'den farkı, saldırıları tespit etmekle kalmaması, saldırılara karşı önlem de alabilmesidir. IDS ve IPS'nin saldırıları tespit ve önlemede başarılı olduğunun görülmesinin ardından çeşitli kuralların belirlenebildiği, ağdan içeri giren ve dışarı çıkan verilerin kontrol edilebildiği Firewall sistemleri geliştirilmiştir. Firewall sistemleri, 1980'li yılların sonunda cihaz olarak kullanılmıştır. 1988 yılında paket filtreleme yapabilen ilk firewall cihazı üretilmiştir. Bu cihazlar sadece belirli kurallara göre trafiğe izin verir veya engeller şekilde çalışmıştır. Güvenlik araçlarının gelişmeye devam etmesiyle statefull (durum bazlı) cihazlar, paket filtrelemeye ek olarak üretilmiştir. Günümüzde ise çeşitli ağ cihazlarına entegre şekilde çalışmasının yanında ayrı bir cihaz olarak da ağ ve sistem güvenliği cihazları ile karşılaşılır. Teknolojinin gelişimine bağlı olarak yapay zekâ ve bulut teknolojilerini kullanan cihazlar yaygın olarak kullanılır.

2.2.1. Ağ ve Sistem Güvenliğine Yönelik Atakları Algılama ve Önleme Cihazları

Ağ ve sistem güvenliğine yönelik atakları algılama ve önleme cihazlarından bazıları şunlardır:

- Firewall
- Proxy
- DLP (Data Loss Prevention-Veri Kaybı Önleme)

- SIEM (Security Information and Event Management-Güvenlik Bilgileri ve Olay Yönetimi)
- IOC (Indicator of Compromise-Güvenlik Açığı Göstergesi)
- VAG (Virtual Air Gap-Sanal Hava Boşluğu)
- Secure Mail Gateway
- Analytics
- WAF (Web Application Firewall-Web Uygulaması Güvenlik Duvarı)

2.2.1.1. Güvenlik Duvarı (Firewall)

Ağa gelebilecek saldırıları önlemede kullanılan en yaygın cihazdır. Belirli kuralların cihaz tarafından uygulanması, kural dışına çıkıldığında ise önlem alınması mantığı ile çalışır. OSI modelinin 3. (Ağ) ve 4. (İletim) katmanlarında çalışan firewall sistemler, yerel ağ ile geniş alan ağı arasında konumlandırılarak gelen ve giden paketleri filtreler. Tüm ağ trafiğinin üzerinden geçmesini sağlayan bu cihazlar, lokal ağa gelen paketin hangi kaynaktan, hangi porttan geldiğine bakar, hangi hedefe ve hangi uygulamaya yönelik bir paket olduğunu inceleyerek paketin girişine onay verir veya paketi engeller. Firewall sistemler; yapısında port filtreleme, IP filtreleme, uygulama filtreleme, web filtreleme, virüs imza tabanı filtreleme gibi özelliklere sahiptir. Kullanılan servisler ve portlar, firewall sistemlerde bir kurallar dizini olarak hazırlanır ve bu kurala bağlı olarak yapılacak işlemler belirlenir. Bu kurallar dizinine **White List (Beyaz Liste)** adı verilir ve bu listeye bağlı olarak ağa erişim izni sağlanır. Bu listenin dışında kalan bütün bağlantı istekleri bloke edilerek engellenir.

Firewall sistemler, donanımsal ve yazılımsal olarak sınıflandırılır. Yazılımsal güvenlik duvarları, yazılım tanımlı ağ (SDN) mimarisi ile çalışabilen firewall sistemlerdir. Ağ trafiğini izleyen ve merkezî kontrolcüye sahip olan bu sistemler, ağ trafiği log kayıtlarından elde edilen sonuçlara göre çalışır. Donanımsal güvenlik duvarları ise özel donanımlar üzerinde çalışan sistemlerdir. Fiziksel olarak lokal ağ sistemi veya bulut teknolojisi üzerine konumlandırılan bu cihazlar sayesinde günümüzde aktif bir şekilde saldırı tespit ve önleme işlemleri yapılır.

Çalışma yapısı bakımından firewall cihazlar, statefull (durum bilgisine sahip) ve stateless (durum bilgisine sahip olmayan) olmak üzere ikiye ayrılır. Statefull firewall cihazları anlık olarak trafiği işleyebildikleri için sahte mesaj ve yetkisiz erişimi engelleme noktasında oldukça başarılıdır. Stateless firewall cihazlarda ise erişim listeleri (ACL) hazırlanarak paketlerin giriş ve çıkışlarının kontrol edilmesi sağlanır. Anlık olarak bilgiye sahip olmayan stateless cihazlar, belirlenen kurallar çerçevesinde paketleri geçirir veya bloklar. Stateless cihazlar, anlık durum bilgisine sahip olmadığı için trafiğin hızlı akmasını sağlar fakat güvenlik düzeyi statefull cihazlara nazaran daha zayıftır.

Teknoloji geliştikçe saldırı yöntem ve türlerinde de farklılıklar meydana gelmiştir. Güvenlik duvarlarını atlatan saldırıların sayısındaki ve yöntemlerindeki farklılıklar, üretici firmalar tarafından farklı model ve türlerde firewall cihazların üretimini zorunlu kılmıştır. Bu türlerden bazıları şu şekilde sıralanabilir:

- **Yeni Nesil Güvenlik Duvarı (Next Generation Firewall-NGFW):** Performanstan ödün vermeden paket içeriğini, kaynak-hedef ve kullanıcı davranışlarını kontrol edebilir. NGFW ile geleneksel mimari arasındaki en belirgin ve önemli fark, yeni nesil güvenlik duvarının trafiği oluşturan uygulamaları tanıyabilen bir mimariye sahip olmasıdır. Bu durum, uygulamaların ayrıştırılmasını ve iş kurallarına göre belirlenmiş kurumsal politikalar oluşturulmasını sağlar. Yeni nesil güvenlik duvarlarında veri paketlerinin çok detaylı incelenebilmesi (DPI) sayesinde veri paketlerinin ikinci bir üründe kontrol edilmesine gerek kalmaz ve bu durum, performans artırır ve maliyeti düşürür.

- **Ağ Adresi Çevirisi Güvenlik Duvarı (Network Address Translation-NAT):** Farklı ağ adreslerine sahip dâhilî cihazların tek bir IP adresi üzerinden ağ bağlantısını sunarak iç ağdaki IP adreslerinin gizliliğini sağlar ve IP adresi üzerinden saldırı planı yapmayı düşünen hackerlara izin vermez.
- **Durum Bilgisi Olan Çok Katmanlı Denetim Güvenlik Duvarı (Stateful Multilayer Inspection Firewall-SMLI):** Ağ iletimi ve uygulama katmanlarında yer alan veri paketlerinin incelemesini ve karşılaştırmasını yaparak yalnızca güvenli paketlerin sistemden geçmesine izin verir. En yaygın kullanılan güvenlik duvarlarından biridir ve öğrendiği etkileşimleri baz alarak karar verir.
- **Bütünleşik Güvenlik Duvarı (Unified Threat Management-UTM):** Günümüzde birçok markanın çeşitli güvenlik önlemleri sunan farklı cihazları bulunur. Bütünleşik güvenlik duvarları ise bu farklılıkları hem maliyet hem de merkezî kontrol sağlama avantajıyla ortadan kaldırarak tek bir cihaz altında bu hizmetleri buluşturur. UTM; saldırılara karşı koruma sisteminin yanı sıra içerik ve mail filtreleme, saldırı ve casus sistemlerin tespiti işlemlerini yapabilir.

2.3. AĞ GÜVENLİK MİMARİLERİ

Ağın içinde bulunan cihaz ve kullanıcıların, hizmet veren servis ve uygulamaların, ağı yöneten güvenlik ekipmanlarının ilke ve politikalarına **ağ güvenliği mimarileri** adı verilir. Ağ güvenliği mimarilerindeki asıl amaç, güvenliği tehdit eden durumlarla başa çıkabilmek ve ağın performanslı bir şekilde çalışmasını sağlamaktır.

Ağ güvenliği mimarileri tasarlanırken dış ağa hizmet verecek servisler, iç ağda bulunan ağ cihazları ve sınırlanmaları net bir şekilde belirlenmelidir. Dış ağa hizmet verecek servislerin erişimi farklı güvenlik seviyelerinde denetim altına alınmalıdır. Ağ güvenliği mimarileri oluşturulurken ağa giren ve çıkan bütün paketlerin kontrol edildiği bir izleme sistemi hazırlanmalıdır.

Ağ güvenliği mimarileri; donanım, yazılım, işletim sistemi, uygulamalardan oluşan ve kurumların mevcut bilgi işlem yapılarına entegre edilen sistemlerdir. Ağ güvenliği mimarileri sayesinde hem yerel ağ hem de dış ağdan gelecek güvenlik tehditleriyle mücadele edebilecek bir yönetim modeline sahip olunur.

2.3.1. Ağ Güvenliği Mimarilerinin Elemanları

Ağ güvenliği mimarilerini oluşturan birçok araç bulunur. Bu araçları aktif ve pasif ağ güvenliği bileşenleri olarak sınıflandırmak mümkündür. Anahtarlama cihazı, yönlendirici, firewall, NIC kartı, gateway gibi cihazlar aktif ağ cihazlarıdır. Kabinet, kanal, priz, kablo, patch panel, patch port gibi ağ elamanları ise pasif ağ cihazları kapsamına girer.

Anahtarlama cihazı, yerel ağda kullanılan en önemli ağ cihazlarından biridir. Geleneksel anahtarlama cihazı, OSI referans modeli 2. katmanında çalışırken günümüzde 3-7 arasındaki katmanlarda çalışan anahtarlama cihazları da vardır. Anahtarlama cihazı birçok güvenlik önlemini üzerinde bulundurur. VLAN, port güvenliği, olay kayıtları, erişim denetimi, port yönlendirme gibi birçok güvenlik teknolojisi, anahtarlama cihazları ile tümleşik şekilde kullanılır.

Yönlendirici, bir ağ topolojisinde dış ortama bakan en önemli cihazdır. Gelen paketlerin yönlendirilip hedefe ulaşmasından sorumludur ve saldırıya en çok uğrayan cihazdır. Yönlendirici, temel güvenlik kontrolleri dışında üzerine ilave edilen yazılımlarla güvenlik duvarı, saldırı tespit sistemi gibi gelişmiş güvenlik özelliklerini de kendi bünyesinde barındırabilir.

Güvenlik duvarı, ağ trafiğini güvenli bir şekilde kontrol etmek amacıyla güvenlik mimarilerine eklenen cihazdır. Bu çerçevede güvenlik duvarı, iç ağdan dış ağa giden veya dış ağdan iç ağa gelen trafiği kontrol ederek istenmeyen paketlerin ağa girmesini ve ağdan çıkmasını engeller.

Silahsızlandırılmış alan (Demilitarized Zone-DMZ) gerek iç ağa gerekse dış dünyaya hizmet veren sunucuların konumlandırıldığı, güvenlik önlemleri iç ağa göre daha az olan bölgedir. Genellikle bu bölgede web, uygulama, e-posta sunucuları konumlandırılır. Dış ağ ise tamamen kontrolsüz, güvenlik dışı olan bölgedir.

Saldırı tespit sistemlerinin temel amacı; ağa yapılan saldırıları tespit edip kayıt altına almak, sistem yöneticisine gerekli uyarılarda bulunarak saldırıya karşı gereken önlemleri zamanında almak ve ağda meydana gelen olayların geriye doğru izinin sürülmesini sağlamaktır. Gelişen teknolojiyle beraber hem saldırı tespiti hem de **saldırı önleme sistemleri** ağ güvenlik mimarilerinde birlikte kullanılır. Güvenlik mimarilerinde IPSec (Internet Protocol Security), PPTP (Point-to-Point Tunneling Protocol), L2TP (Layer 2 Tunneling Protocol) gibi güvenlik protokollerini kullanan donanımsal ve yazılımsal **VPN** cihazları da aktif eleman olarak konumlandırılır.

2.3.2. Ağ Güvenliği Mimarilerinin Modelleri

Ağ mimarilerinin güvenli ve performanslı bir şekilde çalışması çok önemlidir. Bu noktada tasarımlar belirlenirken öncelikle ihtiyaçlar tespit edilmelidir. İhtiyaçlar çerçevesinde ağ içinde hangi hizmetlerin verileceği, kullanılacak protokoller, ihtiyaç duyulan bant genişliği ve öngörülen kullanıcı sayıları hem güvenlik hem maliyet hem de ağın performanslı çalışması açısından planlanmalıdır. Ağ mimarileri ve güvenliği içinde tespit edilen ihtiyaçlara yönelik araçlar ve erişim denetim politikaları mutlaka belirlenmelidir. Kurum ağında erişim politikaları; güvenlik duvarları, anahtarlama, VPN cihazlarıyla uygulanır. Bütün bunların yanında ağın kullanım zamanının kısıtlamaları, büyüme hızı ve maliyet temel alınarak ağ güvenliği mimarilerinin modelleri oluşturulur.

2.3.2.1. Ağ Güvenliği Mimarilerinde Sanal Yerel Ağ (VLAN) Kullanımı

VLAN ile yerel alan ağlarındaki kaynaklar ve ağ kullanıcıları gruplandırılır. Kurum içinde birbirinden farklı alanlarda (muhasebe, insan kaynakları, üretim vb.) ve türlerde kullanıcılar bulunur. VLAN sayesinde farklı görev ve işlerde çalışan kullanıcıların ağ trafiği birbirinden izole edilerek ağ performansı artırılır, güvenlik sağlanır.

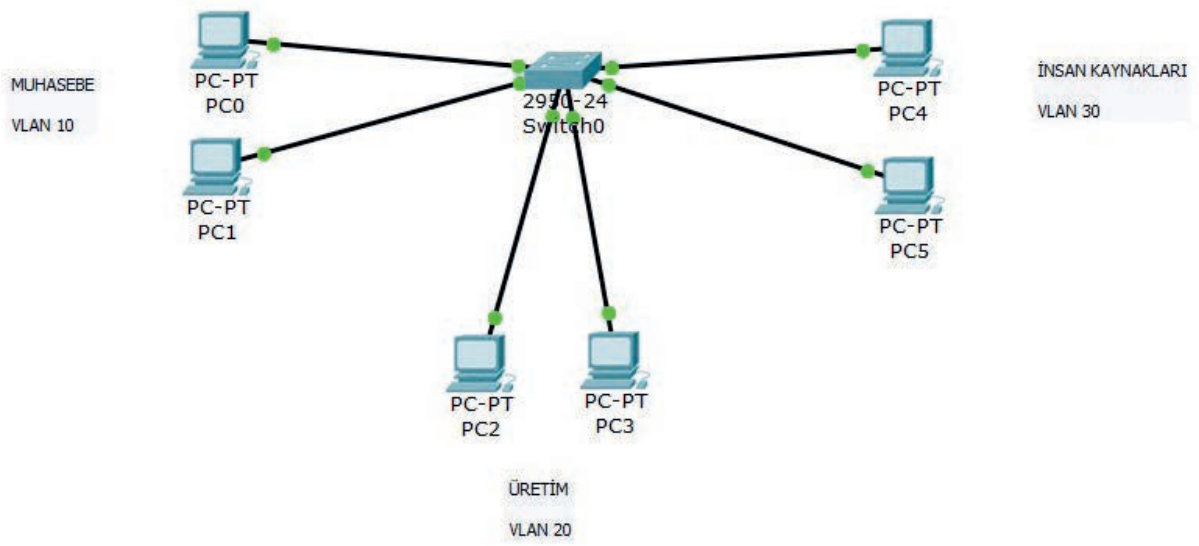
VLAN, ağ yönetimi açısından yüksek oranda verimlilik sağlayan bir teknolojidir. Benzer ağlar kullanan kullanıcılar aynı VLAN ağını paylaşabilir. Böylece şirket içinde departman değişikliği yapan bir personelin ağ değişikliği yapmasına gerek kalmaz. Bu durum, bilgi teknolojileri departmanının iş yükünü azaltmaya yardımcı olur. Hassas verilere sahip firmalar, VLAN sayesinde bu verilerin bulunduğu ağı izole ederek diğer ağlardan ayırır ve güvenliği en üst seviyeye çıkarır.

Yönetilebilen bütün anahtar cihazlarında VLAN desteği bulunur. Her bir anahtar arayüzü için varsayılan VLAN değeri VLAN1'dir. Öncelikle varsayılan VLAN değerini değiştirerek saldırıların önüne geçmek mümkündür.

1. Uygulama

İşlem adımlarına göre yerel ağda VLAN yapılandırmasını gerçekleştiriniz.

1. Adım: Ağ simülasyon programını kullanarak Görsel 2.1'deki topolojiyi oluşturunuz.



Görsel 2.1: VLAN topolojisi

2. Adım: Anahtarlama cihazının komut satırını açınız ve ilgili VLAN'ları oluşturmak için komutları giriniz.

```
Switch>
Switch>enable
Switch#configure terminal
Switch(config)#vlan 10
Switch(config-vlan)#name MUHASEBE
Switch(config-vlan)#exit
Switch(config)#vlan 20
Switch(config-vlan)#name URETIM
Switch(config-vlan)#exit
Switch(config)#vlan 30
Switch(config-vlan)#name IK
```

3. Adım: VLAN10-20-30 için ilgili anahtar portlarının atamasını yapınız.

```
Switch(config)#interface range fastEthernet0/1-5
Switch(config-if-range)#switchport mode access
Switch(config-if-range)#switchport access vlan 10
Switch(config-if-range)#exit
Switch(config)#interface range fastEthernet 0/6-10
Switch(config-if-range)#switchport mode access
Switch(config-if-range)#switchport access vlan 20
Switch(config-if-range)#exit
Switch(config)#interface range fa0/11-15
Switch(config-if-range)#switchport mode access
Switch(config-if-range)#switchport access vlan 30
```

4. Adım: Kullanıcı cihazlarının IP adresi girişlerini yapınız.

VLAN10: 192.168.10.X

VLAN20: 192.168.20.X

VLAN30: 192.168.30.X

5. Adım: Cihazlar arasında ping atarak yapılandırmayı kontrol ediniz.

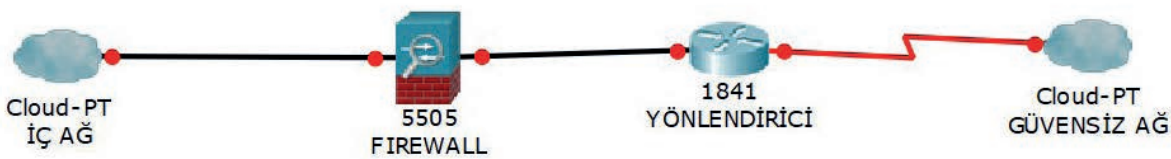
Sıra Sizde

Görsel 2.1'deki topolojiyi kullanarak varsayılan VLAN'ın (default VLAN) numarasını saldırılardan korunmak amacıyla değiştiriniz ve farklı VLAN numarasına sahip cihazların haberleşebilmesi için gerekli yapılandırmayı hazırlayınız. Yaptığınız değişikliği ağ simülasyonu programında anahtar cihazın show komutlarını kullanarak kontrol ediniz.

2.3.2.2. Tek Güvenlik Duvarıyla Gerçeklenmiş Ağ Mimarileri

Ağ güvenliği mimarilerinde Silahsızlandırılmış Alan (Demilitarized Zone-DMZ) olması veya olmaması durumuna göre iki farklı mimari kullanılır. DMZ'ler herkese açık bilgilerin bulunduğu kısımdır. Özel ve kritik öneme sahip veriler, herkesin ulaşabildiği DMZ'lerin arkasına konumlandırılır. DMZ üzerindeki sunucular; özel bilgileri, kaynak kodları ve ticari bilgileri içermemelidir.

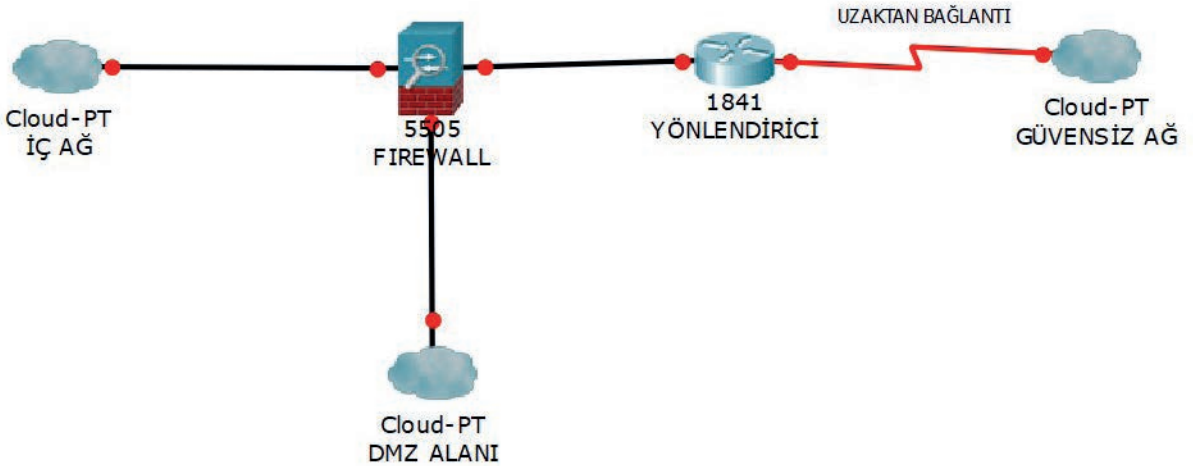
- **DMZ Kullanılmayan Ağ Mimarileri Modeli:** DMZ kullanılmayan ağ mimarilerinde iç ağda sunucu olmaması gerekir. DMZ kullanılmayan sistemlerde iç ağdaki bu sunucular hem yerel hem de dış ağdan gelen saldırılara açıktır. Ağa erişim, tek güvenlik duvarı ile sağlanır. Güvenlik duvarı, dışarıdan gelen istekleri belirli kural ve politikalar gereği iç ağa aktarmak veya engellemekle görevlidir. Normal şartlarda bu tip mimarilere sahip topolojilerde dış ağdan iç ağa olan erişimlere izin verilmemesi gerekir (Görsel 2.2).



Görsel 2.2: Tek güvenlik duvarı ile gerçeklenmiş ağ mimarileri modeli

- **İki veya Daha Fazla DMZ Bölgesine Ayrılmış Ağ Mimarileri Modeli:** Bir yönlendirici ve bir güvenlik duvarının bulunduğu ağ mimarileridir. Çeşitli güvenlik seviyelerinin konumlandırıldığı ağlarda bu tarz mimariler kullanılır. Kullanılan güvenlik duvarının üç tane arayüzü bulunur. Bu arayüzler geniş alan ağı (WAN), iç ağ (LAN) ve DMZ ağlarını birbirine bağlar. İç ağ, güvenlik seviyesi en yüksek olan bölümdür. İç ağda sunucu cihazlar, kullanıcı bilgisayarları, etki alanları, alan adları gibi kritik öneme sahip

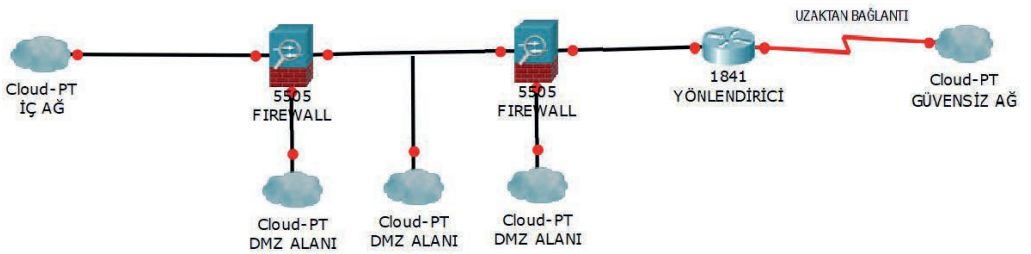
sistemler bulunur. DMZ ağı ise hem iç ağ bölümüne hem de dış ağ bölümüne hizmet veren sunucu ve servislerden oluşur. Güvenlik duvarı, farklı seviyelerdeki ağlarda yer alan trafiği düzenleyerek ağın yüksek güvenlikli ve performanslı çalışmasını sağlar (Görsel 2.3).



Görsel 2.3: İki veya daha fazla DMZ bölgesine ayrılmış ağ mimarileri modeli

2.3.2.3. İki Güvenlik Duvarıyla Gerçeklenmiş Ağ Mimarileri

Ağ güvenliğinin yüksek önem arz ettiği topolojilerde iki veya daha fazla güvenlik duvarı kullanılan mimariler tercih edilir. Bu mimaride yetkisiz bir kullanıcı veya saldırganın bir güvenlik duvarını aşması durumunda diğer güvenlik duvarının devreye girmesi istenir. İki güvenlik duvarıyla gerçeklenmiş ağ mimarileri sayesinde güvenlik duvarları içine yerleştirilmiş bir arka kapı, ajan yazılımın veya güvenlik açığının tüm sistemi etkileme riski azaltılır. İki güvenlik duvarının birbirine bağlanması, güvenliği artırmanın yanında çok sayıda DMZ bölgesi oluşturmayı ve bu bölgelerin her birinde farklı güvenlik seviyesi elde etmeyi sağlar (Görsel 2.4).



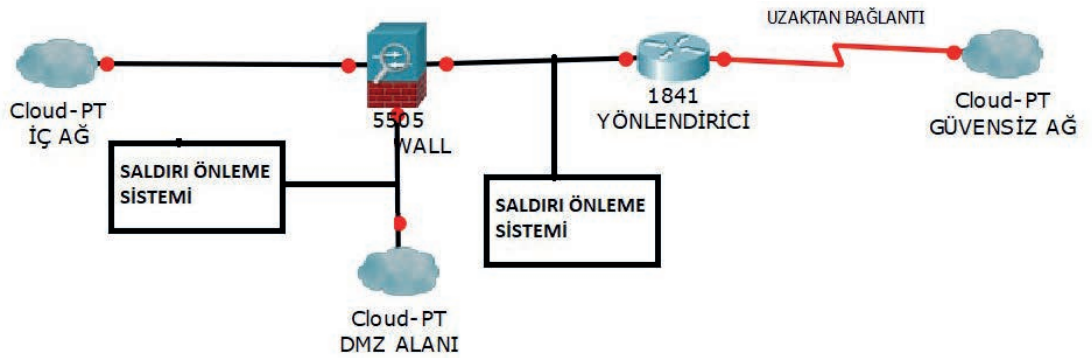
Görsel 2.4: İki güvenlik duvarı ile gerçeklenmiş ağ mimarileri

2.3.2.4. Saldırı Tespit ve Önleme Sistemleriyle Gerçeklenmiş Ağ Mimarileri

Ağa yapılan saldırıların anlaşılabilmesi için saldırı tespit sistemleri; yönlendirici ve güvenlik duvarı arasında, güvenlik duvarı ve iç ağ arasında veya birden çok arayüzü dinleyecek şekilde konumlandırılabilir. Özellikle hassas bilgi işleyen, saklayan bilgi sistem organizasyonlarında gerek iç ağın gerekse DMZ ağ segmentinin maruz kalacağı saldırıların tespiti önemlidir. Bu durumda her bir ağ segmentine veya kritik olarak önceliklendirilmiş ağ segmentlerine yapılacak saldırıların tespiti gereklidir. Bazı ağ tabanlı saldırı tespit cihazları birden fazla arayüze sahiptir. Bu türde saldırı tespit sistemleri birden fazla ağa bağlanıp ağdan geçen trafik incelenebilir. Aynı iş, birden fazla tek arayüze sahip saldırı tespit sistemleriyle de

gerçekleştirilebilir. Güvenlik duvarının arkasındaki bir ağa DMZ ağı gibi bağlanan arayüz sadece ilgili ağa gelen saldırıları tespit edebilir. Birden fazla ağ segmentine kurulacak saldırı tespit sistemleri, olası iç saldırılara (bilinçli/bilinçsiz) karşı farkındalık sağlar.

Saldırı önleme sistemi, kendine gelen paketleri inceleyerek saldırı varsa tespit eder ve gerekli uyarıları verir. Bu türde çalışan cihazların önemli bir özelliği, herhangi bir şekilde elektrik kesilmesi veya cihazın arızalanması durumunda giriş ve çıkışı kısa devre ederek ağın iletişimini durdurmamasıdır. Saldırı önleme sistemi, güvenlik duvarı ile yönlendirici arasında konumlandırılabileceği gibi dışarıdan bakıldığında güvenlik duvarından sonra da konumlandırılabilir (Görsel 2.5).



Görsel 2.5: Saldırı önleme sisteminin konumlandırılması

2.3.2.5. Sanal Özel Ağ Cihazlarıyla Gerçeklenmiş Ağ Mimarileri

VPN cihazları genellikle güvenlik duvarı ve iç ağ arasında yer alır. VPN cihazları veriler üzerinde güvenlik sağlamasına karşın kendini TCP/IP saldırılarına karşı koruyamaz. Bu yüzden güvenlik duvarının arkasında konumlandırılır. DMZ segmenti ile güvensiz ağ arasında gerçekleşecek trafik, VPN cihazı kontrolünde değildir. Tüm ağ segmentlerine ait trafiğin VPN tarafından kontrol edilmesi, gizlilik ve kimlik doğrulama işlemlerinin yapılmasının gerekli olduğu durumlarda tercih edilebilir.

2.4. AĞ SALDIRI TÜRLERİ

Günümüzde siber saldırılar, bireysel kullanıcılardan kurumsal kullanıcılara kadar bütün dijital dünyayı hedef alan bir tehdittir. Her geçen gün siber tehditlerin sayısında ve türlerinde bir artış meydana gelir. Ağ saldırıları; yerel ağdaki istemci ve sunucu bilgisayarları ile ağ cihazlarını devre dışı bırakmak, verileri ele geçirmek, verilerin erişilebilirliğini ortadan kaldırmak veya daha farklı istenmeyen bir faaliyet gerçekleştirmek için ağa yetkisiz erişim sağlama saldırılarıdır. Ağ saldırıları, **pasif** ve **aktif** olmak üzere ikiye ayrılır. Pasif ağ saldırılarında saldırganlar, bir ağa yetkisiz şekilde girerek o ağı izleyebilir ve verileri görüntüleyebilir, ele geçirebilir fakat silemez veya verilerin erişilebilirliğini engelleyemez. Aktif ağ saldırılarında ise yetkisiz erişim sağlayan saldırgan, ağdaki veri bütünlüğünü bozar ve ağın işleyişine zarar verir. Saldırının bir ağ saldırısı olabilmesi için network sistemine dâhil olması ve ardından çeşitli saldırı yöntemleri ile ağın güvenliğini tehlikeye atması gerekir.

Saldırganlar; bilgi hırsızlığı, kimlik hırsızlığı, verileri elde etme, hizmeti yavaşlatma, durdurma gibi çeşitli amaçlar doğrultusunda bilgisayar ağlarına saldırılar gerçekleştirirler. Günümüz dijital dünyasında bilgisayar ve haberleşme ağlarına yapılacak saldırılarda birçok yöntem kullanılabilir. Belli protokol ve standartlara dayanan ağ haberleşmesi, bu protokol ve standartların manipüle edilmesine dayanan birçok saldırı yöntemini de beraberinde getirmiştir. Ağ saldırısı türlerinden bazıları şunlardır:

- **Parola Kırma Saldırıları:** Saldırganın bilgisayar ağına yetkisiz erişim sağlamasıdır. Zayıf parola kullanımları, hatalı yapılandırmalar veya sosyal mühendislik sonucu ele geçirilen şifreler, bu saldırıların temelini oluşturur. Parola kırma saldırıları, çevrimiçi ve çevrimdışı saldırılar olmak üzere iki şekilde gerçekleştirilir. Brute Force (Kaba Kuvvet) saldırıları, çevrimiçi saldırı yöntemlerinden biridir. Kaba kuvvet saldırıları, geçerli parolayı bulmak adına bilinen veya özel araçlarla hazırlanmış listeleri hedef sistemde deneme yanılma yöntemiyle bulmayı amaçlar. Hash Crack ve Rainbow Table saldırıları ise pasif şifre kırma yöntemlerine örnek olarak verilir. Bu saldırılar, sistemde tutulan şifreyle ilgili hash bilgilerini elde etmeye yöneliktir.

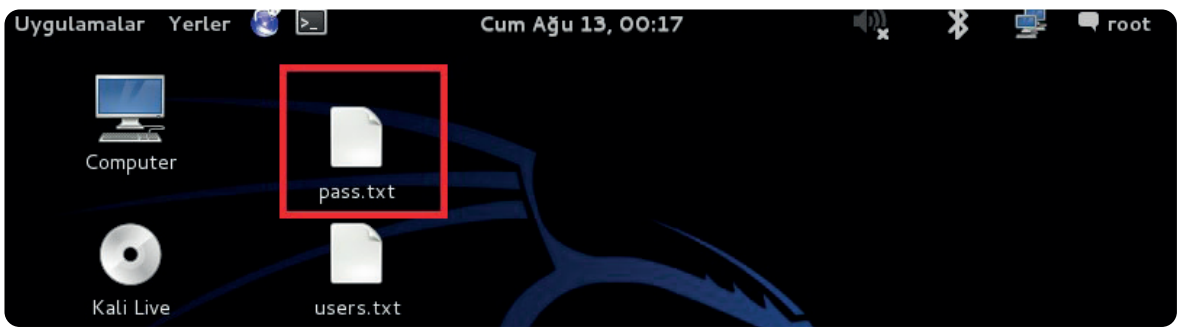
2. Uygulama

İşlem adımlarına göre **hydra** aracını kullanarak 192.168.138.128 IP adresine sahip cihazın SSH servisinin kullanıcı adını (root) ve parolasını (1234) elde etmek için kaba kuvvet saldırısını gerçekleştiriniz.

1. Adım: Kali Linux işletim sisteminde hydra aracını Uygulamalar, Kali Linux, Password Attacks, Online Attacks, hydra yolunu kullanarak açınız.

2. Adım: Yazı hazırlama editörünü (vim, nano vb.) kullanarak parola saldırısı için ihtiyaç duyulan kelime listesini, içinde parolalar ve kullanıcı adları olacak şekilde iki farklı dosya hâlinde hazırlayınız ve kullanıcı adlarını "**users.txt**", parolaları "**pass.txt**" olarak masaüstüne kaydediniz (Görsel 2.6).

Dosya İçerikleri	
user.txt	**pass.txt**
admin	1234
user	12345
kullanici	123456
siber	4321
bilisim	5432
root	654321



Görsel 2.6: pass.txt dosyasının hazırlanması

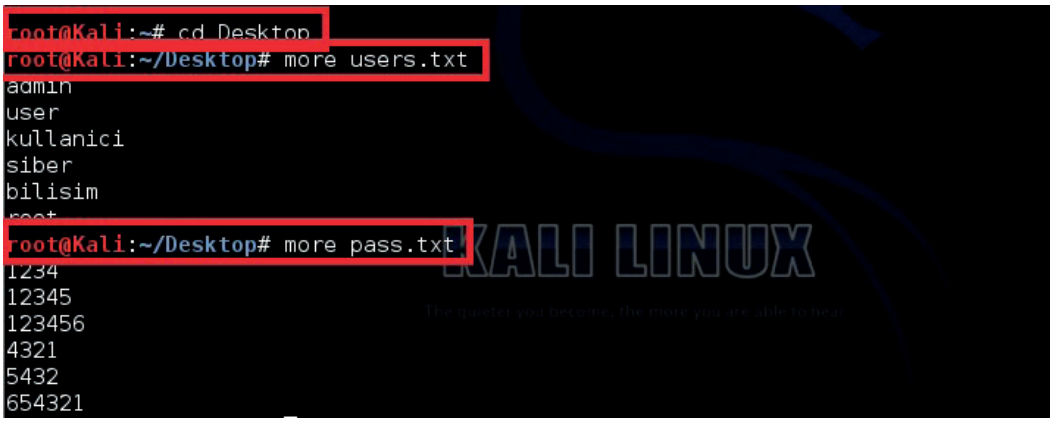
Not Uygulamada SSH servislerinin kullanılan işletim sistemlerinde açık olması gerekmektedir. İşlemler SSH servisinin açık olduğu farzedilerek yapılmıştır.

3. Adım: Kali işletim sistemi terminalini kullanarak komutları giriniz ve kelime listesinin doğruluğunu kontrol ediniz (Görsel 2.7).

root@Kali: cd Desktop

root@Kali:/Desktop more users.txt

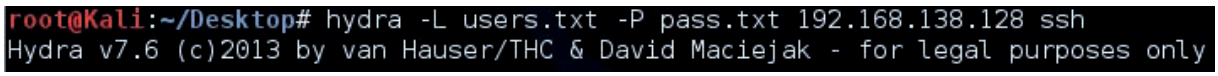
root@Kali:/Desktop more pass.txt



Görsel 2.7: Kelime listesinin doğruluğunun kontrolü

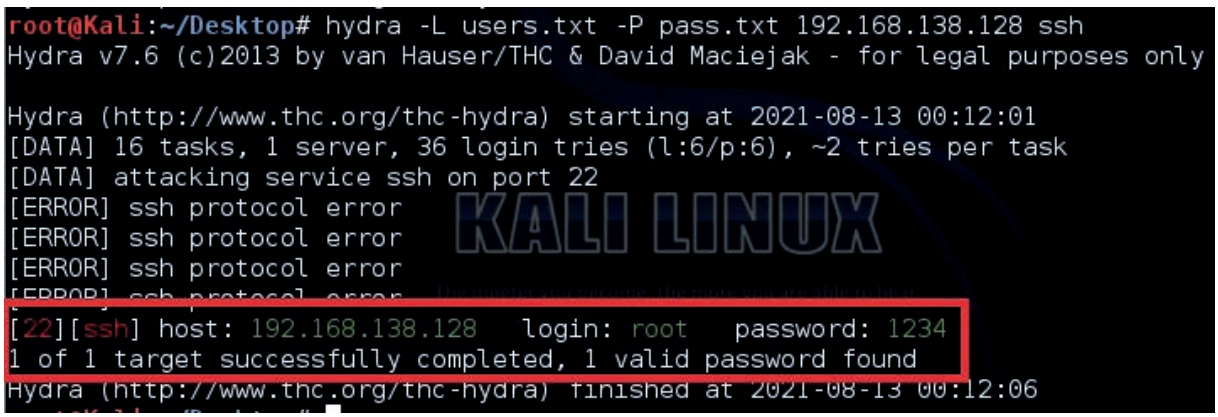
4. Adım: Kelime listesinin kontrolünü sağladıktan sonra yapılacak saldırı ile ilgili komutu hydra aracına giriniz (Görsel 2.8).

root@Kali:/Desktop hydra -L users.txt –P pass.txt 192.168.138.128 ssh



Görsel 2.8: Hydra komut yazımı

5. Adım: Saldırı sonucunda kelime listesindeki doğru kullanıcı adı ve şifresi eşleştiğinde saldırı başarılı olacaktır. Saldırı sonucunu görüntüleyiniz (Görsel 2.9).



Görsel 2.9: Hydra komut çıktısı

Sıra Sizde

Hydra komut satırı ile yapılan kaba kuvvet saldırısını hydra-gtk aracını kullanarak grafik arayüzde gerçekleştiriniz.

- **Hizmet Reddi Saldırıları (DoS-DDoS Saldırıları):** Hizmet reddi saldırıları (DoS); bir hedefe yönelik yapılan, verilen hizmeti engelleme veya kullanıcıların sisteme erişmesini engelleme amacı içeren saldırıların genel ismidir. Bütün sistemler, belli bir kapasiteyle çalışır. Bu kapasitenin üzerinde bir istek gelirse sistemler hizmet veremez hâle gelir. Sistemin verdiği hizmetler, bu saldırılar sonucunda tamamen çöker. DoS saldırılarının oluşturduğu bu aksaklıklar, bireysel ve kurumsal sistemlerde maliyet ve itibar açısından büyük sıkıntılar oluşturur. Bu saldırıları gerçekleştirmek için çok fazla teknik bilgiye gerek duyulmaz. Bu açıdan bakıldığında DoS saldırıları büyük bir risk faktörü oluşturur. Dağıtık hizmet reddi saldırıları (DDos) ise hizmet sunan sistemlerin kaldırabileceği kapasitenin çok üstünde istekler göndererek sistem kaynaklarını tüketmeyi amaçlayan, böylelikle verilen hizmetin aksamasına sebep olan saldırılardır. Bu saldırı türünde zombi adı verilen birçok istemci bilgisayar kullanılır. Zombi bilgisayarlar, sistemlerine yüklenen zararlı yazılımlar ile haberleri dahi olmadan DDos saldırılarına katılan bilgisayarlardır. Zombi bilgisayarların sayısı bazen yüz binlerce olabilir ve yapılan saldırıların önüne geçmede bir hayli zorluk yaşanabilir. İnternet kullanan cihazların (IoT) sayısındaki artış, bu saldırıların popülerliğini de artırır. Bu tarz saldırılar, hedef sisteme büyük veri paketleri gönderilerek veya karmaşık SQL sorguları kullanılarak gerçekleştirilir. Günümüzde DDoS saldırılarının önüne geçmek için birçok cihaz ve yöntem kullanılır. Yetkili ve yetkisiz kullanıcıları ayırt etmek, güvenlik sistemleri için kolay bir işlem değildir. Yönlendirici cihazlarda ICMP ve SYN paketlerini sınırlamak, erişim listeleri oluşturmak, bu saldırılardan korunmak için alınan önlemlerden bazılarıdır.

- **SQL Injection Saldırıları:** Veri tabanında SQL sorgularına yetkisiz bir şekilde müdahale edilerek SQL söz dizimi hatalarından kaynaklanan veya özel karakter kullanılarak yapılacak sorgularla sistemin kandırılmasına dayanan saldırılardır. SQL Injection saldırıları günümüzde OWASP TOP 10 içinde yer alan en tehlikeli saldırı türlerinden biridir. Yetkisiz kullanıcılar; bu saldırı ile veri tabanındaki bilgilere ulaşıp silebilir, değiştirebilir veya bu yöntem ile daha farklı saldırılara bir araç olarak kullanabilir, iç ağda uzun süre fark edilmeyecek şekilde ağı izleyebilir, haritalandırabilir. SQL Injection atak türleri, **In-Band, Blind** ve **Out of Band** ana başlıklarında gruplandırılabilir. Klasik Injection tekniği olarak isimlendirilen In-Band (Bant İçi) SQL Injection, HTTP Post ve Get isteklerinin aynı iletişim yolu üzerinden gönderilip alındığı atak türüdür. Saldırgan SQL Injection atağını başlatmak, devam ettirmek ve durdurmak için aynı kanal üzerinden iletişim kurar. Bu atağın **Error Based** ve **Union Based** olmak üzere iki yaygın çeşidi vardır. **Blind SQL Injection**, saldırgan ile hedef uygulama arasında gerçek herhangi bir veri alışverişinin yapılmadığı bir atak çeşididir. Saldırgan, sunucuya verileri gönderir ve veri tabanının yapısı hakkında daha fazla bilgiye ulaşmak için sunucunun yanıtını ve davranışlarını gözlemler. Saldırgan, sorguların sonucunu göremez veya rastgele bir sorgu göndererek deneme yanılma yoluyla gelen cevaba göre yeni bir yöntem oluşturur. Bu nedenle bu atak türü, **Kör (Blind) SQL Injection** olarak da isimlendirilir. Out of Band (Bant Dışı) SQL Injection da ise saldırgan, bilgileri toplamak ve sonuçları görmek için aynı iletişim kanalını kullanmaz. Çok yaygın bir teknik değildir ve veri tabanı sunucusunda aktif çalışan hizmetlere bağlı olarak gerçekleştirilebilir.

SQL Injection saldırılarından korunmak için şu yöntemler kullanılmalıdır:

- Kullanıcı girişlerini mutlaka doğrulamak gerekir. En çok atak yapılan yer, kullanıcı bilgileri ile giriş yapılan yerlerdir. Bilgilerin tek tek kontrol edilerek içeri alınması sağlanmalıdır.
- Özel karakter kullanımı kontrol edilmeli ve sınırlandırılmalıdır.
- Veri tabanına sorguları yazmak için değişken kullanmak ve parametreli sorgular ile hazırlanan ifadelerle daha korumalı bir veri çekme işlemi sağlanabilir. Böylelikle kullanıcı girişi ile kod arasında bir ayrım yapılır.
- Güncel veri tabanı yamalarını kurmak ve işletim sistemini güncel tutmak her zaman en önemli tedbirlerden biridir.
- Gereksiz deyimler varsa kullanımı için ayrıcalıklı erişim yetki seviyeleri düzenlenmelidir. Örneğin bir web sitesinde sadece SELECT sorguları çalışacaksa diğer deyimlerin (UPDATE, INSERT, DELETE vb.) kullanımı kısıtlanmalıdır. Ayrıca veri tabanına gerektiğinde yönetici düzeyinde erişim sağlanmalıdır. Bir hiyerarşi oluşturularak sisteme giren kişinin yetkileri en aza indirilmelidir.
- Verileri korumak için şifreleme algoritmaları kullanılmalıdır.
- Hata mesajları düzenlenerek gereğinden fazla açıklama yapılmamalıdır.

- **Ortadaki Adam Saldırıları (Man in The Middle-MITM):** Yerel ağda veya uzak ağdan iletişim hâlindeki iki bilgisayarın arasına girerek ağı dinleyen, iletişimin üzerinden geçmesini sağlayarak verileri okuyan veya verileri manipüle ederek iletişimin zarar görmesini sağlayan bir saldırı türüdür. Bu saldırı ile parola, kişisel veriler, kredi kartı bilgileri gibi maddi veya manevi olarak zarar verilecek kritik bilgiler saldırganın eline geçebilir. İletişim protokolleri güvenli değilse veya saldırganlar bu güvenliği aşmanın bir yolunu bulursa iletilen verileri çalabilir, kullanıcı kimlik bilgilerini alabilir ve kullanıcıların oturumunu ele geçirebilir. Çok farklı türde MITM saldırısı bulunur. MITM saldırılarından bazıları şunlardır:
  - ARP Spoofing (ARP sahtekârlığı)
  - DNS Spoofing (DNS sahtekârlığı)
  - DHCP Spoofing (DHCP sahtekârlığı)
  - Rogue Access Point (Sahte erişim noktası)
  - Port Stealing (Port hırsızlığı)
  - STP Mangling (STP bozma ve yönetme)
  - ICMP Redirection (ICMP yönlendirmesi)

2.5. İŞLETİM SİSTEMİ GÜVENLİK İLKELERİ

Bilgisayar sistemlerinin temel işletim altyapısını oluşturan işletim sistemleri, sistem üzerinde çalışan diğer uygulama ve servislere de güvenlik altyapısı sağladığı için işletim sistemlerinin güvenliği, bütün sistemin güvenliği için büyük rol oynar. İşletim sisteminde bulunan bir zafiyete gerekli önlemler alınmadığı ve güvenlik ilkeleri belirlenmediği sürece sistem her zaman risk altındadır. İşletim sistemi varsayılan güvenlik uygulamalarını kullandığı sürece bu ilkelerin herkes tarafından bilindiği unutulmamalıdır. Sistem yöneticileri, güvenlik politikalarını belirleyerek işletim sistemi üzerinde sıkılaştırma yapmalıdır.

2.5.1. İşletim Sistemi Güvenlik Yapılandırmaları

İşletim sistemi sıkılaştırmalarında güvenlik ilkelerini belirlemek en önemli adımdır. Güvenlik ilkeleri belirlendikten sonra işletim sistemine yönelik gerekli kontrollerin yapılması ve güvenlik yapılandırmalarının uygulanması gerekir. Bu yapılandırmalara kullanıcı gruplarını, hesaplarını ve parolalarını sınırlandırmak, gereksiz hizmetleri devre dışı bırakmak, kayıt defterini yapılandırmak, güvenlik duvarı etkinleştirmeleri, tarayıcı sıkılaştırmaları örnek olarak verilebilir.

- **İşletim Sistemi Kullanıcı Hesapları:** İşletim sistemi kullanıcı ve grupları, varsayılan olarak kurulumla birlikte gelir. Yetkisiz erişim sağlayacak kişinin ilk deneyeceği işlem, varsayılan kullanıcı bilgileri ile sisteme giriştir. Bu noktada alınacak önlemlerin ilki, varsayılan kullanıcı isimlerini değiştirmek veya devre dışı bırakmaktır. İşletim sistemi kurulumuyla beraber tüm izin ve haklara sahip yönetici hesabı, varsayılan olarak kurulu gelir. Yönetici hesapları, saldırganlar için ilk hedeflenen kullanıcı hesaplarıdır. Farklı bir kullanıcı adı ile sistem yönetici hesabını oluşturmak alınacak önlemlerin başında gelir. Ayrıca yeni kullanıcılara minimum yetkiler verilerek hesap oluşturmak, saldırıların önüne geçmek için etkili bir yöntemdir. İşletim sisteminde bulunan veri tabanı isimlerini (ör. SQL Server), IUSR_Machine name ismini, ASP.net gibi web uygulamaları için çalışan servislerin varsayılan isimlerini değiştirmek, yapılacak özel saldırıların önüne geçmeyi sağlar.
- **Parola Politikaları:** İşletim sistemlerinde parola güvenliği, üzerinde en çok durulan yapılandırmalardan biridir. Varsayılan işletim sistemi parolaları, sistem üzerinde güvenlik açığı oluşturur. Şifre kullanım süresi, şifre deneme sayısı, şifre uzunluğu gibi yapılandırmalar, güvenlik ilkelerinde açıkça değiştirilerek uygulanmalıdır.

Ulusal Güvenlik Ajansı (National Security Agency-NSA), sistemler için güvenlik politikalarını ve standartlarını belirleyen bir kuruluştur. Şifrelerle ilgili NSA standartları Tablo 2.1'de verilmiştir.

Tablo 2.1: NSA Standartları

NSA Politikası	Süre
Maksimum şifre süresi	42 gün
Minimum şifre süresi	2 gün
Minimum şifre uzunluğu	12 karakter
Parolaların karmaşıklık seviyesi	Evet
Şifre geçmişini zorlama	5 şifre

Ağ ortamının gereksinimlerine göre şifre politikaları belirlenmelidir. Güvenlik önlemlerinin çok karışık olması kullanıcıların şifreleri kolaylıkla unutmasına sebebiyet vereceği için hem güvenli hem de karmaşıklığı belli bir sisteme dayanan şifreler üretilmelidir.

3. Uygulama

İşlem adımlarına göre minimum parola uzunluğunu 15, parola geçmişini 5, şifre karmaşıklık sistemini etkin olarak yapılandırınız ve yerel hesapta boş parola kullanımını sınırlayınız.

1. Adım: Computer Configuration > Policies > Windows Settings > Security Settings > Account Policies > Password Policy yolunu kullanarak parola politikalarını uygulayınız.

- Enforce password history > 5
- Maximum password age > 60
- Minimum password age > 1
- Minimum password length > 15
- Password must meet complexity requirements > Enabled (Varsayılan)
- Store passwords using reversible encryption > Disabled (Varsayılan)

2. Adım: Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > Security Options yolunu kullanarak yerel hesapta boş parola kullanımını sınırlayınız.

- **Accounts:** Limit local account use of blank passwords to console logon only: Enabled
- **Hesap Kilitleme Politikaları:** Yerel güvenlik ilkeleri (local group policy) sadece şifre politikalarını belirleyen bir yapılandırma alanı değildir. Yerel güvenlik ilkeleri ile hesap kilitleme yapılandırmaları ve sıkılaştırmalarını yapmak da mümkündür. Bu politikalar, bir kullanıcının hesap kilitlenmeden önce kaç kez oturum açmaya çalışabileceğini ve hesabın ne kadar süreyle kilitleneceğini belirler. İşletim sistemi varsayılan standartları, hesap kilitleme politikası için yeterli değildir ve hesap kilitleme politikasında değişiklik yapmak gerekir.

Hesap kilitleme politikası	NSA Standardı
Hesap kilitleme süresi	15 saat
Hesap kilitleme sınır noktası	3 deneme
Kullanım sonrası hesap kilitleme süresi	30 dakika

Parola saldırılarına karşı alınabilecek önlemlerden biri de hesapların kilitlenmesini sağlamaktır. Bu ayarlar, uygun bir şekilde yapılandırılmadığında hizmet dışı bırakma (DoS ve DDoS) saldırıları gerçekleşebileceği için dikkatli yapılmalıdır. Kullanıcı hesaplarının kilitlenmesinin temel sebepleri şu şekilde sıralanabilir:

- Bağlantısı oluşturulmuş (map edilmiş) ağ paylaşımları
- runAs ile çalışan kısayollar
- Service Account tanımları
- Zamanlanmış görevler (Schedule Task)
- Farklı sunucu, bilgisayar veya uygulamalar üzerinde çalışan işlemler, servisler
- Kullanıcı kimlik denetimi veya doğrulaması yapan merkezî programlar (AD doğrulaması, uygulama katmanı denetimi vb.)

- Activesync ile çalışan mobil cihazlar
- Kullanıcı bilgisayarına bulaşmış zararlı yazılımlar

Not Hesap kilitleme politikaları için gerekli ayarlar, grup ilkesi nesnesi ile şu şekilde gerçekleştirilir:

1. Computer Configuration > Policies > Windows Settings > Security Settings > Account Policies > Account Lockout Policy
2. Account lockout duration > 15 (veya daha fazla)
3. Account lockout threshold > 10 (veya daha az)
4. Reset account lockout counter after > 15 (veya daha fazla)

- **Kayıt Defteri Yapılandırmaları:** Kayıt defteri, işletim sisteminde ayarları ve seçenekleri depolamak için kullanılan bir veri tabanıdır. Bu veri tabanı; donanım, yazılım, kullanıcı ve gruplar ile ilgili bilgileri ve ayarları bünyesinde bulundurur. Kullanıcıların yaptığı her ayarlama ve değişiklik bu veri tabanına kaydolur. Kayıt defterinde yapılacak bir ayar ve değişiklik, sistemde güvenlik zafiyetine veya sistemin bozulmasına sebebiyet verebilir. Kayıt defterinde değişiklik yapılmadan önce kayıt defterinin yedeğinin oluşturulması, karşılaşılabilecek sorunlar için bir çözümdür. Windows tabanlı işletim sistemlerinde kayıt defterine ulaşabilmek için komut satırına regedit.exe yazılmalıdır. Kayıt defterine ulaşıldığında ana bölümler şu şekilde sıralanabilir:
- **HKEY_CLASSES_ROOT:** Tüm dosya ilişkilendirme türlerini, OLE (Nesne bağlama ve nesne gömme) bilgilerini ve kısayol verilerini içerir.
- **HKEY_CURRENT_USER:** HKEY_USERS adlı bilgisayarda şu anda oturum açan kullanıcı için uygun bölüme bağlanır.
- **HKEY_LOCAL_MACHINE:** Belirli bir bilgisayardaki donanım, yazılım ve diğer tercihler hakkında bilgisayara özgü bilgiler içerir.
- **HKEY_USERS:** Bilgisayarın her kullanıcısı için ayrı ayrı tercihler içerir.
- **HKEY_CURRENT_CONFIG:** HKEY_LOCAL_MACHINE ürününün geçerli donanım yapılandırmasına uygun bölümüne bağlanır.

Not Kayıt defterindeki belirli bir girişe **anahtar** denir. Anahtar, sistemin belirli bir yönüne ilişkin ayarları içeren bir girdidir. Kayıt defteri değiştirilirse belirlenen değerin ayarları da güncellenir.

Boş oturumlar, bilgisayardaki çeşitli paylaşımlardan yararlanılabilecek önemli bir zayıflıktır. Boş oturum, Windows işletim sisteminin anonim bağlantılarını belirlemenin yoludur. Herhangi bir sunucuya anonim bağlantı yapılmasına her izin verildiğinde önemli güvenlik riskleri doğar.

Sıra Sizde

Anahtar yolu **HKLM\SYSTEM\CurrentControlSet\Control\Lsa** olan anonim erişim kayıt defteri ayarının, anonim kullanıcıların etki alanı kullanıcı adlarını listelemelerini ve paylaşım adlarını numaralandırmalarını sağlayan yapılandırmayı kapatınız.

- **Güvenlik Şablonları:** İşletim sistemi sıkılaştırmasını basitleştirmenin en iyi yolu, güvenlik şablonlarını kullanmaktır. Güvenlik şablonu, bir veya birden fazla bilgisayarı kontrol edebilen yüzlerce olası ayar içerir. Güvenlik şablonları; kullanıcı hakları, izinler, parola ilkeleri gibi alanları denetleyebilir ve yöneticilerin, bu ayarları grup ilkesi nesneleri aracılığıyla merkezî olarak dağıtmasını sağlar. Güvenlik şablonları, hedef bilgisayarda tüm güvenlik ayarlarını içerecek şekilde özelleştirilebilir. Windows'a bir dizi güvenlik şablonu yerleştirilmiştir. Bu şablonlar; etki alanı denetleyicileri, sunucular ve iş istasyonları için kategorilere ayrılmıştır. Bu güvenlik şablonlarının üretici firma tarafından tasarlanmış varsayılan ayarları vardır. Bu şablonların tümü C:\Windows\Security\Templates klasöründe bulunur. Bu klasördeki güvenlik şablonlarının kısmi bir listesi şu şekildedir:

  - **Hisecdc.inf:** Etki alanı denetleyicileriyle güvenliği ve iletişimi artırmak için tasarlanmıştır.
  - **Hisecws.inf:** İstemci bilgisayarlar, üye sunucular için güvenliği ve iletişimi artırmak üzere tasarlanmıştır.
  - **Securedc.inf:** Etki alanı denetleyicileriyle güvenliği ve iletişimi artırmak için tasarlanmıştır ancak yüksek DC (Domain Controller) güvenlik şablonu düzeyinde değildir.
  - **Security.inf:** Yeni yüklenen bir bilgisayarın varsayılan güvenlik ayarlarını yeniden uygulamak için tasarlanmıştır. Yanlış yapılandırılmış bir sistemi varsayılan yapılandırmaya döndürmek için de kullanılabilir.

- **İşletim Sistemi Sürümünü Güncelleme ve Güvenlik Eklentileri:** İşletim sistemlerinin yamalarının yüklenmesi, ek sıkılaştırma önlemlerinin alınmasının yanında en güncel sürümlerinin kullanılması, gelişmiş güvenlik önlemlerinden yararlanmayı sağlar. Örneğin mevcut işletim sisteminde yeterli yetkiye sahip olunması durumunda bellek üzerinde kayıtlı parolalar elde edilebilir. Bu parolaların elde edilmesi sonucunda web sayfası ile erişim sağlanan oturumlar gibi parola özetini kabul etmeyen oturumlar açılabilir. Sorunların çözümü için işletim sisteminin resmî sitesinden güncel sürümleri ve yamaları takip ederek gerekli güncelleme yapılmalıdır. Group Policy ile Windows Update ayarlarında değişiklik yapılabilir. Windows Update için sıkılaştırma ayarları vardır. Bu ayarlar, Computer Configuration > Administrative Templates > Windows Components > Windows Update üzerinden yapılandırılabilir. Bütün bu yapılandırmalar ve sıkılaştırmaların yanında şu alanlarda da sıkılaştırmalar yapılmalıdır:

  - Haricî aygıt kullanımı
  - Zararlı yazılım koruması
  - Yönetimsel konsollara erişim
  - SMB iletişimi, ağ üzerinden erişim
  - LLNMR ve NetBIOS iletişimi
  - RDP bağlantısı
  - PowerShell kullanımı güvenliği

Sıra Sizde

Haricî cihaz takılmasını, yerel güvenlik ilkeleri ayarlamalarını kullanarak kapatınız.

2.6. SUNUCU SİSTEMLERİ GÜVENLİK İLKELERİ

Sunucu sistemleri güvenlik ilkeleri, kurulum aşamasından başlayarak büyük bir hassasiyet ile uygulanmalıdır. Sunucu sistemlerin verdiği hizmetler düşünüldüğünde oluşabilecek bir siber saldırı, sunucunun hizmet verdiği kişilerden kurulduğu ağa kadar geri dönüşü olmayan kayıplar yaşanmasına neden olabilir.

2.6.1. Sunucu İşletim Sistemi Güvenlik Yapılandırmaları

Kurulum: Sunucu işletim sisteminin kurulum aşamasında gerekli güvenlik tedbirlerinden önce saldırılardan korunmak amacıyla çeşitli önlemlerin alınması gerekir. Bu önlemler şu şekilde sıralanabilir:

- Sunucunun kurulumu ağdan ayrı veya izole bir ağda yapılmalıdır.
- Kurulum sırasında yalnızca doğrulanmış medya aygıtları kullanılmalıdır.
- Gerekli ek sürücüler yalnızca resmî indirme konumlarından indirilmelidir.
- İndirilen dosyaların bütünlüğü doğrulanmalıdır.
- İndirilen sürücüler, kurulum öncesinde güncellenmiş bir antivirüs yazılımı aracılığıyla taranmalıdır.
- Sürücüler yalnızca kurulum için ayrılmış temiz bir ortam aracılığıyla kuruluma kopyalanmalıdır.
- Tüm dosya sistemleri, NTFS veya başka bir güvenlik destekli dosya sistemi olmalıdır.
- Kurulumda minimum sayıda gerekli hizmet kurulmalıdır.
- Kötü amaçlı yazılımdan koruma sistemleri veya güvenlik duvarları en kısa zamanda kurulmalı ve yapılandırılmalıdır.

Not Güvenlik güçlendirmesi tamamlandıktan sonra işletim sistemi sunucularından en son hizmet paketlerini ve düzeltmeleri almak için sunucular internete bağlanabilir.

Sunucu rolleri kurulmadan önce şu sıkılaştırmaların yapılması gerekir:

- Dosya sistemi izinleri gereken şekilde gözden geçirilmeli ve etkinleştirilmelidir. Her veri klasörüne yalnızca bu bilgilere erişim için gerekli izin düzeylerine sahip kişilerin erişimine izin verilmelidir. Bu işlem, uygun kullanıcı grupları kullanılarak yapılabilir. Okuma ve yazma izinleri ayrı ayrı ele alınmalıdır.
- Tek bir saat ve tarih kaynağının kullanılması, olayların birbiriyle ilişkilendirilmesi açısından son derece önemlidir. Tüm sunucular bir ağ zaman protokolü (Network Time Protocol-NTP) sunucusu ile düzgün şekilde senkronize edilmelidir. Bunun yerel bir sunucu olması önerilir.
- Kötü amaçlı yazılım korumaları, antivirüs, anti-spyware uygulamalarını içermelidir. İş birliği ortamında politikanın uygun şekilde sürdürülmesine olanak tanıyacağı ve uç noktalar üzerinde daha ayrıntılı bir kontrole sahip olacağı için merkezî şekilde yönetilen kötü amaçlı yazılımlara karşı koruma önerilir.

Kullanıcı Hesabı ve Erişim Hakları Politikaları: Minimum parola uzunluğu ayarlanmalıdır. Parola saldırısı türleri arasında sözlük saldırıları ve kaba kuvvet saldırıları bulunur. Ayrıca saldırganlar bazen hesapları ve şifreleri keşfedecek araçları kullanabilmek için hesap veri tabanını ele geçirmeye çalışırlar.

Grup ilkesi aracılığıyla parola karmaşıklığı gereksinimleri etkinleştirilmelidir. Yalnızca alfa numerik karakterler içeren parolaların herkese açık çeşitli araçlarla keşfedilmesi son derece kolaydır.

Uygun bir hesap kilitleme politikası uygulanmalıdır. Uygun bir hesap kilitleme politikasına sahip olmak önemlidir çünkü kaba kuvvete veya şifre tahmin saldırılarına karşı korunmaya yardımcı olacaktır.

Pasif kullanıcılar devre dışı bırakılmalı veya silinmelidir. Kullanılmayan veya gereksiz kullanıcı hesaplarının varlığı, her zaman suistimal edilme riski taşır. Bu nedenle gerekli bir amacı olmayan her türlü kullanıcı hesabının kaldırılması veya devre dışı bırakılması önemle tavsiye edilir.

Sistemlerde kimlerin yerel olarak oturum açabileceği ayarlanmalıdır. Bu işlem, kimlerin doğrudan konsollar aracılığıyla sistemde oturum açabileceğini belirler. Bu kullanıcı hakkı, genel olarak Yönetici gruplarıyla sınırlıdır.

Uzak masaüstünü kullanarak kimlerin oturum açabileceği ayarlanmalıdır. Bu işlem, hangi kullanıcıların veya grupların Terminal Hizmetleri istemcisi olarak oturum açma hakkına sahip olduğunu belirler. Uzak masaüstü kullanıcıları, bu kullanıcı hakkına sahiptir. Kuruluş, yardım masası stratejisinin bir parçası olarak uzaktan yardımı kullanıyorsa bir grup kullanıcısı oluşturulmalı ve doğrudan grup ilkesi aracılığıyla bu kullanıcıya atanmalıdır. Kuruluştaki yardım masası, uzaktan yardım kullanmıyorsa bu kullanıcı hakkı yalnızca Yöneticiler grubuna atanmalı veya hiçbir kullanıcı hesabının, uzak masaüstü kullanıcıları grubunun parçası olmaması için kısıtlı gruplar özelliği kullanılmalıdır.

Genel Sıkılaştırma Politikaları: Oturum açmadan önce bir uyarı mesajının görüntülenmesi, saldırganı hatalı davranışının sonuçları hakkında daha saldırı gerçekleşmeden bilgilendirerek saldırının önlenmesine yardımcı olur. Ayrıca oturum açma işlemi sırasında çalışanları uygun politika konusunda bilgilendirerek kurumsal politikanın güçlendirilmesine katkı sağlar. Konuk (Guest) hesabı durumu **devre dışı** olarak ayarlanmalıdır.

Microsoft sunucu işletim sistemini kullananlar için **üçüncü taraf SMB sunucularına şifrelenmemiş parola gönder** seçeneğinin devre dışı olarak ayarlandığından emin olunmalıdır. Etkinleştirmeye yönelik güçlü bir iş durumu olmadığı sürece bu ilke ayarının **devre dışı** bırakılması önerilir. Bu ilke ayarı etkinleştirilirse ağ genelinde şifrelenmemiş parolalara izin verilir.

Sistem başlatma ayarlarında değişiklik yapılmasını önlemek için BIOS şifresi belirlenmelidir. Kurtarma konsoluna otomatik yönetici girişi devre dışı bırakılmalıdır. Oturum açmaya gerek kalmadan sistemin kapatılmasına izin verilmemelidir.

2.6.2. Rol Tabanlı Erişim Kontrolü

Rol Tabanlı Erişim Kontrolü (Role Based Access Control-RBAC); kuruluş bünyesindeki kullanıcılara, rollerine ve sorumluluklarına uygun şekilde erişim izni vermek için kullanılan bir yapıdır. RBAC ile her bir rol için belirli sorumluluklar ve ayrıcalıklar belirlenerek kullanıcıların yalnızca sorumlu oldukları işlevler için sadece gerekli bilgilere ve kaynaklara erişebilmesi sağlanır. RBAC, beraberinde birçok avantajı getirir. Bu avantajlar şu şekilde sıralanabilir:

- **Gelişmiş Güvenlik Kontrolü:** Yetkisiz erişim ve veri ihlali riskleri, erişim kontrolü sayesinde minimuma iner.
- **Verimli Kullanım:** Roller ve alanlar sınırlandırıldığı için sistem yönetimi kolay ve verimli hâle gelir.
- **Denetim Kolaylığı:** Hangi kaynağa kimin erişiminin olacağı belirlendiği için denetimde kolaylık sağlanır.

- **Ölçeklenebilirlik:** Yeni rol ekleme ve var olan rolü değiştirme imkânı bulunur.

Çeşitli sektörlerde RBAC kullanımına örnekler vardır. Sağlık sektöründe doktorlar, hemşireler, yöneticiler ve diğer çalışanlar farklı rollerde yetkilendirilir. Böylelikle hasta gizliliği ilkesi gereğince kişisel verilerin korunması sağlanır. Ayrıca hata yapma ve saldırı riskleri de minimuma indirilir. Eğitim sektöründe okul yöneticileri, öğretmenler ve öğrenciler farklı yetki ve sorumluluklara sahiptir. Böylece sistemin verimli ve güvenli yönetimi sağlanır.

Rol tabanlı erişim kontrolünün etkili bir şekilde uygulanması için şu adımlar izlenmelidir:

- **Rolleri Tanımlama:** Kuruluştaki farklı roller tanımlanmalıdır. Bu roller tanımlanırken farklı çalışanların sorumluluklarına dair işlevleri, sorumlulukları ve erişim gereksinimleri göz önünde bulundurulmalıdır.
- **İzinleri Tanımlama:** Her rol için gereken özel izinler belirlenmelidir.
- **Kullanıcılara Rol Atama:** Kurum bünyesindeki kullanıcılara, iş pozisyonları ve sorumluluklarına göre roller atanmalıdır.
- **Rol Eşleme:** Roller ve izinler arasında ilişkiler kurulmalıdır. Her bir rolle hangi izinlerin ilişkilendirileceği tanımlanarak belirli bir role atanan kullanıcıların belirli erişim haklarına otomatik sahip olması sağlanmalıdır.
- **Gözden Geçirme ve Güncelleme:** Rollere atanan izinlerin, kuruluş bağlamında gerçekleşen değişikliklerle uyumundan emin olmak için sistem düzenli şekilde gözden geçirilmelidir. İhtiyaca göre yeni roller oluşturmak veya mevcut roller üzerinde değişiklikler gerçekleştirmek için gerekli güncellemeler yapılmalıdır.

2.7. AĞ CİHAZLARININ SIKILAŞTIRILMASI

Ağ sıkılaştırma, bir siber güvenlik altyapısındaki güvenlik açıklarını en aza indirme sürecidir. Ağ sıkılaştırma stratejileri, sistemin ağ güvenliğini optimize için kullanılan sürece rehberlik etmeye yardımcı olur.

Siber sistemlerde ağ sıkılaştırmaları şu güvenlik risklerini azaltmaya yardımcıdır:

- Ağ cihazlarının yapılandırmasından kaynaklı açıklar veya cihazda bulunan güvenlik açıkları
- Ağda çalışan fakat gerekli olmayan sistemlerin sağladığı açıklar

Bilgisayar ağları, saldırganlar için öncelikli hedeflerdir. Ağ sıkılaştırma yapılarak saldırganlara karşı ilk önlem alınmalıdır. Ağ saldırılarına karşı tam savunma mümkün olmasa da hazırlanacak stratejiler ve standartlar kapsamında ağ sıkılaştırma politikaları uygulamak, saldırının başarılı olma ihtimalini azaltır ve potansiyel zararları minimuma düşürür. İyi bir ağ sıkılaştırma politikası için şunlar kullanılmalıdır:

- Ağ haritalandırılmalı ve belgelendirilmelidir.
- Güvenlik açıkları tespit edilmeli ve düzeltilmelidir.
- Gelecekte oluşabilecek risk faktörlerine karşı koruma sağlanmalıdır.

2.7.1. Telnet ve SSH Protokolleri

Ağ cihazlarının yapılandırılması, ağ sıkılaştırmada önemli işlemlerden biridir. Ağ cihazında yapılandırmadan kaynaklı hatalar veya göz ardı edilen risk faktörleri, saldırganlar için birçok açık kapı bırakır. Telnet ve SSH, ağ cihazlarına uzaktan bağlanmak için kullanılan protokollerdir. Telnet ve SSH protokollerinde yapılacak bir yapılandırma hatası, sistemin tamamen saldırganların eline geçmesine yol açar.

2.7.1.1. Telnet Protokolü

Telnet'in açılımı, telecommunication network (iletişim ağı) demektir. Telnet protokolü, bir cihaza uzaktan bağlanmak için kullanılır. Telnet, TCP bağlantıları üzerinden karakter alışverişi gerçekleştiren sunucu-istemci (server-client) protokolüdür. Telnet, cihazlara metin tabanlı kullanılarak uzaktan bağlantı imkânı sağlar. Telnet ile cihazlara uzaktan bağlanılırken **23 numaralı port** kullanılır.

Telnet, ağ ve sunucu yöneticileri için her zaman pratik bir protokol olmuştur. Ağdaki cihazları uzaktan yönetme yeteneği, özellikle protokolün neredeyse tüm cihazlar tarafından desteklenmesi nedeniyle ideal bir uzaktan yönetim protokolü olmasını sağlar.

Telnet bağlantıları pratikte standart TCP bağlantıları olduğu için istemci, aktarım protokolü olarak TCP'ye dayanan diğer hizmetleri kullanmak veya test etmek için kullanılabilir. Örneğin istemci, basit bir istekle bir HTTP sunucusunun işlevselliğini veya bir e-posta sunucusunun durumunu kontrol edebilir. Telnet'in diğer bir avantajı, izin verildiği takdirde kontrollü bir sistemin kaynaklarına sınırsız erişim sağlamasıdır ancak Telnet'in yüksek bir güvenlik riski vardır. Telnet protokolü kullanıldığında hem bağlantı kurulumu hem de veri aktarımı şifrelenmez. Bu nedenle uzaktan erişim için gerekli oturum açma bilgileri de dâhil olmak üzere gönderilen tüm bilgiler, üçüncü şahıslar tarafından ağ izleme yazılımları aracılığı ile **Düz Metin (Clear Text)** olarak ele geçirilebilir. Bu durum, bilgisayar korsanlarının sistemi ele geçirmesinde sorun yaşamayacağı anlamına gelir. Telnet'in güvenli bir alternatifi Secure Shell'dir (SSH).

Telnet protokolünün avantajları ve dezavantajları şu şekildedir;

Avantajları	Dezavantajları
Telnet istemcisi çok yönlüdür.	Şifrelenmemiş veri transferi yapar.
Çeşitli platformlar arası kullanılabilir.	Sınırsız erişim, saldırganların işini kolaylaştırır.
Hedef kaynaklara sınırsız erişim sağlar.	Telnet aracılığı ile erişilebilir sunucu sistemler sınırlıdır.

Not

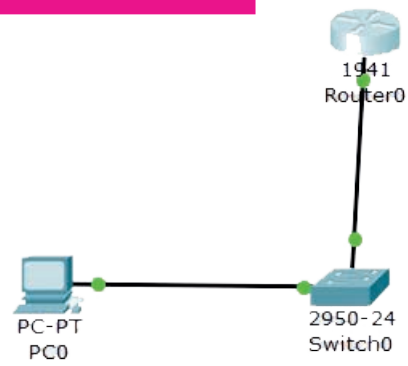
Uzak bağlantılar söz konusu olduğunda Telnet protokolü, pratik ve kullanımının kolay olması nedeniyle uzun süredir listenin başında yer alır ancak yönetilen ve iletilen verilerin güvenliğine ilişkin artan talepler, protokolü birçok senaryo için uygunsuz ve kabul edilemez hâle getirmiştir. Günümüzde internetteki uzaktan bağlantılar, genel anahtar kimlik doğrulaması sayesinde çok daha güvenli bir protokol olan şifreli SSH protokolüne dayanır. Telnet protokolü, çeşitli işletim sistemlerinde pasif olarak gelir ve aktifleştirilmesi gerekir. Ağ cihazlarında Telnet protokolünü aktif etmek için çeşitli komutlar kullanılmalıdır. Telnet kullanım şekli şöyledir:

```
telnet ip_adresi/etki_alanı_adı
```

```
telnet 192.168.1.100
```

4. Uygulama

İşlem adımlarına göre Görsel 2.10'da verilen topolojiyi, 192.168.1.0/24 ağı IP adreslerini kullanarak ağ simülasyon programında hazırlayınız ve Telnet yapılandırmasını gerçekleştiriniz.



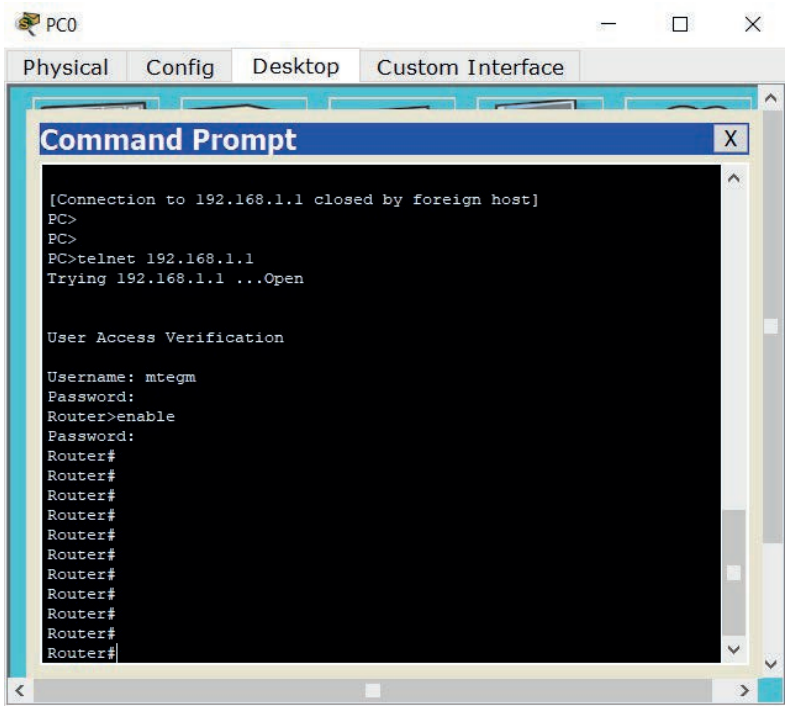
Görsel 2.10: Telnet uygulama topolojisi

1. Adım: Bilgisayar ve yönlendirici IP adres girişlerini yapınız.

2. Adım: Kodların yönlendiriciye girişini sağlayınız.

```
Router>
Router>enable
Router#configure terminal
Router(config)#enable secret meslek
Router(config)#username mtegm password meb
Router(config)#line vty 0 4
Router(config-line)#login local
```

3. Adım: Bilgisayarı kullanarak Telnet üzerinden yönlendirici cihaza bağlantı sağlayınız (Görsel 2.11).



Görsel 2.11: Telnet bağlantı işlemi

Sıra Sizde

Bilgisayarınızın Telnet istemcisini aktif ediniz ve Telnet protokolü ile yapılandırılmış bir anahtarlama cihazına bağlanınız. Wireshark yazılımını kullanarak Telnet şifresini clear text olarak Wireshark programında görüntüleyiniz.

2.7.1.2. Secure Shell Protokolü

SSH (Secure Shell), iki ağ cihazının iletişim kurmasını ve verileri paylaşmasını sağlayan bir ağ iletişim protokolüdür. SSH'ın en önemli özelliği ve Telnet protokolüne göre en önemli avantajı güvenli olmasıdır. İki ağ cihazı arasındaki iletişimin şifrelenmesi, güvenli ağlarda kullanıma daha uygun olduğunun göstergesidir. SSH genellikle uzaktaki bilgisayarlarda oturum açmak ve işlemler gerçekleştirmek için kullanılır. Veri aktarımı için de kullanılabilir. Secure Shell, güçlü parola kimlik doğrulaması ve genel anahtar kimlik doğrulamasının yanı sıra internet gibi açık bir ağ üzerinden bağlanan iki ağ cihazı arasında şifreli veri iletişimi sağlar.

SSH, güçlü şifreleme sağlamasının yanı sıra ağ yöneticileri tarafından sistemleri ve uygulamaları uzaktan yönetmek için yaygın olarak kullanılır. Bu durum, onların ağ üzerinden başka bir bilgisayarda oturum açmasına, komutları yürütmesine ve dosyaları bir bilgisayardan diğerine taşımasına olanak tanır. Bir SSH sunucusu, varsayılan olarak standart TCP bağlantı noktası 22'yi kullanır.

SSH iletişimin güvenliğini sağlarken simetrik şifreleme, asimetrik şifreleme ve hashleme yöntemlerini kullanır. SSH, simetrik şifrelemede **Diffie-Hellman Anahtar Takas Algoritması** yönteminden yararlanır. İstemci ve sunucu cihaz bağlantısı kurulurken anahtarları kendi oluşturur. Her yeni oturumda yeni bir anahtar oluşturulur. Simetrik şifrelemede birçok algoritma kullanılır. AES, 3DES, DES, RC4, Blowfish bu algoritmalardan bazılarıdır. Kaynak ve istemci, iletişimde hangi algoritmayı kullanacaklarını ortak sahip oldukları algoritmaları karşılaştırıp, en çok kullanılan algoritmayı belirleyerek seçer. Asimetrik şifreleme yönteminde hem istemci hem de kaynak açık ve özel anahtarlarını oluşturur. Açık ve özel anahtarlar sayesinde simetrik şifrelemede kullanılacak anahtar oluşturulur. Genelde kullanılan asimetrik şifreleme algoritmaları; RSA, DSA, Diffie-Hellman ve Eliptik Eğri algoritmalarıdır. Hash yönteminde ise her iki taraf da elindeki verinin hashini alarak hashleri karşılaştırır ve mesajın değiştirilip değiştirilmediğini veya doğru mesaj olup olmadığını anlar. Böylece uzaktan bağlantı sağlanır.

SSH güvenliği için sıkılaştırma yöntemleri şunlardır:

- Root ile SSH bağlantıları devre dışı bırakılmalıdır.
- SSH iki versiyonu bulunur. Yapılandırma sırasında varsayılan olarak protokol 1 kullanan SSH, daha güvenli olan versiyon 2 ile değiştirilmelidir.
- SSH kullanımında zamanlama (Timeout) değeri verilerek bağlantı açıkken ve boştayken kullanım süresi ayarlanmalıdır.
- SSH bağlantısını kullanacak kişiler sınırlandırılmalıdır. İzin verilen kullanıcılar dışındaki kişilerin bağlantısı engellenmelidir.
- Şifre denemeleri sınırlandırılmalıdır. Böylelikle kaba kuvvet saldırılarının önüne geçilmelidir.
- 22 numaralı varsayılan TCP portu değiştirilmelidir.

- SSH ile iki faktörlü kimlik doğrulama kullanılmalıdır.
- Şifresi olmayan kullanıcılar için SSH bağlantı talepleri devre dışı bırakılmalıdır.
- SSH anahtarı kimlik doğrulaması aktifleştirilmelidir.

Sıra Sizde

Görsel 2.10'da hazırlanan topolojiyi kullanarak SSH protokolünü aktifleştiriniz ve bağlantı testini gerçekleştiriniz.

2.7.2. Ağ Cihazlarını Yapılandırma Kontrolleri

Ağ cihazlarının (switch, router vb.) güvenliğini sağlamak ve saldırıların önüne geçmek için yapılandırma aşamasında çeşitli sıkılaştırma yöntemleri uygulamak gerekir. Saldırganlar, ağda bulunan bir cihaza eriştiği takdirde verileri ele geçirebilir ve ağ iletişimine zarar verebilir. Cihaz yapılandırmalarındaki bu sıkılaştırmalar, veri kaybının ve ihlallerinin önüne geçmede bir hayli etkilidir.

Ağ cihazlarında yapılan yapılandırma kontrolleri ve sıkılaştırma işlemlerinden bazıları şunlardır:

- Parola kullanımı açık metin olarak yapılandırma dosyasında görünmemelidir. Bunun yanında parolalar gizlenmelidir ve MD5 algoritması ile sistemde kriptolanmalıdır.
- Kimlik doğrulama için AAA sunucusu kullanılmalıdır.
- Her kullanıcı için ayrı ayrı yerel hesaplar oluşturulmalıdır.
- Kimlik doğrulama denemeleri sınırlandırılmalıdır.
- Cihaz yönetimleri belirlenmiş IP'ler üzerinden yapılmalıdır.
- Sistem günlükleri kullanılmalıdır.
- Telnet yerine SSH kullanılmalıdır.
- SNMP erişimi kısıtlanmalıdır.
- NTP kullanılmalıdır.

Not Farklı marka ve modellerde ağ cihazları vardır. Yöntemler benzer olsa bile komut yapılarında farklılıklar görülür. Kullanılan cihazın markasına göre komutlar da değişiklik gösterir.

2.7.3. Güvenlik Duvarı ve Sistemlerinin Yapılandırma Kontrolleri

Bilişim altyapılarında birçok firma tarafından üretilen güvenlik duvarı, atak önleme sistemleri ve ağ ürünleri bulunabilir. Bu bileşenlerin üzerindeki yapılandırmaların belirli mevzuat ve kurumsal standartlara uyumluluğu kontrol edilmelidir.

Güvenlik duvarı kontrolü yardımıyla belirli mevzuat, kurumsal standartlar ve belirlenebilecek özel kriterler kapsamında altyapıdaki tüm güvenlik duvarları üzerinde üretici bağımsız olarak kural analizi ve

normalleştirme, gizli risk faktörleri, zafiyet analizi gibi daha birçok kontrol gerçekleştirilebilir, bunlar için özelleştirilmiş raporlar alınabilir.

Ağ yapılandırma kontrolü yardımıyla orta ölçekli altyapılardan büyük ölçekli altyapılara kadar ölçekte ağ görünürlüğü sağlanır. Uçtan uca fiziksel, sanal ve bulut altyapısı dâhil olmak üzere bir ağ topoloji modeli olarak kullanılarak mevzuat ve kurumsal standartlara uyumluluk kontrol edilebilir. Ayrıca oluşturulan ağ modeli üzerinde hedeflenen yapılandırma test senaryoları ve iletişim sorunları test edilebilir.

Değişim kontrolü yardımıyla altyapıdaki güvenlik duvarı bileşenleri üzerinde yapılan tüm değişikliklerin bir iş akışına girdirilmesi ve sürecin güvenlik birimleri tarafından takibi gerçekleştirilebilir. Bu sayede yapılan her kural değişikliği bir talebe dönüştürülerek, iş akışına uygun bir biçimde uygulanıp uygulanmadığı kontrol edilebilir.

Sıra Sizde

İşletim sisteminizdeki yerel güvenlik duvarını kullanarak, ağınızda bulunan bir IP adresinden gelen bağlantıyı engelleyecek bir güvenlik kuralı oluşturunuz.

2.8. DONANIM GÜVENLİĞİ İLKELERİ

Sisteme zarar verebilecek potansiyel risk faktörleri, güvenlik tehdidi olarak tanımlanır. Bilişim teknolojileri sistem ve araçlarının güvenliği, fiziksel güvenlik ve fiziksel olmayan güvenlik olarak ikiye ayrılır. Bilişim araçlarının kullanıldığı sistemlerde fiziksel, donanımsal hasarlara sebebiyet verebilecek faktörler, fiziksel güvenlik tehditleri başlığı altında tanımlanır. Siber güvenlik tehditleri (virüs, trojan, worms vb.) ise fiziksel olmayan tehditler şeklinde isimlendirilir.

Fiziksel güvenlik tehditleri şunlardır:

- **İç Tehditler:** Hatalı sistem odası kurulumu, yangın, elektrik arızaları, plansız güç kaynağı yönetimi, nem, sıcaklık vb.
- **Dış Tehditler:** Yıldırım, deprem, sel vb.
- **İnsan Faktörü:** Yetkilendirilmemiş personellerin sistem odasına girişi, hatalı cihaz yapılandırmaları, yeterli bilgiye sahip olmayan yetkili personeller, hırsızlık, donanımların talimat dışı kullanımı, iş güvenliği önlemlerine uygun davranmamak vb.

Sistem donanımlarının fiziksel tehditlere karşı güvenli kalmasını sağlamak için belirtilen bu risk faktörlerine karşı güvenlik politikaları uygulanmalıdır.

Sıra Sizde

Ağ simülasyon yazılımı kullanarak bir anahtarlama cihazı ekleyiniz ve temel güvenlik yapılandırmalarını gerçekleştiriniz.

2.8.1. Donanımların Fiziksel Güvenlik İlkeleri

Kurumlar, fiziksel donanımların güvenliğini sağlamak için farklı strateji ve yöntemler uygular. Bazı uluslararası strateji ve yöntemler, fiziksel güvenlik için kullanılırken kurum içi politikalar ve kurumsal yapılara göre de güvenlik ilkeleri farklılıklar gösterir. Genel hatları itibarıyla iç tehditler, dış tehditler ve insan faktörü risklerine karşı donanımların fiziksel güvenlik ilkeleri uygulanır.

Sistem odaları kurulumunda ve diğer bilişim teknolojileri donanımlarının fiziksel güvenliğini sağlamada ISO:27001 standartları kullanılmalıdır. **ISO:27001** standartları kapsamında alınan önlemler hem sistemi hem de ortamın tamamını korumaya yöneliktir. Ortam ısısının ve nem yoğunluğunun doğru şekilde oluşmaması, odada kıvılcımlar çıkarak yangın oluşmasına veya cihazların hasar almasına neden olabilir.

Su basma riski her alanda olası bir durumdur. Su tahliye alanlarının bulunması, odanın su temasından korunaklı alanlara kurulması, eşiklerin ve döşemenin yükseltilmesi, nem uyarı sisteminin olması, her zaman korumayı ön plana çıkartır. Yapılacak bu tarz uygulamalar, **sistem odası standartları** kapsamında güvenliği her zaman ön planda tutar.

Enerji kontrolü; olası kaçak akımlar, fazla elektrik kullanımı, kesintilere karşı UPS sistemlerinin kurulması gibi ince ve önemli detaylar içerir. Topraklama yapılıp yapılmadığı, zeminin antistatik özelliğinin oluşturulması gibi sorun önleyici önlemleri içine alır.

Deprem kontrolü ise oluşabilecek bir depremde cihazların sabit kalabilmesi, kabinlerin uygun şekilde monte edilmesi gibi detayları içerir.

ISO:27001 standardında kablolar, anlaşılır şekilde etiketlenerek belli bir kurala göre sıralanmalı ve monte edilmelidir. İleride oluşabilecek arızalarda, her kim olursa olsun, arızayı bularak müdahale edebilmelidir. Kablolar, uygun şekilde kanallara konularak alanda karmaşa ve anlaşmazlıktan uzaklaştırılmalıdır. Haşerelere karşı koruma yapılmalı, gerekli şekilde kablolar döşenmeli veya borularla muhafaza edilmelidir.

2.8.1.1. Fiziksel Güvenlik Uygulama İlkeleri Örnekleri

Yangın

- Bilişim teknolojileri araçlarına gelen tüm kablolara dışarıdan gelecek ısıyı engellemek için ısı yalıtımı yapılmalıdır.
- Otomatik alarm sistemleri kullanılmalıdır. Alarm durumunda ilgili kişilere e-posta yoluyla sistemler tarafından bilgilendirme yapılmalıdır.
- Taşınabilir yangın söndürücülerin kapıya ve donanımlara yakın olmasına dikkat edilmelidir.

Sıcaklık ve Nem

- Sistem odalarının 15 ^{o}C ile 25 ^{o}C ısı aralığında olması, klima ve iklimlendirme sistemleri ile sağlanmalıdır.
- Donanımlar duvarlara çok yakın yerleştirilmemelidir. Duvarlara çok yakın yerleştirilen donanımlar, havalandırmayı engelleyerek sistemlerin iç ısılarının yükselmesine neden olabilir.
- Nem düzeyi %40 ile %75 arasında tutulmalıdır.

- Klima veya iklimlendirme sistemlerinin ofis alanı gibi kurumun kritik bölgelerinde sürekli çalışır durumda olması sağlanmalıdır.
- Kritik donanımların bulunduğu alanlarda iklimlendirme sistemleri sürekli izlemeye tabi tutulmalıdır.

Deprem

- Donanımlar, zeminden çok yükseğe yerleştirilmemelidir.
- Rack kabinlerin yere, tavana, kendi aralarında rack mount kitlerle sabitlenmesi ve içindeki tüm donanımların vidalarla ve kablo bağlarıyla sabitlenmesi sağlanmalıdır.
- Donanımlar özellikle zeminin üzerindeki katlarda pencerelerden uzak tutulmalıdır.

Enerji ve Kablolama

- Manyetik alan oluşumu sonucu veri kayıplarını minimize etmek için enerji ve ağ kablolaması, ayrı kanal veya ızgaralardan yapılmalıdır.
- Kritik donanımların; birbiriyle paralel çalışan, uygun kapasiteli, en az iki adet UPS'den beslenmesi sağlanmalı ve UPS'lere ayrı trafodan enerji verilmelidir.
- Topraklamanın uygun şekilde yapılması sağlanmalıdır.
- Mümkünse jeneratör kullanılmalıdır.
- Statik elektriğin yaratabileceği sorunlara engel olmak üzere teknik servis olarak gerekli fiziksel önlemler alınmalıdır.
- Ethernet kartları ve switch portlarına giden patch panel portları, bina data uçları ile aynı olacak şekilde etiketlenmelidir. Mümkünse farklı VLAN'lar için farklı renkte kablolar kullanılmalıdır. Uç terminasyon yapısı, switchlerin üzerine etiketlenerek yapıştırılmalıdır.

İnsan Faktörü

- Hırsızlık ve sabotaja karşı bina girişlerinde ve katlarda kamera bulunmalıdır.
- Ofis alanları sürekli kilitli tutulmalı, sadece yetkili personel için erişim izni verilmelidir.
- Kurum binası dışına çıkarılmış gizli donanım, yazılım ve dokümanlar; umumi alanlarda, evde, arabada açıkta bırakılmamalı, mümkünse kilitli çanta veya kutu ile taşınmalıdır.
- Bütün bu faktörlerin dışında su baskını toz, böcekler ve diğer faktörler için kurumun donanımlarının güvenliğini sağlayacak sıkılaştırma ve güvenlik ilkeleri uygulanmalıdır.

ÖLÇME VE DEĞERLENDİRME

A. Aşağıdaki parantezlerin içine cümlelerde verilen bilgiler doğru ise "D", yanlış ise "Y" yazınız.

1. ( ) Siber istihbarat, bir kurum için risk hâline gelebilecek siber tehdit faktörlerini önceden analiz eden, kuruma bu yönde uyarı ve önlem alma fırsatı sağlayan farklı çözüm ve araçlardır.

2. ( ) Aktif ağ saldırılarında saldırganlar; bir ağa yetkisiz bir şekilde girerek o ağı izleyebilir, verileri görüntüleyebilir, ele geçirebilir fakat silmez veya erişebilirliğini engellemez.

3. ( ) Kayıt defteri, işletim sistemindeki ayarları ve seçenekleri depolamak için kullanılan bir veri tabanıdır.

4. ( ) SQL Injection veri tabanında SQL sorgularına yetkisiz bir şekilde müdahale edilerek, SQL söz dizimi hatalarından veya özel karakter kullanılarak yapılacak sorgularla sistemin kandırılmasına dayanan saldırılardır.

B. Aşağıdaki cümlelerde boş bırakılan yerlere doğru sözcükleri yazınız.

5. Ağ güvenliği mimarilerinde ..... olması veya olmaması durumuna göre iki farklı mimari kullanılır.

6. Ağa yapılan saldırıların tespit edilip kayıt altına alınmasını ve ağda meydana gelen olayların geriye doğru izinin sürülmesini ..... sağlar.

7. Bilginin güvenliğini sağlamada ..... standartları kullanılmalıdır.

C. Aşağıdaki soruları dikkatlice okuyarak doğru cevabı işaretleyiniz.

8. Aşağıdakilerden hangisi ağ ve sistem güvenliğine yönelik atakları algılama ve önleme cihazlarından <u>değildir</u>?

A) Proxy
B) Güvenlik Duvarı
C) WAF
D) DHCP
E) DLP

9. Aşağıdakilerden hangisi yerel ağda veya uzak ağdan iletişim hâlindeki iki bilgisayarın arasına girerek ağı dinleyen, iletişimin üzerinden geçmesini sağlayarak verileri okuyan ya da verileri manipüle ederek iletişimin zarar görmesini sağlayan bir saldırı türüdür?

A) ARP Spoofing
B) Brute Force
C) DoS
D) DdoS
E) Phishing

10. Aşağıdakilerden hangisi kaba kuvvet saldırısı uygulamak için kullanılan araçlardan biridir?

A) Hydra
B) Kali
C) Cewl
D) SQLMapping
E) Nmap

D. Aşağıdaki soruların cevaplarını ilgili alana yazınız.

11. Ev ağınızda kablosuz ağ güvenliğinin sağlanması için alabileceğiniz önlemleri yazınız.

12. Siber güvenlik takımlarının kullandığı ana çözümlerden biri olan SIEM entegrasyonunun ne olduğunu yazınız.

13. Sunucu rolleri kurulmadan önce yapılması önerilen sıkılaştırmaların ne olduğunu yazınız.

14. Sistem odasında doğal afetlere karşı alınması gereken fiziksel önlemlerin ne olduğunu yazınız.

KRİPTOGRAFİ

3. ÖĞRENME BİRİMİ

KONULAR

3.1. KRİPTOGRAFİYE GİRİŞ

3.2. KLASİK KRİPTOGRAFİ

3.3. HASH FONKSİYONU

3.4. SİMETRİK KRİPTOGRAFİ

3.5. ASİMETRİK KRİPTOGRAFİ

3.6. STEGANOGRAFİ

NELER ÖĞRENECEKSİNİZ?

- Kriptoloji ve kriptoloji ile ilgili kavramları açıklama
- Sezar şifrelemesini uygulama
- Klasik kriptografik algoritmaları açıklama
- Özet fonksiyonlarını açıklama
- MD5 özet algoritması ile uygulama yapma
- Simetrik kriptografiyi açıklama
- Asimetrik kriptografiyi açıklama
- Steganografi kullanarak şifreleme uygulaması yapma

ANAHTAR KELİMELER

Akış şifreleme, asimetrik şifreleme, blok şifreleme, kimlik doğrulama algoritmaları, kriptografi, kriptoloji, modern şifreleme yöntemleri, özet fonksiyonları

HAZIRLIK ÇALIŞMALARI

1. Son yıllarda adı sıkça duyulan kripto paralarda "kripto" ifadesi neden kullanılmış olabilir? Düşüncelerinizi arkadaşlarınızla paylaşınız.
2. Bir arkadaşınıza göndereceğiniz mesajın başkası tarafından anlaşılamaması için ne yapardınız? Fikirlerinizi arkadaşlarınızla paylaşarak yapılabilecekler listesi oluşturunuz.

3.1. KRİPTOGRAFİYE GİRİŞ

Günümüz dünyasında veriler, bilgisayar ağlarında sorunsuz bir şekilde ilerlerken ve dijital işlemler görülmemiş bir hızla gerçekleşirken güçlü siber güvenlik önlemlerine olan ihtiyaç önemli bir hâl almıştır. Kritik verilerin korunması, veri gizliliğin sağlanması ve dijital iletişimin bütünlüğünün elde edilmesi, gün geçtikçe çeşitlenen siber tehditlere karşı önemli zorluklar olarak ortaya çıkmıştır. Bireysel veya kurumsal olarak siber güvenliği güçlendirmek için kullanılan birçok araç ve teknik arasında kriptoloji teknikleri daha da önemli hâle gelmiş, verileri ve iletişimi korumak için gerekli araçları sağlamıştır.

Kriptoloji; gizlilik, bütünlük, kimlik doğrulama, iletişim güvenliği gibi konuları inceleyen bilim ve matematik dalıdır. Kriptografik yöntemlerin tasarımını, analizini ve uygulanmasını kapsar. Kriptoloji, **kriptografi** ve **kriptoanaliz** olmak üzere iki alandan oluşur. Kriptografi, şifreleme teknikleri ile veri korumayı sağlarken kriptoanaliz, anahtar bilgisine sahip olmadan gizli verinin elde edilmesini veya kırılmasını sağlayan tekniklerdir. Her iki alan da birbirinin gelişmesine katkı sağlar. Kriptografideki gelişmeler sayesinde daha güvenli yapılar oluşur. Bu güvenli yapıları kırmak için kriptoanalizin daha da gelişmesi gerekir. Benzer şekilde kriptoanalizdeki gelişmeler daha güçlü kriptolamanın ortaya çıkmasını tetikler.

Yunanca gizli (cryptos) ve yazı (graphein) kelimelerinden oluşan kriptografi (cryptography); bilgileri yetkisiz kişiler tarafından anlaşılmaz hâle getirmek, verinin kötü niyetli olarak değiştirilip değiştirilmediğinin anlaşılmasını sağlamak, göndericinin veya alıcının kimliğini doğrulamak için kullanılan bir dizi tekniği ifade eder.

Eski çağlardan bu yana köklü bir geçmişe sahip kriptografi, modern siber güvenlikte de yaygın olarak kullanılır. Güvenli bir web sayfasına bağlanırken, e-posta gönderirken, bir forum sayfasına üye olurken veya kablosuz bir ağa bağlanırken genellikle arka planda kriptografik hizmetler çalışır.

Kriptografinin siber güvenlikteki önemi; gizlilik, bütünlük ve doğruluk olmak üzere üç temel güvenlik hedefini gerçekleştirebilme yeteneğinde yatar. **Gizlilik**, bilgilerin yetkisiz kişilerden gizli kalmasını sağlar ve hassas verileri, kötü amaçlı kişilerden veya yazılımlardan korur. Kriptografik algoritmalar kullanarak verileri şifrelemek, bilgilerin yetkisiz kişiler tarafından çözümlenmesini güçleştirir. **Bütünlük**, verinin yaşam döngüsü boyunca değiştirilmeden güvenilir kalmasını sağlar. Dijital imzalar ve özet (hash) fonksiyonları gibi kriptografik mekanizmalar, verinin bütünlüğünü doğrulamak için kullanılır. Bu teknikler; kritik bilgilerin bozulmamış, değiştirilmemiş ve güvenilir olarak kalmasını sağlamada önemli bir rol oynar. **Doğruluk**, siber güvenliğin diğer bir hayati unsurudur ve dijital iletişimde yer alan kişilerin veya cihazların kimliklerini ve güvenilirliğini sağlayan kriptografik mekanizmalar aracılığıyla ele alınır. Doğruluk, güvenli kimlik doğrulamasını sağlar ve kullanıcıların karşı tarafın kimliğini doğrulamalarını ve iletişim kurdukları kişilerin doğru alıcılar olduğunu garanti eder. Bu temel güvenlik hedeflerinin

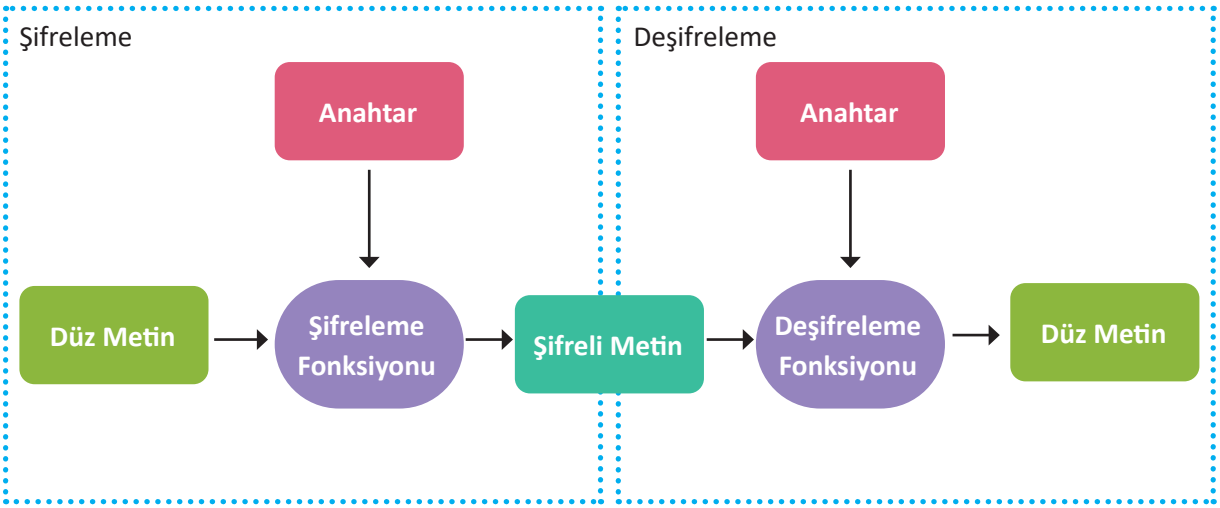
ötesinde, kriptografi aynı zamanda, diğer siber güvenlik mekanizmalarını desteklemede hayati bir rol oynar. Güvenli anahtar değişim protokollerine imkân sağlayarak iletişimdeki taraflar arasında güvenli kanallarının kurulmasına yardımcı olur.

Sonuç olarak kriptografi, modern siber güvenliğin temelini oluşturarak siber tehditlere karşı önemli koruma sağlar ve dijital bilgilerin gizliliğini, bütünlüğünü ve doğruluğunu garanti eder. Teknoloji ilerledikçe ve siber tehditler evrildikçe, sağlam kriptografik tekniklere olan ihtiyaç giderek daha da önemli hâle gelir. Kuruluşlar ve bireyler; kriptografiyi benimseyerek ve uygulayarak siber saldırılara karşı direncini artırabilir, dijital etkileşimlerde güveni sağlayabilir, hassas bilgilerin gizliliğini ve güvenliğini koruyabilir. Modern şifreleme veya diğer bir ismi ile kriptolama, matematiksel prensiplerden oluşan birçok algoritma tarafından sağlanır.

Kriptografi; şifreleme, deşifreleme (şifre çözme), kimlik doğrulama, erişim kontrolü ve inkâr edilemezlik sağlayarak veri ve iletişim güvenliğini oluşturmaya katkıda bulunur. Kriptografi ile ilgili kavramlar şunlardır:

- **Şifreleme:** Bilginin gerçek anlamını gizleyen gizli koda dönüştürüldüğü yöntemdir.
- **Deşifreleme (Şifre Çözme):** Gizlenmiş bir mesajın şifresini çözme işlemidir. Mesajın alıcısı tarafından gerçekleştirilir.
- **Veri Kaynağı Kimlik Doğrulaması:** Mesajın doğrulanmış bir gönderen tarafından gönderildiğinin garantisidir.
- **Erişim Kontrolü:** Kuruluşların verilere ve kaynaklara kimin erişim yetkisi olduğunu yönetmesini sağlayan bir veri güvenliği sürecidir.
- **İnkâr Edilemezlik:** Bir olayın geçerliliğinin inkâr edilemeyeceğinin güvencesidir.

Şifreleme, düz metni alıp şifreli metne dönüştüren bir fonksiyondur. Görsel 3.1'de görüldüğü gibi genellikle açık bir **düz metin (plaintext)**, bir **anahtar** ve bir **şifreleme (encryption)** fonksiyonu aracılığıyla **şifreli metne (ciphertext)** dönüştürülür, uygun şifre çözme anahtarları olmadan şifreli metnin anlaşılamaz hâle getirilmesi sağlanır. Uygun anahtar ve **deşifreleme (decryption)** fonksiyonu ile şifreli metin tekrar düz metne dönüştürülebilir.



Görsel 3.1: Bir anahtar yardımıyla düz metnin şifrelemeye tabi tutulması ve şifreli metin elde edilmesi

Kriptografi çoğunlukla matematiksel temellere dayandığı için Görsel 3.1'de verilen bilgiler daha kapsamlı olarak şu şekilde açıklanır:

P: Düz metinlerden oluşan sonlu bir küme

C: Şifreli metinlerden oluşan sonlu bir küme

K: Olası anahtarların sonlu kümesi (Anahtar uzayı)

ε: Şifreleme fonksiyonlarının sonlu kümesi

D: Şifre çözme fonksiyonlarının sonlu kümesi

$\forall k \in K, \quad e_k \in \varepsilon, \quad \mathrm{d}_k \in D$ olmak üzere;

k, Anahtar uzayından (K) seçilen herhangi bir anahtarı temsil etsin.

e_k**,** k anahtarını kullanan şifreleme fonksiyonlarından biri olsun.

d_k**,** k anahtarını kullanan deşifreleme fonksiyonlarından biri olsun.

Bu durumda şifreleme fonksiyonu bir düz metni (P) alır ve şifreli metne (C) dönüştürür:

$$e_k : P \to C$$

Aynı şekilde deşifreleme fonksiyonu, anahtarı kullanarak şifreli metni (C) düz metne (P) dönüştürür:

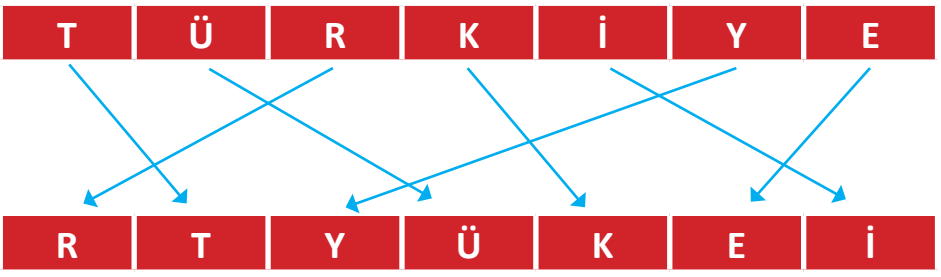
$$d_k : C \to P$$

$\forall x \in P$ için x düz metin kümesinin herhangi bir elemanı olmak üzere;

$$d_k\left(e_k\left(x\right)\right) = x$$

şartını sağlıyorsa bu durumda (*P, C, K, ε, D*) beşli ifadesine **kripto sistem** denir.

Kriptografide bir düz metnin şifrelenmesinde **yer değiştirme (transposition)** veya **yerine koyma (substitution)** adı verilen iki eylem gerçekleştirilir. Görsel 3.2'de görüldüğü gibi yer değiştirme, bir düz metnin karakterlerinin belirli bir mantığa göre kendi aralarında yer değiştirmesidir. Bu yer değiştirme işlemi, karakter bazlı olarak yapılabildiği gibi bit bazlı veya bayt bazlı olarak da yapılabilir.



Görsel 3.2: Mesajın gizliliği için basit bir yöntem olarak kullanılan yer değiştirme işlemi

Görsel 3.2'de yer alan örnek için;

Düz Metin: TÜRKİYE

Şifreli Metin: RTYÜKEİ

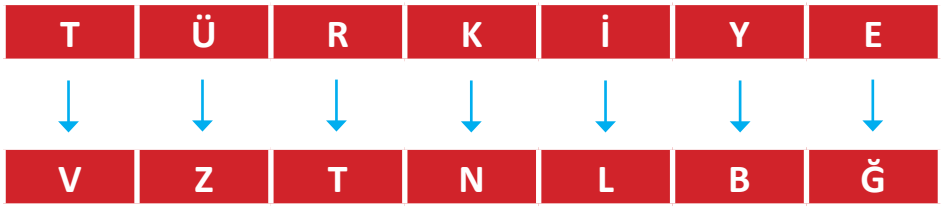
olarak ifade edilebilir. Örnekteki şifreli metnin ilk karakteri "R", düz metnin üçüncü karakteridir. Şifreli metnin alıcı tarafından tekrar düz metne doğru bir şekilde dönüştürülebilmesi için şifrelemenin nasıl yapıldığının belirtilmesi gerekir. Genelde bu işlem için yeni dizilişin nasıl yapıldığı, **permütasyon** adı

verilen bir diziliş sıralaması ile ifade edilir. O hâlde Görsel 3.2'deki örnek için permütasyon dizilişi şu şekildedir:

3-1-6-2-4-7-5

Permütasyon dizilişi hem şifreleme işleminin hem de deşifreleme işleminin nasıl yapılacağını gösterir. O hâlde bu permütasyon dizilişi, yer değiştirme işleminin anahtarıdır. Yer değiştirme işleminde permütasyon dizisi vermek yerine döngüsel kaydırma gibi farklı yöntemler de kullanılabilir. Örneğin TÜRKİYE düz metni, döngüsel 3-sağa kaydırma işlemi sonucunda İYETÜRK şifreli metni olacaktır.

Yerine koyma işlemi ise düz metnin her bir karakterinin yerine diğer bir karakter koymayı ifade eder. Görsel 3.3'te yerine koyma işlemi verilmiştir.



Görsel 3.3: Yerine koyma yönteminde her karakterin yerine başka bir karakter kullanılması

Görsel 3.3'te yer alan örnek için;

Düz Metin: TÜRKİYE

Şifreli Metin: VZTNLBĞ

olarak ifade edilebilir. Bu yöntemde hangi karakterin yerine hangi karakterin geleceği yine bir fonksiyon ile belirtilmelidir. Benzer şekilde bir harf matrisi de kullanılabilir. Görsel 3.3'te yer alan örnekte her harfin yerine alfabetik olarak üç sonraki harf kullanılmıştır. Alfabedeki son harfe gelindiğinde bir sonraki harf için tekrar alfabenin ilk harfine dönülür. Bu örnek şifreleme yöntemi **Sezar şifrelemesi** olarak bilinir. Örnekte düz metnin şifrelenmesinde ve şifrenin çözümlenmesinde "3" sayısı kritik öneme sahiptir. O hâlde Görsel 3.3'teki örnek için "3" sayısı anahtardır.

Permütasyon dizilişi için kullanılabilecek diğer bir yöntem de geometrik diziliş temelli olan **sütun yer değişikliği (column transposition)** yöntemidir. Bu yöntemde düz metin, bir anahtar kelimenin harf sayısı kadar sütundan oluşan bir tabloya satır sırasıyla yazılır. Ardından sütunların yeri, anahtarın harf sıralamasına göre değiştirilir ve sütun bazlı olarak okunacak şekilde şifrelenir.

Örnek

Düz Metin: SAĞ KANADA DESTEK GEREK

Anahtar: KALEM

K	A	L	E	M
S	A	Ğ	K	A
N	A	D	A	D
E	S	T	E	K
G	E	R	E	K

Anahtar kelimenin alfabetik harf sırasına göre (AEKLM), sütun bazında okunduğunda şifreli metin şu şekildedir:

Şifreli Metin: AASEKAEESNEGĞDTRADKK

Görüleceği üzere bu örnekte yapılan şifreleme işlemi, anahtar kelimeye bağlı olarak harflerin yer değiştirmesidir.

3.2. KLASİK KRİPTOGRAFİ

Görsel 3.4: Scytale şifreleme aracı

Bilgiyi gizleyerek saklama uygulaması, iletişimin tarihi boyunca büyük önem taşımıştır. Özellikle askerî alanlarda gönderilen mesajların gizliliği ve olası ele geçme durumlarında anlaşılamaması için yöntemler geliştirilmiştir. Klasik kriptografi, modern bilgisayar tabanlı kriptografinin öncüsü olan geleneksel şifreleme tekniklerini ifade eder. Bu teknikler, yazılı metinleri veya verileri korumak için matematiksel yöntemler ve şifreleme algoritmaları kullanır. Klasik kriptografi, genellikle basit şifreleme yöntemlerini içerir ve tarih boyunca iletişimde kullanılmıştır. Bilinen en eski şifreleme yöntemlerinden biri, Antik Yunanlılar tarafından kullanıldığı düşünülen ve Görsel 3.4'te görülen **Scytale** aracıdır.

Scytale, milattan önce 400 yıllarında Spartalılar tarafından askerî amaçla kullanılan bir mesaj şifreleme aracıdır. Şerit şeklindeki bir parşömen, belirli bir çaptaki çubuğa sarılır ve **düz metin** mesajı çubuk boyunca yazılır. Ardından sadece düz şerit, iletilmek üzere gönderilir. Çubuk olmadan düz bir şerit şeklinde okunduğunda şifreli metin elde edilir. Sadece doğru çaptaki çubuk ile mesajın okunması mümkündür. O hâlde doğru çaptaki çubuğun hem göndericide hem de alıcıda olması gerekir. Bu şifrelemede mesajın boyutu arttıkça çözülmesi karmaşıklaşır. Bu yöntemdeki anahtar (key), çubuğun çapıdır. Kriptografik açıdan kolay bir yöntem olsa da aslında yapılan işlem, karakterlerin basitçe yer değiştirmesidir (transposition). Farklı çaplardaki çeşitli çubuklarla (anahtarlarla) yapılan denemelerde şifreli metinden düz metnin elde edilmesi güç değildir. Mesajın çözülebilmesi için olası bütün çaplara sahip çubukların sırayla denenmesi yönteminde olduğu gibi bir anahtarın sistematik şekilde deneme yanılma yöntemi ile bulunmaya çalışılmasına **deneme yanılma saldırısı** veya **kaba kuvvet saldırısı (brute force attack)** denir. Deneme yanılma saldırısında her ihtimal denenebileceği için nihai olarak çözüm bulunur ancak burada önemli olan unsur, çözümün hangi sürede bulunduğudur. Anahtar ne kadar uzun olursa şifrelemenin kırılması da (üstel olarak) o derece zaman alacaktır. Şifreli metni çözmek için anahtarlar sırasıyla deneniyorsa bu durumda her şifreli metin, belirli bir sürede doğru düz metne dönüşebilir, bir başka deyişle şifre kırılır. Bu süreye **anahtarın kırılabilirlik ömrü** denir. Belirli bir zaman diliminde ne kadar çok anahtar denenebilirse bu ömür o derece kısalır. Günümüz bilişim araçları ile saniyede yapılabilecek işlem sayısı düşünüldüğünde kısa kırılabilirlik ömrüne sahip anahtarların ne kadar hızlı kırılabildiği anlaşılabilir.

3.2.1. Sezar Şifrelemesi

Julius Sezar'a atfedilen **Sezar şifrelemesi**, kullanılan eski şifreleme algoritmalarından biridir. Sezar şifrelemesinde düz metnin her bir harfi yerine alfabetik olarak belirli sayı sonrasındaki başka bir harf kullanılır. Bu yeni harflerden oluşan metin, şifreli metin olur. Örneğin düz metindeki her bir harf yerine alfabedeki bir sonraki harf kullanarak şifreli metin elde edilebilir.

Bir harf kaydırma örneğine göre düz metin "ANKARA" ise şifreli metin "BOLBSB" olacaktır. İki harf kaydırıldığında ise "CÖMCŞC" elde edilir. Şifreyi çözmek için ise kaç harf kaydırıldığının bilinmesi (sadece gönderici ile alıcı tarafından bilinmesi) gerekir. Bu şifreleme yönteminde anahtar, kaç harf sonrasının kullanıldığını gösteren sayıdır. Sezar şifrelemesinde her harf yerine farklı bir karakterden yararlanıldığı için yerine koyma (substitution) metodu kullanılmıştır.

Sezar şifrelemesi, matematiksel olarak şu şekilde ifade edilebilir:

E: Şifreleme fonksiyonu

D: Şifre çözme fonksiyonu

x: Düz metin

n: Anahtar

$E(x) = (x + n) \pmod{29}$

$D(x) = (x - n) \pmod{29}$

1. Uygulama

İngiliz alfabesinde 26 harf olduğunu göz önünde bulundurarak işlem adımlarına göre anahtar 5 olmak üzere SECURITY kelimesini şifreleyiniz ve şifreyi çözünüz.

Düz Metin: SECURITY

Anahtar: 5

1. Adım: Alfabedeki harf sırasına göre düz metni sayısallaştırınız.

18, 4, 2, 20, 17, 8, 19, 24

2. Adım: Şifreleme fonksiyonunu yazınız.

$E(x) = (x + 5) \pmod{26}$

3. Adım: Düz metindeki her karakteri E(x) şifreleme fonksiyonuna göre hesaplayınız. Ardından sayıları tekrar harflere dönüştürünüz.

S	E	C	U	R	I	T	Y
18	4	2	20	17	8	19	24
18 + 5 (MOD 26)	4 + 5 (MOD 26)	2 + 5 (MOD 26)	20 + 5 (MOD 26)	17 + 5 (MOD 26)	8 + 5 (MOD 26)	19 + 5 (MOD 26)	24 + 5 (MOD 26)
23	9	7	25	22	13	24	3
X	J	H	Z	W	N	Y	D

4. Adım: XJHZWNYD şifreli metnini elde ediniz.

5. Adım: Şifre çözme fonksiyonunu yazınız.

$D(x) = (x - 5) \pmod{26}$

6. Adım: Şifreli metindeki harflerin sayısal karşılığını D(x) fonksiyonuna göre hesaplayınız ve elde ettiğiniz değerleri tekrar harflere dönüştürünüz.

X	J	H	Z	W	N	Y	D
23	9	7	25	22	13	24	3
23 - 5 (MOD 26)	9 - 5 (MOD 26)	7 - 5 (MOD 26)	25 - 5 (MOD 26)	22 - 5 (MOD 26)	13 - 5 (MOD 26)	24 - 5 (MOD 26)	3 - 5 (MOD 26)
18	4	2	20	17	8	19	-2 + 26 = 24
S	E	C	U	R	I	T	Y

7. Adım: Deşifreleme sonucunda SECURITY düz metnini elde ediniz.

Not

Modüler aritmetikte negatif bir sayı elde edildiğinde sayının pozitif karşılığını bulmak için negatif sayı ile mod sayısı toplanır.

Negatif Sayı	MOD	Pozitif Karşılığı
-2	27	25
-8	21	13
-5	29	24

Sıra Sizde

Sezar şifrelemesi ile yazılmış aşağıdaki şifreli metni çözmeye çalışınız ve bulduğunuz çözümün deneme yanılma saldırısı ile kaç denemede kesin olarak çözüldüğünü açıklayınız.

"grlmf ğvsfrarbrgv şicibilpr öçz lmküs zmnıa rpvavf"

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. Şifreli metni sayılara dönüştürdü.		
2. Deşifreleme fonksiyonunu yazdı.		
3. 1'den başlamak üzere sırasıyla anahtarları denedi.		
4. Denenen her anahtar için deşifrelenmiş metni yazdı.		
5. Doğru düz metni elde etti.		
6. Toplamda en fazla kaç deneme yapılabildiğini açıkladı.		
"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.		

3.2.2. Vigenére Şifrelemesi

Blaise de Vigenère tarafından 16. yüzyılda geliştirilen **Vigenère şifrelemesi**, klasik şifreleme algoritmalarından biridir. Bu yöntemde, bir anahtar kelime kullanılarak düz metin şifrelenir. Şifreleme işlemi için Görsel 3.5'te görülen Vigenère şifreleme tablosu kullanılır. Bu tabloda alfabedeki harfler ilk satıra sırayla yazılır. Sonraki satırda harfler birer sola kayacak şekilde tablo doldurulur.

	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z
A	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z
B	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A
C	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B
Ç	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C
D	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç
E	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D
F	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E
G	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F
Ğ	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G
H	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ
I	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H
İ	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I
J	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ
K	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J
L	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K
M	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L
N	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M
O	O	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N
Ö	Ö	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O
P	P	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö
R	R	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P
S	S	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R
Ş	Ş	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S
T	T	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş
U	U	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T
Ü	Ü	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U
V	V	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü
Y	Y	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V
Z	Z	A	B	C	Ç	D	E	F	G	Ğ	H	I	İ	J	K	L	M	N	O	Ö	P	R	S	Ş	T	U	Ü	V	Y

Görsel 3.5: Vigenère şifreleme tablosu

Şifreleme işlemi sırasında açık metin ve anahtar kelime kullanılır. Açık metindeki her bir harf, anahtar kelimenin aynı sıradaki harfi ile Vigeneré tablosundaki ilgili satır ve sütunun kesişiminde yer alan harfe bakılarak eşleştirlir. Elde edilen harfler, şifrelenmiş metin olarak kaydedilir. Çoğu durumda anahtar kelime düz metinden daha kısa olacağı için (kolay anlaşılması için) anahtarı düz metin boyutuna gelinceye kadar tekrarlayarak uzatmak gerekir.

Örnek

Şifrelemek istenen düz metin "SİBERSİSTEM" ve anahtar "GİZLİ" kelimesi olsun.

Açık Metin: SİBERSİSTEM

Anahtar: GİZLİ

Düz Metin	S	İ	B	E	R	S	İ	S	T	E	M
Anahtar	G	İ	Z	L	İ	G	İ	Z	L	İ	G
Şifreli Metin	**Z**	Ş	A	**P**	C	**Z**	Ş	**R**	Ğ	**N**	Ş

O hâlde şifreli metin ZŞAPCZŞRĞNŞ elde edilir.

Düz metindeki belirli bir karakter, her zaman aynı karakterle eşleşmez. Örneğin düz metindeki "S" karakteri, "Z" ve "R"; "E" karakteri, "P" ve "N" olacak şekilde şifrelenmiştir. Sezar algoritmasında her harf, her zaman aynı harfle eşleşir ancak Vigenère şifrelemesinde her harf, birden fazla harf olacak şekilde şifrelenebilir. Bu tür şifrelemelere **çoklu alfabetik şifrelemeler (polialfabetik)** denir.

Araştırma

Vigenére şifrelemesinde hangi anahtarların kullanılmasının ciddi güvenlik açığı oluşturduğunu araştırınız. Elde ettiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

3.2.3. Hill Şifrelemesi

Lester S.Hill tarafından 1929 yılında ortaya çıkarılan **Hill şifrelemesi,** klasik şifreleme algoritmalarından biridir. Bu şifreleme, lineer cebir ilkelerine dayalı olan ve yerine koyma işlemini gerçekleştiren bir algoritmadır. Düz metinden şifreli metin elde etmek için matrislerin çarpımından faydalanır. Bu şifrelemede düz metin önce belirli boyutta (n) bloklara ayrılır. Daha sonra düz metin rakamlar ile ifade edilir. Burada kullanılan yöntem genelde A harfi yerine 0, B harfi yerine 1, C harfi yerine 2, Z yerine 28 kullanmak gibi harfleri sıra numarası ile eşleştirmektir. Ardından n x n boyutunda bir anahtar matris üretilir. Düz metnin her bir bloku vektör olarak ifade edilir ve anahtar matrisi ile çarpılır. Çarpım sonucunda elde edilen şifreli matris tekrar harflere dönüştürülerek şifreli metin elde edilir. Şifreli metnin çözülmesinde ise anahtar matrisin tersi kullanılır. Şifreli matris ile anahtar matrisin tersi çarpılır ve tekrar harf dönüştürme işlemi yapılarak düz metin elde edilir. Burada dikkat edilmesi gereken nokta, işlemler sonucunda 28'den büyük bir değer elde edildiğinde (örneğin 29) tekrar alfabenin ilk harfine dönülmesi gerekliliğidir. Bir başka deyişle işlemler matematiksel olarak MOD 29'a göre yapılmalıdır.

2. Uygulama

İşlem basamaklarına göre Hill algoritması kullanarak SB düz metnini şifreleyiniz ve şifreyi çözünüz.

1. Adım: Düz metin, 2 boyutunda bir mesajdır. Bu mesajı sayısal olarak gösteriniz. S'yi yirmi birinci harf, B'yi birinci harf olarak dikkate alınız. Düz metni bir vektör olarak gösteriniz.

$$\begin{bmatrix} 21 \\ 1 \end{bmatrix}$$

2. Adım: n = 2 olduğu için 2 x 2 boyutunda bir anahtar matris üretiniz. Bu adım için aşağıdaki anahtar matrisini kullanınız.

$$\begin{bmatrix} 2 & 3 \\ 5 & 7 \end{bmatrix}$$

3. Adım: Düz metin vektörü ile anahtar matrisini çarpınız.

$$\begin{vmatrix} 21 \\ 1 \end{vmatrix} \cdot \begin{bmatrix} 2 & 3 \\ 5 & 7 \end{bmatrix} = \begin{bmatrix} 21.2+1.3 \\ 21.5+1.7 \end{bmatrix} = \begin{bmatrix} 45 \\ 112 \end{bmatrix} (\text{MOD } 29) \equiv \begin{bmatrix} 16 \\ 25 \end{bmatrix} (\text{MOD } 29)$$

4. Adım: Türkçe için 16 ve 25. harfe göre şifreli metni bulunuz.

Şifreli Metin: "NU"

5. Adım: Şifreli metinden düz metnin elde edilmesi için anahtar matrisin MOD 29'a göre tersini bulunuz.

$$\begin{bmatrix} 2 & 3 \\ 5 & 7 \end{bmatrix}^{-1} \equiv \begin{bmatrix} -7 & 3 \\ 5 & -2 \end{bmatrix} \equiv \begin{bmatrix} 22 & 3 \\ 5 & 27 \end{bmatrix} (\text{MOD } 29)$$

6. Adım: Şifreli matris ile ters anahtar matrisini çarpınız.

$$\begin{bmatrix} 16 \\ 25 \end{bmatrix} \begin{bmatrix} 22 & 3 \\ 5 & 27 \end{bmatrix} = \begin{bmatrix} 16.22+25.3 \\ 16.5+25.27 \end{bmatrix} = \begin{bmatrix} 427 \\ 755 \end{bmatrix} \equiv \begin{bmatrix} 21 \\ 1 \end{bmatrix} (\text{MOD } 29)$$

7. Adım: Altıncı adımda elde edilen vektördeki sayıları harflere dönüştürünüz. 21. harf "S" ve 1. harf "B" olduğu için düz metnin "SB" şeklinde olduğunu gözlemleyiniz.

3.2.4. Vernam Şifrelemesi

Vernam şifreleme, metinler üzerinde şifreleme yapabilen klasik bir şifreleme algoritmasıdır. Hill şifrelemesinde olduğu gibi burada da harflerin rakamlara dönüştürülmesi gerekir. Benzer şekilde harflerin sıra numarası (A=0, B=1 gibi), harfler yerine kullanılır. Algoritma, yerine koyma metoduna göre şifreleme yapar. Şifrelemede kullanılan anahtar ve düz metin sayılara dönüştürülür. Anahtar, metin boyunca uzatılır. Böylece hem düz metin hem de anahtar, sayılarla ifade edilir. Sonraki aşamada bu karakterler ikilik sayı sistemi ile ifade edilir. Düz metnin her bir biti ile anahtarın aynı sıradaki biti, XOR (Özel Veya) işlemine tabi tutulur. Elde edilen değer, onluk sayı sistemine çevrilir ve denk gelen harfe dönüştürülür. Hill algoritmasında olduğu gibi XOR işlemi sonucunda elde edilen değer, Türk alfabesi için MOD 29'a göre hesaplanır.

Not

XOR işlemi, bit düzeyinde yapılan mantıksal bir işlemdir.

P	Q	P XOR Q
1	1	0
1	0	1
0	1	1
0	0	0

3. Uygulama

İşlem basamaklarına göre Vernam algoritmasını kullanarak "KONYA" düz metnini "VAN" anahtar kelimesi ile şifreleyiniz ve şifreyi çözünüz.

1. Adım: Düz metni harf sırasına göre sayısallaştırınız.

K	O	N	Y	A
13	17	16	27	0

2. Adım: Anahtarı harf sırasına göre sayısallaştırınız.

V	A	N
26	0	16

3. Adım: Anahtarı düz metin boyutunca tekrar ettiriniz.

Düz Metin	13	17	16	27	0
Anahtar	26	0	16	26	0

4. Adım: Elde ettiğiniz değerleri ikilik sayı sistemi ile yazınız.

Düz Metin	13	17	16	27	0
	01101	10001	10000	11011	00000
Anahtar	**26**	0	16	**26**	0
	11010	00000	10000	**11010**	00000

5. Adım: Düz metnin bitleri ile anahtarın bitleri arasında sırasıyla XOR işlemini uygulayınız.

Düz Metin Bitleri	01101	10001	10000	11011	00000
Anahtar Bitleri	11010	00000	10000	11010	00000
XOR Sonucu	**10111**	10001	00000	**00001**	**00000**

6. Adım: İkilik sayıları onluk sayı sistemine dönüştürünüz.

XOR Sonucu	10111	10001	00000	00001	00000
XOR Onluk	23	17	0	1	0
Harf Karşılığı	**T**	O	A	**B**	**A**

O hâlde "KONYA" düz metninin "VAN" anahtarına göre Vernam algoritması sonucu "TOABA" şifreli metni elde edilir.

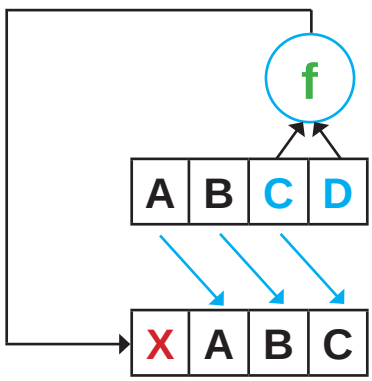
7. Adım: Şifreli metnin çözülmesi için aynı işlemleri yapınız.

Şifreli Metin	T	O	A	B	A
Anahtar	V	A	N	V	A
Şifreli Metin (Sayısal)	23	17	0	1	0
Anahtar (Sayısal)	26	0	16	26	0
Şifreli Metin (İkilik)	10111	10001	000000	00001	00000
Anahtar (İkilik)	11010	00000	10000	11010	00000
XOR Sonucu	01101	10001	10000	11011	00000
Onluk Karşılığı	13	17	16	26	0
Düz Metin	K	O	N	Y	A

3.2.5. LFSR Algoritması

Lineer geri beslemeli yazmaç (Linear Feedback Shift Register-LSFR) kriptografide ve bilgisayar bilimlerinde yaygın olarak kullanılan bir kavramdır. Bit bazlı çalışan ve basit bir yapıya sahip olan LFSR; dizi oluşturma, rastgele sayı üretme, şifreleme gibi alanlarda kullanılır. Özellikle akış şifreleme (stream cipher) yöntemlerinde LFSR'ler önemli bir rol oynar. LFSR tabanlı akış şifreleme yöntemleri, bilgiyi blok şifreleme yöntemlerine göre daha hızlı ve etkili bir şekilde şifreleyebilir.

LFSR, belirli boyuttaki bitler üzerinde yapılan lineer işlemler sonucunda elde edilen bitlerin kullanılması ve bit kaydırma işlemlerinin yapılması mantığına dayanır. Genelde kullanılan işlem XOR fonksiyonudur. Fonksiyona girdi olarak verilen bitler **tap bitleri** şeklinde adlandırılır. Görsel 3.6'da 4-bit uzunluğundaki bir LFSR'nin çalışma prensibi verilmiştir.



Görsel 3.6: Dört bitlik LFSR tasarımı

Görsel 3.6'da A, B, C ve D olarak gösterilen 4-bitlik, başlangıç değerleri bilinen bir dizi yer alır. Bu dizinin tap bitleri (C ve D) üzerinde lineer bir F fonksiyonu çalışır ve X olarak gösterilen bir değer elde edilir. Daha sonra bu dizinin tüm elemanları birer sağa kaydırılır ve en başa hesaplanan X değeri eklenir. Burada D'nin değeri diziden çıkacaktır. Bu işlem tekrar ettikçe yeni sayılar üretilir.

4. Uygulama

İşlem basamaklarına göre başlangıç bitleri "1011" olan 4-bitlik bir diziyi kullanarak LFSR algoritmasıyla sayılar üretiniz. Tap bitleri olarak C ve D bitlerini kullanınız. F fonksiyonu için XOR işlemini uygulayınız.

Not

Bu uygulamada 4-bit kullanıldığı için üretilen rastgele sayılar 0 ile 15 arasındadır. Daha fazla bit kullanımı, daha büyük bir sayı aralığı anlamına gelir.

1. Adım: Başlangıç bitlerinin yerleştirilmesi işlemini yapınız.

1	0	1	1

2. Adım: Tap bitleri üzerinde f-fonksiyonunu (XOR) çalıştırınız.

x = (1 XOR 1) = 0

3. Adım: Bit kaydırma işlemini yapıp oluşan boşluğa x değerini yerleştiriniz.

0	1	0	1

4. Adım: Üretilen değerin onluk sistemdeki karşılığını yazınız.

0101=5

5. Adım: Dördüncü adımda üretilen sayıdan sonraki altı sayının hesaplamasını yapınız.

1	0	1	0
1	1	0	1
1	1	1	0
1	1	1	1
0	1	1	1
0	0	1	1

Hesaplama sonucu bulunan sayılar onluk sayı sistemine dönüştürülürse 5-10-13-14-15-7-3 sayıları elde edilir. Ayrıca diziden çıkarılan bitler, anahtar üretmek için kullanılabilir.

Sıra Sizde

Başlangıç değeri "10101" olan 5-bitlik bir LFSR tasarlayınız. F-fonksiyonu olarak (B XOR C) XOR D kullanınız.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi		
Değerlendirme Ölçütleri	**Evet**	**Hayır**
1. LFSR tasarımını yaptı.		
2. Başlangıç değerini doğru yerleştirdi.		
3. Tap bitleri üzerinde F fonksiyonunu doğru uyguladı.		
4. LFSR sonucunda üretilen sayıyı hesapladı.		
"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.		

3.2.6. Klasik Şifrelemede Kripto Analiz

Klasik şifrelemeler, günümüz modern dünyasında oldukça zayıf kabul edilir. Özellikle bilgi işlem cihazlarının hızlı işlem yapabilme yetenekleri, bu tür şifrelemelerin kolaylıkla kırılmasını sağlar. Örneğin Sezar şifrelemesine bakıldığında Türkçe için anahtar uzayı en fazla 29 olabilir. Bir başka deyişle anahtar bilinmese bile şifreli metin üzerinde yapılacak 29 deneme yanılma sonucunda ortaya çıkan çözümlerden biri düz metni verecektir. Bilgisayarlar ile yapılacak basit bir kodlamayla çok kısa sürede şifreli metin çözülebildiği için günümüzde bu yöntem kullanılmaz.

Sezar şifrelemesi gibi her harf yerine farklı bir harfin kullanıldığı şifreleme türlerinde çeşitli dil kuralları uygulanarak şifreli metnin daha kısa sürede çözümlenmesi mümkündür. Sezar şifrelemesi tek alfabetik bir şifrelemedir. **Tek alfabetik şifreleme (mono alfabetik)**, düz metnin harflerinin tek bir alfabetik anahtara dayalı olarak şifreli metin harfleriyle eşlendiği herhangi bir şifredir. Bir başka deyişle her harf, her zaman aynı harfle eşleşir. Örneğin "1" anahtarlı Sezar şifrelemesinde düz metindeki "A" harfi, her zaman şifreli metinde "B" harfi olarak görünür. Bu durum, şifreleme algoritmasının zayıf olmasına yol açar ve çeşitli dil kuralları ile şifre kırılabilir. Örneğin harflerin bir dildeki kullanım sıklıklarından bir ipucu olarak yararlanılabilir. Türk alfabesindeki harflerin kullanım sıklığı Tablo 3.1'de verilmiştir.

Tablo 3.1: Türk Alfabesindeki Harflerin Kullanım Sıklığı

Harf	Kullanım Sıklığı (%)	Harf	Kullanım Sıklığı (%)	Harf	Kullanım Sıklığı (%)	Harf	Kullanım Sıklığı (%)
A	11,920	**Ğ**	1,125	**N**	7,487	**U**	3,235
B	2,844	**H**	1,212	**O**	2,476	**Ü**	1,854
C	0,963	**I**	5,114	**Ö**	0,777	**V**	0,959
Ç	1,156	**İ**	8,600	**P**	0,886	**Y**	3,336
D	4,706	**J**	0,034	**R**	6,722	**Z**	1,500
E	8,912	**K**	4,683	**S**	3,014		
F	0,461	**L**	5,922	**Ş**	1,780		
G	1,253	**M**	3,752	**T**	3,314		

Tablo 3.1'e göre en sıklıkla kullanılan harf, %11,92 ile "A" harfidir. En az kullanılan harf ise %0,034 ile "J" harfidir.

Örnek

Sezar algoritması ile şifrelenmiş bir şifreli metin örneği verilmiştir.

> *"lçtçşçf içtegzotig, djyösjy ğöslözgegy gmsgyotig zuyctzcf ğöy ajrösij ösjysjyrjt dj iöpöbgs öasjşsjy lüyçsşjşöa ğöy nofsg ljyıjrsjaöyrjt, zgmsgş zöğjy lçdjtsör ütsjşsjyötj usgt önböegı ütjşsö ğöy önböegı nâsötö gsşoaboy. ryöbör djyösjyöt ruyctşgzo, djyö löfsösömöt zgmsgtşgzo dj iöpöbgs ösjböaöşöt ğçbçtsçmçtçt jsij jiössjzö, lçt ljıbörıj ıjaöbsjtjt zöğjy bjniöbsjysj rgyao ütjşsö fuyscrsgy usgygr uybgeg ıorşoaboy. ğöyjezjs djeg rcycşzgs zöğjy lçdjtsömö lçısjtiöyşjr öıöt rcssgtosgt ğöyıur gygı dj bjrtör gygzotig, ryövbulygkör bjrtörsjy löijyjr igng ig ütjşsö ngsj ljsşöa dj djyösjyö dj ösjböaöşö ruycşgr öıöt ljyjrsö gygısgyo zgmsgşoaboy."*

Örnekteki şifreli metnin harf sıklık tablosu çıkarılırsa en sık kullanılan harflerin düz metindeki "A" harfi olma ihtimali yüksektir. Şifreli metin ne kadar uzun olursa bu ihtimal o kadar artacaktır. Metinde en sık kullanılan harfler ve harflerin kullanım sayısı Tablo 3.2'de verilmiştir.

Tablo 3.2: Örnek Metinde En Sık Kullanılan Harfler ve Harflerin Kullanım Sayısı

ö	j	s	g	t
69	63	52	46	35

Sezar şifrelemesi mono alfabetik bir şifreleme olduğuna göre Tablo 3.2'de "ö" harfi, şifreleme sonucunda "a" harfi ile eşleşmiş olabilir. Bu olasılıkta düz metindeki 1. harf (a), şifreli metindeki 19. harf (ö) olmuştur. Bir başka deyişle Sezar algoritmasının anahtar değeri 18'dir. Bu ihtimale göre şifreli metindeki birkaç kelime incelendiğinde anlamlı bir ifade elde edilemediği görülür ancak harf sıklığında 4. sıradaki "g" harfi "a" olarak alındığında anahtarın 7 olduğu görülür. Her harf 7 öncesi ile değiştirildiğinde anlamlı düz metin elde edilir. Dolayısıyla çok daha az deneme yanılma sayısı ile şifreli metin çözülebilir. Çözülmüş metin şu şekildedir:

> *"günümüz dünyasında, veriler bilgisayar ağlarında sorunsuz bir şekilde ilerlerken ve dijital işlemler görülmemiş bir hızla gerçekleşirken, sağlam siber güvenlik önlemlerine olan ihtiyaç önemli bir ihtiyaç halini almıştır. kritik verilerin korunması, veri gizliliğin sağlanması ve dijital iletişimin bütünlüğünün elde edilmesi, gün geçtikçe çeşitlenen siber tehditlerle karşı önemli zorluklar olarak ortaya çıkmıştır. bireysel veya kurumsal siber güvenliği güçlendirmek için kullanılan birçok araç ve teknik arasında, kriptografik teknikler giderek daha da önemli hale gelmiş ve verileri ve iletişimi korumak için gerekli araçları sağlamıştır."*

Benzer şekilde çeşitli dil kuralları (Bir kelimede üç sesli harf yan yana olmaz, dört sessiz harf yan yana olmaz vb.) kullanılarak da deneme yanılma sayısı azaltılabilir. Yine "ve", "bu" gibi sık kullanılan harf grupları, deneme yanılma sayısını azaltarak çözümü daha kısa sürede bulmayı sağlayabilir.

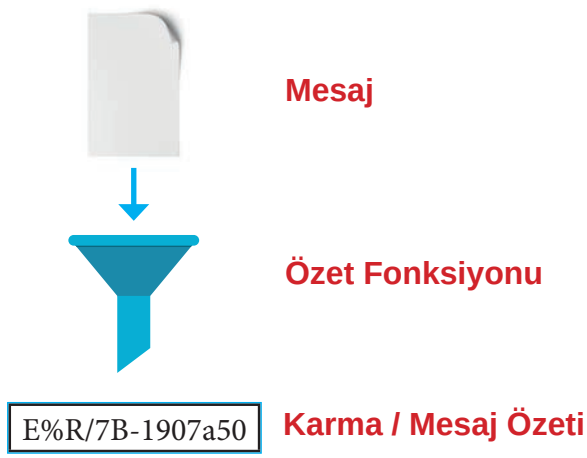
Vigenère şifrelemesi, Sezar şifrelemesine göre daha güvenli olsa da modern sistemlerde güvenli bir yapı oluşturmaktan oldukça uzaktır. Benzer şekilde frekans analizleri ve dil kuralları yine burada kırılması için ipuçları sağlar. Vigenère şifrelemesinde, düz metindeki bazı istatistiksel özellikler kaybolur. Örneğin tek harfler yerine çift veya üçlü harf grupları ortaya çıkar. Bu istatistiksel düzensizlikler, anahtarın uzunluğunun ve dolayısıyla anahtarın tahmin edilmesini kolaylaştırır.

Günümüzde klasik şifreleme yerini daha karmaşık ve güvenli modern şifreleme yöntemlerine bırakmıştır. Bu modern şifreleme yöntemlerinin yanı sıra kriptografide önemli bir husus olan anahtarın güvenli bir şekilde değişimi için de çeşitli teknikler geliştirilmiştir. Son yıllarda ortaya çıkan kuantum kriptografisi, geleneksel kriptografi yöntemlerine kıyasla daha güçlü güvenlik özellikleri sunar. Kuantum kriptografisinin, kuantum fiziğinin temel özelliklerini kullanarak anahtar değişimi ve şifreleme süreçlerinde güçlü özellikler sunması beklenir. Kuantum kriptografisinin güvenlik temelinde kuantum mekaniğinin belirsizlik ilkesi yer alır. Kuantum fiziği, belirli bir ölçüm yapılmadığı sürece bir parçacığın durumunun kesin olarak belirlenemeyeceği gerçeğine dayanır. Bu özellik, kuantum kriptografisine uygulandığında anahtar dağıtımı sırasında izinsiz dinlemelerin veya saldırıların tespitinin mümkün olacağı ön görülebilir.

3.3. HASH FONKSİYONU

Siber güvenliğin üç temel unsurlarından biri olan bütünlük, verinin değiştirilmediğinin anlaşılmasını ifade eder. Veri bütünlüğünü sağlamak için **özet (hash) fonksiyonları** kullanılır. Özet fonksiyonlarına **karma fonksiyonları** da denir. Özet fonksiyonu, kriptografide bir girdinin (mesaj, veri veya dosya), hash değeri veya hash kodu olarak bilinen sabit boyutlu bir karakter dizisine dönüştürülmesi işlemidir. Özet fonksiyonu, bu dönüşümü giriş verilerine matematiksel bir algoritma uygulayarak gerçekleştirir. Özet fonksiyonu sonucunda elde edilen değer, orijinal mesajın (veya dosyanın) sabit uzunluktaki bir parmak izi olarak düşünülebilir. Bu nedenle bazen mesaj özeti (message digest) olarak adlandırılır.

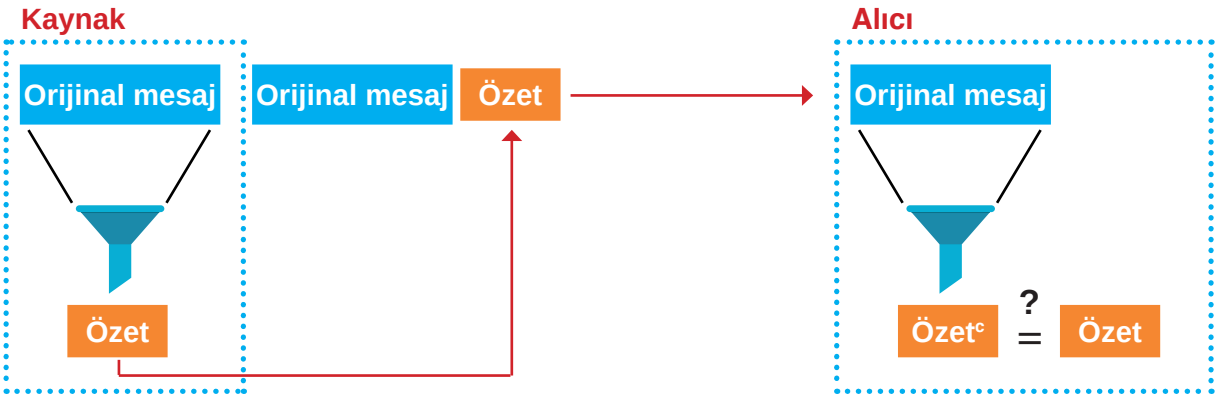
Görsel 3.7'de görüldüğü gibi bir özet fonksiyonu, değişken boyutlu bir mesajdan sabit uzunlukta bir mesaj özeti üretmek için kullanılır.



Görsel 3.7: Bütünlük için özet fonksiyonunun kullanılması

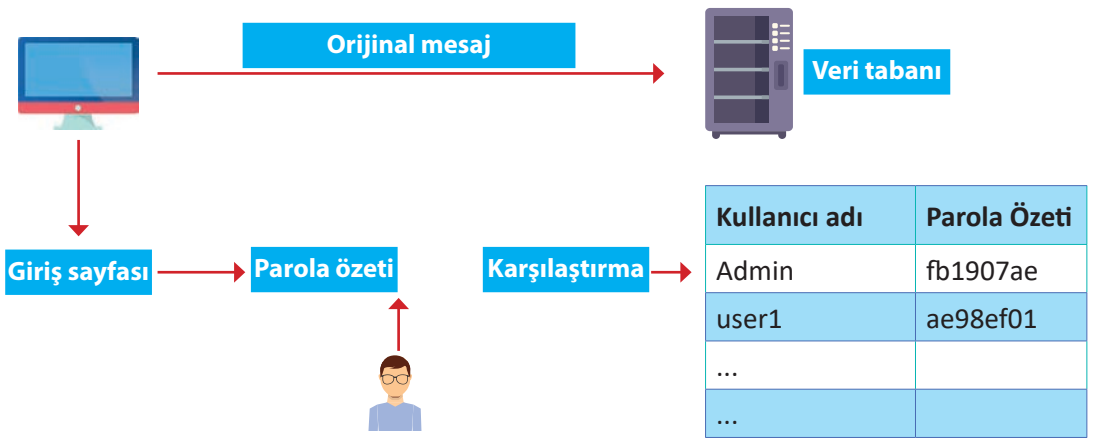
En yaygın kullanılan mesaj özetlerinin uzunluğu 160-512 bit arasındadır. Özet fonksiyonu, sıkıştırma yeteneğine sahip olmalıdır. Ayrıca özet fonksiyonu herhangi bir girdi için kolayca çıktıyı hesaplayabilmelidir. Özet fonksiyonları, matematiksel olarak tek yönlü fonksiyonlardır. Bu nedenle bu fonksiyonların tersi yoktur ve şifreleme amaçlı kullanılmaz. Bir başka deyişle bir özet değerinden fonksiyonun tersi kullanılarak orijinal mesaj elde edilemez ancak daha önce hesaplanmış bir özet değerinden yola çıkılarak orijinal mesajın ne olduğu tespit edilebilir.

Özet fonksiyonu, iletişimde verinin orijinalliğinin korunup korunmadığını belirlemek için kullanılabilir. Görsel 3.8'de görüldüğü gibi gönderici, orijinal mesajın özetini hesaplar ve mesaja ekleyip gönderir. Alıcı, mesajı alır ve mesaj için aynı özet fonksiyonunu hesaplar (Görsel 3.8'de Özet_C olarak gösterilmiştir.). Hesaplanan özet değer ile mesaja eklenen değer eşitse bu durumda mesajın orijinalliğinin değişmediği farz edilir.



Görsel 3.8: Mesajın değiştirilip değiştirilmediğini anlamak için özet fonksiyonunun kullanılması

Özet fonksiyonu çeşitli web sayfalarından dosya indirilirken de kullanılır. Dosyanın indirileceği kaynak web sitesinde MD5 veya SHA özet değerleri yer alır. Dosya indirildikten sonra özet hesaplayıcı yazılımlar ile indirilen dosyanın özeti hesaplanır. Hesaplanan değer ile web sitesinde yer alan değer aynı ise dosyanın indirilirken değiştirilmediği anlaşılabilir. Diğer bir kullanım senaryosu da Görsel 3.9'da görüldüğü gibi veri tabanında parolaların açık bir şekilde kaydedilmemesi mantığına dayanır. Bu senaryoda, veri tabanında parola gibi kritik bilgilerin özet değerleri saklanır. Böylece veri tabanı, saldırganlar tarafından ele geçirilse veya ağ trafiği dinlense de parolanın kolaylıkla elde edilememesi sağlanır.



Görsel 3.9: Özet fonksiyonlarıyla çeşitli uygulamalarda güvenlik sağlanması

Özet fonksiyonları teorik olarak çok çeşitli boyutlarda girdiyi alır ve sabit uzunlukta çıktılar üretir. Çıktının (özet değeri) sabit bir uzunlukta, girdinin ise çok daha geniş bir uzunlukta olabilmesi nedeniyle birden fazla farklı mesajın aynı özet değeri elde etmesi olasıdır. Buna **çakışma (collision)** denir. Bir özet fonksiyonunda çakışmanın çok düşük olasılıkta gerçekleşmesi istenir. Güvenli özet fonksiyonları şu özellikleri desteklemelidir:

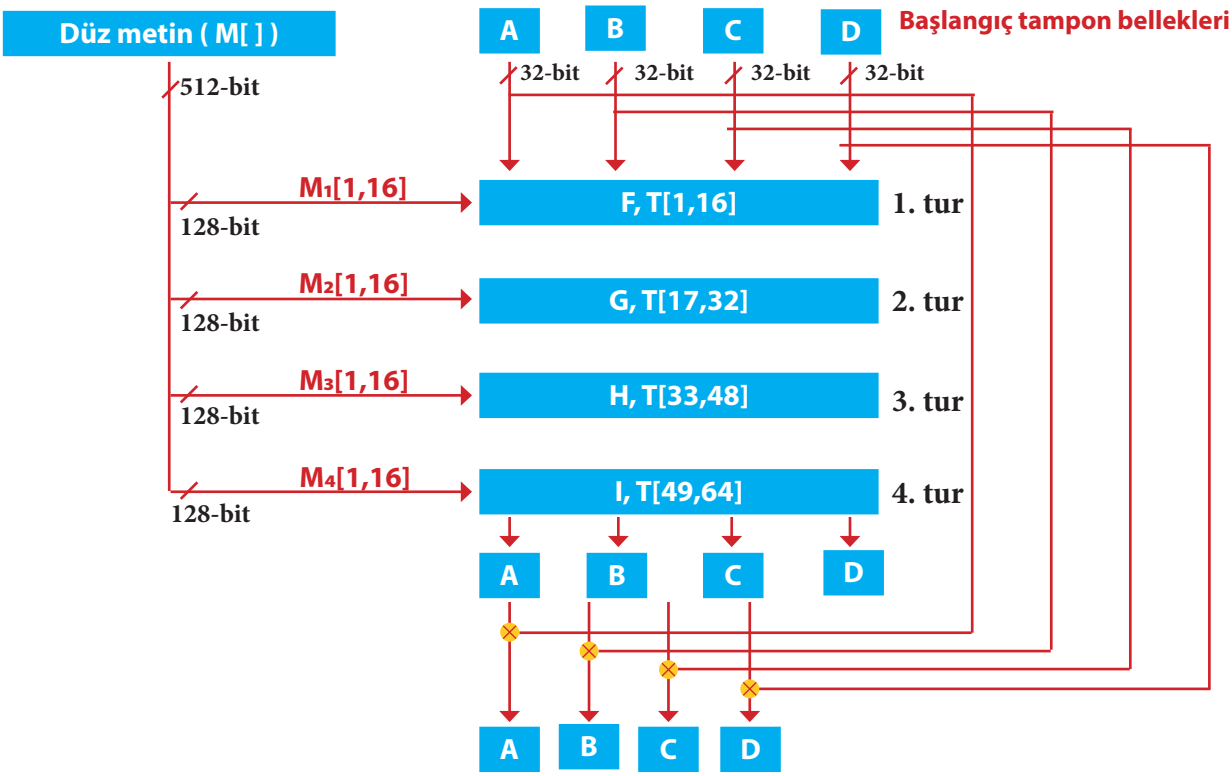
- Özet fonksiyonu tek yönlü olmalıdır. Bir başka deyişle herhangi bir özet değerinden orijinal mesajın elde edilmesi son derece güç olmalıdır.

- Bir özet fonksiyonu ve özet değeri verildiğinde aynı özet değeri üreten farklı bir mesaj bulmak son derece güç olmalıdır.
- Özet fonksiyonu çakışmaya karşı dirençli olmalıdır. Bir mesaj ve mesaja ait özet değeri verildiğinde aynı değeri üreten başka bir mesajı bulmak son derece güç olmalıdır.

Sonuç olarak özet; veri bütünlüğünü sağlamak, parolaları güvence altına almak, veri aramayı kolaylaştırmak, dijital imzaları etkinleştirmek, veri depolama ve alma süreçlerini optimize etmek için güvenilir ve verimli bir yol sağlar. Özet fonksiyonları, modern kriptografi ve veri yönetimi sistemlerinde kullanılan temel bir araçtır. MD5 ve SHA, en yaygın kullanılan özet fonksiyonlarıdır.

3.3.1. Veri Özeti 5 (Message Digest 5-MD5)

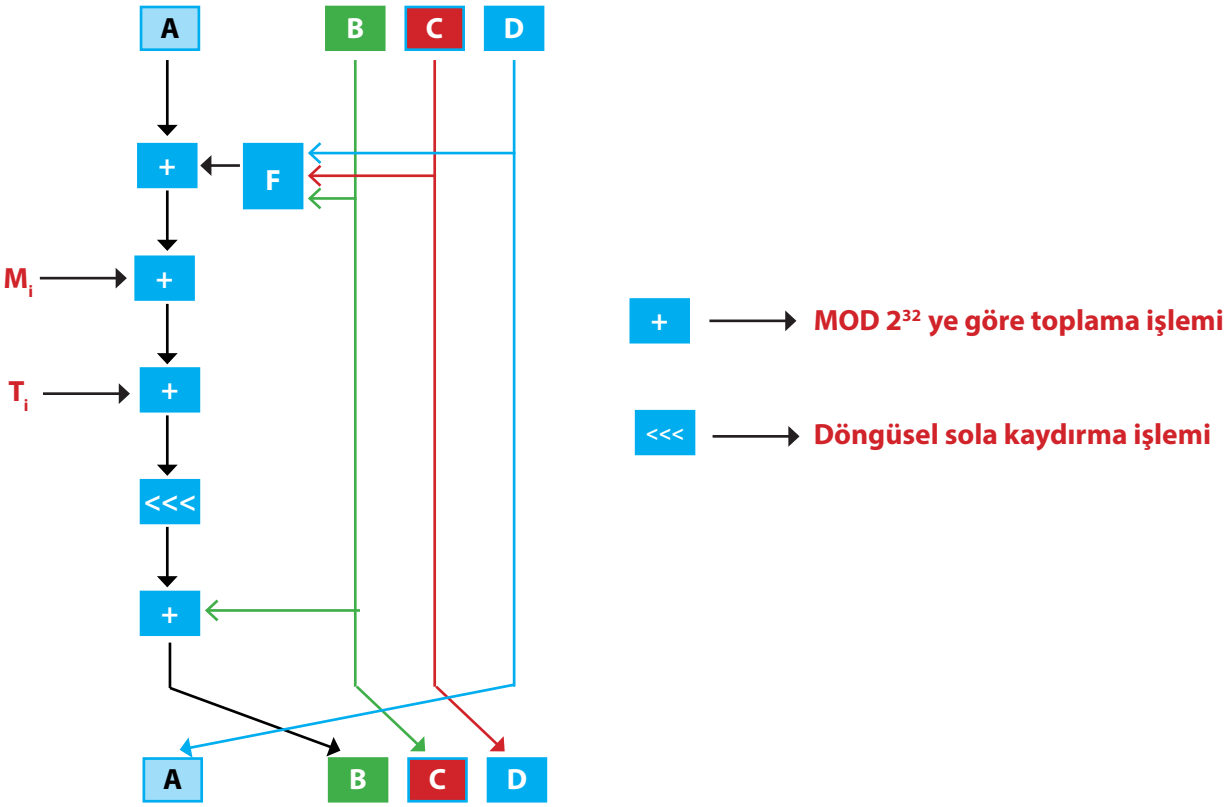
Ron Rivest tarafından geliştirilen MD (Message Digest), en çok bilinen kriptografik özet fonksiyonlarından biridir ve içinde birçok işlemi barındırır. 1989 ve 1991 yılları arasında MD2, MD4 ve MD5 fonksiyonları tanıtılmıştır. Özellikle MD5, uzun yıllar boyunca güvenli bir bütünlük sağlayıcı fonksiyon olarak yaygınca kullanılmıştır. MD5, 128-bitlik özet değeri (mesaj özeti) üreten bir fonksiyondur. Bu durumda 2^{128} farklı mesaj özeti elde edilebilir. Görsel 3.10'da MD5 özet fonksiyonunun çalışma diyagramı verilmiştir.



Görsel 3.10: MD5 özet fonksiyonunun çalışma prensibi

MD5 özet fonksiyonunun çalışma prensibinde her biri 32-bit olan dört adet tampon bellek yer alır. **A, B, C** ve **D** olarak adlandırılan bu tampon belleklerde, başlangıçta bilinen değerler bulunur. Düz metinden parça parça alınan verilere göre (Görsel 3.10'da M_1, M_2... olarak gösterilmiştir.) dört farklı tur sonunda bu tampon bellekteki değerler değiştirilir. Her bir turda 16 adım (toplamda 64 adım) bulunur. Her bir turda mantıksal bit fonksiyonları (F, G, H ve I) ve tur sabitleri (T) ile düz metnin 128-bitlik parçaları (M) tarafından tampon bellekteki değerler değiştirilir. Ardından başlangıçtaki tampon bellek değerleri ile son durumdaki tampon bellek değerleri XOR'lanır ve 128-bitlik özet değeri elde edilir.

Düz metin, 512-bitlik bloklar şeklinde alınır ve her bir turda kullanılmak üzere 128-bitlik (16 bayt) kısımlara ayrılır. Düz metnin kısımları, tur sabitleri ile mantıksal işleme tabi tutulur. Dört turun her birinde on altışar adım bulunur. Görsel 3.11'de sadece **F** fonksiyonunun kullanıldığı bir adımın nasıl gerçekleştiği verilmiştir. Diğer turlarda sırasıyla **G, H** ve **I** fonksiyonları kullanılır. Bu fonksiyonlar; VE, VEYA, DEĞİL, XOR gibi mantıksal işlemlerden oluşur.



Görsel 3.11: MD5 algoritmasının bir adımında gerçekleşen işlemler

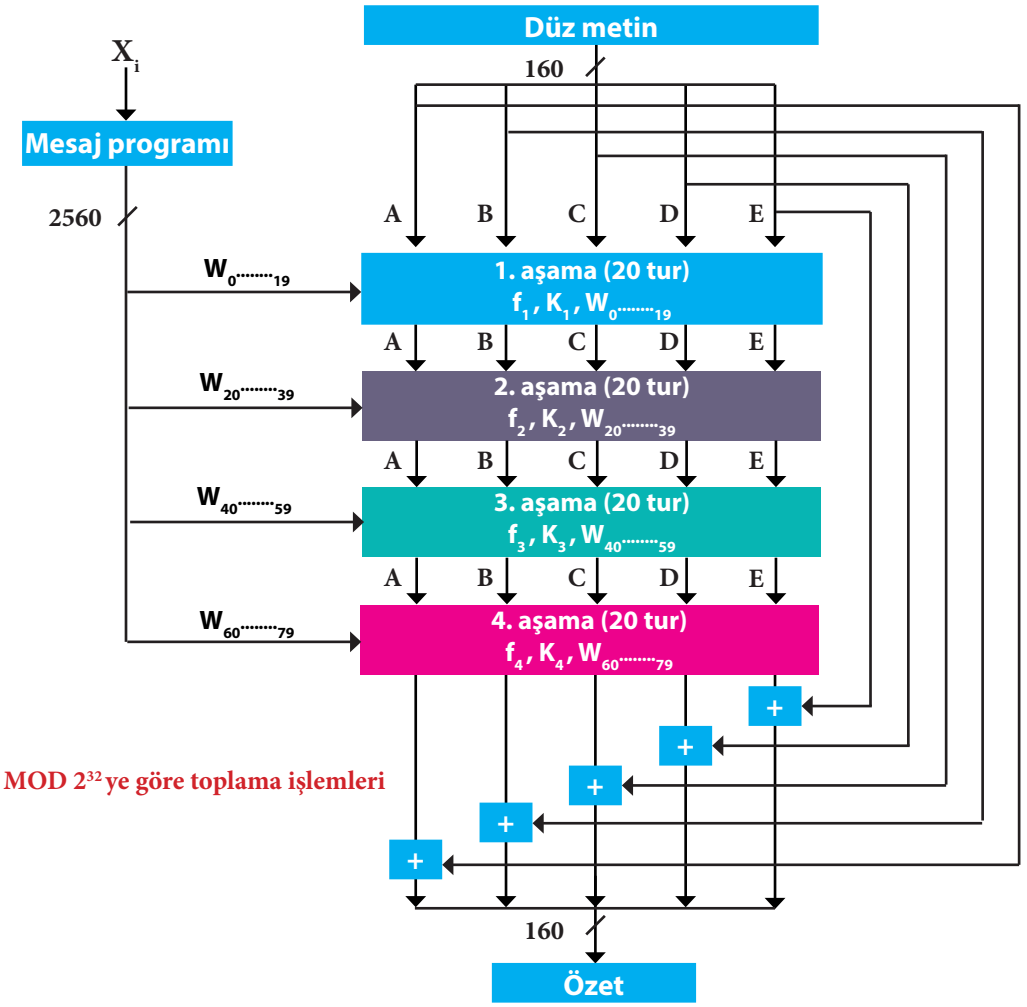
Toplamda 64 adım ve XOR işleminin ardından A, B, C ve D belleklerindeki değerler, 512-bitlik düz metnin özet değerini verir ancak düz metnin boyutu 512-bitten daha fazla ise bu durumda tampon belleklerin değerleri bir sonraki 64 adım için başlangıç tampon değerini oluşturur. Çeşitli çevrimiçi web sayfalarında veya ücretsiz yazılımlarda MD5 mesaj özeti hesaplanabilir.

1995 yılında yapılan bazı çalışmalarda MD5'in çakışmaya karşı yeterince dirençli olmadığı ortaya çıkmıştır. Bu nedenle günümüzde MD5 yerine daha güvenli bir fonksiyon olan SHA'nın kullanılması önerilir.

3.3.2. Güvenli Özet Algoritması (Secure Hash Algorithm-SHA)

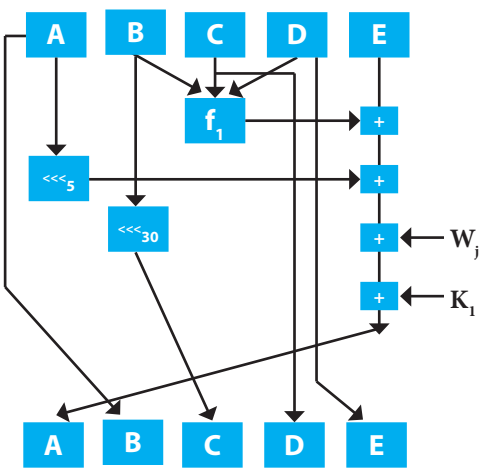
SHA algoritması, değişken uzunluktaki verilerden sabit uzunlukta özet değerleri oluşturmak için yaygın olarak kullanılan bir kriptografik özet fonksiyonudur. Tek yönlü bir fonksiyon şeklinde tasarlanmıştır. SHA ailesi, versiyon 1 ve versiyon 2 olmak üzere ikiye ayrılır. SHA-1, versiyon 1 ailesine aitken SHA-256, SHA-384 ve SHA-512, versiyon 2 ailesine aittir.

Görsel 3.12'de 80 turdan oluşan SHA-1 özet fonksiyonunun çalışma diyagramı verilmiştir.



Görsel 3.12: SHA algoritmasının çalışma prensibi

SHA fonksiyonu 160-bitlik düz metinleri alır, bir mesaj programı (W) doğrultusunda A, B, C, D ve E yazmaçlarındaki değerleri değiştirir. Başlangıçta yazmaçlarda yer alan değerler, bilinen sabit değerlerdir. Her aşamada farklı bir fonksiyon (f^1, f^2...) ve sabitler (K) kullanılarak 32-bitlik yazmaçlardaki bu değerler değiştirilir. Görsel 3.13'te herhangi bir aşamadaki bir turun çalışma diyagramı verilmiştir.



Görsel 3.13: SHA adım işlemleri

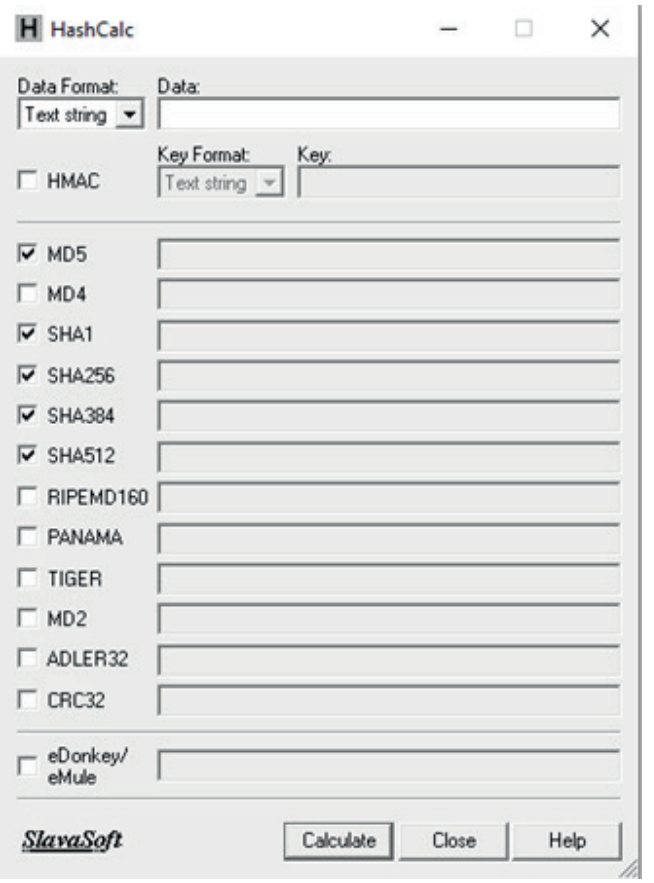
MD5 yapısında olduğu gibi SHA yapısında da F fonksiyonları mantıksal bit işlemleridir. SHA yapısında sola döngüsel bit kaydırma işlemleri yapılır. Son olarak beş yazmaçtaki değerler, ilk değerler ile MOD 32 toplama işlemine tabi tutulur ve 160-bitlik özet değeri elde edilir.

SHA için dört farklı seçenek bulunur. Bu seçenekler arasındaki temel farklar, özet boyutlarına ve kullanılan kriptografik işlemlere göre değişkenlik gösterir. **SHA-1**, 160-bitlik bir özet değeri üretir. En eski ve en az güvenli olanıdır. Bazı eski sistemlerde hâlâ kullanılır ancak modern kriptografik uygulamalarda kullanımı önerilmez. **SHA-256**, 256-bitlik bir özet değeri üretir. SHA-1'e kıyasla önemli ölçüde daha iyi güvenlik sunar. Genellikle veri bütünlüğü kontrolleri, dijital imzalar ve çeşitli kriptografik protokoller için kullanılır. **SHA-384**, 384-bitlik bir özet değeri üretir. Daha büyük bir özet boyutuna sahip olduğu için SHA-256'ya kıyasla daha yüksek güvenlik sağlar. **SHA-512**, 512-bitlik bir özet değeri üretir. Bu dört seçenek arasında en büyük özet boyutunu sunar ve daha yüksek düzeyde güvenlik sağlar.

5. Uygulama

İşlem adımlarına göre özet algoritmalarını kullanarak çeşitli metinlerin MD5 ve SHA özetlerini elde ediniz. Bunun için HashCalc masaüstü yazılımını kullanınız.

1. Adım: Uygulamayı indirip bilgisayarınıza kurunuz. Uygulamayı açtığınızda Görsel 3.14'teki gibi bir ekran ile karşılaşacaksınız.



Görsel 3.14: HashCalc yazılımı ana ekranı

2. Adım: Veri biçimi için Data Format açılır kutusundan Text string seçeneğini seçiniz.

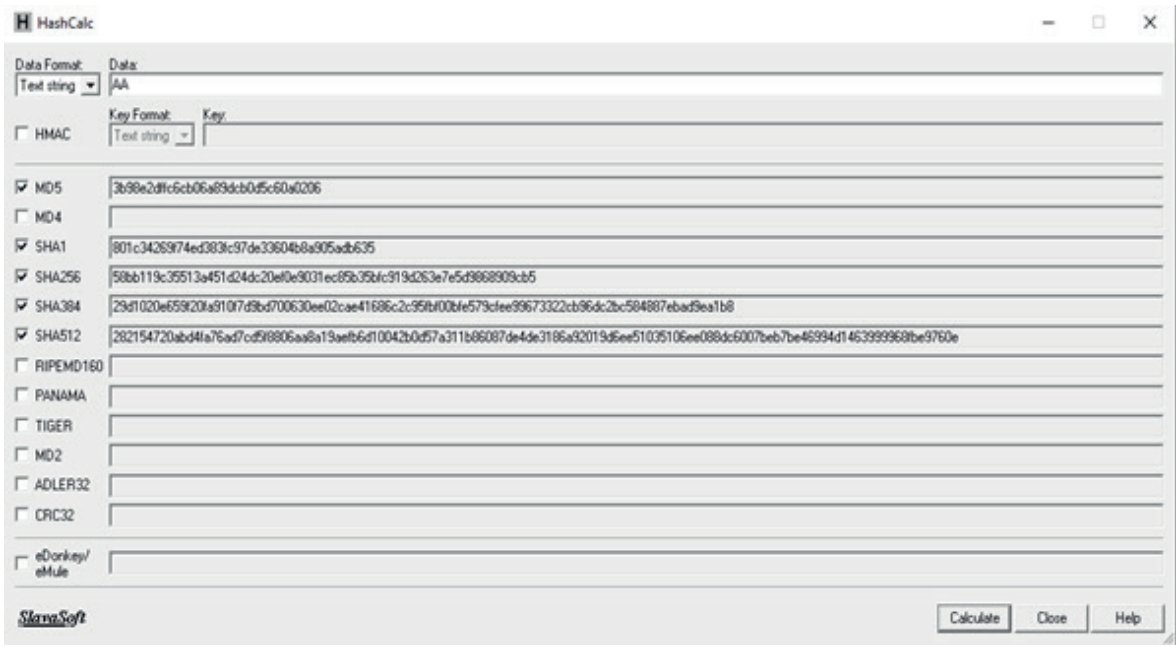
3. Adım: Özet algoritmaları için MD5 ve SHA1, SHA256, SHA384, SHA512 seçiniz.

4. Adım: Data alanına sadece "A" yazınız ve Calculate düğmesine tıklayarak özet değerlerini hesaplayınız. Seçilen özet fonksiyonlarının sonuçları Görsel 3.15'teki gibi görüntülenir.

Görsel 3.15: "A" harfinin farklı özet algoritmalarındaki görüntüsü

5. Adım: Özet fonksiyonlarının değerlerini uzunluk yönünden karşılaştırınız. MD5 özet değerinin 32 adet onaltılık (hexadecimal) sayılardan oluştuğuna dikkat ediniz. Bu karakterlerin her biri 4-bit olduğu için 128-bit çıktı üretir.

6. Adım: Data alanına "AA" harflerini yazıp tekrar hesaplayınız. Görsel 3.16'da "AA" metninin özet değerleri verilmiştir.



Görsel 3.16: "AA" harflerinin farklı özet algoritmalarındaki görüntüsü

7. Adım: "A" harfinin özet değerleri ile "AA" harflerinin özet değerleri arasında doğrudan bir ilişki olmadığını gözlemleyiniz.

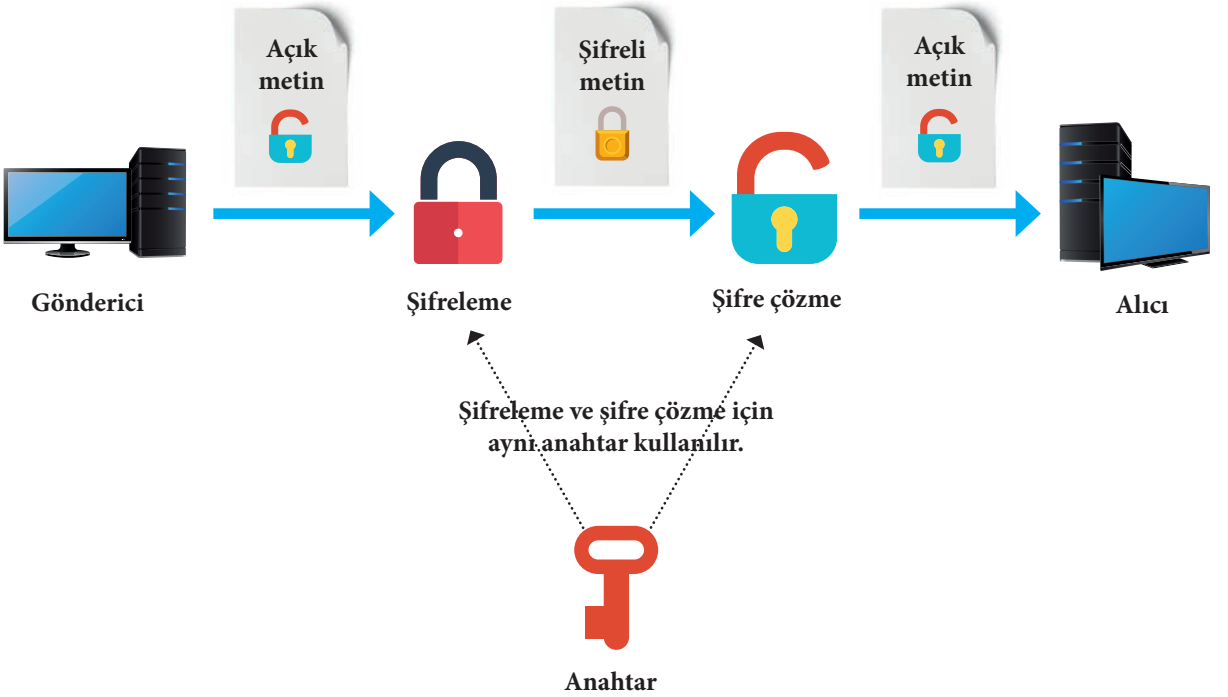
8. Adım: Çeşitli harf, rakam ve kelimeleri kullanarak özet değerlerini karşılaştırınız.

Araştırma

Bilgisayardaki herhangi bir dosyanın MD5 ve SHA özet değerlerini çevrimiçi hesaplayan sayfaları araştırınız. Elde ettiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

3.4. SİMETRİK KRİPTOGRAFİ

Bir düz metnin şifrelenmesinde ve deşifre edilmesinde aynı anahtarın kullanılmasına **simetrik kriptografi** denir. Görsel 3.17'de hem göndericinin hem de alıcının aynı anahtarı kullanması verilmiştir.



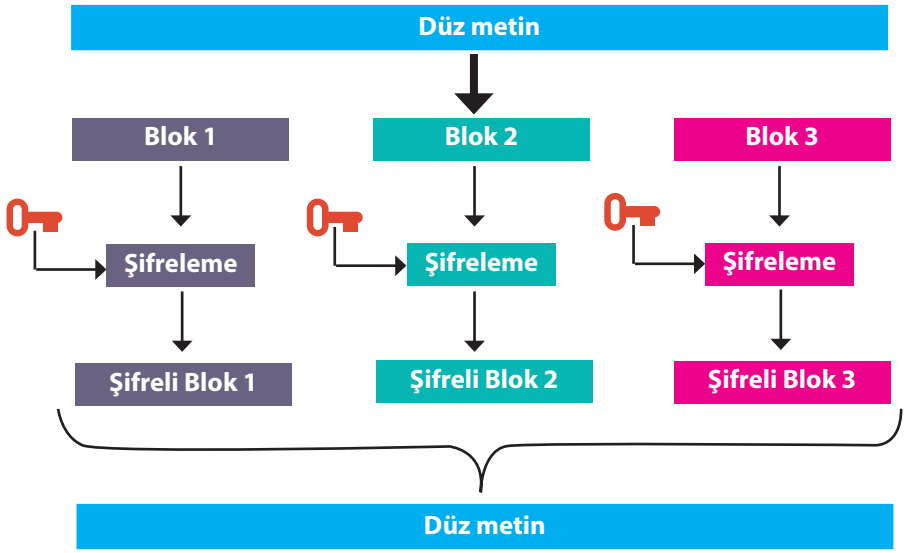
Görsel 3.17: Simetrik şifrelemede aynı anahtarın hem şifreleme hem deşifreleme için kullanılması

Simetrik şifrelemede kullanılan anahtarın gönderici, alıcı veya alıcılar arasında paylaşılması ve paydaşlarca gizli tutulması gerekir. Bu nedenle simetrik şifrelemede kullanılan anahtara **paylaşılan anahtar (shared key)** veya **paylaşılan gizli anahtar (shared secret key)** denir.

3.4.1. Blok Şifreleme

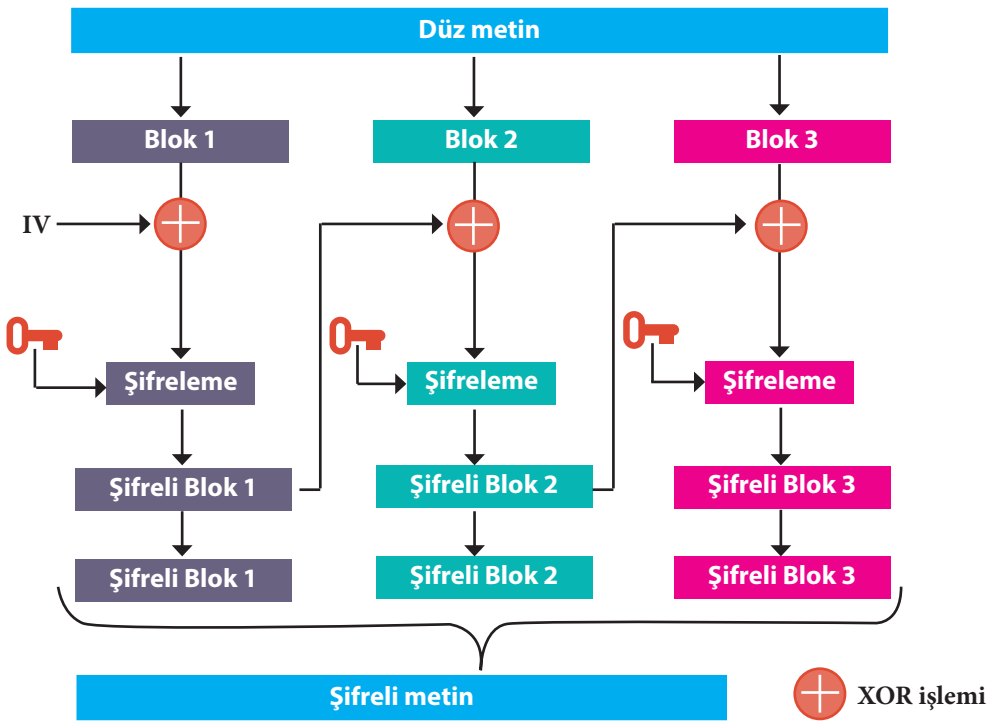
Şifreleme algoritmalarında akış şifreleme ve blok şifreleme olmak üzere iki tür şifreleme görülür. Blok şifrelemede düz metin 64-bit, 128-bit, 256-bit gibi sabit boyutlardaki bloklara bölünür ve her bir blok ayrı ayrı şifrelenir. Blok şifrelemede bloklar ayrı ayrı işlendiğinden dolayı tamamlanması için tüm blokların kullanılması gerekir.

Günümüzde ağ tabanlı simetrik kriptografik uygulamaların çoğunda blok şifreleme kullanılır. Blok şifrelemede çeşitli çalışma modları yer alır. Elektronik Kod Kitabı (Electronic Code Book-ECB), en basit blok şifreleme modlarından biridir. Görsel 3.18'de görüldüğü gibi her bir blok, aynı anahtar ile şifrelenir. Şifre çözme işlemi ise Görsel 3.18'in ters sıralamasıyla gerçekleşir.



Görsel 3.18: ECB'nin çalışma prensibi

Şifre Blok Zinciri (Cipher Block Chaining-CBC) algoritmasının çalışma şeklinde ise bir blokta yapılan şifreleme işleminin çıktısı, diğer bloka girdi olarak eklenir. Görsel 3.19'daki diyagramda CBC'nin çalışma prensibi verilmiştir.



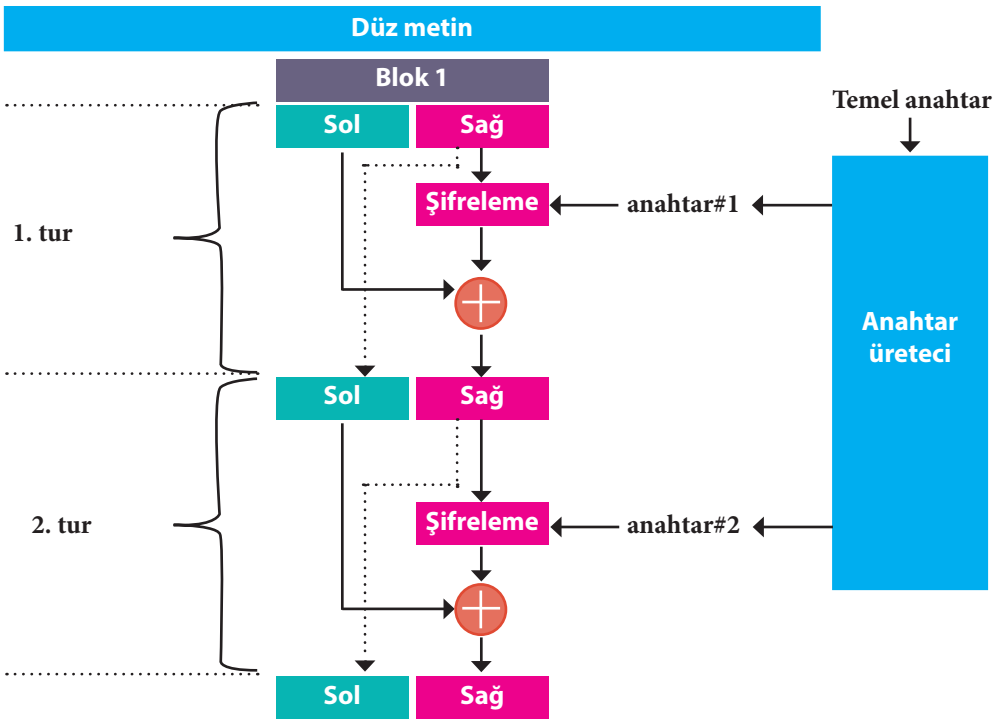
Görsel 3.19: CBC'nin çalışma prensibi

CBC'deki bloklarda aynı anahtar kullanılır. Düz metnin ilk blokundaki şifreleme işleminin sonucunda elde edilen değer önce düz metnin ikinci bloku ile XOR işlemine tabi tutulur, ardından şifreleme işlemi gerçekleştirilir. Bir başka deyişle önceki blokun şifrelenmiş hâli ve mevcut blokun kendi olmak üzere her blokun iki girdisi vardır. İlk blokun öncesi olmadığı için sabit bir **ilklendirme dizisi (Initialization Vector-IV)** kullanılır.

CBC modunda ilk bloktaki herhangi bir değişiklik diğer blokları da etkileyecektir. **Çığ etkisi (avalanche effect)** olarak adlandırılan bu özellik, şifreleme algoritmalarında istenen bir özelliktir. Bir başka deyişle güvenli bir şifreleme algoritmasından beklenen, açık metindeki veya anahtardaki küçük bir değişikliğin şifreli metinde büyük oranda bir değişiklik oluşturmasıdır.

Güvenli bir blok şifreleme algoritmasının uygulanması kolay ancak kırılması zor olmalıdır. Ayrıca düz metin ile şifreli metin arasında istatistiksel bir ilişkinin olmaması beklenir. Sezar ve Vigénere şifrelemelerinde bu istatistiksel ilişkiler kırılabilir. 1970'li yıllarda Horst Feistel tarafından hem yerine koyma hem de permütasyon içeren blok tabanlı bir şifreleme önerildi. **Feistel Blok Şifrelemesi** olarak adlandırılan bu model, **karışıklık (confusion)** ve **yayılma (diffusion)** sağlıyordu. Karışıklık, şifreli metin ile anahtar arasındaki ilişkiyi olabildiğince karmaşık hâle getirmeyi amaçlar. Yayılma ise düz metin ile şifreli metin arasında istatistiksel bir ilişki kurulamamasını amaçlar. Bir başka deyişle anahtarın ve düz metnin her bitinin, şifreli metni tamamen etkilemesi ve değiştirmesi hedeflenir.

Görsel 3.20'de iki tekrarlı Feistel blok şifrelemesi örneği verilmiştir.



Görsel 3.20: Feistel blok şifrelemesi örneği

Feistel blok şifrelemesinde bir blok öncelikle ikiye ayrılır. Ardından bu iki parçaya birkaç tur (round) işlem yapılır. Feistel blok şifrelemesinde her bir blok için farklı bir anahtar kullanılır. Her bir tur için temel bir anahtardan algoritmik olarak tur anahtarları üretilir. Bu anahtarlar Görsel 3.20'de **anahtar#1, anahtar#2** olarak gösterilmiştir.

Her bir tur içinde yapılan işlemler şu şekilde sıralanabilir:

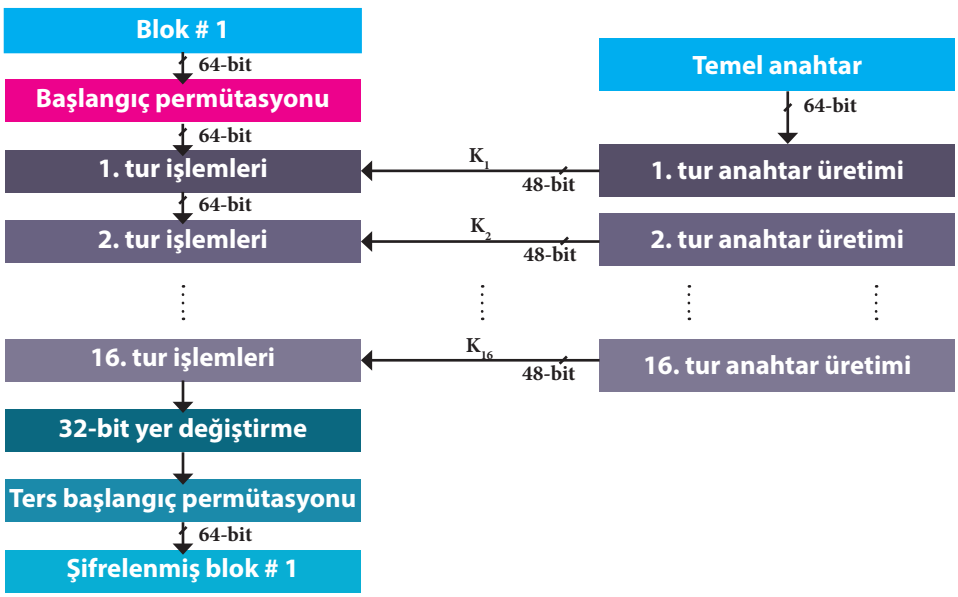
- Blok sağ ve sol olmak üzere ikiye ayrılır.
- Sağ blok bir sonraki turda sol blok olarak kullanılmak üzere aynen aktarılır.
- Anahtar üretecinden her bir tur için bir tur anahtarı üretilir.
- Tur anahtarı ile sağ blok şifrelenir. Şifreleme sonucunda elde edilen çıktı, sol blok ile XOR işlemine tabi tutulur. XOR sonucunda elde edilen değer, bir sonraki tur için sağ blok olarak kullanılır.

Feistel tarafından önerilen bu model, uzun yıllar standart bir modern şifreleme algoritması olan DES şifrelemesinin temelini oluşturdu.

3.4.2. DES Kripto Algoritması

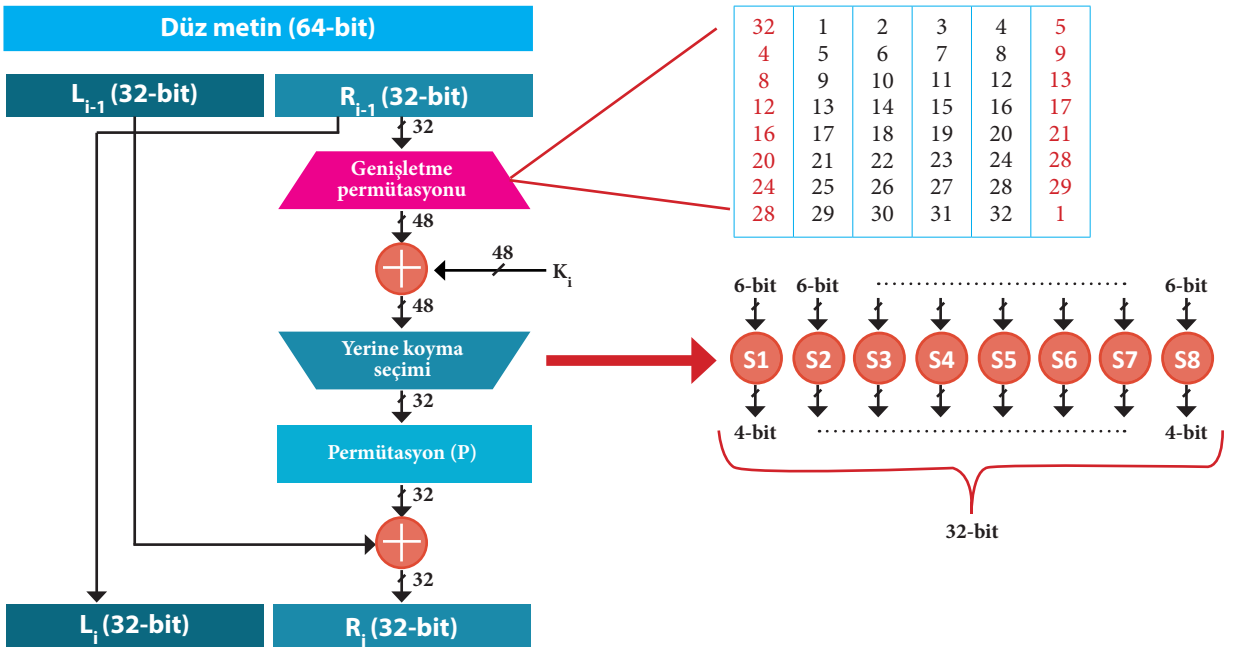
Veri Şifreleme Standardı (Data Encryption Standard-DES), 1970'lerde geliştirilen, 1977'de ABD Ulusal Standartlar ve Teknoloji Enstitüsü (NIST) tarafından standardize edilen simetrik bir şifreleme algoritmasıdır. 2000'li yıllara kadar güvenli iletişim ve veri koruması için yaygın olarak kullanıldı ancak bilgi işlem gücündeki ilerlemeler nedeniyle yerini daha güvenli algoritmalar aldı. DES, 64-bit uzunluğundaki veri blokları üzerinde çalışır ve 56-bitlik şifreleme anahtarları kullanır. Algoritma, anahtara dayalı olarak giriş verilerine uygulanan bir dizi permütasyon ve yerine koyma işleminden oluşur. DES şifrelemesinde 16 tur bulunur. Her turda yapılan bu işlemler, bir saldırganın doğru anahtarı bilmeden verilerin şifresini çözmesini zorlaştırır.

Görsel 3.21'de görüldüğü gibi DES şifrelemesi, anahtar üretimi ve tur işlemleri olmak üzere iki bölümde incelenir. Anahtar üretimindeki amaç, temel anahtardan her turda kullanılan tur anahtarları elde etmektir. 64-bitlik temel anahtar, çeşitli permütasyon ve bit kaydırma işlemlerinden oluşan tur anahtarı üretim aşamasından geçer. 64-bitlik anahtardan 48-bit uzunluğunda tur anahtarları üretilir. Görsel 3.21'de verilen K1, K2...K16 , 48-bit uzunluğundaki tur anahtarlarıdır. Tur anahtarları her turda şifreleme için bir girdi olarak kullanılır.



Görsel 3.21: DES şifrelemesi anahtar üretimi ve tur işlemleri

DES şifrelemesinde on altı tur işlemi yer alır. Her bir turda çeşitli yerine koyma ve permütasyon işlemleri uygulanır. Yapılan işlemler Görsel 3.22'de diyagram olarak verilmiştir.



Görsel 3.22: Bir DES adımının çalışma prensibi

Düz metin, 64-bitlik bloklar hâlinde alınır ve Feistel şifrelemesinde olduğu gibi her biri 32-bit şeklinde iki parçaya ayrılır. Bu parçalar, diyagramda R ve L olarak ifade edilmiştir. Sağ parça (R), bir sonraki turda sol parça (L) olur. Bir sonraki turun sol parçasını oluşturmak için bir dizi işlem yapılır. Öncelikle **genişletme permütasyonu** kullanılarak 32-bitlik sağ taraf, 48-bite çıkarılır. Ardından üretilen tur anahtarı ile XOR işlemi uygulanır ve 48-bit elde edilir ancak elde edilen sonucun tekrar 32-bite dönüştürülmesi gerekir. Bu nedenle **S-Kutuları** olarak ifade edilen **yerine koyma yöntemi** uygulanır. Sekiz adet S-kutusunun her biri altışar biti alarak dörder bite dönüştürür. Son olarak bir permütasyon uygulanır. On altı turun ardından 64-bitlik şifreli bir blok elde edilir. Düz metnin her 64-bitlik bloku bu işlemlerden geçer ve nihai olarak şifreli metin elde edilir.

DES şifrelemesi, uzun süre standart olarak kullanılmıştır ancak bazı güvenlik zayıflıkları barındırır. Örneğin 56-bitlik anahtar yeterince uzun değildir ve deneme yanılma saldırıları sonucunda göreli olarak kısa sürede kırılabilir. S-kutusu yapısındaki farklı 6-bitlik girdilerin aynı 4-bitlik çıktıyı üretebilmesi bir zayıflıktır. DES yapısındaki bazı anahtarlar (0x0101010101010101 veya 0x0000000000000000 gibi) güvenlik açığı oluşturur. Bu nedenle günümüzde DES şifrelemesinin yeterince güvenli olmadığı varsayılır ve kullanılmaması önerilir.

Araştırma

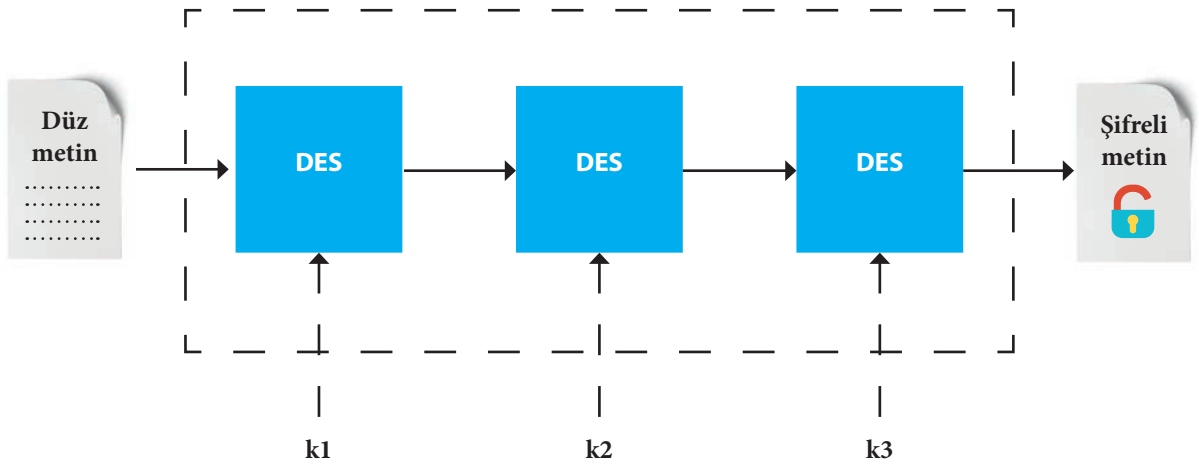
DES şifrelemesini ve deşifrelemesini çevrimiçi yapan güvenli sayfaları araştırınız. Elde ettiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

3.4.3. Üçlü DES (3DES) Kripto Algoritması

DES kripto algoritmasında S-kutularındaki zafiyet ve bazı anahtarların zayıf olması, bu algoritmanın yeterince güçlü olmamasına yol açar. DES algoritmasının deneme yanılma saldırılarına karşı yeterince güvenli olmadığı da ortaya çıkmıştır. **Üçlü DES**, DES algoritmasının art arda üç kez farklı anahtarlar ile uygulanması sonucunda elde edilen bir şifreleme algoritmasıdır. Bir düz metin öncelikle ilk anahtar ile şifrelenir. Elde edilen mesaj, ikinci anahtar kullanılarak deşifre edilir. Son olarak da üçüncü anahtar ile tekrar şifrelenir ve şifreli metin elde edilir.

DES algoritmasının üç kez uygulanması, kaba kuvvet saldırılarına ve çeşitli analitik saldırılara karşı daha güçlü bir koruma sağlar.

Görsel 3.23'te k1, k2 ve k3 anahtarları ile 3DES şifreleme algoritması verilmiştir.



Görsel 3.23: Üçlü DES şifreleme algoritması

Üçlü DES algoritmasında k1=k2=k3 olarak kullanılırsa doğrudan DES uygulanmış olur. Bu durum özellikle 3DES ile DES'in uyumlu olması istenen bazı senaryolar ve eskiye uyum için mantıklıdır.

Üçlü DES blok şifrelemesinde ECB, CBC gibi modlar kullanılarak şifreleme sunulabilir. CBC modundan yararlanıldığında bir ilklendirme vektörünün kullanılması gerekir.

3.4.4. AES Kripto Algoritması

Amerikan Ulusal Standartlar ve Teknoloji Enstitüsü, iki Belçikalı kriptograf Joan Daemen ve Vincent Rijmen tarafından oluşturulan bir algoritma olan **Rijndael'i** yeni standart olarak seçti ve Gelişmiş Şifreleme Standardı (Advanced Encryption Standard-AES) olarak kabul edildiğini duyurdu. Rijndael, AES'in algoritmasıdır.

AES, blok tabanlı simetrik bir şifrelemedir. AES'in blok boyutu 128-bittir, anahtar boyutu ise değişkenlik gösterir. AES şifrelemesinde 128-bit, 192-bit ve 256-bit anahtar uzunlukları kullanılabilir. AES şifrelemesindeki tur sayısı, anahtar uzunluğuna bağlıdır. 128-bit anahtar uzunluğu için 10 tur, 192-bit anahtar uzunluğu için 12 tur ve 256-bit anahtar uzunluğu için 14 tur kullanılır. DES, bit tabanlı çalışırken AES, bayt tabanlı çalışır. Bir başka deyişle 128-bitlik bloklar, baytlara ayrılır ve 16 baytlık bloklar kullanılır. Bu bloklar, 4x4 matrisler olarak ifade edilir. Düz metinler ve anahtarlar öncelikle ASCII kodları olarak ifade edilir, ardından matrislere dönüştürülür.

Örnek

Düz Metin: 29 MAYIS 1453.

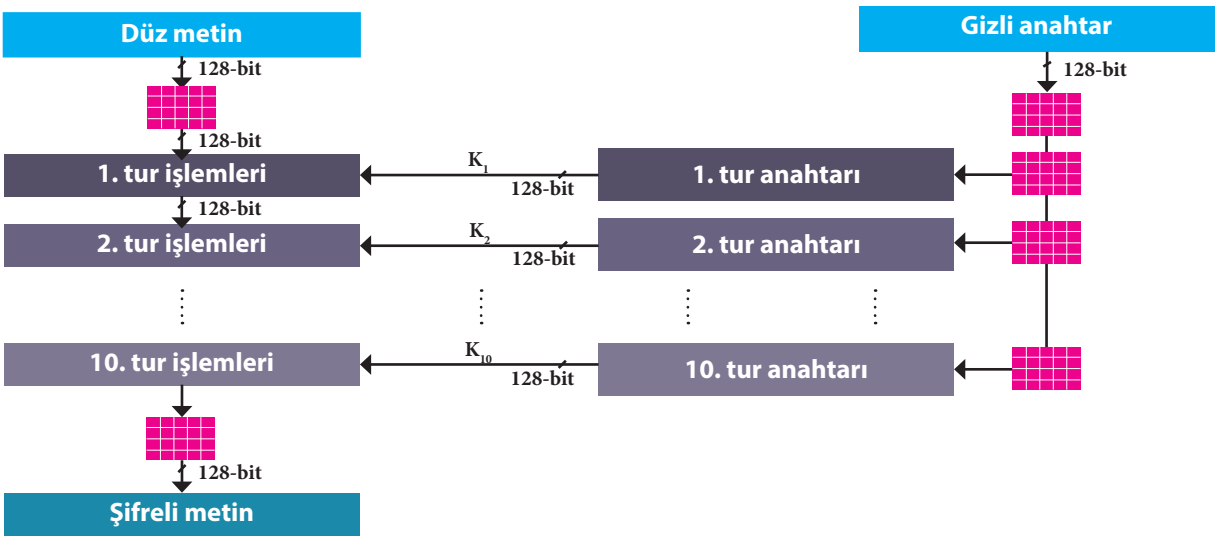
Her bir karakter ASCII koduna dönüştürülürse şu şekilde bir dizi elde edilir:

2	9		M	A	Y	I	S		1	4	5	3	.
050	057	032	077	065	089	073	083	032	049	052	053	051	046

Ardından bu dizi önce sütunları dolduracak şekilde 4x4 matrise dönüştürülür. Bayt sayısı 16'dan az olduğunda ise sonuna dolgu karakterleri eklenerek tamamlama işlemi yapılır. Örnekte dolgu karakteri için boşluk (ASCII: 32) kullanılmıştır.

$$\begin{vmatrix} 50 & 65 & 32 & 51 \\ 57 & 89 & 49 & 46 \\ 32 & 73 & 52 & \mathbf{32} \\ 77 & 83 & 53 & \mathbf{32} \end{vmatrix}$$

Son olarak bu matris, ikilik sayı sistemi ile ifade edilir ve AES operasyonları için hazır hâle getirilir. Görsel 3.24'teki diyagramda AES şifrelemesinin aşamaları verilmiştir.

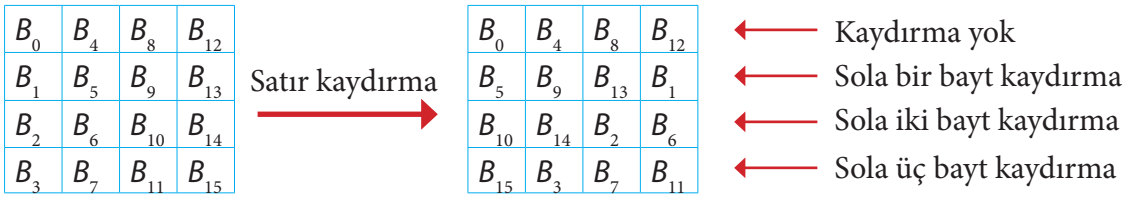


Görsel 3.24: AES şifreleme aşamaları

DES şifrelemesinde olduğu gibi AES şifrelemesinde de her bir tur için anahtar üretme mekanizması vardır. Gizli anahtardan her tur için kullanılmak üzere 128-bit uzunluklarında tur anahtarları üretilir. Bu anahtarlar, her aşamada yeni bir matris üretilerek elde edilir. Üretilen anahtarlar, her tur için bir dizi işlemde kullanılır. Her tur işlemi, sıralı şu dört aşamadan oluşur:

- Tur anahtarı ekleme (AddRoundKey)
- Bayt yerine koyma (SubBytes)
- Satır kaydırma (ShiftRows)
- Sütun karıştırma (MixColumns)

Tur anahtarı ekleme, basit bir XOR işlemidir ve düz metin matrisi ile tur anahtarı matrisinin XOR'lanmasını ifade eder. **Bayt yerine koyma** işleminde matrisin her baytı değiştirilir. Matris, belirli bir yer değiştirme tablosuna göre değiştirilir ve yeni değerlerden oluşan bir matrise dönüşür. Bu yeni matris üzerinde Görsel 3.25'te verilen satır kaydırma işlemi uygulanır.



Görsel 3.25: AES satır kaydırma işlemi

Satır kaydırma işleminin ardından sütun karıştırma işlemi yapılır. Bu işlemde her bir sütun, sabit bir matris ile çarpılır ve elde edilen değerler ile yeni bir matris oluşturulur. Elde edilen matris bir sonraki tur için kullanılır. Tur işlemlerinin ardından elde edilen ifade, şifreli metni verir.

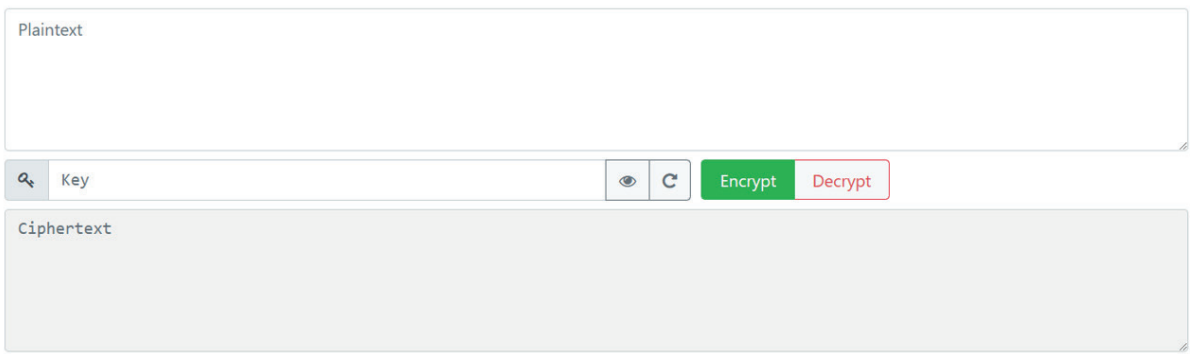
AES, güvenli ve sağlam bir şifreleme algoritması olarak kabul edilir. AES, kapsamlı analiz ve incelemelere tabi tutulmuştur ve temel tasarımında pratik olarak hiçbir güvenlik açığı henüz keşfedilmemiştir. AES, hassas veriler için yüksek düzeyde gizlilik ve bütünlük sağlar. AES, hesaplama açısından verimli olacak şekilde tasarlanmıştır. Hızlı ve verimli şifreleme ve şifre çözme avantajı nedeniyle hem yazılım hem de donanım uygulamalarında kullanılır.

AES hâlen yaygın olarak kullanılır ve modern kriptografik sistemlerin temel yapı taşı olarak kabul edilir. AES; güvenli iletişim protokolleri, disk şifreleme, dosya şifreleme, veri tabanı şifreleme, sanal özel ağlar ve veri korumanın çok önemli olduğu çeşitli uygulamalarda kullanılır.

6. Uygulama

İşlem adımlarına göre çevrimiçi siteleri kullanarak AES şifrelemeyi ve şifre çözmeyi yapınız. Bu uygulamayı gerçekleştirmek için ikişerli gruplar oluşturunuz. Grup üyelerinden birinin şifrelemeyi yapmasını, diğerinin şifreli mesajı çözmesini sağlayınız.

1. Adım: Çevrimiçi AES şifrelemesi yapan bir web sayfasını açınız. Görsel 3.26'da verilen sayfaya benzer bir ekran açılacaktır.



Görsel 3.26: AES şifreleme örneği

2. Adım: Grup üyelerinden birinin solda yer alan boşluğa istediği bir il ismini yazmasını sağlayınız. "Key" alanına anahtar olarak seçtiği 1 ile 100 arasında herhangi bir sayıyı yazmasını isteyiniz. Görsel 3.26'daki örnekte düz metin olarak "ANKARA", anahtar olarak da "73" seçilmiştir.

3. Adım: Encrypt düğmesini tıklayarak seçilen il adının şifrelenmesi sağlayınız ve şifreli metni diğer grup üyesine gönderiniz.

4. Adım: Grubun diğer üyesinden anahtarı vermeden şifreli metni çözmesini isteyiniz (Decrypt). Bu işlem için şifreli metni çözecek grup üyesine üç deneme hakkı veriniz. Üç deneme sonunda şifreli metni çözemezse anahtarın doğru değerini grup üyesine söyleyiniz ve şifreli metni çözmesini sağlayınız.

3.4.5. SEAL Kripto Algoritması

SEAL, 1994 yılında ortaya çıkan ve daha sonrasında patenti alınan simetrik modern bir akış şifreme algoritmasıdır. SEAL'de amaç, gizli bir anahtardan üretilen akış anahtarlar dizisi elde etmektir.

SEAL, 160-bit uzunluğunda gizli anahtar (secret key) kullanır. Bu anahtardan yola çıkarak bir dizi tur anahtarı üretir. Bu anahtarlar k(n) olarak gösterilir. Burada "n", 32-bitlik bir sayıdır ve tur anahtarını temsil eder. Örneğin ilk tur anahtarı k(0), ikinci tur anahtarı k(1) ile gösterilir.

Tur anahtarlarının boyutuna **uzunluk** denir ve "L" ile gösterilir. Uzunluk en fazla 216 bayt (64 KB) olabilir. SEAL'in belirgin özelliği, bu diziyi elde etmek için en baştan başlama zorunluluğunun olmamasıdır.

3.4.6. RC Kripto Algoritması

İsmini Ron Rivest'dan alan (Ron's Code) bu algoritma ailesi, birden çok algoritmadan (RC2, RC4, RC5, RC6 vb.) oluşur. **RC4**, 1987 yılında tasarlanmış bir algoritmadır. Bayt tabanlıdır ve değişken anahtar boyutlu bir akış şifresidir. Yakın zamana kadar TLS protokolünü kullanan web trafiklerinin güvenliğini sağlamak için tarayıcılarda yaygın olarak kullanıldı. RC4 algoritması, sözde-rastgele (pseudo-random) bir permütasyon kullanımına dayanır. Bu algoritmanın basit ancak çok hızlı mantığında **S, K** ve **T** olarak adlandırılan diziler kullanılır. Dizi elemanının indisini gösteren "n", 0 ile 255 aralığındaki değerleri alır. S[n] dizisi, S[0], S[1]… S[255] olmak üzere her biri 1 baytlık değer taşıyan dizidir. K, anahtarı taşıyan dizidir. T ise anahtar üretim sürecinde takas işlemleri için kullanılan geçici bir dizidir.

RC4, **anahtar planlama algoritması** ve **rastgele üretim algoritması** adı verilen iki adımdan oluşur. Anahtar planlama algoritmasında amaç, anahtarı taşıyan K dizisindeki değere göre S dizisini permütasyona uğratmaktır. Rastgele üretim algoritmasında ise akış anahtarlarının değerleri değiştirilir. RC4 algoritmasının adımları şu şekilde sıralanabilir:

1. Adım: İlklendirme

2. Adım: Permütasyon oluşturma

3. Adım: Akış anahtarlarını üretme

4. Adım: Şifreleme

RC4 algoritmasında "j" isimli bir değişken (pointer) kullanılır. Buna göre başlangıçta i=0 ve j=0 olmak üzere algoritma şu şekilde tanımlıdır:

1. **İlklendirme:** Başlangıç değerleri şu algoritmaya göre sağlanır:

   for i = 0 **to** 255 **do**

   S[i] = i;

   T[i] = K[i **mod** KeyLength];

KeyLength, anahtar uzunluğudur. Buna göre S dizisinin içeriği şu şekilde olur:

S[0] = 0; S[1] = 1;..S[255] = 255

Böylece T dizisinin içeriği, anahtar boyunca doldurulur. Örneğin anahtar "135" ise;

T[0] = 1; T[1] = 3; T[2] = 5; T[3] = 1; T[4] = 3;....olacak şekilde T dizisi oluşturulur.

2. **Permütasyon Oluşturma:** İlk permütasyon işlemi şu algoritmaya göre gerçekleştirilir:

   j = 0;

   for i = 0 **to** 255 **do**

   j = (j + S[i] + T[i]) **mod** 256;

   Swap (S[i], S[j]);

Anahtara bağlı olarak S dizisinin elemanları yer değiştirilir ve S dizisinin bir permütasyonu oluşturulur.

3. **Akış Anahtarları Oluşturma:** Şifrelemede kullanılacak anahtarlar bu aşamada üretilir. Algoritmada gösterilen "k", akış anahtarıdır.

   i, j = 0;

   while (true)

   i = (i + 1) **mod** 256;

   j = (j + S[i]) mod 256;

   Swap (S[i], S[j]);

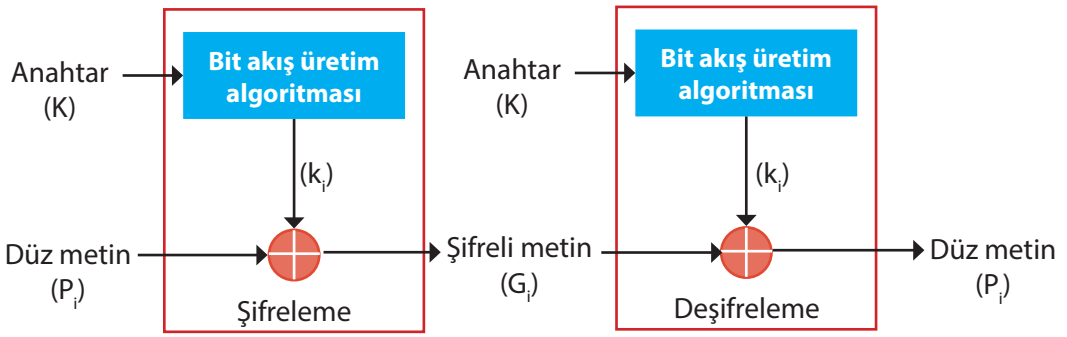
   t = (S[i] + S[j]) **mod** 256;

   k = S[t];

4. **Şifreleme:** Düz metnin baytları, sırasıyla üretilen "k" anahtarı ile XOR işlemine tabi tutulur ve şifreli metin elde edilir.

3.4.7. Akış Şifreleme

Akış şifreleme, düz metni genelde bit bit alır ve her bir biti ayrı ayrı şifreler. Şifreleme işlemi, bir akış anahtarı kullanılarak gerçekleştirilir. Görsel 3.27'de görüldüğü gibi bit-akış üretim algoritmaları (bit-stream generation algorithm), bit bit veya bayt bayt işlem yaptığı için sürekli bir veri akışı sağlar. Bu algoritmaların amacı, anahtar üretmektir.



Görsel 3.27: Akış şifrelemede verilerin bit bit veya bayt bayt şifrelenmesi

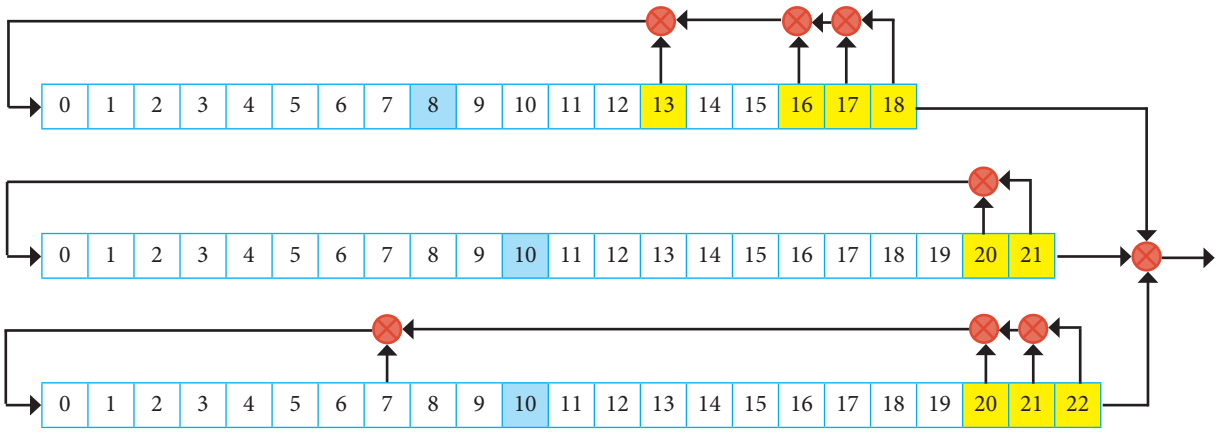
Her bir girdi biti için bir şifreleme işlemi uygulanır. Blok şifreleme algoritmaları, her bir blokun tamamlanması için tam blok boyutu kadar hafıza kullanır ancak akış şifreleme algoritmaları genellikle daha az hafıza kullanır.

3.4.8. A5/1 Algoritması

1987 yılında geliştirilen A5/1 algoritması, GSM mobil telefon ağlarında iletişim güvenliğini sağlamak için yapılan bir akış şifreleme örneğidir. Bu algoritma, üç yazmaçtan oluşan LFSR mantığına dayanır. Algoritma genellikle **ilklendirme, akış anahtarı üretme** ve **şifreleme** olmak üzere üç aşamadan oluşur.

İlklendirme aşamasında, **akış anahtarı (key stream)** üretmek için gerekli olan bitler hazırlanır. Akış anahtarı üretimi aşamasında, şifrelemek için kullanılacak anahtar bitleri üretilir. Şifreleme aşamasında ise düz metin veriler, üretilen anahtarla şifrelenir ve şifreli metin elde edilir.

Görsel 3.28'de A5/1 yapısındaki üç LFSR'nin diyagramı verilmiştir. İlk LFSR 19-bitten, ikincisi 22-bitten, üçüncüsü ise 23-bitten oluşur. Görsel 3.28'de her LFSR yapısındaki tap bitleri sarı renkle, **zamanlayıcı biti (clocking bit)** ise mavi renkle gösterilmiştir.



Görsel 3.28: Üç Adet LFSR İçeren A5/1 Algoritması

Zamanlayıcı bit, LFSR işleminde bit kaydırma işleminin yapılıp yapılmayacağını belirtir. Her LFSR için bu bitler Tablo 3.3'te verilmiştir.

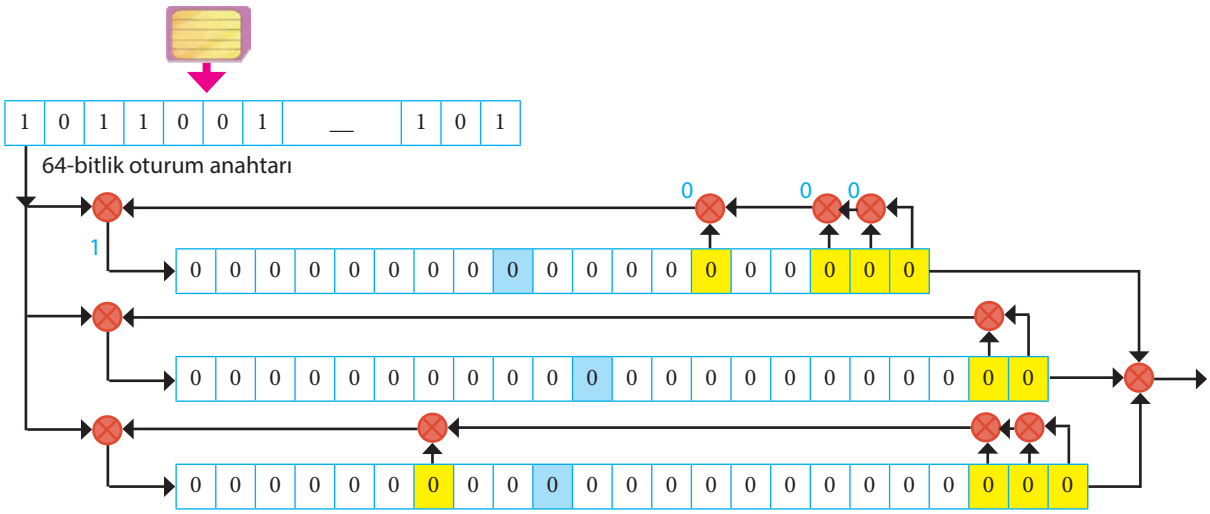
Tablo 3.3: Üç adet LFSR içeren A5/1 algoritması

LFSR	Bit Uzunluğu	Zamanlayıcı Bit	Tap Bitleri
LFSR#1	19	8	13, 16, 17 ve 18
LFSR#2	22	10	20, 21
LFSR#3	23	10	7, 20, 21 ve 22

İlklendirme aşaması (initialization phase) şu adımlardan oluşur:

1. Adım: Başlangıçta her üç LFSR'de yer alan tüm bitler 0 olarak ayarlanır.

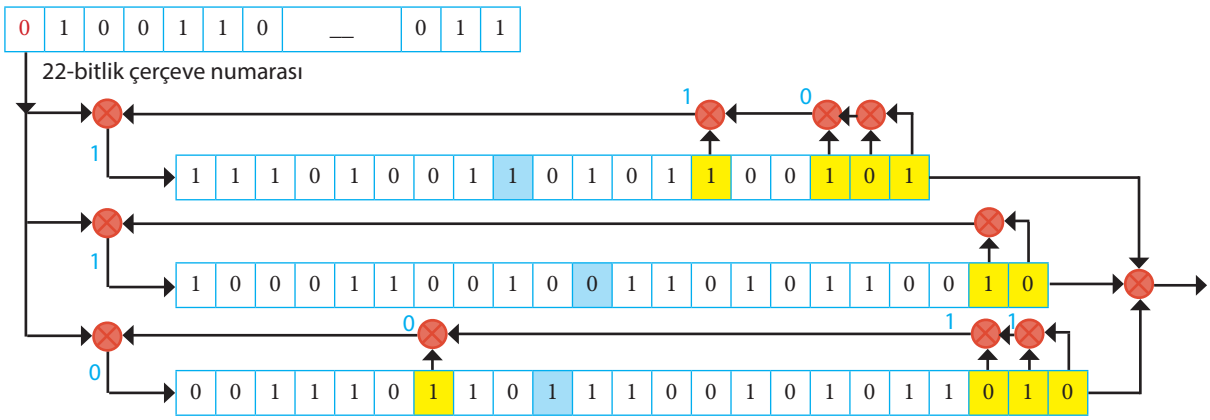
2. Adım: Bu adımda mobil cihazın SIM kartında yer alan bir algoritmaya göre üretilen 64-bitlik oturum anahtarı ile LFSR arasında Görsel 3.29'da gösterilen XOR işlemleri uygulanır.



Görsel 3.29: A5/1 ilklendirme aşaması

Görsel 3.29'daki ilk bitlere göre XOR sonuçları mavi renkte ve LFSR#1 için eklenecek bit örnekte 1 olarak gösterilmiştir. Bu adım, tüm oturum bitleri boyunca (64 tur) devam eder.

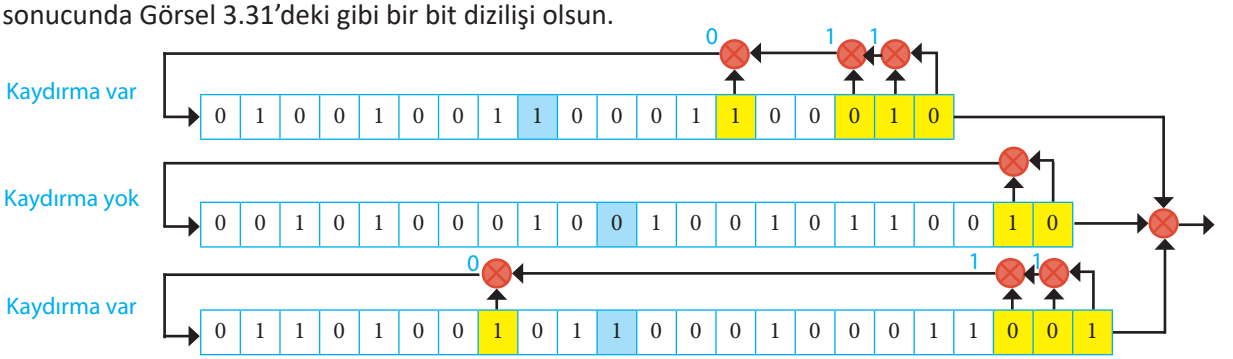
3. Adım: Çerçevenin (frame) sırasını gösteren bitlere göre işlem yapılır. İkinci adım sonucunda elde edilen bitlerin dizilişi Görsel 3.30'daki gibi olsun.



Görsel 3.30: İlklendirme aşaması ikinci adım sonrası örnek bit dizilişi

Üçüncü adımdaki XOR işlemleri, çerçeve bitleri bitinceye kadar devam eder (22 tur).

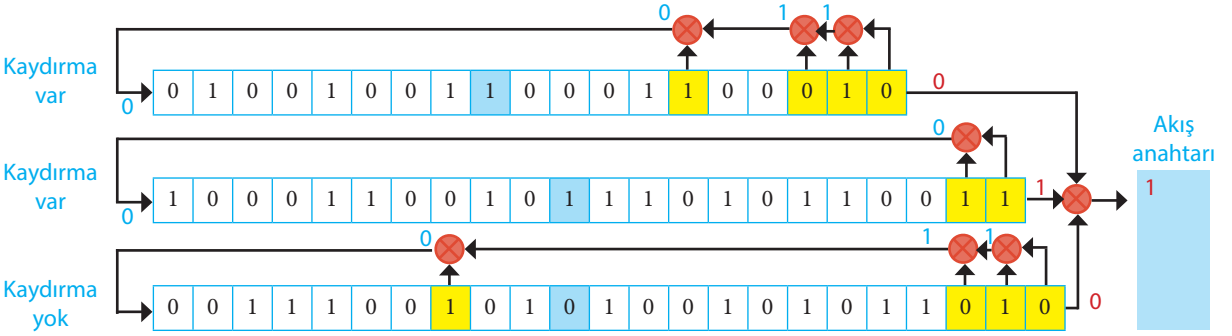
4. Adım: Zamanlayıcı bitlerine göre kaydırma işleminin yapılıp yapılmadığı belirlenir. Üçüncü adım sonucunda Görsel 3.31'deki gibi bir bit dizilişi olsun.



Görsel 3.31: İlklendirme aşamasında zamanlayıcı bitlerin kullanım örneği

Dördüncü adımda her bir LFSR üzerinde işlem yapılıp yapılmayacağı, zamanlayıcı bitlerine bağlı olarak değişir. Zamanlayıcı bitlerin sayısal çoğunluğu ile LFSR'nin biti aynı ise o LFSR için işlem yapılır. Görsel 3.31'deki örnekte zamanlayıcı bitlerinin sayısal çoğunluğu "1" bitindedir çünkü iki tane "1", bir tane de "0" vardır. Örnekte LFSR#2'de işlem yapılmadığına dikkat ediniz. Bu adım, 100 kez tekrar edilir ve ilklendirme aşaması tamamlanır. Artık LFSR'ler anahtar üretmeye hazırdır.

Akış anahtarı üretme aşamasında, ilklendirmede elde edilen son LFSR yazmaçlarının durumlarına göre akış anahtarı oluşturulur. İlklendirme aşamasının sonucunda elde edilen yapı Görsel 3.32'deki gibi olsun.



Görsel 3.32: İlklendirme aşaması sonrasında LFSR'lerin örnek gösterimi

Tap bitleri üzerinde yapılan XOR işlemi sonucunda elde edilen değer ilk bite yazılır ancak burada zamanlayıcı bit ile çoğunluktaki zamanlayıcı bit aynı ise bu işlem gerçekleştirilir. Görsel 3.32'deki örnekte çoğunluğun "1" biti olduğuna dikkat edilmelidir. LFSR#1'in zamanlayıcı biti de "1" olduğu için işlem yapılır. Örnekte elde edilen "0" biti, sonraki adımda ilk bite yazılır. Son bit ise (örnekte 0) çıktı olarak üretilir.

LFSR#2 için benzer şekilde "0" girecek ve "1" çıkacaktır ancak LFSR#3'te zamanlayıcı biti "0" çoğunluk bitine "1" eşit olmadığı için kaydırma işlemi yapılmadan son bit "0" çıktı olarak üretilecektir.

Son olarak her bir yazmaçtan elde edilen bitler, XOR işlemine tabi tutulur ve akış anahtarının ilk biti üretilir. Örnekte "1" biti, üretilecek akış anahtarının ilk bitidir. Bu işlem, 228 kez tekrar ettirilir ve 228-bit uzunluğunda akış anahtarı üretilir. Bu anahtarın ilk 114-biti indirme (download) işleminde, sonraki 114 biti ise yükleme (upload) işleminde şifreleme için kullanılır.

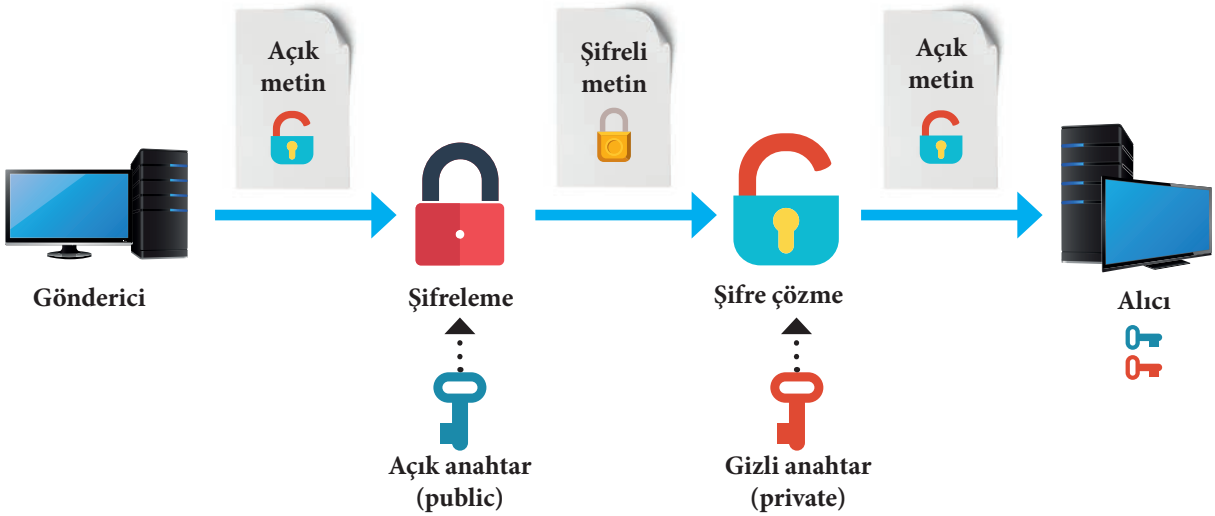
Bir sonraki çerçeve için aynı oturum anahtarı kullanılarak aynı işlemler devam eder ancak bir sonraki çerçevenin sıra numarası 1 artar.

Şifreleme aşamasında, üretilen akış anahtarı ile düz metnin XOR işlemine tabi tutulması sonucunda şifreli metin elde edilir.

A5/1 algoritması, GSM şebekelerinde veri şifrelemek için kullanılır. Bu algoritma, başlangıçta güvenliyken daha sonra algoritması öğrenildikçe tersine mühendislik ile kırılabilir hâle gelmiştir.

3.5. ASİMETRİK KRİPTOGRAFİ

Şifrelemede kullanılan diğer bir yöntem asimetrik şifrelemedir. Bir düz metnin şifrelenmesinde ve deşifre edilmesinde farklı anahtarlar kullanılıyorsa bu yönteme **asimetrik şifreleme** denir. Asimetrik şifrelemede birbirine matematiksel ilkeler ile bağlı olan bir **açık anahtar (public key)** ve bir **gizli anahtar (private key)** vardır. Görsel 3.33'te görüldüğü gibi açık anahtar (genel anahtar), herkese açık bir şekilde paylaşılabilir ve veriyi şifrelemek için kullanılır. Özel anahtar ise sadece alıcının elinde tutulur ve şifreli veriyi çözmek için kullanılır.



Görsel 3.33: Asimetrik kriptografide birbiriyle ilişkili açık ve gizli iki farklı anahtarın kullanılması

Simetrik şifrelemenin uygulaması kolaydır ve günümüz modern kriptografisinde de yaygın olarak kullanılır. Simetrik şifreleme özellikle hızlı şifreleme ve deşifreleme gerektiren uygulamalarda, cihazların daha fazla kaynak (işlemci gücü ve bellek) tüketmemesi istenen durumlarda kullanılır. Asimetrik şifreleme ise daha fazla kaynak tüketir ve genelde daha zaman alıcı bir şifrelemedir.

Asimetrik fonksiyonlarda birbiriyle matematiksel olarak ilişkili açık ve gizli anahtarlar kullanılır. Bu anahtarların hesaplanmasının kolay, kırılmasının zor olması beklenir. Bu nedenle asal sayıların sağladığı güvenlik; şifrelemede, deşifrelemede, dijital imza gibi kriptografik unsurlarda etkili bir şekilde kullanılır. Asal sayılar, çeşitli kriptografik algoritmaların temel yapı taşları olarak kullanıldığı için kriptografide büyük öneme sahiptir.

Asal sayılar, yalnızca 1 ve kendiyle tam bölünebilen sayılardır. Bu özellikleri, asal sayıları benzersiz kılar ve onların daha küçük çarpanlara sahip sayılara ayrıştırılmamasını sağlar. Asal sayılar kullanılarak oluşturulan şifreleme anahtarları veya rastgele sayılar, kriptografi algoritmalarının güvenliğini artırır. Birçok güçlü kriptografik algoritma, temel işlemlerinde büyük asal sayılar kullanır. Örneğin RSA (Rivest-Shamir-Adleman) şifrelemesi, büyük asal sayılar üzerine kurulmuş açık anahtarlı bir şifreleme algoritmasıdır.

Kriptografik algoritmaların dayandığı matematiksel zorluklardan biri, çarpanlarına ayırma problemleridir. Bu problemler, büyük bir sayının çarpanlarına ayrıştırılmasını gerektirir. Bu ayrıştırmadaki güçlük, büyük bir asal sayıyı veri anahtarı olarak kullanmanın pratikte çözülmesi güç bir şifre sağladığı anlamına gelir.

3.5.1. Çarpanlara Ayırma Tekniği

Büyük sayıların çarpanlarına ayrıştırılması, matematiksel açıdan oldukça karmaşık ve hesaplama açısından yoğun bir işlem olabilir. Bu durum, büyük asal sayıların kullanıldığı kriptografik algoritmaların temel güvenlik unsurlarından biridir.

"p" ve "q" iki büyük asal sayı olmak üzere **n = p x q** olsun. Burada "n", büyük bir bileşik sayı olacaktır. İki asal sayıyı bilerek "n" sayısını elde etmek kolaydır ancak "n" sayısı verildiğinde "p" ve "q" asal sayılarını bulmak oldukça zordur. Bu zorluk, RSA şifreleme algoritmasının güvenliğinin temelini oluşturur.

Kriptografide "n" sayısının çarpanları gizli tutulur ve bu değerlerden yola çıkılarak gizli ve açık anahtarlar üretilir. Bir saldırgan, "n" sayısını çarpanlara ayırmayı başarırsa asal çarpanlarını bulabilir ve böylece gizli anahtarı elde edebilir. Günümüzde bilinen en iyi çarpanlara ayırma algoritmaları, büyük sayıları asal çarpanlarına ayrıştırmak için oldukça uzun süreler ve yoğun hesaplamalar gerektiren yöntemler kullanır. Bu nedenle yeterince büyük asal sayılar kullanılarak daha yüksek güvenlik sağlanabilir.

Öklid öncesinden beri her "n" tam sayısının benzersiz bir şekilde asal sayıların bir çarpımına ayrıştırılabileceği bilinir. Örneğin 18 sayısı;

$18 = 2 \cdot 3^2$

olarak ifade edilebilir. 2 ve 3 asal sayılardır. Bu örnek şu şekilde genelleştirilebilir:

$$n = p_1^{a1} p_2^{a1} \ldots p_k^{ak}$$

Burada $p_1 < p_2 < p_3 \ldots < p_k$ dir ve a_i ifadeleri pozitif tam sayılardır.

Sayıların çarpanlarına ayrılmasında çeşitli yöntemler bulunur. Fermat'ın yöntemi, bir sayının, iki sayının karelerinin farkı olarak gösterilmesine dayanır. Fermat'a göre "*n*" bileşik bir sayı olmak üzere $n = a^2 - b^2$ olarak yazılabilirse $n = (a - b)(a + b)$ olarak çarpanlarına ayrılabilir.

Örnek

84 sayısını iki tam sayının çarpımı olarak yazalım.

1. 84'ten büyük ve 84'e yakın bir kare sayı seçilir (100).
2. $84 = 100 - a^2$ olacak şekilde bir a sayısı aranır. $a^2 = 16$ olacaktır. Bu durumda a = 4 olabilir.
3. 84 sayısı $10^2 - 4^2$ olarak yazılır.
4. 84 = (10 - 4) (10 + 4) olarak ifade edilir.
5. 84'ün çarpanları, 6 x 14 olarak gösterilebilir.

Fermat'ın yöntemi, büyük sayılar için etkili olmayabilir. Büyük asal çarpanlara sahip sayıları ayırmak için daha gelişmiş ve verimli algoritmalar kullanılmalıdır.

3.5.2. Asallık Testi

Asal sayılar kriptografide önemlidir ancak kriptografi için seçilen sayının asal olup olmadığı belirlenmelidir. Küçük asal sayıların bulunması için sırayla bölünebilme testleri uygulanabilir ancak çok büyük sayıların kullanıldığı modern kriptografide sırasıyla deneme yöntemi, zaman alıcı ve sistem kaynaklarını tüketici olabilir. Bu nedenle asallık testleri için çeşitli yöntemler geliştirilmiştir.

Bazı geleneksel yöntemler sırayla deneme yanılmayı kullanırken bazı yöntemler ise sayıların asal olma olasılığı temeline dayanır. Fermat'ın asallık testi, olasılık temellidir.

P: Bir asal sayı olsun.

a: P ile aralarında asal bir sayıdır. O hâlde EBOB (P, a) = 1 olmalıdır.

$a^{p-1} \equiv 1$ (MOD P) ve $a^{P} \equiv a$ (MOD P)

Fermat'ın asallık testi bu denkliği kullanarak sayının asal olma ihtimalini kontrol eder. Fermat'a göre P bir asal sayı ise;

$1 \leq a < P$ aralığındaki her a sayısı için;

$a^{p-1} \equiv 1$ (MOD P) şartını sağlıyorsa P asaldır.

Örnek

5 sayısının asal olup olmadığı Fermat'ın asallık testi ile şu şekilde kontrol edilir:

1. P = 5 ve $1 \leq a < 5$ aralığındaki tüm sayılar için $a^{5-1} \equiv 1$ (MOD 5) olup olmadığı kontrol edilir.

$a = 1$ için $1^4 \equiv 1$ (MOD 5)

$a = 2$ için $2^4 \equiv 16 \equiv 1$ (MOD 5)

$a = 3$ için $3^4 \equiv 81 \equiv 1$ (MOD 5)

$a = 4$ için $4^4 \equiv 256 \equiv 1$ (MOD 5)

2. Tüm şartları sağladığı için 5, asal sayıdır.

7. Uygulama

İşlem adımlarına göre 6 sayısının asal olup olmadığını kontrol ediniz.

1. Adım: P = 5 ve $1 \leq a < 6$ aralığındaki tüm sayılar için $a^5 \equiv 1$ (MOD 6) olup olmadığını kontrol ediniz.

2. Adım: Sırasıyla $a^{(p-1)}$ değerini hesaplayalım.

$a = 1$ için $1^5 \equiv 1$ (MOD 6)

$a = 2$ için $2^5 \equiv 32 \equiv 2$ (MOD 6) şartını sağlamadığı için 6, asal sayı değildir.

Fermat'ın asallık testi, büyük sayılar için hızlı ve basit bir ön kontrol sağlar ancak kesin bir sonuç vermez. Bu nedenle kritik sistemlerde veya yüksek güvenlikli uygulamalarda Miller-Rabin veya AKS (Agrawal-Kayal-Saxena) gibi algoritmalar tercih edilir.

Euler Totient: Bir tam sayının o sayıdan daha küçük ve o sayı ile aralarında asal olan sayıları belirten fonksiyondur ve genelde ϕ ile gösterilir.

Örnek

10 sayısının φ (fi) değerini hesaplayalım. 10'dan küçük olan ve 10 ile aralarında asal olan (1 dâhil edilir) sayıların oluşturduğu kümeyi ve bu kümenin eleman sayısını bulalım.

A = {1, 3, 7, 9} O hâlde;

s(A) = ϕ (10) = 4

Örnek

11 asal sayısı için ϕ değerini bulalım. 11'den küçük ve 11 ile aralarında asal olan sayıların kümesi B olmak üzere;

B = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} olacaktır. O hâlde;

s(B) = ϕ (11) = 10 olur.

Dikkat edilirse "n" bir asal sayı olmak üzere asal sayılar için ϕ değeri n-1 olacaktır. O hâlde;

ϕ(5) = 4, ϕ(7) = 6 olur.

Benzer şekilde bir tam sayı, iki asal sayının çarpımı olarak gösterilirse bu durumda bu sayının Euler Totient değeri, ϕ değerlerinin çarpımıdır. Örneğin 15 sayısı, 3 ve 5 asal sayılarının çarpımıdır. Bu durumda ϕ(15) = ϕ(3) x ϕ(5) = 2 x 4 = 8 olacaktır. O hâlde 15'ten küçük, 15 ile aralarında asal sayıların sayısı 8'dir. Bu küme şu şekilde doğrulanabilir:

C = {1, 2, 4, 7, 8, 11, 13, 14}

Bir x sayısı, a asal olmak üzere a^n şeklinde yazılabilirse bu durumda $\varphi(x) = a^n - a^{(n-1)}$ olur. Örneğin x = 8 ise $8 = 2^3$ şeklinde yazılabilir. O hâlde $\phi(8) = 2^3 - 2^2 = 4$ olur.

3.5.3. Kesikli Logaritma

Kesikli (ayrık) logaritma, yüksek düzeyde matematiksel hesaplamalar gerektiren bir problemdir ve kriptografi alanında önemli bir rol oynar. Kriptografide kesikli logaritma, özellikle açık anahtarlı kriptografik algoritmalarda kullanılır. Kesikli logaritma, anahtar çiftleri oluşturmak ve şifreleme, şifre çözme işlemlerini gerçekleştirmek için kullanılır. Basit bir ifade ile logaritma, üstel fonksiyonların tersi olan bir fonksiyondur. Örneğin $2^3 = 8$ ifadesi şu şekilde yazılabilir:

$$\log_2 8 = 3$$

Daha genel bir ifade ile $a \neq 1$ ve b > 0 olmak üzere $a^x = b$ ifadesi $\log_a b = x$ olarak yazılabilir.

Kesikli logaritma, verilen bir sayının başka bir sayının hangi üssü olarak elde edileceğini bulmak için kullanılan bir işlemdir.

Özetle ifade etmek gerekirse, P çift olmayan bir asal sayı ve g bir üreteç olmak üzere,

$g^a = y \pmod P$ için;

"g" ve "a" verildiğinde "y" değerini hesaplamak kolaydır ancak "y" ve "g" verildiğinde "a" değerini hesaplamak zordur. Buna **kesikli logaritma problemi** denir.

Örneğin,

$3^{29} = y \pmod{17}$ için "y" sayısını hesaplamak kolay (y = 12), $3^a = 12 \pmod{17}$ için "a" değerini bulmak zordur çünkü çözmek için sayıların tek tek denenmesi gerekir. Küçük sayılar için bu işlem çok zor olmasa da kriptografide yüzlerce basamaklı bir asal sayı için deneme yanılma yöntemi ile bulunması pratikte (anlamlı zaman diliminde) mümkün değildir. Kesikli logaritma çoğunlukla grup teorisiyle ilişkilidir. Grup teorisine ait tanımlar ve özellikler şunlardır:

Tanımlar

- P tek bir asal sayı olmak üzere;
  {1, 2, 3, ... P - 1} kümesi $\mathbb{Z}_P^*$ olarak ifade edilsin.
- $\alpha, \beta \in \mathbb{Z}_P^*$ olmak üzere $\alpha . \beta = (\alpha . \beta) \pmod P$ işlemi tanımlı olsun.

I. Özellik

$\alpha.\beta \in \mathbb{Z}_P^*$ ise $\mathbb{Z}_P^*$ kümesi, " . " işlemine göre kapalıdır.

II. Özellik

$\alpha, \beta, \gamma \in \mathbb{Z}_P^*$ olmak üzere $\alpha . (\beta . \gamma) = (\alpha . \beta) . \gamma$ *ise*, " . " işlemine göre birleşme özelliği vardır.

III. Özellik

$\alpha \in \mathbb{Z}_P^*$ için $\alpha . I = \alpha$ ise I birim elemanı vardır.

IV. Özellik

I birim eleman ve $\alpha, \beta \in \mathbb{Z}_P^*$ olmak üzere $\alpha . \beta = I$ ise α, β*'nin tersidir.*

$\mathbb{Z}_P^*$ kümesinde her elemanın tersi olmalıdır.

I, II, III ve IV. özellikleri gösteren her küme, **Grup** olarak adlandırılır ve "**G**" ile gösterilir.

O hâlde ($\mathbb{Z}_P^*$, ".") bir gruptur.

Örnek

P = 11 ve çarpma (.) işlemi için $\mathbb{Z}_{11}^*$ in bir grup olup olmadığını test edelim.

$\mathbb{Z}_{11}^*$= {1, 2, 3, … 10} olacaktır.

I. Özellik Testi

3, 5 ∈ $\mathbb{Z}_{11}^*$ dir.

3.5≡4 (MOD 11) ve 4 ∈ $\mathbb{Z}_{11}^*$ dir.

Tüm elemanlar test edilirse $\mathbb{Z}_{11}^*$ in kapalı olduğu anlaşılır. O hâlde bu örnekte I. özellik vardır.

II. Özellik Testi

2, 3, 5 ∈ $\mathbb{Z}_{11}^*$ olmak üzere;

2 . (3 . 5) ≡ (2 . 3) . 5 (MOD 11)

2 . (15) ≡ (6) . 5 (MOD 11)

2 . 4 ≡ 6 . 5 (MOD 11)

8 ≡ 30 (MOD 11)

8 ≡ 8 (MOD 11) O hâlde II. özellik vardır.

III. Özellik Testi

5 ∈ $\mathbb{Z}_{11}^*$ ve

5 . 1 ≡ 5 (MOD 11)

olduğuna göre III. özellik vardır. Tüm elemanlar, "1" ile işleme tabi tutulduğunda "1"in birim eleman olduğu görülür.

IV. Özellik Testi

x, 3 ∈ $\mathbb{Z}_{11}^*$

3 . x ≡ 1 (MOD 11) olacak şekilde bir x elemanı aranır.

Denkliğin her tarafı "4" ile çarpılırsa,

4. 3 . x ≡ 4.1 (MOD 11)

12x ≡ 4 (MOD 11)

1x ≡ 4 (MOD11)

x ≡ 4 (MOD 11) bulunur.

O hâlde $\mathbb{Z}_{11}^*$ kümesinde tanımlı çarpama (.) işlemi için 3'ün tersi 4'tür. Tüm elemanlar için bu işlem tekrarlanabilir.

$\mathbb{Z}_{11}^*$ kümesi, bu dört özelliği desteklediğine göre ($\mathbb{Z}_{11}^*$,".") bir gruptur.

α ∈ $\mathbb{Z}_P^*$ ve n ∈Z olmak üzere;

α^n lerden oluşan küme $\{\alpha^1, \alpha^2, a_{...}^3 \alpha^{P-1}\} = \mathbb{Z}_p^*(\mathrm{MOD}P)$ şartını sağlıyorsa α, $\mathbb{Z}_P^*$ grubunun üretecidir (generator). Bu üreteç sayıya MOD P'nin **asal kökü (primitive root)** denir. Böyle bir gruba **döngüsel grup (cyclic group)** denir.

Örnek

p = 11 ve α = 2 olsun.

$\mathbb{Z}_{11}^{*}$ = {1, 2, 3, … 10} olacaktır.

$2^1 \equiv 2$ (MOD 11)	$2^6 \equiv 9$ (MOD 11)
$2^2 \equiv 4$ (MOD 11)	$2^7 \equiv 7$ (MOD 11)
$2^3 \equiv 8$ (MOD 11)	$2^8 \equiv 3$ (MOD 11)
$2^4 \equiv 5$ (MOD 11)	$2^9 \equiv 6$ (MOD 11)
$2^5 \equiv 10$ (MOD 11)	$2^{10} \equiv 1$ (MOD 11)

Bu işlemlerin sonucunda elde edilen küme şu şekildedir:

{2, 4, 8, 5, 10, 9, 7, 3, 6, 1}

Bu küme aynı zamanda $\mathbb{Z}_{11}^{*}$ kümesidir. O hâlde $\alpha = 2$ $(Z_{11}^{*}, ".")$ döngüsel grubunun üretecidir. Bir grubun birden fazla üreteci olabilir.

Döngüsel gruplar, açık anahtarlı kriptografi kapsamında önemli bir rol oynar. Açık anahtarlı kriptografide Diffie-Hellman anahtar değişimi, ElGamal şifreleme, DSA (Digital Signature Algorithm) gibi algoritmalar, ayrık logaritmayı çözmenin zorluğuna dayanır.

3.5.4. Diffie-Hellman Algoritması

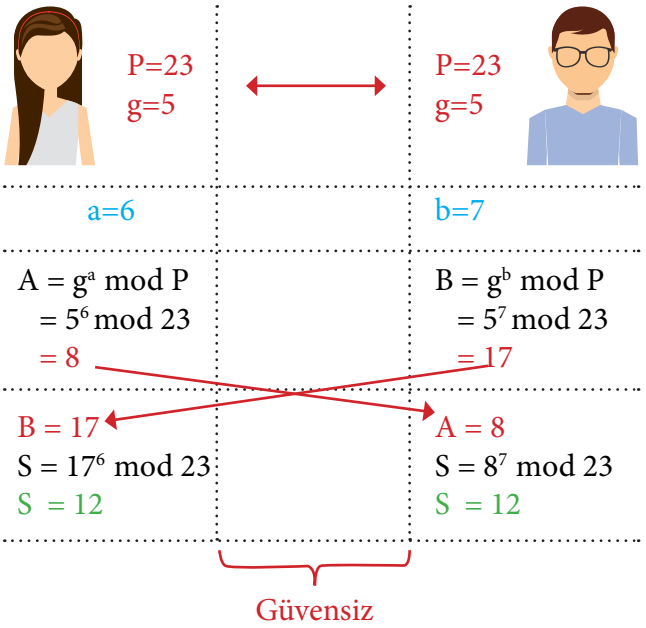
Simetrik şifrelemede, dijital imzalar gibi kriptografik sistemlerde iletişim kuran taraflarca paylaşılan bir gizli anahtarın varlığı gereklidir ancak bu anahtarın güvenli olmayan bir kanal üzerinden karşı tarafa gönderilmesi ve taraflarca paylaşılması riskler barındırır. Gizli anahtar, taraflar arasında doğrudan paylaştırıldığında iletişimi dinleyen saldırganlar tarafından da ele geçirilebilir. Bu nedenle gizli anahtarın doğrudan gönderilmeden her iki taraf için de aynı olmasını sağlayacak bir yöntemin geliştirilmesi önemli bir ihtiyaçtır. Diffie-Hellman (DH), gizli anahtarı doğrudan iletmeden her iki taraf için aynı olmasını sağlayan matematiksel bir çözüm sunar. DH algoritması, Whitfield Diffie ve Martin Hellman tarafından 1976 yılında yayımlanmıştır. DH, kriptografik anahtarları ortak bir kanal üzerinden güvenli bir şekilde değiş tokuş etmenin matematiksel bir yöntemidir ve ilk açık anahtar protokollerinden biridir.

Diffie-Hellman anahtar değişimi şu aşamalardan oluşur:

- **Anahtar Üretimi:** Her iki taraf, birer özel anahtar oluşturur ve bu anahtara karşılık gelen açık anahtarı hesaplar.
- **Anahtar Değişimi:** Taraflar, açık anahtarlarını şifreleme olmadan açık bir şekilde birbirleri ile paylaşır.
- **Paylaşılan Gizli Anahtar Hesaplama:** Her iki taraf, kendi özel anahtarını ve alınan açık anahtarı kullanarak, paylaşılan aynı gizli değeri bağımsız olarak hesaplar.
- **Paylaşılan Gizli Anahtar:** Her iki taraf için hesaplanan bu gizli değer, simetrik şifreleme veya diğer kriptografik işlemler için paylaşılan anahtar görevini görür.

Örnek

DH anahtar değişimi algoritmasının çalışma prensibiyle ilgili örnekte sol taraf Ayşe (A), sağ taraf Berk (B) olsun. Görsel 3.34'te DH anahtar değişimi algoritmasının çalışma prensibi örnek sayılar üzerinden verilmiştir.



Görsel 3.34: Diffie-Hellman ortak anahtar üretme aşamaları

İlk aşamada taraflar, büyük bir asal sayı (P) ve taban olarak kullanılacak bir sayı (g) üzerinde anlaşırlar. Burada "g" sayısı üreteçtir ve "p" sayısının asal köküdür. Bu iki sayı güvensiz ortamdan gönderilir. Dolayısıyla saldırganlar tarafından bilinebilir. Görsel 3.34'teki örnekte P = 23 ve g = 5 seçilmiştir. Daha sonra tarafların her biri, gizli bir sayı seçer. Bu sayı, tarafların gizli anahtarı olarak düşünülebilir. Görsel 3.34'teki örnek için a = 6 ve b = 7 sayıları gizli anahtarlardır ve paylaşılmaz.

Sonraki aşamada taraflar şu formüle göre açık anahtar üretirler:

$A = g^a \ MOD \ P$

$B = g^b \ MOD \ P$

O hâlde sol taraf (Ayşe) için açık anahtar şu şekilde hesaplanır:

$A = g^a \ MOD \ P$

$A = 5^6 \ MOD \ 23$

A = 15625 MOD 23

A = 8 (Sol tarafın açık anahtarı)

Sağ taraf (Berk) için açık anahtar şu şekilde hesaplanır:

$B = g^b \ MOD \ P$

$B = 5^7 \ MOD \ 23$

B = 78125 MOD 23

B = 17 (Sağ tarafın açık anahtarı)

Bu açık anahtarlar, güvensiz ortamda taraflarca paylaşılır. Taraflar, karşı taraftan gelen açık anahtarı ve kendi gizli anahtarını kullanarak paylaşılan gizli anahtarı (S olsun.) şu şekilde hesaplar:

$S = B^a \ MOD \ P$ veya $S = A^b \ MOD \ P$

$S = 17^6 \ MOD \ 23$ veya $S = 8^7 \ MOD \ 23$

$S = 12$

Her iki tarafta da S = 12 bulunur. Burada S, sadece Ayşe ile Berk tarafından hesaplanan gizli anahtar olacaktır. DH yapısında büyük asal sayılar kullanıldığı için güvensiz ortamdaki iletişim, saldırganlar tarafından dinlense bile anlamlı bir süre içinde "S" sayısının hesaplanması çok zordur.

Diffie-Hellman anahtar değişim sürecinde kullanılan anahtarların gücü, DH grupları olarak anılır. Bazı grupların bit uzunlukları şu şekildedir:

- DH Group 1 (768-bit)
- DH Group 2 (1024-bit)
- DH Group 5 (1536-bit)
- DH Group 14 (2048-bit)
- DH Group 15 (3072-bit)
- DH Group 19 (random 256-bit eliptik eğri)
- DH Group 20 (random 384-bit eliptik eğri)
- DH Group 21 (random 521-bit eliptik eğri)

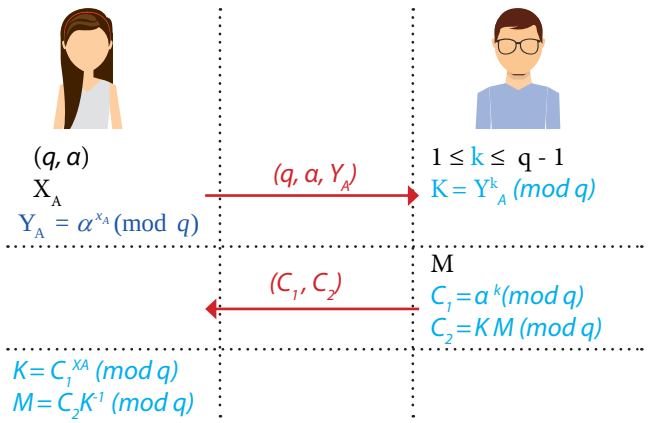
Son gruplarda yer alan eliptik eğri, diğer şifreleme algoritmalarına kıyasla daha küçük anahtar boyutlarıyla güçlü güvenlik sağlar.

3.5.5. ElGamal Algoritması

Taher ElGamal, 1984 yılında kesikli logaritmaya dayalı bir açık anahtar algoritması önermiştir. Diffie-Hellman kriptografisine oldukça benzeyen ElGamal, dijital imza kimlik doğrulama sürecinde kullanılır.

Örnek

Elgamal algoritması kullanılarak oturum anahtarı üretme, şifreleme ve deşifreleme örneğinde sol taraf Ayşe (A), sağ taraf Berk (B) olsun. Ayşe (A) ve Berk (B) arasındaki iletişim için ElGamal algoritmasının oturum anahtarı üretme, şifreleme ve deşifreleme aşamaları Görsel 3.35'te verilmiştir.



Görsel 3.35: ElGamal algoritmasının çalışma prensibi

- **Oturum Anahtarı Üretme**

  A tarafında şu adımlar gerçekleştirilir:

1. Adım: q bir asal sayı ve α bunun asal kökü olmak üzere {q, α } çifti seçer.

2. Adım: 1 < XA < q - 1 aralığında rastgele bir XA sayısı seçer. Bu sayı, A'nın gizli anahtarıdır.

3. Adım: $Y_A = \alpha^{X_A} (\text{mod } q)$ hesaplar. Burada {q, α, YA } üçlüsü, A'nın açık anahtarıdır.

A'nın açık anahtarına sahip olan B, herhangi bir mesajı (örnekte M) şifreleyebilir.

- **Şifreleme**

1. Adım: 0 ≤ M ≤ q - 1 olacak şekilde M mesajını yazar. Mesaj daha büyük ise mesajı bloklara ayırır ve her bloka aynı işlemleri yapar.

2. Adım: 1 ≤ k ≤ q - 1 aralığında bir tam sayı seçer.

3. Adım: $K = Y_A^k (MOD\ q)$ olacak şekilde bir oturum anahtarı hesaplar.

4. Adım: $C_1 = \alpha^k$ (MOD q) ve C_2 = KM(MOD q) olacak şekilde C_1 ve C_2 hesaplar ve gönderir. Burada {C_1, C_2} çifti şifreli metindir. Alıcı tarafında oturum anahtarını elde etmek için C_1, düz metni elde etmek için C_2 kullanılacaktır.

- **Deşifreleme**

  Mesaj, A tarafından şu şekilde deşifre edilir:

1. Adım: Oturum anahtarını elde etmek için $K = C_1^{X_A} (MOD\ q)$ hesaplar.

2. Adım: Elde edilen K anahtarını kullanarak C_2 mesajı şu formülle deşifre edilir:

$$M = C_2 K^{-1} (MOD\ q)$$

Örnek

Ayşe ile Berk arasında ElGamal algoritması kullanılarak şifreli mesaj iletilecektir. Berk, Ayşe'ye 4 sayısını mesaj olarak göndersin.

1. Adım: Oturum anahtarı üretme...*q = 19, α = 3* olsun.

2. Adım: XA = 5 olsun (Ayşe'nin gizli anahtarı).

3. Adım: YA hesaplaması yapılır.

YA = 3^5 (MOD 19)

YA = 243 (MOD 19)

YA = 15

Bu aşamada {19, 3 ve 15} açık anahtarı Berk'e iletilir. Berk tarafında şu işlemler yapılır:

4. Adım: 1 ≤ k ≤ q - 1 aralığında k = 7 olsun.

5. Adım: $K = Y_A^k (MOD\ q)$ olacak şekilde bir oturum anahtarı hesaplanır.

$$K = Y_A^k (MOD\ q)$$
$$K = 15^7 (MOD\ 19)$$
$$K = 13$$

6. Adım: $C_1 = \alpha^k$ (MOD q) ve $C_2 = KM$ (MOD q) hesaplanır. Burada M, düz metindir ve örnekte 4 sayısı olarak verilmiştir.

$C_1 = 3^7$ (MOD 19) ve $C_2 = 13 . 4$ (MOD 19)

$C_1 = 2$ ve $C_2 = 14$ olarak hesaplanır.

7. Adım: {2, 14} şifreli metni Ayşe'ye gönderilir.

Şifreli metni alan Ayşe, düz metin elde etmek için şu adımları uygular:

8. Adım: Oturum anahtarını şu şekilde hesaplar:

$$K = C_1^{X_A} (MOD\ q)$$
$$K = 2^5 (MOD\ 19)$$
$$K = 13$$

9. Adım: Düz metni şu şekilde hesaplar:

$M = C_2 K^{-1}$ (MOD q)

$M = 14 . 13^{-1}$ (MOD 19)

Burada 13^{-1} işlemi, MOD 19'a göre 13'ün tersini bulma işlemidir. T sayısı, 13'ün MOD 19'a göre tersi olmak üzere şöyle ifade edilebilir:

13 . T = 1 (MOD 19)

Her taraf 3 ile çarpılırsa;

39 . T = 3 (MOD 19) elde edilir.

T = 3 olur (39 ≡ 1 MOD 19).

O hâlde deşifreleme işlemi şu şekilde olur:

M = 14 . 3 (MOD 19)

M = 42 (MOD 19)

M = 4

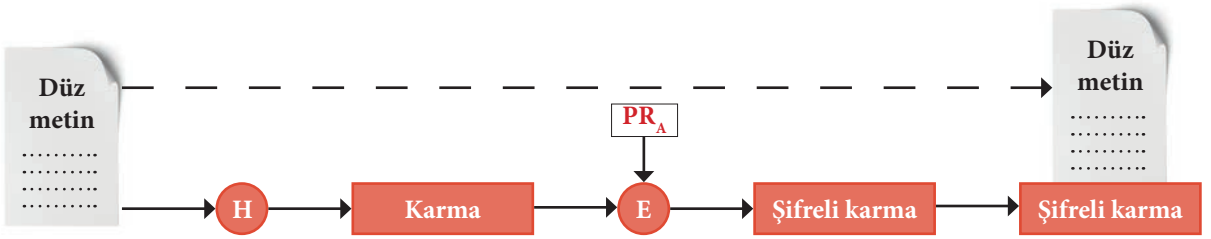
Berk'in göndermek istediği düz metin elde edilmiştir

3.5.6. RSA Algoritması

AES, DES gibi şifreleme algoritmaları, simetrik anahtar yapısını kullanır. Hem gönderici hem de alıcı tarafından anahtarın bilinmesi gerekir. Simetrik şifrelemenin sağladığı hız avantajının yanında bazı dezavantajlar vardır. Anahtarın paylaşımının zorluğu, anahtar yönetimi ve kimlik doğrulama eksiklikleri simetrik şifreleme yönteminin dezavantajlarıdır. Asimetrik şifreleme ise anahtar paylaşımı sorununu ortadan kaldırır, kimlik doğrulama ve bütünlük sağlama imkânı sunar. Bu nedenle asimetrik şifreleme, simetrik şifrelemenin dezavantajlarını aşmak ve güvenli iletişimi sağlamak için kullanılır. Asimetrik şifrelemenin en başında gelenlerinden biri de RSA şifrelemesidir. RSA (Rivest-Shamir-Adleman), adını 1977 yılında Ron Rivest, Adi Shamir ve Leonard Adleman tarafından geliştirilen bu algoritmanın tasarımcılarından almıştır. RSA algoritması, büyük sayılar üzerinde modüler aritmetik işlemlerine dayanır. Şifreleme işlemi, alıcının genel anahtarını kullanarak veriyi şifreler. Şifrelenmiş veri, sadece alıcının özel

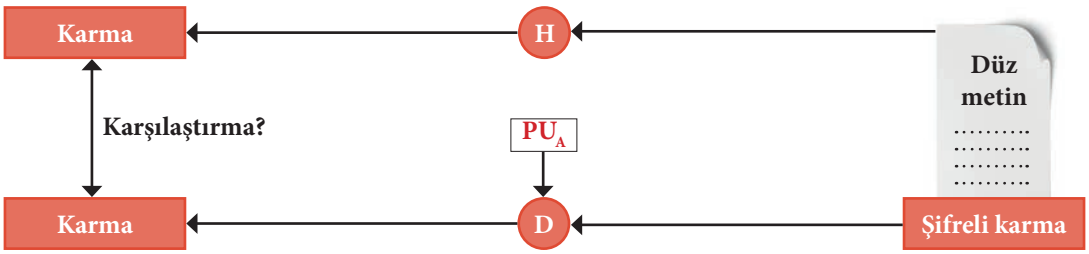
anahtarıyla çözülebilir. Bu sayede veri güvenli bir şekilde iletilirken alıcının kimliği doğrulanabilir ve verinin bütünlüğü sağlanır.

Görsel 3.36 ve Görsel 3.37'de RSA şifrelemesinin çalışma diyagramı verilmiştir.



Görsel 3.36: RSA şifrelemesinde gönderim aşaması

DSA yapısında olduğu gibi burada da hem şifreleme hem özet algoritmaları birlikte kullanılır. Gönderici, içeriğini hazırlar ve içeriğinin özet değerini hesaplar. Hesaplanan bu özet değerini, gizli anahtarı ile şifreler ve şifreli bir özet değeri elde eder. Ardından iletilmek üzere mesaj içeriğine ekler. Burada dikkat edilmesi gereken nokta, mesaj içeriğinin şifrelenmediğidir. Alıcı tarafında gerçekleşen işlemler Görsel 3.37'de verilmiştir.



Görsel 3.37: RSA şifrelemesinde alım ve doğrulama aşaması

Alıcıya ulaşan mesajda şifrelenmemiş bir içerik ve şifrelenmiş bir özet değeri vardır. Şifreli özet değeri, göndericinin açık anahtarı ile alıcı tarafından deşifre edilir. Hesaplanan bu özet değeri ile alınan özet değeri karşılaştırılır. Eşleşme sağlanırsa göndericinin kimliği doğrulanır. Aynı zamanda mesajın bütünlüğü doğrulanır.

RSA şifrelemesinde modüler aritmetik, bölünebilme, en büyük ortak bölen ve asal sayılar gibi birçok cebirsel özellik yer alır. RSA şifrelemesinde kullanılan anahtarların üretiminin matematiksel altyapısı şu şekildedir:

1. Adım: İki büyük asal sayı seçilir. Bu sayılar, sırasıyla "p" ve "q" olsun.

2. Adım: Bu iki asal sayının çarpımı olan "n" sayısı hesaplanır ($n = p \cdot q$).

3. Adım: "n" sayısı için Euler totient [$\phi(n)$] değeri hesaplanır.

4. Adım: Öyle bir "e" sayısı bulunmalıdır ki $\phi(n)$ den küçük ve $\phi(n)$ ile aralarında asal olsun. EBOB [e, Q(n)] = 1 olmalıdır.

5. Adım: Öyle bir "d" sayısı bulunmalıdır ki $d \cdot e \equiv 1 \pmod{\phi(n)}$ olsun.

Seçilen "n", "d" ve "e" sayılarından (n, e) çiftine **açık anahtar**, (n, d) çiftine de **gizli anahtar** denir.

"x" şifrelenecek düz metin olmak üzere **şifreleme** için gönderici tarafında şu matematiksel işlem uygulanır:

$$y = x^e \ (MOD\ n)$$

"y" şifreli metin olmak üzere **şifre çözme** için alıcı tarafında şu matematiksel işlem uygulanır:

$$x = y^d \ (MOD\ n)$$

Örnek Ayşe, Berk'e 4 sayısını mesaj olarak göndermek istesin. Göndereceği mesajın sadece Berk tarafından açılacak şekilde şifrelenmesini de istesin. Bu işlem için RSA algoritmasını kullansın.

Not Bu örnekte kolaylık olsun diye küçük sayılar seçilmiştir. Ayrıca matematiksel işlemler kullanıcı tarafından değil, kullanılan yazılımlar tarafından otomatik olarak gerçekleşir.

Gönderici: Ayşe **Alıcı:** Berk **Açık Mesaj:** 4

Ayşe, göndereceği mesajı Berk'in açık anahtarı ile şifreleyecektir. Bu nedenle Berk tarafında gerçekleşen RSA anahtar üretim işlemleri şunlardır:

1. Adım: İki asal sayı olan "p" ve "q" seçilir.

p = 3 ve q = 11 olsun.

2. Adım: "n" değeri hesaplanır.

n = p . q = 3 . 11 = 33

3. Adım: Φ(n) değeri hesaplanır.

Φ(n) = (p - 1) . (q - 1) = 2 . 10 = 20

4. Adım: Şifreleme için kullanılacak "e" sayısı bulunur. 1 ile 19 arasında olmak üzere 20 ile aralarında asal olan "e" sayısı 3 olsun.

e = 3

5. Adım: Deşifreleme için kullanılacak "d" sayısı hesaplanır.

d . e ≡ 1 (MOD ϕ(n)) olacak şekilde bir "d" sayısı bulunur.

d . 3 ≡ 1 (MOD 20) ise d = 7 olabilir çünkü 7 . 3 ≡ 1 (MOD 20) dir.

d = 7 olarak hesaplanır.

O hâlde şifreleme için kullanılan açık anahtar {n, e} çifti, gizli anahtar ise {n, d} çifti olacaktır.

Berk'in açık anahtarı (PUb) = **{33, 3}**

Berk'in gizli anahtarı (PRb) = **{33, 7}** olacaktır.

Açık anahtar sadece şifreleme için kullanılır ve isteyen herkes ile paylaşılabilir ancak Berk'in gizli anahtarı sadece Berk tarafından bilinir, paylaşılmaz.

Berk tarafından Ayşe tarafına açık anahtar gönderilir. Ayşe, bu açık anahtarı kullanarak "4" mesajını Berk'e göndermek istesin. Bu durumda öncelikle Ayşe'nin açık anahtarı (33, 3) kullanılarak şu şekilde şifreleme yapılır:

$$y = x^e \ (MOD\ n)$$

$$y = 4^3 \ (MOD\ 33) \equiv 64 \ (MOD\ 33) \equiv 31$$

Şifreli mesaj (y = 31) Berk'e gönderilir. Berk, gizli anahtarını (33, 7) kullanarak şu deşifreleme işlemini yapar:

$x = y^d \text{ (MOD } n)$

$x = 31^7 \text{ (MOD 33)}$

$x = 27.512.614.111 \text{ (MOD 33)}$

x = 4 (MOD 33) olarak hesaplanır. Böylece Ayşe'nin gönderdiği "4" mesajı elde edilir.

Not Örnekte anlaşılabilirlik için "p" ve "q" değerleri çok küçük asal sayılar olarak alınmıştır ancak RSA uygulamasında çok daha büyük asal sayılar kullanılır.

RSA algoritmasının sayı dışındaki harfler ve işaretler gibi karakterlere uygulanabilmesi için harflerin ve işaretlerin sayısal hâle dönüştürülmesi gerekir. Bu işlem için yaygın olarak kullanılan yöntem, ASCII kodlarından yararlanmaktır.

RSA; güvenli iletişim, dijital imzalama, anahtar değişimi, kimlik doğrulama gibi çeşitli kriptografik uygulamalarda yaygın olarak kullanılan bir şifreleme algoritmasıdır. RSA'nın hesaplama maliyeti yüksek olduğu için genellikle küçük boyuttaki verilerin şifrelenmesinde, anahtar değişimi gibi spesifik kullanım senaryolarında tercih edilir. RSA şifrelemesi, AES gibi şifrelerden birkaç kat daha yavaş olduğu için simetrik şifrelerin yerini alması amaçlanmamıştır. Uygulamada genellikle AES gibi simetrik bir şifreyle kullanılır.

Sıra Sizde

e = 3, d = 7 ve n = 33 olmak üzere "EDA" kelimesinin RSA ile şifrelenmiş hâlini hesaplayınız.

Burada harflerin sayısallaştırılması için ASCII kodları yerine harflerin alfabedeki sırasını kullanınız (A = 0, B = 1, C = 2, Ç = 3, D = 4, E = 5 ...).

Değerlendirme

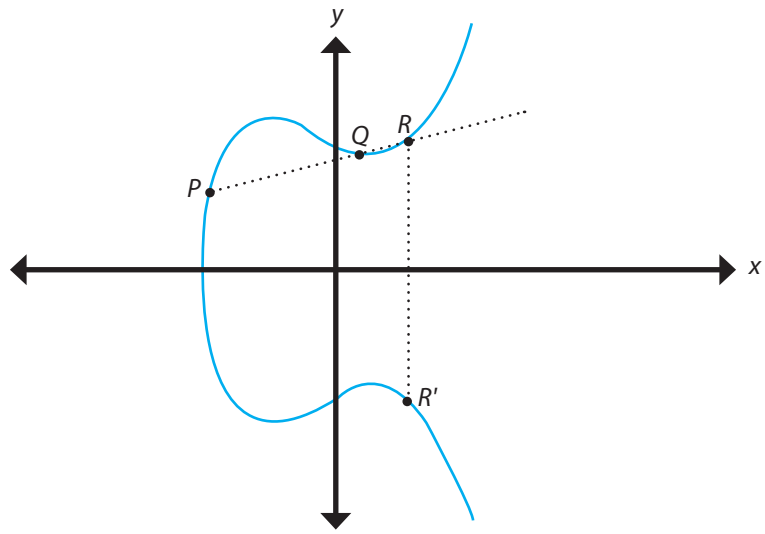
Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. Düz metni doğru şekilde sayısallaştırdı.		
2. Açık anahtarı doğru bir şekilde belirtti.		
3. Gizli anahtarı doğru bir şekilde belirtti.		
4. Düz metnin her harfine şifreleme algoritmasını uyguladı.		
5. Şifreleme sonucunda elde edilen sayıları harflere dönüştürdü.		
"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.		

3.5.7. Eliptik Eğri Algoritması

Eliptik Eğri Kriptografisi (Elliptic Curve Cryptography-ECC), eliptik eğrilere dayanan modern bir açık anahtar algoritmasıdır (Görsel 3.38). ECC; kriptografide açık ve gizli anahtar çifti üretme, dijital imza üretme, anahtar değişimi gibi alanlarda kullanılır.



Görsel 3.38: Eliptik eğri örneği

Matematikte eliptik eğri, $y^2 = x^3 + ax + b$ şeklinde ifade edilen eğrilerdir. Burada "a" ve "b" sayıları sabit sayılardır. Bu eğrinin x eksenine göre simetrik olması, herhangi bir noktadan çizilen doğrunun bu eğriyi en fazla üç noktada kesebilmesi gibi geometrik özellikler, eliptik eğri kriptografisi için etkili ve güçlü bir yöntem sunar.

"M" asal bir sayı olmak üzere bu eğride alınan herhangi bir başlangıç noktasının P (X_1, Y_1) bir "d" tam sayısı ile skaler çarpımı sonucunda elde edilen yeni nokta Q (X_2, Y_2), MOD M'ye göre yine bu eğri üzerindedir. Başlangıç noktası P (X_1, Y_1) olan bir noktanın 3 ile çarpımı (ilk üç tur) işlemi şu şekilde bulunur:

P = (X_1, Y_1)

2P = P + P demektir. O hâlde *(X_1, Y_1) + (X_1, Y_1) = (X_2, Y_2)* (MOD M) elde edilir.

3P = P + 2P demektir. O hâlde *(X_1, Y_1) + (X_2, Y_2) = (X_3, Y_3)* (MOD M) ile elde edilir.

Eliptik bir eğride başlangıç noktası ve kaç tur işlem yapıldığını gösteren "d" değeri verildiğinde son olarak elde edilecek noktanın koordinatının hesaplanması kolaydır ancak başlangıç noktası ve son noktanın koordinatı verildiğinde kaç tur işlem yapıldığının hesaplanması oldukça güçtür. Buradaki güçlük, anlamlı bir zaman diliminde algoritmanın kırılabilir olmayışıdır ve Kesikli Logaritma Problemi'dir. (ECDLP). ECC, anahtar çifti üretiminde kullanılır. Eliptik eğride anahtar üretim aşamaları şu şekildedir:

"E" bir eliptik eğri ve "P" asal sayısı için;

1. Adım: E eliptik eğrisi üzerinde bir üreteç nokta seçilir. Bu üreteç nokta "G" olsun.

2. Adım: Rastgele veya sözde-rastgele bir şekilde, kaç tur işlem yapılacağını belirten bir "d" sayısı seçilir.

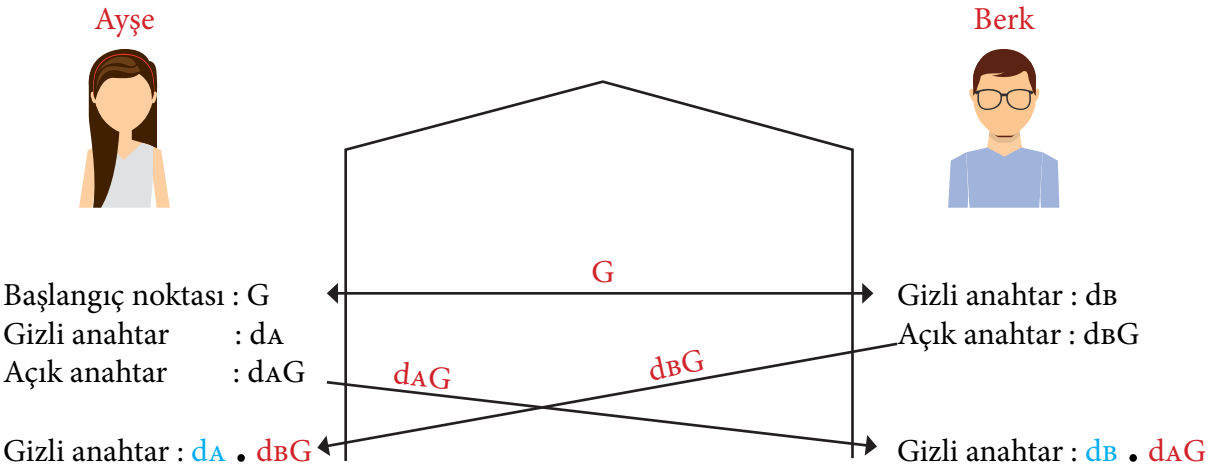
3. Adım: "dG" işlemi gerçekleştirilir ve sonuçta koordinatları (X_T, Y_T) olan bir T noktası elde edilir.

Burada "d" tam sayı cinsinden gizli anahtardır.

"dG" ise (X_T, Y_T) koordinat cinsinden açık anahtardır.

Diffie-Hellman sürecinde bu anahtar çifti, gizli anahtar veya oturum anahtarı üretmek için kullanılabilir. Bu durumda ECC'ye dayalı Diffie-Hellman (ECDH) uygulanmış olur. Benzer şekilde ECC, dijital imza (DSA) üretmek için de kullanılabilir (ECDSA).

Görsel 3.39'daki örnekte ECDH işlemleri, bir başka deyişle eliptik eğri kullanılarak gizli anahtar hesaplanması grafiksel olarak verilmiştir.



Görsel 3.39: Eliptik eğri kriptografisinde anahtar değişimi

ECC, daha kısa anahtar uzunluklarında bile RSA'ya kıyasla daha yüksek güvenlik sunar. Örneğin 256-bit ECC ile sağlanan güvenlik RSA'da 3072-bit ile sağlanır. Ayrıca anahtar üretim süreci daha hızlıdır. Dolayısıyla daha az kaynak kullanılarak işlem yapılmasını sağlar. Bu özellikler, ECC'yi modern ve güçlü bir kriptografi aracı yapar.

3.6. STEGANOGRAFİ

Veriyi gizlemenin güncel bir yöntemi; gönderilecek bir veriyi resim, ses, video gibi bir dosyaya gömmektir. Bu yönteme **steganografi** denir. Gizlenecek veri, örneğin bir resim dosyasına kolaylıkla görünemeyecek şekilde gömülür. Böylece istenmeyen kişilerin gizli bilgiyi okumaları zorlaştırılır.

Steganografide çeşitli teknikler bulunur ancak yapılan işlem genelde bitler üzerine fark edilemeyecek kadar küçük değişiklikler yapma mantığına dayanır. Örneğin 24-bitlik resim steganografisinde piksellerin renkleri gösteren bitlerinin son değerleri değiştirilir. Bilindiği üzere her piksel; kırmızı, yeşil ve mavi (RGB) karmasından oluşan bir değere göre renk alır. Her biri bir bayt olan bu üç değerin karışımı, bir pikselin rengini gösterir. Her piksel için bu üç rengi ifade eden 8-bitin örneğin sadece son bir biti değiştirilirse bir pikselin 3-bit gizli bilgi taşıyabildiği anlamına gelir. Son iki bit değiştirilirse piksel 6-bitlik bir veri taşıyabilir ancak ne kadar çok bit kullanılırsa resimde o derece bozulma yaşanır.

8. Uygulama

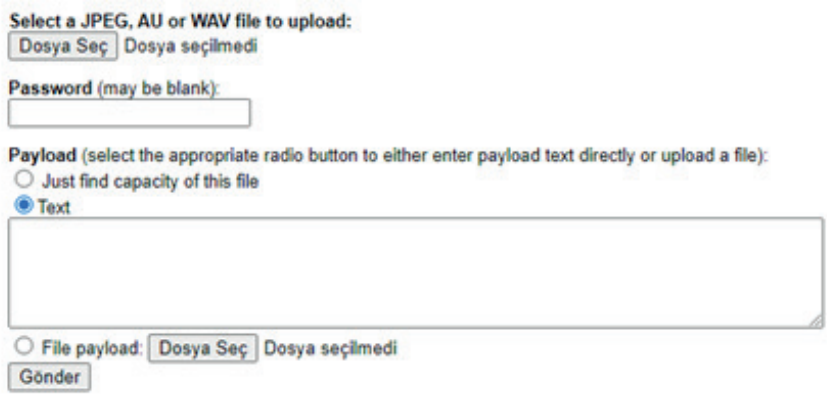
İşlem adımlarına göre https://futureboy.us/stegano/ sayfasındaki çevrimiçi bir steganografi aracını kullanarak hazırlayacağınız bir dokümanı bir resim içine gizleyiniz. Bu uygulama için hazırlayacağınız doküman, bir metin belgesi olabildiği gibi bir resim de olabilir.

1. Adım: Gizlemek istediğiniz mesajı bir metin editöründe açıp yazınız. Örnekteki metin "Ankara" olarak seçilmiştir. Dosyayı GizliMetin.txt olarak kaydediniz.

2. Adım: Gizleyeceğiniz metni gömmek istediğiniz .jpeg formatında bir resim seçiniz.

3. Adım: Çevrimiçi steganografi hizmeti veren bir web sayfasını açınız.

4. Adım: Web sayfasını açtığınızda Encode an image bağlantısını tıklayınız. Açılacak sayfa Görsel 3.40'ta verilmiştir.



Görsel 3.40: Steganografi çevrimiçi sayfa örneği

5. Adım: Sayfanın sol üstünde yer alan Dosya Seç düğmesini kullanarak seçtiğiniz resmi yükleyiniz.

6. Adım: Sayfanın sol altında yer alan Dosya Seç düğmesini kullanarak gizlemek istediğiniz metin dosyanızı yükleyiniz.

7. Adım: Gönder butonuna basınız. Dosyanız resim dosyasına gömülecek ve indirilecektir.

Not Orijinal resim (Görsel 3.41) ile gizli metni içeren resim (Görsel 3.42) arasında gözle görülür bir farklılık bulunmaz.

Görsel 3.41: Orijinal resim

Görsel 3.42: Mesajın gizlendiği resim

9. Uygulama

İşlem adımlarına göre sekizinci uygulamadaki gizli mesajı içeren resim dosyasını, aynı web sayfasını kullanarak resimden elde ediniz.

1. Adım: https://futureboy.us/stegano sayfasında Decode an image bağlantısına tıklayınız.

2. Adım: Açılan sayfada Dosya seç düğmesini kullanarak gizli mesajı içeren resim dosyasını yükleyiniz.

3. Adım: Gönder düğmesine basarak gizli resmi elde ediniz.

Not Türkçe karakterler, kod dönüşümü nedeniyle doğru görüntülenemeyebilir.

ÖLÇME VE DEĞERLENDİRME

A. Aşağıdaki parantezlerin içine cümlelerde verilen bilgiler doğru ise "D", yanlış ise "Y" yazınız.

1. ( ) Klasik kriptografi, modern bilgisayar tabanlı kriptografinin öncüsü olan geleneksel şifreleme tekniklerini ifade eder.

2. ( ) MD5, bir giriş verisinden sabit uzunlukta bir hash değeri üretmeyi sağlayan bir özet fonksiyonudur.

3. ( ) Feistel bir akış şifreleme türüdür.

4. ( ) Kesikli logaritma, verilen bir sayının başka bir sayının hangi üssü olarak elde edileceğini bulmak için kullanılan bir işlemdir.

5. ( ) RSA algoritması simetrik bir şifreleme türüdür.

B. Aşağıdaki cümlelerde boş bırakılan yerlere doğru sözcükleri yazınız.

6. Verinin şifrelenmesinde ve şifresinin çözümlenmesinde aynı anahtar kullanılıyorsa buna ..... denir.

7. ....., gizli anahtarı doğrudan iletmeden her iki taraf için aynı olmasını sağlayan kriptografik bir çözümdür.

8. ..... işlemi, düz metnin her bir karakterinin yerine diğer bir karakter koymayı ifade eder.

9. Resim içinde metin gizleme uygulaması bir ..... örneğidir.

10. ..... daha kısa anahtar uzunluklarında bile RSA'ya kıyasla daha yüksek güvenlik sunan asimetrik şifreleme türüdür.

C. Aşağıdaki soruları dikkatlice okuyarak doğru cevabı işaretleyiniz.

11. Aşağıdakilerden hangisi mesajın değiştirilmediğini doğrulamaya çalışan bir yöntemdir?

A) AES B) DES C) Hill
D) MD5 E) Vigenere

12. Aşağıdakilerden hangisi MD5 fonksiyonunda kullanılan tampon bellek ve bit sayısıdır?

A) 1 adet, 128-bit B) 2 adet, 8-bit C) 4 adet, 32-bit
D) 4 adet, 64-bit E) 8 adet, 32-bit

13. Aşağıda yer alan bit sayılarından hangisi DES yapısında kullanılan anahtarın boyutudur?

A) 32 B) 56 C) 128
D) 256 E) 512

14. Aşağıdakilerden hangisi AES yapısında 256-bit anahtarın kullanımındaki tur sayısıdır?

A) 10 B) 12 C) 14
D) 16 E) 18

15. p = 3 ve q = 13 ise Φ(n) aşağıdakilerden hangisidir?

A) 16 B) 24 C) 33
D) 39 E) 56

D. Aşağıdaki soruların cevaplarını ilgili alana yazınız.

16. Kriptografi neden önemlidir ve hangi alanlarda kullanılır? Açıklayınız.

17. Simetrik ve asimetrik şifreleme arasındaki farklar nelerdir? Açıklayınız.

18. Kuantum kriptografisi hangi yenilikleri getirir? Açıklayınız

MOBİL UYGULAMA GÜVENLİĞİ

4. ÖĞRENME BİRİMİ

KONULAR

4.1. MOBİL UYGULAMA KAVRAMLARI

4.2. MOBİL UYGULAMA KOD EDİTÖRÜ

4.3. MOBİL UYGULAMALARDA GÜVENLİK DENETİMİ

4.4. MOBİL SİSTEMLERDE ZARARLI YAZILIM ANALİZİ

NELER ÖĞRENECEKSİNİZ?

- Mobil uygulamanın temel kavramlarını açıklama
- Android Studio Emülatör yapısını kullanma
- Mobil uygulama geliştirmeleri için kod blok yapısını kavratma
- Android Studio Emülatörü üzerinden farklı mobil işletim sistemlerini kullanma
- Mobil uygulama güvenlik denetimi oluşturma
- Mobil uygulama için kaynak kod denetimi
- Çeşitli mobil işletim sistemlerini kavratma
- Mobil uygulama geliştirme aşamasında statik kod analiz tekniklerini kullanma
- Mobil uygulama geliştirme aşamasında dinamik kod analiz teknikleri kullanma

ANAHTAR KELİMELER

Android, application, apps, cross, developer, emülatör işletim sistemi, IOS, mobil uygulama, platform, security

HAZIRLIK ÇALIŞMALARI

1. Daha önce duyduğunuz farklı mobil işletim sistemlerini arkadaşlarınızla paylaşınız.
2. Size göre dünyada en çok kullanılan mobil işletim sistemleri hangileri olabilir? Düşüncelerinizi arkadaşlarınızla paylaşınız.
3. Simülasyon kavramı size ne ifade ediyor? Arkadaşlarınızla birlikte tartışınız.

4.1. MOBİL UYGULAMA KAVRAMLARI

Mobil uygulama, gündelik hayatta sıkça kullanılan telefon, tablet, televizyon, akıllı saat gibi mobil işletim sistemine sahip cihazlar üzerinde çalışabilmesi için tasarlanmış uygulama yazılımlarına verilen genel bir isimdir. Mobil uygulamaların, işletim sistemine ait bir mağaza üzerinden indirilebilen bir yapısı olsa da sadece paketlenmiş (sıkıştırılmış ve kod yapısı gizlenmiş) hâliyle de mobil cihaz üzerine kurulumu sağlanabilir.

Çalışma stilleri bakımından mobil uygulamalar; yerel çalışan, ağ tabanlı çalışan ve karma (hibrit) tipli çalışan olarak sınıflandırılabilir (Görsel 4.1).

Görsel 4.1: Mobil uygulama geliştirmesinde alternatifler

Yerel mobil uygulamalar, belirli bir işletim sistemi için yazılmıştır (Android, IOS vb.), ait olduğu native kodlara (IOS için Xcode olarak, Android için ise Kotlin ve Java temelli olarak) göre derlenir ve sadece o platform üzerinde çalıştırılabilir. Ağ tabanlı mobil uygulamalar, cihaz belleğinde yer kaplamadan, internet veya ağ bağlantısı sayesinde uzaktan erişime olanak tanıyarak çalıştırılır. Hibrit tipli mobil uygulamalar ise hem web kodlarını içinde barındırır hem de ait olduğu native kodlara bağlı kalır. Bu

sayede istenen her platformda (işletim sistemi) çalıştırılabilir bir hâl alır. Aslında mobil uygulama, mobil görünümlü bir web uygulaması olur (Görsel 4.2).

Görsel 4.2: Cross platform ile hibrit mobil uygulama

4.1.1. Mobil Uygulama Simülatörleri

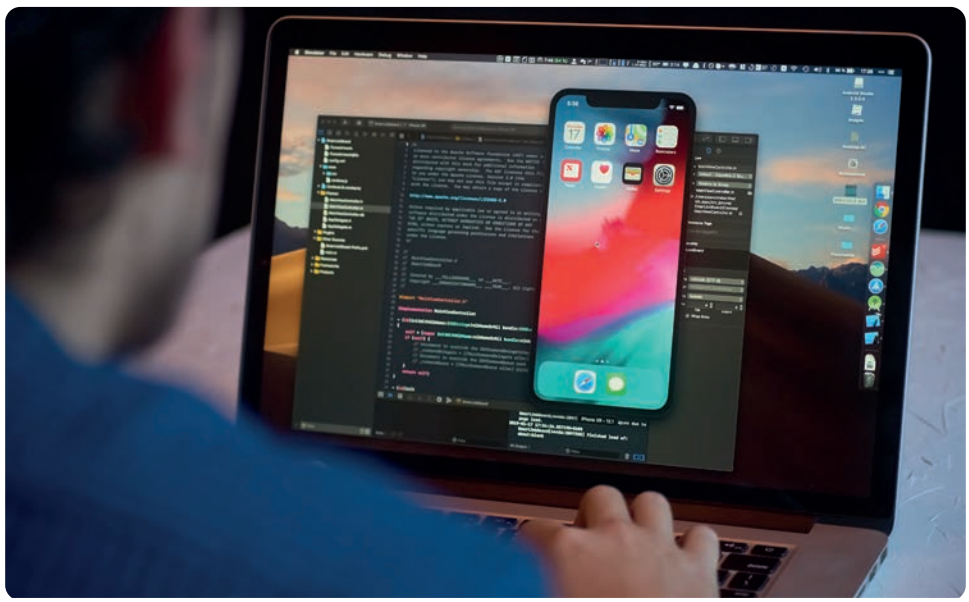
Geliştirilen mobil uygulamaların belirli bir test sürecinden geçmesi gerekir. Mobil uygulamaların testi; donanım uyumluluğunu kontrol etmek, uygulamanın işlevselliği hakkında karara varmak, donanım uyumluluğunu denetlemek ve yazılım güvenirliliği hakkında karara varmak gibi farklı amaçlar için farklı teknikler kullanılarak yapılır (Görsel 4.3).

Görsel 4.3: Simülatör test işlemleri

Mobil uygulama geliştirme aşamasında kullanılan test tekniklerinin amaçları şunlardır:

- **Mobil Fonksiyon ve Davranış Testi:** Gerçekleşebilecek tüm durumlarda mobil fonksiyonların ve davranışlarının doğruluğunun test edilmesidir.
- **Mobil Sistem QoS (Quality of Service) Testi:** Sadece mobil uygulamalarda değil tüm yazılım uygulamalarında karşılaşılan performans testidir. Artan iş yüküne karşı veri trafiğini kontrol etmek ve denetlemek amacıyla kullanılır.
- **Uyumluluk Testi:** Geliştirilen mobil uygulamanın diğer uygulamalarla olan uyumluluğunu ve çapraz bir mobil uygulama geliştirildiyse bu uygulamanın diğer işletim sistemlerinde de çalışma durumunu test etmek amacıyla kullanılan test grubudur.
- **Global Perspektif:** Uluslararası kullanım amacıyla geliştirilen bir mobil uygulamanın çeşitli kültür ve dilsel ortamlara adapte olabilme durumunun kontrol ve analiz edilmesi sürecidir. Bu testin amacı, mobil uygulamanın özellikle global pazara girmeden maliyet ve zaman gibi kaynak yüklerinde azalma sağlamak, belirlenen pazara girmesini de kolaylaştırmaktır.
- **Mobil Sistem Güvenlik Testi:** Kritik veriler için kullanılan, mobil cihazlarda kişisel olarak depolanmış bilgileri hedef alabilecek saldırılardaki durumu ifade eden test grubudur. Saldırılar; MMS, SMS, Wi-Fi, telefon interneti veya Bluetooth üzerinden gelebilir ve yazılımın güvenlik açıklarından yararlanabilir.

Mobil uygulama simülatörleri, gerçek bir mobil cihaza gerek duyulmadan uygulamanın geliştirildiği bilgisayar veya farklı cihazlar üzerine kurulum yapılarak, bunların mobil bir cihaz olarak davranmasına yol açan uygulamalardır (Görsel 4.4). Test sırasında gerçek bir cihaza ihtiyaç duyulmamasından dolayı mobil uygulama simülatörlerinin uygun maliyetli bir yaklaşım olduğu düşünülebilir. Özellikle simülatörlerin çoğunun ücretsiz olduğu düşünüldüğünde, simülatörler etkili bir test yöntemidir. Simülatörlerin dezavantajı, mobil uygulama geliştirme sürecinde uygulanacak testlerin, gerçek sonuçları tam anlamıyla yansıtamamasıdır.

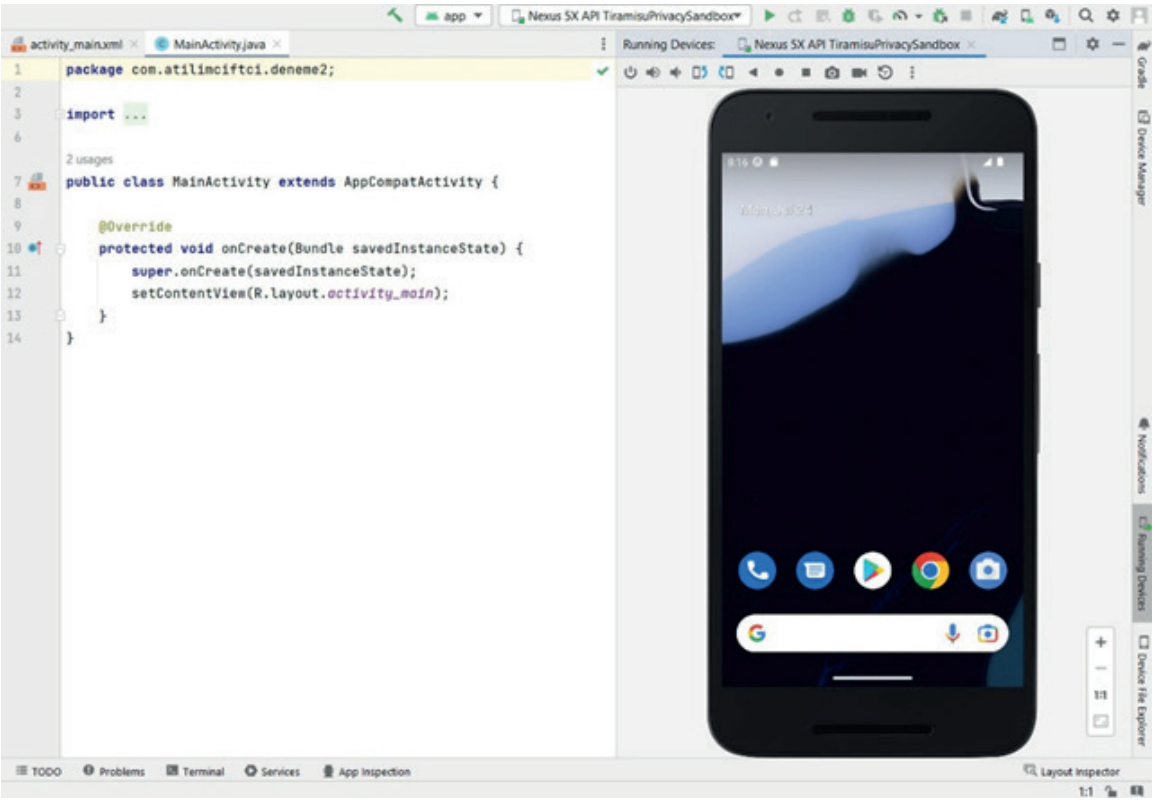


Görsel 4.4: Mobil simülatör örneği

Araştırma

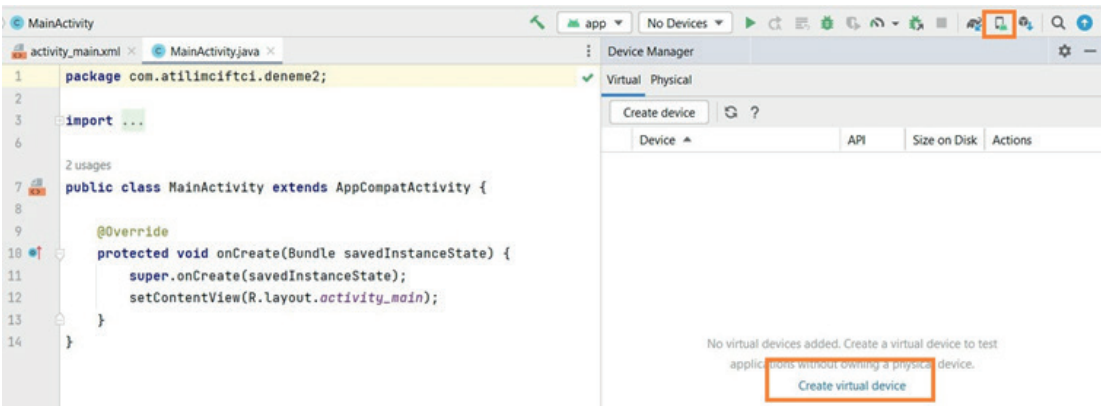
Mobil uygulama geliştirme sürecinde en sık kullanılan simülatörleri araştırınız. Edindiğiniz bilgileri sınıf ortamında arkadaşlarınızla paylaşınız.

Farklı firmaların farklı türevlerine göre mobil uygulama simülatörlerine ait örnekler sıralanabilir. Bunlar arasında **Android Studio Emulator, Genymotion, Xamarin Android Player, Bluestacks, Apple iOS Simulator, Appetize.io** gibi firmalar yer alır (Görsel 4.5).



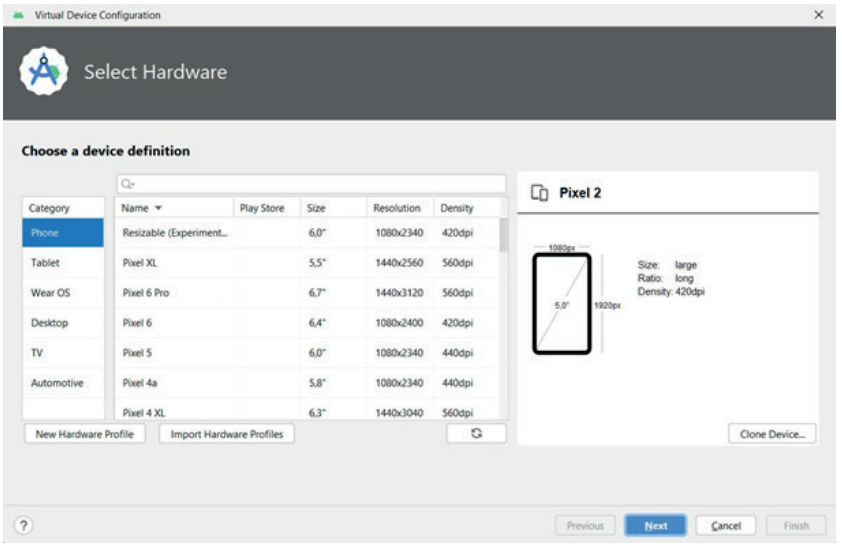
Görsel 4.5: Simülatör ekranı

Android Studio ortamında simülatör kurmak için Görsel 4.6'da görülen Device Manager butonuna basılır. Açılan pencereden Create virtual device (Sanal aygıt kurulumu) butonuna tıklanır.



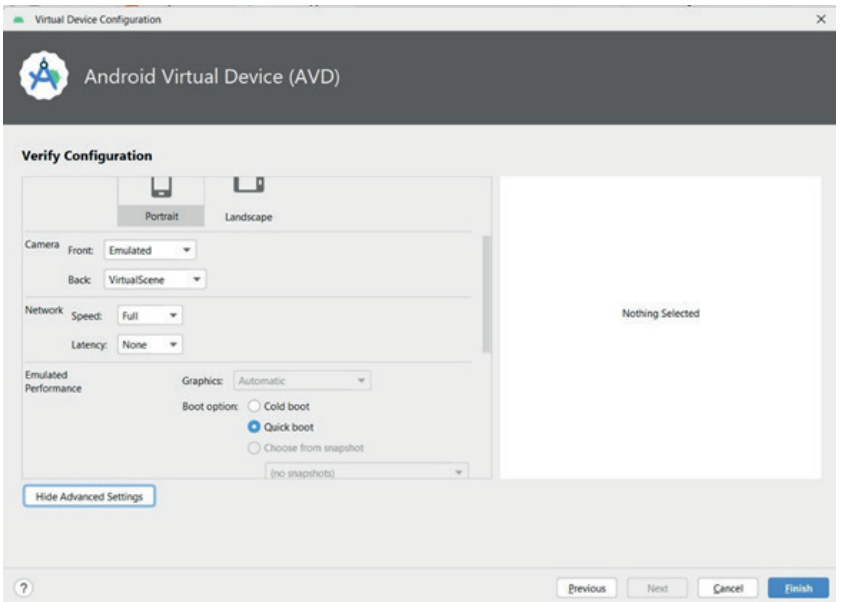
Görsel 4.6: Sanal aygıt kurulumu

Create virtual device butonuna tıklandıktan sonra Virtual Device Configuration menüsü açılır (Görsel 4.7). Bu menüden bir sanal cihaz seçimi yapılır.



Görsel 4.7: Donanım seçimi

Gelen menüden API paketlerine göre bir sistem imajı (system image) seçilir. İmaj dosyası indirilmemişse seçim yapabilmek için imaj dosyasının yanındaki ok işaretine tıklanır ve paket indirilir. İndirme işlemi bittikten sonra Next butonuna basılarak Android Virtual Device (AVD) menüsüne geçilir (Görsel 4.8). Sanal cihaz ayarlarını özelleştirmek için Show Advanced Device seçeneği seçilir. Ayarlar özelleştirilip Finish butonuna tıklanır.



Görsel 4.8: Android Virtual Device menüsü

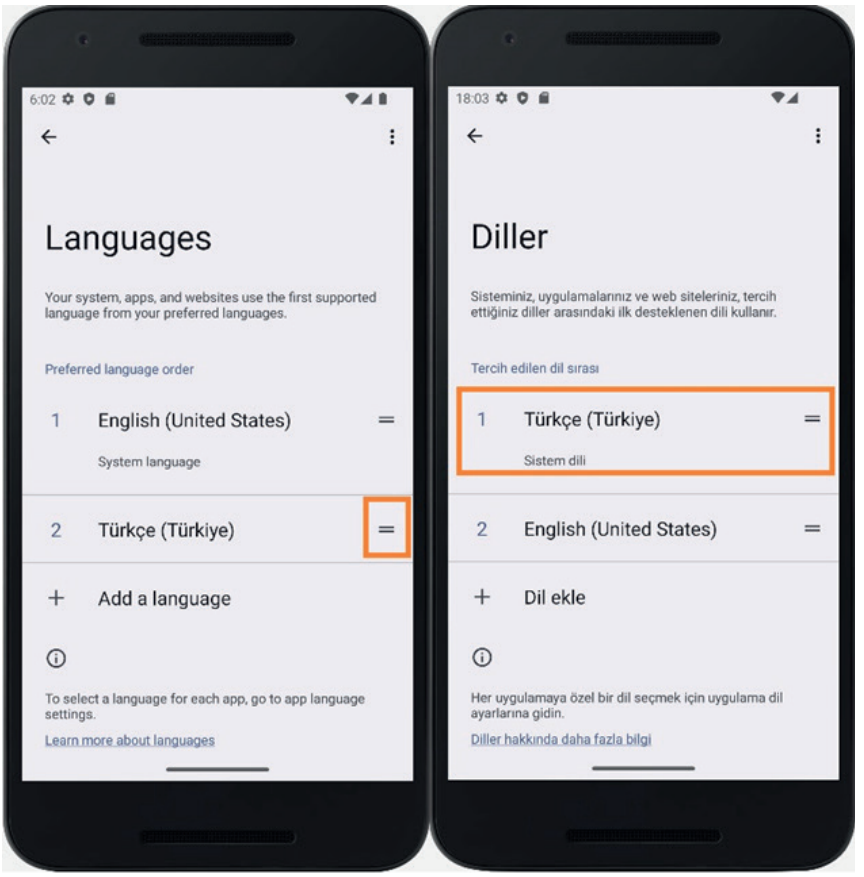
Not

Seçilebilecek en yüksek sistem imajı, API 34'ün yükseltilmesi olan UpsideDownCakePrivacySandbox paketidir. Genelde son sürümlerde hatalar görülebilir. Bu nedenle son sürümün bir önceki versiyonu olan TiramisuPrivacySandbox paketi kullanılabilir

Kurulum işlemi bittikten sonra Device Manager menüsünden kurulan sanal aygıta karşılık gelen Play (Çalıştırma) butonuna tıklanır ve simülatör ilk kez çalıştırılır.

Simülatör ekranı yukarı doğru kaydırıldığında Ayarlar veya Settings menüsü görüntülenir. Görüntülenme seçeneği İngilizce ise (Sistem ayarları genelde ilk yüklemeyi İngilizce yapar.) Settings menüsünden aşağıya kaydırılarak Languages & Input (Diller ve Giriş) seçeneği seçilir ve Languages (Diller) sekmesine tıklanır. Açılan menüden Add a languages (Bir dil ekleyiniz.) sekmesine tıklanır. Seçenekler arasından önce Türkçe seçilir ve yeni gelen penceredeki seçenekler arasından Türkiye seçeneğine tıklanır. Bu sayede Türkçe dil seçeneği simülatöre indirilir.

Görsel 4.9'da görülen simgenin üzerine fare ile tıklanıp Türkçe dil seçeneği 1. sıraya sürüklendiğinde sistem dili Türkçe olarak ayarlanır



Görsel 4.9: Sistem dilini değiştirme

Not Verilen görseller ve bu görsellere ait anlatımlar, simülatörün Android 13 versiyonu içindir. API seviyelerine güncellemeler geldiğinde bunlar farklılık gösterebilir

Sıra Sizde

Bluestacks mobil uygulama simülatörünü kurunuz ve gerekli yapılandırmaları yaparak bu simülatörü test için uygun duruma getiriniz.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. Kurulum dosyasını indirdi.		
2. Kurulumu başlattı.		
3. Yapılandırma işlemini gerçekleştirdi.		
4. Sistem üzerindeki imkânlara göre hafıza belirledi.		
5. Kullanım dilini Türkçe olarak ayarladı.		
"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.		

4.1.2. Mobil Uygulama Kod Blok Yapısı

Mobil uygulamalar, metin tabanlı ve blok tabanlı olmak üzere iki biçimde geliştirilebilir. Blok tabanlı geliştirilen mobil uygulamalar, metin tabanlı uygulamalara göre daha basit ve hızlı çözümler sunsa da etkin, esnek değildir. Ayrıca blok tabanlı mobil uygulamanın geliştirme sürecinde kod yapısına müdahalenin kısıtlı olması sebebiyle dezavantajları bulunur. Metin tabanlı kodlamalarda programlama, ilgili kod editörünün kullandığı yazılım diline göre (Java, Kotlin, XCode vb.) belirli bir algoritmik düzenle oluşturulur (Görsel 4.10). Mobil geliştirme sürecinde kullanılan programlama dilinin söz dizimi kurallarına uyulmalıdır.

```
public class MainActivity extends AppCompatActivity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);
  }
  @Override
  public boolean onCreateOptionsMenu(Menu menu) {
     MenuInflater menuInflater = getMenuInflater();
     menuInflater.inflate(R.menu.gezi_menu,menu);
     return super.onCreateOptionsMenu(menu);
  }
  @Override
  public boolean onOptionsItemSelected(@NonNull MenuItem item) {
     if(item.getItemId()==R.id.girisYap){
     }else if(item.getItemId()==R.id.yerEkle){
     }else if(item.getItemId()==R.id.yerSil){
     }else if(item.getItemId()==R.id.ayarlar){
     }
     return super.onOptionsItemSelected(item);
  }
}
```

Görsel 4.10: Metin tabanlı mobil uygulama geliştirme

Blok tabanlı kodlama ise daha çok küçük yaş grubu bireyler için kodlamanın mantığını en sade ve basit biçimde aktarmaya odaklanır. Bu sayede analitik düşünmeyi geliştirirken eğlenceli görsellerle de bu bireylerin dikkatlerini çeker. Blok tabanlı yazılan kodlamalar aslında metin tabanlı programlamanın kurallarının paketlenmiş biçimlerini kullanmaktan geçer (Görsel 4.11). Her bir blok, başka bir görevi temsil eder.

Görsel 4.11: Blok tabanlı mobil uygulama geliştirme

Blok tabanlı mobil uygulama geliştirme ile metin tabanlı mobil uygulama geliştirme kendi içinde kıyaslanırsa blok tabanlı mobil uygulama geliştirme, metin tabanlı mobil uygulama geliştirmeye göre daha hızlı ve pratiktir. Çabuk öğrenilir, algoritma becerisini daha çok geliştirir, küçük yaş grubu bireyler için daha eğlencelidir. Blok tabanlı mobil uygulamanın dezavantajı, sınırlı bir yapıya sahip olmasıdır. Projeler tam kapsamlı olamaz. Metin tabanlı mobil uygulamada ise geliştirilebilecek projelerin sınırları yoktur.

Araştırma

En sık kullanılan metin tabanlı ve blok tabanlı mobil uygulama geliştirme kod editörlerini araştırınız. Edindiğiniz bilgileri sınıf ortamında arkadaşlarınızla paylaşınız.

4.2. MOBİL UYGULAMA KOD EDİTÖRÜ

Mobil uygulama geliştirme için kullanılabilecek birçok metin tabanlı kodlama editörü bulunur. Mobil uygulama güvenliği içerik yönetiminde bu editörlerden Android Studio ele alınacaktır. Android Studio, Android işletim sistemine sahip cihazlarda çalışabilecek uygulamalar tasarlamak isteyen yazılımcıların tercih listesinde yer alan bir entegre geliştirme ortamıdır (Integrated Development Environment-IDE).

4.2.1. Mobil Uygulama Geliştirmede Güvenlik Denetimi İçin Ortam Hazırlama

Mobil uygulama yazılırken güvenlik denince akla gelebilecek senaryolar arasında öncelikle kullanıcıların kişisel güvenliklerini, cihazlarını ve verilerini korumak üst düzeyde bir öneme sahiptir. Öncelikli amaç, uygulamanın güvenliğini artırmak ve potansiyel olarak sorun yaşanabilecek güvenlik açıklarını maksimum düzeyde azaltmaktır.

Android işletim sisteminde kullanılabilecek mobil uygulamalar için kullanıcılardan alınan izin (permission) durumu mevcuttur. Google bu konuda uzunca bir süre sadece Google Play Store'da (Google oyun mağazası) izni sorar, sonrasında uygulama içinde herhangi bir izin istemezdi. Son dönemde ise kamera, rehber, mesaj, banka bilgileri gibi birçok gizli bilginin günlük hayata ve doğal olarak mobil cihazlara entegresiyle bu uygulamasından vazgeçmiştir. Android, uygulamaların kullanıcıların gizliliğine ve güvenliğine erişimini kontrol etmek için izin sistemini değiştirmiştir. Android, izin sisteminde çok uzun süreden beri yeni benimsediği kuralları kullanır. Kullanıcılar, mağazadan bir uygulama indirmek istediğinde karşılarına gelecek izinler toplu hâlde görünür ve bunları kabul ederek uygulamayı indirir ancak indirme sürecinde sadece mağaza ekranında gösterilen izinler yetmez.

Yüklenen uygulamada kritik izinlerin kabul veya reddedilmesine dair bir onay ekranıyla karşılaşılır (Görsel 4.12). Bir kamera uygulamasının kamera kullanımı için izin istemesi, konum tabanlı bir uygulamanın konum bilgisi için izin istemesi, kritik izinlere örnek olarak verilebilir. Kullanıcılar, uygulamaların izinlerini ayrı ayrı yönetebilir ve ihtiyaç duymadıkları izinleri iptal edebilirler.

Görsel 4.12: İzin onayı

4.2.1.1. Güvenlik İçin Temel Kontrol Listesi

Görsel 4.13: Güvenlik için liste

1. Uygulama içi yetkiler gözden geçirilmelidir (Görsel 4.13). Kullanılacak izinler mantıklı olmalı ve gerekliliğe bağlı olarak sınırlı tutulmalıdır. Gereksiz alınan izinler, kullanıcı mahremiyetini riske atabilir.
2. Hassas verileri güvenli bir biçimde depolamak için Mobil işletim sisteminin ve kod editörünün sunduğu güvenli veri depolama yöntemleri kullanılmalıdır. Gerekirse veriler şifrelenmeli (kriptografi yöntemleri) ve veri tabanı güvenlik önlemleri alınmalıdır.
3. Mobil uygulama yerel bir veri tabanı kullanıyorsa veri tabanını şifrelemek için güvenli bir şifreleme yöntemi tercih edilmelidir. Özellikle kullanıcı kimlik bilgileri kesinlikle kriptografi kurallarına uygun olarak şifrelenmelidir.
4. Web bağlantısı sunan bir mobil uygulama tasarlanırsa web bağlantıları için HTTPS protokolü kullanılmalıdır. HTTPS protokolündeki "S" takısı, Secure (Güvenli) kelimesini ifade eder ve verilerin şifrelenmesini sağlar. Bu şifreleme yöntemi sayesinde veriler, izinsiz erişime karşı korunur.

4.2.1.2. Mobil Uygulamayı Test Etme

1. OWASP (Open Web Application Security Project-Açık Web Uygulama Güvenlik Projesi) kendini, kâr amacı gütmeyen ve yazılım güvenliklerini geliştirmek amaçlı çalışan bir kurum olarak tanımlar. OWASP güvenlik test kılavuzları, mobil uygulamalar için güvenlik testlerini içeren kapsamlı bir kaynak olarak sunulmuştur. Bu kılavuzlar incelenerek uygulamaların güvenlik açıkları belirlenebilir ve düzeltilebilir.
2. Mobil uygulama, kötü amaçlı yazılım taraması araçlarıyla taranmalı ve zararlı içeriklere karşı korunmalıdır.

4.2.1.3. Güvenlik İçin Mobil Uygulamayı Sürekli İzleme

1. Mobil uygulama kullanıcılara gönderilmeden önce mobil uygulamanın beta sürümleri paylaşılmalı ve kullanıcılardan mobil uygulamayla ilgili geri bildirimler alınmalıdır. Kullanıcıların tespit ettiği güvenlikle ilgili sorunları zamanında düzenlemek kritik öneme sahiptir.
2. Güvenlik tehditlerine karşı mobil uygulama güncel kalmalıdır. Mobil uygulamada güvenlik açıklarını düzelten periyodik güncellemeler yayınlanmalıdır.

4.2.2. Mobil Uygulama Geliştirmede Kod Denetimi

Android Studio, çok uzun süredir Android mobil uygulamaları geliştirmek için kullanılan resmî ve en popüler entegre geliştirme ortamlarından biridir. Kod yazma aşamasında geliştiricilere (developer) yardımcı olmak amacıyla bir dizi yerleşik araç ve özellik sunar (Görsel 4.14). Bu algoritmik özellikler sayesinde hata tespiti yapmak kolaylaşır, kod kalitesi artar ve olası hatalar önlenir.

Görsel 4.14: Kod denetimi

4.2.2.1. Hata ve Uyarılar

Kod yazma aşamasında Android Studio, olası hataları otomatik şekilde tespit eder ve kırmızı renkte gösterir (Görsel 4.15). Bu hatalar, uygulamada olabilecek hataların tespiti ve düzenlenmesi için hazırlanan hata ayıklama arayüzü sayesinde bulunur.

```
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    if(item.getItemId()==R.id.girisYap){

    }else ifi(item.getItemId()==R.id.yerEkle){

    }else if(item.getItemId()==R.id.yerSil){

    }else if(item.getItemId()==R.id.ayarlar){

    }
    return super.onOptionsItemSelected(item);
```

Görsel 4.15: Hata denetimi sonucu "if"

4.2.2.2. Kod İçin İpucu ve Kod Tamamlama

Android Studio, kod yazma sırasında yazılımcı (geliştirici) için kod tamamlama önerisi sunar. Bu önerinin yanı sıra kodun kullanımı hakkında bilgi içeren ipuçları verir (Görsel 4.16). Bu sayede kod daha temiz, hızlı ve güvenilir olur.

```
    }else if(item.getItemId()==R.id.yerSil){

    }else if(item.getItemId()==R.id.ayarlar){

m public void onOptionsMenuClosed(Menu menu) {.... Activity
m protected boolean onPrepareOptionsPanel  FragmentActivity
m public boolean onPrepareOptionsMenu(Menu menu)... Activity
m public boolean onMenuOpened(int featu...  AppCompatActivity
Ctrl+Down and Ctrl+Up will move caret down and up in the editor Next Tip
    onOp
}
```

Görsel 4.16: Kod tamamlama için ipucu

4.2.2.3. Hata Ayıklama Özelliği

Android Studio, mobil uygulama geliştiricilere, yazdığı kodları adım adım çalıştırma ve hata ayıklaması yapma imkânı sunar (Görsel 4.17). Uygulama üzerinde oluşabilecek hataları tespit etmek ve düzeltmek için oldukça kullanışlı bir hata ayıklama arayüzü bulunur.

Görsel 4.17: Hata ayıklama arayüzü

4.2.2.4. Lint Araçlarını Kullanma

Lint aracı veya çok kullanılan ismi ile Linter; programlama hatalarını, oluşan bugları, biçimsel hataları ve şüphe duyulan yapıları işaretlemek için kullanılan bir statik kod analiz aracıdır. Linter genellikle kodun okunurluğunu artırır ve belirli standart kalıplarda yazılmasını sağlar. Android Studio da içinde yerleşik bir Lint aracı içerir. Lint; olası sorunları, performans sorunlarını ve stil sorunlarını belirlemede öncü görev üstlenir. Bu durum, uygulamanın kalitesini artırır.

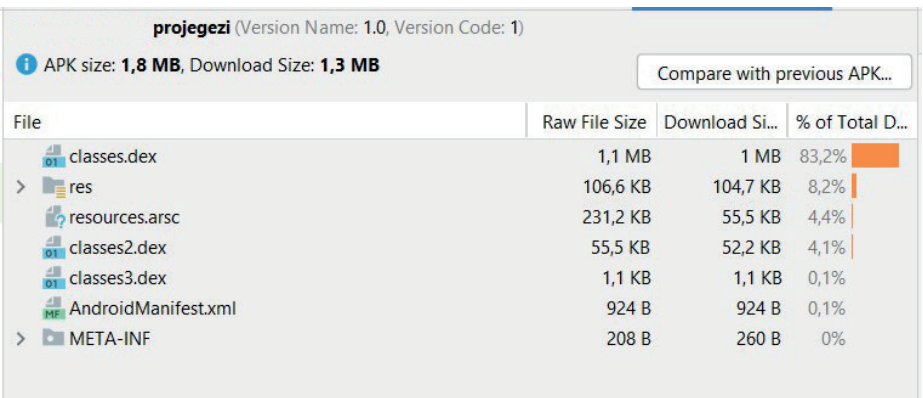
Lint araçları, Android Studio içinde aslında otomatik olarak çalışır. Proje derleme (compile) aşamasında hataları bulması, uyarılar vermesi, Lint'in otomatik olarak çalışmasının göstergesidir ancak Linter, Android Studio üzerinde istendiği zaman manuel olarak da çalıştırılabilir.

4.2.2.5. Statik Kod Analizi Tekniklerini Kullanma

Android Studio kod editörü, istenmesi durumunda statik kod analizi araçları ile yazılan koddaki olası hataları, sorunlu tasarımları ve yaşanabilecek performans problemlerini belirleyebilir. Bu nedenle mobil uygulama yazım sürecinde statik kod analiz teknikleri kullanılmalıdır.

Statik kod analizi ve Linter kavramları birbirine karıştırılabilir. İki kavram birbirine çok benzemekle birlikte Linter, statik kod analizinin sadece bir parçasını oluşturur. Kullanılmayan değişkenleri tespit etmek, kaynakların düzgün kullanılmadığı zamanlarda uyarılar vermek veya Android performansını etkileyebilecek sorunları göstermek gibi çeşitli tarama kuralları ile Lint araçları ilgilenir.

Statik kod analizinde ise bir yazılım çalıştırılmadan önce kodun analiz edilmesi baz alınır (Görsel 4.18). Bu analizde kodun yapısı ve mantığı incelenir ancak çalışma zamanında kodun nasıl davranacağı değerlendirilmez.



Görsel 4.18: Proje analizi

4.2.2.6. Test Araçlarından Faydalanma

Android Studio içinde otomatik test araçları kurulu olarak gelir. Belirtilen bu araçlar, yazılan kodun test edilmesine yardımcı olur. Bu durumda mobil uygulama daha güvenilir bir hâle gelir.

4.3. MOBİL UYGULAMALARDA GÜVENLİK DENETİMİ

Mobil uygulamalarda güvenlik; kullanıcıların şahsi bilgilerini korumak, mobil uygulamayı dış saldırılara karşı korumak ve genellikle güvenilir bir kullanıcı deneyimi sunmak olarak sıralanabilir. Bu noktada dikkat edilmesi gereken temel prosedürler vardır (Görsel 4.19).

Görsel 4.19: Mobil güvenlik denetimi

Mobil uygulama güvenliği için temel prosedürler şunlardır:

- **Veri Güvenliği**

Kullanıcıların, verilerinin güvenli bir biçimde korunduğundan emin olunmalıdır. Şifreler, kredi kartı numaraları, rehber, mesaj kayıtları gibi hassas bilgiler depolanmak zorundaysa güvenilir bir biçimde depolanmalı ve bu bilgilerin aktarılması gereken bir yer varsa hassas veriler oraya aktarılmalıdır. Aktarma ve depolama işlemlerinde de veri şifreleme yöntemleri (kriptografi), güvenli ağ bağlantıları, güvenilir sunucu altyapısı gibi gelişmiş önlemler alınmalıdır.

- **Güvenirlilik Kontrolü**

Uygulamanın kodları düzenli olarak gözden geçirilmelidir. Güvenlik açıkları için çeşitli testler yapılmalıdır. Bu testler aracılığı ile zayıf noktalar ortaya çıkarılmalı ve düzeltilmesi gereken alanlar da belirlenmelidir.

- **Kimlik Doğrulama**

Kimlik doğrulamada, kullanıcı adı ve parola gibi ilkel metotlar yerine iki faktörlü doğrulama (cep telefonu veya maile gelen kod gibi) yöntemleri tercih edilmelidir.

- **Güvenilir Kaynaklar**

Mobil uygulamanın geliştirilme sürecinde kullanılacak kütüphane, Class, API vb. yapılar dikkatlice seçilmelidir. Güncel ve güvenilir kaynakların kullanılmasıyla mobil uygulamanın güvenliği artırılır.

- **Yetkilendirme ve İzinler**

Mobil uygulamanın minimum gereklilikte yetkilendirme yapması sağlanır. Kullanıcılardan, özel erişim izni istenirken neden bu izne ihtiyaç duyulduğu detaylıca açıklanmalıdır.

- **Oturum Yönetimi**

Kullanıcıların login olduğu (oturumun açık olduğu) durumlarda oturumların zaman aşımı süreleri belirlenmeli ve otomatik oturum kapatma özelliği ayarlanmalıdır.

- **Sahte Mobil Uygulama Sorunları**

Geliştirilen mobil uygulamaya karşı mobil uygulamanın sahte versiyonları üretilebilir. Mobil uygulama mağazaya yüklenmeden önce imzalı çıktı alınır. İmzalı olmayan çıktıların mağazaya yüklenmesine izin verilmez. Sahte mobil uygulamalar mağazaya yüklense bile ayrı listelenir veya APK şeklinde güvenilir olmayan sitelerden paylaşılır. Kullanıcılar bu konuda bilgilendirilip mobil uygulamanın orijinal hâlinin indirilmesi teşvik edilmelidir.

- **Güncelleme Denetimi**

Mobil uygulama periyodik aralıklarla güncellenmeli ve mobil uygulamanın güvenlik açıkları kapatılmalıdır. Aynı zamanda mobil uygulama bir kullanıcı gibi izlenmeli, anormal durumlar ve aktiviteler tespit edilmelidir.

- **Saldırılara Karşı Hazırlıklı Olmak**

Her mobil uygulama potansiyel olarak siber saldırılara maruz kalabilir. Geliştiriciler, mobil uygulamalarının kötü niyetli saldırılara karşı dayanıklılığından emin olmak zorundadır. Güvensiz veri girişleri (veri tabanlarına izinsiz veri girmek), güvenlik açıklarına dikkat vb. durumlara hazırlıklı olmak önemlidir.

- **Mobil Uygulamayı Kullanacak Kitlenin Eğitimi**

Özellikle mağazada yayımlanma noktasında mobil uygulama geliştiricisi tarafından uyarılar konulmalıdır. Mobil uygulamanın güvenli bir biçimde kullanılabilmesi için iki faktörlü doğrulama ve güvenli şifre kullanılmalı, halka açık Wi-Fi noktalarına dikkat edilmeli, dolandırıcı linklerden uzak durulmalıdır. Kullanıcıların bu noktalara karşı hassas olması sağlanmalıdır.

Sıra Sizde

Bir sosyal medya uygulamasının potansiyel güvenlik risklerini belirleyiniz ve olası tehdit senaryolarını modelleyerek sınıfta arkadaşlarınızla tartışınız.

4.3.1. Çevrimiçi Araçlardan Yararlanarak Mobil Uygulamalardan Bilgi Toplama

Mobil uygulamalardan bilgi toplamak için günümüz şartlarında oldukça fazla yöntem ve teknik bulunur. Bu teknik ve yöntemler; uygulama ve kullanım analizi, kullanıcıların genel geri bildirimleri, veri trafik analizi, veri kaynaklarına erişim analizi gibi sıralanabilir (Görsel 4.20).

Görsel 4.20: Çevrimiçi veri analizi

Günümüzde çevrimiçi bilgi toplamak için yaygın olarak kullanılan yöntemler şunlardır:

- **Uygulama Analiz Araçları**

Android Studio'da bir mobil uygulama geliştirilmişse ve bu uygulama, Google'ın bulut altyapısı Firebase teknolojisini kullanarak bulut depolama yapıyorsa **Google Analytics** (Firebase teknolojisine sahip tüm uygulamalar kullanır.), **Flurry** (Skype, Facebook, Andry Birds gibi şirketler kullanır.), **Fabric** (Microsoft'a ait veri analiz uygulamasıdır. Son dönemde yapay zekâ desteği duyurdu.) gibi üçüncü taraf uygulama analiz araçları ile kullanıcıların mobil uygulama içindeki hareketleri tespit edilerek kullanım alışkanlıkları genellenebilir. Bu sayede mobil uygulamanın hangi bölümlerinin veya yüzlerinin (Activity, Fragment vb.) daha fazla kullanıldığının analizi yapılabilir.

- **Kullanıcı Geri Bildirimi Sunumu**

Mobil uygulama içine konulacak kullanıcı destek ve geri bildirim formu, mobil uygulama geliştirici için yararlı olacaktır. Kullanıcıların mobil uygulamayı deneyimledikten sonra düşüncelerini ve geri bildirimlerini geliştiriciye sunması önemli bir detaydır. Geri bildirimler incelendiğinde kullanıcıların hangi sorunları yakaladığı ve hangi iyileştirmeleri önerdiği filtrelenerek mobil uygulamanın güvenliğine bir yön çizilebilir.

- **Kullanıcı Yorum İncelemeleri**

Mobil uygulamanın konulduğu mağazalarda (Android için Google Play, IOS için ise App Store), mobil uygulamayı indiren kullanıcıların yaptığı yorumlar izlenebilir. Bu yorumlar sayesinde mobil uygulamanın güçlü ve zayıf yönleri analiz edilebilir.

- **Kullanıcı Davranış Analizi**

Kullanıcılar, indirdikleri mobil uygulamaların bütün özelliklerini kullanmayabilir. Kullanıcı davranış analizi araçları (Appsee, UXCam vb. üçüncü taraf yazılımlar) sayesinde mobil uygulamanın hangi özelliklerinin daha fazla kullanıldığı, kullanıcıların etkileşimleri, mobil uygulamada gezinme türleri incelenebilir ve kullanıcı deneyimini iyileştirmek için bir yol haritası hazırlanabilir.

- **API İzlenmeleri**

Veri transferi yapan mobil uygulamalar için sunucu kullanması nedeniyle sunucu tarafındaki API'lar izlenerek kullanıcı tarafındaki davranışlar incelenebilir ve ortaya çıkan veriler analiz altına alınabilir.

Not Çevrimiçi veri toplama yöntemlerinde en çok dikkat edilecek konu, kullanıcıların veri gizliliği ve güvenliği olmalıdır. Kullanıcı rızası olmadan toplanacak bilgiler sonradan büyük sorunlara yol açabilir. Bu nedenle mümkün olduğunca az kişisel veri toplamaya hassasiyet gösterilmelidir. Toplanan bu bilgilerin etkileşime sunulacağı durumlarda kriptografi teknikleri ile şifrelenerek etkileşime girildiğinden emin olunmalıdır.

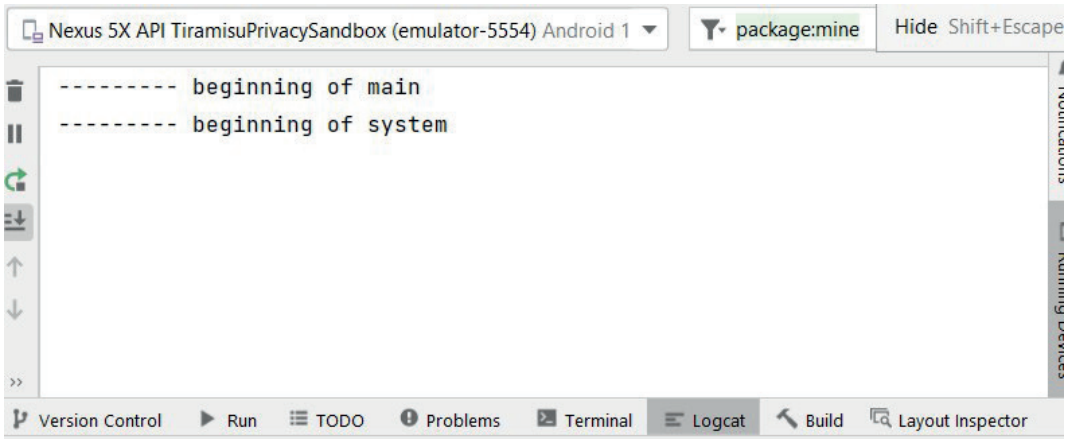
4.3.2. Çevrimdışı Araçlardan Yararlanarak Mobil Uygulamalardan Bilgi Toplama

Her mobil uygulama geliştirme editörünün kendine has çevrimdışı analiz araçları bulunur. Android Studio, temelinde bir entegre geliştirme ortamıdır. Bir başka deyişle yazılımcıların verimli biçimde yazılım kodu geliştirmesi için tasarlanan bir yazılım uygulamasıdır. Doğal olarak Android Studio, mobil uygulamalardan bilgi toplamak için üretilmemiştir. Bu nedenle bilgi toplamak için kısıtlı özelliklere sahiptir. Buna rağmen bazı özellikleri sayesinde performans, hata gidermeleri ve parametre hataları hakkında bilgiler alınabilir ve çözüm yolları sunulabilir. Bu bilgiler, mobil uygulamanın geliştirilmesinde referans noktasıdır.

Android Studio'da çevrimdışı kullanıma uygun bazı araç ve yöntemler şunlardır:

- **Logcat**

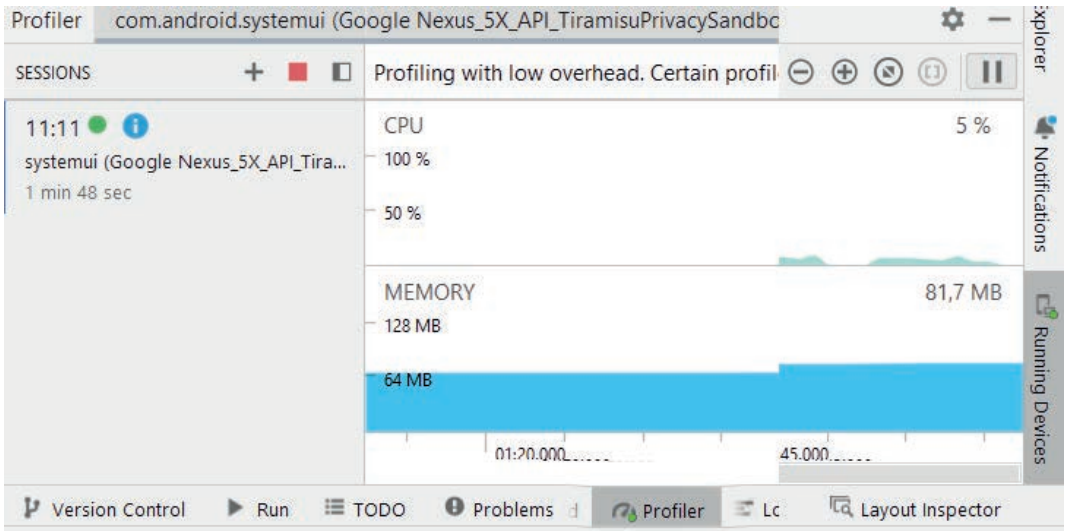
Android Studio kod editörüyle birlikte gelen Logcat, yazılımın çalışma zamanlarında oluşturduğu günlük kayıtları tutabilen bir yapıdadır (Görsel 4.21). Bu veriler, analiz edilmeye de uygundur. Hata ayıklama işlemlerinin bu bölümden yapılması nedeniyle mobil uygulama her çalıştırıldığında alınan hata ve uyarılar depolanır ve sonrasında görüntülenir.



Görsel 4.21: Logcat bilgi ekranı

- **Profil**

Android Studio içinde yerleşik olarak bulunan Profiler; mobil uygulamanın bellek kullanımı, işlemci performansı, veri ve ağ trafiği gibi parametreleri takip etme konusunda referans olarak bilgiler sağlar (Görsel 4.22). Ayrıca memory profil özelliği ile olabilecek sızıntıların ve gereksiz bellik kullanımlarının önüne geçebilecek bir analiz sunar. Bu sayede mobil uygulamanın performansı analiz edilebilir ve geliştirilebilir.



Görsel 4.22: Profiler bilgi ekranı

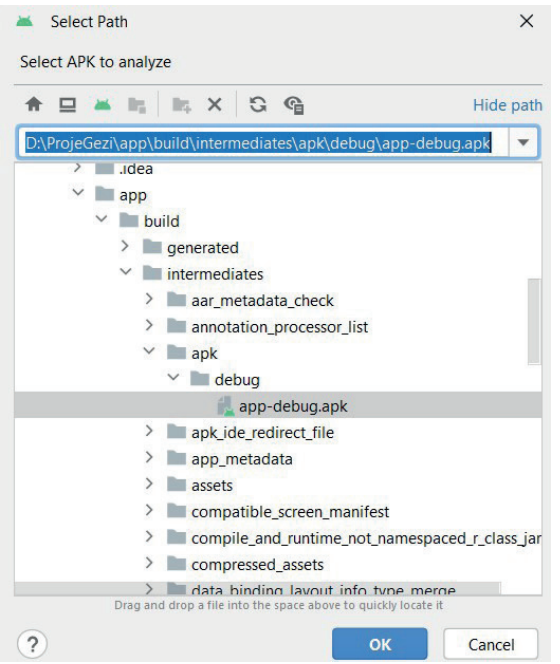
- **Lint Analizi**

Lint analizleri, Android Studio içinde proje dosyalarını analiz eder ve potansiyel olarak oluşabilecek hataları ortaya çıkartır. Stil ile ilgili problemleri, performans iyileştirmeleri için gerekenleri tespit eder. Android Studio içindeki Lint araçları ile kod kalitesi de artırılabilir.

- **APK Dosyası Analizi**

Android Studio ile APK dosyalarının analiz edilmesi ve içeriğinin incelenmesi imkânı mevcuttur. Dosya içinde yer alan kaynaklar, ekli kütüphaneler vb. diğer özellikler bu analiz ile incelenebilir.

Analiz edilmesi istenen APK dosyası, proje klasörü içinde bir konuma çıkartıldıktan sonra Build sekmesindeki Analyze Apk seçeneği ile istenen apk seçilir (Görsel 4.23). Bu seçenek, compile edilen (derlenen) APK dosyasını decompile (çözme) işlemine tabi tutar ve kaynak kodlarını açığa çıkartır. APK dosyasının çıkarımı sol tarafta yer alan Project menüsünde gösterilir. Buradan AndroidManifest.xml dosyasına, görsellere ve diğer kaynaklara göz atılabilir.



Görsel 4.23: Select APK to analyze

Not Bir başkasına ait APK dosyaları analiz edilirse (decompile) içerik ve kodlar, etik ve yasal kullanım açısından dikkatle incelenmelidir. Herhangi bir telifli içerik ve kaynak kod kullanımı veya kaynak kodun değiştirilmesi yasal olarak sorunlara yol açar.

4.4. MOBİL SİSTEMLERDE ZARARLI YAZILIM ANALİZİ

Mobil sistemlerde zararlı yazılım analizleri için çeşitli yöntemler bulunur. Bu yöntemler, mobil cihazlarda tespit edilen yazılımları algılamak ve bu yazılımlardan kurtulmak için geliştirilen ve kullanılan teknikleri kapsar.

Zararlı yazılım analizleri genellikle araştırmacılar ve güvenlik uzmanları tarafından kullanılır. Normal bir developer çok sık bu yöntemlere başvurmaz. Uzmanlar, tespit ettiği zararlı yazılımları bulmak ve etkilerini anlamak için türlü teknikleri birleştirerek farklı sonuçlara ulaşabilir. Yapılan analizler de kullanıcıların cihazlarına bulaşmayı önlemeye önem verirken veri sızdırmasını, istenmeyen reklam

gösterimini, cihazlara zarar veren çeşitli aktiviteleri sınırlamaya özen göstermek üzerine bir çerçeve çizer (Görsel 4.24).

Görsel 4.24: Zararlı yazılım (Malware)

Analiz yöntemleri; statik kod analizi, dinamik kod analizi, kullanıcı incelemeleri, imza tabanlı tarama, güvenlik çerçeveleri ve yapısal analiz olarak sıralanabilir.

4.4.1. Statik Kod Analizi

Android uygulama geliştirme işleminde statik olarak (staticall-typed) yazılan Java ve Kotlin yazılım geliştirme mimarisi dilleri kullanılır. Bu dillerin en önemli avantajı, yazılım çalıştırılma zorunluluğu olmadan kaynak kodlarının analizinin yapılabilmesidir. Statik kod analizi, derlenmeye (compile) hazır durumdaki bir uygulamanın projesinin kaynak kodlarının incelenerek olası güvenlik açıklarının ve zafiyetlerinin tespit edilmesi işlemidir.

Android Studio üzerinden Checkstyle, Ktlint, Android Lint Tool, Detekt, Sonarqube gibi statik kod analizi yöntemleri kullanılabilir. Statik kod analizini gerçekleştirmek için bunların dışında da etkili paket araçlar vardır. Bunlar arasında MobSF (Mobile Security Framework), QARK (Quick Android Review Kit), AndroBugs, FindBugs, Fortify Static Code Analyzer, DroidMat, RiskRanker, Woodpecker gibi ek paket uygulamaları öne çıkar.

4.4.1.1. Checkstyle

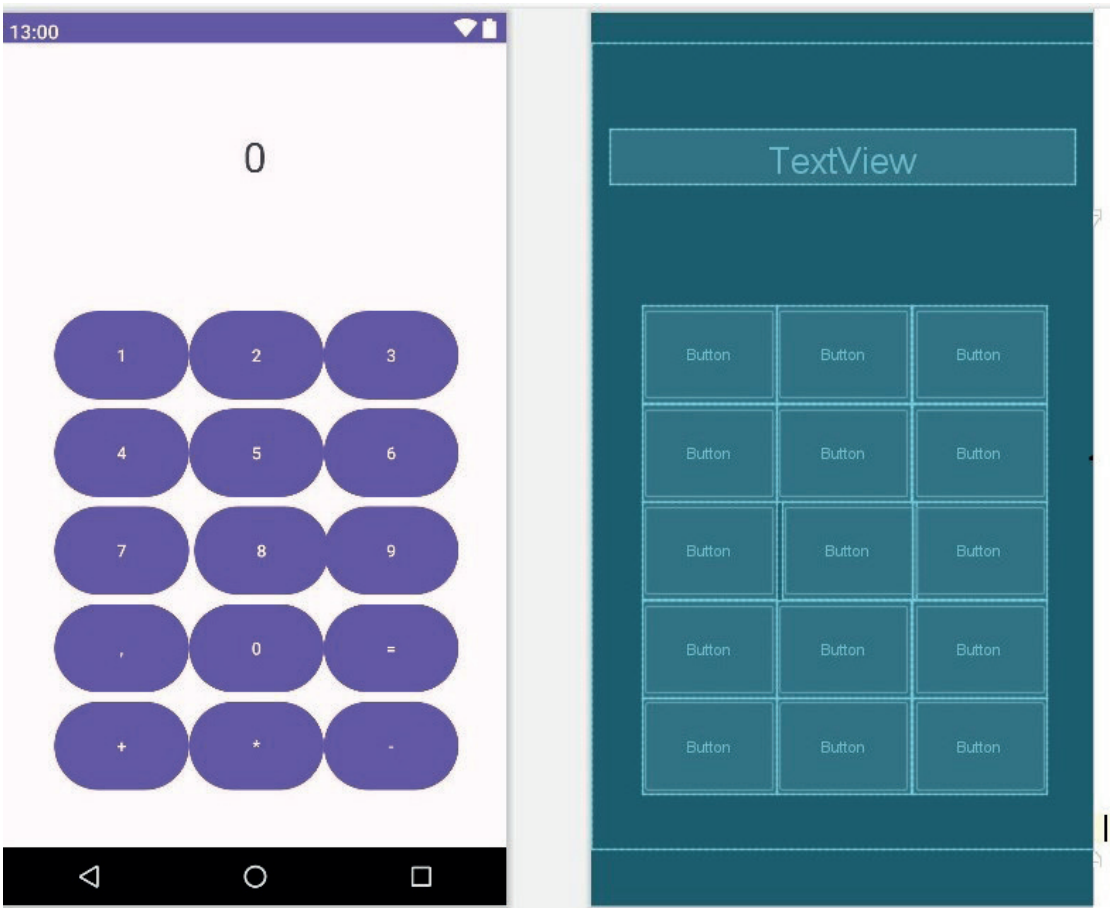
Checkstyle, mobil uygulama geliştiricilerin, Java dilinin kurallarını baz alarak kodlama yapıp yapmadığını kontrol etmek için kullanılan güçlü ve esnek bir statik kod analiz yöntemidir. Java kodlarını otomatik olarak kontrol etmekle birlikte bu statik kod analiz yöntemi Android Studio içinde yüklü şekilde gelmez. Android Studio, eklenti olarak Checkstyle kullanılmasına izin verir. Checkstyle eklentisi, projenin Java kaynak kodları üzerinde Checkstyle analizi kullanarak kalite kontrolleri gerçekleştirir ve bu kontrollerden raporlar oluşturur.

1. Uygulama

İşlem basamaklarına göre Android Studio üzerinde, Java dilinde hazırlanan bir Hesap Makinesi uygulamasının görsel arayüzünü ve arayüzdeki nesnelerin tanımlanmasını içeren Java kaynak kodlarını, Checkstyle yöntemiyle statik kod analizi yapınız.

1. Adım: Android Studio'da boş bir proje açınız.

2. Adım: Görsel 4.25'te görüldüğü gibi görsel bir arayüz oluşturunuz.



Görsel 4.25: Hesap makinesi

Not

Button isimleri sırası ile; "tus1", "tus2", "tus3", "tus4", "tus5", "tus6", "tus7", "tus8", "tus9", "tusVirgul", "tus0", "tusEsittir", "tusArti", "tusCarpi", "tusEksi" şeklinde verilecektir. TextView ismi ise "textView_Gosterge" şeklinde girilecektir.

3. Adım: Java dosyası için şu kodları kullanınız (Görselde de aynı kodlar verilmiştir.):

```
package com.ad_soyad.hesapmakinesi;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
                Button
tus1,tus2,tus3,tus4,tus5,tus6,tus7,tus8,tus9,tus0,tusCarpi,tusArti,tusEksi,tusVirgul,
tusEsittir;
   String metrik="0";
   TextView textViewGosterge;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      tus1 = findViewById(R.id.tus1);
      tus2 = findViewById(R.id.tus2);
      tus3 = findViewById(R.id.tus3);
      tus4 = findViewById(R.id.tus4);
      tus5 = findViewById(R.id.tus5);
      tus6 = findViewById(R.id.tus6);
      tus7 = findViewById(R.id.tus7);
      tus8 = findViewById(R.id.tus8);
      tus9 = findViewById(R.id.tus9);
      tus0 = (Button) findViewById(R.id.tus0);
      tusArti = findViewById(R.id.tusArti);
      tusEksi = findViewById(R.id.tusEksi);
      tusCarpi = findViewById(R.id.tusCarpi);
      tusVirgul = findViewById(R.id.tusVirgul);
      tusEsittir = findViewById(R.id.tusEsittir);
      textViewGosterge = findViewById(R.id.textView_Gosterge);
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
}
```

Not findViewById içinde yazan tus1, tus2 vb. atamalar, görselde eklenen butonların id değerleridir. Butonların id'leri ilgili şekilde düzenlenmelidir.

4. Adım: File (Dosya) menüsünden Settings (Ayarlar) veya Preferences (Tercihler) seçeneğine gidiniz (Bu seçenekler, platformunuzun versiyonuna göre farklılık gösterebilir.).

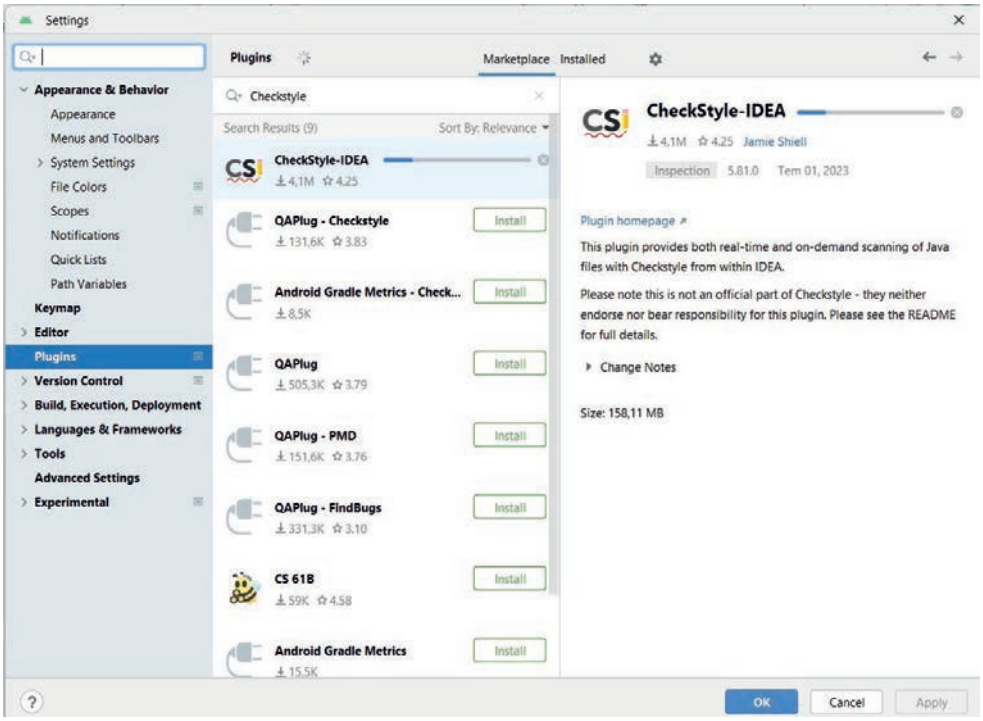
5. Adım: Açılan pencereden Plugins (Eklentiler) sekmesini seçiniz.

6. Adım: Marketplace veya Browse repositories (Depolara göz at) butonunu tıklayarak eklenti arama penceresini açınız.

7. Adım: Arama kutusuna Checkstyle yazınız ve uygun eklentiyi bulunuz.

8. Adım: Eklentiyi bulduğunuzda Install (Yükle) düğmesine tıklayarak eklentiyi yükleyiniz (Görsel 4.26).

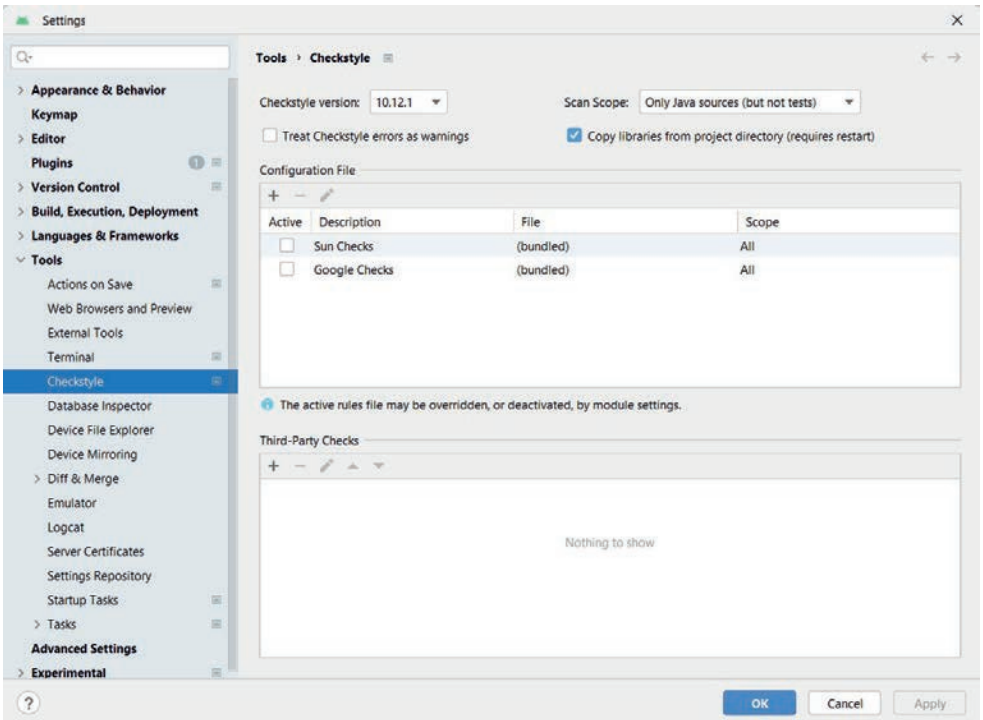
9. Adım: Restart (Yeniden başlatma) istediğinde Android Studio'nun yeniden başlatılmasını onaylayınız.



Görsel 4.26: Chekstyle eklentisi yükleme ekranı

10. Adım: Projeyi Android Studio'da açınız.

11. Adım: File>Settings seçeneğine tıklayınız. Açılan pencereden Tools menüsü altındaki Checkstyle seçeneğine tıklayınız (Görsel 4.27).



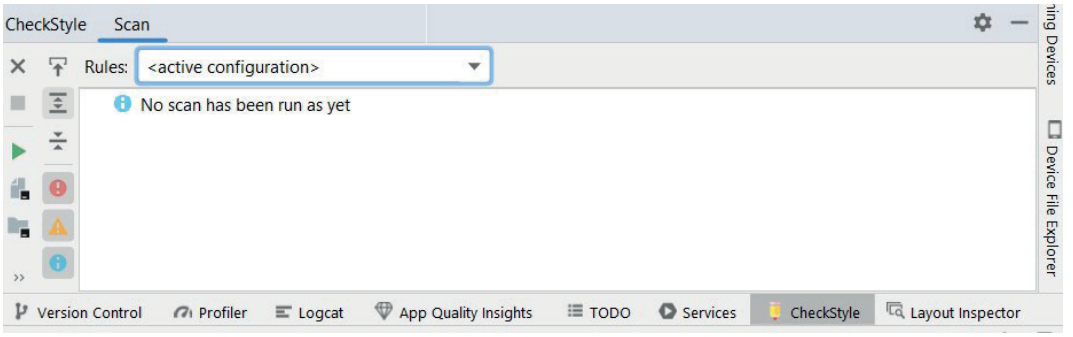
Görsel 4.27: Settings menüsü ve Checkstyle seçeneği

12. Adım: Checkstyle seçeneğinde görünen Sun Checks ve Google Checks seçeneklerini seçiniz. OK butonuna tıklayınız.

Not

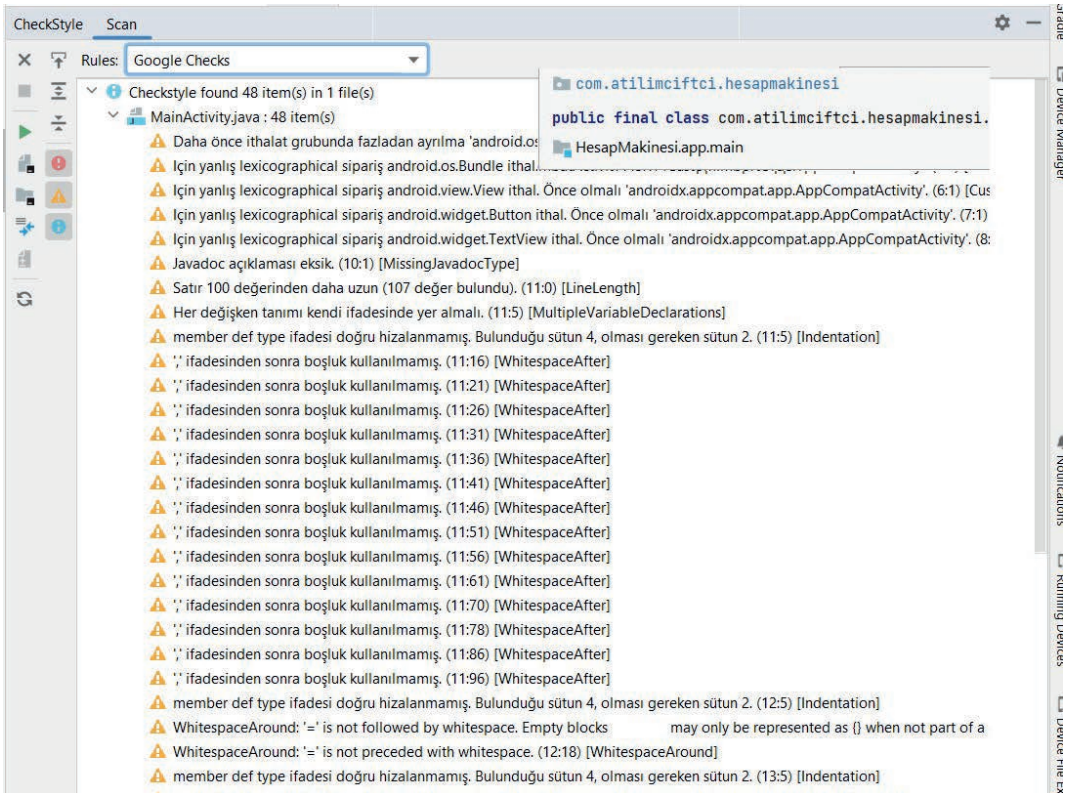
Sun Checks ve Google Checks, hazır birer stil inceleme dosyalarıdır. Bu dosyalar, .xml biçimlidir. Sun Checks, checkstyle eklentisine ait style kuralları içerirken Google Checks, Java diline özgü yazılım biçimi kurallarını içerir. Proje, istenen kurallara göre analiz edilebilir. İstenirse geliştiricinin kendi kurallarının yer aldığı bir .xml dosyası da hazırlanabilir.

13. Adım: Android Studio'nun Footer menüsüne eklenti olarak gelen Checkstyle butonuna tıklayınız (Görsel 4.28).



Görsel 4.28: Checkstyle seçeneği

14. Adım: Rules (Kurallar) seçeneğinde bulunan Combobox'a tıklayınız. Yazılan kodların hangi kurallara göre taranmasını isterseniz (Google Checks, Sun Checks vb.) bu kurallardan birini seçiniz. Seçimin solunda bulunan Play butonuna tıklayınız. Sonuçlar Görsel 4.29'da görüldüğü gibi listelenecektir. Düzeltilmek istenen uyarının üzerine çift tıklanırsa imleci belirtilen satıra götürecektir.



Görsel 4.29: Google Checks sonuçları

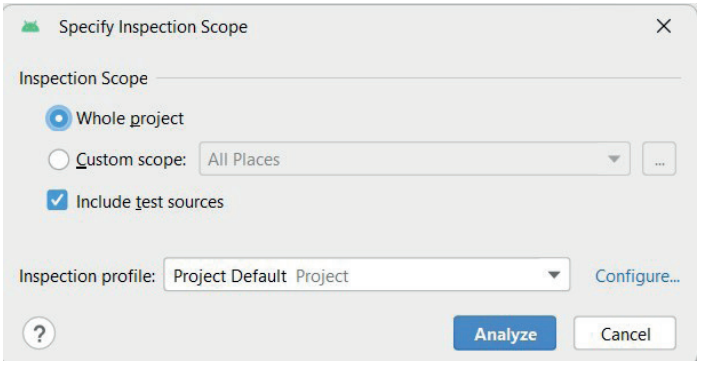
4.4.1.2. Ktlint

Android Studio için resmî dillerden biri de Kotlin'dir. Kotlin, Java diline bir alternatif olarak ortaya çıkarılmıştır. Kotlin, Java ile tam uyumlu çalışmakla birlikte mobil uygulama geliştirmede yaygın olarak kullanılır. Checkstyle, Java için neyi ifade ederse Ktlint de Kotlin için aynı özelliği ifade eder. Başka bir deyişle Kotlin için Ktlint kullanılabilirken Java için Ktlint uygun değildir. Bunun yerine Java için diğer kod stil denetleyicilerinin kullanılması gerekir.

4.4.1.3. Android Lint Tool

Android Lint, Android uygulamalarında kod kalitesini ve performansını iyileştirmek için kullanılan bir statik kod analiz aracıdır. Bu araç; kodda olası hataları, stil kılavuzuna uymayan kodları ve performans sorunlarını tespit etmek için kullanılır. Android Studio ile birlikte gelir ve projelerde doğrudan kullanılabilir. Android Lint Tools kullanımı için yapılması gerekenler şunlardır:

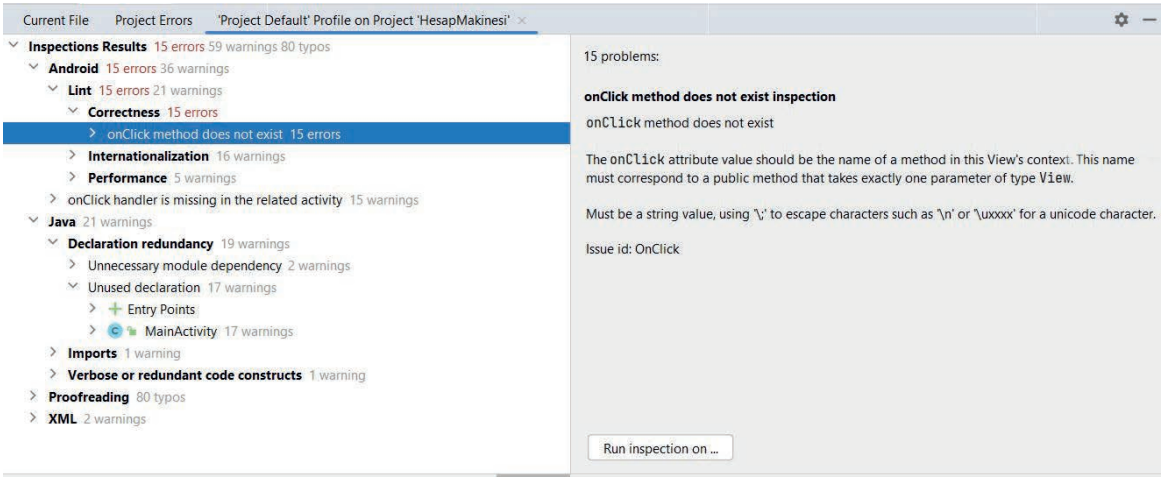
- Android projesi oluşturulmalıdır ve kodlar hazır hâle getirilmelidir.
- Lint raporlarını görüntülemek ve Android Lint'i çalıştırmak için Code menüsü altındaki Inspect Code seçeneğini seçilir. Çıkan pencerede aktif projeyi analiz etmek için Whole project seçeneği seçilip Analyze butonuna tıklanır (Görsel 4.30). Bu seçenek, kod tabanını analiz eder ve Lint raporunu oluşturur (Görsel 4.31). Lint'in hangi kontrolleri yapacağı ve hangi kontrollerin raporlanacağı bu pencereden seçilebilir.



Görsel 4.30: Inspect Code

- Lint sonuçlarını incelemek için Lint analizi tamamlandığında sonuçlar, Inspection Results veya Lint panelinde görülür (Görsel 4.31). Bu panelde, proje dosyalarında tespit edilen Lint sorunları ve uyarılarını listeleyen bir rapor görülür.

- Lint sorunlarının düzeltilmesi için Lint raporunda listelenen her bir sorunun yanında açıklama ve düzeltme önerileri bulunur (Görsel 4.31). Sorunları düzeltmek için önerilen adımlar takip edilebilir.



Görsel 4.31: Analiz sonuçları

Sıra Sizde

Checklist ile analiz edilen Hesap Makinesi uygulamasının statik kod analizini Android Lint aracıyla gerçekleştiriniz.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. Hesap Makinesi isminde yeni bir Android projesi oluşturdu.		
2. Projenin Activity (uygulamanın görünen yüzü) sayfasını oluşturdu.		
3. Activity'e butonları ekledi.		
4. Butonların tanımını yaptı.		
5. Lint aracı ile Analyze penceresine girdi.		
6. Proje kodları için analyze çalıştırdı.		
7. Ulaşılan sorunların açıklamasını yaptı.		
"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.		

4.4.1.4. MobSF (Mobile Security Framework)

MobSF (Mobile Security Framework), geliştirilmiş bir mobil uygulamanın güvenliğini test etmek için kullanılan açık kaynaklı bir yazılımdır. MobSF hem Android hem de iOS uygulamalarının güvenlik testlerini gerçekleştirebilir. Sunulan bu çerçeve, statik ve dinamik analiz tekniklerini kullanarak mobil uygulamaların güvenlik açıklarını tespit etmeye yardımcı olur.

MobSF'yi kullanabilmek GitHub üzerinden kurulum dosyası cihaza indirilmelidir veya bu aracı online kullanmak için MobSF'nin resmî sitesindeki "https://mobsf.live" adresine gidilmelidir. "Upload & Analyze" butonu ile test edilmesi istenen mobil uygulamaya ait APK veya IPA dosyası yüklenir. Ardından MobSF, mobil uygulamayı analiz etmeye başlar. Statik analiz, dinamik analiz ve diğer güvenlik testleri otomatik olarak gerçekleşir. MobSF, analiz tamamlandığında sonuçları bir rapor olarak sunar. Bu rapor; mobil uygulamanın güvenlik durumu, tespit edilen güvenlik açıkları ve riskler hakkında detaylı bilgi içerir.

Sıra Sizde

Hesap Makinesi uygulamasının APK dosyasını çıkartarak MobSF aracıyla statik kod ve güvenlik analizini gerçekleştiriniz.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. MobSF kurulum dosyasını indirdi.		
2. Kurulumu başlattı veya MobSF'nin resmî sitesine online giriş yaptı.		
3. APK dosyasını çıkarttı.		
4. APK dosyasının analizini başlattı.		
5. Analiz sonuçlarını inceledi.		
"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.		

4.4.1.5. QARK (Quick Android Review Kit)

QARK (Quick Android Review Kit), MobSF gibi mobil uygulamalarının güvenlik testlerini gerçekleştirmek için kullanılan açık kaynaklı bir yazılım aracıdır. QARK, statik kod analiz tekniklerini kullanarak mobil uygulamalarında potansiyel güvenlik açıklarını tespit etmeye yardımcı olur. Bu açıklar; kötü niyetli kullanım, veri sızıntısı, diğer güvenlik riskleri gibi konuları içerebilir. Ayrıca QARK, Python tabanlı bir yazılım aracıdır ve bu nedenle Python'un veya bileşenlerinin cihazda yüklü olması gerekir.

QARK öncelikle bilgisayara indirilmelidir. Bu indirme işlemi, QARK'ın GitHub deposundan veya resmî web sitesinden yapılabilir. Test edilmek istenen Android uygulamasının APK dosyasının yolu ve adı

belirlenir. Bu APK dosyasını QARK ile analiz etmek için terminal veya komut istemcisi açılır ve QARK çalıştırılır. QARK'ı terminalde çalıştırmak için örnek kod "python qark.py --apk C:\Users\username\Desktop\myapp.apk" şeklindedir. QARK, APK dosyasını analiz ettiğinde potansiyel güvenlik açıklarını belirler. Analiz sonuçları bir rapor olarak sunulur ve bu rapor, konsol ekranında veya ayrı bir rapor dosyasında görüntülenebilir.

4.4.1.6. AndroBugs

AndroBugs da QARK gibi mobil uygulamaların statik kod analizi için kullanılan Python tabanlı bir analiz aracıdır. Bu aracın kullanımı da QARK'a benzer. AndroBugs kullanmak için öncelikle AndroBugs'ın resmî sitesinden veya GitHub adresinden dosyalar cihaza indirilir. Ardından analizi istenen mobil uygulamanın APK dosyası oluşturulur ve dosya adı ile yolu belirlenir. Terminal veya komut istemcisi açılır ve AndroBugs çalıştırılır. AndroBugs'ı terminalde çalıştırmak için örnek kod "python androbugs.py -f C:\Users\username\Downloads\example.apk" şeklindedir. AndroBugs, APK dosyasını analiz ettiğinde potansiyel güvenlik açıklarını tespit eder. Analiz sonuçları bir rapor olarak sunulur ve bu rapor, konsol ekranında veya ayrı bir rapor dosyasında görüntülenebilir.

Sıra Sizde

Sınıfınızda eşit üye sayısına sahip beş grup oluşturunuz. Her bir grubun FindBugs, Fortify Static Code Analyzer, DroidMat, RiskRanker, Woodpecker statik kod analiz yöntemlerinden birini seçerek araştırmasını sağlayınız. Araştırma sonuçlarınızı sınıfta diğer gruplarla paylaşınız.

Okuma Parçası

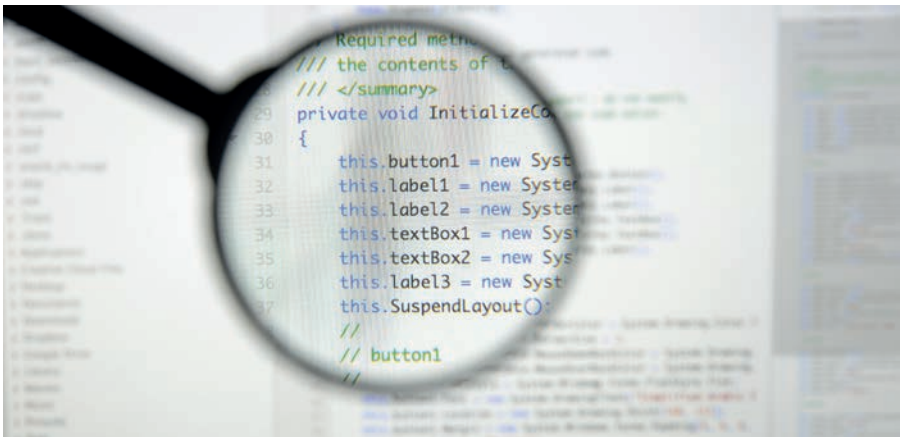
Ülkemizin 2023 yılı vizyonu çerçevesinde belirlenen ULUSAL SİBER GÜVENLİK HEDEFLERİMİZ

- Kritik altyapılarımızın siber güvenliğinin 7/24 korunması.
- Ulusal seviyede siber güvenlik alanında en son teknolojik imkânlara sahip olunması.
- Operasyonel ihtiyaçlar çerçevesinde yerli ve milli teknolojik imkânların geliştirilmesi.
- Siber olaylara müdahalenin olay öncesi, esnası ve sonrasını kapsayan bir bütün olmasından hareketle; proaktif siber savunma anlayışının geliştirilmeye devam edilmesi.
- Siber olaylara müdahale ekiplerinin yetkinlik seviyelerinin ölçülmesi ve izlenmesi.
- Siber olaylara müdahale ekiplerinin yetkinliklerinin artırılması.
- Kurumsal, sektörel ve ulusal bazda siber olaylara hazırlık seviyelerinin risk temelli analizler ve planlamalara dayalı yaklaşımlarla artırılması.
- Kurum ve kuruluşlar arası veri paylaşımının güvenli biçimde sağlanması.
- Kaynağı ve hedefi yurt içi olan veri trafiğinin yurt içinde kalması.
- Kritik altyapı sektörlerinde düzenleme ve denetlemeye dayalı siber güvenlik yaklaşımının geliştirilmesi.
- Kritik altyapı sektörlerinde, BT ürünlerinde üretici bağımlılığının önüne geçilmesi.
- Yeni nesil teknolojilerin güvenliğinin sağlanmasına yönelik gereksinimlerin belirlenmesi.
- Yenilikçi fikirlerin ve Ar-Ge faaliyetlerinin desteklenerek yerli ve milli ürün ve hizmetlere dönüşümünün gerçekleştirilmesi.

https://hgm.uab.gov.tr/uploads/pages/strateji-eylem-planlari/ulusal-siber-guvenlik-stratejisi-ve-eylem-plani-2020-2023.pdf

4.4.2. Dinamik Kod Analizi

Mobil uygulamalarda dinamik kod analizi, uygulamanın çalışma zamanında davranışlarını anlamak ve var olan sorunları tespit etmek üzere yazılım kodlarını gerçek zamanlı olarak izleme sürecine verilen genel bir isimdir (Görsel 4.32). Dinamik kod analizi; kodun ne kadar verimli çalıştığı, hataları, hafıza sorunu veya sızıntıları tespit etmeyi ve CPU işlem şişmelerine neden olan noktaları belirleyebilmeyi hedefler. Android Studio, bir mobil uygulama geliştirme editörü olarak dâhilî şekilde dinamik kod analiz araçları sunar.



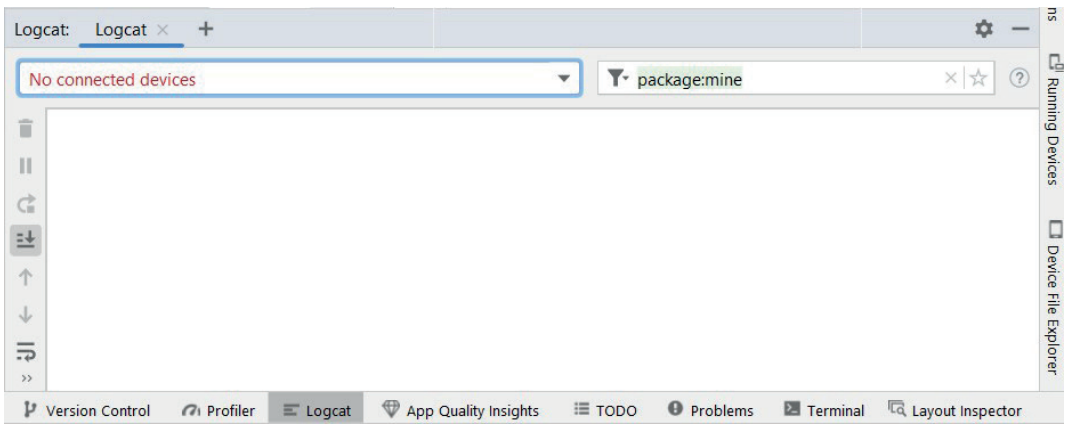
Görsel 4.32: Dinamik kod analizi

Statik kod analizi, olası sorunlarla ilgilenirken dinamik kod analizi, çalışma zamanında ortaya çıkan sorunlarla ilgilenir. Android Studio içinde dinamik kod analizi yapılabilecek araçlar şunlardır:

- **Logcat**

Logcat, Android Studio içinde dâhilî olarak yüklenir. Mobil uygulamanın çalışma zamanında ürettiği mesajları ve log kayıtlarını görüntüler. Belirtilen bu log kayıtları, mobil uygulama içindeki mesaj ve hataları listeler. Bu mesajlar izlenerek ve analiz edilerek mobil uygulamanın gerçek zamanlı çalışması sırasında karşılaşılan hatalar, sızmalar, performans problemleri, farklı davranışları tespit edilebilir.

Logcat ekranı, Android Studio'da View menüsünden Tool Windows ve Logcat seçimi ile açılabilir. Bu işlemin kısayolu için Android Studio'nun Footer bölümündeki araçlardan Logcat seçeneğine tıklanır (Görsel 4.33).



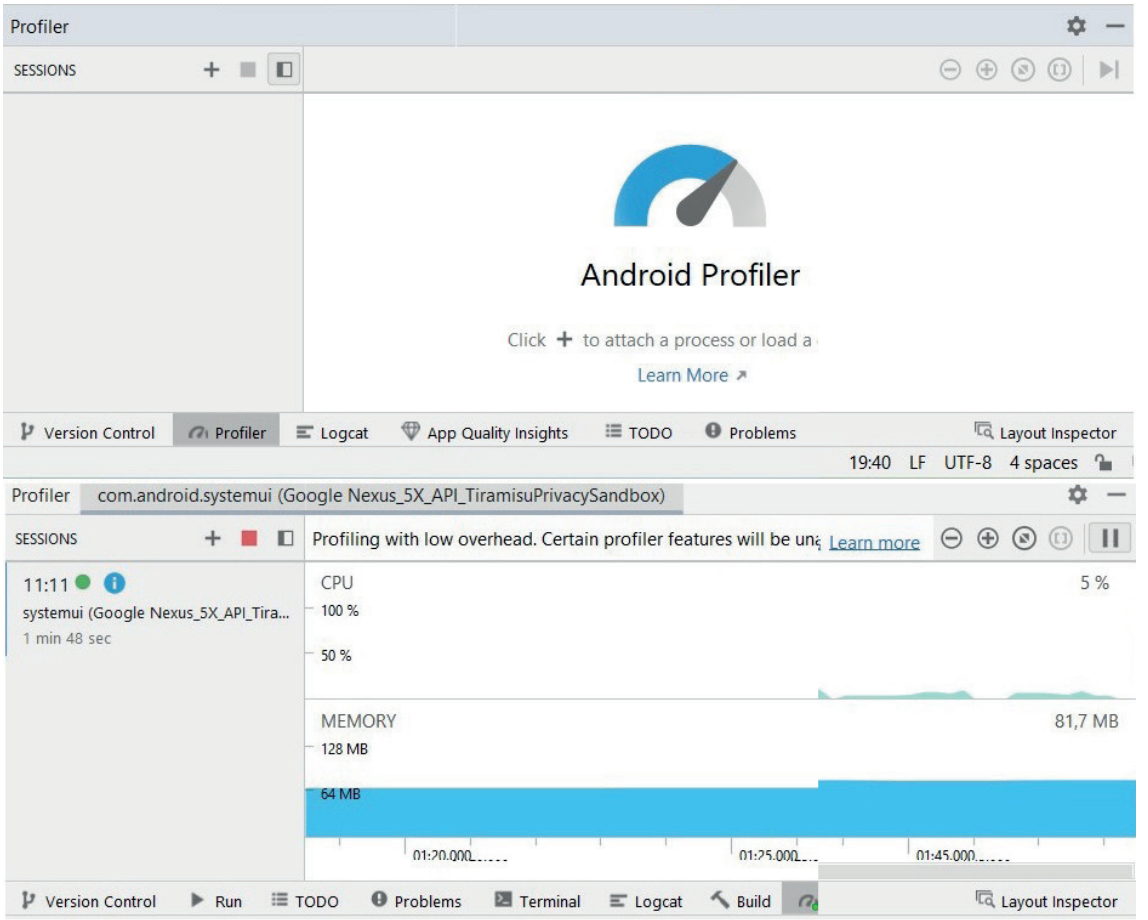
Görsel 4.33: Logcat ekranı

Logcat ekranından filtreleme yapılarak mesajların görülmesi mümkündür. Görsel 4.28'de görülen **package:mine** yazısının olduğu filtreleme kutusuna, görülmek istenen anahtar kelimeler yazılarak sadece o mesajların log kayıtlarının gösterilmesi sağlanabilir.

- **Profile**

Profile, uygulamanın anlık çalışması sırasında gerçekleştirdiği performansı analiz etmek için kullanılan Android Studio'ya ait güçlü bir dinamik kod analizi aracıdır. İşlemci kullanımı, ağ ve veri trafik denetimi, hafıza tüketimi ve her türlü performans olayını analiz ederek geliştiricinin, durumu izlemesini sağlar.

Profile aracı, Android Studio'da View menüsünden Tool Windows ve Profile seçimi ile açılabilir. Bu işlemin kısayolu için Android Studio'nun Footer bölümündeki araçlardan Profile seçeneğine tıklanır (Görsel 4.34).



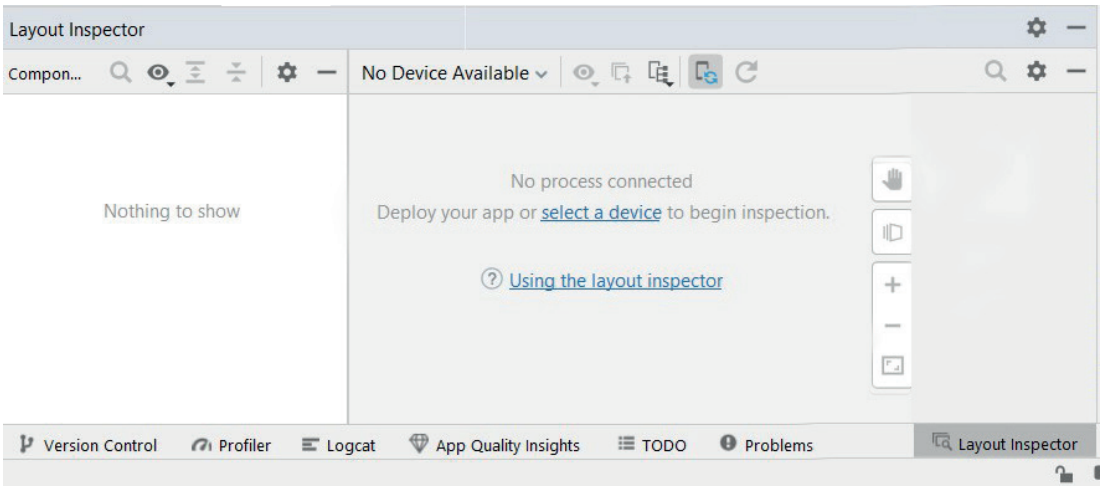
Görsel 4.34: Profiler ekranı

Profile aracı içinde simülatör ile veri analizi yapılmasına gerek yoktur. Android Studio'ya bağlanıp tanıtılmış gerçek bir cihaz üzerindeki performans da bu araç ile izlenebilir. Bunun için yapılması gereken işlem, Click seçeneği ile analizi yapılacak mobil uygulamanın gerçek zamanlı çalışacağı sanal veya gerçek cihazın seçilmesidir.

- **Layout Inspector**

Layout Inspector, mobil uygulamanın kullanıcıların etkileşime girdiği kullanıcı arayüzünü analiz etmeye yardımcı bir dinamik kod analiz aracıdır. Geliştirilen mobil uygulamanın arayüz yapısını görsel olarak görüntüler ve her ögenin (nesnenin) özelliklerinin incelenmesine olanak tanır. Bu sayede doğru düzenlenmemiş veya hatalı yapılandırılmış bileşenler tespit edilebilir. Örneğin ConstraintLayout olarak ayarlanan bir Layout seçeneğinin tüm nesne bağlantı ilişkilendirmelerinin yapılması gerekir. Boşta kalan bir ilişkilendirme, mobil uygulama geliştiricinin karşısına ya bir uyarı ya da bir sorun olarak çıkar. Böyle bir durumun yaşanmaması veya mobil uygulama, kullanıcılarla henüz etkileşime girmeden problemin çözülmesi Layout Inspector ile mümkündür.

Layout Inspector aracı, Android Studio'da View menüsünden Tool Windows ve Layout Inspector seçimi ile açılabilir. Bu işlemin kısayolu için Android Studio'nun Footer bölümünün sağındaki Layout Inspector seçeneğine tıklanır (Görsel 4.35).



Görsel 4.35: Layout Inspector aracı

Okuma Parçası

Air India veri sızıntı (2021): 21 Mayıs 2021'de Air India adlı havayolu şirketinin bir siber saldırıya maruz kaldığı ve dünya çapında yaklaşık 4,5 milyon müşterinin pasaport, kredi kartı bilgileri, doğum tarihleri, isim ve bilet bilgileri dâhil kişisel bilgilerinin ele geçirildiği bildirildi (Satija, 2021).

https://www.tubitak.gov.tr/sites/default/files/25506/siber_guvenlik_ortaokul.pdf

Sıra Sizde

Hesap Makinesi uygulamasının dinamik kod analizini, Logcat, Profile ve Layout Inspector kullanarak gerçekleştiriniz. Bulunan hataları ve sorunları not ediniz.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. Hesap Makinesi isminde yeni bir Android projesi oluşturdu.		
2. Projenin Activity sayfasını oluşturdu.		
3. Activity'e butonları ekledi.		
4. Butonların tanımını yaptı.		
5. Logcat aracından uygulamanın eksikliklerini tespit etti.		
6. Profile aracı yardımı ile uygulamanın kullandığı işlemci ve hafıza durumunu inceledi.		
7. Layout Inspector aracıyla görsel arayüzü inceleyerek kodun eksikliklerini ortaya çıkarttı.		

"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.

Kod editörü programının sunduğu dinamik kod analizi araçlarının dışında da dinamik kod analizini gerçekleştirmek için etkili paket araçlar vardır. Bunlar arasında AndroGuard, CuckooDroid, DroidBox, MobiSec, MobSF, Frida, Drozer gibi araçlar öne çıkar.

4.4.2.1. AndroGuard

AndroGuard, mobil uygulamaların statik ve dinamik kod analizini gerçekleştirmek için kullanılan açık kaynaklı bir analiz aracıdır. Bu araç; mobil uygulamaların APK dosyalarını analiz ederek içerdikleri bileşenleri, izinleri, aktiviteleri ve diğer özellikleri incelemeye olanak tanır. Ayrıca uygulamanın kodunu çıkartabilir, analiz edebilir ve mobil uygulamadan istenen bilgileri çekebilir.

AndroGuard, Python dilinde hazırlanmış bir analiz aracı olduğu için Python yazılımının cihazda yüklü olması gerekir. AndroGuard, Python paket yükleyici olan pip aracılığı ile yüklenir. Bu aracı pip ile cihaza yüklemek için gerekli kod "pip install androguard" şeklindedir. Kurulum sonrası analiz edilmesi istenen mobil uygulamanın APK dosyası çıkartılıp dosya konumu da belirlenmelidir. Terminal veya komut istemcisi açılıp, AndroGuard kullanılarak APK dosyası analiz edilmeye başlanır. Analizi başlatmak için "androguard C:\Users\username\Desktop\myapp.apk" şeklinde örnek bir kod kullanılır.

AndroGuard, APK dosyasını analiz ettikten sonra çeşitli sonuçlar çıkartır. Bu sonuçlar; mobil uygulamanın içerdiği bileşenleri, izinleri, aktiviteleri, manifest dosyası içeriğini ve diğer özellikleri içerebilir. Analiz sonuçları, terminalde veya ayrı bir çıktı dosyasında görüntülenebilir.

4.4.2.2. CuckooDroid

CuckooDroid, mobil uygulamaları analiz etmek ve kötü amaçlı yapıları tespit etmek için kullanılan açık kaynaklı bir projedir ve bir Cuckoo Sandbox eklentisidir. Cuckoo Sandbox ise kötü amaçlı yazılımların analizini gerçekleştirmek için kullanılan bir otomatik tehdit analiz platformudur. CuckooDroid'in amacı, Cuckoo Sandbox platformunun mobil uygulamaları analiz etme yeteneğini genişletmektir.

CuckooDroid'i kullanmak için öncelikle cihazda Cuccoo Sandbox'ın kurulu olması gerekir. Cuccoo Sandbox'ın cihaza kurulması için öncelikle resmî web sitesi ziyaret edilmeli veya GitHub deposu kullanılmalıdır. Cuccoo Sandbox kurulduktan sonra eklentilerden CuccoDroid, platforma eklenmelidir. Ayrıca analizlerin yapılabilmesi için platformda bir sanal makine (simülatör) oluşturulmalıdır. Analizi gerçekleştirilmek istenen mobil uygulama Coccoo Sandbox'a eklenerek CuccoDroid ile analiz başlatılmalıdır. Cuckoo Sandbox, Android uygulamasını çalıştırdıktan sonra analiz sonuçlarını toplar. Bu sonuçlar; uygulamanın davranışını, ağ trafiğini, dosya etkileşimlerini ve diğer aktiviteleri içerir. Sonuçlar incelenerek potansiyel tehditler ve kötü amaçlı faaliyetler belirlenebilir.

4.4.2.3. Frida

Frida; mobil uygulama analizi, güvenlik testleri, tersine mühendislik gibi alanlarda kullanılan oldukça etkili açık kaynaklı bir araçtır. Temel olarak mobil uygulamaların çalışması sırasında (runtime) yazılımları manipüle etmek için kullanılır. Frida'nın temel amacı; mobil uygulama içi davranışları gözlemlemek, değiştirmek veya etkilemek için bir platform sağlamaktır. Bu araç; sızma testleri, güvenlik açığı analizi ve diğer birçok senaryo için kullanılabilir.

Frida, çapraz platform desteği de sunar. Windows, MacOS, Linux gibi çeşitli platformlarda rahatça çalışabilir. Bu sayede farklı ortamlarda mobil uygulama analizi ve manipülasyonlar yapmaya olanak tanır. Frida diğer analiz testlerinden farklı olarak Scripting desteklerine de sahiptir. Bu araç ile Python, JavaScript, Swift, Objective-C gibi diller kullanılarak farklı senaryolar oluşturulabilir.

Frida'nın kullanımı için öncelikle Frida'nın resmî web sitesinden veya GitHub deposundan kurulum dosyası cihaza indirilmeli ve kurulmalıdır. Ardından analiz edilmesi istenen mobil uygulama seçilir ve çalıştırılır. Test ihtiyaçlarına bağlı olarak metot çağrıları değiştirilebilir, bellek manipülasyonları yapılabilir. Analiz ve gerekli manipülasyonların ardından Frida tarafından sağlanan sonuçlar incelenir.

Sıra Sizde

Sınıfınızda eşit üye sayısına sahip iki grup oluşturunuz. Gruplardan birinin MobiSec, diğerinin Drozer dinamik kod analiz yöntemini araştırmasını sağlayınız. Araştırma sonuçlarınıza göre bu yöntemlerin olumlu ve olumsuz yönlerini sınıfta tartışınız.

ÖLÇME VE DEĞERLENDİRME

A. Aşağıdaki parantezlerin içine cümlelerde verilen bilgiler doğru ise "D", yanlış ise "Y" yazınız.

1. ( ) Veri şifrelemesi, kişisel bilgilerinin güvenliğini sağlamak için yaygın olarak kullanılan yöntemdir.

2. ( ) Mobil uygulamalar, cihazlardaki hassas verilere erişmek için izin aldıktan sonra bu verilere sınırlama getirmek için ek güvenlik önlemleri almaz.

3. ( ) Mobil uygulama geliştiricileri, güvenlik açıklarını tespit etmek ve düzeltmek için periyodik güvenlik testleri yapmamalıdır.

4. ( ) Kullanıcıların, cihazlarındaki uygulamalara veri sağlamadan önce izin yetkilerini dikkatlice gözden geçirmeleri ve gereksiz izinleri reddetmeleri gerekir.

5. ( ) Mobil uygulamalar, kullanıcıların kişisel verilerini depolamak ve işlemek için gizlilik odaklı yaklaşımlar kullanmaz.

B. Aşağıdaki cümlelerde boş bırakılan yerlere doğru sözcükleri yazınız.

6. Kullanıcıların güvenliği sağlamak için mobil uygulama geliştiricileri genellikle veri ..... kullanırlar.

7. Güvenli bir mobil uygulama geliştirmek için geliştiriciler, kullanıcıların ..... kullanmasını sağlamalıdır.

8. Mobil uygulama güvenliğini artırmak için kullanıcıların oturum açma işlemleri sırasında ..... doğrulama yöntemleri kullanılabilir.

9. Mobil uygulama geliştiricileri, kullanıcıların cihazlarında yetkilendirilmemiş erişimlerin önlenmesi için genellikle ..... yöntemlerini kullanırlar.

C. Aşağıdaki soruları dikkatlice okuyarak doğru cevabı işaretleyiniz.

10. Aşağıdakilerden hangisi mobil uygulama güvenliğinin amaçlarından biridir?

A) Kullanıcıların cihazlarına ücretsiz uygulamalar indirilmesini sağlamak.
B) Reklam engellemelerini sağlamak.
C) Antivirüs gibi uygulamalara gereksiz para harcanmasını önlemek.
D) Kullanıcıların kişisel verilerinin korunmasına yardımcı olmak.
E) Geliştirme sürecini uzatmak.

11. Aşağıdakilerden hangisi mobil uygulamaların güvenliği için önemli bir adımdır?

A) Otomatik testlerin daha az önemsenmesi.
B) Lisanssız uygulama sürümlerini de kullanmak.
C) Uygulamalarda içeriği belirsiz bağlantılara da izin verilmesi.
D) Sadece resmî uygulama mağazalarından indirmeler yapılması.
E) Bol görsel içeriğinin olması.

12. Aşağıdakilerden hangisi mobil uygulama güvenliği açısından zayıf bir uygulamadır?

A) Sürekli güncellemeler alması.
B) Kullanıcıların kişisel verilerini şifrelemeden depolaması.
C) Sık sık güvenlik testlerinden geçmesi.
D) Birden çok uygulama mağazasından indirilebilir olması.
E) Çift doğrulamalı şifre metodu kullanması.

13. Aşağıdakilerden hangisi mobil uygulama güvenliği için kullanılan güvenlik önlemi türlerindendir?

A) Yazılım güncelleştirmeleri ve kullanıcı kimlik doğrulaması.
B) Veri depolama yapması ve oturum yönetimi sunması.
C) Uygulama içi satın alma ve reklamların kullanılması.
D) Ücretsiz oyun sunması.
E) Verilerin Activity ekranlarında gösterilmesi.

14. Aşağıdakilerden hangisi mobil uygulama geliştirmek için kullanılan yazılım dili ve editörlerinden biridir?

A) Python ve PyCharm
B) Java ve Android Studio
C) C# ve Visual Studio
D) HTML ve Sublime Text
E) SQL ve SQL Server Management Studio

15. Aşağıdakilerden hangisi güvenlik açıklarını tespit etmek için mobil uygulama geliştirme kod editörlerinin sağladığı araçlardır?

A) Özgür yazılım kütüphaneleri
B) Veri tabanı yöneticileri
C) Uygulama mağazaları
D) Simülatör destekleri
E) Hata ayıklayıcı ve statik kod analiz araçları

16. Aşağıdakilerden hangisi mobil uygulamalarda kod incelemesi ve denetiminin bir sonucudur?

A) Geliştirme sürecini uzatır.
B) İşletme maliyetini artırır.
C) Uygulamanın güvenlik açıklarını kapatır.
D) Uygulamanın kullanımını engeller.
E) Uygulama mağazalarını hoş tutar.

17. Aşağıdakilerden hangisi kötü niyetli bir uygulamanın cihaza bulaşmasına yol açar?

A) Uygulama mağazası dışında bir kaynaktan indirilmiş olabilir.
B) Sadece resmî uygulama mağazasında yayımlanır.
C) Yalnızca IOS işletim sisteminde çalışacak şekilde programlanmıştır.
D) Uygulamayı kullanan kişilerin kimliklerine bağlıdır.
E) Mobil cihazın üretiminden kaynaklıdır.

18. Aşağıdakilerden hangisinde mobil uygulamalarda kullanıcı kimlik doğrulamasının önemi belirtilmiştir?

A) Uygulamanın daha hızlı çalışmasını sağlar.
B) Kullanıcı verilerini şifreler.
C) Yetkisiz erişim ve hesap sahtekârlığının önüne geçer.
D) Kişiye özel reklam gösterme olanağı sağlar.
E) Uygulama sahiplerine kullanıcılarını takip imkânı sunar.

19. Aşağıdakilerden hangisi mobil uygulama güvenliği için SSL sertifikası kullanmanın önemini açıklar?

A) Uygulamanın daha hızlı çalışmasına olanak tanır.
B) Kötü amaçlı yazılımları engeller.
C) Uygulama sahiplerine daha fazla maddi kazanç sağlar.
D) Verilerin şifrelenmesini sağlar ve bağlantıyı güvenli hâle getirir.
E) Uygulamanın sadece belirli ülkelerde kullanılmasına olanak tanır.

20. Aşağıdakilerden hangisi mobil uygulama güvenliği açısından riskli bir davranıştır?

A) Uygulamayı orijinal mağazadan indirmek.
B) Ücretsiz Wi-Fi ağlarına bağlanmak.
C) Oturum açarken güçlü parola kullanmak.
D) Uygulama içi satın alımları aktif hâle getirmek.
E) E-posta veya telefon ile iki faktörlü doğrulama girişi kullanmak.

21. Aşağıdaki tekniklerden hangisi mobil uygulamaların kötü niyetli yazılımlarını analiz etmek için kullanılabilir?

A) Dizin saldırıları
B) SQL enjeksiyonu
C) Tersine mühendislik
D) Şifreleme ve algoritmalarını kırmak
E) Checkstyle aracı kullanmak

D. Aşağıdaki soruların cevabını ilgili alana yazınız.

22. Mobil uygulama geliştiricilerinin, kullanıcıların verilerini korumak için kullandığı yöntemlerin uygulanmasında karşılaştığı zorluklar nelerdir?

23. Mobil uygulama güvenliğinde kullanılan şifreleme teknikleri, kullanıcıların veri bütünlüğünü nasıl etkiler?

24. Mobil uygulamaların güvenlik açıkları, saldırganların hangi tür siber saldırıları gerçekleştirmelerine olanak tanır.

WEB UYGULAMA GÜVENLİĞİ

5. ÖĞRENME BİRİMİ

KONULAR

5.1. WEB UYGULAMA KAVRAMLARI

5.2. WEB UYGULAMA KOD EDİTÖRLERİ

5.3. WEB UYGULAMALARI

5.4. WEB UYGULAMA GÜVENLİĞİ

NELER ÖĞRENECEKSİNİZ?

- Web uygulama temel kavramlarını açıklama
- .Net Core web editörü arayüzünü açıklama
- PHP web editörü arayüzünü açıklama
- Web servis güvenliği protokollerini yapma
- .Net Core tabanlı web uygulaması yapma
- PHP tabanlı web uygulaması yapma
- İstemci taraflı girdi denetimini yapma
- Sunucu taraflı girdi denetimini yapma
- Pozitif girdi denetimini açıklama
- Çıktı denetimini açıklama
- XSS denetimini açıklama
- Kör SQLi (Blind SQLi) saldırısını açıklama
- Oturum yönetimi temel ilkelerini hazırlama

ANAHTAR KELİMELER

Arama motoru, bağlantı, dinamik sayfa, erişim, güvenlik, internet adresi, işletim sistemi, statik sayfa, sunucu, tarayıcı, üye, veri tabanı, yetkilendirme, yönetim, web sitesi, web uygulaması

HAZIRLIK ÇALIŞMALARI

1. Tarayıcı çeşitlerinden bildiklerinizi sınıfta arkadaşlarınızla paylaşınız.
2. Dolandırıcılık için hazırlanan sahte site ile gerçek site arasındaki farklar neler olabilir? Düşüncelerinizi arkadaşlarınızla tartışınız.

5.1. WEB UYGULAMA KAVRAMLARI

Web sitesi ve web uygulamaları ile ilgili kavramlar birbirine benzetilebilir. Her iki kavrama da **Tarayıcı (Browser)** adı verilen programlarla ulaşılır ve sonuç görüntülenir. Tarayıcı uygulaması genellikle akıllı cihazlarda kurulu olarak gelir. Telefon, televizyon, bilgisayar, tablet gibi internete bağlanabilen akıllı cihazlarda bir internet adresine girildiğinde site veya uygulama aynı şekilde ekranda görüntülenir.

Bir web uygulaması geliştirecekse platformdan (işletim sisteminden) bağımsız olarak proje geliştirme imkânı bulunur. Bu işlem, tüm donanım imkânlarına tam performans ile ulaşılabildiği anlamına gelmez. Örneğin tarayıcının WebGL (Web Graphics Library) özelliği ile üç boyutlu nesneler oluşturabilir ancak ekran kartının tüm özelliklerinin tarayıcıda çalışması zordur.

Not WebGL, tarayıcının grafik işlem birimi (Graphics Processing Unit-GPU) üzerinde donanım hızlandırma kullanarak 3D grafikler oluşturmayı sağlar.

Araştırma

İşletim sistemlerinin güncel kullanım oranlarını araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

Görsel 5.1: Siber saldırılar ve güvenlik

Web sitesi ile web uygulaması, HTML altyapısı olarak aynı gibi görünse de birbirinden farklıdır. Web sitesi, tek sayfa hâlinde de olabilir veya birçok web sayfası oluşturularak bağlantılar (linkler) ile web sayfaları birbirine bağlanabilir. Web sitesi ile genelde sabit içerik paylaşımı yapılırken web uygulamaları daha fazla kullanıcı etkileşimine sahiptir. Web uygulamalarındaki kullanıcı etkileşimi içinde beğenmek, yorum yazmak, puan vermek, üye olmak gibi işlemler sayılabilir.

Web sitesinde sadece yetkili kullanıcı değişiklik yapabildiği için **sabit (statik)** bir HTML sayfası daha güvenlidir ancak uzaktan erişim ile sisteme girmeyi başaran kötü niyetli biri veya bot adı verilen programlar, web sitesine zarar verebilir (Görsel 5.1).

Statik bir sayfayı güncelleyebilmek için HTML, CSS ve JavaScript kodlarını bilmek gereklidir. Günümüzde statik sayfa yöntemi ile site yapımı, web sitesi geliştirmede çok az kullanılır.

Dinamik sayfa adı verilen uygulamalarda ise web sitesinin içeriği, yönetim panelindeki (control panel) sayfalar kullanılarak hazırlanır. Yetkilendirme ve belli sayfaların kullanıcı gruplarına göre davranması sayesinde web sitesinin güvenliği artırılır. Burada önemli olan hususlar, kodda güvenlik açıklarının kalmaması ve güncel bir altyapının bulunmasıdır.

Web sitesinin yönetim ve içeriği, veri tabanı altyapısı ile çalışır. Bu altyapıda **SQL (Structured Query Language-Yapılandırılmış Sorgu Dili)** programlama dili kullanılır. Uygulamaya, bir başka deyişle proje içine veri tabanı desteği eklendiğinde veri tabanının açıklarıyla da ilgilenilmelidir. Sisteme üye olan kişilerin bilgilerini korumak, web sitesi yöneticisi (administrator, root, superuser) için en önemli görevlerin başında gelir. Verilerin yedeklenmesi, verilerdeki bozulmaların düzeltilmesi de web sitesi yöneticisinin diğer görevlerindendir (Görsel 5.2).

Görsel 5.2: Dizüstü bilgisayar ve uzak sunucu bağlantısı

Kullanıcılar tarafından hem statik hem de dinamik sayfalarda kaynak görülebildiği için web sitelerinden resim, JavaScript vb. dosyalar kolaylıkla indirilir. Bir başka deyişle çok bilgili olmayan herhangi biri, sayfaya sağ tıklayıp kaynağı görebilir hatta o anda sayfada değişiklikler yapabilir. Sayfa yeniden yüklendiğinde web sitesi, eski bilgileri tekrar ekrana getirir. Web sitelerinin kopyası oluşturularak sahte web siteleriyle insanlardan kişisel bilgiler toplanabilir. **E-dolandırıcılık (Phishing)** adı verilen yöntem ile kredi kartı ve diğer kişisel bilgiler elde edilebilir.

URL (Uniform Resource Locator) adı verilen web sitesi adreslerinin akılda tutulması ve adres çubuğuna sürekli el ile girilmesi zor olacağı için arama motorları geliştirilmiştir. Arama motorları, algoritmalar ile arama sonuçlarını liste hâlinde verir. Arama sonucunda üst sıralarda çıkan web sayfaları her ne kadar en doğru sonuçlar gibi görünse de gezilen web sitesinde geçersiz bilgiler olabilir. Arama sonuçları birden fazla web sitesinden karşılaştırılarak doğru sonuca ulaşılabilir. Bazı arama motorlarında sonuçlar, büyük veri içinden yapay zekâ desteği ile değerlendirilerek kullanıcıya sunulur. Arama motoruna yazılan doğru istem (prompt) ile başarılı sonuçlar döndürülür.

HTML, CSS ve tüm kaynaklar (video, resim, müzik, dosya), bir web sunucusunda (server) saklanır. Kullanıcılar, sayfanın HTML hâlini görebilirken programlama dilleri ile yazılan kodları göremezler. Programlama dillerine örnek olarak ASP.NET, PHP ve Java sayılabilir (Görsel 5.3).

Görsel 5.3: Programlama dilleri ile çalışan akıllı cihazlar

Araştırma

Açık kaynak Java ve JavaScript dillerindeki farkları araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

Web sitesi geliştirme dünyasında çalışma grupları şu şekilde görevlendirilir:

- **Frontend:** Bir web uygulamasının kullanıcı tarafında çalışan kısmını ifade eder. Kullanıcı arayüzünü oluşturan HTML, CSS, JavaScript gibi teknolojiler kullanılarak web sayfalarının ve kullanıcı deneyiminin geliştirilmesi sağlanır. Frontend geliştiriciler, kullanıcıların etkileşimde bulunabileceği ve görsel olarak çekici web sayfaları oluştururlar.
- **Backend:** Bir web uygulamasının sunucu tarafında çalışan kısmını ifade eder. Veri tabanı yönetimi, sunucu taraflı programlama ve iş mantığının yönetimi ile ilgili görevler, backend geliştiriciler tarafından gerçekleştirilir. Backend; kullanıcıların verilerini işleyen, sunucu taraflı işlemleri yöneten ve veri tabanıyla etkileşime geçen bir yapıdır.
- **Full Stack:** Hem frontend hem de backend geliştirme alanlarında bilgi ve deneyime sahip olan geliştiricileri ifade eder. Full stack geliştiriciler, kullanıcı arayüzünü tasarlayabilir ve sunucu tarafındaki işlemleri yönetebilirler. Genellikle tüm projenin farklı katmanlarında çalışabilme yeteneğine sahiptirler.

Web sitesi geliştirme dünyasında bu görevlerin dışında veri tabanı yöneticisi, sistem yöneticisi, veri analisti, UI (kullanıcı arayüzü) ve UX (kullanıcı deneyimi) tasarımcısı, bilgi güvenliği uzmanı, proje yöneticisi, test mühendisi gibi roller de vardır.

Görsel 5.4'te yazılım geliştirme sürecindeki aşamalar verilmiştir.

Görsel 5.4: Yazılım geliştirme aşamaları

Web programlama dilleri ile proje hazırlanırken şu teknolojiler bilinmelidir:

- **HTML (HyperText Markup Language):** Web sayfalarının görüntülenmesi için kullanılan altyapı dilidir.
- **CSS (Cascading Style Sheets):** HTML ögelerinin stillerinin ve biçimlerinin düzenlenmesi için kullanılan bir dildir.
- **JavaScript:** Yerelde çalışan ve etkileşim desteği sağlayan açık kaynak dildir.
- **Bootstrap:** Web sayfaları için tutarlı bir görünüm sağlayan açık kaynak CSS kütüphanesidir.
- **jQuery:** HTML ögelerini ve olaylarını kontrol etmek için işlevsellik sağlayan açık kaynak bir kütüphanedir.
- **AJAX (Asynchronous JavaScript and XML):** Web sayfalarının arka planda sunucuyla veri alışverişi yapabilmesini sağlayan teknolojidir. Web sayfasını yenilemeden veri alışverişi yapılmasını sağlar ve kullanıcı deneyimini geliştirir.
- **MVC (Model View Controller):** İş mantığı katmanı (model), sunum katmanı (view) ve veri erişim katmanı (controller) olarak ayrılan yazılım modelidir.
- **C#:** Açık kaynağı destekleyen genel amaçlı, nesne yönelimli programlama dilidir.
- **LINQ (Language Integrated Query):** Veri tabanları, XML dosyaları, koleksiyonlar gibi farklı kaynaklardan veri sorgulamak için kullanılan bir teknolojidir.
- **SQL (Structured Query Language):** Veri tabanlarıyla etkileşim kurmak için kullanılan bir dildir.

Not ASP.NET ve PHP'nin dışında JavaScript dilini temel alan Node.js, Angular, React, Vue.js, Svelte açık kaynak teknolojileri de vardır.

Tablo 5.1'de web sitesi geliştirmek için gerekli yazılımlar ve bu yazılımların açıklamaları verilmiştir.

Tablo 5.1: Web Sitesi Geliştirmek İçin Gerekli Yazılımlar ve Bu Yazılımların Açıklamaları

Yazılım	Açıklamalar
After Effects	Hareketli grafikler ve görsel efektler oluşturma uygulamasıdır.
Animate	2D animasyon ve interaktif içerikler hazırlama uygulamasıdır.
Dreamweaver	Web tasarımı ve kodlama uygulamasıdır.
Photoshop	Gelişmiş fotoğraf düzenleme ve grafik tasarım uygulamasıdır.
Premiere Pro	Profesyonel video düzenleme ve kurgu hazırlama uygulamasıdır.
Visual Studio	Programlama dili çalıştırma ve geliştirme ortamı uygulamasıdır.
Visual Studio Code	Birçok dili destekleyen kod düzenleyici uygulamasıdır.
Git	Versiyon kontrol sistemi uygulamasıdır.
Chrome	Web tarayıcısı uygulamasıdır.

Web uygulama güvenliği ile ilgili anahtar kavramlar şunlardır:

- **Güvenlik Duvarı (Firewall):** Bilgisayar ağını dış tehditlere karşı korumak için kullanılan güvenlik önlemidir.
- **Güncelleme Yönetimi:** Güvenlik açıklarını gidermek için web uygulamalarının düzenli olarak güncellenmesi ve yamalanmasıdır.
- **Oturum Yönetimi:** Kullanıcı oturumlarını takip etmek, kimlik doğrulama bilgilerini saklamak ve güvenli bir şekilde iletmek için kullanılan yöntemlerdir.
- **Güvenlik Denetimleri:** Web uygulamalarının güvenlik açıklarının tespit edilmesi ve düzeltilmesi için düzenli olarak yapılan kontrollerdir.
- **Kimlik Doğrulama ve Yetkilendirme:** Web uygulamalarında kullanıcı kimliklerini doğrulama ve erişim haklarını belirleme sürecidir.
- **Şifreleme (Encryption):** Verilerin güvenli bir şekilde iletilmesi veya depolanması için kullanılan bir yöntemdir.
- **XSS (Cross-Site Scripting):** Web uygulamalarında bulunan bir zayıflık türüdür. Saldırganın, kötü amaçlı kodu kullanıcının tarayıcısına enjekte etmesi işlemidir.
- **CSRF (Cross-Site Request Forgery):** Saldırganın, kullanıcının tarayıcısında yetkilendirme bilgilerini kullanarak izinsiz işlemler gerçekleştirmesine olanak tanıyan bir saldırı türüdür.
- **SQL Injection (Aşılama, Enjeksiyon):** Web uygulamalarındaki güvenlik açıklarından yararlanarak saldırganın SQL sorgularını manipüle etmesine izin veren bir saldırı türüdür.
- **DoS (Denial of Service):** Bir web uygulamasına aşırı yük bindirilerek servis dışı kalmasına neden olan saldırı türüdür. Saldırı kaynağının tespit edilmesi zordur.

Web uygulamasında güvenlik önlemleri; sunucu, kod ve veri tabanı için ayrı ayrı alınmalıdır. Yazılımdaki açıklar nedeniyle web uygulamasının güvenirliği azalır. Web uygulama geliştirme zincirindeki bir zayıf halka nedeniyle devlet, firma ve kişiler zarar görebilir. Aslında web uygulamasında zincirin en zayıf halkası insandır. Sosyal mühendislik adı verilen yöntem ile insanlara sahte arama, mesaj ve ileti gönderilir. "Ödül kazandınız." şeklinde yönlendirmeler ile ağa düşürülen insanlardan çeşitli kişisel bilgiler ve maddi kazançlar elde edilir (Görsel 5.5).

Görsel 5.5: Web uygulama güvenliği

Araştırma

Yakınlarınız dışında kimsenin bilmemesi gereken kişisel bilgileriniz nelerdir? Paylaşılması sakıncalı olabilecek kişisel bilgileri broşür hâline getirerek sınıfta arkadaşlarınızla paylaşınız.

5.2. WEB UYGULAMA KOD EDİTÖRLERİ

Web uygulaması hazırlanırken açık kaynak Visual Studio Code, açık kaynağı destekleyen Visual Studio Community ve açık kaynak Notepad++ kullanılır.

5.2.1. .NET Core Web Editörü

Web editör programı, eklentiler (extensions) sayesinde birçok dilde kod yazımını destekler ve projenin test edilmesini sağlar. Programın arayüzünde tema ve simge seçenekleri ile değişiklikler yapılabilir. Web editörü olarak Community sürümü tercih edilirse gereken tüm paketler (bileşenler) kurulu olarak gelir.

Not .NET Core açık kaynak olarak 2016 yılında yayımlandı. .NET 5 ve sonraki sürümleri .NET Core ve .NET Framework'ü birleştirerek tek bir platform olan .NET'i oluşturmuştur.

Araştırma

HTML sayfa düzenleme programlarını araştırarak bu programları, üstün ve zayıf yönleri açısından birbiriyle karşılaştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

1. Uygulama

İşlem adımlarına göre komut istemi aracılığıyla web projesi yapımını gerçekleştiriniz.

1. Adım: Kod editör programını çalıştırınız.

2. Adım: View menüsünden Terminal panelini açınız.

3. Adım: **dotnet new mvc -o coreprojem** komutunu yazınız. Bu komut sonucunda coreprojem klasörü içine gerekli birçok klasör ve dosya oluşur.

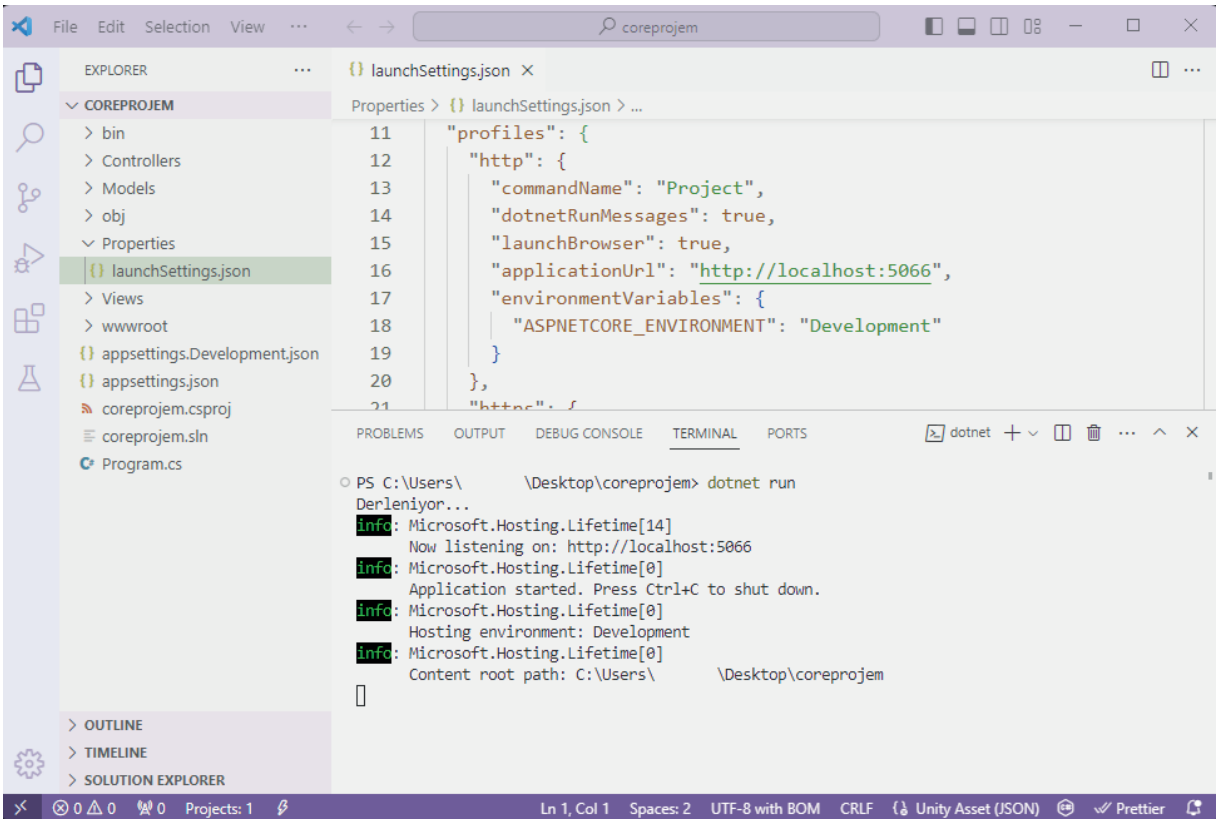
4. Adım: File menüsündeki Open Folder… komutu ile proje klasörünü açınız. Diğer bir yol ise komut isteminde **code -r coreprojem** komutunu yazmaktır.

5. Adım: Projenin çalışmasını test etmeden önce HTTPS (Hyper Text Transfer Protocol Secure) sertifikasına güven verilmesi gerekir. **dotnet dev-certs https --trust** komutu ile gelen iletişim kutusuna onay veriniz.

6. Adım: **dotnet run** komutu ile projeyi derleyiniz ve uygulamanın arka planda çalıştığını gözlemleyiniz.

7. Adım: Tarayıcının adres çubuğuna **http://localhost:5066** yazarak uygulamayı test ediniz. Port numarası olan 5066 rakamı her uygulamada değişir.

Görsel 5.6'da web uygulamasının test edilmesi ve **launchSettings.json** dosyası ile başlangıç ayarlarının yapılması verilmiştir.



Görsel 5.6: Web uygulamasının test edilmesi

5.2.2. PHP Web Editörü

Açık kaynak bir dil olan web programlama dili için kod tamamlama ve renklendirme özellikleri bulunan bir düzenleyici kurmak yeterlidir. Düzenleme için Visual Studio Code veya Notepad++ kullanılabilir. VS Code programında web programlama dilinin eklentileri kurulduğunda daha etkin bir şekilde kod yazılır.

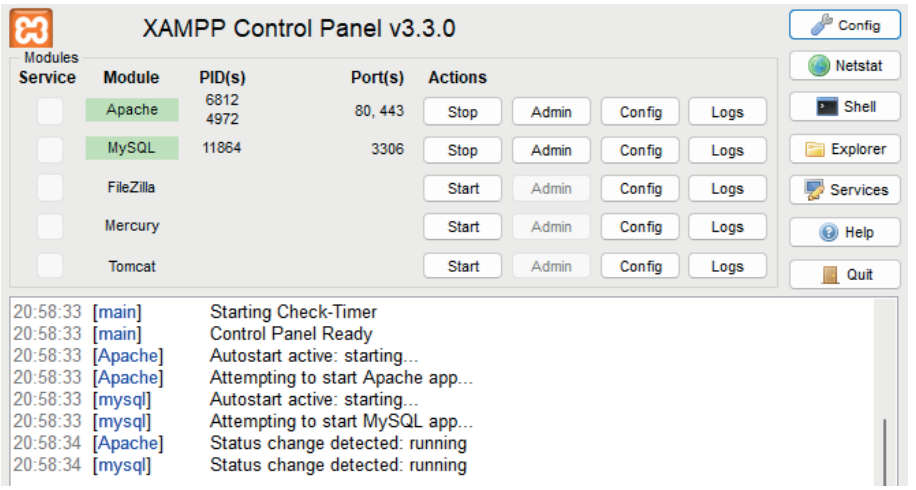
Web programlama dili ile beraber kurulması gereken programlar ayrı ayrı kurulabilir. Web sunucusu olarak açık kaynaklı Apache, veri tabanı sunucusu olarak açık kaynaklı MySQL indirilerek kurulabilir.

2. Uygulama

İşlem adımlarına göre XAMPP programının kurulumunu yapınız.

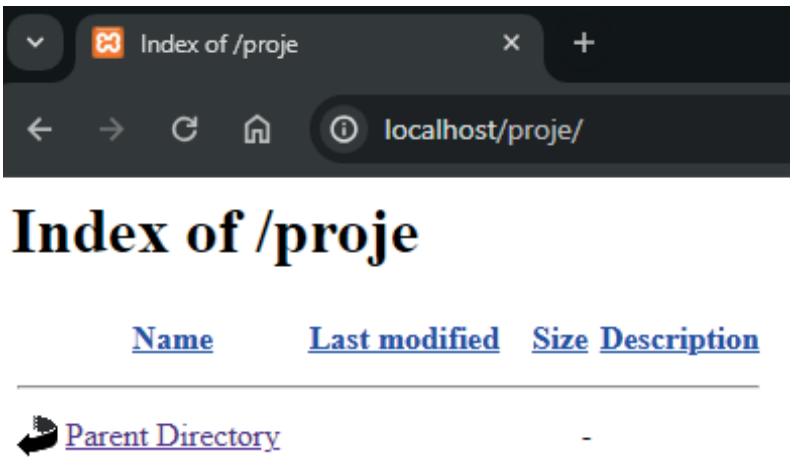
1. Adım: Açık kaynak projeleri destekleyen XAMPP programını kurunuz.

2. Adım: Görsel 5.7'de görüldüğü gibi kontrol panelini çalıştırdığınızda Apache ve MySQL görevlerini başlatınız.



Görsel 5.7: Web sunucusu ve veri tabanı hizmetlerinin başlatılması

3. Adım: Görsel 5.8'de görüldüğü gibi sayfanın çalışması için **C:\xampp\htdocs** klasörü içine proje adında bir klasör açınız ve tarayıcının adresine **localhost/proje** yazarak sunucunun çalışıp çalışmadığını test ediniz.



Görsel 5.8: Projenin test edilmesi

Not **C:\xampp\htdocs** yayımlanan sitenin **kök** dizinidir. Proje dosyaları kök dizinde olmalıdır. Proje dosya türü olarak **.php, .html** şeklinde düzenleme yapılabilir.

80 portu, varsayılan internet web servisi sunucusunun kapısı olduğu için site adresinde **localhost:80** şeklinde yazmaya gerek yoktur. 443 portu, **HTTPS (Hyper Text Transfer Protocol Secure)** güvenli internet desteği için kullanılır. Bilgisayarda kurulu ve çalışan başka bir program varsa, bu portları kullanıyorsa çakışma durumu ayarlardan düzenlenir. Değiştirilen servis ayarlarının geçerli olması için servisin durdurulup tekrar çalıştırılması gerekir.

Tablo 5.2'de önemli ayar dosyalarının listesi verilmiştir.

Tablo 5.2: Program Ayarları İçin Gereken Dosyalar

Program Adı	Program İşlevi	Ayar Dosyası
Apache	Web sunucusu	C:\xampp\apache\conf\httpd.conf
MySQL	Veri tabanı sunucusu	C:\xampp\mysql\bin\my.ini
PHP	Web programlama dili	C:\xampp\php\php.ini

Web sitesi adresinde **localhost** yazmak yerine şunlar da kullanılabilir:

- **127.0.0.1:** Çalışılan bilgisayarın adresi (loopback address) yazılabilir.
- **192.168.1.128:** Başlat menüsü>cmd>ipconfig ile yerel ağ adresi bulunur.
- **bilgisayaradi:** Başlat menüsü>Ayarlar>Sistem>Sistem bilgisi ile cihaz adı bulunur.

4. Adım: Web programlama dilini test etmek için aşağıdaki kod örneğini yazarak sayfada gösteriniz.

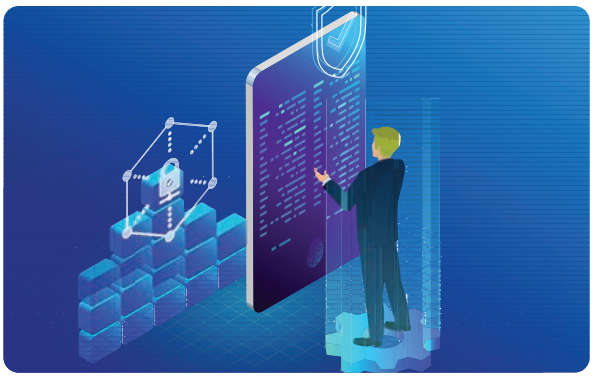
```
<?php
        echo "Merhaba Dünya";
?>
```

5.3. WEB UYGULAMALARI

Web uygulamaları; programlama dilleri kullanılarak verilerin depolandığı, verilerin işlendiği ve internet üzerinden istenen görevlerin yerine getirildiği etkileşimli programlardır.

5.3.1. Web Servis Güvenliği Protokolleri

Web programlama dili Core Web **API (Application Programming Interface)** ile diğer uygulamalara destek amaçlı servis yapılabilir. Örneğin bir ildeki hava durumunu ekrana getirmek, sosyal medyadaki kullanıcı sayısını almak gibi belli veriler, web uygulamasından servis ile çekilebilir. Bu servisi kullanacak uygulamalar; masaüstü, mobil veya başka bir web uygulaması olabilir (Görsel 5.9).



Görsel 5.9: Web servis güvenliği

Web servisleri güvenliği ile ilgili temel kavramlar şunlardır:

- **SOAP (Simple Object Access Protocol):** İki uygulama arasında XML tabanlı mesaj alışverişi için kullanılan bir protokoldür. Basit Nesne Erişim Protokolü olarak adlandırılan bu protokol, web servisleri için yaygın olarak kullanılır.
- **SAML (Security Assertion Markup Language):** Taraflar arasında güvenlik bildirimlerinin değiş tokuşu için kullanılan XML tabanlı bir standarttır. SAML onayları; kimlik doğrulama bilgilerini, yetkilendirme bilgilerini veya her ikisini birden iletmek için kullanılabilir.
- **WS-I (Web Services Interoperability Organization):** Web servislerinin birlikte çalışabilmesi için şartnameler geliştiren, kâr amacı gütmeyen bir kuruluştur.
- **REST (Representational State Transfer):** İstemci ve sunucu arasında hızlı bir şekilde iletişim sağlayan servis türüdür.
- **WebSocket:** Gerçek zamanlı ve çift yönlü iletişimi destekleyen bir protokoldür. Sunucuyla sürekli bağlantı kurar ve verilerin anlık olarak aktarılmasını sağlar. Sohbet, oyun, gerçek zamanlı bilgi akışı için kullanılabilir.

Araştırma

Statefull (durum bilgisi olan) ve stateless (durum bilgisi olmayan) API kullanım farklarını araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

Web uygulamalarında işlemler, tarayıcılar tarafından desteklenen şu yöntemlerle işlemler yapılır:

- **POST:** Oluşturmak (Create), sunucuya veri yollamaktır.
- **GET:** Okumak (Read), sunucudan veri almaktır.
- **PUT:** Güncellemek (Update), sunucudaki belli bir kaynağı güncellemektir.
- **PATCH:** Değiştirmek (Modify), sunucudaki belli bir kaynağı kısmi olarak güncellemektir.
- **DELETE:** Silmek (Delete), sunucudaki bir kaynağı silmektir.

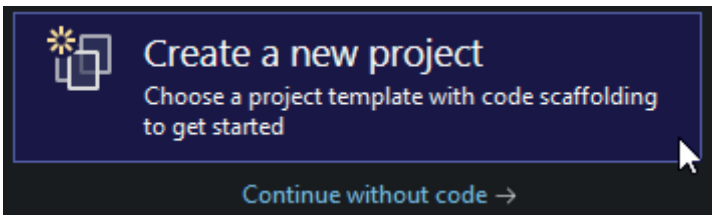
Web uygulamasında oluşabilecek durum kodları şu şekilde gruplandırılabilir:

- **1xx:** Akış ile ilgili bilgi kodlarıdır.
- **2xx:** İsteğin başarılı olduğu belirtilir. 200 OK, işlem tamam demektir.
- **3xx:** İsteğin tamamlanması için ek bilgi gereklidir.
- **4xx:** İstemciden dolayı oluşan hatalardır. 404 Not Found, sayfa bulunamadı demektir.
- **5xx:** Sunucudan kaynaklanan hatalardır. 503 Service Unavailable, servis kullanım dışı demektir.

3. Uygulama

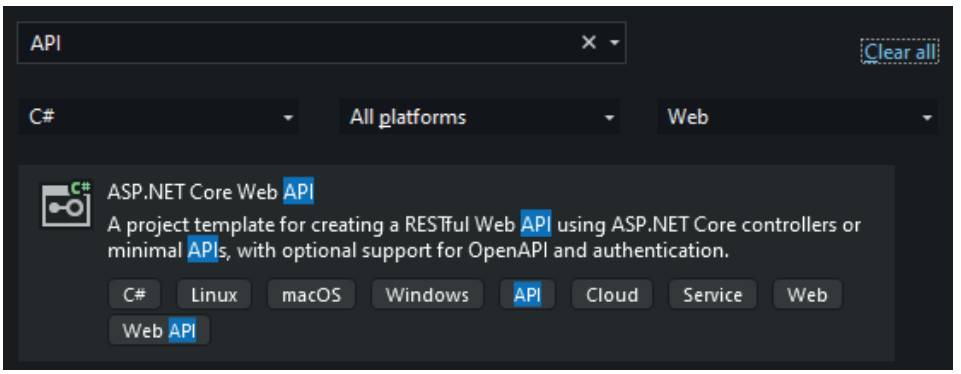
İşlem adımlarına göre web servisi yapınız ve tablolama programı ile web servisinin kullanımını gerçekleştiriniz.

1. Adım: Görsel 5.10'daki gibi web uygulama editörü ile yeni proje açınız.



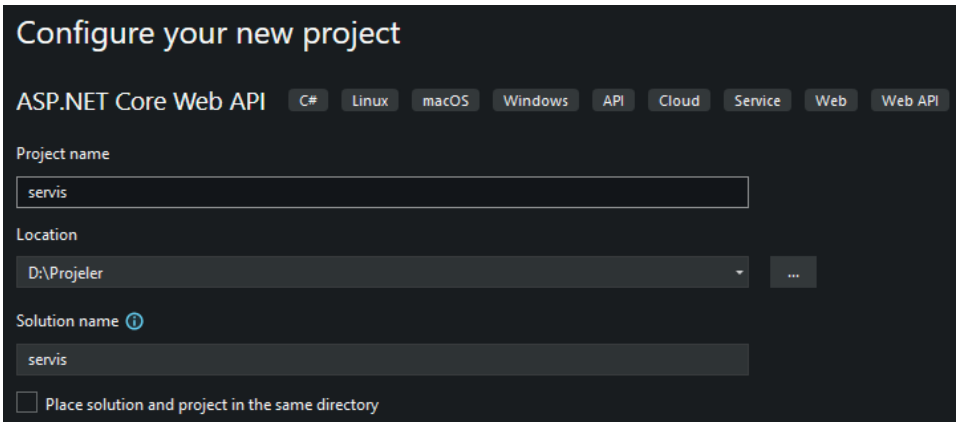
Görsel 5.10: Web uygulama editörü ile yeni proje açılması

2. Adım: Görsel 5.11'deki gibi proje şablonunu seçiniz.



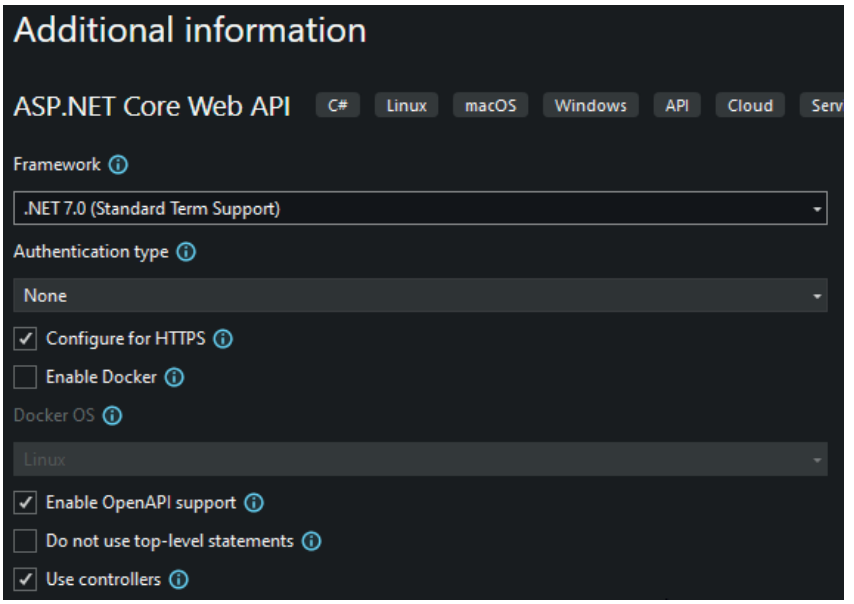
Görsel 5.11: Proje şablonunun seçilmesi

3. Adım: Görsel 5.12'deki gibi projeye ad veriniz.



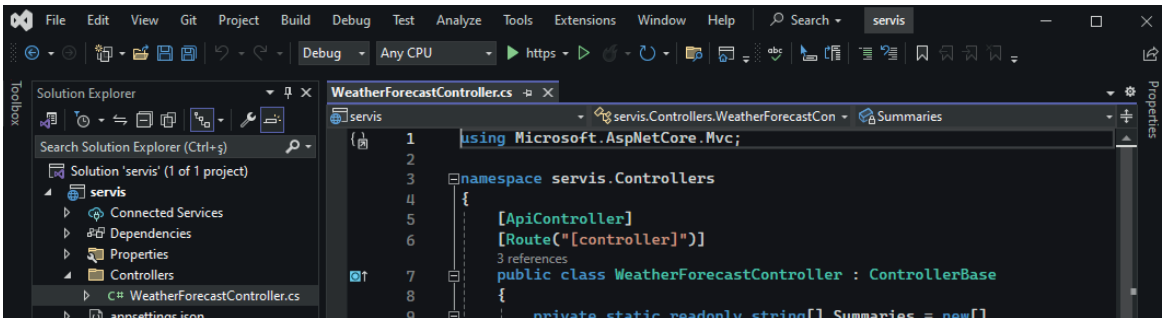
Görsel 5.12: Projeye isim verilmesi ve projenin kaydedileceği klasörün seçilmesi

4. Adım: Görsel 5.13'teki gibi Framework sürümünü seçiniz.



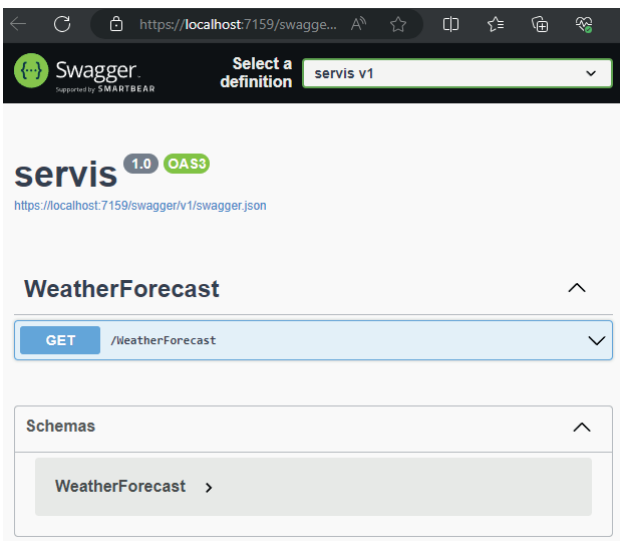
Görsel 5.13: .NET 7.0 olarak Framework sürümünün seçilmesi

5. Adım: Görsel 5.14'teki gibi projedeki hazır kodu inceleyiniz.



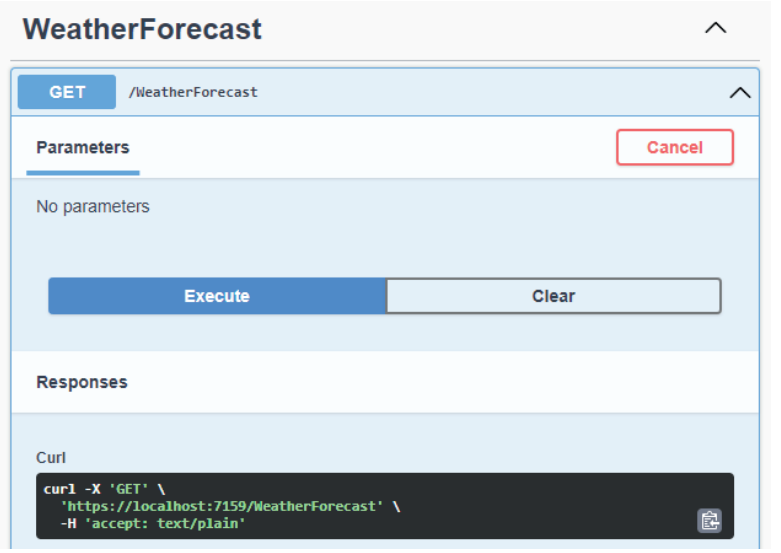
Görsel 5.14: Hazır gelen projenin içindeki WeatherForecastController.cs kodunun incelenmesi

6. Adım: Görsel 5.15'teki gibi projeyi çalıştırınız.



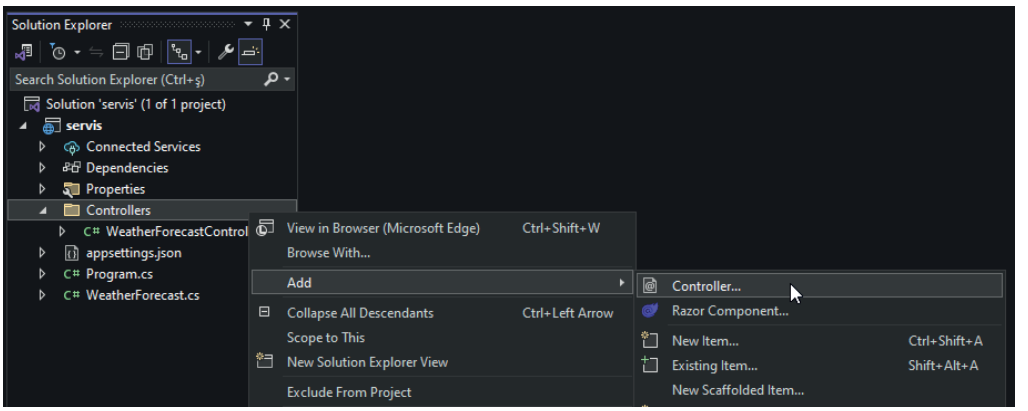
Görsel 5.15: Örnek servisin çalışan görünümü

7. Adım: Görsel 5.16'daki gibi servisi test ediniz.



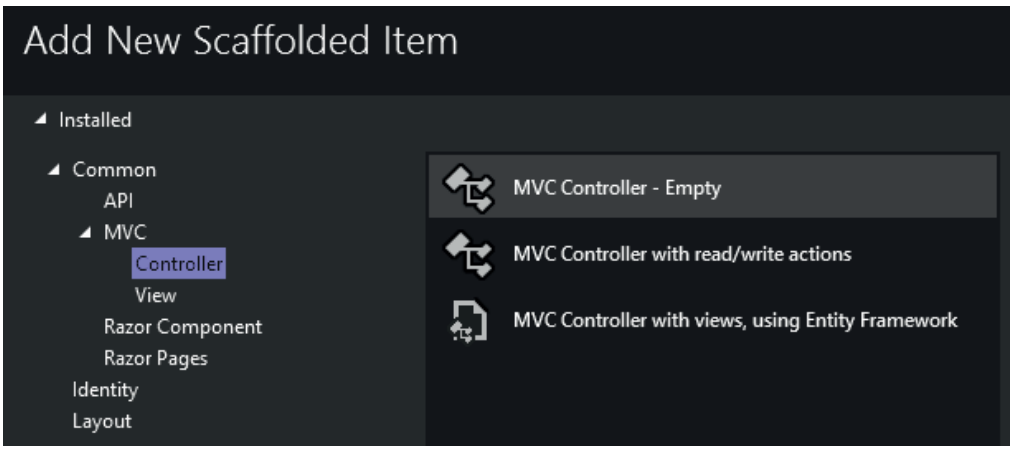
Görsel 5.16: Var olan servisin test edilmesi

8. Adım: Görsel 5.17'deki gibi Controllers klasörüne sağ tıklayıp projeye yeni kontrolcü ekleyiniz.



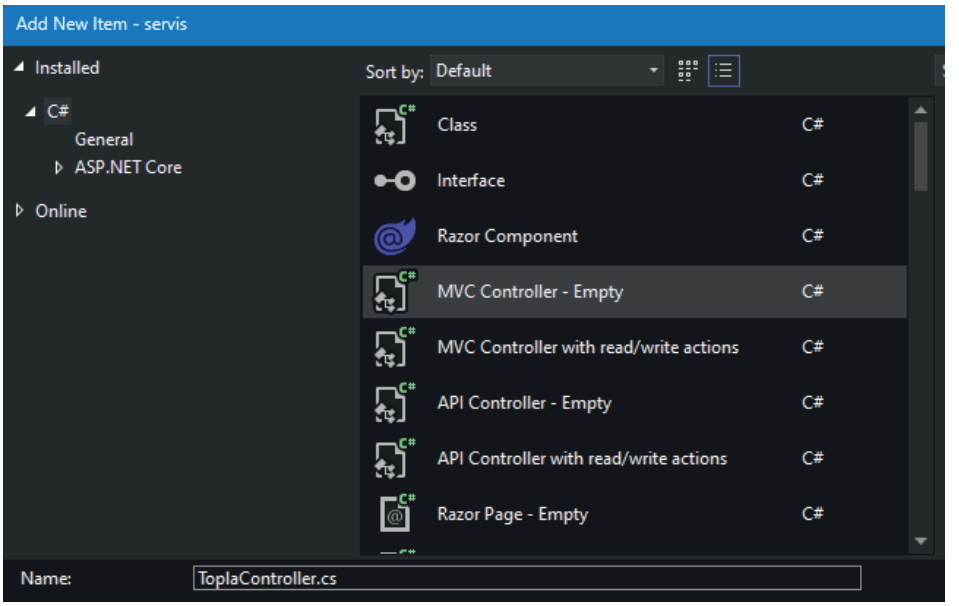
Görsel 5.17: Projeye yeni kontrolcü eklenmesi

9. Adım: Görsel 5.18'deki gibi yeni kontrolcünün türünü seçiniz.



Görsel 5.18: MVC Controller-Empty olarak yeni kontrolcünün seçilmesi

10. Adım: Görsel 5.19'daki gibi yeni kontrolcünün adını seçiniz.



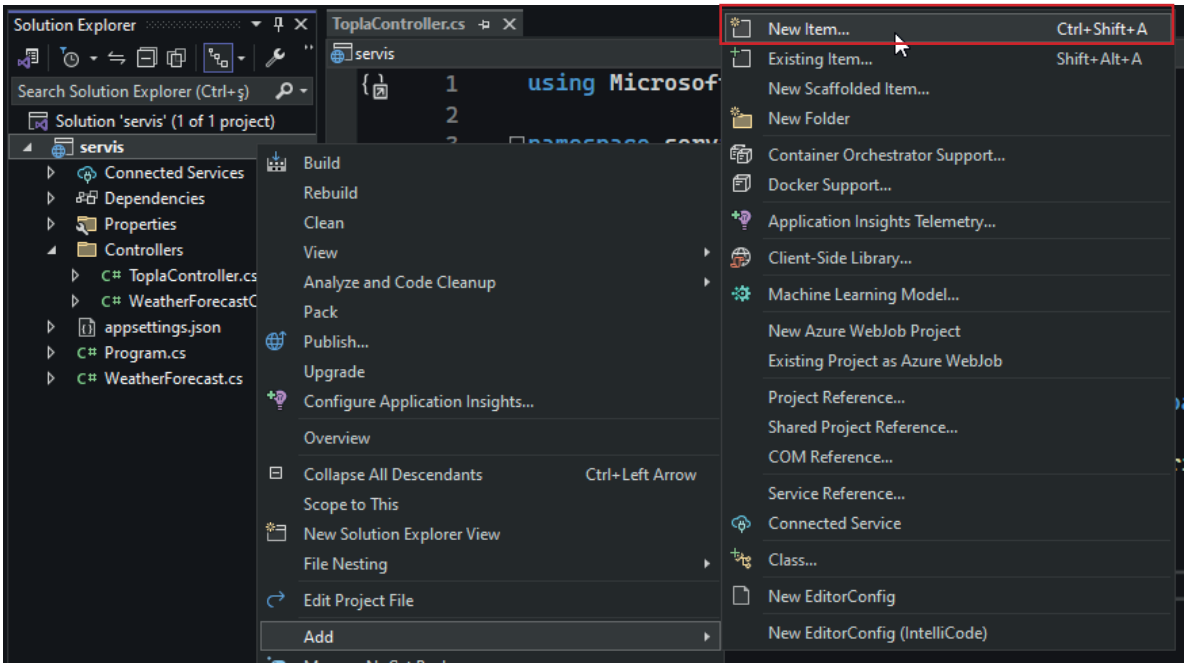
Görsel 5.19: Eklenen yeni kontrolcünün adının ToplaController.cs olarak seçilmesi

11. Adım: Görsel 5.20'deki gibi ToplaController.cs dosyasının içine kodları yazınız.

```
ToplaController.cs
servis                    servis.Controllers.ToplaController    Get(float sayi1, float sayi2)
1   using Microsoft.AspNetCore.Mvc;
2
3   namespace servis.Controllers
4   {
5       [ApiController]
6       [Route("ToplaServisi")]
        0 references
7       public class ToplaController : Controller
8       {
9           [HttpGet(Name = "Topla")]
            0 references
10          public IActionResult Get([FromQuery] float sayi1, [FromQuery] float sayi2)
11          {
12              return Content((sayi1 + sayi2).ToString());
13          }
14      }
15  }
```

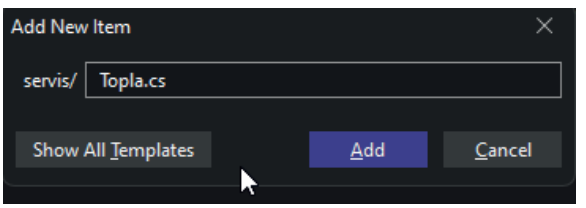
Görsel 5.20: Topla kontrolcüsünün kodlarının yazılması

12. Adım: Görsel 5.21'deki gibi projenin kök dizinine yeni bir sınıf ekleyiniz.



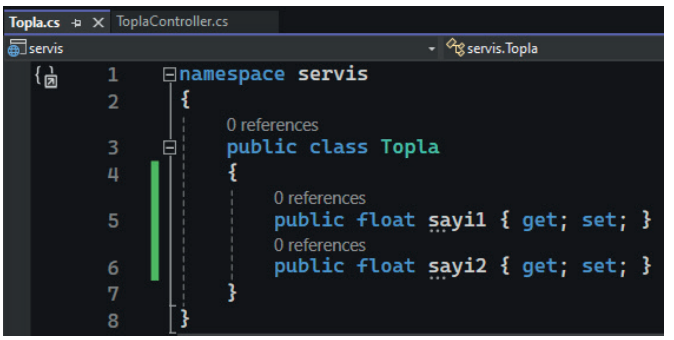
Görsel 5.21: Projenin kök dizinine yeni bir sınıfın eklemesi

13. Adım: Görsel 5.22'deki gibi projenin kök dizinine eklenen sınıfın adını değiştiriniz.



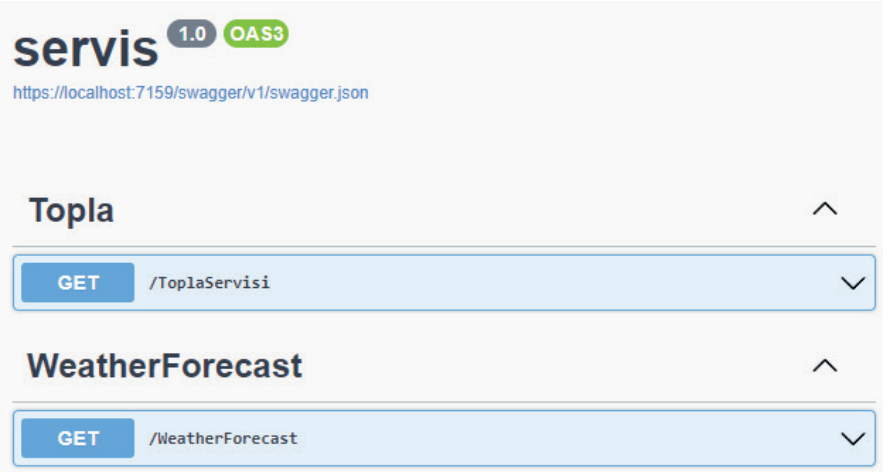
Görsel 5.22: Projenin kök dizinine eklenen sınıf adının Topla.cs olarak değiştirilmesi

14. Adım: Görsel 5.23'teki gibi Topla.cs dosyasının içine kodları yazınız.



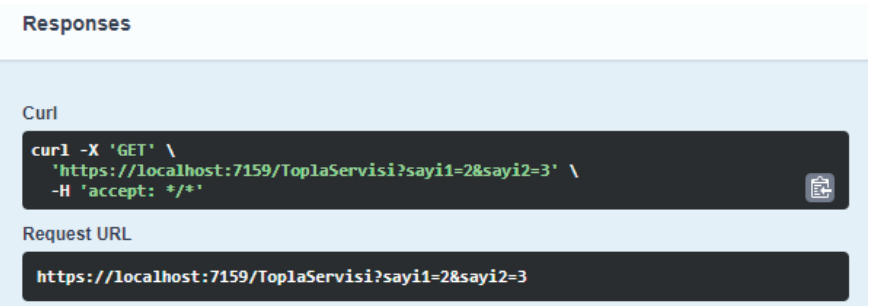
Görsel 5.23: Topla.cs dosyasının içine kodların yazılması

15. Adım: Görsel 5.24'teki gibi yeni servisi çalıştırarak test ediniz.



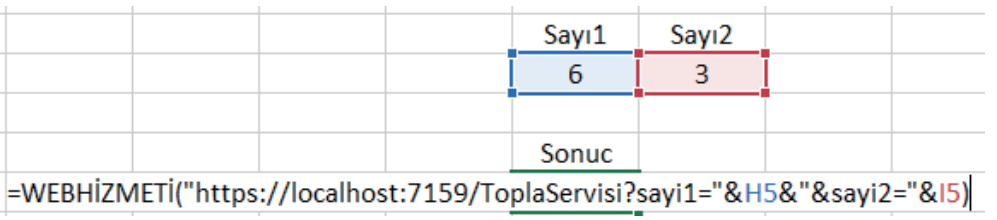
Görsel 5.24: Yeni servisin test edilmesi

16. Adım: Görsel 5.25'teki gibi sonuç sayfasından istek adresini **(Request URL)** elde ediniz.



Görsel 5.25: Request URL adresinin elde edilmesi

17. Adım: Görsel 5.26'daki gibi servis arka planda çalışırken, tablolama programında bir çalışma kitabı açarak servisi deneyiniz.



Görsel 5.26: Sayı1 ve Sayı2 değerlerinin servise yollanması

Not

Bilgisayarda İngilizce ofis programı kurulu ise aynı komut, =WEBSERVICE(url) olarak kullanılabilir. Açık kaynak projeler olan OpenOffice, LibreOffice gibi diğer tablolama uygulamalarında da benzer komutlar vardır.

18. Adım: Görsel 5.27'deki gibi hücrelere veriler giriniz.

Sayı1	Sayı2
33	3
Sonuc	
36	

Görsel 5.27: Sayı1 ve Sayı2 hücrelerine verilerin girilmesi

Not Türkçe bölgesel ayarda ondalık ayracı virgül "," olduğu için ofis yazılımında 3,7 şeklinde bir değer gönderilemez. Programlama dilinde ondalık sayılar, virgül yerine nokta "." ile kullanılır.

19. Adım: Görsel 5.28'deki gibi diğer servisi deneyiniz.

```
=WEBHİZMETİ("https://localhost:7159/WeatherForecast")
```

Görsel 5.28: Hava durumu servisinin hücre içinde kullanılması

20. Adım: Görsel 5.29'daki gibi hava durumu servisinde JSON tipinde çok sayıda verinin ekrana geldiğini gözlemleyiniz.

[{"date":"2024-02-09","temperatureC":45,"temperatureF":112,"summary":"Chilly"},{

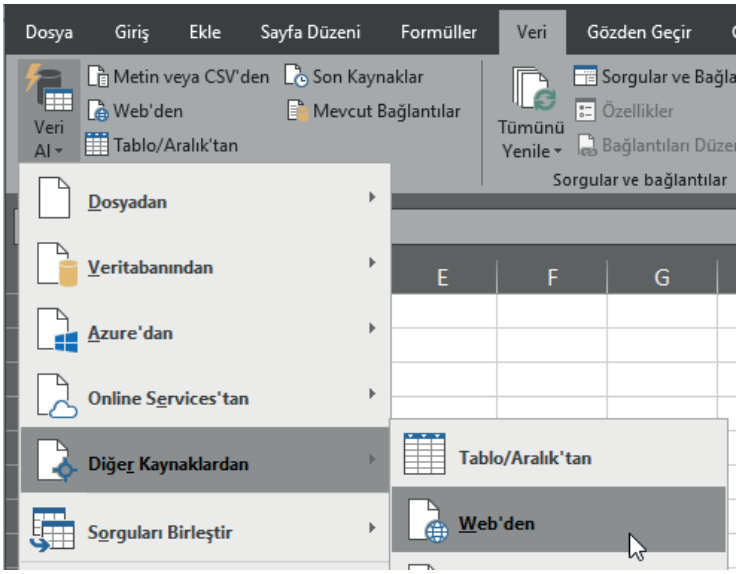
Görsel 5.29: Diğer servisten gelen bilgilerin JSON tipinde bir hücreye getirilmesi

Not **JavaScript Nesnesi Gösterimi (JavaScript Object Notation-JSON)**; verilerin taşınması, depolanması için kullanılan ve dosya boyutu küçük olan bir veri değişim formatıdır. İnsanlar tarafından okunması ve yazılması kolay olan metin tabanlı bir format sunar. JSON genellikle web uygulamaları arasında veri alışverişi için tercih edilir.

Araştırma

JSON olarak gelen bilgilerin hesap tablosu programında parçalanmasını araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

21. Adım: Görsel 5.30'daki gibi hesap tablosu programında **Web'den** komutunu kullanarak veri getiriniz.

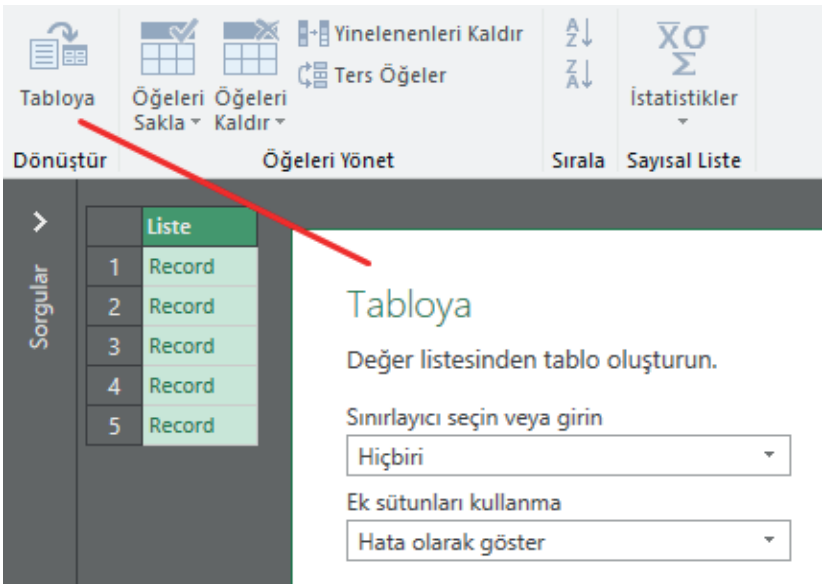


Görsel 5.30: Hesap tablosu programında Web'den komutuyla veri getirilmesi

22. Adım: Görsel 5.31'deki gibi projedeki diğer servis adresini URL metin kutusuna ekleyiniz.

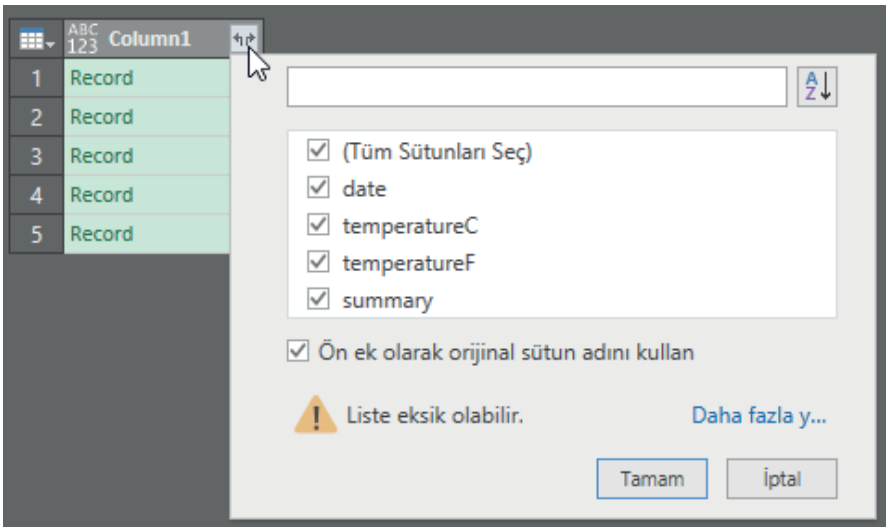
Görsel 5.31: Diğer servis adresinin URL metin kutusuna eklenmesi

23. Adım: Görsel 5.32'deki gibi listedeki kayıtları bölerek tabloya çeviriniz.



Görsel 5.32: Listedeki kayıtların bölünerek tabloya çevrilmesi

24. Adım: Görsel 5.33'teki gibi sütunları seçiniz. Kapat ve Yükle düğmesine basarak pencereyi kapatınız.



Görsel 5.33: Görüntülenecek sütunların seçilmesi

25. Adım: Görsel 5.34'teki gibi gelen bilgileri sayfa içinde görüntüleyiniz.

	A	B	C	D
1	Column1.date	Column1.temperatureC	Column1.temperatureF	Column1.summary
2	2024-02-09	11	51	Bracing
3	2024-02-10	-12	11	Freezing
4	2024-02-11	22	71	Warm
5	2024-02-12	8	46	Balmy
6	2024-02-13	-2	29	Mild

Görsel 5.34: Sayfaya gelen bilgilerin tablo olarak gösterilmesi

Servis, parametre alıp istenen bilgiyi döndürebilirken parametresi olmayan servisler de tablo hâlinde veri döndürebilir. Bilgiler, veri tabanından çekilebilir. Ayrıca bir alışveriş sitesinde stokta ne kadar ürün kaldığı, güncel fiyat bilgileri gibi veriler veri tabanından getirilebilir.

Servis bir uzak sunucuda çalışırken bilgi akışı anlık olmayabilir. Ağda oluşan gecikmelerden dolayı servisin kullanılacağı yazılımın **asenkron** veri alışverişini desteklemesi gerekir.

Örnek

Index.cshtml.cs dosyasının düzenlenmesi için örnek bir metot tanımlama şu şekilde yapılır:

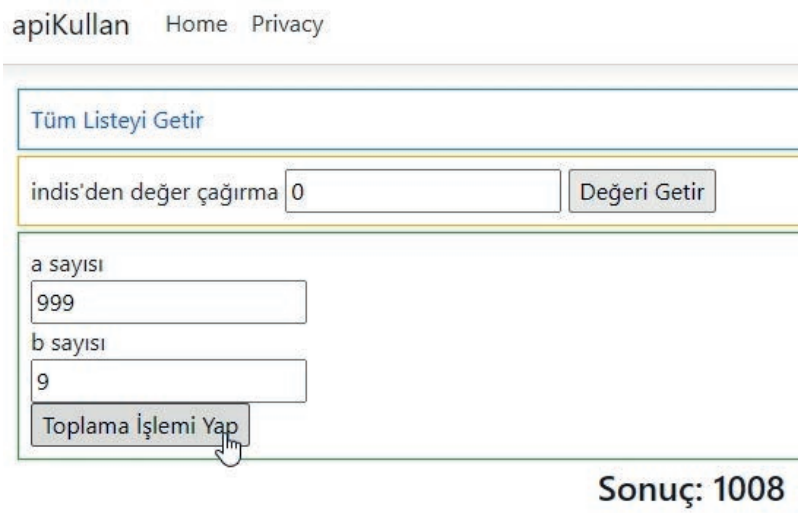
```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.RazorPages;
using Microsoft.Extensions.Logging;
using Newtonsoft.Json;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net.Http;
using System.Net.Http.Headers;
using System.Net.Http.Json;
using System.Text;
using System.Threading.Tasks;

namespace apiKullan.Pages
{
    public class IndexModel : PageModel
   {
        private readonly ILogger<IndexModel> _logger;
        public IndexModel(ILogger<IndexModel> logger)
   {
            _logger = logger;
        }
        public string gelen { get; set; } = "-ilk-"; //Varsayılan değer atama

private const string istekSitesi = "https://localhost:44354/api/hesaplama/";
//Sonunda / karakteri ekli olmalıdır.
        static HttpClient client = new HttpClient();

        public static async Task<string> getir1()
        {
            HttpResponseMessage response = await client.GetAsync(istekSitesi);
            HttpContent content = response.Content;
            var sonuc = await content.ReadAsStringAsync();
            return sonuc;
        }

        public static async Task<string> getir2(string indisi)
        {
            HttpResponseMessage response = await client.GetAsync(istekSitesi + indisi);
            HttpContent content = response.Content;
            var sonuc = await content.ReadAsStringAsync();
            return sonuc;
        }

        public static async Task<string> getir3(string a, string b)
        {
          HttpResponseMessage response = await client.GetAsync(istekSitesi + a + "/" + b);
            HttpContent content = response.Content;
            var sonuc = await content.ReadAsStringAsync();
            return sonuc;
        }
```

```
        public void OnGet()
        {
            gelen = "Hoşgeldiniz";
        }

        public void OnGetTumu()
        {
            var sonuc = getir1();
            Task.WaitAll(sonuc);
            gelen = sonuc.Result.ToString();
        }

        public void OnPostListeden()
        {
            if (Request.Form["indis"] != "")
            {
               var sonuc = getir2(Request.Form["indis"]);
                Task.WaitAll(sonuc);
                gelen = sonuc.Result.ToString();
            }
            else
            {
                gelen = "Boş giriş yapıldı! (1)";
            }
        }

        public void OnPostHesapla()
        {
            if (Request.Form["a"] != "" || Request.Form["b"] != "")

            {
                var sonuc = getir3(Request.Form["a"], Request.Form["b"]);
                Task.WaitAll(sonuc);
                gelen = sonuc.Result.ToString();
            }
            else
            {
                gelen = "Boş giriş yapıldı! (2)";
            }
        }
    }
}
```

Index.cshtml dosyasında tasarım kısmının kodları şu şekildedir:

```
@page "{handler?}"
@model IndexModel
@{
    ViewData["Title"] = "Web API Client - İstemcisi";
}

<div class="border border-info p-2 m-1">
    <a asp-page="/Index" asp-page-handler="Tumu">Tüm Listeyi Getir</a>
</div>
<form method="post" asp-page-handler="Listeden" class="border border-warning p-2 m-1">
    indis'den değer çağırma
    <input type="text" name="indis" value="0" autofocus />
    <input type="submit" value="Değeri Getir" />
</form>
<form method="post" asp-page-handler="Hesapla" class="border border-success p-2 m-1">
    a sayısı
    <br />
    <input type="text" name="a" value="0" />
    <br>
    b sayısı
    <br />
    <input type="text" name="b" value="0" />
    <br>
    <input type="submit" value="Toplama İşlemi Yap" />
</form>
<div class="text-center h4">
    Sonuç: @Model.gelen
</div>
```

Asenkron veri alışverişi örneğinin ekran görüntüsü Görsel 5.35'te verilmiştir.



Görsel 5.35: Asenkron veri örneği

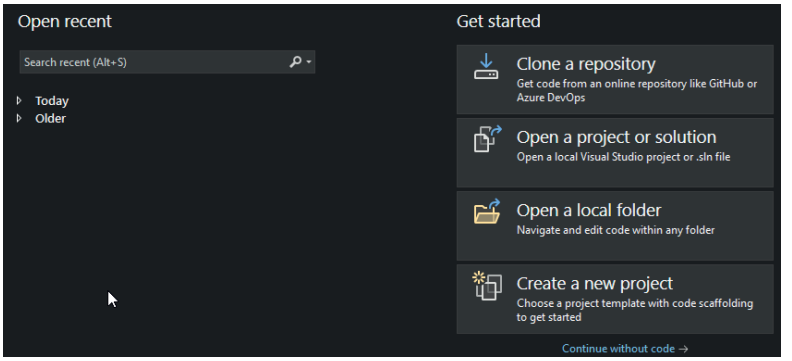
5.3.2. .Net Core Web Uygulaması

Web uygulama editörü ile basit bir web projesi oluşturularak web uygulamaları geliştirilmeye başlanabilir.

4. Uygulama

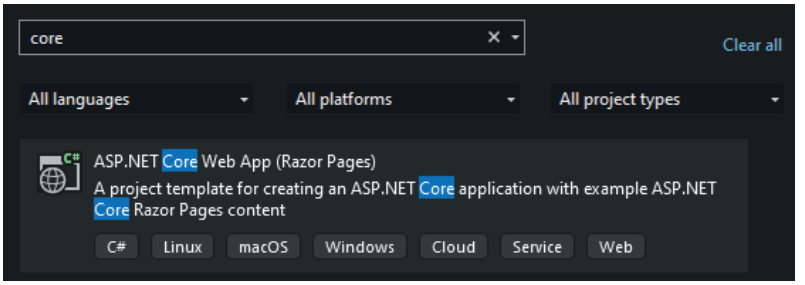
İşlem adımlarına göre yeni bir web programlama dili Core Web uygulaması yapınız.

1. Adım: Görsel 5.36'daki gibi web uygulama editörünü açarak yeni bir proje oluşturunuz. Web uygulama editörü, Başlat menüsü>devenv komutu verilerek açılabilir.



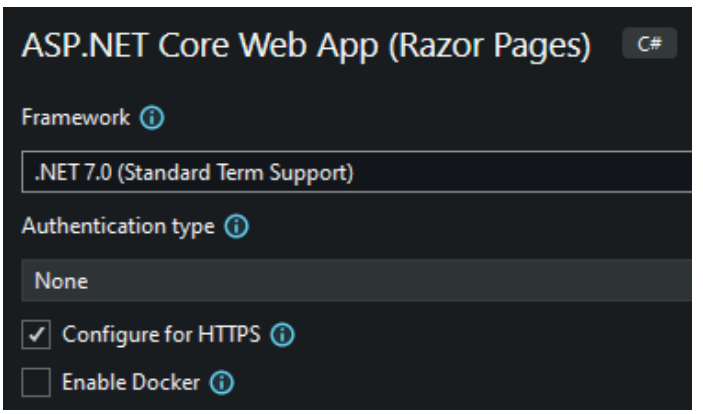
Görsel 5.36: Yeni projenin (Create a new project) oluşturulduğu veya var olan projelerin seçildiği açılış ekranı

2. Adım: Görsel 5.37'deki gibi yeni proje oluştururken arama kutusuna **core** yazınız ve programlama dilini seçiniz.



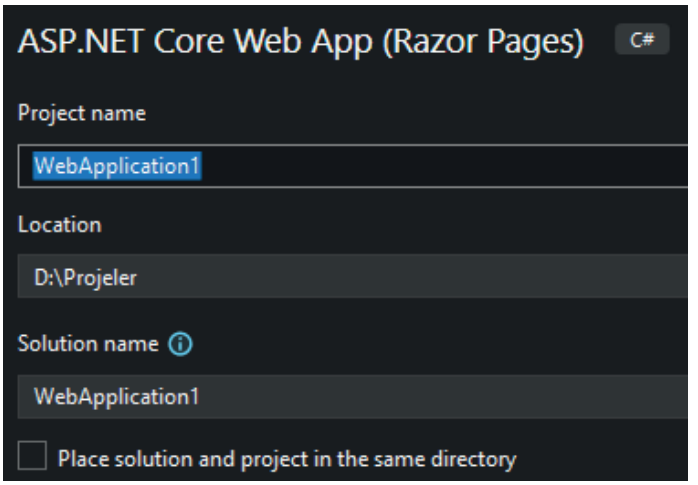
Görsel 5.37: Arama kutusuna core yazılması ve programlama dilinin seçilmesi

3. Adım: Görsel 5.38'deki gibi yeni proje için Framework sürümünü seçiniz.



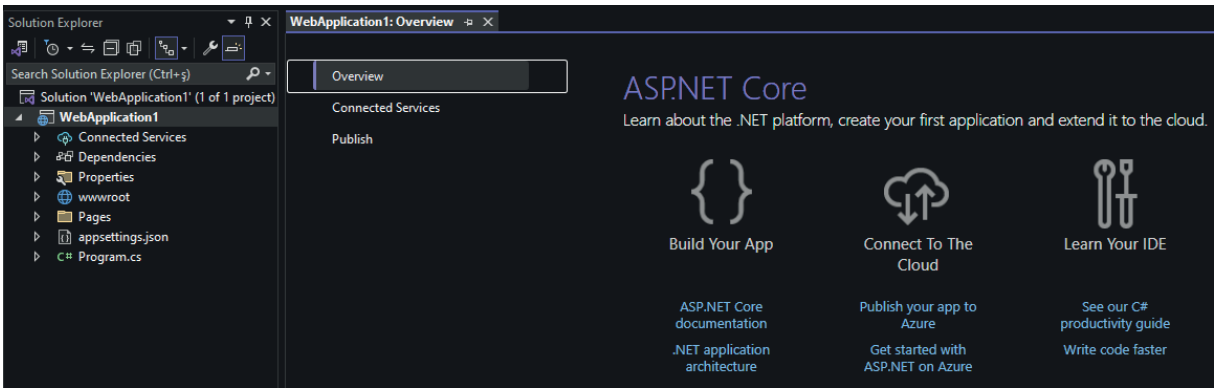
Görsel 5.38: Framework sürümünün seçilmesi

4. Adım: Görsel 5.39'daki gibi yeni proje adını ve klasörünü seçiniz.



Görsel 5.39: Proje adının ve klasörünün seçilmesi

5. Adım: Görsel 5.40'taki gibi projenin ilk açılış penceresini inceleyiniz.

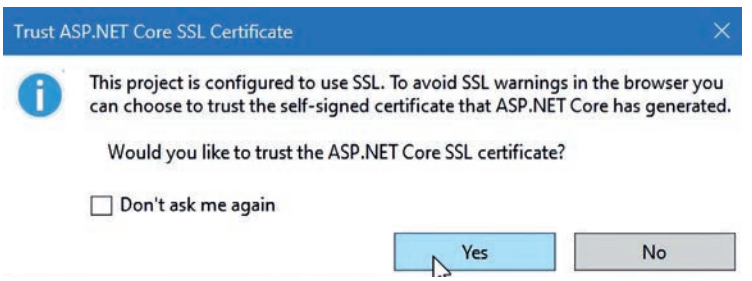


Görsel 5.40: Projenin ilk açılış penceresi

6. Adım: F5 kısayol tuşu ile projeyi test ediniz.

Not Web uygulama başlatıldığında arka planda **IIS (Internet Information Services)** adındaki web sunucusu programının küçük bir sürümü çalışır.

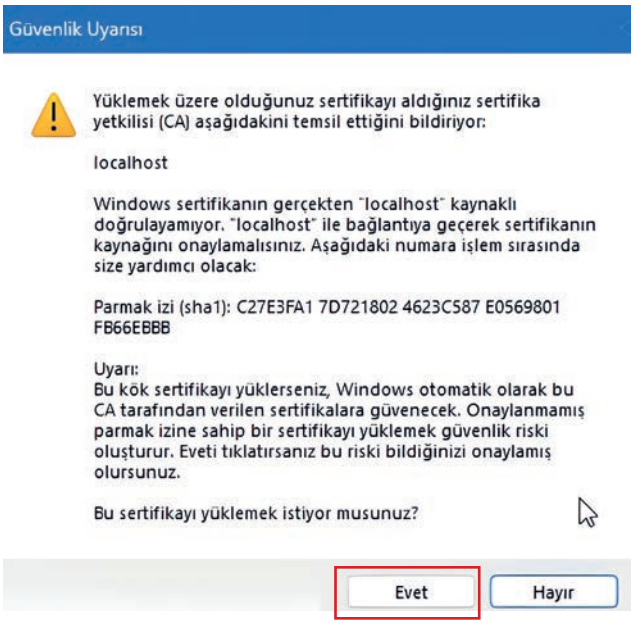
7. Adım: Projenin ilk çalıştırmasında **HTTPS** ile ilgili güvenlik uyarılarını kabul etmek için Yes kutucuğunu tıklayınız (Görsel 5.41).



Görsel 5.41: SSL sertifikasına güvenmek için Yes kutucuğunun tıklanması

Not

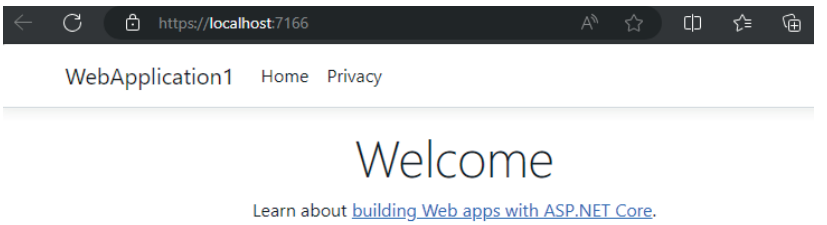
SSL (Secure Sockets Layer), veri iletişiminde güvenliği sağlamak için kullanılan bir protokoldür. Veri iletişiminin şifrelenmesinde ve kimlik doğrulamasında SSL'den yararlanılır. SSL genellikle web tarayıcıları ile sunucular arasındaki iletişimde kullanılır. Günümüzde sitelerin çoğunluğunda HTTPS, bir başka deyişle SSL desteği vardır. SSL desteği ile ilgili diğer güvenlik uyarı diyaloğu Görsel 5.42'de verilmiştir.



Görsel 5.42: SSL sertifikasının yüklenmesi için Evet kutucuğunun tıklanması

Not

Dördüncü uygulamada https://localhost:7137 olarak bir adres verilmiştir fakat 7137 kapı (port) numarası her bilgisayarda farklıdır ve istenirse proje seçeneklerinden port değiştirilebilir (Görsel 5.43).



Görsel 5.43: Projenin tarayıcıda görüntülenmesi

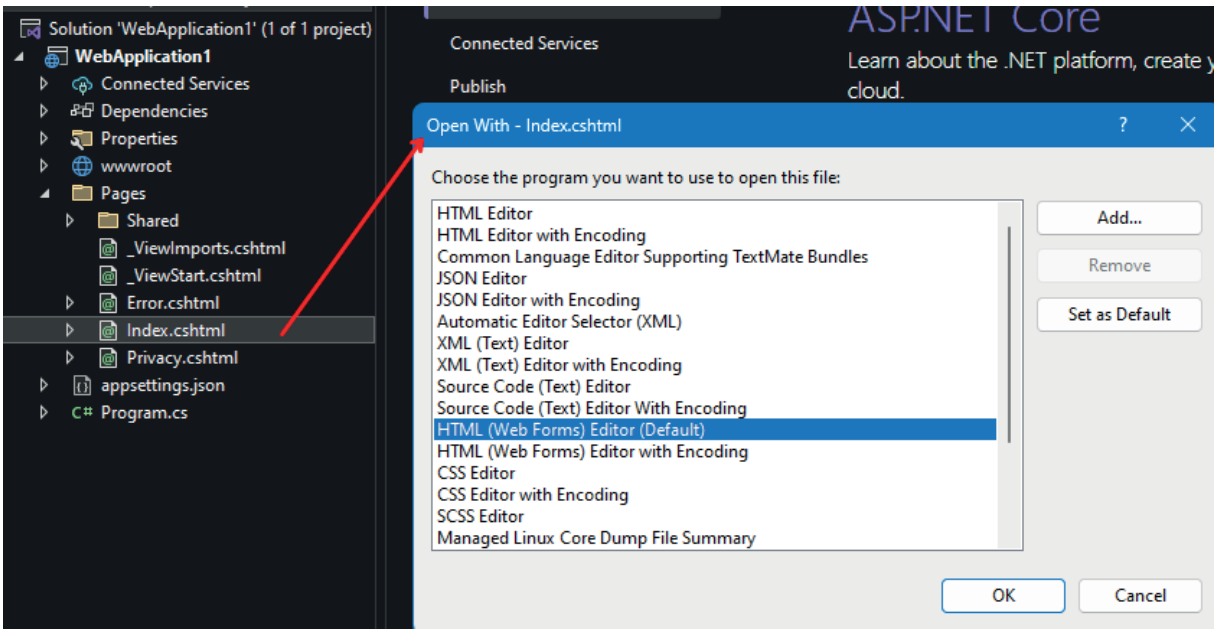
Çözüm Gezgini (Solution Explorer) penceresinde şu dosya ve klasörler listelenir:

- **Connected Services:** Çeşitli veri tabanı servisleri eklenir.
- **Dependencies:** Bileşen paketleri yönetilir.
- **Properties:** launchSettings.json ile sitenin sunucu ayarları yapılır.
- **wwwroot:** css, js, bootstrap gibi statik kaynaklar yönetilir.
- **Pages:** Web sayfaları .cshtml olarak düzenlenir.
- **appsettings.json:** Uygulama yapılandırması düzenlenir.
- **Program.cs:** Proje çalıştırılırken hangi servislerin açılacağı belirtilir.

Not

NuGet, .NET platformunda paketleme ve paketleri dağıtmak için kullanılan bir açık kaynak paket yöneticisidir. .NET uygulamalarında kullanılan üçüncü taraf kütüphaneleri; bileşenleri ve araçları kolayca keşfetmek, indirmek, güncellemek için kullanılır. Bu platform için Tools>NuGet Package Manager>Manage Packages for Solution... menüsü kullanılır.

8. Adım: Görsel 5.44'teki gibi anasayfanın form tasarımını düzenleyiniz.



Görsel 5.44: Anasayfanın tasarlanması için HTML (Web Forms) Editor seçilmesi

Not

Kullanıcıdan veri almak için form etiketi kullanılır. Metin kutuları, onay ve açılır kutu tiplerinde birçok form elemanı vardır. HTML komutları kullanılarak form ve form elemanlarının görünüm kısmı hazırlanır.

Tablo 5.3'te sık kullanılan HTML form etiketleri ve bu etiketlerin açıklamaları verilmiştir.

Tablo 5.3: Sık Kullanılan HTML Form Etiketleri ve Bu Etiketlerin Açıklamaları

Etiket	Açıklamalar
<form>	Formun başlangıcını belirtir ve verilerin sunucuya gönderilmesini sağlar.
<button>	Tıklanabilir bir düğme oluşturur.
<input>	Kullanıcıdan veri alır.
<label>	<input> etiketiyle ilişkili metin etiketidir.
<select>	Seçeneklerin listesini oluşturur.
<option>	<select> etiketi içindeki seçenekleri belirtir.
<textarea>	Çok satırlı metin girişini sağlar.

Form ile bilgi yollamak için kullanılan yöntemler şunlardır:

- **POST:** Form verileri, HTTP isteği gövdesinde **gizli** bir şekilde iletilir. Kullanıcı girdileri, form elemanlarının değerleri olarak paketlenir ve sunucuya gönderilir. POST yöntemi, form verilerinin güvenli bir şekilde iletilmesini sağlar ve sınırlı veri boyutu olmadığı için daha büyük veri setlerinin gönderilmesine uygundur.
- **GET:** Form verileri, HTTP isteği URL'nin bir parçası olarak **açık** bir şekilde iletilir. Kullanıcı girdileri, form elemanlarının adı ve değeriyle birlikte URL'ye eklenir. GET yöntemi, daha küçük veri boyutları için uygundur. GET yöntemi genellikle sorgu parametreleri için kullanılır ancak GET istekleri URL üzerinden yapıldığı için veriler, tarayıcı geçmişinde ve URL paylaşımlarında görülebilir. Web tarayıcılarının URL boyutunda sınırlandırmaları vardır. Bu nedenle büyük veri iletmek için GET yöntemi uygun değildir.

Not

POST yönteminin güvenli olduğu düşünülse de HTTPS kullanılmadığında bilgiler, yerel ağda açık metin şeklinde izlenebilir. Programcı, yöntemi POST olarak seçse de kullanıcı tarafında yöntemler zorla değiştirilebilir. GET yönteminde tüm bilgiler adres çubuğunda görüldüğü için gönderilen bilgilerin içinde şifre, güvenlik sorusunun cevabı gibi önemli bilgiler olmamalıdır.

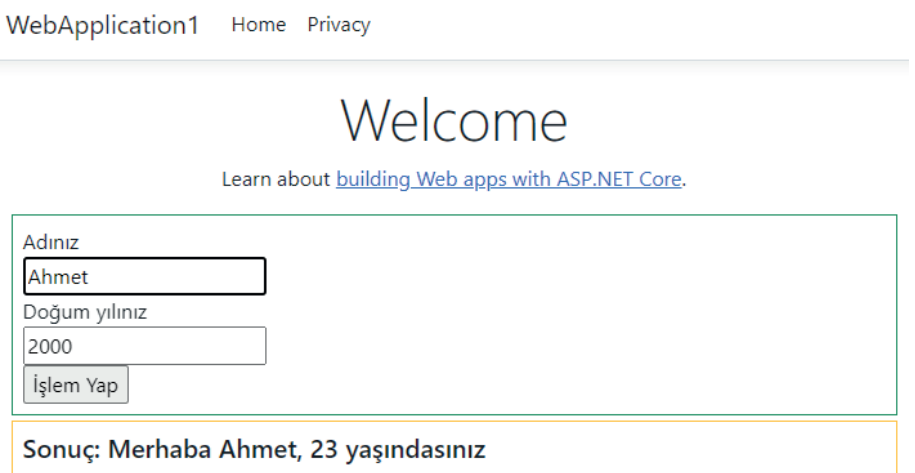
9. Adım: Sayfanın görünümü için **Index.cshtml** dosyasını düzenleyiniz.

```
@page
@model IndexModel
@{
    ViewData["Title"] = "Home page";
}

<div class="text-center">
    <h1 class="display-4">Welcome</h1>
    <p>Learn about <a href="https://docs.microsoft.com/aspnet/core">building Web apps with ASP.NET
Core</a>.</p>
</div>
<form method="post" class="border border-success p-2 m-1">
    Adınız <br />
    <input type="text" name="ad" value="" placeholder="Adınızı giriniz" /><br />
    Doğum yılınız <br />
    <input type="number" name="yil" value="2000" placeholder="Doğum yılınızı giriniz" />
    <br />
    <input type="submit" value="İşlem Yap" />
</form>

<div class="border border-warning p-2 m-1 h5">
    Sonuç: @Model.gelen
</div>
```

10. Adım: Kodlama için Index.cshtml.cs dosyasını düzenleyiniz.

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.RazorPages;

namespace WebApplication1.Pages
{
    public class IndexModel : PageModel
    {
        private readonly ILogger<IndexModel> _logger;
        public string gelen = "varsayılan değer"; //Global bir değişken tanımlama

        public IndexModel(ILogger<IndexModel> logger)
        {
            _logger = logger;
        }

        public void OnGet()
        {
            gelen = "ilk açılış";        //Sayfanın ilk görüntülendiği an
        }

        public void OnPost()
        {                                          //Sayfadan kullanıcı bilgilerinin geldiği an
            if (Request.Form["ad"] != "" && Request.Form["yil"] != "")
            {
                int yas = 2023 - Convert.ToInt32(Request.Form["yil"]);

                var sonuc = String.Format("Merhaba {0}, {1} yaşındasınız", Request.Form["ad"], yas);

                gelen = sonuc.ToString();
            }
            else
            {
                gelen = "Boş giriş vardır!";
            }
        }
    }
}
```

11. Adım: Verileri girdikten sonra İşlem Yap düğmesine basıp sayfayı yenileyiniz (Görsel 5.45).



Görsel 5.45: Veri girilerek anasayfanın (Index) test edilmesi

Basit bir uygulamada bile beklenmedik sorunlar olabilir. Kullanıcılardan gelen giriş bilgileri ve bu bilgilerde olabilecek tüm ihtimaller denenmelidir. Örneğin doğum yılının sayısal olacağı planlansa bile kullanıcı, sayfanın kaynağından **number** tipini **text** yaparsa istediği bilgiyi girebilir. Bunun sonucunda, web uygulamasında hesaplama hatası görülür. Veri girişlerinde gerekli kısıtlamalar yapılmazsa istenmeyen veriler oluşabilir.

Web programlama dili içinde **doğrulama (validation)** komutları vardır. Boş giriş, belli biçime uygunluk ve hatalı metinler işlemden geçirilerek kullanıcıya bu bilgileri doğru şekilde girmesi için uyarı verilir.

Başka bir hata durumu da sunucuya binlerce istekte bulunarak sunucuyu servis veremez hâle getirmektir. İstekler arasına belli bir süre kısıtlaması eklenerek **sel (flood)** saldırısı engellenebilir.

Zararsız gibi görünen ad bilgisinin ekrana yazdırılmasında **tehlikeli HTML, JavaScript ve SQL kodları** da olabilir. Örneğin <script>alert("mesaj")</script> gibi bir metin girildiğinde ekrana mesaj kutusu gelirse uygulama XSS (Cross-Site Scripting) saldırısına açık hâldedir. Web programlama dili, gelen bilgileri otomatik olarak temizler.

HTML temizliği (sanitation), kullanıcı tarafından sağlanan veya güvenilmeyen kaynaklardan alınan HTML içeriğinin güvenli hâle getirilmesi işlemidir. Bu işlem; zararlı HTML etiketleri, özellikleri veya içerikleri filtrelenerek gerçekleştirilir.

Kod temizliği için şu işlemler yapılarak güvenli bir HTML içeriği oluşturabilir:

- **HTML Etiketlerini Filtreleme:** Sadece güvenli etiketlere izin verilirken zararlı etiketler kaldırılır veya "\" kaçış karakterleriyle etkisiz hâle getirilir.
- **Özellik Filtreleme:** Sadece güvenli özelliklere izin verilirken, potansiyel olarak tehlikeli özellikler kaldırılır veya devre dışı bırakılır.
- **İçerik Temizleme:** Zararlı içerikler, HTML içeriğinden kaldırılır veya etkisiz hâle getirilir.
- **Doğrulama:** Güvenli ve beklenen bir yapıya sahip olup olmadığının doğrulanması için HTML içeriği analiz edilir ve düzeltilir.

HTML temizliği, güvenilir bir HTML **ayrıştırma (parser)** veya bir sanitasyon kütüphanesi kullanılarak gerçekleştirilebilir. Bu kütüphaneler, güvenli HTML içeriği üretmek için gelişmiş filtreleme ve doğrulama mekanizmaları sağlar. Burada önemli olan nokta, güvenli HTML temizliği sürecini uygulamak ve güvenilmeyen kaynaklardan gelen **HTML içeriğini kullanmadan önce** sanitasyon işlemini yapmaktır.

Web uygulamasındaki bir metin kutusunda unutulan açık, tüm web sitesini etkiler. Bu tür açıklar sadece metin kutularında değil oturum, çerez, veri tabanı gibi konularda da geçerlidir. Değişik yollarla web uygulamasına ve veri tabanına sızan zararlı bir kod, web uygulama her çalıştırıldığında sisteme zarar vermeye devam eder.

5.3.3. PHP Web Uygulaması

Web programlama dillerinin tamamında benzer güvenlik problemleri yaşanır. Kötü niyetli kişi veya kişiler; hedef ortamın işletim sistemini, kod altyapısını ve veri tabanı sürümlerini öğrenip, belli açıklardan yararlanarak saldırır. Kullanılan sistem eski sürüm ise birçok açık barındırır.

Tablo 5.4'te açık kaynak olan bazı içerik yönetimi sistemleri **(Content Management System-CMS)** ve bu sistemlerin altyapısında kullanılan diller verilmiştir.

Tablo 5.4: CMS Altyapısında Kullanılan Diller ve CMS'nin Açıklamaları

CMS	Kullanılan Dil	Açıklamalar
Drupal	PHP	Güçlü ve özelleştirilebilir bir CMS'dir.
Joomla	PHP	Esnek ve ölçeklenebilir bir CMS'dir.
Magento	PHP	E-ticaret siteleri için özel olarak tasarlanmış bir CMS'dir.
TYPO3	PHP	Ölçeklenebilir ve çok dilli web siteleri için kullanılan bir CMS'dir.
Umbraco	.NET	.NET tabanlı bir CMS'dir. Kurumsal web siteleri için uygundur.
WordPress	PHP	En yaygın kullanılan CMS'dir. Bloglar ve web siteleri için idealdir.

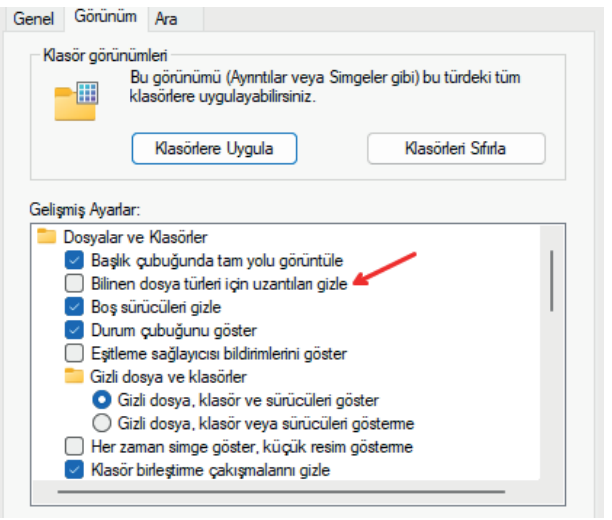
5. Uygulama

İşlem adımlarına göre web programlama dilini kullanarak Merhaba Dünya uygulaması yapınız.

1. Adım: XAMPP kontrol panelini açarak diskinizdeki **C:\xampp\htdocs\proje** klasörüne geliniz.

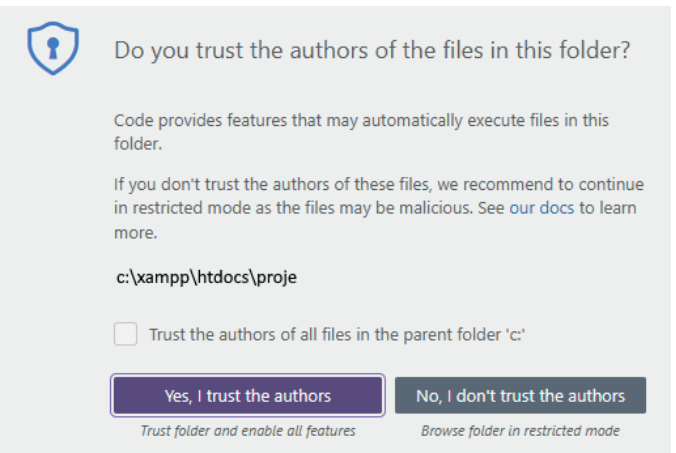
Not

İşletim sisteminde varsayılan olarak dosya uzantıları gösterilmez. Başlat>Dosya Gezgini Seçenekleri penceresinden **Görünüm>Bilinen dosya türleri için uzantıları gizle** seçeneğini boş bırakılır (Görsel 5.46).



Görsel 5.46: Dosya uzantılarının gösterilmesi

2. Adım: Web uygulama editörü ile çalışma klasörünü açınız (Görsel 5.47).



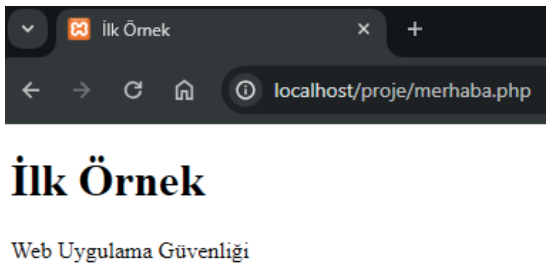
Görsel 5.47: Web uygulama editörü içinde klasör açılması

3. Adım: File>New File... komutu ile **merhaba.php** adında bir dosya oluşturunuz.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>İlk Örnek</title>
</head>
<body>
    <h1>İlk Örnek</h1>
    <?php
        echo "Web Uygulama Güvenliği";
    ?>
</body>
</html>
```

4. Adım: Oluşturduğunuz dosyayı çalıştırmak için tarayıcıda adres çubuğuna **localhost/proje/merhaba.php** yazınız (Görsel 5.48).

5. Adım: Web uygulamasına form ve veri giriş etiketlerini ekleyiniz.



Görsel 5.48: Web programlama dili ile yapılan sayfanın çalıştırılması

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>İlk Örnek</title>
</head>

<body>
    <h1>İlk Örnek</h1>
    <?php
        echo "Web Uygulama Güvenliği";
    ?>
    <hr>
    <form action="" method="post">
        Adınız <br />
        <input type="text" name="ad" placeholder="Adınızı giriniz"> <br />
        Doğum Yılınız<br />
        <input type="number" name="yil" value="2000" placeholder="Doğum yılınızı giriniz">
        <br />
        <input type="submit" value="İşlem Yap">
    </form>
    <br />
    <?php
        if ($_POST) {
            if (!empty($_POST["ad"]) && !empty($_POST["yil"])) {
                echo "Merhaba ".$_POST["ad"].", ".(2023-$_POST["yil"]). " yaşındasınız.";
            } else {
                echo "Boş giriş vardır!";
            }
        }
    ?>
</body>

</html>
```

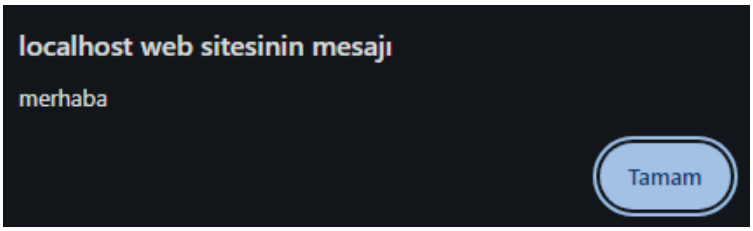
Not <?php ile ?> arasında yazılanlar, web programlama dilinin kodlarıdır. HTML ve PHP, iç içe kullanılabilir. Hem programlama dili içine HTML etiketleri hem de HTML içine programlama kodları yazılabilir.

6. Adım: Görsel 5.49'daki gibi projeyi test ediniz.

Görsel 5.49: Web programlama dili ile yapılan form örneği

7. Adım: Adınız yerine <script>alert("merhaba")</script> yazarak JavaScript kodunun çalıştığını gözlemleyiniz (Görsel 5.50).

Bu programda doğrulama, XSS, DoS gibi saldırılar için alınan bir önlem yoktur.



Görsel 5.50: JavaScript enjeksiyonunun sonucu

Not Gelen bilgideki script veya <script> bölümleri silinerek saldırı engellenmeye çalışılsa da <scriscriptpt>alert("merhaba")</scscriptript> gibi atlatma yöntemleri vardır. "<" ve ">" karakterleri silinse de bu karakterler yerine özel karakterler kullanılarak saldırı yapılabilir.

Araştırma

Arama motorunu kullanarak XSS yazım şekillerini araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

Sıra Sizde

Web programlama dilini kullanarak yapılan projede XSS saldırısı gerçekleştiriniz. Oluşan hataları not ediniz.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. C:\xampp\htdocs\ çalışma klasörüne proje klasörü oluşturdu.		
2. Web uygulama editörü ile çalışma klasörünü açtı.		
3. merhaba.php adında bir dosya oluşturdu.		
4. Tarayıcıda localhost/proje/merhaba.php sayfasının açılmasını sağladı.		
5. Uygulama kodlarına form ve veri giriş etiketlerini ekledi.		
6. Projeyi çalıştırarak güvenlik sorunlarının açıklamasını yaptı.		
"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.		

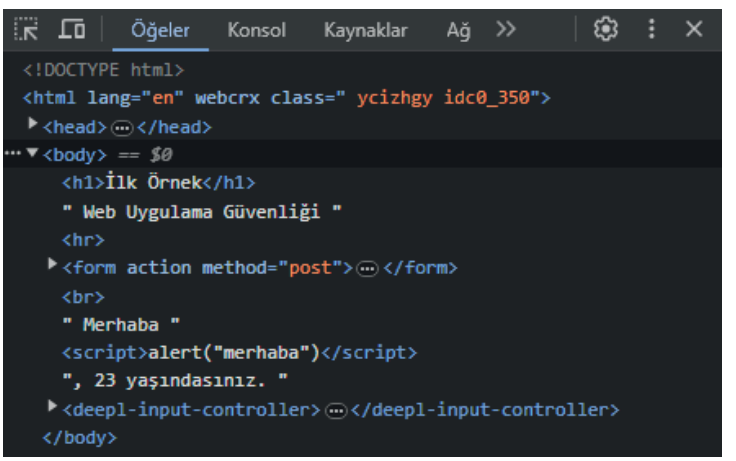
5.4. WEB UYGULAMA GÜVENLİĞİ

Web uygulamaları hem istemci hem de sunucu tarafında daha güvenli hâle getirilmelidir.

5.4.1. İstemci Taraflı Girdi Denetimi

Frontend, istemci (kullanıcının gördüğü alanda) bazı önlemler alınabilir fakat ne kadar kısıtlama eklense de kötü niyetli bir kullanıcı kolayca HTML, CSS ve JavaScript komutlarını değiştirebilir, sayfanın kaynak kodlarını inceleyebilir, **Geliştirici araçları** ile sayfada düzenlemeler yapabilir.

Tarayıcıdaki Geliştirici araçları sekmesi, **F12** kısayolu veya menüden Diğer Araçlar>Geliştirici Araçları Ctrl+Shift+I ile açılabilir (Görsel 5.51).



Görsel 5.51: Geliştirici araçları sekmesi

Geliştirici araçlarını özellikle istemci tarafında tasarım yapan geliştiriciler çok kullanır. İstemci taraflı girdi denetimi, web uygulamalarında kullanıcılar tarafından sağlanan girdilerin güvenli bir şekilde işlenmesi ve doğrulanması sürecini ifade eder. Kullanıcının tarayıcısı veya istemcisi tarafında da gerçekleştirilen denetimleri içerir. Bu şekilde kullanıcının girdileri doğrulanırken hatalı veya kötü niyetli girişlerin hemen tespit edilmesi sağlanır ve gerektiğinde kullanıcıya hata mesajları gösterilerek kullanıcıdan girdileri düzeltmesi istenir.

Not

İstemciden farklı yollar kullanılarak gelen **hiçbir bilgiye** güvenilmemelidir. Web geliştiricisi, gelen bilgileri kullanmadan önce zararlı içeriklerden temizlemelidir. Örneğin bir giriş formunda kullanıcının adını ve e-posta adresini girmesi istenirse istemci taraflı girdi denetimi, tarayıcıda JavaScript gibi bir programlama dilinin kullanılmasıyla gerçekleştirilebilir. Kullanıcının girdileri, tarayıcıda anlık olarak doğrulanır ve hatalı giriş olduğunda kullanıcıya bir uyarı gösterilir. Bu sayede hatalı veya saldırgan girdiler, sunucuya gönderilmeden önce tespit edilir. İstemci taraflı girdi denetimi, kullanıcının doğru ve güvenli veri girişi yapmasını teşvik etmekle birlikte sunucu tarafında gerçekleştirilen denetimlerle kullanıldığında daha güvenli bir web uygulaması sağlar.

6. Uygulama

İşlem adımlarına göre güvenli web formu hazırlayınız.

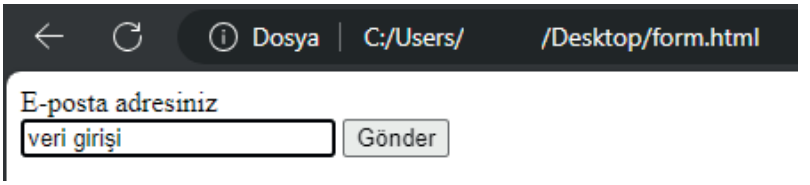
1. Adım: Masaüstünde boş bir alana sağ tıklayıp **Yeni>Metin Belgesi** ile bir form.html dosyası oluşturunuz.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Form Güvenliği</title>
</head>
<body>
    <form action="" method="post">
        E-posta adresiniz <br/>
        <input type="text" name="eposta">
        <input type="submit" value="Gönder">
    </form>
</body>
</html>
```

Not

Form etiketinin action özelliğine farklı bir sitenin adresi yazılabilir. Örneğin http://www.ornekbiraramamotoru.com hedefine GET yöntemi ile doğru parametreler sayesinde arama bilgisi gönderilebilir.

2. Adım: Kaydedilen dosyaya çift tıklayıp varsayılan tarayıcıda sonucu gösteriniz (Görsel 5.52).

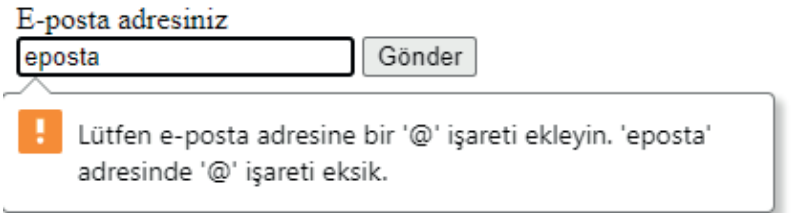


Görsel 5.52: Gönder tuşuna basıldığında önlem alınmadan veri gönderen form örneği

Not E-posta bilgisi istense de bu örnekte kullanıcı; boş giriş, herhangi bir yazı, çok uzun bir metin, sayı veya telefon numarası şeklinde yanlış veri türleri girebilir.

3. Adım: İlk önlem olarak giriş tipini email yapınız (Görsel 5.53).

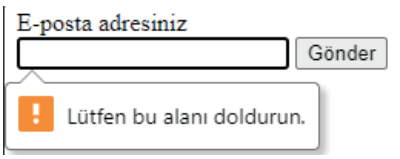
```
<input type="email" name="eposta">
```



Görsel 5.53: Giriş tipinin email olarak değiştirilmesi

4. Adım: Metin kutusunun boş geçilmesini engelleyiniz (Görsel 5.54).

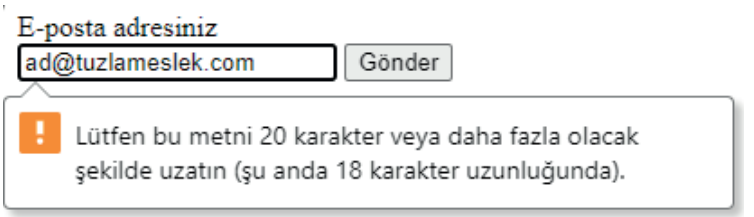
```
<input type="email" name="eposta" required>
```



Görsel 5.54: Girişin zorunlu olması

5. Adım: Metin kutusunda boyut sınırı belirleyiniz (Görsel 5.55).

```
<input type="email" name="eposta" required minlength="20" maxlength="100">
```



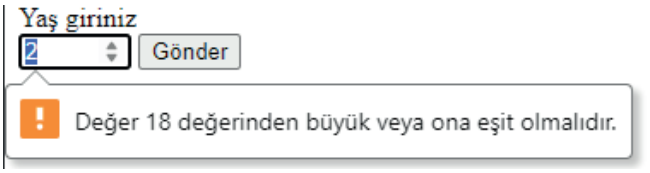
Görsel 5.55: Girişin minimum ve maksimum karakter sayısının belirlenmesi

6. Adım: Metin kutularına önceden girilmiş bilgilerin görülmemesi için otomatik tamamlama özelliği kapatınız.

```
<input type="email" name="eposta" required minlength="20" maxlength="100" autocomplete="off">
```

7. Adım: Sayı ve tarih bilgileri ile ilgili kutucuklara minimum ve maksimum değer sınırı ekleyiniz (Görsel 5.56).

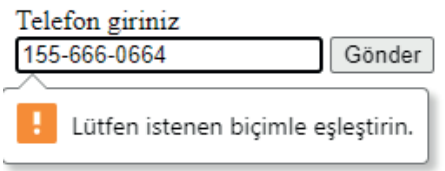
```
<input type="number" name="yas" min="18" max="99">
```



Görsel 5.56: Yaş bilgisinin değer olarak sınırlandırılması

8. Adım: Bir metin kutusuna veri girişini belli bir kalıba (şablona) uyacak şekilde düzenleyiniz (Görsel 5.57).

```
<input type="text" name="telefon" pattern="5[0-9]{2}-[0-9]{3}-[0-9]{4}"
placeholder="5 rakamı ile başlayan telefon">
```



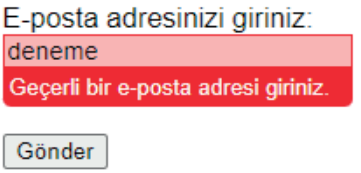
Görsel 5.57: Telefon şablonunun kullanılması

Not

HTML etiketlerinin yetersiz kaldığı yerlerde CSS ve JavaScript kodları ile kullanıcı etkileşimi iyileştirilebilir. Bootstrap, jQuery gibi hazır kütüphaneler ile daha gelişmiş form tasarımları yapılabilir.

Örnek

Masaüstüne yeni bir form2.html sayfası oluşturarak kodu deneyiniz (Görsel 5.58).



Görsel 5.58: Form denetiminin gerçek zamanlı olarak sağlanması

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Form Güvenliği 2</title>
  <style>
    body {
      font: 1em sans-serif;
      width: 200px;
      padding: 0;
      margin: 0 auto;
    }
    p * {
      display: block;
    }
    input[type=email] {
      -webkit-appearance: none;
      appearance: none;
      width: 100%;
      border: 1px solid #333;
      margin: 0;
      font-family: inherit;
      font-size: 90%;
      box-sizing: border-box;
      background-color: rgb(188, 250, 207);
    }
    input:invalid {
      border-color: rgb(255, 0, 0);
      background-color: rgb(250, 188, 188);
    }
    input:focus:invalid {
      outline: none;
    }
    .hata {
      width: 100%;
      padding: 0;
      font-size: 80%;
      color: white;
      background-color: rgb(255, 0, 0);
      border-radius: 0 0 5px 5px;
      box-sizing: border-box;
    }
    .hata.active {
      padding: 0.3em;
    }
  </style>
</head>

<body>
  <form novalidate>
```

```
    <p>
      <label for="eposta"> E-posta adresinizi giriniz:
        <input type="email" id="eposta" name="eposta" required minlength="8" />
        <span class="hata" aria-live="polite"></span>
      </label>
    </p>
    <button>Gönder</button>
  </form>
  <script>
    const form = document.getElementsByTagName('form')[0];
    const email = document.getElementById('eposta');
    const emailError = document.querySelector('#eposta + span.hata');
    email.addEventListener('input', function (event) {
      if (email.validity.valid) {
        emailError.innerHTML = '';
        emailError.className = 'hata';
      } else {
        showError();
      }
    });
    form.addEventListener('submit', function (event) {
      if (!email.validity.valid) {
        showError();
        event.preventDefault();
      }
    });
    function showError() {
      if (email.validity.valueMissing) {
        emailError.textContent = 'E-posta adresi girilmesi zorunludur.';
      } else if (email.validity.typeMismatch) {
        emailError.textContent = 'Geçerli bir e-posta adresi giriniz.';
      } else if (email.validity.tooShort) {
        emailError.textContent = `E-posta en az ${email.minLength} karakter olmalıdır; ${email.value.length}
karakter girdiniz.`;
// “`” karakterini elde etmek için AltGr ile virgül işareti ve ardından boşluk tuşu kullanılır.
      }
      emailError.className = 'hata active';
    }
  </script>
</body>
</html>
```

Notlar

- JavaScript, tarayıcı ayarlarından kapatılabilir. Bu nedenle uygulamanın JavaScript iptal edildiğinde de çalışmaya devam etmesi için önlemler alınmalıdır.
- Kullanıcıya cezalandırıcı hata mesajları vermek yerine ödüllendirici, teşvik edici mesajlarla yönlendirme yapılabilir.
- Gelen verinin veri tabanına yazılmasında ve veri tabanından geri okunmasında temizlik işlemlerinin yapılmasına dikkat edilmelidir.

5.4.2. Sunucu Taraflı Girdi Denetimi

Sunucu taraflı girdi denetimi, sunucunun arka yüzünde **(backend)** kodlama ile ilgilenen geliştiricilerin alabileceği önlemlerdir. Bir başka deyişle web uygulamalarında kullanıcıların sağladığı girdilerin güvenli bir şekilde işlenmesi ve doğrulanması için sunucu tarafında gerçekleştirilen denetimlerdir.

Sunucu taraflı girdi denetiminin doğrulamasında kullanılan yöntemler şunlardır:

- **Veri Türü Doğrulaması:** Sunucu, kullanıcının girdiğini iddia ettiği veri türünü kontrol eder. Örneğin bir sayı bekleniyorsa sunucu, bunun bir sayı olduğunu doğrular ve sayı dışında bir giriş varsa kullanıcıya bir hata mesajı gösterir.
- **Uzunluk Sınırlamaları:** Gelen verilerin belirli bir uzunluk sınırına uyup uymadığını kontrol eder. Şifre veya e-posta adresi alanları için önemlidir.
- **Veri Temizleme ve Filtreleme:** Gelen verileri temizler, zararlı karakterleri veya potansiyel saldırıları filtreler. SQL Injection (aşılama), XSS (cross-site scripting) gibi güvenlik açıklarını önlemek için önemlidir.
- **Veri Biçimi Doğrulama:** Gelen verilerin belirli bir formata uyup uymadığını kontrol eder. Örneğin bir e-posta adresi, telefon numarası, tarih gibi bilgiler belirli bir formata uygun olmalıdır.
- **Veri Tabanı Sorgu Parametreleri:** Veri tabanı sorgularını, kullanıcı girdilerinden oluşan parametrelerle güvenli bir şekilde işlemek için gerekli denetimleri yapar. SQL enjeksiyonu saldırılarını önlemek için önemlidir.

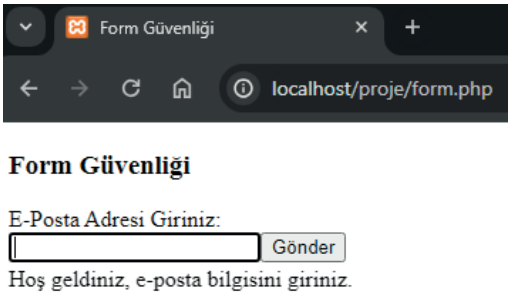
7. Uygulama

İşlem adımlarına göre web programlama dilini kullanarak form denetimlerini hazırlayınız.

1. Adım: C:\xampp\htdocs\proje klasörü içine form.php dosyası ekleyiniz.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Form Güvenliği</title>
</head>
<body>
    <h3>Form Güvenliği</h3>
    <form action="" method="post">
        E-Posta Adresi Giriniz: <br>
    <input type="email" name="eposta" autofocus><input type="submit" value="Gönder">
    </form>
</body>
</html>
```

Not Formun ilk hâlinde HTML etiketi olarak önlem alınmamıştır. E-posta biçimi dışında kullanıcı, kısıtlamasız olarak veri gönderilebilir (Görsel 5.59).



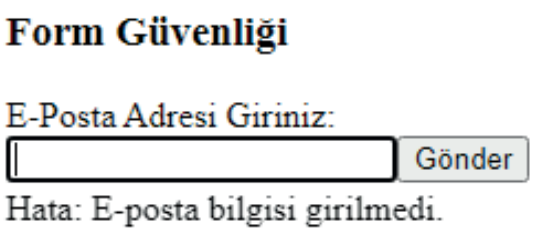
Görsel 5.59: Gönder tuşuna basıldığında veri gönderen form örneği

2. Adım: Gönderilen bilginin PHP kodları ile boş olup olmadığının kontrolünü form etiketinin altına ekleyiniz.

```
<?php
  if($_POST){
     if(empty($_POST["eposta"])){
        echo "Hata: E-posta bilgisi girilmedi.";
     }else{
        echo "E-posta bilgisi girildi.";
     }
  } else {
     echo "Hoş geldiniz, e-posta bilgisini giriniz.";
  }
?>
```

Ekranda şu mesajlardan biri görülür (Görsel 5.60):

- İlk açılışta “Hoş geldiniz, e-posta bilgisini giriniz.”
- Boş girişte “Hata: E-posta bilgisi girilmedi.”
- Normal girişte ise “E-posta bilgisi girildi.”



Görsel 5.60: Form bilgisinin boşluk kontrolü

3. Adım: Gelen bilginin metin boyutlarını kontrol ediniz ve yeni hata mesajına göre denemeler yapınız (Görsel 5.61).

```
<?php
  if($_POST){
     if(empty($_POST["eposta"])){
        echo "Hata: E-posta bilgisi girilmedi.";
     }else{
        if(strlen($_POST["eposta"])<20 || strlen($_POST["eposta"])>150) {
     echo "Hata: E-posta bilgisi uzunluğu 20-150 arasında uygun girilmedi.";
        } else {
        echo "E-posta bilgisi girildi.";
        }
     }
  } else {
     echo "Hoş geldiniz, e-posta bilgisini giriniz.";
  }
?>
```

Görsel 5.61: Form bilgisinin karakter sayısı olarak kontrol edilmesi

4. Adım: E-posta biçiminde giriş yapılıp yapılmadığını kontrol ediniz ve girilen bilgilerin Türkçe karakterler içermesi, @ karakteri eksik olması gibi durumları gözlemlemek için yeni hata mesajına göre deneme yapınız (Görsel 5.62).

```
<?php
   if($_POST){
      if(empty($_POST["eposta"])){
         echo "Hata: E-posta bilgisi girilmedi.";
      }else{
         if (strlen($_POST["eposta"])<20 || strlen($_POST["eposta"])>150) {
    echo "Hata: E-posta bilgisi uzunluğu 20-150 arasında uygun girilmedi.";
         } else {
            if (!filter_var($_POST["eposta"], FILTER_VALIDATE_EMAIL)) {
               echo "Hata: E-posta biçimine göre uygun girilmedi.";
            } else
            echo "E-posta bilgisi girildi.";
         }
      }
   } else {
      echo "Hoş geldiniz, e-posta bilgisini giriniz.";
   }
?>
```

Görsel 5.62: E-posta şablonunun kullanılması

Not Gelen bilgi, veri tabanından karşılaştırılarak önceden kaydolmuş ve e-posta bilgisi onaylanmış bir kullanıcı olup olmadığı kontrol edilmelidir.

5.4.3. Pozitif Girdi Denetimi

Pozitif girdi denetiminde gelen bilgi, olumlu olarak görülen bilgilere göre test edilerek izin verilir. Web uygulamaları tarafından geçerli kabul edilen ve işlenen girdi kalıpları tanımlanırken geçersiz veri girdilerinin bir listesi hazırlanır ve web uygulama, bu belirli girdiler reddedilir. Olası geçerli girdileri tanımlamak veya listelemek, tüm olası geçersiz girdileri tanımlamaktan daha kolay olduğu için izin listesi tercih edilen bir yaklaşımdır. Örneğin il bilgisinin İstanbul, Ankara, İzmir vb. seçilmesi istenirse metin kutusuna serbest bir şekilde veri girilmesi yerine bir açılır listeden (select, drop down list) veri girilmesi tercih edilebilir. Liste kutusu, radyo düğmeleri ve onay kutuları da benzer şekilde kullanıcının belli şeyleri kolay ve doğru seçmesi için kullanılır.

Pozitif girdi denetimi ile ilgili diğer bir örnek ise dosya gönderiminde dosya türünün kontrolüdür. Resim gönderilmesine izin verilecekse sadece **.jpg, .png** ve **.gif** uzantıları olarak kısıtlanabilir, diğer dosyalar zararlı olarak görülebilir. Çalıştırılabilir olduğu için **.aspx, .asp, .css, .swf, .xhtml, .rhtml, .shtml, .jsp, .js, .pl, .php, .cgi** gibi dosyaların gönderimine izin verilmemelidir.

5.4.4. Çıktı Denetimi

Çıktı denetimi, kullanıcı girdisinin istemciye gönderilmeden önce güvenli bir biçime dönüştürülmesi işlemidir. “<”, “>” ve “&” gibi özel karakterlerin kod olarak yorumlanamayacak şekilde kullanılmasını içerir.

Web uygulamanın çıktı doğrulamasının güvenliğini sağlamak için alınabilecek bazı önlemler şunlardır:

- Geliştirme sürecinin bir parçası olarak girdi doğrulama ve çıktı kodlamayı içeren güvenli bir kodlama yöntemi kullanılmalıdır.
- Web uygulamayı olası güvenlik açıklarına karşı taramak için otomatik araçlar kullanılmalıdır.
- Web uygulama, güvenlik açıklarına karşı el ile (manuel) olarak test edilmelidir.
- Web uygulama yazılımı güncel tutulmalıdır.

5.4.5. XSS Denetimi

XSS denetimi için yaygın olarak kullanılan bazı web programlama dili komutları şunlardır:

- **addslashes:** Özel karakterlerin önüne ters eğik çizgi “\” karakteri eklenerek metindeki belirli karakterler, kaçış karakteriyle (escape character) işaretlenir.
- **htmlentities:** HTML etiketleri ve diğer özel karakterler HTML kodlarına dönüştürülerek kullanıcı tarafından gönderilen verilerin güvenli bir şekilde görüntülenmesi sağlanır. Bu sayede kullanıcının gönderdiği verilerdeki HTML etiketleri işlenmez ve sayfada düz metin olarak görüntülenir.
- **htmlspecialchars:** HTML etiketleri özel karakterlerle değiştirilerek kullanıcı tarafından gönderilen verilerin güvenli bir şekilde görüntülenmesi sağlanır. Örneğin “<” karakteri, “<” ile değiştirilir.
- **strip_tags:** Kullanıcı tarafından gönderilen verilerden HTML etiketleri kaldırılarak yalnızca düz metin içeriği bırakılır ancak içerikteki tüm HTML işlevselliği de kaybolur.

- **preg_match:** Veri içinde oluşturulan düzenli ifadeler **(regular expression-regex)** ile kriter veya ifadeye göre ilk eşleşen sonuç döndürülür.
- **filter_input:** Kullanıcı tarafından gönderilen veriler belirli bir filtreyle temizlenir.
- **filter_var:** Belirli bir değişken, belirtilen bir filtreye tabi tutularak temizlenir.
- **trim:** Metnin başındaki ve sonundaki boşlukları silme işlemi yapılır.
- **str_replace:** İstenmeyen karakterleri değiştirme işlemi yapılır.

5.4.6. Kör SQLi (Blind SQLi) Saldırısı

Blind SQL Injection (SQLi), web uygulamasının veri tabanıyla etkileşimde bulunmak için kullanılan bir saldırı yöntemidir. Saldırgan, web uygulamasına SQL sorguları enjekte ederken hedef uygulamadaki veri tabanı üzerinde sorguların sonuçlarını göstermez veya hata mesajlarını üretmez. Blind (kör) olarak adlandırılan bu SQL Injection çeşidi, hedef uygulamadan yanıt almadan veri tabanı üzerinde çalışmayı gerektirir. Bu durum, güvenlik açığından yararlanmayı daha zor hâle getirir ancak yine de veri tabanından veri çıkarmak mümkündür.

Kör SQLi saldırılarının tespit edilmesi çok zor olabilir. Bu saldırılar; bir veri tabanından parolalar, kredi kartı numaraları, diğer kişisel bilgiler gibi hassas verileri çıkarmak için kullanılabilir. Saldırgan, farklı girdiler göndererek ve uygulamanın yanıtını gözlemleyerek bunu kendi avantajına kullanabilir. Örneğin saldırgan, web uygulamanın belirli bir tablo üzerinde COUNT(*) sorgusu çalıştırmasına neden olacak bir girdi gönderebilir.

Kör SQLi saldırılarının iki türü vardır:

- **İkilik (Boolean) Tabanlı Kör SQLi:** Koşulun doğru (true) veya yanlış (false) olmasına bağlı şekilde farklı sonuçlar döndüren sorgular göndererek çalışır. Örneğin saldırgan, veri tabanında admin kullanıcı adı varsa 1, yoksa 0 döndüren bir sorgu gönderebilir.
- **Zaman Tabanlı Kör SQLi:** Koşulun doğru veya yanlış olmasına bağlı şekilde yürütülmesi için farklı süreler alan sorgular göndererek çalışır. Örneğin saldırgan, veri tabanında admin kullanıcı adı varsa yürütülmesi 1 saniye, yoksa yürütülmesi 2 saniye süren bir sorgu gönderebilir.

Kör SQLi saldırıları şu yöntemlerle önlenebilir:

- **Tüm Kullanıcı Girdilerinin Temizlenmesi:** Veri tabanına kötü amaçlı kod enjekte etmek için kullanılabilecek tüm özel karakterleri kaldırmak veya dönüştürmek anlamına gelir.
- **Hazırlanmış İfadelerin (Prepared Statement) Kullanılması:** Hazırlanmış ifadeler, kullanıcı girdisini belirli bir değişkene bağlayarak SQL enjeksiyon saldırılarını önlemenin bir yoludur.
- **Bir Web Uygulaması İçin Güvenlik Duvarı Kullanılması:** Kötü niyetli trafiği engellemeye ve SQL enjeksiyon saldırılarını önlemeye yardımcı olabilir.

Web programlama dili için **Veri Nesneleri (PHP Data Object-PDO)** kullanımı ile veri tabanı işlemleri ve mysqli_real_escape_string komutlarıyla temizlik yapılması, alınabilecek diğer önlemlerdendir.

5.4.7. Oturum Yönetiminin Temel İlkeleri

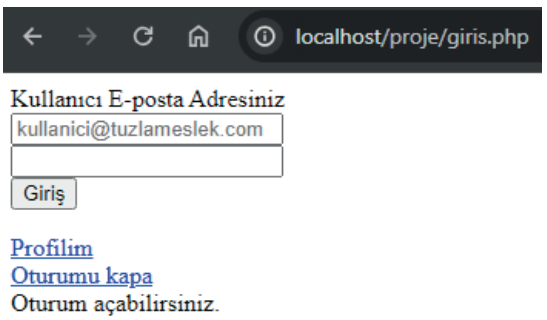
Web uygulamalarında oturum yönetimi ile ilgili kodlar yazılarak oturum yönetiminin temel ilkeleri sağlanır.

8. Uygulama

İşlem adımlarına göre web programlama dilini kullanarak veri tabanı gerekmeden oturum yönetimini gerçekleştiriniz.

1. Adım: **C:\xampp\htdocs\proje** klasöründe çalışınız ve oturum açmak için **giris.php** dosyasını oluşturunuz (Görsel 5.63).

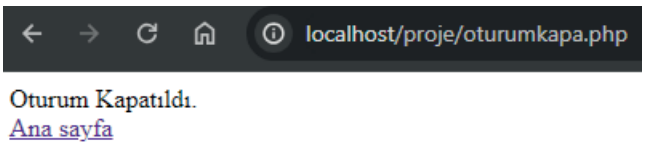
```
<?php
  session_start();
  //Oturum desteği sağlamak için bu satır, her sayfanın üstünde olmalıdır.
 ?>
<form action="" method="post">
  Kullanıcı E-posta Adresiniz <br>
  <input type="email" name="eposta" placeholder="kullanici@tuzlameslek.com" required minlength="8"><br>
  <input type="password" name="parola" required minlength="5"><br>
  <input type="submit" value="Giriş">
</form>
<a href="profil.php"> Profilim</a> <br>
<a href="oturumkapa.php"> Oturumu kapa</a> <br>
<?php
  if($_POST){
     if($_POST["eposta"]=="kullanici@tuzlameslek.com" && $_POST["parola"]=="abc+123"){
        $_SESSION["eposta"] = $_POST["eposta"];
        $_SESSION["oturumAcik"] = "evet";
        echo $_POST["eposta"]." için oturum açıldı.";
        echo "<br> Profilinizi inceleyebilirsiniz.";
     }
  }
  if(isset($_SESSION["eposta"])){
    echo "Hoş geldiniz, oturumunuz açıktır." ;
  } else
  echo "Oturum açabilirsiniz."
?>
```



Görsel 5.63: Oturum açma sayfası

2. Adım: Aynı klasör içine **oturumukapa.php** kodlarını yazınız (Görsel 5.64).

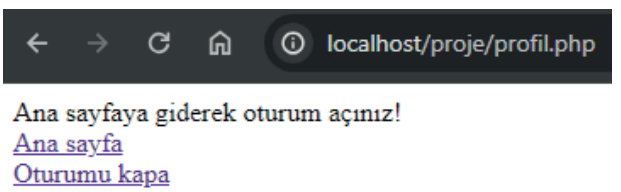
```
<?php
    session_start();
    session_destroy();
?>

Oturum Kapatıldı. <br>
<a href="giris.php"> Anasayfa</a>
```



Görsel 5.64: Oturum kapatma sayfası

3. Adım: Aynı klasör içine **profil.php** kodlarını yazınız (Görsel 5.65).

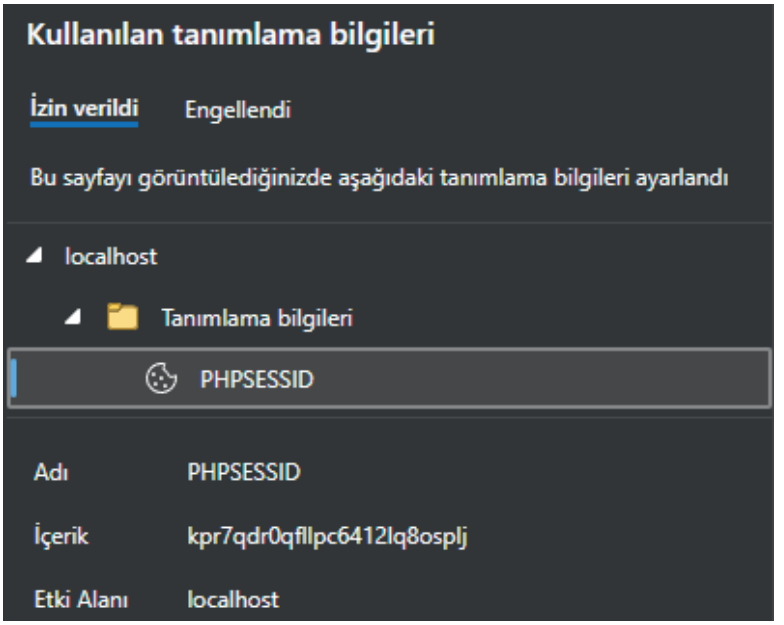
```
<?php
    session_start();
if (isset($_SESSION["oturumAcik"]) && $_SESSION["oturumAcik"]=="evet") {
        echo "Hoş geldiniz ".$_SESSION["eposta"];
    } else {
        echo "Anasayfaya giderek oturum açınız!";
    }
?>
<br>
<a href="giris.php"> Anasayfa</a> <br>
<a href="oturumkapa.php"> Oturumu kapa</a>
```



Görsel 5.65: Profil sayfası

Oturum açmak için kullanici@tuzlameslek.com ve parola olarak abc+123 ile giriş yapılabilir. Temel oturum işlevleri çalıştırılabilir, giriş ve çıkış yapabilir, profil sayfasına girilebilir. Veri kısıtlaması olarak önlem alınmamıştır.

Adres çubuğunda adresin sol tarafındaki **kilit** simgesinden Tanımlama bilgileri tıklandığında **PHPSESSID** isminde çerez görülebilir, bu çerez engellenebilir veya kaldırılabilir (Görsel 5.66).

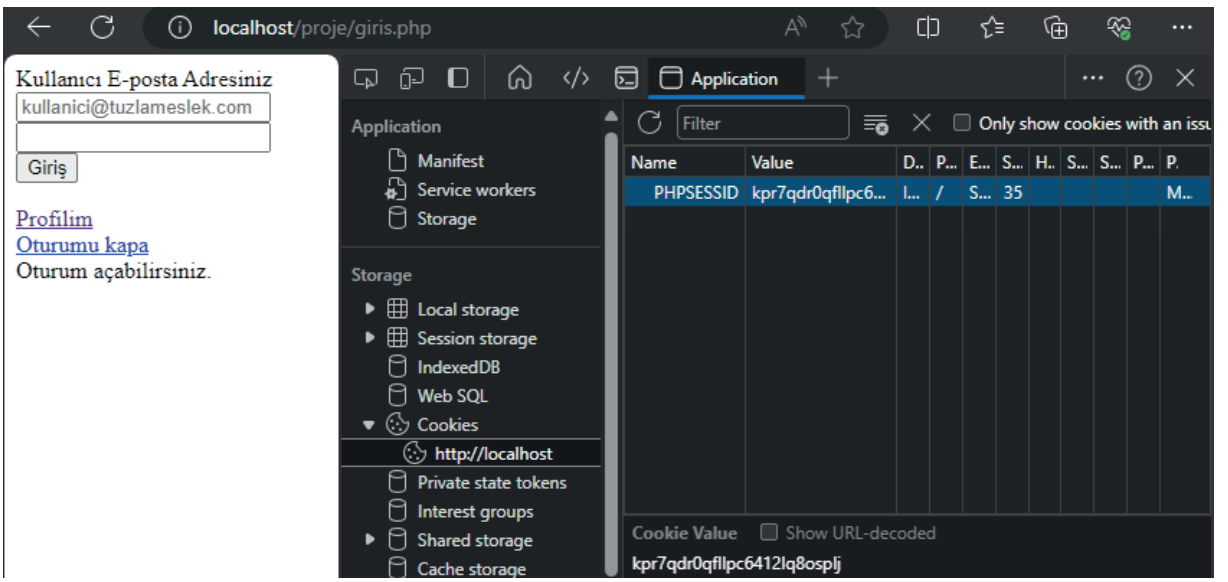


Görsel 5.66: Tanımlama bilgilerinin görülmesi

Not

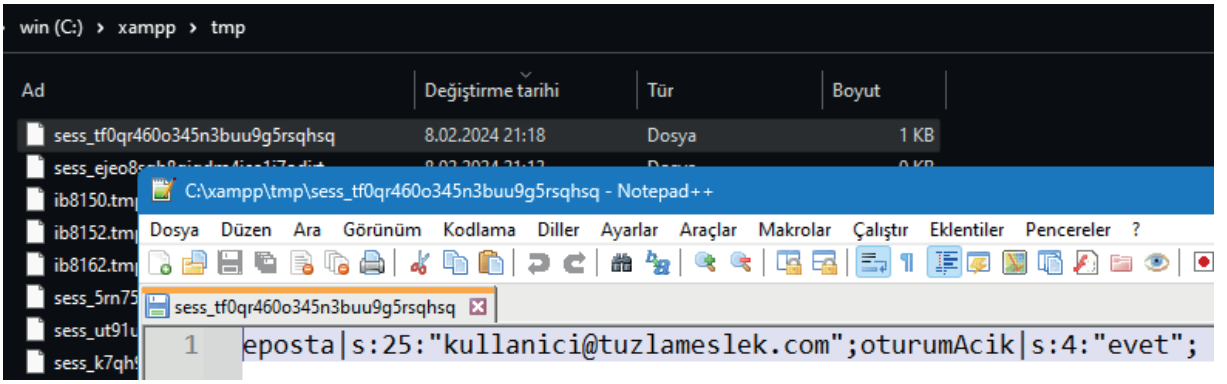
Cookie (Çerez), tarayıcının kullanıcı cihazında depoladığı küçük metin dosyalarıdır. Web sitesi tarafından kullanıcıya ait bazı bilgileri kaydetmek veya kullanıcının etkileşimlerini takip etmek için kullanılır. Genellikle oturum durumunu korumak, tercihleri hatırlamak, kullanıcı deneyimini kişiselleştirmek, reklamları hedeflemek gibi amaçlarla kullanılır.

Oturum kimlik bilgisi (session id), **F12** kısayoluyla açılan Geliştirici araçları menüsünden öğrenilebilir, silinebilir ve değiştirilebilir (Görsel 5.67).



Görsel 5.67: Geliştirici araçları ile oturum bilgilerinin görüntülenmesi

Geliştirici araçlarından PHPSESSID'nin değeri değiştirildiğinde kullanıcının girdiği sayfadaki oturum ile sunucudaki oturum arasındaki eşleşme bozulur. Web programlama dilinde oturumlar, **C:\xampp\tmp** içinde dosya olarak saklanır. Oturum dosyası, not defteri ile açıldığında proje içinde kullanılan değerler görülebilir (Görsel 5.68).

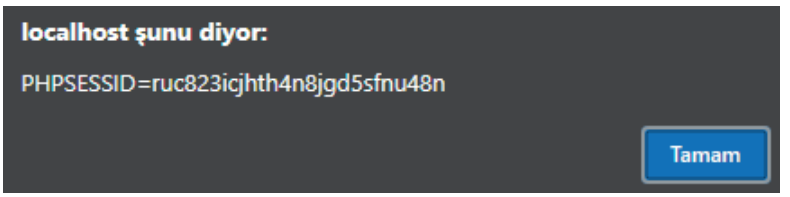


Görsel 5.68: Oturum bilgilerinin not defteri ile açılması

4. Adım: Oturum kimliğini alan kötü niyetli biri, kullanıcının parolasını bilmeden, **sahte oturum** açarak profil bilgilerine erişebilir. **sess_** kısaltması ile başlayan dosya içindeki bilgileri diğer bir tarayıcı oturumunda kullanarak deneyiniz ve kullanıcı adı ile parolasını girmeden **profil.php** sayfasının açılmasını sağlayınız.

5. Adım: Önlem almadan önce JavaScript kodunu **giris.php** sayfasının sonuna ekleyiniz. Görsel 5.69'da görüldüğü gibi sayfa açılışında ekrana mesaj olarak oturum bilgisi gelir.

```
<script>
alert(document.cookie);
</script>
```



Görsel 5.69: JavaScript ile oturum bilgisinin görüntülenmesi

6. Adım: Oturumu daha güvenli hâle getirmek için önlem alınız. Giriş için kritik olan giris.php sayfasının kodunu düzenleyiniz.

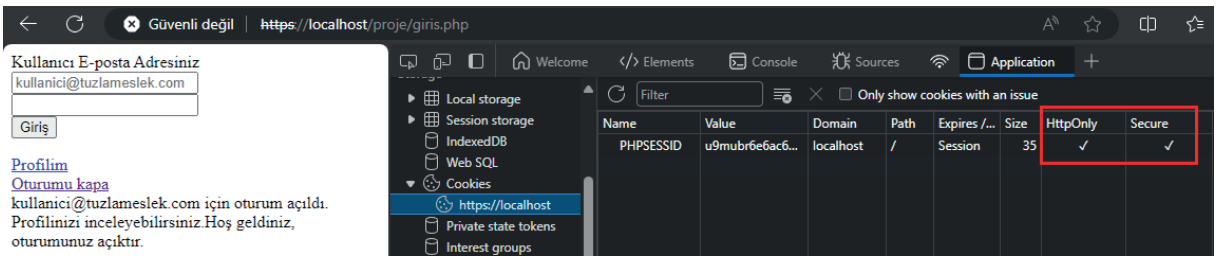
```
<?php
function sessionBaslat($bayatlamaZamani, $yol, $alanAdi, $guvenli, $sadeceHttp){
 session_set_cookie_params($bayatlamaZamani, $yol, $alanAdi, $guvenli, $sadeceHttp);
 session_start();
}
    sessionBaslat(0, '/', 'localhost', true, true);
    //session_start();
    //Oturum desteği sağlamak için bu satır, her sayfanın üstünde olmalıdır.
?>
```

```
<form action="" method="post">
  Kullanıcı E-posta Adresiniz <br>
  <input type="email" name="eposta" placeholder="kullanici@tuzlameslek.com" required minlength="8"><br>
  <input type="password" name="parola" required minlength="5"><br>
  <input type="submit" value="Giriş">
</form>
<a href="profil.php"> Profilim</a> <br>
<a href="oturumkapa.php"> Oturumu kapa</a> <br>
<?php
  if($_POST){
if($_POST["eposta"]=="kullanici@tuzlameslek.com" && $_POST["parola"]=="abc+123"){
        session_regenerate_id(true);
        //Her oturum açılmasında ID bilgisi değişir.
        $_SESSION["eposta"] = $_POST["eposta"];
        $_SESSION["oturumAcik"] = "evet";
        echo $_POST["eposta"]." için oturum açıldı.";
        echo "<br> Profilinizi inceleyebilirsiniz.";
      }
  }
  if(isset($_SESSION["eposta"])){
    echo "Hoş geldiniz, oturumunuz açıktır." ;
  } else
  echo "Oturum açabilirsiniz."
?>
<script>
alert(document.cookie);
</script>
```

Görsel 5.70'te görüldüğü gibi PHPSESSID bilgisinin JavaScript ile okunması engellenmiştir.

Görsel 5.70: JavaScript ile oturum bilgilerinin görüntülenmesinin iptali

HttpOnly ve **Secure** onayları, **https://localhost/proje/giris.php** olarak girildiğinde görülür (Görsel 5.71).



Görsel 5.71: HTTPS ile güvenli oturum açılması

Sıra Sizde

Web programlama dilini kullanarak oturum yönetimini gerçekleştiriniz. Oturumda oluşan hataları not ediniz.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. C:\xampp\htdocs\ çalışma klasörüne giris.php dosyasını oluşturdu.		
2. Oturum kapanması ile ilgili oturumukapa.php dosyasına kodları ekledi.		
3. Oturum açıldığının test edilmesi ile ilgili profil.php dosyasına kodları ekledi.		
4. Giriş sayfasına kullanıcı adını ve parolasını girmeden profil.php sayfasının açılmasını sağladı.		
5. Oturumu daha güvenli hâle getirmek için eksiklikleri gidererek önlemler aldı.		
6. Oluşan güvenlik sorunlarının açıklamasını yaptı.		
"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.		

Not

Oturum yönetiminde dikkat edilmesi gerekenler şu şekilde sıralanabilir:

- Uygulama, oturum tekil tanımlayıcısını (Session ID) korumalıdır.
- Oturum tekil tanımlayıcısı (Session ID) URL'de gönderilmemeli veya başlık bilgisi içine dâhil edilmemelidir.
- Oturum yönetimi için kullanılan ve uygulamayı kullanan bütün kullanıcılar için tekil olması gereken değerler (Session ID, token v.b.) güçlü bir rastgele veri üretecinden temin edilmeli ve tahmin edilemez derecede karmaşık olmalıdır
- Belirlenen süre boyunca aktif olmayan oturumlar otomatik olarak kapatılmalıdır. Uygulama, sistem yöneticilerine bu süreyi ayarlayabilecekleri bir ekran sunmalıdır.

https://hgm.uab.gov.tr/uploads/pages/siber-guvenlik/gygtk-doc.pdf

ÖLÇME VE DEĞERLENDİRME

A. Aşağıdaki cümlelerin başındaki paranteze verilen bilgiler doğruysa "D", yanlışsa "Y" yazınız.

1. ( ) Ön yüz ve arka yüz tasarımını bir arada yapabilen geliştiricilere proje yöneticisi denir.

2. ( ) ASP.NET ve PHP uzantılı dosyalar tarayıcıda doğrudan görüntülenebilir.

3. ( ) Sorgu sonucunun görülemediği saldırı türüne kör SQLi denir.

B. Aşağıdaki cümlelerde boş bırakılan yerlere doğru sözcükleri yazınız.

4. Bir web uygulamasında istemci olan bilgisayarda saklanması gerekli veri dosyasına ..... denir.

5. Uygulama tarafında geçerli olarak görülen girdilere ..... denir.

6. Sayfa bulunamadığı zaman görülen hata koduna ..... denir.

C. Aşağıdaki soruları dikkatlice okuyarak doğru cevabı işaretleyiniz.

7. "Hesabınız askıya alındı." şeklinde bir e-posta alındığında bu saldırının, aşağıdaki saldırı türlerinden hangisinin olma olasılığı daha yüksektir?

A) Fidye yazılımı
B) Kaba kuvvet
C) Kimlik hırsızlığı
D) Kötü amaçlı yazılım
E) Oltalama

8. Aşağıdakilerden hangisi herhangi bir saldırı öncesi alınabilecek en doğru önlemdir?

A) Çalışanlara eğitim vermek.
B) Hasar kayıtlarını incelemek.
C) Sistem günlüklerini incelemek.
D) Veri tabanının yedeğini almak.
E) Yazılımları güncellemek.

9. Aşağıdakilerden hangisi kod editörü olarak kullanılamaz?

A) Dreamweaver
B) Notepad++
C) .NET Core
D) Visual Studio Code
E) WordPad

10. Aşağıdakilerden hangisi veri gönderme yöntemlerinden değildir?

A) DATA B) DELETE C) GET
D) PATCH E) POST

11. Aşağıdakilerden hangisi .NET web uygulaması geliştirmek için gerekli değildir?

A) C# B) CSS C) JavaScript
D) SQL E) VUE.JS

12. Aşağıdakilerden hangisi uygulama geliştirmenin bir aşaması değildir?

A) Girilen bilgilere güvenmek.
B) Girişleri test etmek.
C) Kod yazmak.
D) Uygulamayı yayımlamak.
E) Yenilikler planlamak.

13. Aşağıdakilerden hangisinin çerez dosyasında metin olarak saklanması tehlikelidir?

A) E-posta adresi
B) Konum bilgisi
C) Kullanıcı adı
D) Parola bilgisi
E) Yaş bilgisi

14. Aşağıdakilerden hangisi sunucuya gönderilen dosyalarda izin verilen dosya türüdür?

A) .css B) .jpg C) .js
D) .php E) .pl

15. Aşağıdaki form bilgilerinden hangisinin kontrol edilmesi zorunlu değildir?

A) Bilgi uzunluğu
B) DROP, DELETE gibi kelimeler
C) @, . gibi karakterler
D) SCRIPT, IMG gibi kelimeler
E) < > gibi karakterler

D. Aşağıdaki soruların cevabını ilgili alana yazınız.

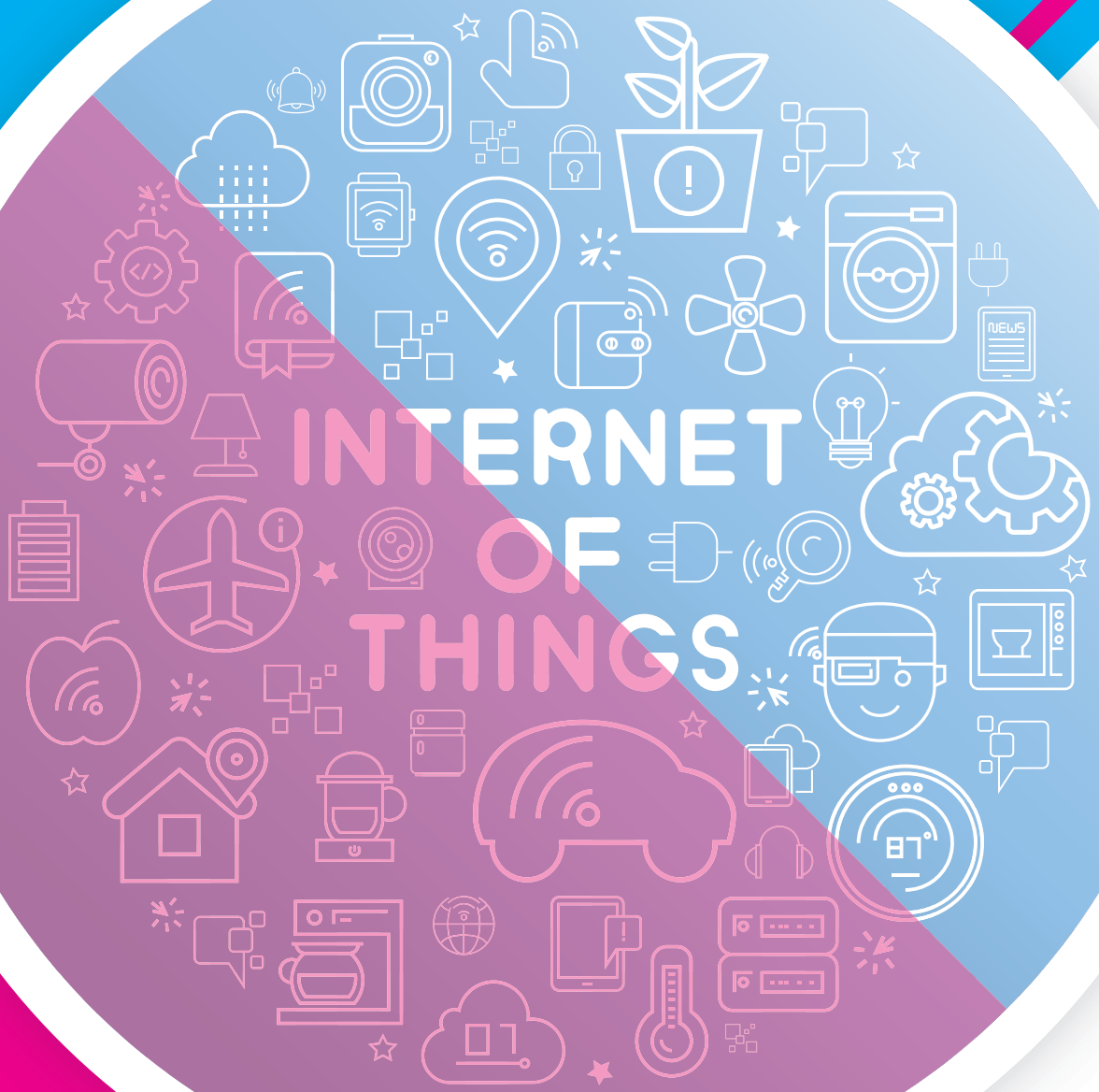
16. Sunucu ve istemci tarafında güncel olarak en fazla kullanılan web geliştirme teknolojilerinin neler olduğunu yazınız.

17. Web uygulamalarında en sık oluşan güvenlik problemlerini ve çözüm önerilerini yazınız.

18. Web uygulamasının kullanım ve güvenlik kalitesini artırmak için alınabilecek önlemlerin neler olduğunu yazınız.

19. Web uygulaması yayımlandıktan sonra uygulamada oluşabilecek istenmeyen durumların tespit edilmesi ve giderilmesi ile ilgili çalışmaların neler olduğunu yazınız.

IoT GÜVENLİĞİ



6. ÖĞRENME BİRİMİ

KONULAR

6.1. NESNELERİN İNTERNETİ (IoT) MİMARİSİ

6.2. IoT ZAFİYET VE TEHDİTLERİ

6.3. IoT GÜVENLİK TEDBİRLERİ

6.4. IoT GÜVENLİK TESTİ

ANAHTAR KELİMELER

Akıllı şehirler, güvenlik politikaları, IoT ağları, IoT güvenliği, IoT mimarileri, IoT protokolleri, Nesnelerin İnterneti (IoT), sensör ağlar

NELER ÖĞRENECEKSİNİZ?

- Nesnelerin İnterneti (IoT) kavramını açıklama
- Üç katmanlı IoT mimarisini açıklama
- Beş katmanlı IoT mimarisini açıklama
- IoT referans mimarisini açıklama
- IoT zafiyetlerini tespit ederek tehditleri açıklama
- IoT güvenlik tedbirlerini açıklama
- IoT güvenlik testlerini uygulayarak açıklama
- IoT donanım güvenlik testlerini gerçekleştirme

HAZIRLIK ÇALIŞMALARI

1. Akıllı cihazlar ve akıllı şehirler kavramlarının size çağrıştırdıklarını arkadaşlarınızla paylaşınız.
2. Referans mimarileri neden tercih edilir? Düşüncelerinizi arkadaşlarınızla paylaşınız.

6.1. NESNELERİN İNTERNETİ (IoT) MİMARİSİ

Nesnelerin İnterneti (Internet of Things-IoT), otomatik olarak görevlerini yerine getiren ve bir ağ üzerinden fiziksel şekilde birbirine bağlanan cihazların oluşturduğu ağ olarak tanımlanır. Birlikte çalışma yeteneğine sahip her cihaz, bu yapının bir ögesidir. Bu cihazlar, evde kullanılan sıradan bir nesne olabildiği gibi endüstriyel üretimde kullanılan bir üretim nesnesi de olabilir. IoT, günümüzün en önemli teknolojilerinin biridir. Her geçen gün bir ağa bağlanma yeteneği olan nesnelerin (things) sayısı artar ve bu nesneler, günlük hayatın bir parçası hâline gelir. Otomobiller, mutfak araç gereci, buzdolabı, kombi, akıllı TV'ler gibi günlük hayatın bir parçası olan nesneler internete bağlanarak birçok bilgiyi gerçek zamanlı şekilde paylaşabilir. Bu nesnelerden alınan bilgiler; büyük veri, bulut teknolojileri ve mobil teknolojilerle toplanarak analiz edilebilir. Dünyanın farklı noktalarında bulunan milyarlarca cihazdan elde edilen verilerin büyüklüğü düşünüldüğünde IoT ağ ve teknolojilerinin ne kadar önemli olduğu ve güvenliğinin sağlanmasında yaşanabilecek zorluklar tahmin edilebilir.

IoT fikri bazı temel sensörlerin tasarımı ve haberleşme protokolleri ile 1980'lerde ortaya çıkmasına rağmen bu teknolojinin gelişimi 2000'li yıllardan sonra hızlanmıştır. IoT örneklerinden ilki, 1980'lerde Carnegie Mellon Üniversitesi'nde kullanılan içecek otomat makinesidir. İçecek otomat makinesi, bir bilgisayar ağını kullanarak makinenin içinde içecek olup olmadığını, makinede içecek varsa bu içeceğin soğuk olup olmadığını kontrol etmiştir. Bu yıllarda geliştirilen çiplerin boyutunun büyüklüğü ve yüksek maliyet gibi sebepler, IoT teknolojilerinin gelişmesini olumsuz etkilemiştir. 1991 yılında Cambridge Üniversitesi'nde akademisyenlerin ortak kullandıkları kahve makinesini görebilmek için kurdukları kamera sistemi ilk IoT uygulaması olarak kabul edilir. Bu uygulamada internet bağlantısı yoktur sadece gerçek zamanlı haberleşme özelliği bulunur. İnternet ağlarının genişlemesi ve hız verilerinin artması, bu alanı popüler hâle getirmiştir. 1999 yılında Kevin Ashton tarafından "Nesnelerin İnterneti" kavramı literatüre eklenmiş, Radio-Frequency Identification (RFID) gibi teknolojilerin gelişimiyle de "Nesnelerin İnterneti"nin yaygınlaşması hızlanmış ve 2005 yılında Uluslararası Telekomünikasyon Birliği (ITU) tarafından hazırlanan raporlarda popülerliği giderek artmıştır.

IoT ekosistemi yapısal olarak akıllı cihazlardan oluşur. Bu akıllı cihazlar; ağa bağlanma, veri toplama, veri gönderme, veri işleme gibi temel görevleri yerine getirir. Cihazların sahip olduğu kullanıcı arayüzleri ile temel konfigürasyonları yapılır. Bu yeteneklere sahip cihazlar, diğer akıllı cihazlarla iletişim kurar ve topladığı verilere göre hareket eder. Bu durumun temelinde kullanıcı müdahalesine ihtiyaç duymadan çalışan akıllı cihazların oluşturduğu bir ekosistem yer alır. Kullanıcılar, gerekli durumlarda sistemle etkileşime de geçebilir. Bu durum, ekosistemin çalışması noktasında bir problem oluşturmaz. IoT ekosistemindeki veri toplama katmanında algılayıcılar (sensörler) aracılığıyla veri toplanır. Bu veriler, denetleyici ve diğer gömülü sistemler yardımıyla anlamlı hâle getirilir.

Ağ ve güvenlik katmanında toplanan verilerin diğer sensör ağlarına iletilmesi sağlanır (Görsel 6.1).

Görsel 6.1: IoT temel yapısı

IoT ekosisteminde katman boyunca güvenliğin sağlanması önemlidir. Toplanan verilerin kullanılabilirliği analiz katmanında test edilir. Günümüzde ekosistem; yapay zekâ teknolojileri, bulut teknolojileri, bağlantı protokolleri, yeni nesil algılayıcılar, sürekli online (çevrimiçi) ve boyut olarak daha küçük, daha ucuz cihazlarla desteklenir. IoT teknolojilerinin gelişmesi ve büyümesi noktasında düşük maliyeti olan algılayıcıların tasarımı, bulut bilişim teknolojilerinin gelişmesi, bağlantı protokollerinin çeşitlendirilmesi, yapay zekâ veya makine öğrenmesi yaklaşımları etkili olmuştur.

Günümüzde yaklaşık 22 milyar olan IoT cihaz sayısının önümüzdeki birkaç yıl içinde 50 milyara ulaşması beklenmektedir. IoT teknolojileri yapısal olarak endüstriyel üretimle de doğrudan ilişkilidir. Bu nedenle Endüstri 4.0 kavramının çıkış noktasında önemli bir yere sahiptir. Günlük hayattaki yaygın uygulama alanları, yaşamı kolaylaştıran birçok teknolojiyi de beraberinde getirir. Günümüzde IoT uygulamalarının yaygın olarak kullanıldığı alanlar şunlardır:

- Akıllı şehir uygulamaları
- Ulaşım ve otomotiv teknolojileri uygulamaları
- Tarım uygulamaları
- Akıllı bina ve ev uygulamaları
- Askerî alandaki uygulamalar
- Sağlık alanındaki uygulamalar

Akıllı şehir; insanlara sürdürülebilir bir yaşam ortamı sunan, modern, çevre dostu ve ileri teknolojilerle donatılmış şehir olarak tanımlanır. Akıllı şehirler, IoT sayesinde algılayıcılardan toplanan verilerle yönetim ve kaynakların verimli kullanılması noktasında tasarlanan alanlardır. Etkili toplu taşıma sistemlerinin tasarlanması, sürdürülebilir bir veri akış kontrolü, su ve enerji yönetimi, atık yönetimi gibi birçok alanda planlı bir çevre yönetim anlayışını kapsar. Bu sürecin yönetilmesi, IoT teknolojileri sayesinde yapılır. Trafik kontrollerinin yapılmasında, akıllı park sistemlerinin kurulumunda, akıllı aydınlatma sistemleri kullanılmasında, hava veya su kirliliğinin ölçülmesi ve ölçümlerin takip edilmesinde, güvenliğin sağlanmasında IoT teknolojilerinden etkin şekilde yararlanılır. Dünyada New York, Amsterdam, Hong Kong, Singapur, Moskova, Barselona ve Londra akıllı şehirlere örnek olarak verilebilir. Ülkemizdeki büyükşehirlerde de akıllı şehir uygulama girişimleri artmıştır.

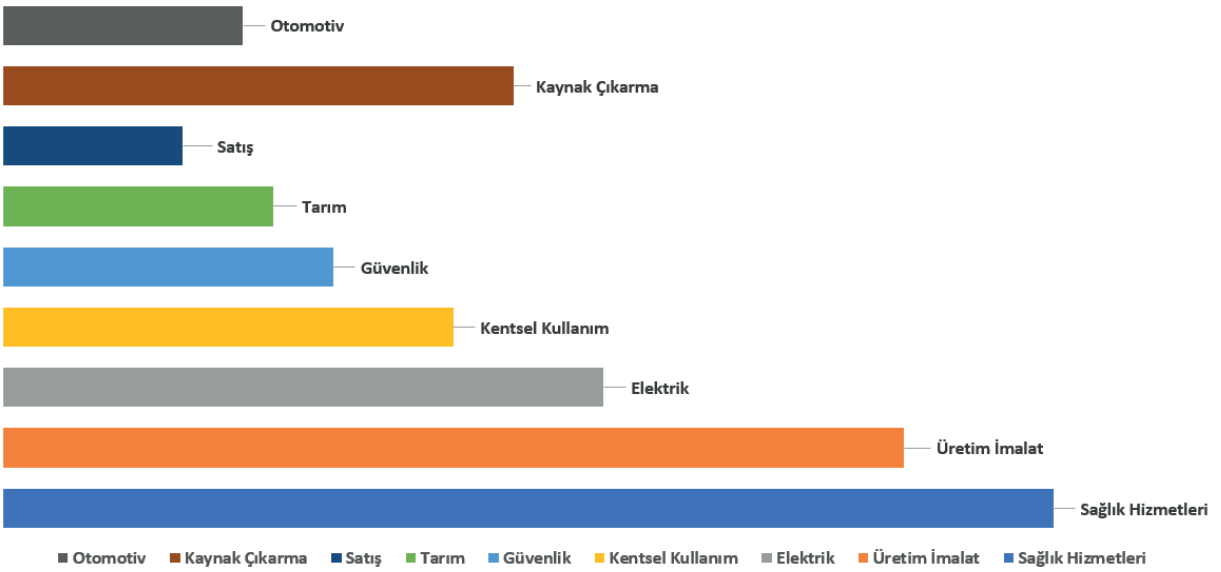
Ulaşım ve otomotiv teknolojileri alanındaki uygulamalarda rota oluşturma ve rotanın yönetimi, yakıt kontrolü, yolcuların güvenliğinin sağlanması ve takibinde IoT teknolojilerinden faydalanılır.

Tarım uygulamalarında etkili ve verimli üretimle ilgili olarak hayvancılık ve hasat takibi yapılır. Algılayıcılardan toplanan nem, sıcaklık gibi verilerle en uygun üretim ortamı sağlanarak yüksek verimlilik hedeflenir. Takılan algılayıcılarla hayvanların takibi yapılır.

Akıllı bina ve evlere yerleştirilen algılayıcılar ile ısı, ışık gibi veriler alınarak güvenlikten enerji verimliliğine kadar birçok alanda veriler toplanır.

Askerî alanda lojistik hizmetlerinin sağlıklı bir şekilde yönetilmesi, sınır noktalarında güvenliğin sağlanması, saldırı ve hedef tespitlerinin yapılmasında IoT teknolojilerinden yararlanılır.

Sağlık alanında hastanelerdeki verilerin yönetilerek ilaç ve hasta takibinin yapılmasında da IoT verileri kullanılır (Görsel 6.2).



Görsel 6.2: 2025 IoT uygulamalarının sektörlere göre tahmini dağılımı

Kullanım alanlarına göre IoT incelendiğinde yaşamı kolaylaştırarak birçok avantaj sağladığı görülür. İş hayatında sürecin izlenmesi, zaman ve paradan tasarruf, doğru ve hızlı karar verme sürecinin yönetilmesi, çalışanların daha verimli şekilde iş hayatında yer alması gibi faydalar sağlanır. Fonksiyonel üretim süreci sayesinde iş gücü azaltılarak verimliliğin artırılmasına, kaynakların daha etkin kullanılmasına imkân tanınır.

IoT teknolojilerinin dezavantajlarının başında güvenlik ve gizlilik problemleri gelir. Algılayıcıların düşük güvenlik mimarileri, verilerin manipüle edilip bütünlüğünün bozulmasına sebep olur. Bu noktada IoT güvenlik yaklaşımlarının önemi ortaya çıkar. Endüstriyel alanda kullanılan IoT sayesinde otomasyonel üretim süreci iş kayıplarına yol açar.

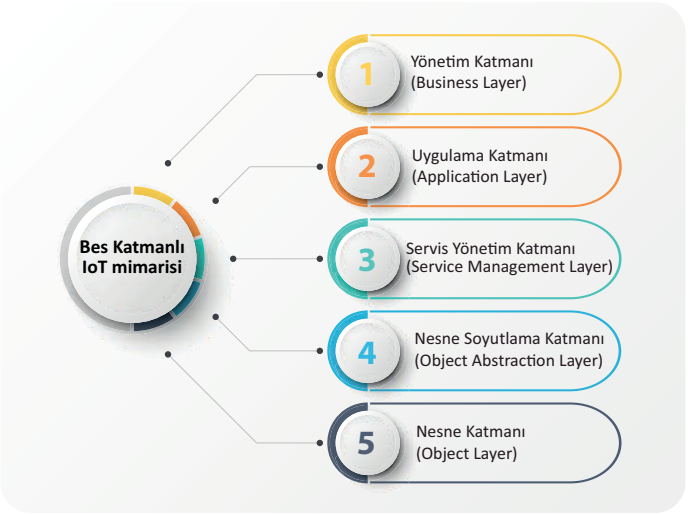
6.1.1. Nesnelerin İnterneti Referans Mimarisi Modeli ve Güvenlik Yaklaşımları

IoT ekosistemlerinin yaygınlaşması ile ortaya çıkan platformlar, IoT cihazlarındaki güvenlik gereksinim ihtiyaçlarını artırmıştır. Sensörler ve cihazlar birçok nesneden veri toplayarak ekosistemine bağlı diğer cihazlara verilerini aktarır. Yapılan veri alışverişinde IoT cihazları dışındaki aygıtlarla uyumlu şekilde

çalışması gerekir. Bu özellikleri IoT cihazlarının desteklemesi gerekir. ISO (Uluslararası Standardizasyon Örgütü) ve IEC (Uluslararası Elektroteknik Komisyonu) tarafından ISO/IEC 30141:2018 ile veri alışverişi için standartlaştırılmış bir IoT referans mimarisi oluşturulmuştur. Bu modelde IoT sistemlerinin özellik tanımları ile kavramsal bir model önerilmiştir. IoT cihazlarının gerçek zamanlı olarak verileri kullanması, verilerin işlenmesi ve analiz edilmesi aşamalardaki güvenlik gereksinimleri için katman mimari yaklaşımları ortaya çıkmıştır. Güvenlik gereksinimlerinin belirlenmesi, veri güvenliğinin ve bütünlüğünün sağlanması için IoT teknolojilerinde farklı katman mimarileri önerilmiştir..

6.1.1.1. Beş Katmanlı IoT Güvenlik Referans Mimarisi

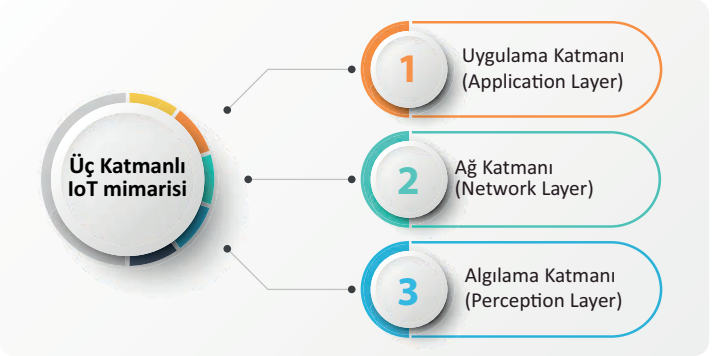
Temelde algılayıcılar, ağ ve güvenlikle ilgili analiz katmanlarından oluşan mimaride International Telecommunication Union (ITU) tarafından beş katmanlı IoT mimari modeli önerilmiştir. Bu mimaride sürecin izlenmesi ve yönetilmesi için Yönetim Katmanı (Business Layer) bulunur. Mimari yaklaşımlar arasında tercih edilen yaklaşım, beş katmanlı IoT mimarisidir (Görsel 6.3).



Görsel 6.3: Beş katmanlı IoT mimarisi

6.1.1.2. Üç Katmanlı IoT Güvenlik Referans Mimarisi

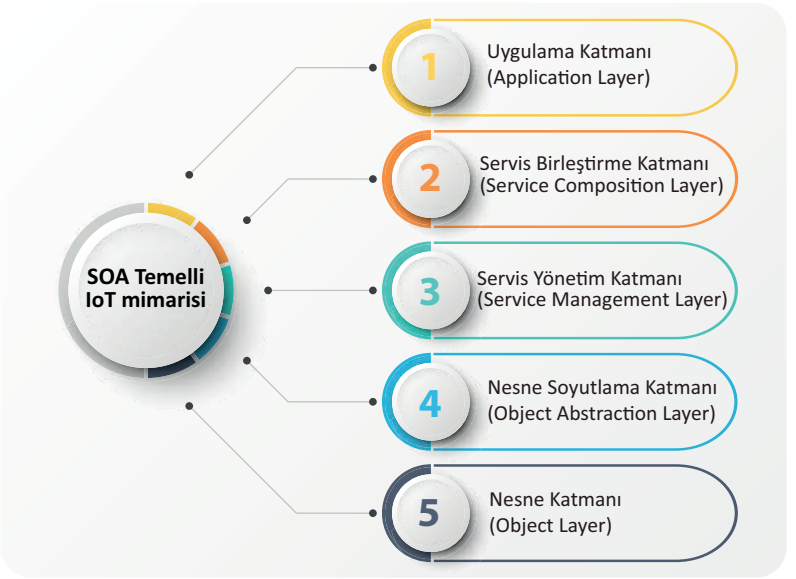
Üç katmanlı IoT mimarisi; algılama, ağ ve uygulama katmanlarından oluşur (Görsel 6.4.). Veri toplama amaçlı kullanıldığı için algılama katmanı, IoT teknolojilerinde duyu organı gibidir. RFID gibi iletişim protokolleri burada aktif olarak kullanılır. Ağ katmanı, Internet Protocol (IP) ve Internet Control Message Protocol (ICMP) şeklinde iletişim protokollerinden oluşur. Constrained Application Protocol (CoAP) ise verilerin gözlemlenebildiği uygulama katmanında kullanılır.



Görsel 6.4: Üç katmanlı IoT mimarisi

6.1.1.3. Simple Object Access Protocol (SOA) Güvenlik Referans Mimarisi

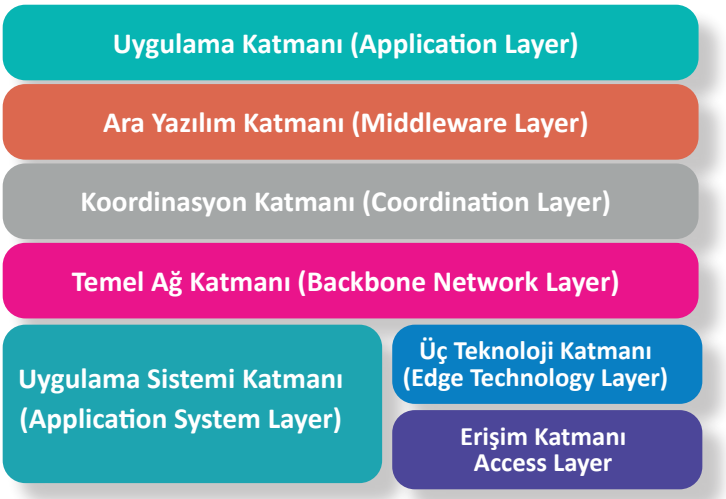
Servis odaklı IoT mimarisi beş katmandan oluşur. Nesne katmanı, verilerin toplanarak tahminlerin yapıldığı katmandır (Görsel 6.5). Analog verilerin dijital verilere çevrilmesi burada gerçekleştirilir. Büyük veriler, nesne katmanında oluşmaya başlar. Kablosuz bağlantı alanı (Wi-Fi), ZigBee gibi protokollerle verilerin dönüşümü nesne soyutlama katmanında sağlanır. Nesne soyutlama katmanında bulut bilişim ve veri yönetimi yapılır. Servis yönetim katmanında veriler alınır ve işlenir. İşlenen veriler, servis birleştirme katmanına iletilir. Servis birleştirme katmanında toplu olarak hizmet sunumu yapılır. İşlenen veriler, uygulama katmanı sayesinde kullanıcıya sunulur.



Görsel 6.5: SOA temelli IoT mimarisi

6.1.1.4. Middleware Temelli Güvenlik Referans Mimarisi

Ara yazılım temelli IoT mimarisi 2010 yılında Neng Wang ve Lu Tan tarafından geliştirilmiştir. Yedi katmandan oluşur. Fiziksel katman içinde tekil uygulama, erişim ve uç teknoloji katmanı bulunur. Temel ağ katmanında veri iletimi yapılır. Koordinasyon katmanında ağ katmanından gelen veriler anlamlı hâle getirilir. Ara yazılım katmanında yeni eklenen nesnelerin entegrasyonu sağlanır. Uygulama katmanı, verilerin kullanıcı tarafından kontrol edilmesinde etkindir (Görsel 6.6).



Görsel 6.6: Middleware temelli IoT mimarisi

Tablo 6.1'de IoT katman mimarilerinin gizlilik veya güvenlik, dinamiklik, akıllı kontrol ve birlikte çalışma özelliklerine göre karşılaştırılması verilmiştir.

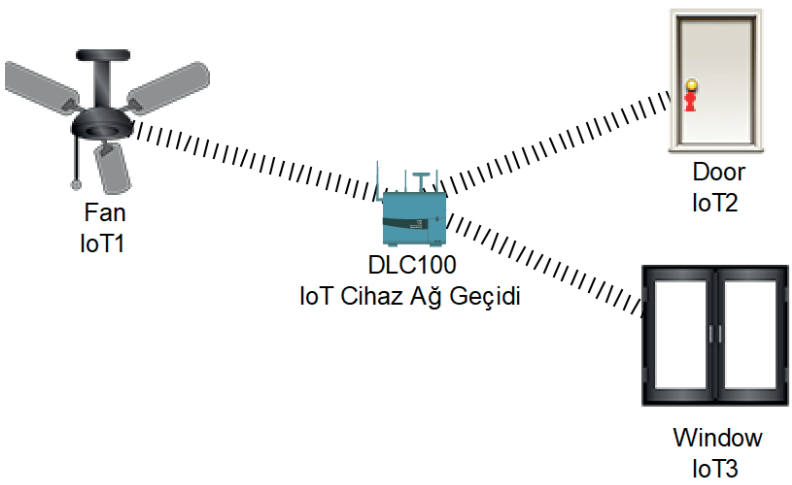
Tablo 6.1: Nesnelerin İnterneti Mimarilerinin Karşılaştırılması

Ölçütler	Üç Katmanlı IoT Mimarisi	Beş Katmanlı IoT Mimarisi	Servis Odaklı IoT Mimarisi	Ara Yazılım Temelli IoT Mimarisi
Gizlilik veya Güvenlik	Var	Var	Var	Var
Dinamiklik	Yok	Var	Yok	Yok
Akıllı Kontrol	Yok	Yok	Yok	Yok
Birlikte Çalışma	Var	Var	Var	Var

1. Uygulama

İşlem adımlarına göre IoT cihazlara web arayüzünden erişimi sağlayarak kontrolü gerçekleştiriniz.

1. Adım: Ağ simülasyon programını kullanarak Görsel 6.7'deki topolojiyi oluşturunuz.



Görsel 6.7: IoT cihaz bağlantı topolojisi

2. Adım: IoT cihazlarına kullanıcıların arayüz üzerinden erişimi için birinci adımda çizilen topolojiye TabletPC ekleyiniz.

3. Adım: IoT cihaz ağ geçidi DLC100 cihazına tıklayınız.

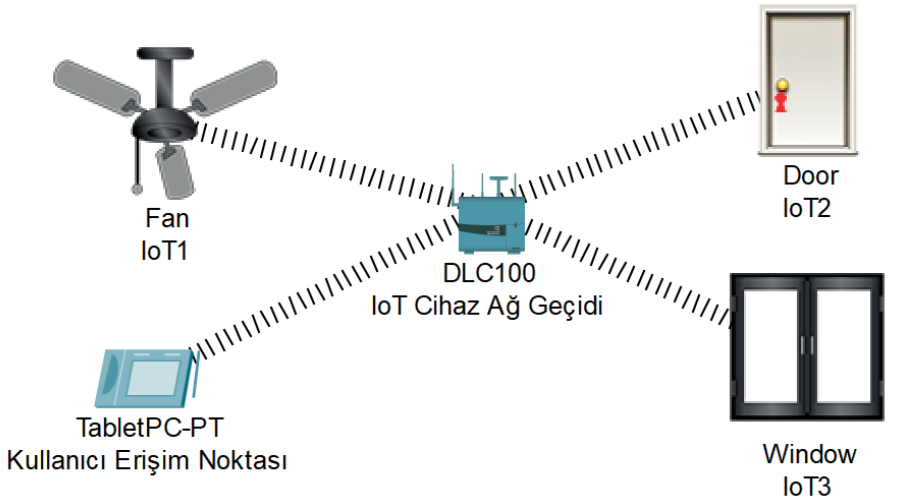
4. Adım: Açılan pencereden **Config** sekmesinin altında bulunan **Wireless0** kısmına tıklayınız.

5. Adım: **Wireless Settings** penceresindeki SSID bölümünde yazan Home Gateway yayın adını not ediniz.

6. Adım: Kullanıcı erişim noktası olarak eklenen **TabletPC** cihazına tıklayınız.

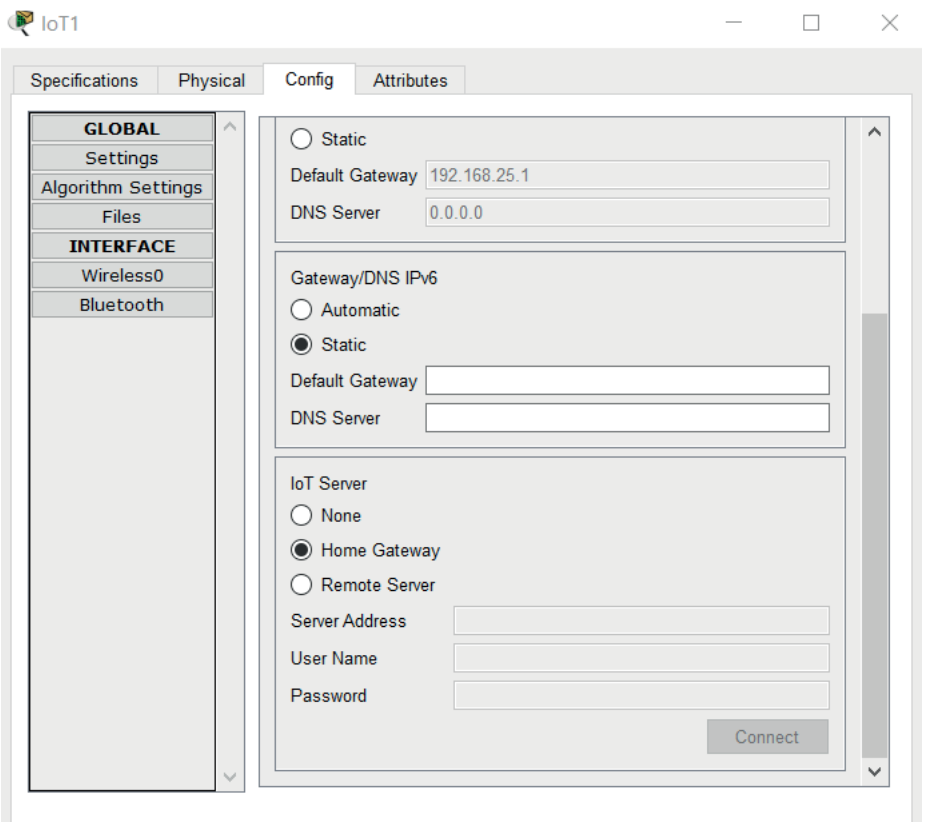
7. Adım: Açılan pencereden **Config** sekmesinin altındaki **Wireless0'**a tıklayınız.

8. Adım: **SSID** kısmındaki **Default** yazan metni silip bu kısma **Home Gateway** yazınız. Bu işlemden sonra IoT cihazlarının arayüzlerine erişim için kullanılacak TabletPC'nin Görsel 6.8'deki gibi IoT cihaz ağ geçidi ile haberleştiğini kontrol ediniz.

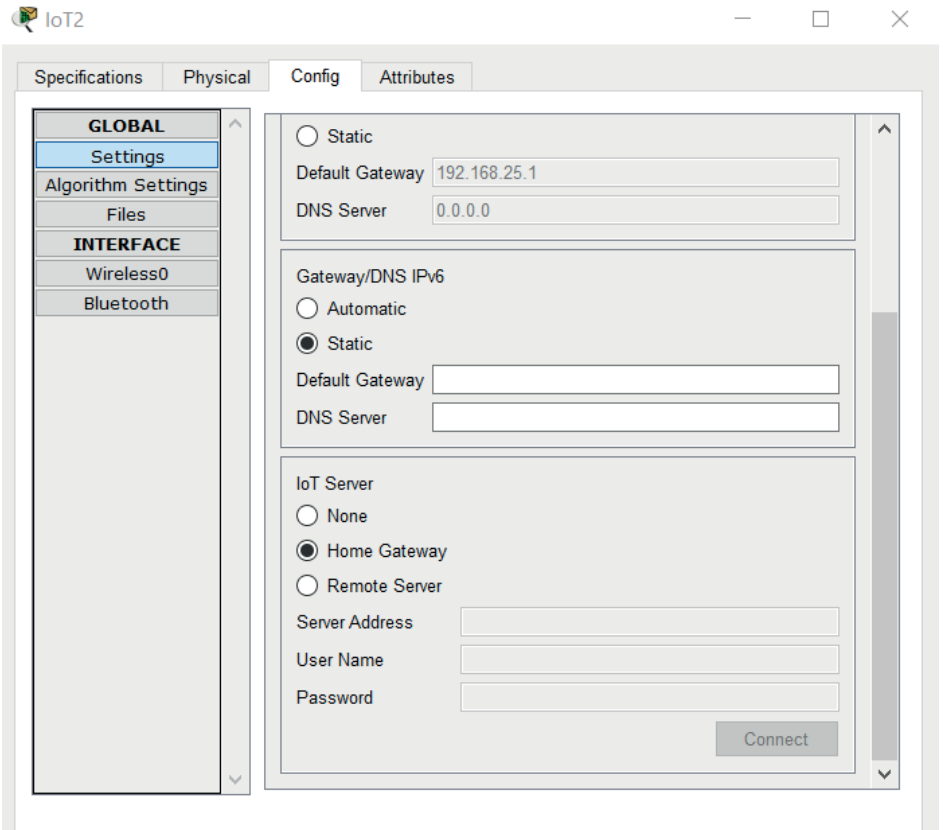


Görsel 6.8: Kullanıcı erişim noktası eklenen IoT cihaz bağlantı topolojisi

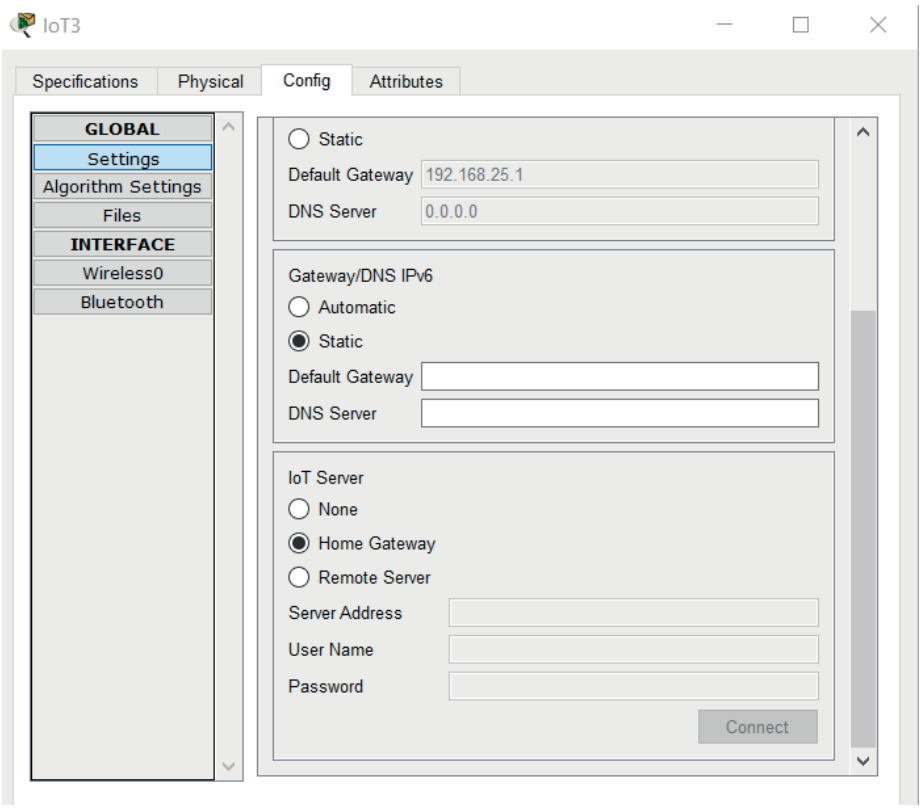
9. Adım: Oluşturduğunuz topolojide yer alan IoT cihazların **IoT Server** ayarlarını yapınız. Bu işlem için Görsel 6.9, Görsel 6.10 ve Görsel 6.11'de görüldüğü gibi her IoT cihaz için ayrı ayrı **Config** sekmelerinde bulunan **GLOBAL Settings** altındaki **IoT Server** ayarını **Home Gateway** olarak değiştiriniz.



Görsel 6.9: IoT1 cihazının IoT server ayarlarının yapılması



Görsel 6.10: IoT2 cihazının IoT server ayarlarının yapılması

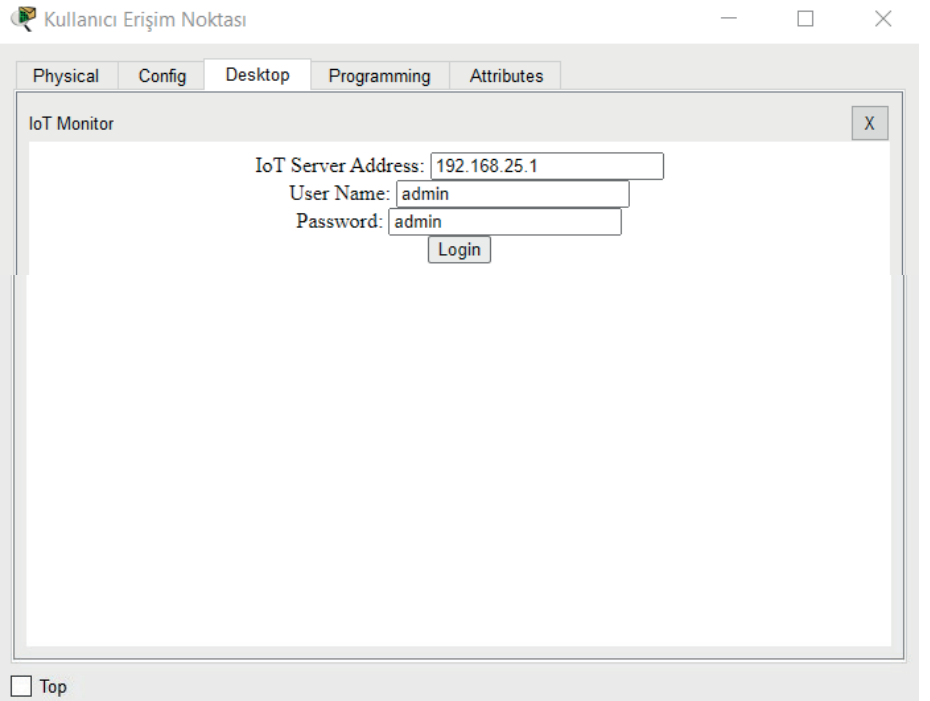


Görsel 6.11: IoT3 cihazının IoT server ayarlarının yapılması

10. Adım: Kullanıcı erişim noktası olarak kullanılan **TabletPC**'ye tıklayınız.

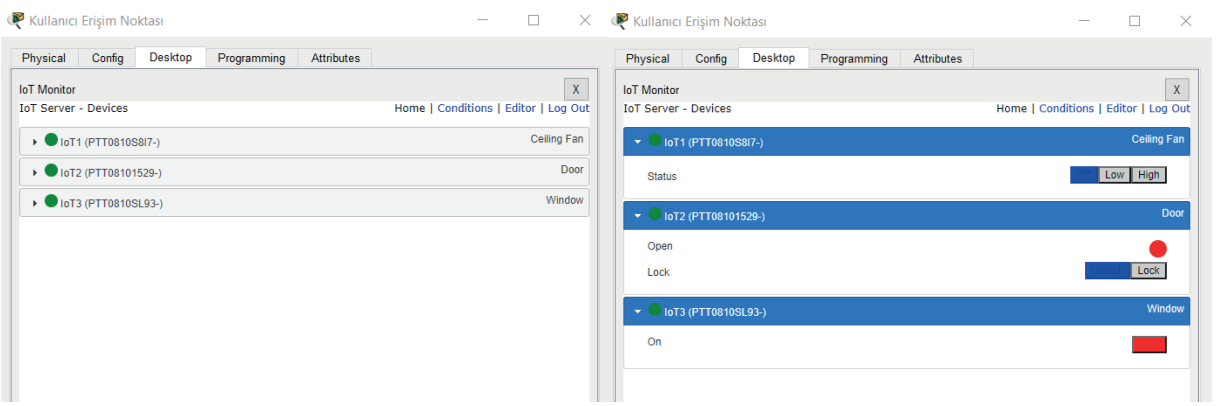
11. Adım: Desktop sekmesinin altında bulunan **IoT Monitor** simgesine tıklayınız.

12. Adım: IoT Server üzerinden gerçekleştirilen bağlantıyla ekrana gelen **IoT Server IP adresi**, kullanıcı adı ve şifresi altındaki Login butonuna tıklayınız (Görsel 6.12).



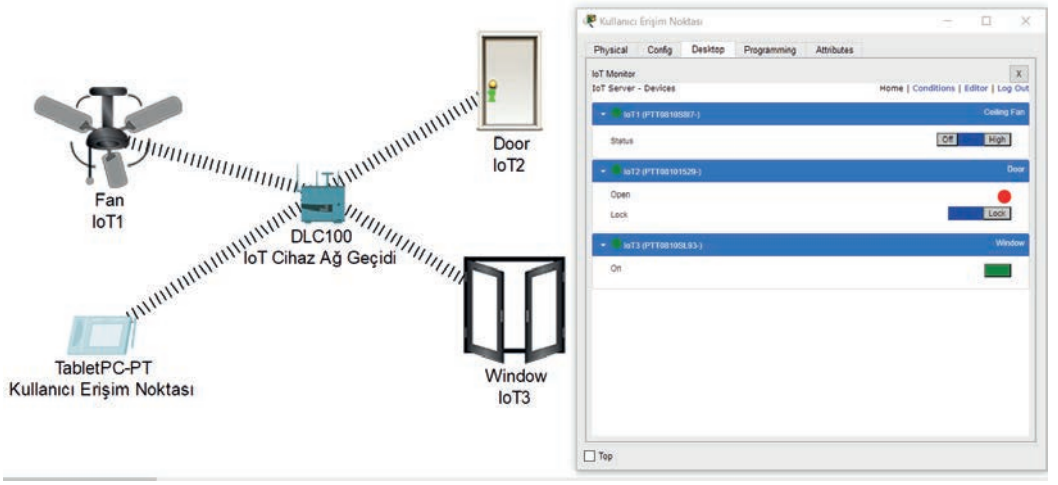
Görsel 6.12: Kullanıcı erişim noktası TabletPC IoT Monitor ekranı

13. Adım: **IoT Monitor** ekranıyla erişim sağlanan IoT cihaz simgelerine tıklayarak Görsel 6.13'teki her IoT cihazının çalışma durumunu gösteren butonlar aracılığı ile IoT cihazlarını kontrol ediniz.



Görsel 6.13: IoT Monitor web arayüzü üzerinden kontrol edilen IoT cihazları

14. Adım: **TabletPC** üzerinden IoT cihazlarının çalışma durumlarını Görsel 6.14'te görüldüğü gibi değiştirerek topolojiyi kontrol ediniz.



Görsel 6.14: IoT cihazlarının web arayüz üzerinden kontrol edilmesi

Sıra Sizde

IoT cihaz ağ geçidini kullanarak dört farklı IoT cihazının laptop üzerinden kontrolünü sağlayan uygulamayı yapınız.

Araştırma

IoT ile kullanılan haberleşme teknolojilerini araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

Okuma Parçası

Türkiye'de Siber Güvenlik

Ülkemizde, 2000'li yılların sonundan itibaren dijital teknolojilerin gelişmesiyle birlikte siber alanda güvenlik daha önemli hale gelmiştir. 2013 yılında, Bilgi Teknolojileri ve İletişim Kurumu (BTK) bünyesinde Ulusal Siber Olaylara Müdahale Merkezi (USOM) kurulmuş; kritik altyapı sektörleri, kurum ve kuruluşlarda Siber Olaylara Müdahale Ekipleri (SOME) faaliyetlerine başlamıştır. Şu anda, USOM koordinasyonunda; 2154 SOME bulunmakta ve bu mekanizmalarda 7 bin siber güvenlik uzmanı çalışmaktadır. Ayrıca, Cumhurbaşkanlığı Dijital Dönüşüm Ofisi altında 2021 yılında 13 kurumun yer aldığı Yerli Siber Güvenlik Ürünlerinin Yaygınlaştırılması Platformu oluşturulmuştur. Sanayi ve Teknoloji Bakanlığı bünyesinde ise Organize Sanayi Bölgelerinde (OSB) bulunan firmalar için toplamda 36 ilde 85 OSB'ye farkındalık eğitimleri düzenlenmiş; bireylere ve sanayiye yönelik üç adet siber güvenlik rehberi geliştirilmiştir. Ülkemiz, bu vesile ile siber güvenlik alanında ilerleme kaydetmiştir. Uluslararası Telekomünikasyon Birliği'ne (ITU) göre, Küresel Siber Güvenlik Endeksi'nde, Türkiye, 2020'de Avrupa'da 11. sıradan 6. sıraya, dünya genelinde ise 20. sıradan 11. sıraya çıkmıştır (T.C. Ulaştırma ve Altyapı Bakanlığı, 2023; T.C. Cumhurbaşkanlığı Dijital Dönüşüm Ofisi, 2023; T.C. Sanayi ve Teknoloji Bakanlığı, 2023).

https://edergi.sanayi.gov.tr/File/Journal/2023/12/12_2023.pdf

6.2. IoT ZAFİYET VE TEHDİTLERİ

IoT ağlarını oluşturan cihazlar günlük hayatın bir parçası hâline gelmiştir. Üretimden sağlığa, akıllı şehirlerden askerî uygulamalara kadar her alanda yaşamı kolaylaştırmak için geliştirilen bu cihazlar; günlük yaşamı daha verimli kılar. Akıllı cihazların hızla artışı, güvenlik risk ve tehditlerini de ortaya çıkarır. IoT cihazlarının mimari olarak tasarımında bütünleşmiş bir güvenlik yaklaşımı bulunmadığı için düşük seviyede güvenli bir alan oluşur. Bu durum, IoT ağlarının siber saldırılara karşı hedef olmasına yol açar. Algılayıcıların aktif şekilde birbiriyle sürekli iletişim hâlinde olması, ağa bağlı tüm cihaz ve sistemlerin güvenlik risklerini artırır. Bu durum; sağlık, enerji, üretim, akıllı şehirler gibi kritik altyapılarda kullanılan algılayıcıların güvenlik gereksinimlerini ön plana çıkarır. Güvenlik ve gizlilik zafiyetleri; kimlik doğrulama, veri bütünlüğü, cihaz yönetimi, erişim kontrolü gibi tehditlerin artmasına yol açar. IoT cihazları sınırlı bellek kapasitesine sahiptir. Bu durum, şifreleme algoritmalarının yetersiz kalmasına neden olur. Sınırlı bellek kapasitesine sahip bu cihazların oluşturduğu ağlara yönelik saldırılarda yaygın olarak veri paketleri manipüle edilir. Örneğin IoT ağlarına yönelik yapılan **Dağıtılmış Hizmet Reddi (DDoS)** saldırılarında sürekli tekrar ederek cihazların bellek kaynaklarını tüketen paketler gönderilir. Kötü amaçlı yazılım saldırılarında cihaza yüklü olan yazılımların çalışması engellenir. IoT cihazlarındaki veri alışverişi esnasında veri iletişim ağının manipülasyonuyla ağın parçası gibi görünerek ağdaki cihazları taklit eden ve kötü amaçlı davranan cihazlar kullanılır. Bu tür saldırılar, IoT cihazlarının konfigürasyonlarının doğru yapılandırılmasının ve yazılımlarının güncel tutulmasının önemini gösterir. IoT katman mimarisinde güvenlik düzeyleri Tablo 6.2'de verilmiştir. Her katman düzeyinde olası saldırılara karşı alınacak güvenlik önlemlerine dikkat edilmelidir.

Tablo 6.2: IoT Mimarilerinde Katman Bazlı Saldırı Türleri ve Güvenlik Politikaları

IoT Katman Mimarisi	Saldırı Türleri	Güvenlik Önlem ve Algoritmaları
Uygulama Katmanı	Mirai- malware IRCTelnet Injection Poisoning Evasion Impersonate Inversion	AES RSA
İletim Katmanı	TCP flooding UDP flooding SYN flooding De-synchronization	IDS Authentication
Ağ Protokol Katmanı	DDoS Privacy tracking MiTM Bluebugging	RSA 3DES AES DSA ECDH
Fiziksel Katman	Eavesdropping RFIS tracking	Authentication Faraday Cage Present

6.2.1. IoT Zafiyetleri

IoT ağlarının sahip olduğu güvenlik açıklarının belirlenmesinde çeşitli yöntemler izlenir. IoT mimarisi ve güvenlik düzeyleri belirlenirken saldırı türleri ile bu saldırılara karşı alınacak güvenlik önlemleri katman bazlı değerlendirilmelidir. Böylece olası tehditler için protokol bazlı önlemler alınmalıdır. IoT ağlarında güvenlik zafiyetlerinin sebepleri şunlardır:

- **Güvenlik Düzeyi Düşük Ağ Servisleri:** Bilgi güvenliğinin temelinde yer alan verinin gizliliği, bütünlüğü ve erişebilirliği noktasında cihaz güvenliğini riske atan durumlara karşı güvensiz ağ servislerinin devre dışı bırakılmasıdır.
- **Yetersiz Savunma Önlemleri:** Sistem yöneticisi veya kullanıcıların bilgilerinin güvensiz ortamlarda saklanarak yetkisiz kullanıcıların erişimine olanak sağlanmasıdır.
- **Güvenlik Düzeyi Düşük Donanım Arayüzleri:** Kimlik doğrulama, kimlik yetkilendirme gibi temel güvenlik açıklarının özellikle cihazların mobil arayüzleri üzerinden ortaya çıktığı zafiyetlerdir.
- **Depolama ve Veri İletimindeki Zafiyetler:** IoT cihazlarındaki hassas verilerin şifrelenmemiş ve düşük güvenlik seviyesinde bulunmasıdır.
- **Ön Tanımlı (Default) Ayarlar:** Ön tanımlı ayarlarında bırakılan IoT cihazlarının bir ağa bağlandığında kaba kuvvet saldırıları ile servis dışı kalmasıdır.
- **Yetersiz Cihaz Yönetimi:** Cihazlar üzerinden sistemlerin izlenmesi, cihazların güncellenmesi, sistem yönetimi gibi birçok özelliğin aktif olarak kullanılamamasıdır. Üretimi esnasında cihazlara yüklenen güvenlik politikaları, kullanıcılar tarafından güncellenerek değiştirilmelidir.
- **Zayıf Şifre ve Sabit Parola Kullanımı:** Cihaz arayüzlerine erişimde kullanılan panellerin parola ve şifrelerinin güvenlik kriterlerine göre güncellenmesidir. Aksi takdirde yetkisiz erişim ve arka kapı saldırılarına karşı zafiyetler oluşur.

IoT cihazları, bilgisayarların aksine güvenlik duvarına sahip değildir. Olası zararlı yazılımları tespit etme yetenekleri yoktur. Saldırıya uğrayan cihazların tespitinin zor olması, saldırı sürecini uzatarak saldırının etkisini artırır.

6.2.2. IoT Saldırıları

IoT ağlarına yönelik saldırılarını gerçekleştiren saldırganların genel olarak bir cihaz üzerinden birçok cihaza saldırma stratejileri ortaya koyduğu görülür. IoT ağlarına saldıranların temel motivasyon kaynağı, güvenlik seviyesi düşük cihaz sayısının fazla olmasıdır. Özellikle ulusal güvenlik ve endüstriyel üretim alanlarındaki saldırılar ekonomik açıdan büyük kayıplara yol açar.

Stuxnet zararlı yazılımı, 2010 yılında bilgisayar solucanı olarak USB flash sürücülerine bulaşmış ve pasif saldırı yöntemi ile kendini sisteme enjekte etmiştir. 2016 yılında ortaya çıkan Mirai kötü amaçlı yazılımı, yapısal olarak kapsamlı IoT cihazlarına yönelik yapılan saldırıdır. Mirai zararlı yazılımı, interneti tarayarak varsayılan kullanıcı adı ve şifresine sahip cihazların listesini oluşturur. Bir IoT cihazına bulaşan bu zararlı yazılım, botnet ağının bir parçası olur. Saldırgan, kurbanına yönelik oluşturduğu botnet ağı üzerinden DDoS saldırısını gerçekleştirir. Brickerbot kötü amaçlı yazılımı 2017 yılında keşfedilmiştir. Yapısal olarak Mirai zararlı yazılımına benzer. Hedef ağı tarayarak varsayılan ve zayıf parolalara sahip IoT cihazlarında oturum açmaya çalışır. Enjekte olduğu cihazdaki ağ yapılandırma bilgilerini siler. IoT cihazını fiziksel olarak müdahale edilmediği sürece kullanılamaz hâle getirir.

IoT ağlarına yönelik saldırı ve zararlı yazılımlar incelendiğinde IoT cihazlarının konfigürasyon ve kurulumlarının yapılması sırasında güvenlik tedbirlerinin alınmasının önemi ortaya çıkar. Yetkilendirme sorunları, kimlik doğrulama zafiyetlerine yol açar. Varsayılan şifrelerin güncellenmemesi, bağlantı sırasında saldırı risklerini artırır. Botnet ve Kalıcı Hizmet Reddi saldırıları bu tür durumlarda tespit edilemez. Tablo 6.3'te IoT kullanım platformlarına göre saldırı noktaları ve saldırı türleri verilmiştir. Bu saldırılardan korunmak için enerji üretim ve iletim merkezlerinde dış ağdan yalıtılmış alanların oluşturulması gerekir. Bu alanlar, aktif ağlara bağlı olmadığı için yaygın şekilde tek kanallı ve yetki yükseltme saldırıları görülür. Endüstriyel IoT uygulamalarında özellikle enerji üretim ve iletim alanları tehdit altındadır. Hücresel bağlantı ve radyo sinyalleri ile veri ileten ağlarda yaygın olarak DDoS saldırılarına rastlanır.

Tablo 6.3: IoT Saldırı Türleri ve Saldırı Alanları

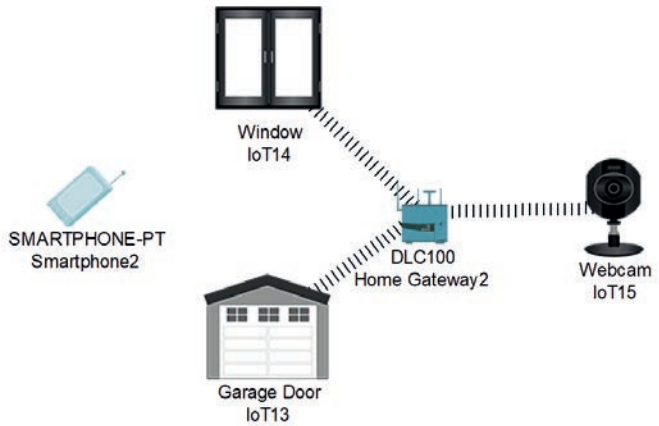
IoT Platformları	Saldırı Tipleri	Saldırı Noktaları	IoT Uygulama Alanları
Son Kullanıcı Alanları	DDoS saldırıları	Kablosuz ağlar	E-sağlık
	Spoof saldırıları	Hız sensörleri	Ulaşım
	Replay saldırıları	Near-Field Communication (NFC)	Finans
	Yetki yükseltme saldırıları	Personel Area Network (PAN)	Dijital giriş alanları
Endüstriyel Uygulamalar	Kaba kuvvet saldırıları	Bilgi iletim ağları	Enerji sahaları
	DDoS saldırıları	Radyo sinyal ağları	Trafik kontrol
	GPS Snoofing	Global Position System (GPS) ağları	Ulaşım noktaları
Üretim ve İletim Merkezleri	Tek kanal saldırıları	Universal Serial Bus (USB) aygıt giriş noktaları	Üretim kontrol merkezleri
	Yetki yükseltme saldırıları	Haricî ağ çıkış noktaları	Üretim iletişim merkezleri

IoT cihazlarına yönelik saldırılar ile sisteme giriş yapılan bir IoT cihazı üzerinden tüm ağdaki IoT cihazlarına erişilir. Bu saldırılardan korunmak amacıyla dışa açık ağ topolojileri yerine özel ağ altyapıları kurulmalıdır. Uç ağlar, hücresel iletişim kanalıyla cihazlar arasındaki ağların yalıtılmasını sağlar. Günümüzde yaygınlaşan hücresel güvenlik duvarı kullanımı bu noktada saldırılara karşı alınacak bir önlemdir. Hücresel güvenlik duvarları, yapılandırılmış IP adres kümesi ile ağ bağlantısına izin verir.

2. Uygulama

İşlem adımlarına göre IoT ağlarında default (ön tanımlı) olarak bırakılan kablosuz yayın adını değiştirerek, kablosuz erişim parolasını belirleyip ön tanımlı ayarlarında bırakılan IoT cihazları için cihaz yönetim zafiyetini tespit ediniz..

1. Adım: Ağ simülasyon programını kullanarak Görsel 6.15'teki topolojiyi oluşturunuz. İlk durumda eklediğiniz cihazlar, **Home Gateway** şeklinde eklenen cihaza fabrika ayarlarında ön tanımlı ayarlarıyla otomatik olarak bağlantı kurar. IoT cihazlarına erişim ve kontrol için tablet, telefon gibi cihazlara ihtiyaç duyulur. Görsel 6.15'te görülen **Smartphone'da** Home Gateway için tanımlanan SSID (kablosuz yayın adı) ve parola yapılandırılmadığı için bağlantı kurulmaz.



Görsel 6.15: IoT cihaz bağlantı topolojisi

Not

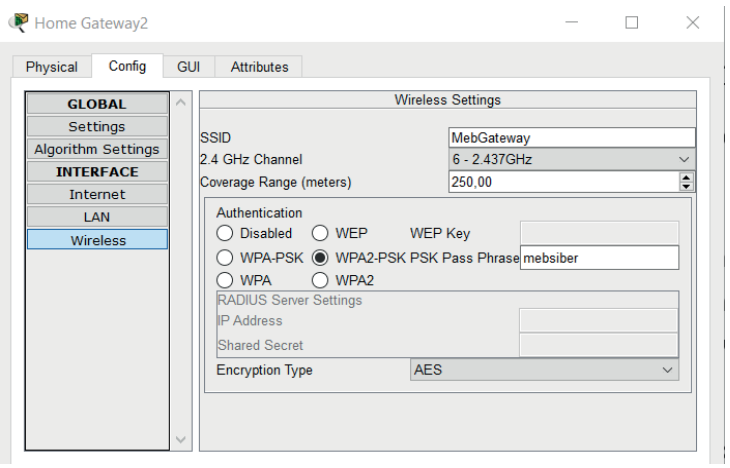
Topolojide kullanılan cihazların adları; IoTx, Home GatewayX veya SmarphoneX şeklinde değişiklik gösterebilir. Cihazların adları değiştirilebilir. Örneğin Görsel 6.15'te görülen IoT14, Pencere; IoT13, Garaj; IoT15, Kamera olarak adlandırılabilir.

2. Adım: IoT cihaz ağ geçidi **DLC100 Home Gateway** cihazına tıklayınız.

3. Adım: Açılan pencereden **Config** sekmesinin altında bulunan Wireless kısmına tıklayınız.

4. Adım: **Wireless Settings** penceresindeki SSID kısmında ön tanımlı olarak HomeGateway'ı görüntüleyiniz. Ağ geçidi olarak kullanılan cihazlardaki bu ön tanımlı yayın adı değiştirilmediği takdirde bağlı olduğu IoT ağındaki cihazlara erişim sağlandığını test ediniz.

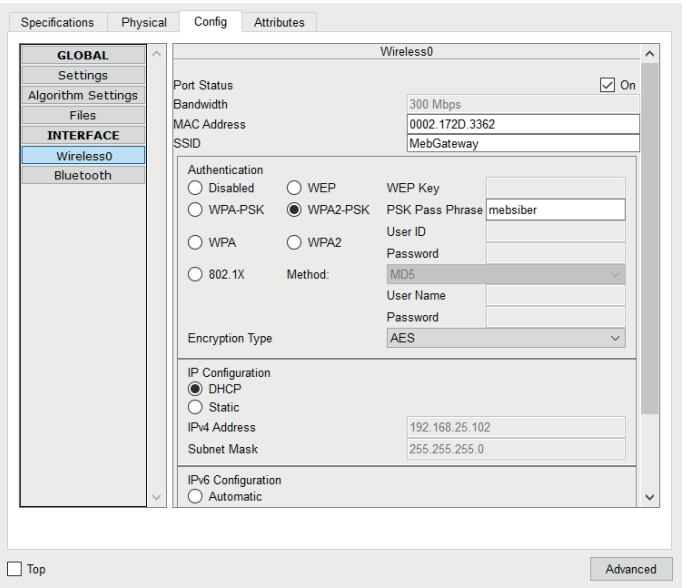
5. Adım: **Authentication** bölümündeki **WPA2-PSK'yı** seçerek sağ bölümde açılan kutuya mebsiber yazınız. Böylece kablosuz yayın adı **MebGateway** olan bu cihaza kablosuz bağlanmak isteyen diğer IoT cihazları **mebsiber** parolası ile bağlanabilir (Görsel 6.16).



Görsel 6.16: Erişim noktası yayın adının ve kablosuz bağlantı parolasının belirlenmesi

6. Adım: Oluşturduğunuz topolojide yer alan IoT cihazlara tıklayıp **Config** sekmelerinde bulunan **INTERFACE** altındaki **Wireless0**'ı seçiniz.

7. Adım: SSID kutucuğuna **MebGateway** yazınız (Görsel 6.17).

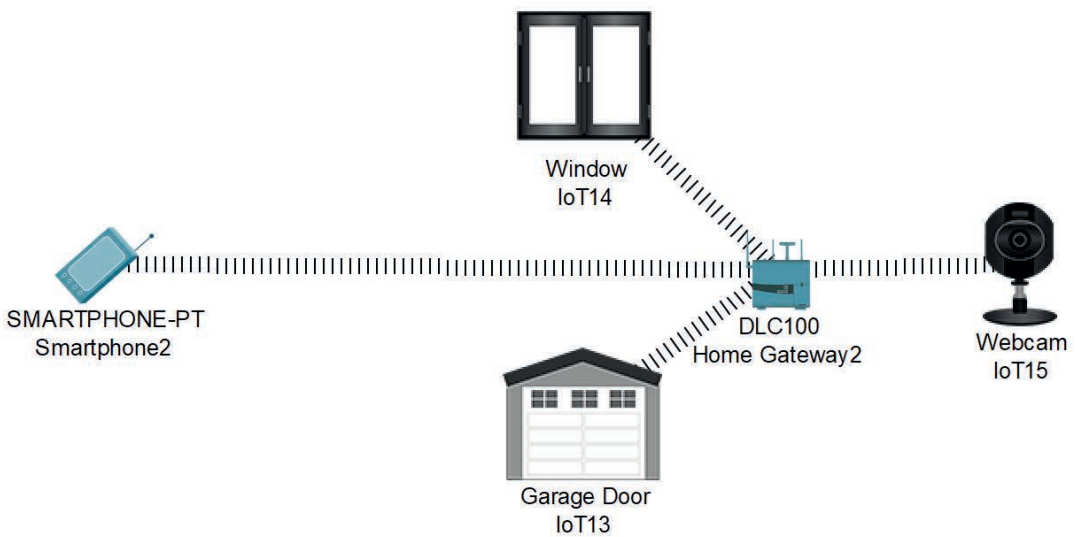


Görsel 6.17: Home Gateway cihazına IoT cihazlarının bağlanabilmesi için IoT cihazlarında SSID ve parola bilgisinin tanımlanması

8. Adım: Authentication bölümünden **WPA2-PSK** seçip açılan kutuya **mebsiber** yazınız. Bu işlem, her IoT cihazı için ayrı ayrı yapılmalıdır. Böylece IoT cihazlar kablosuz şekilde IoT ağ geçidi olarak kullanılan Home Gateway adlı cihaza bağlanabilir.

Not

Her cihaz için Wireless0 ayarları SSID: **MebGateway**, WPA2-PSK: **mebsiber** olarak değiştirildiğinde Görsel 6.18'de görüldüğü gibi tüm cihazlar kablosuz olarak yeni belirlenen güvenlik politikası sayesinde birbiriyle bağlantı kurar.



Görsel 6.18: Fabrika ayarları değiştirilen IoT cihazları ile akıllı cihazların birbiriyle kablosuz olarak bağlantı kurabilmesi

Sıra Sizde

Ağ simülasyon programında IoT ağ geçidini kullanarak bir IoT ağ topolojisini aşağıdaki işlem basamaklarına göre oluşturunuz.

1. Dört farklı IoT cihaz ekleyiniz.
2. **SSID:**TURKIYEM, **Authentication türü:** WEP, **şifre:** NICE100YILLARA yazınız.
3. Bağlantıları kontrol ediniz.
4. IoT Monitor üzerinden IoT cihazları kontrol ediniz.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. Ön tanımlı SSID adını değiştirdi.		
2. Kimlik doğrulamayı aktif ederek şifre belirledi.		
3. Akıllı bir cihaz üzerinden IoT cihazlara bağlandı.		
4. IoT cihazlarının ön tanımlı ayarlarını değiştirdi.		
5. Bir istemci üzerinden IoT cihazlarını kontrol etti.		

"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.

Araştırma

IoT katman bazlı saldırı ve güvenlik politikaları incelendiğinde algoritmalar sayesinde cihazlarda alınan önlemler görülür. Burada kullanılan güvenlik algoritmalarını ve algoritmaların çalışma şeklini araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

6.3. IoT GÜVENLİK TEDBİRLERİ

IoT cihazları, algılayıcılar aracılığı ile verileri toplayarak bağlı oldukları ağ üzerinden bir veri merkezine veya bulut teknolojisine gönderir. Toplanan verilerin analizi sırasında iletişimi sağlayan ortamın ve veri protokollerinin güvenli olması gerekir. IoT ağlarının genişlemesi özellikle güvenlik, enerji, üretim alanlarındaki riskleri artırır. Bu alanlarda sadece anahtarlama görevini yerine getiren IoT cihazlarındaki zafiyetler saldırı kaynağına dönüşebilir. Güçlü parola ve şifre seçimi, cihazların yazılımlarının güncellenmesiyle başlayacak güvenlik tedbirleri, her bir IoT cihazına bağlı olarak dikkatli bir şekilde yapılandırılmalıdır.

6.3.1. IoT Veri Protokolleri

IoT cihazlarının oluşturduğu ağlar hızla genişler. Ağa bağlı IoT cihazlarının işlevlerini hızlı ve doğru şekilde yerine getirebilmesi için cihazların arasındaki veri akışının sorunsuz olması gerekir. Bilgisayar biliminde **protokol** kavramı, cihazların birbiriyle anlaşabildikleri ortak dil olarak tanımlanır. Veri akışının sağlanması için cihazların ortak protokol üzerinden haberleşmesi istenir. Veri protokolleri, iletişimin düzenli ve güvenli şekilde yürütülmesini sağlar. IoT ağlarında cihazların iletişimi için kullanılan veri protokolleri şunlardır:

- **HTTP (Hypertext Transfer Protocol):** Hiper-metin transfer protokolü anlamına gelir. Yüksek veri kaynağı ihtiyacı ve performans problemleri gibi nedenlerle tercih edilmese de yalıtılmış ağlarda kullanılan IoT cihazları arasında yüksek boyutlu veri transferi ihtiyacı olduğunda HTTP kullanılır.
- **XMPP (Extensible Messaging and Presence Protocol):** Bir ağ üzerinden iletişim kurarak veri alışverişi yapabilen IoT cihazları için uçtan uca şifreleme, kimlik doğrulama ve yetkilendirme sağlar. Farklı alanlarda kullanılan cihazların birbirleriyle iletişim kurmasını destekler. Çeşitli uygulamalar için tercih edilen güçlü bir protokoldür.
- **MQTT (Message Queuing Telemetry Transport):** Message Queuing Telemetry Transport, basit veri protokolüdür. Kısıtlı donanım kaynaklarına sahip cihazlar için tasarlanmıştır. Birden fazla IoT cihazı arasındaki haberleşme için sıkça kullanılır. IoT cihazı, istenen veri yayınını yapar. Bu cihazla iletişim hâlinde olan diğer cihazlar da yayınlanan verileri alır. Cihazların düşük güç tüketimini sağlar ve TCP/IP protokolü üzerinden verileri daha geniş alanlara ulaştırır. MQTT, endüstriyel IoT ağlarında tercih edilen bir veri protokolüdür.
- **OPC UA (OPC Unified Architecture):** OPC kuruluşunca akredite edilen ve endüstriyel üreticiler tarafından kabul gören protokoldür. Websocket, MQTT gibi protokolleri destekler. OPC UA, endüstriyel IoT cihazlar arasında tercih edilen standart hâline gelmiştir.
- **DDS (Data Distribution Service):** Veri dağıtım hizmeti anlamına gelen DDS, yüksek kalitede veri transferi için kullanılan bir protokoldür. Özellikle IoT ağlarının bulut teknolojileri ile entegrasyonu sonucunda oluşan veri iletişiminde sıkça kullanılır. Gömülü sistemlerin yaygın olduğu IoT cihazlarında gerçek zamanlı iletişim için ölçeklenebilir çözümler sunar.
- **AMQP (Advanced Message Queuing Protocol):** Gelişmiş mesaj sıralama anlamına gelen AMQP, açık kaynaklı bir veri protokolüdür. IoT cihazlara gelen verileri saklayıp önceliklendirme gibi ilişkisel özelliklere sahiptir. IoT ağlarının sunucu tabanlı veri aktarım ihtiyaçları için güvenli ve kararlı bir şekilde çalışır. Bu yönüyle özellikle bankacılık sektöründeki IoT veri iletişim çözümlerinde tercih edilir. Yüksek veri kaynağına ihtiyaç duyduğu için basit işlevi olan IoT ağlarında kullanılması uygun değildir.
- **CoAP (Constrained Application Protocol):** Kısıtlı uygulama protokolü anlamına gelen CoAP, kısıtlı kaynaklara sahip cihazlar arasında veri iletişimi için tasarlanmıştır. Düşük bant genişliğine sahip ağlarda User Datagram Protocol (UDP) üzerinden çalışır. Akıllı şehir, akıllı bina ve enerji iletim IoT uygulamalarında kullanılır.
- **WebSocket:** Yüksek sayıda cihazın bağlı olduğu IoT ağlarında kesintisiz veri iletişimi için tercih edilir. Sunucu ve istemci olarak verilerin iletilmesini sağlar. Özellikle internet üzerinden yönetilen IoT cihazlar arasındaki veri iletiminde kullanılır.

6.3.2. IoT İletişim Protokolleri

IoT iletişim mimarisi; cihazlar, ağ geçitleri ve veri merkezlerinden oluşur. Veriler, iletişim protokolleriyle bu yapı içinde sürekli iletim hâlindedir. Endüstriyel üretimde kullanılan algılayıcılar, üretim süresince birbirleriyle iletişim kurar. Bu tür iletişimler, aygıttan aygıta şeklinde tanımlanır. Ağ geçitleri, farklı IoT ağlarını birbirine bağlayan noktalardır. Ağ geçitleri, IoT cihazlarından gelen verileri birleştirip veri merkezlerine aktarır. Verilerde bütünlük ve doğruluk açısından problem olduğunda gelen verileri IoT cihazlarına geri gönderir. Ağ geçitlerinden veri merkezlerine iletilen verilerin hangi protokol ile geri gönderileceği, güvenlik gereksinimleri ve iletimdeki tıkanıklık gibi durumlar sonucunda belirlenir. Farklı IoT ağlarına bağlı veri merkezleri arasında kurulacak bağlantıyla uygun protokoller seçilir. IoT ağ türleri, iletişim protokollerinin seçiminde önemli rol oynar. IoT cihazları arasındaki mesafeye göre iletişim ağları şu kategorilere ayrılır:

- **Nano Ağ:** Depolama, hesaplama, algılama gibi temel görevler için cihazların oluşturduğu basit ağdır. Nano teknolojileri, biyometrik sistemler ve askerî alanda uygulanır.
- **Yakın Mesafe İletişimi (NFC):** Farklı kategorilerdeki elektronik aygıtların düşük hızla 1-4 cm aralığında birbiriyle kablosuz iletişim kurduğu ağdır. Akıllı kimlik, temassız ödeme, akıllı kartlar gibi alanlarda uygulanır.
- **Gövde Alanı Ağı (BAN):** Giyilebilir teknolojilerde kullanılan, vücuda monte edilen implant, RFID gibi cihazların bilgisayarlarla iletişim kurması için tasarlanan kablosuz ağdır.
- **Kişisel Alan Ağı (PAN):** Günümüzde yaygın olarak kullanılan bir ağdır. Kullanıcı, farklı işlevlere sahip kişisel cihazlarını iletişim protokolleriyle birbirine bağlar. Oluşturduğu ağ, dar bir alanda aktif olarak çalışır.
- **Yerel Alan Ağı (LAN):** Genel bilgisayar ağlarında olduğu gibi bir binanın kapladığı alanda iletişim ortamının bulunduğu ağdır.
- **Kurumsal Alan Ağı (CAN):** Yerel ağların sınırlı bir alanda birleşmesiyle oluşturulan ağdır.
- **Metropolitan Alan Ağı (MAN):** Genel olarak mikrodalga iletim teknolojisi veya hücresel ağ ile desteklenen geniş alan ağıdır.
- **Geniş Alan Ağı (WAN):** Farklı büyüklükte yerel ve genel olarak tasarlanan ağların birleşerek oluşturduğu ağdır.

Cihazlar arasında yönlendirme yapılarak veri kayıplarını azaltan mesh ağları da IoT cihazlarında kullanılır. Farklı işlevlere sahip birçok IoT cihazının birbirine bağlandığı ve sürekli veri akışının sağlandığı IoT ağlarında veri gecikmesinin minimum olması beklenir. Cihaz bazlı veri trafiğinin izlenerek yönetilmesi, ağdaki verilerin güvenliği için önemlidir. Bu gereksinimler; iletişim protokollerinin nasıl, nerede ve ne zaman kullanılması gerektiğini de ortaya çıkarır. IoT cihazları için en çok kullanılan iletişim protokolleri şunlardır:

- **Wi-Fi:** Günümüzde yaygın olarak kullanılır. Seçilen kanal frekansına göre yüksek veri aktarım özelliği sayesinde IoT ağlarında sıkça tercih edilir. Wi-Fi sayesinde hızlı veri aktarımı sağlanır. Yüksek kapasitede veri işleyebilme özelliği sayesinde verimliliği fazladır.
- **ZigBee:** Düşük güçte çalışabilme özelliği sayesinde yaygın bir IoT iletişim protokolü hâline gelmiştir. Veri hızının düşük olması dezavantajlarından biridir. Kablosuz iletişim alanı Wi-Fi'den düşük olsa da IoT cihazlarının iletişiminde sıkça kullanılır.

- **NFC:** Son yıllarda kullanımı giderek artan bir iletişim protokolüdür. İç içe yerleştirilmiş iki anten arasında oluşturulan elektromanyetik indüksiyon sayesinde ortalama 4 cm'den daha yakın mesafeden iletişim sağlar. Akıllı telefonlar üzerinden temassız ödeme gibi hizmetleri alınmasında yaygın olarak kullanılır.
- **Bluetooth Low-Energy (BLE):** IoT teknolojilerinin gelişmesiyle iletişim protokolü olarak kullanılmaya başlandı. Düşük güç tüketimi ve esnek kullanımı, BLE'nin en önemli özellikleridir.
- **LoRaWAN:** Düşük güce sahip birçok IoT cihazından oluşan ağlarda iletişimi sağlamak için tasarlanan protokoldür. Uzun menzilli geniş alan ağı olarak da bilinir. Çift yönlü iletişim ve düşük maliyet özellikleri ile endüstriyel IoT alanında tercih edilir. Akıllı şehir projelerindeki milyonlarca algılayıcı, LoRaWAN protokolüyle iletişim kurar.
- **Hücresel Bağlantı:** Geniş iletişim alanına ihtiyaç duyulan coğrafi bölgelerdeki IoT cihazları için tercih edilir. Büyük miktarda veri akışına imkân sağlar. Farklı lokasyonlarda hizmet veren IoT ağları arasında iletişim için kullanılan hücresel bağlantı, 5G teknolojisinin de yaygınlaşmasıyla daha önemli hâle gelmiştir.
- **SigFox:** Güç tüketimi en az olan IoT iletişim protokolüdür. İşletme maliyetlerini düşürür. Isı, ışık, nem gibi akıllı ölçümler yapan IoT cihazlarının birbiriyle haberleşmesi için kullanılır.
- **Z Dalgası:** Kablosuz bir iletişim türüdür ve RF teknolojisine dayanır. Akıllı evler, akıllı aydınlatma sistemlerindeki algılayıcıların haberleşmesinde kullanılır.

Tablo 6.4'te IoT iletişim protokollerinin güçlü ve zayıf yönleri verilmiştir. IoT cihazlarının kullanım alanlarına göre uygun iletişim protokollerinin belirlenmesinde protokol kaynaklı zafiyetlerin göz önünde tutulması gerekir.

Tablo 6.4: IoT İletişim Protokolleri ve Özellikleri

İletişim Protokolleri	Güçlü Özellikleri	Zayıf Özellikleri
BLE	Düşük güç tüketimi	Kimlik takibi
NFC	Basit veri iletimi	Sınırlı aralıkta iletişim
Hücresel Bağlantı	Taşınabilirlik	Pil sınırlılığı
ZigBee	Düşük güç tüketimi	Korumasız tek kanal iletim
Wi-Fi	Verimli ve hareketli	Sınırlı alanda erişim
MQTT	Basit veri akışı	Zayıf protokol, sınırlı hizmet
DDS	Bulutta dinamik kullanım	Standart alanda kullanım
AMQP	Mesaj odaklı kullanım	Standart alanda kullanım
CoAP	M2M protokolü	Sınırlı düğüm
LoRaWAN	Geniş çaplı ağları destekleme	Pil sınırlılığı

Bilgisayar ağlarında uzaktan başka bir ağa erişim sağlamak isteyen kullanıcılar, kimlik denetimi işlemleri için **Uzaktan Kimlik Doğrulama Çevirmeli Kullanıcı Hizmeti (Remote Authentication Dial in User Service-Radius)** protokolünü kullanır. Radius protokolü ile kimlik denetimi, yetkilendirme ve hesap verilerinin yönetimi yapılır. **Kimlik Denetimi, Yetkilendirme, Hesap Yönetimi (Authentication, Authorization, Accounting-AAA),** cihazlara güvenli erişim için kullanılan bir servistir.

6.3.3. IoT Mimarilerine Göre Saldırı Önlemleri

Katman mimarilerine göre bir güvenlik yaklaşımı, saldırıların tespiti ve engellemesinde önemli rol oynar. IoT ağlarına bağlı cihazların basit arayüzleri ve bu arayüzlerde kullanılan güvenlik seviyesinin düşük olması, cihazların bağlı olduğu tüm ağı tehdit eder. Son zamanlarda bu tür durumlara karşı makine öğrenmesi ve yapay zekâ algoritmaları kullanılarak önerilen yaklaşımların başarılı çözümler ürettiği görülür. Özellikle akıllı şehirler, akıllı evler, stratejik öneme sahip enerji üretim ve iletim sistemlerinde kullanılan IoT cihazlarına yönelik saldırılar artmıştır. Algılayıcılara bulaşan zararlı yazılımların tetiklediği olağan dışı trafik üreten saldırılarda buna sebep olan cihazların hızlı ve doğru şekilde tespit edilmesi gerekir. IoT ağlarında kimlik doğrulama, güvenli ağ politikaları ve yetkilendirme şeklinde bir saldırı tespit ve önleme yaklaşımı geliştirilmelidir. Bu noktada cihazlara yönelik dijital sertifika ve anahtarlar, kullanıcı adı ve parola politikaları da IoT mimarilerine göre katman güvenlik yaklaşımını zorunlu kılar.

6.3.3.1. Fiziksel Katman Saldırıları ve Alınacak Önlemler

IoT cihazlarının haberleşmesi sırasında yaydığı sinyallerin izlenmesi ile yapılan saldırılar, **fiziksel katman saldırıları** olarak adlandırılır. Dijital cihazlardan yayılan manyetik alan, çevresindeki diğer sistem ve cihazların veri aktarması sırasında veri kayıplarının oluşmasına yol açabilir. Bu duruma **Elektromanyetik müdahale (Electromagnetic interference-EMI)** denir. Bu olumsuzluktan etkilenmeden cihazların sorunsuz şekilde çalışmasına ise **Elektromanyetik uyumluluk (Electromagnetic compatibility-EMC)** denir. Elektromanyetik yayılım yoluyla cihazlardaki verilerin ele geçirilmesine **TEMPEST (Transient Elektromagnetic Pulse Emanation Standard)** adı verilir. Özellikle istihbarat toplamak ve karşı istihbarat amacıyla yaygın olarak kullanılan bilgiye erişim yöntemlerinde IoT cihazları da tehlike altındadır. Yayını bozma (jamming) saldırılarında da IoT cihazlarındaki veri akışını engellemek amaçlanır. Verilerin farklı türlerde algılanması (ısı, ışık, basınç, sıcaklık, nem, hareket) özelliği, IoT cihazlarına yönelik bu katmandaki saldırıların önemini artırır. Örneğin telefon sinyalleri dinlenerek yapılan dinleme saldırıları, fiziksel altyapıyı oluşturan kablolama sistemleri üzerinden yapılan saldırılar, akıllı şehir ve evlerdeki sensörlerin elde ettiği verilere yönelik yapılan saldırılar bu kategoride değerlendirilir.

Donanımların sahte cihazlar yardımıyla işlem dışı bırakılması da sıkça görülen saldırılardandır. Saldırgan, IoT cihazına erişim sağlayarak cihazda kayıtlı güvenlik ve anahtar verilerine ulaşır. Saldırgan tarafından oluşturulan sahte IoT cihazları sayesinde tekrarlanan veri trafiğiyle IoT cihazının bağlı olduğu ağ veya sistem devre dışı bırakılır. Düğüm noktalarını ele geçirme, IoT cihaz yayınlarını bozma, sisteme zararlı kod enjekte etme gibi saldırı türleri de bu katmanda yapılır. **Gizlice dinleme (Eavesdropping)** şeklinde yapılan saldırılarda özellikle sağlık alanında mahrem bilgilerin elde edilmesinde RFID gibi algılayıcıların kullanıldığı görülür. Fiziksel katmanda donanım seviyesinde alınacak önlemlerin başında kötü niyetli kullanıcıların IoT cihazlarına doğrudan bağlanmasının engellenmesi gelir. Kapsadığı alandaki cihazları elektromanyetik koruma mantığı ile çalışan Faraday kafesi gibi yaklaşımlar önlem olarak tercih edilir. TEMPEST gibi elektromanyetik yöntemlerle yapılan saldırı ve bilgi hırsızlığına karşı fiziksel kablolama yapılırken uluslararası standartlara uyulmalıdır. Cihazların yapılandırılacağı bina ve lokasyonlar doğru seçilmelidir. Cihazların ağ topolojileri, manyetik alan vb. etkiler azaltılacak şekilde tasarlanmalıdır.

6.3.3.2. Veri Katmanı Saldırıları ve Alınacak Önlemler

Veri bağı katmanı seviyesinde yapılan saldırılarda cihazların fiziksel adresleri (Media Access Control-MAC) temel parametredir. Belleklerine yönelik yapılan tekrarlama saldırıları ile cihazların devre dışı bırakılması amaçlanır. Bu katmanda yapılan saldırılarda veri bütünlüğü değiştirilerek silinir. Sahte olarak yapılandırılan IoT cihazları ile gerçek IoT cihazları arasındaki veri akış yönleri değiştirilir. Ağ geçidi olarak kullanılan aygıtlar üzerinden ağdaki MAC ve adres çözümleme protokolü (Address Resolution Protocol-ARP) kayıtları da manipüle edilir.

Veri katmanı seviyesinde yapılan saldırılara karşı öncelikle kimlik doğrulama ve veri trafiğinin izlenmesi şeklinde önlem alınmalıdır. Özellikle akıllı ulaşım ağları, sinyalizasyon ve üretim hatlarında kullanılan IoT cihazlarına yönelik saldırıların bu katman üzerinden senaryolaştırıldığı görülür. IoT ağlarına yönelik yapılan ortadaki adam saldırısına (Man in The Middle-MiTM) karşı makine öğrenmesi ve derin öğrenme algoritmalarıyla saldırı önleme yöntemleri geliştirilmiştir.

6.3.3.3. Ağ Katmanı Saldırıları ve Alınacak Önlemler

Ağ katmanında görülen saldırıların başında DDoS gelir. Veri erişim, gizli dinleme ve MiTM saldırıları bu katmanda yaygın olarak görülür, e-sağlık ve akıllı şehir sistemlerinde kullanılan IoT cihazlarını etkiler. Hedef IoT cihazlarının IP ve ağ adresleri üzerinden tekrarlanan paketler IoT ağına gönderilir. Gönderilen paketler nedeniyle bir süre sonra IoT cihazı ve bağlı olduğu ağ yanıt veremez hâle gelir. Çalışan bir IoT cihazı **kesintiye (interruption)** uğrayarak bu saldırı sonrasında kapanma moduna geçebilir. Saldırganlar cihazın iletişim protokollerini yanıltarak güvenlik gereksinimlerinde **değişiklik (alteration)** yapar. Bu katmandaki saldırılara yönelik iletişimde protokol bazlı önlemlerin yanında **Diffie-Hellman** ve **RSA** algoritmalarıyla geliştirilen yaklaşımlar kullanılır. DDoS gibi sürekli devre dışı bırakan saldırılara karşı da saldırı tespit sistem tabloları önerilir. Akıllı şehir ve akıllı evlerde kullanılan IoT cihazlarında kimlik doğrulama sorunlarını ortadan kaldırmak için **Gelişmiş Şifreleme Standardı (Advanced Encryption Standard-AES)** kimlik doğrulaması önerilir. Büyük çaplı botnet tabanlı DDoS saldırılarına karşı da **SYN çerezleri (Syn-cookies)** gibi önlemler alınır.

6.3.3.4. Uygulama Katmanı Saldırıları ve Alınacak Önlemler

Uygulama katmanında IoT cihazlarının davranışları için bir standart yoktur. Bu katmanda oturum açma, kimlik doğrulama ve verilerin görüntülenmesi işlemleri yapılır. Bu işlemlere yönelik saldırılarda IoT cihazları büyük tehdit altındadır. Bu sebeple gizlilik ve zafiyet tespit taramaları, erişim kontrol saldırılarına sık rastlanır. Veri paketleri manipüle edilerek yapılan saldırılarda UDP ve TCP üzerinden gönderilen paketlerle IoT cihazlarının devre dışı kalması sağlanır. Saldırgan, **mesaj yenileme (message replay)** saldırılarında tekrarlanan mesajlarla IoT cihazlarını düzensiz çalıştırır. TELNET ve uzaktan yönetim gibi bağlantılarla zararlı yazılımlar sisteme yüklenebilir. Bu sebeple TELNET gibi bağlantı protokollerinde oluşacak güvenlik açıklarında kullanıcı bağlantı bilgilerinin devre dışı kalması istenir. Şifreler sık sık değiştirilmeli ve güçlü uzunlukta seçilmelidir. Günümüz 5G teknolojilerinde bu katmandaki IoT cihazlarına yönelik önlemleri artıran teknolojiler üzerine çalışılır.

IoT katman mimarisindeki saldırıların yanında IoT cihazlarına erişim düzeyindeki saldırılar, pasif ve aktif saldırı olarak sınıflandırılır. Veri gizliliği ihlal edilerek, trafik analizi yapılarak gerçekleştirilen saldırılara **pasif saldırı** denir. Burada temel amaç, cihazların periyodik şekilde dinlenerek konfigürasyon bilgilerinin öğrenilmesidir. Veri bütünlüğüne yönelik saldırılar ise **aktif saldırılar** olarak adlandırılır. Veriler engellenir, içeriği değiştirilir veya yeniden gönderilir. Saldırganların konumuna göre dâhilî ve haricî olarak görülen

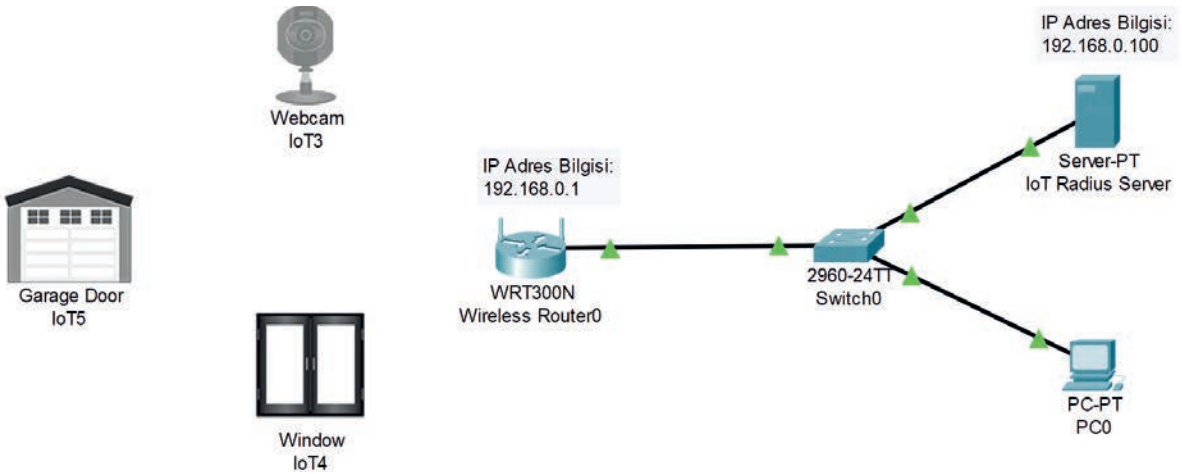
saldırılar da mevcuttur. IoT ağları içinde yer alan noktalardan yapılan dâhilî saldırıların cihaza verdiği zarar daha büyüktür. Haricî olarak yapılan saldırılarda deneme yanılma yöntemi kullanıldığı için saldırganın bilgi toplayıp zarar vermesi zaman alır. Günümüzde özellikle blok zincir ağları, bu tür saldırılara yönelik PoW (Proof of Work), PoS (Proof of Stake) gibi yöntemler kullanır.

IoT cihazlarına yönelik yapılan saldırılara karşı katman bazlı güvenlik mimari yaklaşımı etkili sonuçlar verir. Güvenliğin sağlanması için standart çözümlerin olmaması, siber güvenliğin temel sorunlarındandır. Özellikle cihazların yer aldığı ağın topolojisinin doğru yapılandırılması, alınacak önlemlerin başında gelir.

3. Uygulama

İşlem adımlarına göre IoT Radius sunucusunu yapılandırarak sunucu üzerinden IoT cihazlarında kullanıcı adı ve parola güvenliğinin kontrolünü gerçekleştiriniz.

1. Adım: Ağ simülasyon programını kullanarak Görsel 6.19'daki topolojiyi oluşturunuz.



Görsel 6.19: IoT cihaz bağlantı topolojisi

2. Adım: **IoT Radius Server IP** adres bilgilerini yapılandırınız.

IPv4 Adress: 192.168.0.100
Subnet Mask: 255.255.255.0
Default Gateway: 192.168.0.1

3. Adım: **WRT300N Wireless Router** IP adres bilgilerini yapılandırınız.

IPv4 Adress: 192.168.0.1
Subnet Mask: 255.255.255.0

4. Adım: **PC0** IP adres bilgilerini yapılandırınız.

IPv4 Adress: 192.168.0.2
Subnet Mask: 255.255.255.0
Default Gateway: 192.168.0.1

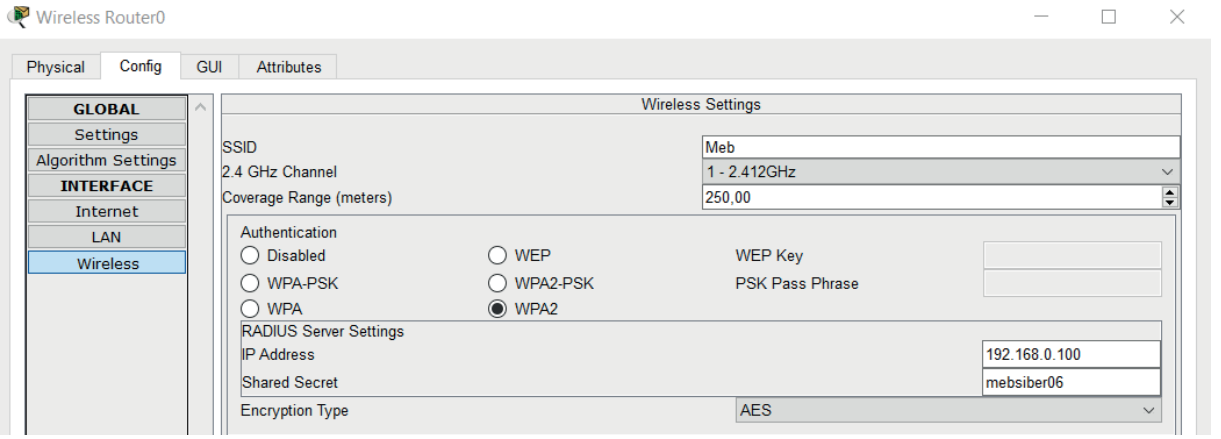
5. Adım: Kablosuz yönlendiriciye (Wireless Router) tıklayınız.

6. Adım: Açılan pencerede **Config** sekmesinin altında bulunan Wireless kısmına tıklayınız.

7. Adım: Açılan **Wireless Settings** penceresinde **SSID** kısmına **Meb** yazınız (Görsel 6.20).

8. Adım: **Authentication** kısmında **WPA2** seçiniz.

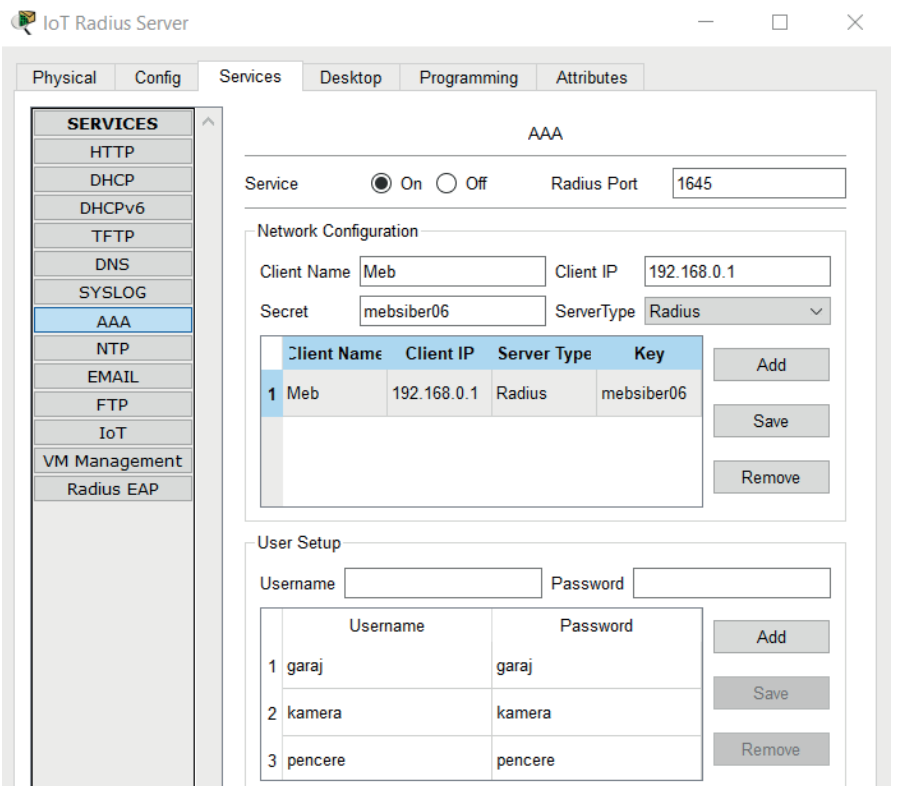
9. Adım: **RADIUS Server Settings IP Address** kutusuna **192.168.0.100** ve **Shared Secret** kutusuna da **mebsiber06** yazınız.



Görsel 6.20: Kablosuz yönlendiricinin kablosuz yayın ve kimlik denetimi ayarlarının yapılması

10. Adım: IoT Radius sunucusuna tıklayınız.

11. Adım: **Services** sekmesinin altında bulunan **AAA** servisine tıklayıp açılan pencereden **Service** özelliğini **On** konumuna getiriniz (Görsel 6.21).



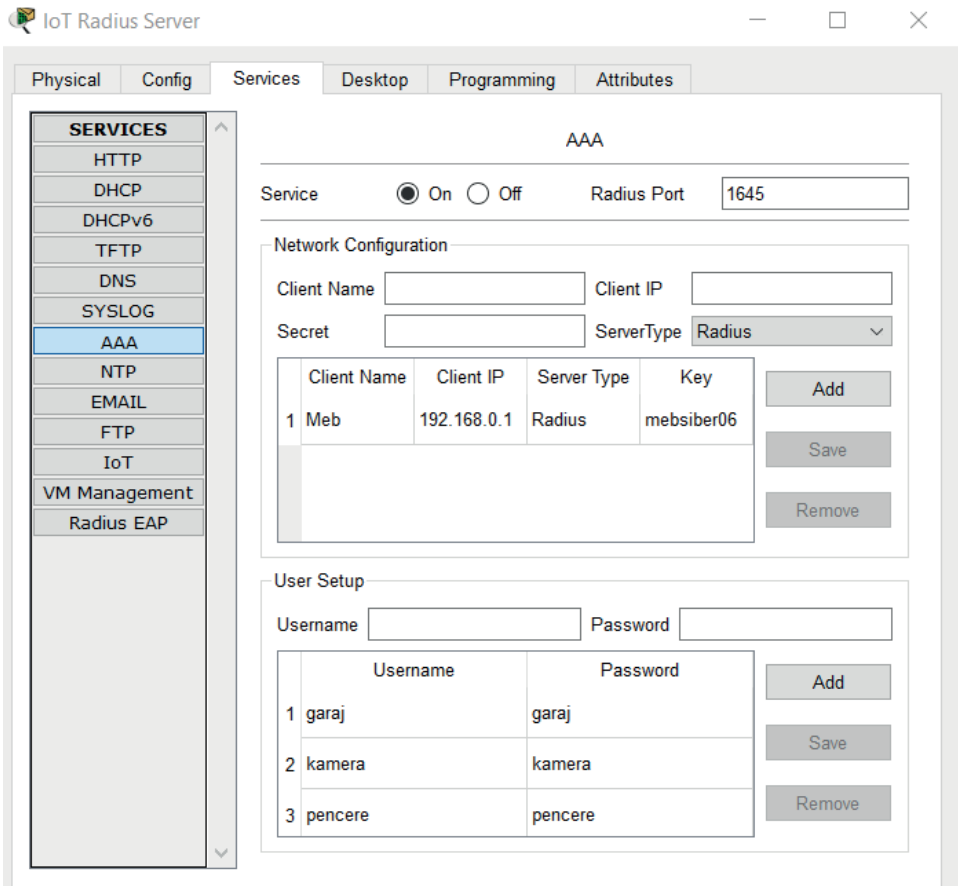
Görsel 6.21: IoT Radius sunucusunda AAA servisinin yapılandırılması

12. Adım: **Network Configuration** bölümünde **Client Name** kutusuna **Meb, Client IP** kutusuna **192.168.0.1** ve **Secret** kutusuna **mebsiber06** yazarak, **ServerType** kısmında da **Radius** seçerek **Add** butonuna basınız.

13. Adım: IoT Radius sunucusuna tıklayınız.

14. Adım: Services sekmesinin altındaki **AAA** servisine tıklayıp IoT ağına bağlı cihazların kimlik denetim bilgilerini Görsel 6.22'de görüldüğü gibi **User Setup** bölümüne giriniz.

Username: kamera Password: kamera
Username: pencere Password: pencere
Username: garaj Password: garaj



Görsel 6.22: AAA servisi için IoT cihazlarının kullanıcı adı ve şifre bilgilerinin tanımlanması

15. Adım: Topolojide kullandığınız IoT cihazlarından **Webcam'a** tıklayınız.

16. Adım: Açılan pencerenin sağ alt kısmında bulunan **Advance** butonuna tıklayınız.

17. Adım: **Config** sekmesine tıklayınız.

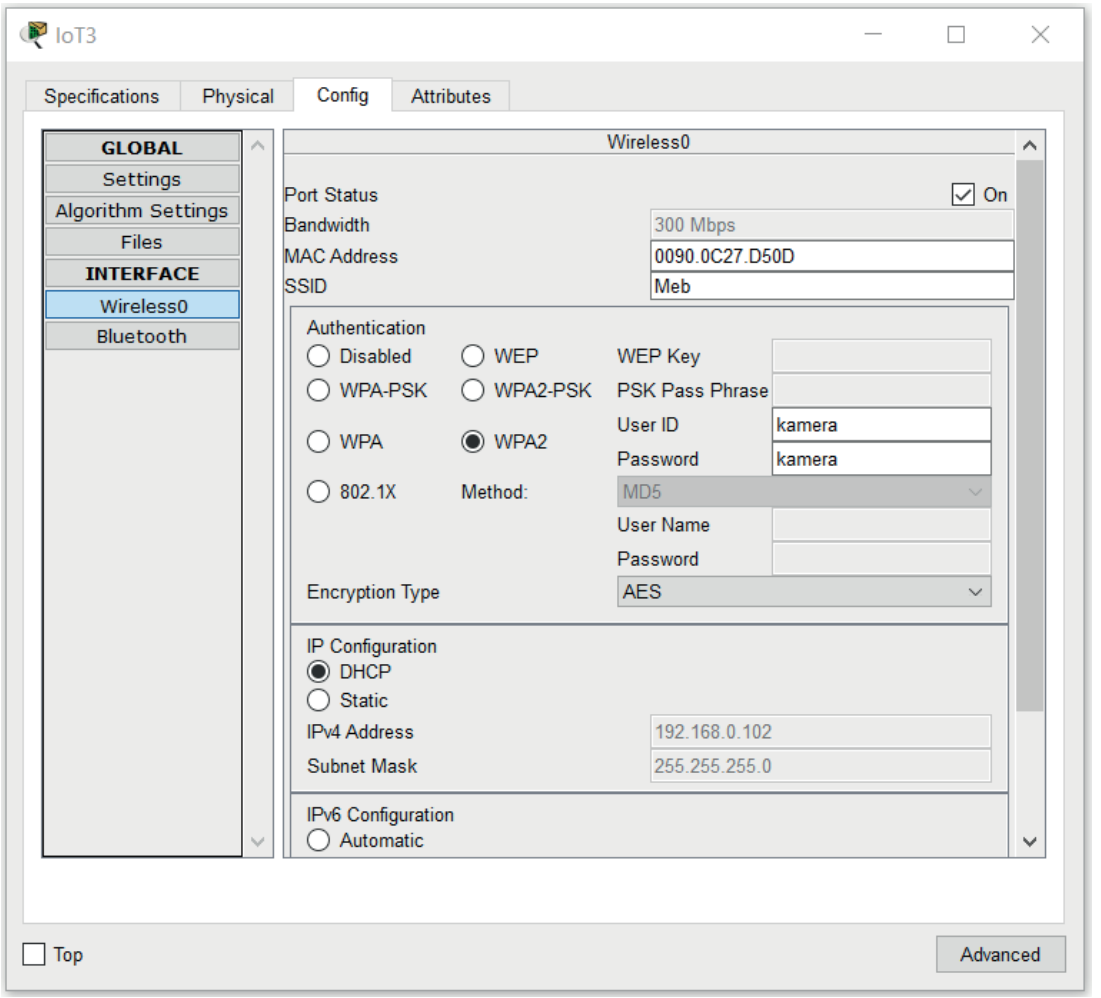
18. Adım: **INTERFACE** kısmının altında bulunan **Wireless0'a** tıklayınız.

19. Adım: Açılan pencere **Port Status** kısmını **On** seçiniz.

20. Adım: **SSID** kısmına **Meb** yazınız.

21. Adım: **Authentication** bölümünden **WPA2** seçiniz.

22. Adım: Açılan ekranda hem **User ID** kutusuna hem de **password** kutusuna **kamera** yazınız (Görsel 6.23).



Görsel 6.23: Webcam IoT cihazının bağlanacağı kablosuz ağ yayın ve parola bilgilerinin girilmesi

23. Adım: Topolojide kullandığınız IoT cihazlarından **Window'a** tıklayınız.

24. Adım: Açılan pencerenin sağ alt kısmındaki **Advance** butonuna tıklayınız.

25. Adım: **Config** sekmesine tıklayınız.

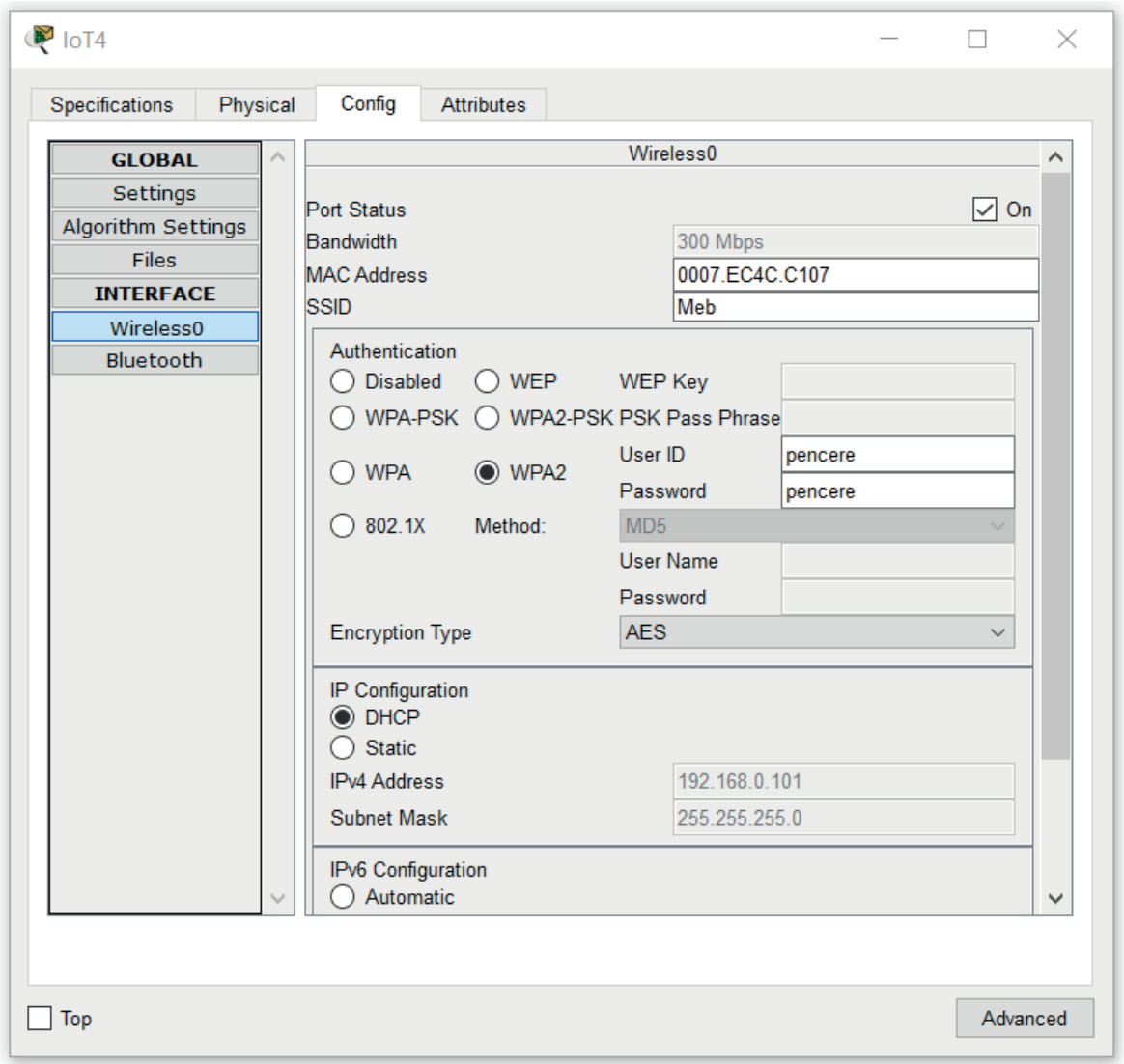
26. Adım: **INTERFACE** kısmının altında bulunan **Wireless0'a** tıklayınız.

27. Adım: Açılan pencere **Port Status** kısmını **On** seçiniz.

28. Adım: **SSID** kısmına **Meb** yazınız.

29. Adım: **Authentication** bölümünden **WPA2** seçiniz.

30. Adım: Açılan ekranda hem **User ID** kutusuna hem de **password** kutusuna **pencere** yazınız (Görsel 6.24).



Görsel 6.24: Window IoT cihazının bağlanacağı kablosuz ağ yayın ve parola bilgilerinin girilmesi

31. Adım: Topolojide kullandığınız IoT cihazlarından **Garage Door'a** tıklayınız.

32. Adım: Açılan pencerenin sağ alt kısmında bulunan **Advance** butonuna tıklayınız.

33. Adım: **Config** sekmesine tıklayınız.

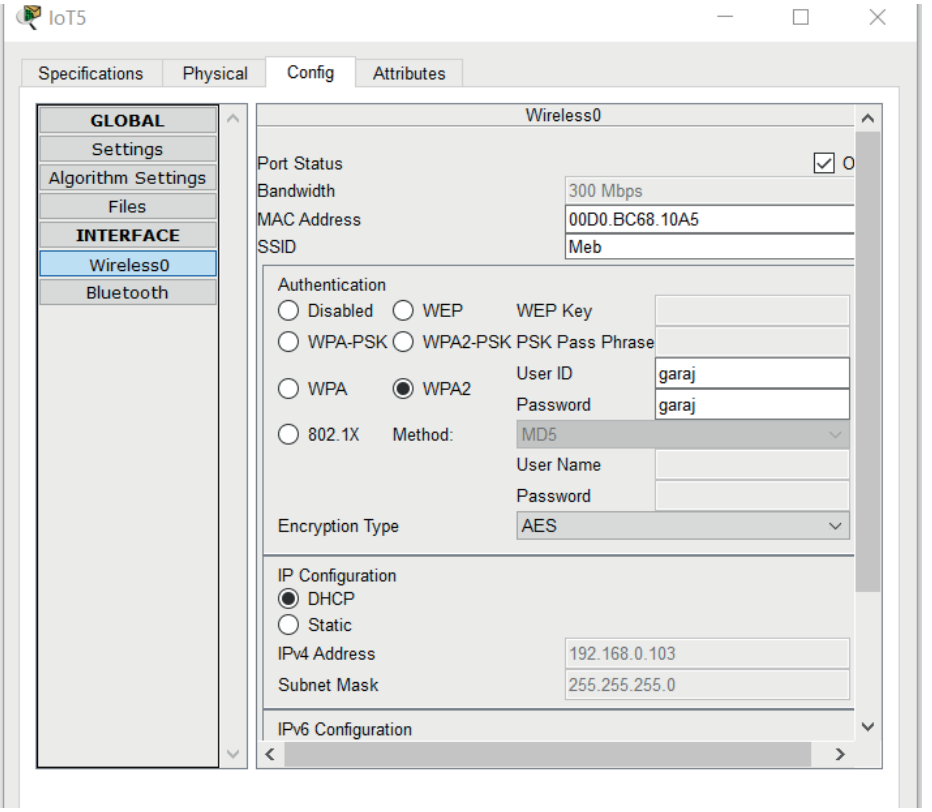
34. Adım: **INTERFACE** kısmının altında bulunan **Wireless0'a** tıklayınız.

35. Adım: Açılan pencere **Port Status** kısmını **On** seçiniz.

36. Adım: **SSID** kısmına **Meb** yazınız.

37. Adım: **Authentication** bölümünden **WPA2** seçiniz.

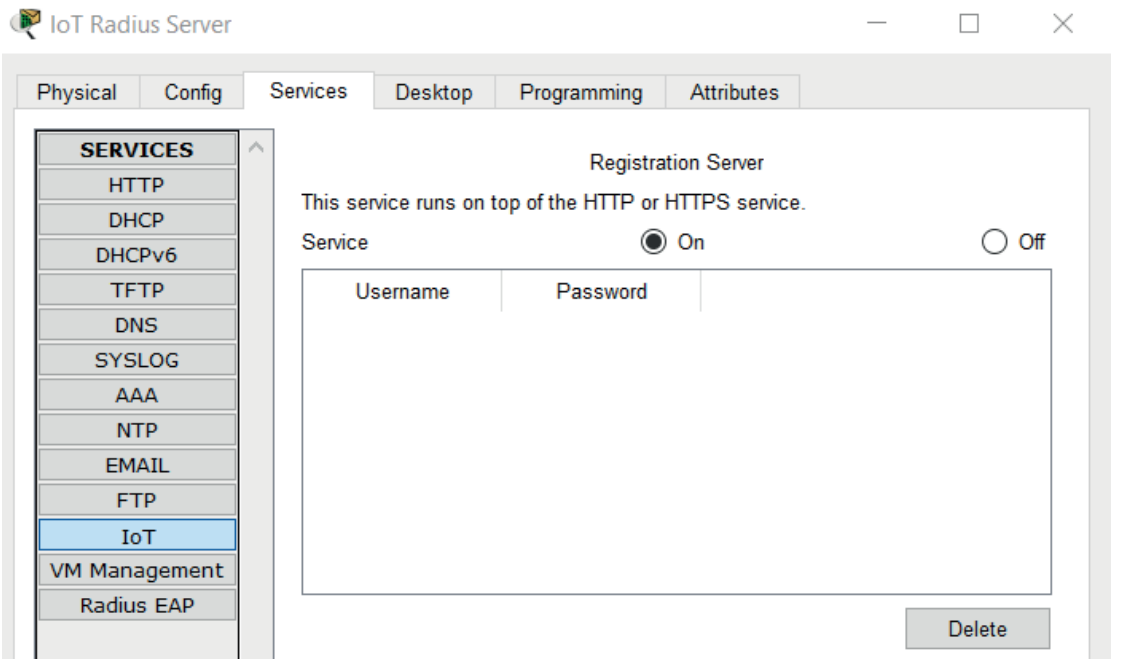
38. Adım: Açılan ekranda hem User ID kutusuna hem de **password** kutusuna **garaj** yazınız (Görsel 6.25).



Görsel 6.25: Garage Door IoT cihazının bağlanacağı kablosuz ağ yayın ve parola bilgilerinin girilmesi

39. Adım: **IoT Radius** sunucusuna tıklayınız.

40. Adım: **Services** sekmesinin altındaki IoT servisine tıklayınız (Görsel 6.26).

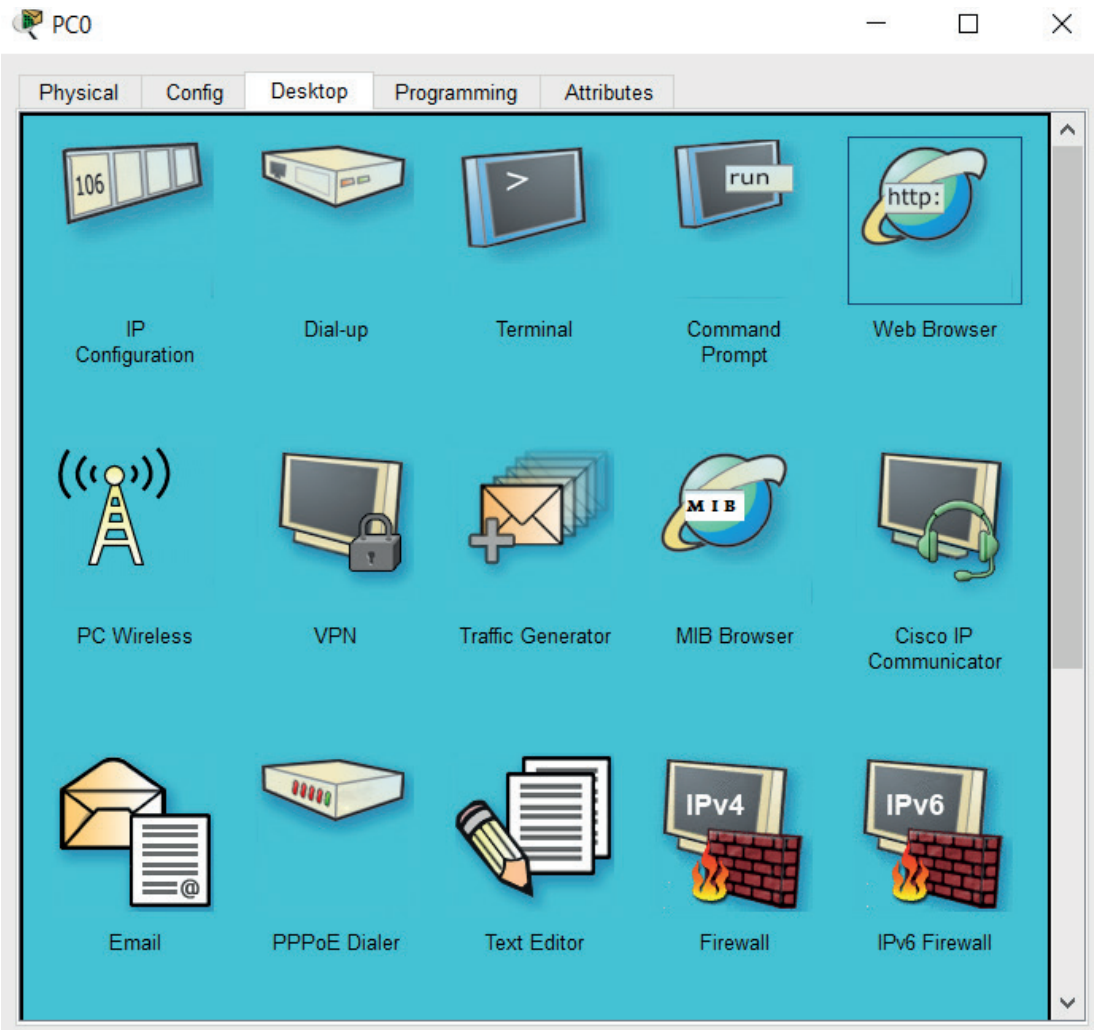


Görsel 6.26: IoT Radius sunucusunda IoT sunucu kayıt servisinin aktif edilmesi

41. Adım: **Registration Server** servisini **On** konumuna getiriniz.

42. Adım: **PC0** istemcisine tıklayınız.

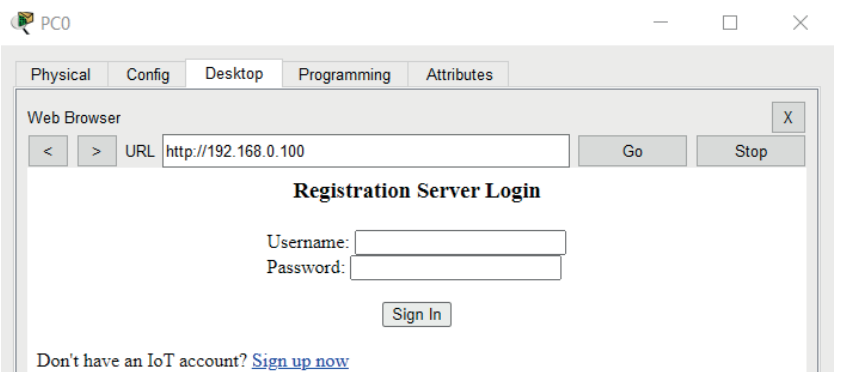
43. Adım: **Desktop** sekmesinin altındaki **Web Browser** simgesine tıklayınız (Görsel 6.27).



Görsel 6.27: PC0 istemcisi ile IoT sunucusuna erişim sağlanması

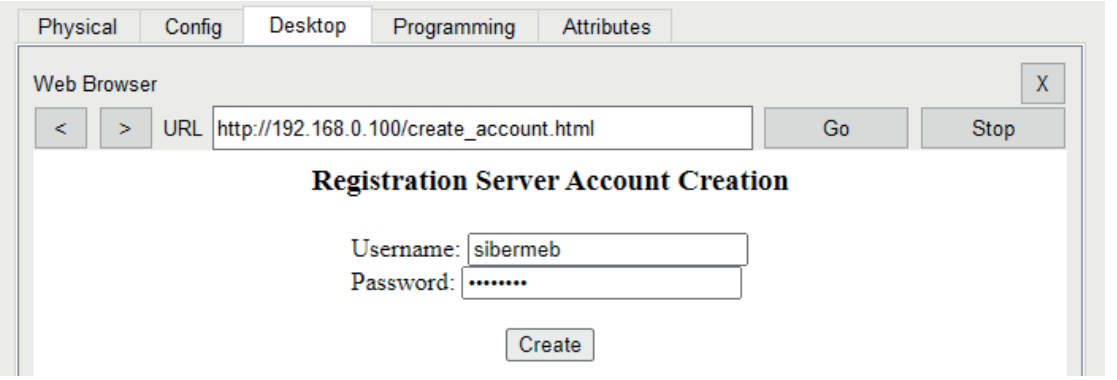
44. Adım: URL kısmına IoT Radius sunucusunun IP adresini yazınız.

45. Adım: Açılan pencerede **Sign up now** linkine tıklayınız (Görsel 6.28).



Görsel 6.28: IoT Radius sunucusundaki bilgilerle IoT cihazlarına erişim sağlanması

46. Adım: Açılan ekranda hem **Username** kısmına hem de **Password** kısmına **sibermeb** yazarak **Create** butonuna tıklayınız (Görsel 6.29).

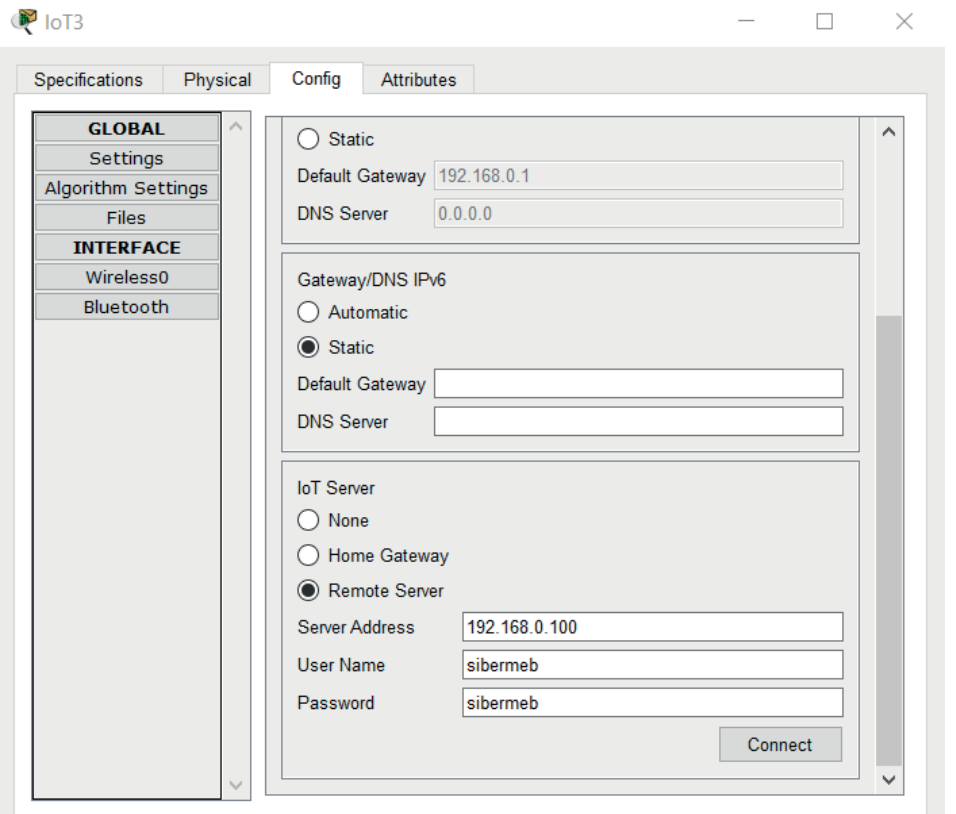


Görsel 6.29: IoT sunucusunda kullanıcı adının ve şifresinin yapılandırılması

47. Adım: **Webcam IoT** aygıtına tıklayınız.

48. Adım: Görsel 6.30'da görüldüğü gibi **Config** sekmesinin altındaki **Global Settings** penceresinde bulunan **IoT Server** erişim ayarlarından **Remote Server** seçeneğini işaretleyerek her IoT cihazı için (Webcam, Window, Garage Door) şu bilgileri ayrı ayrı giriniz:

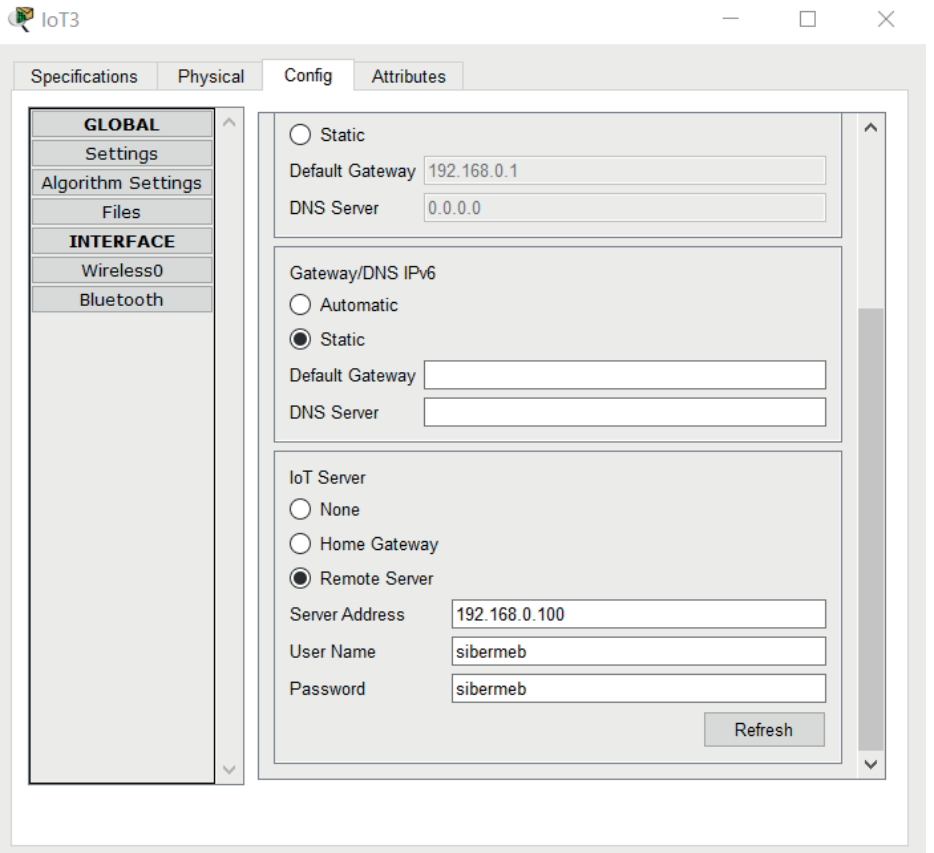
Server Address: 192.168.0.100
User Name: **sibermeb**
Password: **sibermeb**



Görsel 6.30: IoT sunucusunda uzaktan erişim için kullanıcı adının ve şifresinin yapılandırılması

49. Adım: **Connect** butonuna tıklayınız.

Not İşlem basamakları doğru bir şekilde yapılmışsa Görsel 6.31'de görüldüğü gibi **Connect** butonu yerine **Refresh** butonu ekrana gelir.



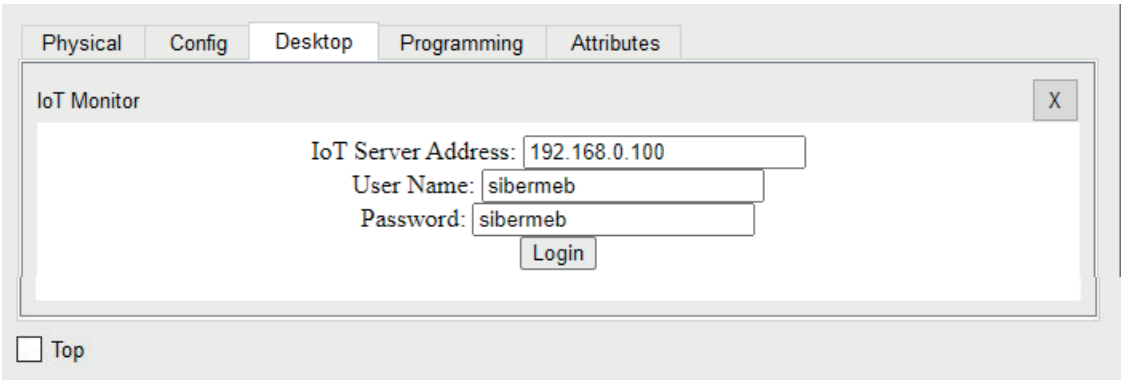
Görsel 6.31: IoT sunucusu bağlanmış IoT cihazı

50. Adım: **PC0** istemcisine tıklayınız.

51. Adım: **Desktop** sekmesinin altındaki **IoT Monitor** simgesine tıklayınız (Görsel 6.32).

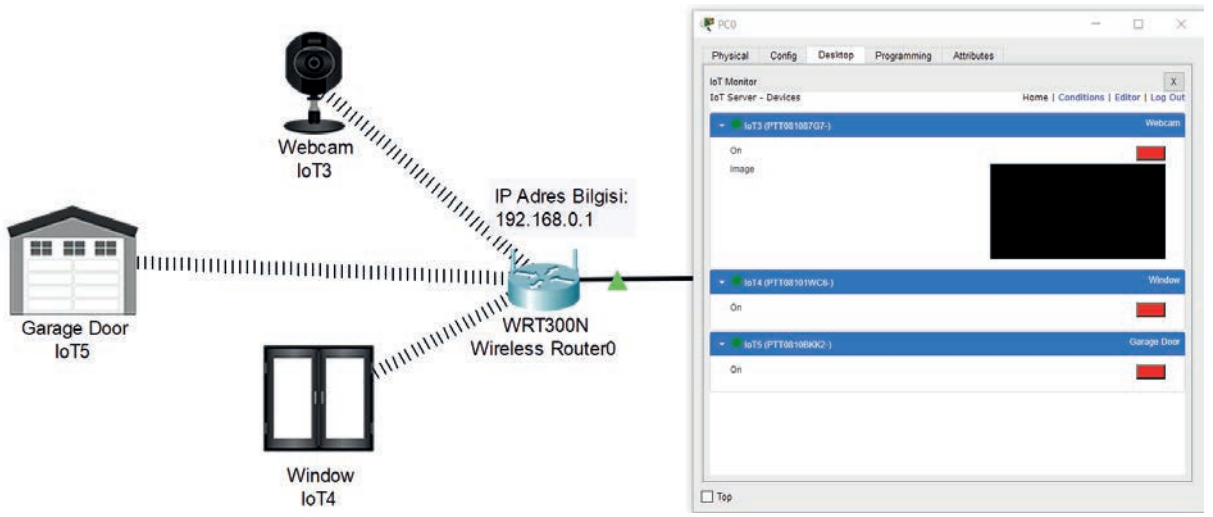
Görsel 6.32: PC0 üzerinden IoT Monitor'e erişim sağlanması

52. Adım: Görsel 6.33'te görülen formda **IoT Server Address** kısmına **192.168.0.100** (IoT Radius sunucu IP adresi) girerek hem **User Name** kutusuna hem de **Password** kutusuna **sibermeb** yazınız.



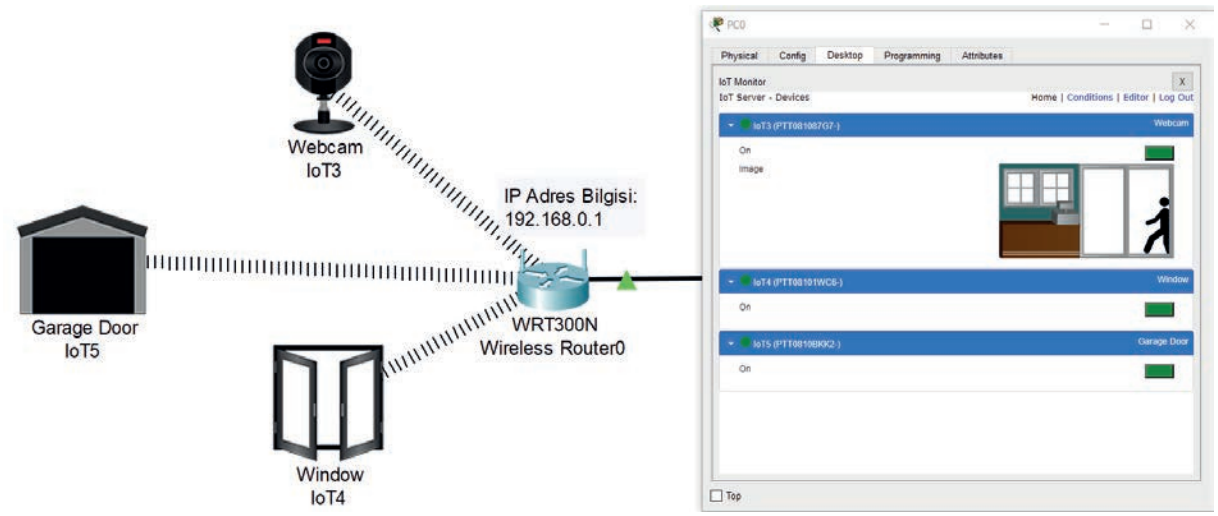
Görsel 6.33: PC0 üzerinden IoT Monitor'e erişim sağlanması

Not Görsel 6.34'te IoT Monitor üzerinden sunucuya bağlı cihazların listesi görülür. Bu aşamada tüm cihazlar **Off** konumundadır.



Görsel 6.34: PC0 üzerinden IoT cihazlarının kapalı konumdaki durumları

Not Görsel 6.35'te görüldüğü gibi IoT Monitor üzerinden sunucuya bağlı cihazlar aktif konuma getirilerek çalıştırılabilir.



Görsel 6.35: PC0 üzerinden IoT cihazlarının açık konumdaki durumları

Not Bu uygulama ile IoT cihazlarına yapılacak olası saldırılara karşı cihazlara erişim bilgilerinin Radius sunucusu üzerinde tutulması ile alınacak önlemler gösterilmiştir.

Okuma Parçası

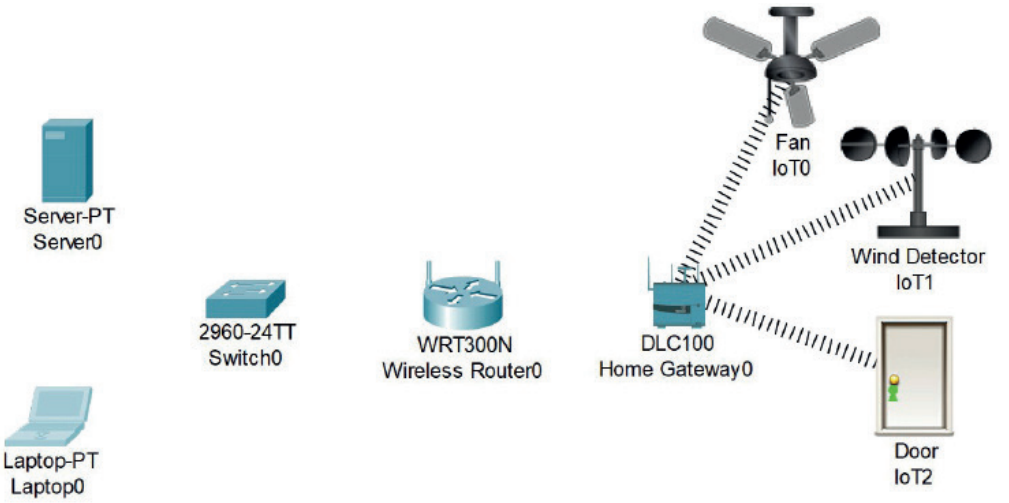
Siber Güvenlikte Uluslararası İşbirliği: Türkiye- AB Ortaklığı

Uluslararası işbirliğine yönelik olarak, Türkiye, başta, NATO kapsamında güvenlik girişimlerinde yer almaktadır. Son dönemde, Milli Güvenlik Kurulu Genel Sekreterliği, NATO CMX-2023 Kriz Yönetim Tatbikatına ve NATO Kilitli Kalkan Tatbikatına katılmıştır. Türkiye- AB ortak siber güvenlik çalışmaları hali hazırda yürütülmeye devam etmektedir. Yakın geçmişte, USOM ve Avrupa Konseyi işbirliğinde ortak siber güvenlik tatbikatları düzenlenmiştir. Yine, USOM'un CyberEast Projesi altında, Romanya'da, 19-21 Nisan 2023 tarihlerinde, "CyberEast: Regional Cybercrime Co- operation Exercise" gerçekleştirilmiştir. Projedeki bu faaliyet kapsamında, Türkiye ve Avrupa devletlerinden siber uzmanlar bir araya gelmiştir (T.C. Ulaştırma ve Altyapı Bakanlığı, 2023). Uluslararası düzeyde etkin çalışmalar yerine getirilse de siber güvenlik alanındaki işbirliği yeterli düzeyde değildir. Siber güvenliği etkili bir şekilde sağlamak için sınır ötesi karşılıklı bağımlılık, bağlantı, koordinasyon ve işbirliğine ihtiyaç vardır. " ...Siber uzayın mekândan bağımsız olması bu alandaki suçluların ulusal sınırların ötesinden faaliyet gösterebilmelerine imkân sağlamaktadır." Bu durumda, siber suçların kaynağına ve suçlulara etkin biçimde ulaşılması için bilgi paylaşımı ve uluslararası iş birliğinin geliştirilmesi önemli hale gelmektedir (T.C. Ulaştırma ve Haberleşme Bakanlığı, 2020).

https://edergi.sanayi.gov.tr/File/Journal/2023/12/12_2023.pdf

Sıra Sizde

Ağ simülasyon programında Görsel 6.36'da görülen IoT ağ topolojisini işlem basamaklarına göre oluşturunuz.



Görsel 6.36: Tasarlanacak IoT ağ topolojisi

1. **Aşağıda verilen bilgilere göre IoT cihazların IP yapılandırmasında boş bırakılan yerleri doldurunuz.**

Server0	Wireless Router0	Laptop0
IPv4 ..... Subnet ..... Default Gateway	IPv4 192.168.0.1 Subnet 255.255.255.0	IPv4 ..... Subnet ..... Default Gateway .....

2. **IoT cihazlarının kablosuz olarak birbirleriyle haberleşmesi için ilgili cihaz üzerinde bazı ayarların yapılması gerekir. Aşağıdaki kutucukları eşleştirerek ilgili yapılandırmayı yapınız.**

Tanımlar	Kavramlar
1. (....) parola	**A. WEP**
2. (....) SSID	**B. TURKIYEM**
3. (....) Authentication	**C. NICE100YILLARA**

3. **Sunucu ve IoT cihazlarında ilgili ayarları yaparak IoT Monitor üzerinden cihazları kontrol ediniz.**

Araştırma

IoT mimarilerine göre saldırılara karşı alınacak önlemlerle ilgili önerilen veya yapılan diğer çalışmaları araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

6.4. IoT GÜVENLİK TESTİ

IoT güvenliği, IoT cihazları ile cihazların bağlı olduğu ağın her türlü saldırıya karşı korunması olarak tanımlanır. IoT ağına bağlı algılayıcı, anahtar, yönlendirici, sunucu ve iş istasyonları, mobil cihazların internete bağlanabilme özelliği dikkate alındığında bu geniş ağ için güvenlik faktörünün önemi ortaya çıkar. IoT sistemlerine yönelik saldırıların cihazlara verdiği zararın çok olmasının sebebi, saldırı sonuçlarının fiziksel sistemlerin yanında sanal sistemleri de etkilemesidir. Birçok IoT üretici firması, cihazlardaki güvenlik zafiyetlerini dikkat almaz ve güncellemeleri göz ardı eder. Bu tür cihazların kullanıldığı IoT sistemleri, varsayılan zayıf kullanıcı adı ve parola bilgilerinin değiştirilmediği durumda da güvenlik riskleri ile karşı karşıya kalır.

Güvenlik testleri, siber güvenlik için önemli bir süreçtir. Bu süreç; test yapılacak sisteme bağlı her türlü ağ, yazılım, uygulama ve donanımlardaki olası zafiyetlerin tespit edilmesini amaçlar. İlgili ekipler tarafından sızma test sürecinin takibi yapılır ve elde edilen sonuçlar güvenlik açığı değerlendirme formları ile kontrol edilir.

IoT güvenlik testleri için öncelikle cihazlar arasında veri akışına engel olan bir durumun tespit edilmesi gerekir. Ortamda kablosuz haberleşen aygıtlar belirlenerek, kablosuz trafiğe engel bir durumun var olup olmadığı gözlemlenebilir. Güvenlik ve akıllı ev sistemlerinde kullanılan IP kamera, ısı ve ışık algılayıcıları, akıllı ev aletleri, temizlik robotları gibi donanımlardaki yetersiz şifreleme ve kimlik doğrulama eksiklikleri, güvenlik açıklarının çıktığı noktalardır. Güvenlik testleri için cihazların bağlı olduğu ağ topolojisinin çıkarılması gerekir. Böylece olası saldırı seviye ve noktaları kolayca tespit edilir. Bu tür başlangıç testleri için açık kaynak dağıtım sistemlerinden biri olan AttifyOS ücretsiz kullanılabilir. AttifyOS, IoT cihazlarına yönelik sızma testlerinin yapılabileceği bir dağıtımdır. Dağıtım, açık kaynak işletim sistemi olan Ubuntu temelinde geliştirilmiştir.

Penetrasyon (sızma), bilişim sistemlerindeki olası güvenlik zafiyetlerinin tespit edilmesi ve bu zafiyetlerden dolayı sistemlerin etkilenme şeklini ortaya çıkaran bir güvenlik testidir. Bu yöntemle saldırganlara karşı güvenlik stratejileri geliştirilir. Düzenli olarak yapılan sızma testleri, sistemlerin güvenlik gereksinimlerini güncel tutar. Bu testler, bilişim sistemleri alanındaki uzmanlar tarafından yapılmalıdır. Yapılan sızma testlerinde sistemlerin zafiyet noktaları belirlenir. Alınan güvenlik önlemlerinin etki seviyesi görülür. Sisteme entegre edilmiş yeni donanım ve yazılım birimleri bu işlemden sonra test edilir. Periyodik yapılan bu testler, verilerin güvenli bir ortamda tutulmasında kullanıcılara güvence verir. IoT ağlarına yönelik güvenlik testleri için de sızma test yöntemleri kullanılır. Sızma testleri üç genel yöntemle ile yapılır:

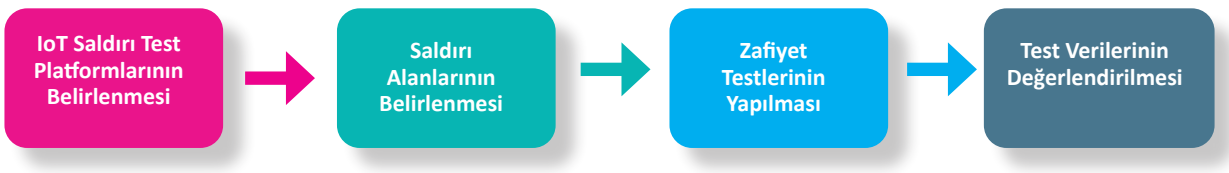
- **Kara Kutu Sızma Testi:** Black Box da denen bu test yönteminde saldırgan, saldırı yapacağı bilişim sistemi hakkında bilgi sahibi değildir. Kara kutu sızma testi, gerçek bir saldırı senaryosu gibidir. Saldırganın sahip olduğu tek bilgi, sistemin dışa açık olan noktalarıdır. Bu senaryoda saldırganın sistemdeki kritik donanım ve yazılımlara saldırıda bulunma yetkisi vardır. Bu testin temel amacı, cihazların

fabrika ayarlarına döndürülmesi (root atma) veya yazılımların kurtarılmasıdır. Kara kutu sızma testi, sistemdeki ciddi güvenlik açıklarının tespiti için önemlidir. Bu test sayesinde saldırganın zamana bağlı olarak sisteme verebileceği zararın boyutu görülür.

- **Beyaz Kutu Sızma Testi:** White Box da denen bu test yönteminde saldırgan, tüm sistem kaynakları hakkında bilgi sahibidir. Beyaz kutu testinde yaygın görülen durum, çalışan olarak kabul edilen saldırganın dış ağdan yaptığı saldırıların sonuçlarının değerlendirilmesidir. Bu testi yapan uzman, sistemdeki donanımsal ve yazılımsal olarak tüm topolojiye hâkimdir. Test öncesi belirlenen görevler aşama aşama gerçekleştirilir.
- **Gri Kutu Sızma Testi:** Grey Box da denen bu testlerde saldırgan, sistem hakkında bilgi sahibi olan personel olarak kabul edilir. Bu test sayesinde saldırganın yetkisiz şekilde sisteme verebileceği zararın boyutu görülür. Ağdaki cihazların eksik ve yetersiz yapılandırma noktaları denetlenir. Siber saldırıların verdiği zararın büyük bir kısmının kurum içi çalışan kaynaklı olduğu düşüldüğünde bu testin önemi ortaya çıkar.

Açık Web Uygulaması Güvenlik Projesi anlamına gelen OWASP (Open Web Application Security Project), web tabanlı uygulamaların güvenliği gibi alanlarda çalışmalar yürüten açık kaynaklı bir bilgi güvenliği topluluğudur. OWASP; bilişim sistemlerindeki güvenlik açıklarının sebebi, oluşma şekli ve alınacak önlemlerle ilgili çalışmalar yürütür. Yürütülen çalışmalar, ihtiyaç duyulan güvenlik gereksinimleri doğrultusunda geliştirilir. Platformlara yönelik (web, IoT ağları, uygulamalar vb.) güvenlik riskleri listelenerek periyodik olarak paylaşılır. Bu topluluk tarafından oluşturulan araç ve dokümanlar ücretsiz sunulur.

IoT ağlarına yönelik gerçekleştirilecek sızma testleri için öncelikle bilgisayar ağ ve güvenliği alanında uzmanlık gerekir. Cihazların arayüz ayarları ve temel ağ arayüz testleri hakkında bilgi toplanmalıdır. Bu noktada arka kapı (backdoor) arayüzlerinin tespiti için çeşitli uygulamalar kullanılır. IoT cihazların birçoğu Linux tabanlı özel gömülü işletim sistemine sahip olduğu için Linux kullanımı hakkında bilgi sahibi olunması önemlidir. Linux işletim sistemine sahip olmayan IoT cihazlar için de **tersine mühendislik (reverse engineering)** bilgisi gerekir. Görsel 6.37'de IoT sızma testleri için izlenen aşamalar verilmiştir. İlk olarak IoT cihazlarının kullanıldığı alan ve görevleri belirlenir. Bu alanlar; akıllı şehirler, kişisel kullanılan teknolojiler, endüstriyel üretim ortamları, enerji üretim ve iletim sistemleri olabilir. Atak yüzeyi olarak tanımlanan saldırı alanı belirlenerek cihaz tabanlı zafiyetler listelenir. IoT cihazın bağlı olduğu ağ, cihaz belleği, arayüzü gibi güvenlik açıklarının tespiti gereklidir. Güvenlik açıklarına yönelik testlerde cihaz ekosistemindeki zafiyetler; OWASP tarafından da tavsiye edilen yetersiz yetkilendirme, güvensiz servisler, yetersiz yapılandırma gibi noktalar değerlendirilir. Sızma testi sonucu elde edilen veriler, zafiyetlerin derecesi ve nasıl ortadan kaldırılacağı açısından değerlendirilir.



Görsel 6.37: IoT sızma testi aşamaları

- **IoT Platformunun Belirlenmesi:** IoT ağlarının sürekli genişlemesi, cihaz sayısı ve kullanım alanlarına göre çeşitlenmesine sebep olur. Akıllı evler ve kişisel alan ağlarında kullanılan IoT cihazlarından üretim tabanlı endüstriyel IoT cihazlarına kadar tüm algılayıcıların risk alanları belirlenir. Bu alanlara

yönelik zafiyetler listelenerek sızma testlerinin kapsamı ortaya çıkarılır. Özellikle e-sağlık, enerji üretim ve iletim, akıllı şehir uygulamaları, kontrol ve takip uygulamalarında kullanılan IoT cihazları, **kritik altyapı** olarak adlandırılan sızma test platformlarıdır.

- **Saldırı Alanının Belirlenmesi:** Saldırganın doğrudan etkilediği kapsamı belirtir. IoT cihazlarının oluşturduğu ekosisteme yönelik saldırı alanında kalan ve bu saldırıdan etkilenen cihazların sayısı ile etkilenme oranı, cihazlarda alınan önlemlere göre değişir. IoT cihazlarının düşük bellek kapasiteleri, zayıf arayüz güvenlik protokolleri, güncel olmayan cihaz yazılımları, ağ trafiği gibi etkenler saldırganın bu alanda kullandığı noktalardır. Algılayıcıların donanımsal olarak basit anahtarlama şeklinde çalıştığı IoT ağlarında kimlik doğrulama ve yetkilendirme gizliliğinden kaynaklanan saldırıların verdiği zarar artar.
- **Zafiyet Testlerinin Yapılması:** Siber saldırı yöntem ve sayısı, her geçen gün artar. Bu artışta saldırganın motivasyonu ve sistemde görülen zafiyetler önemli etkenlerdir. OWASP tarafından periyodik olarak paylaşılan güvenlik açıkları, bu açıkların oluşma sebebi ve alınacak önlemleri içeren çalışmalar tüm dünyada kabul görür. IoT cihazlarında belirlenen genel güvenlik açıklarına yönelik yapılan testlerde dikkat edilmesi gereken noktalar vardır. Cihazların sahip olduğu yetersiz fiziksel güvenlik protokolleri, güvensiz firmware yazılımları, güvensiz web veya mobil arayüzleri, ağ servisleri gibi noktalarda güvenlik testleri öncelikli olarak yapılmalıdır. Yetkilendirme ve kimlik doğrulamanın yetersiz olduğu ve şifreli bağlantıdan kaynaklanan güvenlik açıkları da dikkat edilmesi gereken noktalardır.
- **Verilerin Değerlendirilmesi:** Sızma testi sonucunda belirlenen riskler, saldırı yöntem ve şekli hakkında bilgi verir. Bu noktada saldırının olası etkisinin tahmini ile riskler derecelendirilir. Güvenlik açıklarının giderilmesine yönelik yapılacak çalışma listesi oluşturulur. Risk derecelendirme modeli bu aşamada mutlaka hazırlanmalıdır. Periyodik olarak yapılan sızma testleri, risk analiz sonuçlarının yorumlanmasını sağlar.

Okuma Parçası

EĞİTİMDE METAVERSE VE NESNELERİN İNTERNETİ UYGULAMALARININ İNCELENMESİ

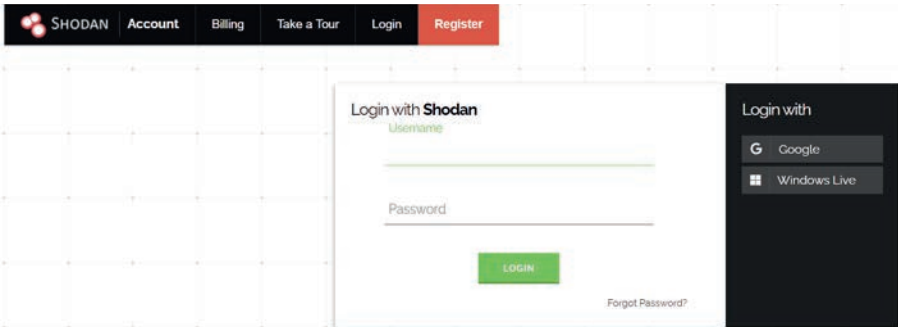
Bilişim teknolojilerinde ortaya çıkan hızlı gelişim ve dönüşümler beraberinde birçok alanda toplumsal değişiklikleri getirmektedir. **Toplum 5.0** kavramı bilgi toplumunun oluşmasının bir sonucu olarak ortaya çıkan dijital dönüşümün geldiği nokta olarak tanımlanmaktadır. Bu dönüşüm sonucu eğitim ve öğretim faaliyetleri günümüzde teknolojik gelişmelerin hızlı bir şekilde uygulandığı alanların başında gelmektedir. Yeni teknolojilerin eğitimde kullanılmasıyla kesintisiz öğrenme ortamları yaygınlaşmaktadır. **Metaverse** kavramı, insanların kendilerini bir avatar aracılığı ile ifade ettiği, bağlı oldukları bilişim ağı üzerinden nesneler ve diğer kullanıcılar ile etkileşim içinde bulunduğu, gerçek yaşamlarını örnekledikleri bir sanal evren olarak tanımlanmaktadır. **Nesnelerin İnterneti** de yeni bir kavram olmakla birlikte hızla yaygınlaşan, algılayıcıların birbirleri ile haberleşmesiyle oluşan bir bilişim ağını ifade eder. Her iki kavramın da birçok farklı teknolojiyi üzerinde barındırmasından dolayı eğitimde kesintisiz öğrenme ortamlarının oluşturulması, deney ve deneyime dayalı eğitim ortamlarının yaygınlaştırılmasındaki etkisi büyüktür. Özellikle senaryolaştırılmış öğrenme içeriklerinin hazırlanmasıyla kullanıcı etkileşimli öğrenme fırsatlarının yaratılması sonucu, deneyimle sahadan bilgi toplanarak eğitim ortamının oluşturulması Metaverse ve Nesnelerin İnterneti teknolojileri ile mümkün olmaktadır. Sanal ortamda oluşturulan dijital dünyalar sayesinde bilginin bu sanal evrende uygulanması, alternatifli senaryolar sayesinde deneyerek öğrenme imkânları, mekân, zaman ve maliyet gibi parametreleri olumlu etkilemektedir.

https://ttkbyayin.meb.gov.tr/yayin/175

4. Uygulama

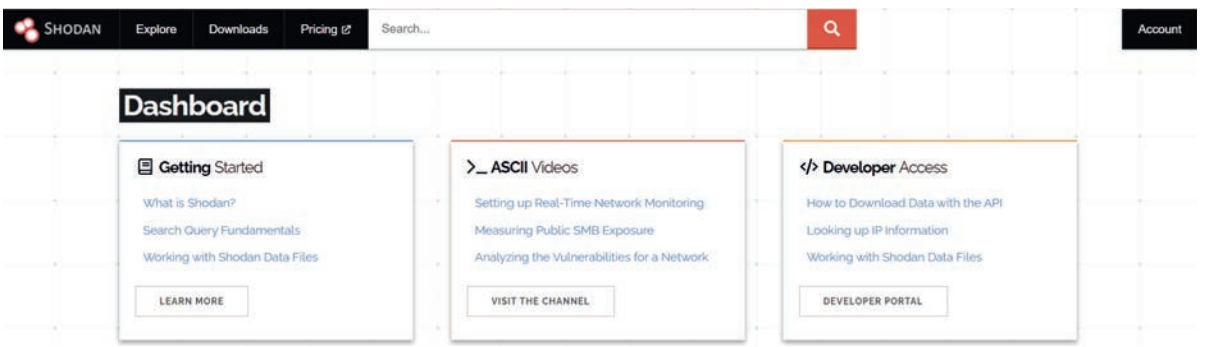
İşlem adımlarına göre internet üzerindeki cihazları taramak ve analiz etmek için kullanılan bir arama motoru aracılığıyla güvenlik kameralarındaki yazılım ve donanım zafiyetleri tespit ediniz.

1. Adım: İnternet tarayıcınız aracılığı ile www.shodan.io adresine bağlanınız. Login kısmına tıklayıp ekrana gelen Register (Kayıt) menüsünden kayıt işlemini gerçekleştiriniz (Görsel 6.38).



Görsel 6.38: IoT cihazları taramak ve analiz etmek için kullanılan arama motoruna kayıt arayüzü

2. Adım: Kayıt işleminin ardından ekrana gelen panelde arama motoru kullanımı ve yapılacak sorgulama formatları hakkındaki bilgileri inceleyiniz (Görsel 6.39).

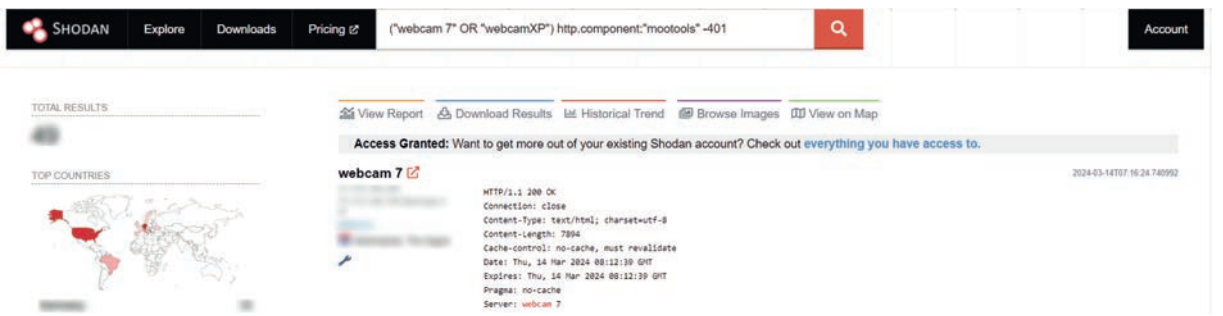


Görsel 6.39: İnternet üzerinden cihazları aramak için kullanılan site gösterge paneli

3. Adım: Search (Arama) çubuğuna ("webcam 7" OR "webcamXP") http.component:"mootools" -401 yazınız.

Not

Üçüncü adımdaki arama cümlesi yazılarak internete bağlı bir web kamera içeren IP'lerin tarama işlemi yapılır. **http/1.1 200** OK verisi içeren arama sonuçlarına göre yetkisiz erişim izni olan IoT cihazları incelenir (Görsel 6.40).

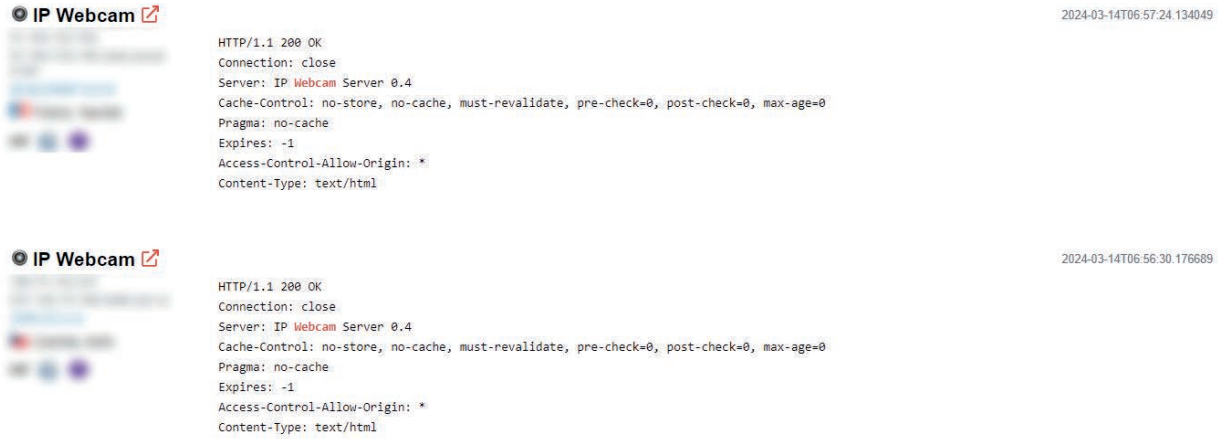


Görsel 6.40: Zafiyet içeren web kameralarının tarama işlem sonuçları

4. Adım: Arama çubuğuna “Server: IP Webcam Server” “200 OK” yazınız.

Not

Dördüncü adımdaki arama cümlesi yazılarak, internete bağlı IP kameralardan çevrimiçi olup 200 OK yanıt kodu ile sorunsuz bir şekilde çalışanların bulunup bulunmadığı sorgulanır (Görsel 6.41).



Görsel 6.41: Arama motoru ile IP kameraları tarama işlemi

Sıra Sizde

Cihazları taramak ve analiz etmek için kullanılan arama motoru ile aşağıdaki sorguları gerçekleştirerek sorgu sonuçlarını yorumlayınız.

“Server: Prismview Player”	http.title:"Tesla PowerPack System" http.component:"d3" -ga3ca4f2
title:"Weave Scope" http.favicon. hash:567176827	"Server: gSOAP/2.8" "Content-Length: 583"
“Server: AV_Receiver” “http/1.1 406”	"Cobham SATCOM" OR ("Sailor" "VSAT")
"\x08_airplay" port:5353	http.title:"Nordex Control" "Windows 2000 5.0 x86" "Jetty/3.1 (JSP 1.1; Servlet 2.2; java 1.6.0_14)"
“Model: PYNG-HUB”	"Siemens, SIMATIC" port:161
"Chromecast:" port:8008	"Server: Microsoft-WinCE" "Content-Length: 12581"
"in-tank inventory" port:10001	"HID VertX" port:4070
mikrotik streetlight	title:"xzeres wind"
	"html:"PIPS Technology ALPR Processors""

Araştırma

OWASP topluluğu tarafından her yıl açıklanan IoT ağlarına yönelik güncel güvenlik açıklarını gösteren listeyi internet üzerinden araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

ÖLÇME VE DEĞERLENDİRME

A. Aşağıdaki cümlelerin başındaki paranteze verilen bilgiler doğruysa "D", yanlışsa "Y" yazınız.

1. ( ) Nesnelerin İnterneti kavramı 1999 yılında Kevin Ashton tarafından literatüre eklenmiştir.

2. ( ) IoT ekosistemindeki veri toplama katmanında algılayıcılar aracılığıyla veriler toplanır.

3. ( ) Güvenlik ve gizlilik problemleri IoT teknolojilerindeki dezavantajların başında gelir.

4. ( ) Wi-Fi, IoT cihazlarının desteklemediği kablosuz iletişim protokolüdür.

5. ( ) IP kamera, ısı ve ışık algılayıcıları, akıllı ev aletleri, temizlik robotları gibi donanımlardaki yetersiz şifreleme ve kimlik doğrulama eksiklikleri zafiyet noktalarının başında gelir.

B. Aşağıdaki cümlelerde boş bırakılan yerlere doğru sözcükleri yazınız.

6. Otomatik olarak görevlerini yerine getiren, bir ağ üzerinden fiziksel olarak birbirine bağlanan cihazların oluşturduğu ağa ..... denir.

7. Büyük çaplı botnet tabanlı ..... saldırılarına karşı SYN çerezleri gibi önlemler alınır.

8. Elektromanyetik yayılım yoluyla cihazlardaki verilerin ele geçirilmesine ..... adı verilir.

9. AttifyOS, IoT cihazlarına yönelik sızma testlerinin yapabileceği bir dağıtımdır. Dağıtım, açık kaynak işletim sistemi olan ..... temelinde geliştirilmiştir.

10. Açık kaynaklı bir veri protokolü olan ..... gelişmiş mesaj sıralama anlamına gelir.

C. Aşağıdaki soruları dikkatlice okuyarak doğru cevabı işaretleyiniz.

11. Aşağıdaki ağ türlerinin hangisinde kullanıcı farklı işlevlere sahip kişisel cihazlarını iletişim protokolleriyle birbirine bağlayarak dar bir alanda aktif olarak çalışabilir?

A) CAN B) LAN C) NET
D) PAN E) WAN

12. Servis odaklı mimarinin oluştuğu katman sayısı aşağıdakilerden hangisidir?

A) 2 B) 3 C) 5
D) 7 E) 9

13. Aşağıdakilerden hangisi güvenlik algoritması ağ ve uygulama katmanlarında kullanılır?

A) AES B) DSA C) ECDH
D) RSA E) 3DE

14. Aşağıdakilerden hangisi sızma testlerinde kullanılan yöntemlerden biridir?

A) Beyaz kutu sızma testi B) Lacivert kutu sızma testi C) Mavi kutu sızma testi
D) Sarı kutu sızma testi E) Turuncu kutu sızma testi

15. Aşağıdakilerden hangisi IoT iletişim protokollerinden <u>değildir</u>?

A) CoAP B) IXP C) LoRaWAN
D) MQTT E) NFC

D. Aşağıdaki soruların cevabını ilgili alana yazınız.

16. Nesnelerin İnternetinde güvenlik mimarilerini açıklayınız.

17. IoT ağlarına yönelik yapılan saldırı türleri ve saldırı alanları nelerdir?

18. Nesnelerin İnternetinde benimsenen güvenlik tedbirlerini açıklayınız.

19. IoT ağlarında kullanılan veri protokolleri nelerdir?

20. IoT cihazları için kullanılan iletişim protokollerini açıklayınız.

21. Nesnelerin İnternetinde güvenlik zafiyetlerine yönelik alınacak önlemler nelerdir?

22. IoT ağlarında yapılan sızma testi aşamalarını açıklayınız.

BULUT BİLİŞİM GÜVENLİĞİ

7. ÖĞRENME BİRİMİ

KONULAR

7.1. BULUT MİMARİSİ VE ALTYAPI GÜVENLİĞİ

7.2. BULUT GÜVENLİĞİ VE RİSK YÖNETİMİ

7.3. BULUT BİLİŞİMDE VERİ GÜVENLİĞİ

7.4. KİMLİK YÖNETİMİ VE DENETİMİ

ANAHTAR KELİMELER

Bulut bilişim, bulut güvenliği, bulut tabanlı hizmetler, Cloud System, erişim kontrolü, kimlik doğrulama, veri güvenliği, veri izleme ve denetim, veri izolasyonu, veri şifreleme, yetkilendirme

NELER ÖĞRENECEKSİNİZ?

- Bulut bilişim temel kavramlarını açıklama
- Bulut ortamında veri güvenliğini sağlama
- Kullanıcı kimlik doğrulama, yetkilendirme ve roller temelli erişim kontrolünü yapma
- Bulut ortamındaki ağ güvenliği önlemlerini alma ve güvenli ağ tasarımını yapma
- Bulut bilişim güvenliğinin sağlanması için uygulamaları ve güvenlik testlerini yapma
- Güvenli bir bulut hizmet sağlayıcısının seçimi için kriterleri ve değerlendirme yöntemlerini belirleme
- Yedekleme ve felaket kurtarma sistemlerinin tekniğini kavrama

HAZIRLIK ÇALIŞMALARI

1. Bulut bilişim size neleri çağrıştırıyor? Düşüncelerinizi arkadaşlarınızla paylaşınız.
2. Bulut bilişim güvenliğinde ne gibi tehditler olabilir? Düşüncelerinizi arkadaşlarınızla tartışınız.

7.1. BULUT MİMARİSİ VE ALTYAPI GÜVENLİĞİ

Bulut bilişim; internet üzerinden erişilen, depolanan ve paylaşılan verilerle ilgili bir hizmettir. Bu veriler, kullanıcıların fiziksel aygıtlarında yer almamakla birlikte sunucularında da tutulmaz. Veriler, bulut hizmeti veren sağlayıcının sunucularında barındırılır. Bulut Bilişim Güvenliği (Cloud Computing Security) ise bulut tabanlı ortamlarda erişim, uygulama, veri ve hizmetlerin güvende tutulmasını sağlamaya yönelik oluşturulan bir söz dizilimidir.

Bulut mimarisi; bilişim teknolojileri ve teknikleri arasında veri depolama, veri işleme, sunucu kaynakları hizmetlerine dayanan, bu hizmetlerin internet üzerinden erişimini ve yönetimini sağlayan bir sistemdir. Bulut mimarisi, kullanıcıların fiziksel cihazlarında tek bir aygıta mahkûm kalmadan internetin erişebileceği her aygıt üzerinden hizmet almasını ve işlerini yönetmesini sağlayan konumdadır. Bulut mimarisi genellikle internet tabanlı hizmet ve uygulamaların işlendiği, verilerin depolandığı uzak sunucu kümeleri diye tanımlanabilir (Görsel 7.1).

Görsel 7.1: Bulut yönetimi

7.1.1. Bulut Mimarisi Modeli

Bulut mimarisi şu ana hizmet modelleri üzerine kuruludur:

- **Yazılım Hizmetleri (Software as a Service-SaaS):** Kullanıcıların uygulamalara erişimini internet üzerinden sağlayan bir hizmet modelidir. Bu hizmetler, yazılımı barındıran ve aynı zamanda da güncelleyen bulut hizmet sağlayıcıları tarafından yönetilir. Kullanıcılar, bir yazılımı fiziksel cihazlarına kurmak

ve cihazında yönetmek yerine internet üzerinden bu hizmete erişebilirler (Görsel 7.2). Üç boyutlu çizimler yapılmasını sağlayan Autodesk Fusion 360, dosyaları sunucuda barındıran Google Drive, evrak işlerinin yönetilmesini sağlayan Microsoft 360, yazılım hizmetlerine örnek olarak verilebilir.

Görsel 7.2: Bulut erişim modeli

- **Platform Hizmetleri (Platform as a Service-PaaS):** Kullanıcıların, kendi yazılımlarını geliştirmek için yararlanacağı platformların altyapısını ve araçlarını bulut server üzerinde tutan mimari modele verilen isimdir (Görsel 7.3). Kullanıcılar, uygulama ve yazılımlarını geliştirirken bu altyapı hizmetlerini ve platformlarını kullanır. Bu kullanım sırasında altyapının içeriği ve mimari modeliyle ilgilenmezler. Platform hizmetlerine Google App Engine, Microsoft Azure vb. örnek olarak verilebilir.

Görsel 7.3: Bulut platform hizmetleri

- **Altyapı Hizmetleri (Infrastructure as a Service-IaaS):** Kullanıcılar tarafından sanal sunucu, depolama, ağ ve diğer kaynaklar kiralanır. Kiralanan bu unsurlar, bulut bilişimin temel altyapı gereklilikleridir. Hizmeti kiralayan kullanıcılar, bu sanal altyapıları kendi ihtiyaçları doğrultusunda yapılandırabilir ve yönetebilirler (Görsel 7.4). Bu hizmet sağlayıcılara Google Cloud Platform (GCP), Microsoft Azure, Amazon Web Service (AWS) vb. örnek olarak verilebilir.

Görsel 7.4: Tasarlanabilir bulut altyapısı

7.1.2. Bulut Mimarisinin Avantajları

Diğer veri tabanı uygulamalarına göre bulut mimarisinin oldukça fazla avantajı vardır. Bu avantajlar şu şekilde sıralanabilir:

- **Maliyet Etkinliği ve Kısıtlanabilirlik:** İhtiyaca göre hizmet alınabilir ve ihtiyaç olmayan hizmetler, istendiği zaman ölçeklendirilerek kullanımdan kaldırılabilir. Bu sayede kullanıcılara standart bir ödeme planından ziyade ihtiyaca yönelik esnek bir ücretlendirme politikası sunulur. Böylece uygulamaların ve alınan hizmetlerin yüksek maliyetleri azaltılır.
- **Yüksek Erişebilirlik Teknolojisi:** Bulut bilişim yönetiminde birden fazla veri merkezi ve sunucu kullanılır. Böylece gereksiz sunucu çökmelerinin önüne geçilir. Çoklu veri erişimi sayesinde bir veri merkezi veya sunucuda oluşabilecek sorunlarda diğer veri merkezleri, görevi üstlenebilecek bir mimari modele sahiptir.
- **Esneklik ve Ölçülebilirlik:** Bulut bilişim hizmeti alan kullanıcılar, ihtiyaçları doğrultusunda hizmet kaynaklarını istedikleri gibi artırabilir veya azaltabilir.
- **Veri Güvenliği ve Yedeklenebilirlik:** Bulut sağlayıcıları tarafından özellikle verilerin yedeklenmesine ve şifreli tutulmasına özen gösterilir. Sunucularda oluşabilecek bir sorundan dolayı veri kaybının telafisi olmayacağı için veriler sürekli yedeklenir.

Not

Bulut mimarisi hizmeti alan kullanıcılar, veri gizliliklerine ekstra önem vermelidir. Kişiler veya kurumlar özellikle kişisel verilerinin korunması konusunda, hizmet aldıkları sağlayıcının güvenirliliğinden emin olmalı ve uygun güvenlik önlemlerini alarak veri güvenliklerini korumalıdır.

Araştırma

Bulut bilişim güvenliğinin mevcut tehditlerini ve bu tehditlerin engellenmesindeki zorlukları araştırınız.

7.2. BULUT GÜVENLİĞİ VE RİSK YÖNETİMİ

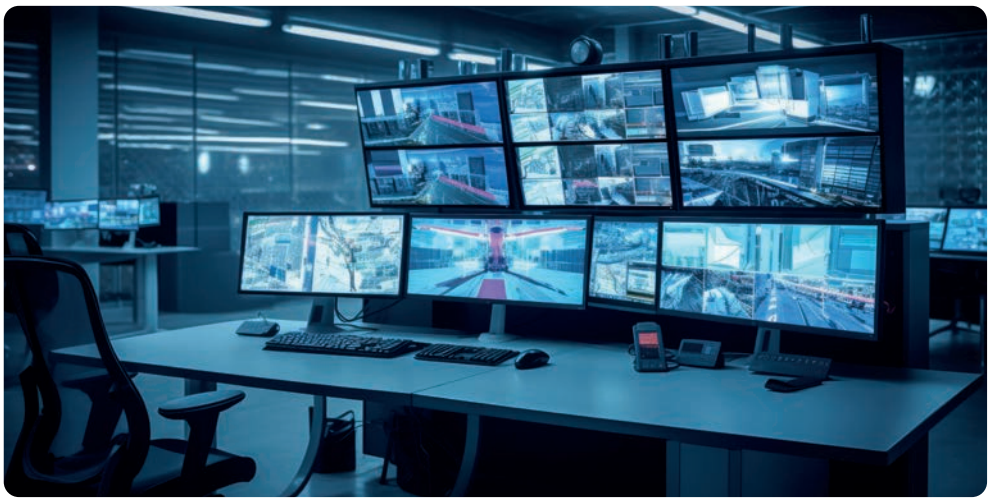
Bulut bilişim (Cloud Computing) altyapı güvenliği, bulut tabanlı olarak çalışan hizmetleri kullanırken verilerin güvenliği ve sistemsel güvenliği esas alan bir dizi ilkeyi içerir (Görsel 7.5). Bu ilkeler; fiziksel güvenlik, veri erişim kontrolü, veri şifreleme, denetim ve izleme, yedekleme ve kurtarma, güvenlik duvarları, sözleşme ve yetkilendirme olarak sıralanabilir.

Görsel 7.5: Bulut güvenlik

7.2.1. Fiziksel Güvenlik

Bulut hizmet sağlayıcıları, veri merkezlerini fiziksel tehlike ve tehditlerden uzak tutmak ve korumak için sıkı bir güvenlik tedbiri almak zorundadır. Bu korumalar içinde erişim kontrolü, kapalı devre TV (Closed Circuit TV-CCTV), çok sayıda güvenlik personeli kullanmak gibi birtakım önlemler sıralanabilir.

Kapalı devre TV veya diğer adıyla CCTV, gözetim kameraları ve izleme ekipmanları elemanlarından oluşan bir televizyon sistemi türüdür (Görsel 7.6). Bu sistem; belirli bir alandaki olayları ve aktiviteleri izlemek, gözetlemek veya kaydetmek amacıyla kullanılır. Kapalı devre TV genellikle güvenlik amacıyla kullanılır ve birçok farklı alanda bulunabilir. Kapalı devre TV sistemi ile kameralar tarafından yakalanan görüntüler, kontrol merkezine veya kayıt cihazlarına iletilir. Burada yer alan görüntüler, gerçek zamanlı olarak gözetim görevlileri tarafından izlenebilir, ilerideki incelemeler veya soruşturmalar için kaydedilebilir. Bu tür sistemler sayesinde güvenlik ve gözetim ihtiyaçlarını karşılamada etkili bir çözüm sunulur ancak özel alanlara müdahale, gizlilik ihlalleri gibi konularda bu sistemler dikkatli bir şekilde kullanılmalıdır.



Görsel 7.6: Kapalı devre TV

7.2.2. Veri Erişim Kontrolü

Veri erişim kontrolü, bulut tabanlı veri hizmetlerinin güvenliğini sağlamak için önemlidir. Veri erişim kontrolü; kimin, ne zaman, hangi verilere erişebileceğini ve bu erişimin nasıl denetleneceğini belirleyen bir dizi önlem ve politikayı içerir. Ayrıca bu kontroller; veri gizliliğini, bütünlüğünü ve güvenliğini sağlamak için tasarlanır.

Verilere yetkisiz bir biçimde erişim hem bulut sistem veri tabanında hem de yerel veri tabanında istenmeyen bir durum oluşturur. Yetkisiz veri erişimlerini önlemek için güçlü bir kimlik doğrulama (authentication) (Görsel 7.7) ve yetkilendirme (authorization) mekanizmaları kullanılmalıdır.

Görsel 7.7: Kimlik doğrulama (Authentication)

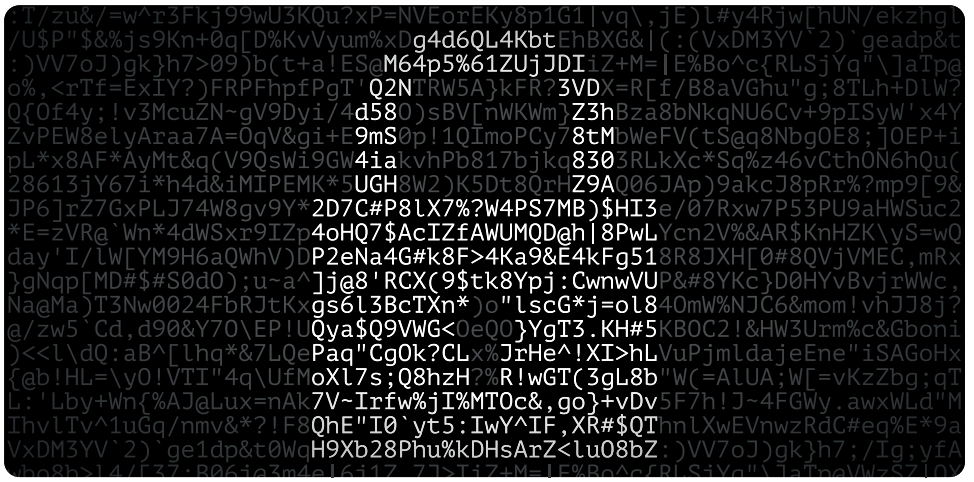
Kimlik doğrulama, kullanıcıların kimliklerinin doğrulanması işlemidir. Bu işlem, kullanıcıların doğru kimlik bilgileri ile sisteme giriş yapmasını sağlar. Yetkilendirme (Görsel 7.8) ise kimlik doğrulamasının ardından kullanıcılara hangi kaynaklara erişebileceklerinin ve bu kaynaklarda ne tür işlemleri gerçekleştirebileceklerinin belirlenmesi olarak tanımlanır.

Görsel 7.8: Yetkilendirme (Authorization)

Bulut bilişim sistemlerinde sağlıklı biçimde veri erişim kontrolü sağlamak için veri izolasyonu, güvenli veri paylaşımı seçenekleri de düşünülmelidir.

7.2.3. Veri Şifreleme

Kriptografi yöntemleri kullanılarak hem veri iletimi sırasında hem de depolama aşamasında verilerin şifrelenmesi (Data Encryption) kritik öneme sahiptir. **Bulut tabanlı veri şifreleme**, bulut tabanlı veri depolama hizmetlerinde (Cloud-Based Data Storage Services) kullanılan bir güvenlik önlemidir. Bu hizmetlerde, kullanıcıların verileri bulut sunucularında saklanır ve yönetilir. Bu veriler, hassas bilgiler içerebilir. Bulut tabanlı veri şifreleme, verilerin bulut sunucularında şifrelenmesini içerir. Bu sayede verilere yetkisiz erişim engellenir, ele geçirilen ifadeler olursa veriler anlamsız hâlde bulunacağı için veri gizliliği sağlanır (Görsel 7.9). İki temel türde bulut tabanlı veri şifrelemesi vardır.



Görsel 7.9: Verilerin şifrelenmesi (Data encryption)

1. Uygulama

İşlem adımlarına göre masaüstünde bulunan **Dosya.txt** dosyasının içeriğine isminizi ve soy isminizi yazarak C# programlama dilinde hazırlayacağınız bir kodla dosya içeriğini AES yöntemiyle şifreleyiniz.

1. Adım: Masaüstünüzde **Dosya.txt** isimli bir metin belgesi oluşturunuz.

2. Adım: Dosya.txt dosyasını açıp dosyanın içeriğine isminizi ve soy isminizi yazınız, dosyayı kaydedip kapatınız.

3. Adım: Kod editörü programını açarak C# programlama dilini seçiniz ve Console uygulamasını açınız.

4. Adım: Kod editörüne **System.Security.Cryptography** kütüphanesini dâhil ederek dosyanın şifrelenmesi için şu metodu oluşturunuz:

```
static void EncryptFile(string girisDosyasi, string cikisDosyasi, string sifre)
{
    using (FileStream fsGiris = new FileStream(girisDosyasi, FileMode.Open),
                      fsCikis = new FileStream(cikisDosyasi, FileMode.Create))
    {
        using (Aes aesAlg = Aes.Create())
        {
            Rfc2898DeriveBytes sifreleme = new Rfc2898DeriveBytes(sifre, new byte[] { 0x49, 0x76, 0x61, 0x6e,
0x20, 0x4d, 0x65, 0x64, 0x76, 0x65, 0x64, 0x65, 0x76 });
            aesAlg.Key = sifreleme.GetBytes(32);
            aesAlg.IV = sifreleme.GetBytes(16);
            using (CryptoStream cryptoStream = new CryptoStream(fsCikis, aesAlg.CreateEncryptor(), CryptoStre-
amMode.Write))
            {
                fsGiris.CopyTo(cryptoStream);
            }
        }
    }
}
```

5. Adım: Oluşturduğunuz metodu kullanabilmek için masaüstündeki Dosya.txt isimli metin dosyasını kod editörünün klasörünün içinde bulunan bin\Debug\net8.0 klasörünün içine atınız.

Oluşturulan uygulama için örnek veri yolu şu şekildedir:

C:\Users\Admin\source\repos\ConsoleApp1\ConsoleApp1\bin\Debug\net8.0

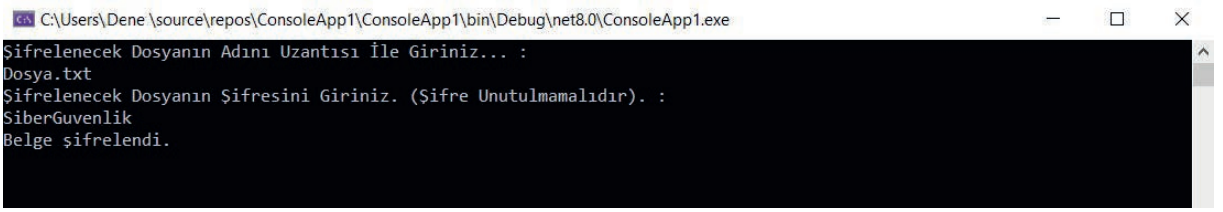
Not

Uygulamada kullanılan kod editörü programı 2022 versiyonudur. Daha düşük versiyonlarda klasör yolu farklılık gösterebilir. bin dosyasında bulunan Debug klasörü altındaki yer sabit kalır. Versiyona göre klasör ayarlanmalıdır.

6. Adım: Dosya.txt isimli metin dosyasının içeriğini şifrelemek için gerekli metotları kullanarak ana metodu oluşturunuz. Bu işlem için ilgili kodlar şunlardır:

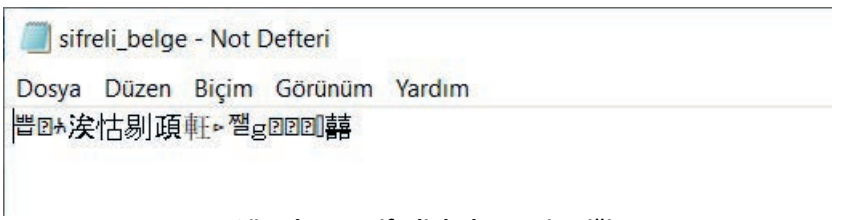
```
static void Main()
{
    Console.WriteLine("Şifrelenecek Dosyanın Adını Uzantısı İle Giriniz... : ");
    string orjinalDosya = Console.ReadLine();
    Console.WriteLine("Şifrelenecek Dosyanın Şifresini Giriniz. (Şifre Unutulmamalıdır). : ");

    string sifreliOlusacakDosya = "sifreli_belge.txt";
    string sifreleme = Console.ReadLine(); // Güçlü bir şifre kullanmanız önemlidir.
    EncryptFile(orjinalDosya, sifreliOlusacakDosya, sifreleme);
    Console.WriteLine("Belge şifrelendi.");
    Console.ReadLine();
}
```

7. Adım: Programı çalıştırınız ve gelen ekranda dosya adına Görsel 7.10'da görüldüğü gibi **Dosya.txt**, şifre sorusuna da **SiberGuvenlik** giriniz.



Görsel 7.10: Şifreleme uygulamasının çalıştırılması

Program çalıştırıldığında metin dosyası, **SiberGuvenlik** şifresi ile şifrelenerek **sifreli_belge.txt** şeklinde klasöre kaydedilir. **sifreli_belge.txt** açıldığında Görsel 7.11'deki gibi bir görüntü oluşur.



Görsel 7.11: sifreli_belge.txt içeriği

Sıra Sizde

Gorsel1 isimli **jpg** uzantılı bir dosyayı birinci uygulamadaki kriptografi yöntemi ile şifreyelerek **jpg** uzantılı **SifreliGorsel** isminde bir dosya oluşturunuz. **GorselSifreleme** ifadesini şifre olarak kullanınız.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. Görsel1 dosyasını oluşturarak ilgili klasörün içine aldı.		
2. Kod editörü programında gerekli metodu oluşturdu.		
3. Kod editörü programında gerekli ana programı oluşturdu.		
4. Hazırladığı programı çalıştırarak SifreliGorsel isimli jpg dosyasını oluşturdu.		
"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.		

7.2.3.1. Veri Transferinde Şifreleme (Data In Transit Encryption)

Veriler, kullanıcının cihazından bulut servis sağlayıcısının sunucusuna veya bulut hizmet sağlayıcının sunucusundan kullanıcı cihazına transfer edilirken şifrelenir. Bu yöntem kullanılarak şifrelenen veriler, iletim hattı üzerinde şifrelendiği için trafik esnasında meydana gelebilecek olası bir sızıntı da güven altındadır. Bu nedenle sistem de oldukça güvenilirdir. Her güvenlik önleminde olduğu gibi bu veri güvenlik önleminde de avantajlar kadar dezavantajlar da bulunur. Veri transferinde şifrelemenin dezavantajları şunlardır:

- **Performans:** Veri şifreleme, oldukça yoğun bir performans ve işlem gücü gerektirir. Bu işlemin hâlihazırda bir transfer sürecinde yapılıyor olması, gereken performansı kat kat artırır. Dolayısıyla bu tekniği kullanmak, zaman zaman veri iletiminin yavaş olmasına ve bazen de sistemin daha fazla kaynak tüketmesine yol açar. Özellikle yüksek hacimli verilerde (video vb.) veri iletimi sırasında performans gerekliliği ciddi biçimde gözlenir. Düşük donanım özelliğine sahip cihazlarda veri iletilirken ara ara donmalar olabilir.
- **Maliyet:** Şifreleme çözümleri genellikle özel yazılım veya donanım gerektirir. Bu durum, ek maliyetler anlamına gelir. Bu tür güvenlik önlemleri kullanılırken verilerde kayıplar yaşanmaması için güvenlik ve bakım amaçlı ekstra personel ihtiyacıyla karşılaşılır.
- **Karmaşıklık:** Şifreleme teknolojisi ve tekniği (kriptografi) karmaşık olabilir ve bu teknolojiyi doğru biçimde yönetip yapılandırmak, sürdürmek ve güncellemek de ek bir uzmanlık gerektirir. Yanlış yapılandırılan şifreleme teknolojisi, güvenlik açıklarına ve veri kayıplarına yol açabilir.
- **Uyum Sorunları:** Bazı endüstri alanları (Telekomünikasyon, Havacılık ve Uzay, Hükümet ve Kamu Kuruluşları, Yüksek Teknoloji Şirketleri vb.) ve yargı kuruluşları ile çalışılırken belirli güvenlik standartlarına uyulmalıdır. Bu standartlar, veri iletimi şifrelemesi gibi güvenlik önlemlerinin belirli bir şekilde uygulanmasını şart koşarak ek algoritmaların kullanılmasını gerektirebilir. Bu durum, olası uyumluluk sorunlarına yol açar.
- **Kullanıcı Deneyimi:** Güvenlik önlemleri, veri iletimi sırasında bazen kullanıcı deneyimini olumsuz yönde etkiler. Belirli uygulamaların veya cihazların daha yavaş çalışması veya ek doğrulama adımları gerektirmesi, kullanıcıların rahatsızlık yaşamasına neden olabilir.

Not

Dezavantajlar, veri iletimi şifrelemesinin faydalarını gölgelememelidir. Veri güvenliği, çoğu durumda bu dezavantajlara değecek kritik bir özelliktir. Bu nedenle uygun şekilde planlandığında ve yönetildiğinde bu dezavantajlar minimize edilebilir.

7.2.3.2. Veri Transferi Durduğunda Şifreleme (Data At Rest Encryption)

Veri transferi durduğunda şifreleme tekniğiyle veriler, bulut sunucuda saklandığı esnada şifrelenir. Bu sayede veriler, sunucularda depolandığı zamanlarda güvenli bir şekilde muhafaza edilir. Bu yöntemin de kendine göre dezavantajları bulunur. Veri transferi durduğunda şifrelemenin dezavantajları şunlardır:

- **Kullanıcı Dostu Olmaması:** Şifrelenmiş verilere erişim ve bu verilerin kullanımı, kullanıcılar için zorlu süreçler ortaya çıkarabilir. Kullanıcıların şifreleri yönetmesi ve şifrelenmiş verileri günlük süreçlerine katması, kullanıcı deneyimini zorlaştırır.
- **Yedekleme ve Kurtarma Noktasında Zorluklar:** Şifrelenmiş verileri yedeklemek, gerektiğinde kurtarmak karmaşık olabilir. Veri kaybını önlemek ve hızlı bir şekilde veriye erişim sağlamak için uygun yedekleme ve kurtarma yöntemleri gereklidir. Bu yöntemleri kullanmak da işleri zorlaştırır.
- **Anahtar Yönetimi:** Şifreleme için kullanılan anahtarların güvenli bir şekilde yönetilmesi veya muhafaza edilmesi gerekir. Anahtarların kaybolması veya yetkisiz kişilerin eline geçmesi, şifrelemenin etkinliğini zedeler ve verilerin esas şekliyle ortaya çıkmasına yol açar.
- **Veri İletimi Esnasında Sızıntı:** Veriler her ne kadar depolama esnasında şifrelenmiş olsa da veri iletim trafiği zamanında şifreleme açısından savunmasız durumdadır. Veri trafiği esnasında olabilecek herhangi bir sızıntı (ihlal), verilerin yetkisiz kişilerin eline geçmesine sebep olur.

7.2.4. Denetim ve İzleme

Denetim ve izleme, bulut sağlayıcılarının temel görevleri arasında yer alır. Bulut sağlayıcılar, sistemlerin ve verilerin etkin bir biçimde denetlenmesini ve izlenmesini sağlar. Olası bir güvenlik ihlalinde, ihlali tespit etmek ve ihlale müdahale etmek kritik öneme sahiptir. Bulut sistemlerde en çok karşılaşılan siber tehdit, APT (Advanced Persistent Threat-Gelişmiş Kalıcı Tehdit) olarak düşünülebilir (Görsel 7.12). Bilgisayar güvenliği ve siber saldırı alanında APT ile sıkça karşılaşılır. APT ile gelişmiş ve sürekli devam eden saldırılar yapılarak çeşitli sorunlar ortaya çıkartılır.

Görsel 7.12: Gelişmiş kalıcı tehdit

APT'lerin özellikleri şunlardır:

- **Gelişmiş Teknik Yetenekler Kullanmak:** APT'ler, hedef sistemlere sızmak ve sızdıkları yerde uzun süre kalabilmek için gelişmiş teknik bilgi ve yeteneklere sahiptir.
- **Hedefte Kalıcılık:** APT'ler genellikle uzun vadeli olarak faaliyet gösterir. Saldırıları başarılı bir şekilde gerçekleştirdikten sonra bile hedef sistemde uzun süre varlıklarını sürdürmeyi amaçlar.
- **Hedef Belirleme Yeteneği:** APT'ler genellikle belirli hedeflere yönelik saldırılar yapar. Bu hedefler; devlet kurumları, büyük şirketler, askerî kuruluşlar veya stratejik öneme sahip diğer kurumlar olabilir.
- **Gizlilik ve Görünmezlik Özelliği:** APT'ler genellikle gizliliklerini korumak ve tespit edilmemek için yoğun çaba harcar. Bu durum, APT ile yapılan saldırıların uzun süre fark edilmeden sürmesine olanak tanır.
- **Amaç ve Duruma Uyarlanabilme Özelliği:** APT'ler genellikle belirli bir hedefe yönelik saldırılarını gerçekleştirmek için uygun yöntemleri ve araçları seçebilme yeteneğine sahiptir. APT'ler taktiklerini ve araçlarını zaman içinde değiştirebilir.

APT'ler, siber saldırılarda en gelişmiş ve karmaşık grupları ifade eder. Hedeflerine gizlice sızmak, uzun süreli erişim kurmak ve değerli bilgileri çalmak en büyük kullanılma amaçlarındandır.

7.2.5. Yedekleme ve Kurtarma

Verilerin bulut depolama sistemlerinde güvenli bir şekilde saklanması kadar sağlıklı bir biçimde yedekleme planı oluşturulması da verilerin kurtarılması ve sistemin yeniden aktif edilebilmesi için kritik öneme sahiptir. Bulut sağlayıcının görevi, olası siber saldırı veya veri iletim kayıplarından kaynaklanan veri bozulmalarının ve veri kayıplarının önüne geçmektir (Görsel 7.13).

Görsel 7.13: Yedekleme ve kopyalama

Bulut bilişimde kullanılan yaygın yedekleme ve kopyalama hizmetleri şunlardır:

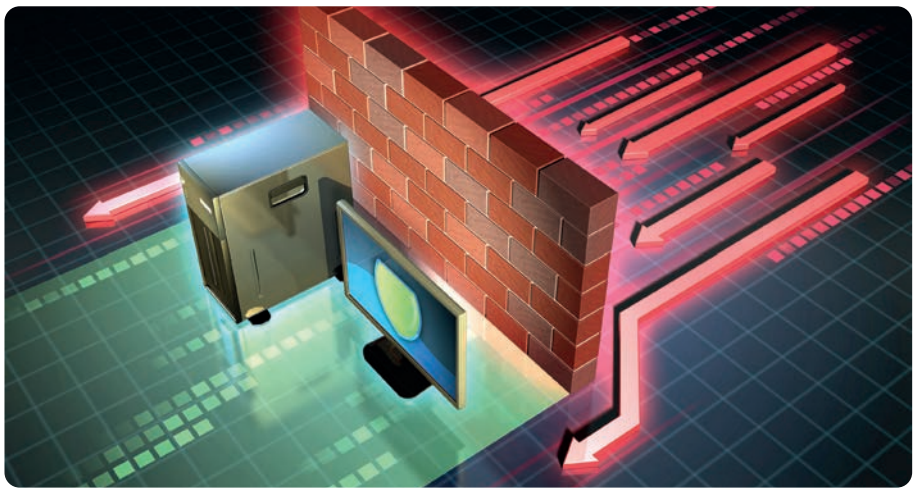
- **Bulut Depolama Hizmetleri:** Kullanıcılar için bu hizmetler, verilerin uzaktaki sunucularda depolanmasına olanak tanır. Veriler, bu sunuculara yedeklenir ve gerektiğinde geri alınabilir. Popüler bulut depolama sağlayıcıları arasında Amazon S3, Microsoft Azure Blob Storage, Google Cloud Storage, Dropbox, Box.com gibi hizmetler bulunur.

- **Bulut Yedekleme Hizmetleri:** Verileri otomatik olarak belirli bir plana göre buluta yedekler. Veriler düzenli aralıklarla (günlük, haftalık, aylık) yedeklenerek veri kaybı riski azaltılabilir. Örnek hizmetler arasında Acronis True Image, Backblaze, Carbonite gibi yedekleme çözümleri yer alır.
- **Veri Senkronizasyon Hizmetleri:** Belirli bir klasörde veya cihazda yapılan değişiklikleri otomatik olarak buluta ve diğer bağlı cihazlara senkronize eder. Bu durum, verilerin her yerde güncel olmasını sağlar. Dropbox, Google Drive, Microsoft OneDrive gibi hizmetler bu tür senkronizasyonu sağlar.
- **Bare Metal Yedekleme Hizmetleri:** Fiziksel sunucuların veya makinelerin tüm işletim sistemleri, uygulamalar ve verileriyle birlikte yedeklendiği bir süreçtir. Bu tür yedeklemeleri, bulut tabanlı hizmetler aracılığıyla yapmak mümkündür.
- **Veri Tabanı Yedekleme Hizmetleri:** Özellikle büyük miktarda veri barındıran sistemlerde veri tabanı yedeklemesi önemlidir. Bulut tabanlı veri tabanı hizmetleri (Amazon RDS (Relational Database Service), Azure SQL Database vb.) genellikle otomatik yedekleme seçenekleri sunar.
- **Arşivleme Hizmetleri:** Bazı verileri uzun süreli saklamak veya arşivlemek gerekebilir. Bulut arşivleme hizmetleri, verileri düşük maliyetle uzun süre saklama imkânı sunar.

Veri transferi esnasında veya veri transferi durduğunda yöntemlerinden hangisinin seçileceği; ihtiyaçlara, veri miktarına, bütçeye ve veri güvenliğine bağlı olarak değişir. Ayrıca verilerin hassaslığı, hizmet sağlayıcının güvenlik önlemleri, hizmetin hızı gibi faktörler göz önünde bulundurulmalıdır.

7.2.6. Güvenlik Duvarları

Bulut mimarisi yapısını dış tehditlerden korumak için kullanılacak ilk öncelik güvenlik duvarlarıdır (Görsel 7.14). Veri merkezi ve sunucu arasındaki trafiği denetleyen güvenlik duvarları, zararlı trafiği engellemeye çalışır. Güvenlik duvarları, farklı hedefleri ve kullanım senaryolarını karşılamak üzere çeşitli tiplere sahip olabilir. Bu tipler, ihtiyaca göre ayarlanabilir ve farklı güvenlik gereksinimlerini karşılamak üzere tasarlanır.



Görsel 7.14: Güvenlik duvarı (Cloud firewall)

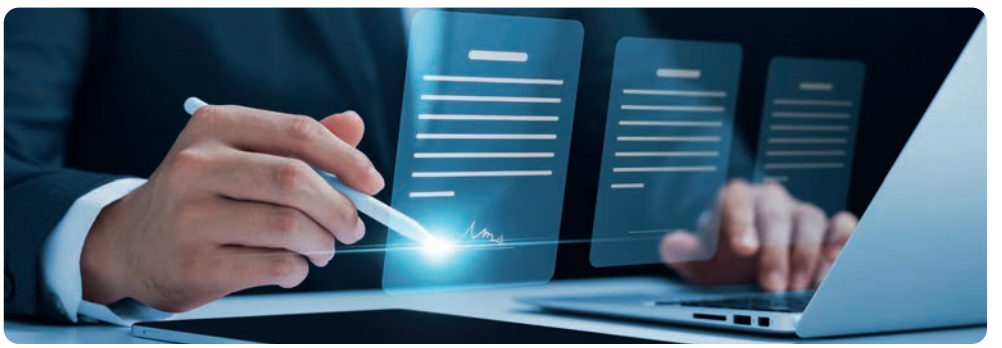
Güvenlik duvarlarının amaçlarına göre çok fazla seçeneği olmakla birlikte bulut hizmet tabanlı kullanılan güvenlik duvarları şu şekilde sıralanabilir:

- **Ağ Tabanlı Bulut Güvenlik Duvarları:** Ağ trafiğini analiz eder, gelen veya giden veri trafiğini filtreler. Kötü niyetli içerikleri, saldırıları ve istenmeyen trafiği engellemek için kullanılır.
- **Uygulama Tabanlı Bulut Güvenlik Duvarları:** Uygulama katmanı düzeyinde çalışır ve özellikle web uygulamalarının güvenliğini sağlamak için tasarlanmıştır. Web uygulama saldırılarına karşı koruma sağlar.
- **Veri Tabanlı Bulut Güvenlik Duvarları:** Veri tabanlarındaki verilere erişimi ve manipülasyonu kontrol etmek, hassas verilere yetkisiz erişimi engellemek, veri tabanı güvenliğini artırmak amacıyla kullanılır.
- **Sanal Makine (VM) Tabanlı Bulut Güvenlik Duvarları:** Bulut ortamlarında sanal makineleri korumak için kullanılır. Sanal makineler arasında trafiği izleyebilir ve izole edebilir.
- **API Tabanlı Bulut Güvenlik Duvarları:** API'lar (Application Programing Interface-Uygulama Programlama Arayüzleri) aracılığıyla sağlanan hizmetlerin güvenliği için kullanılır. API trafiğini denetler ve kimlik doğrulama kontrolleri yapar.
- **Görüntü Tabanlı Bulut Güvenlik Duvarları:** Sanal makine görüntülerinin (image) güvenliğini sağlamak için kullanılır. Görüntülerin doğrulanması, güvenlik açıklarının tespiti gibi işlemleri yapar.
- **Bütünleşik Bulut Güvenlik Duvarları:** Farklı güvenlik katmanlarını entegre ederek çok yönlü bir koruma sağlar. Ağ güvenliği, uygulama güvenliği, veri güvenliği gibi farklı alanları kapsar.
- **Bulut Hizmet Sağlayıcısı Tarafından Sağlanan Güvenlik:** Birçok bulut hizmet sağlayıcısı, kendi bulut güvenliği hizmetlerini sunar. Bu hizmetler, bulut kaynaklarını kullanan müşterilere ek güvenlik sağlamak amacıyla sunulur.

Her güvenlik duvarı türünün kendine has avantaj ve dezavantajları, kullanım alanları bulunur. İlgili hizmet yetkilileri, ihtiyaç ve bulut hizmet kullanım türlerine göre uygun güvenlik duvarını seçerek kullanmalıdır.

7.2.7. Sözleşme ve Yetkilendirme

Bulut hizmetlerini kullanmak isteyen kullanıcılar öncelikle hizmet sağlayıcılar ile dijital veya kâğıt üzerinde bir sözleşme imzalarlar (Görsel 7.15). Bulut veri tabanı güvenliğinde sözleşme ve yetkilendirmenin içeriği, veri sahibi (müşteri) ile Bulut Hizmet Sağlayıcı (Cloud Service Provider-CSP) arasındaki ilişkinin düzenlendiği ve veri güvenliğini sağlamaya yönelik temel hususları belirten belgelerdir. Bu sözleşme ve yetkilendirmenin içeriği; veri gizliliği, veri bütünlüğü, erişim kontrolleri, siber güvenlik önlemleri, hizmet sağlayıcının sorumlulukları gibi konuları kapsar.



Görsel 7.15: Bulut bilişim sözleşme ve yetkilendirme

Bulut veri tabanı güvenliğinde önemli olan sözleşme ve yetkilendirmenin bazı içerik maddeleri şunlardır:

- **Veri Gizliliği ve Mahremiyeti:** Verinin gizliliğini ve mahremiyetini sağlamak için alınacak önlemleri tanımlar. Veri sahibinin, verilerinin üçüncü taraflarla paylaşılmasını sınırlayacak hükümler içerir.
- **Veri Bütünlüğü:** Verinin değiştirilmesi veya bozulmasını önlemeye yönelik önlemleri belirler. Verinin güvenli bir şekilde saklanması ve değiştirilemezliğinin sağlanması amaçlanır.
- **Erişim Kontrolleri:** Veriye erişim, yetkilendirilmiş kullanıcılar ve roller aracılığıyla sıkı bir şekilde kontrol edilmelidir. Bu madde, veriye erişim izinlerinin nasıl düzenleneceğini ve denetleneceğini tanımlar.
- **Siber Güvenlik Önlemleri:** Güvenlik açıklarını ve siber tehditleri önlemek için alınacak önlemleri içerir. Fiziksel güvenlik, veri şifrelemesi, güvenlik duvarları, saldırı tespit sistemleri gibi teknik detayları ele alabilir.
- **Süreklilik ve Kurtarma Planları:** Bulut hizmet sağlayıcısının, veri kaybı durumunda nasıl hareket edeceğini ve hizmet kesintilerinin nasıl ele alınacağını belirten bir madde içermelidir.
- **Sorumluluklar ve Taahhütler:** Bulut hizmet sağlayıcının ve kullanıcıların hangi sorumlulukları üstlendiği ve taahhütlerinin neler olduğu bu başlık altında açıkça belirlenir. Sorumluluklar ve taahhütler, ihlal durumlarında tarafların nasıl hareket edeceğini de kapsar.
- **Denetim ve İzleme:** Veri sahibi, hizmet sağlayıcının veri güvenliği önlemlerini düzenli olarak denetlemek ve izlemek isteyebilir. Bu madde, denetimlerin nasıl yapılacağını ve sonuçlarının nasıl paylaşılacağını tanımlar.
- **Sözleşmeyi Sonlandırma:** Sözleşmenin nasıl sonlandırılacağını, verilerin nasıl iade veya imha edileceğini düzenler.

Sözleşme ve yetkilendirme belgesi yasal bir zorunluluk doğurur. Sözleşme ve yetkilendirme; yerel (ulusal) yasalara ve düzenlemelere, sektör standartlarına ve veri sahibinin özel gereksinimlerine uygun olmalıdır. Yüksek öneme sahip verileri olan kullanıcıların (özel şirket, kamu kurumları vb.) bu belgeleri, profesyonel hukukçulara ve siber güvenlik uzmanlarına inceletmesi de faydalıdır.

7.3. BULUT BİLİŞİMDE VERİ GÜVENLİĞİ

Bulut bilişim, iş dünyasının ve bireylerin verilerini depolama, işleme ve paylaşma yöntemlerini kökten değiştiren önemli bir teknolojik dönüşüm sağlamıştır (Görsel 7.16). Bu durum, yeni dönemdeki fırsatlar kadar ciddi güvenlik zorluklarını da beraberinde getirir. Bulut bilişim, verilerin fiziksel sınırların ötesine taşınması ve farklı hizmet sağlayıcılar arasında paylaşılmasını içerdiği için geleneksel güvenlik yaklaşımları yetersiz kalır. Bu nedenle bulut bilişimde veri güvenliğini sağlamak için kapsamlı ve etkili önlemler alınmalıdır.

Görsel 7.16: Bulut bilişim senkronizasyon

Etkin veri güvenliği stratejileri; güçlü şifreleme yöntemleri, kimlik doğrulama mekanizmaları, erişim kontrolü, sürekli izleme gibi unsurları içermelidir. Yapılacak etkinlikler hem kurumsal hem de bireysel düzeyde verilerin gizliliğini, bütünlüğünü ve erişilebilirliğini korumak adına çok önemlidir.

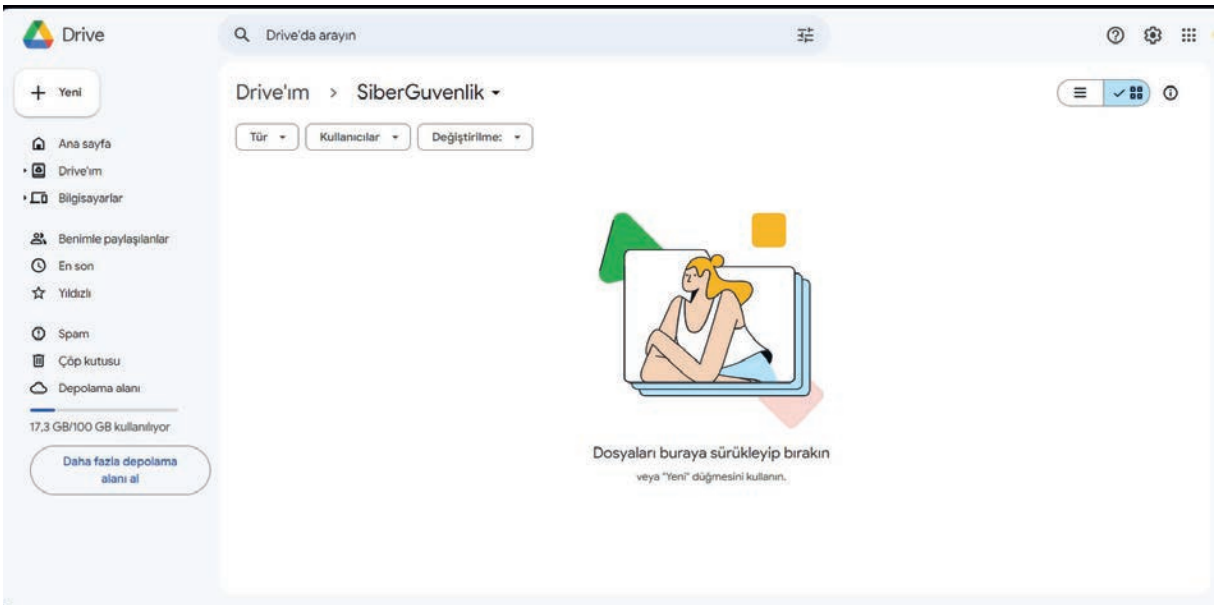
2. Uygulama

İşlem adımlarına göre bulut hizmeti kullanarak birinci uygulamada oluşturduğunuz **sifreli_belge.txt** dosyasını bulut server'a yükleyip bir arkadaşınızla görseli güvenli şekilde paylaşınız.

1. Adım: Bulut hizmeti sayfasını açarak bu sayfada üyeliğiniz yoksa yeni üyelik oluşturunuz.

2. Adım: Bulut hizmeti sayfasında yeni klasör oluşturunuz.

3. Adım: Oluşturduğunuz yeni klasörü Görsel 7.17'de görüldüğü gibi SiberGuvenlik olarak adlandırarak klasörün içini açınız.

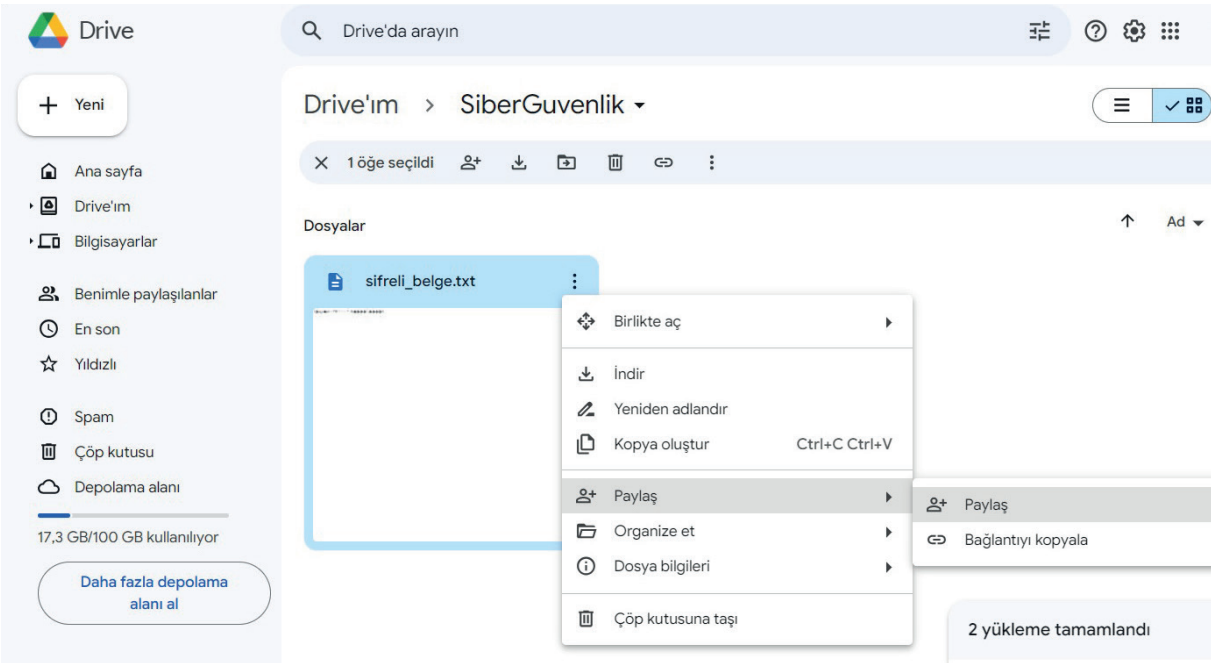


Görsel 7.17: Bulut hizmeti ekranı

4. Adım: Birinci uygulamada oluşturduğunuz **sifreli_belge.txt** dosyasını **SiberGuvenlik** klasörünün içine sürükleyip bırakınız.

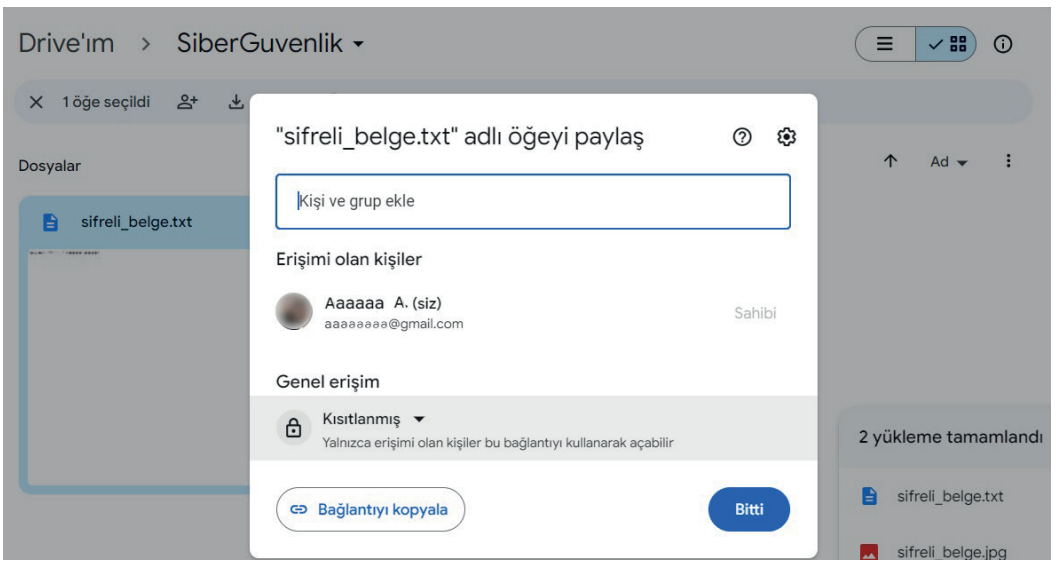
Not Şifreli belge, bulut hesabına güvenli bir biçimde kaydedilir. Bulut hesabının ele geçirilmesi ile dosya açığa çıksa bile şifreli olduğu için şifre bilinmeden herhangi bir biçimde dosya içeriği çözülemez.

5. Adım: Dosyanın güvenli bir biçimde istenen kişilerle paylaşılabilmesi için belgenin sağ üst köşesinde bulunan üç nokta işaretine tıklayıp Görsel 7.18'de görüldüğü gibi **Paylaş** seçeneğini seçiniz.



Görsel 7.18: Dosyanın güvenli paylaşılması

Açılan pencerede Görsel 7.19'da görüldüğü gibi **Kişi ve grup ekle** seçeneğiyle karşılaşılır. Buraya eklenecek kişiler veya mail adresleri **Erişimi olan kişiler** sekmesi altında görülür. Şifreli dosya, erişimi olan kişiler tarafından açılıp indirilebilir ve değiştirilebilir hâle gelir. **Genel erişim** sekmesi altında ise **Kısıtlanmış** ve **Bağlantıya sahip olan herkes** seçenekleri bulunur. Bu sekmede **Kısıtlanmış** seçeneği seçildiğinde şifreli dosya sadece erişimi olan kişilere açıktır. Bu sayede dosyanın paylaşımda olduğu kişiler kontrol altında tutulabilir. **Bağlantıya sahip olan herkes** sekmesi seçilirse dosya tüm kişilere açık hâle gelir ve güvenlik zafiyeti karşılaşılır. Bu uygulama, her ne kadar dosya güvenli bir biçimde şifreli olsa da siber güvenlikte tavsiye edilmez.



Görsel 7.19: Dosyanın güvenli paylaşım yöntemleri

Sıra Sizde

Oluşturduğunuz **SifreliGorsel.jpg** dosyasını bir arkadaşınızla bulut hizmeti üzerinden güvenli bir biçimde paylaşınız.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız.

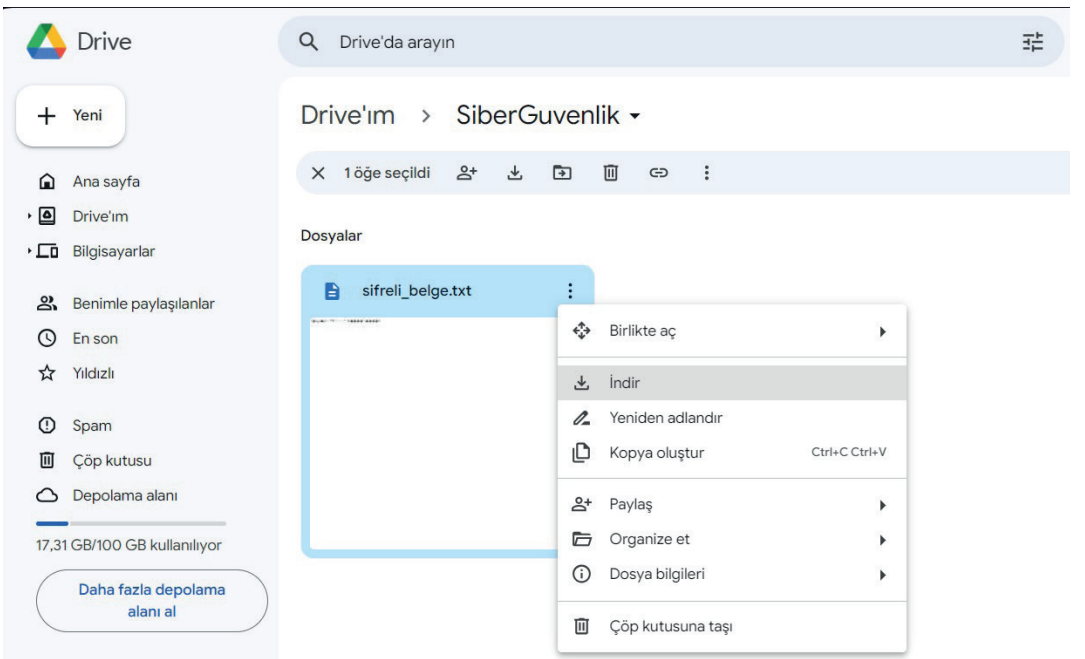
Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. Bulut hizmeti için üyelik oluşturup üyeliğe giriş yaptı.		
2. SifreliGorsel dosyasını bulut hizmete yükledi.		
3. SifreliGorsel dosyasını bir arkadaşı ile paylaştı.		
"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.		

3. Uygulama

İşlem adımlarına göre paylaşılan **sifreli_belge.txt** dosyasını Drive üzerinden indirip, şifresini çözerek dosyayı güvenli bir biçimde açınız.

1. Adım: Bulut hizmeti uygulamasını açınız. Görsel 7.20'de görüldüğü gibi kayıtlı olan **sifreli_belge.txt** dosyasının sağ tarafındaki üç nokta işaretine tıklayıp **İndir** seçeneğini seçiniz.



Görsel 7.20: Şifreli belgenin indirilmesi

2. Adım: İndirdiğiniz şifreli_belge.txt dosyasının şifresini çözebilmek için C# kod editörü programını açınız.

3. Adım: C# dilinde yeni bir console projesi oluşturunuz.

4. Adım: Kod editörüne System.Security.Cryptography kütüphanesini dâhil ederek dosyanın şifresinin çözümü için şu metodu oluşturunuz:

```
static void DecryptFile(string girisDosyasi, string cikisDosyasi, string sifre)
{
    using (FileStream fsGiris = new FileStream(girisDosyasi, FileMode.Open),
                      fsCikis = new FileStream(cikisDosyasi, FileMode.Create))
    {
        using (Aes aesAlg = Aes.Create())
        {
            Rfc2898DeriveBytes sifreleme = new Rfc2898DeriveBytes(sifre, new byte[] { 0x49, 0x76, 0x61, 0x6e,
0x20, 0x4d, 0x65, 0x64, 0x76, 0x65, 0x64, 0x65, 0x76 });
            aesAlg.Key = sifreleme.GetBytes(32);
            aesAlg.IV = sifreleme.GetBytes(16);
            using (CryptoStream cryptoStream = new CryptoStream(fsCikis, aesAlg.CreateDecryptor(), CryptoStre-
amMode.Write))
            {
                fsGiris.CopyTo(cryptoStream);
            }
        }
    }
}
```

5. Adım: Hazırladığınız metodu kullanabilmek için indirdiğiniz sifreli_belge.txt isimli metin dosyasını kod editöründe oluşturduğunuz proje klasörünün içindeki bin\Debug\net8.0 klasörüne atınız.

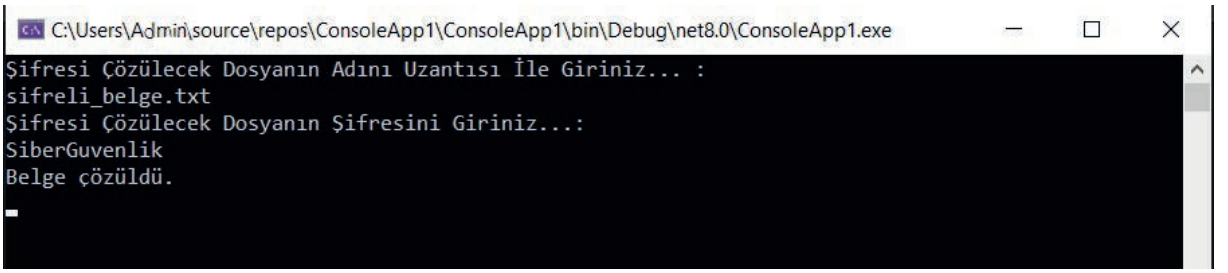
Oluşturulan uygulama örnek için veri yolu şu şekildedir:

C:\Users\Admin\source\repos\ConsoleApp2\ConsoleApp2\bin\Debug\net8.0

6. Adım: Hazırladığınız metodu kullanmak için gereken ana programı yazınız. Ana program şu şekildedir:

```
Console.WriteLine("Şifresi Çözülecek Dosyanın Adını Uzantısı İle Giriniz... : ");
 string sifreCozulecekDosya = Console.ReadLine();
 Console.WriteLine("Şifresi Çözülecek Dosyanın Şifresini Giriniz...: ");

 string sifreleme = Console.ReadLine(); // Güçlü bir şifre kullanmanız önemlidir.

    string cozulmusSekildeDosya = "cozulmus_belge.txt";
    DecryptFile(sifreCozulecekDosya, cozulmusSekildeDosya, sifreleme);
    Console.WriteLine("Belge çözüldü.");

 Console.ReadLine();
```

7. Adım: Yazdığınız programı çalıştırarak önce dosya adını sonra da dosyanın şifrelenmesi aşamasında kullanılan şifreyi Görsel 7.21'de görüldüğü gibi giriniz. Bilgilerin doğru olması durumunda klasörün içine **cozulmus_belge.txt** oluşturulur.



Görsel 7.21: Metin dosyasının şifresinin çözülmesi

Sıra Sizde

Oluşturduğunuz **SifreliGorsel** isimli **jpg** şifreli dosyasını, şifrelerken kullanılan **GorselSifreleme** şifresi ile çözünüz.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. Bulut hizmeti açarak SifreliGorsel dosyasını indirdi.		
2. SifreliGorsel dosyasını ilgili klasöre taşıdı.		
3. Kod editörü programında gerekli metodu hazırladı.		
4. Kod editörü programında gerekli ana programı hazırladı.		
5. Hazırladığı programı çalıştırarak SifreliGorsel dosyasını çözdü.		

"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.

4. Uygulama

İşlem adımlarına göre bulut bilişimde veri şifrelemenin önemini anlatan bir grup etkinliği gerçekleştiriniz ve sunum hazırlayınız.

1. Adım: Teorik temeli pekiştirmek amacıyla bulut bilişimde veri güvenliğini ve şifrelemenin rolünü anlatan bir mini sunum yapınız (Kişi sayısını iki olarak belirleyiniz.). Bu sunumda veri güvenliğinin neden önemli olduğunu, veri sızıntılarının potansiyel sonuçlarını ve şifrelemenin nasıl çalıştığını anlatınız.

2. Adım: Gruplar oluşturunuz ve veri sahibi bir kuruluşun bulut depolama hizmetini kullanarak veri güvenliği sorununu ele aldığı bir senaryo hazırlayınız.

Not

Her grup, farklı bir senaryo hazırlar. Örneğin bir sağlık kuruluşuna ait hasta verilerinin bulut depolama hizmetine yüklendiği ve bu verilerin korunması gerektiği vb. senaryolar çoğaltılabilir.

3. Adım: Verileri bulut depolama hizmetinde saklamadan önce hangi tür şifreleme stratejilerinin kullanılabileceğini grupça tartışınız.

4. Adım: Grupça belirlenen senaryoya uygun olarak hangi tür şifreleme yönteminin kullanılacağını ve nasıl bir şifreleme anahtarı yönetimi sağlanacağını seçiniz.

5. Adım: Grupça seçilen şifreleme stratejilerini ve planlarını içeren kısa bir sunum hazırlayınız. Hazırladığınız sunumu sırayla gerçekleştiriniz ve ardından diğer gruplardan geri bildirim alınız.

6. Adım: Gruplar sunumlarını tamamladıktan sonra sınıfta sunumlarla ilgili farklı senaryolara yönelik farklı şifreleme yaklaşımlarının nasıl değerlendirildiğini, bulunan stratejilerin avantajlarını ve dezavantajlarını tartışınız.

Araştırma

Bulut bilişim şifreleme uygulamalarındaki potansiyel güvenlik açıklarını araştırınız ve bunları gidermek için öneriler geliştiriniz.

7.4. KİMLİK YÖNETİMİ VE DENETİMİ

IAM (Identity and Access Management-Kimlik ve Erişim Yönetimi), bilgi teknolojileri ve bilgisayar güvenliği alanında kullanılır. IAM, bir organizasyonun sistemlerine kimlerin erişebileceğini yönetmek ve bu erişimleri denetlemek için kullanılan bir yönetim sürecini açıkça ifade eder (Görsel 7.22). IAM, güvenlik açısından oldukça önemlidir. Bulut bilişim ile yapılan veri trafiğinde verilere yetkisiz erişimin önlenmesi ve veri güvenliğinin sağlanması gerekir.

Görsel 7.22: Kimlik ve Erişim Yönetimi (Identity and Access Management)

IAM, kullanıcı kimlik doğrulamasını ve yetkilendirme süreçlerini içinde barındırır. Kullanıcıların kimlikleri doğrulanır ve hangi kaynaklara (dosyalar, veri tabanları, uygulamalar, ağlar vb.) erişebileceklerine dair kontrolleri belirlenerek güvenlik sağlanır.

IAM, kurumsal ağlarda ve bulut tabanlı hizmetlerde yaygın olarak kullanılır. Organizasyonların güvenlik politikalarını uygulamasına yardımcı olur. Bu kullanım; veri ihlallerini, yetkisiz erişimi ve diğer güvenlik risklerini azaltır.

5. Uygulama

İşlem adımlarına göre Kimlik ve Erişim Yönetimi (IAM) temel kavramlarını kullanarak belirlediğiniz senaryoda tanımlama ve yetkilendirme yapınız.

1. Adım: Kimlik (Identity) kavramını tanımlayınız. Kullanıcı kimlikleri, roller, gruplar, hizmet kimlikleri vb. hakkında temel bilgiler veriniz.

2. Adım: Erişim Yönetimi (Access Management) kavramını açıklayınız. İzinler, politikalar, roller ve erişim denetimleri hakkında temel bilgiler veriniz.

3. Adım: Güvenlik artışı, veri gizliliği, kaynakların etkin kullanımı yönünden bulut hizmetlerinde IAM kullanmanın avantajlarını sınıfta tartışınız.

4. Adım: Gruplar oluşturunuz ve her grup için bir kullanıcının belirli bir kaynağa erişiminin sınırlandığı farklı bir senaryo hazırlayınız.

5. Adım: Grupça belirlenen senaryoları çözümleyiniz ve çözüm tekniklerini sınıf ortamında sununuz.

6. Adım: Gruplar sunumlarını tamamladıktan sonra sınıfta sunumlarla ilgili farklı senaryolara yönelik farklı çözümleme yaklaşımlarını, bulunan çözüm yöntemlerinin avantajlarını ve dezavantajlarını tartışınız.

Okuma Parçası

Microsoft Exchange Server veri sızıntısı (2021): Şirket içi Microsoft Exchange Server'daki dört sıfırıncı gün açıklığı (henüz varlığı bilinmeyen veya bilindiği halde giderilmemiş açıklık) üzerinden Ocak 2021'de başlatılan küresel bir siber saldırı dalgası sonucunda saldırganlar, etkilenen sunuculardaki kullanıcı e-postalarına ve parolalarına tam erişim, sunucuda yönetici ayrıcalıkları ve ağa bağlı cihazlara erişim izni elde ettiler. Saldırganlar, sunuculara tam erişime izin veren bir arka kapı kurarak saldırıları gerçekleştirdiler. 9 Mart 2021 itibariyle, ABD'deki yaklaşık 30.000 kuruluş, Birleşik Krallık'taki 7.000 sunucu ve Avrupa Bankacılık Otoritesi, Norveç Parlamentosu ve Şili Mali Piyasa Komisyonu sunucuları da dâhil olmak üzere toplam 250.000 sunucunun saldırılara kurban gittiği tahmin edildi (Duffy, 2021; O'Donnell, 2021). Küçük ve orta ölçekli işletmeler, yerel kurumlar ve yerel yönetimler, siber güvenliği sağlama konusunda genellikle daha küçük bütçelere ve daha az deneyime sahip oldukları için saldırının ana kurbanları olarak öne çıktılar (Whittaker, 2021).

https://www.tubitak.gov.tr/sites/default/files/25506/siber_guvenlik_ortaokul.pdf

ÖLÇME VE DEĞERLENDİRME

A. Aşağıdaki parantezlerin içine cümlelerde verilen bilgiler doğru ise ”D”, yanlış ise “Y” yazınız.

1. ( ) Bulut bilişim güvenliğinin en büyük tehditlerinden biri, bulut hizmet sağlayıcısının altyapısına dışarıdan yetkisiz erişimdir.

2. ( ) Bulut bilişim kullanımıyla kurum ve kuruluşlar, kendi veri güvenliklerini tamamen hizmet sağlayıcısına devrederler.

3. ( ) Bulut bilişim hizmetleri kullanırken, kullanıcılar genellikle güvenlik önlemlerini kendileri yönetmek zorunda değildir.

4. ( ) Bulut bilişim güvenliğinde, veri şifreleme her zaman kullanılmalıdır.

5. ( ) Bulut bilişim hizmetleri, veri güvenliği endişeleri nedeniyle hiçbir zaman yerel veri depolama sistemlerinden daha güvenli değildir.

B. Aşağıdaki cümlelerde boş bırakılan yerlere doğru sözcükleri yazınız.

6. Bulut bilişim hizmetleri; ..... üzerinde depolama, işleme ve ağ kaynakları sağlayan esnek bir modeldir.

7. Veri şifrelemesi, verilerin ..... iletilmesini ve yetkisiz erişime karşı korunmasını sağlar.

8. Kimlik avı (phishing), kötü niyetli kişilerin sahte ..... kullanarak kullanıcıları kişisel bilgilerini vermeye ikna etmeye çalıştığı saldırılardır.

9. Veri yedeklemeleri, veri kaybını önlemek ve veri ..... durumunda verileri kurtarmak için önemlidir.

C. Aşağıdaki soruları dikkatlice okuyarak doğru cevabı işaretleyiniz.

10. Aşağıdakilerden hangisinde bulut bilişimde veri güvenliğinin önemi belirtilmiştir?

A) Sadece fiziksel sunucuları korumak.
B) Verilerin depolandığı fiziksel sunucuların yerini belirlemek.
C) İnternet bağlantısını hızlandırmak.
D) Verileri daha hızlı yedeklemek.
E) Verilerin güvenliğini sağlamak ve verilere yetkisiz erişimi önlemek.

11. Aşağıdakilerden hangisinde bulut bilişimde veri güvenliğinin önemi belirtilmiştir?

A) Verileri daha kolay paylaşmak.
B) Verileri daha hızlı yedeklemek.
C) Verilerin yetkisiz erişime karşı korunmasını sağlamak.
D) Verileri sıkıştırmak.
E) Verileri daha hızlı depolamak.

12. Aşağıdakilerden hangisi çok faktörlü kimlik doğrulama (MFAD) işleminde bulut hesaplarının güvenliğini artırma yöntemidir?

A) Sadece bir kimlik doğrulama faktörü kullanmak.
B) Yalnızca kullanıcı adı ve şifre kullanmak.
C) Verileri şifrelemeyi devre dışı bırakmak.
D) Sadece güçlü bir parola kullanmak.
E) Birden fazla kimlik doğrulama faktörünü kullanmak.

13. Aşağıdakilerden hangisi bulut veri güvenliğini tehdit eden bir faktör değildir?

A) Veri şifrelemesi
B) Fiziksel güvenlik zayıflıkları
C) Kötü niyetli iç personel
D) Güçlü parolalar
E) Veri hırsızlığı

14. Veri yedeklemelerinde aşağıdakilerden hangisi amaçlanır?

A) Sadece depolama alanını boşaltmak.
B) Veri kaybını önlemek ve kurtarmak.
C) Verilerin daha hızlı iletilmesini sağlamak.
D) Verileri daha hızlı sıkıştırmak.
E) Verileri daha hızlı şifrelemek.

15. Aşağıdakilerden hangisi kimlik avı (phishing) saldırılarına karşı alınacak önlemlerden biridir?

A) Güçlü parolalar kullanmak.
B) Sadece kamusal ağlarda işlem yapmak.
C) Sadece bilinen e-postalardaki bağlantılara tıklamak.
D) Herkesle verileri paylaşmak.
E) Sadece resmî web sitelerine erişim sağlamak.

16. Aşağıdakilerden hangisi veri güvenliği açısından daha güçlü bir işlemdir?

A) Tek faktörlü kimlik doğrulama
B) Çok faktörlü kimlik doğrulama
C) Kimlik avı (phishing) saldırısı
D) Açık metin şifreleme
E) Sayısal parola kullanmak

17. Veriler ile yapılabilecek işlemler için aşağıdakilerden hangisi veri güvenliği politikalarının ve prosedürlerinin amaçlarını ifade eder?

A) Hızlı bir şekilde paylaşmak.
B) Herkesle paylaşmak.
C) Güvenliğini sağlamak ve yönetmek.
D) Yedeklememek.
E) Herhangi bir cihazdan erişilebilir hâle getirmek.

18. Aşağıdakilerden hangisi veri sızıntısıyla ilgilidir?

A) Verileri daha hızlı sıkıştırmak.
B) Verileri yedeklemek.
C) Verilerin yetkisiz kişilerce ele geçirilmesi.
D) Verileri şifrelemek.
E) Verileri sıkıştırmamak.

19. Aşağıdakilerden hangisi güçlü bir parola örneği değildir?

A) 123456
B) P@ssw0rd
C) Tr0ub4dor&
D) Veri2023Güv3nl1ği
E) 8*Yenilik$987

20. Aşağıdakilerden hangisi bilgisayar sistemlerine sızarak, verileri kilitleyip fidye talep eden saldırı türüdür?

A) DoS
B) Kimlik avı (phishing)
C) Ransomware
D) Veri sızdırma
E) Ağ tarama

21. Aşağıdakilerden hangisi veri şifrelemesinin avantajlarından biridir?

A) Verileri daha hızlı yedeklemek.
B) Verileri daha hızlı iletmek.
C) Verileri sıkıştırmak.
D) Verileri yetkisiz erişime karşı korumak.
E) Verileri açık metin olarak saklamak.

22. Aşağıdakilerden hangisi fiziksel veri merkezi güvenliğini artırmak için kullanılabilir?

A) Güçlü parolalar
B) Veri şifrelemesi
C) Çevrimdışı yedekleme
D) Biyometrik kimlik doğrulama
E) Kamusal Wi-Fi kullanımı

23. Aşağıdakilerden hangisi güvenli bir bulut hizmeti seçilirken göz önünde bulundurulması gereken bir faktördür?

A) Zayıf parolalar kullanma
B) Ücretsiz hizmetlerin cazibesi
C) Sağlanan hizmet düzeyi anlaşması (SLA)
D) Veri şifrelemesini devre dışı bırakma seçeneği
E) Verileri herkese açık hâle getirme

24. Aşağıdakilerden hangisi herhangi bir kuruluş için veri güvenliğinin önemini açıklar?

A) Sadece yasal yükümlülükleri yerine getirmek.
B) İnsan hatalarını önlemek.
C) Müşteri güvenini kazanmak, yasal uyumluluğu sağlamak ve itibarı korumak.
D) Sadece maliyetleri düşürmek.
E) Verileri kamusal ağlarda paylaşmak.

D. Aşağıdaki soruların cevaplarını ilgili alana yazınız.

25. Bulut bilişim güvenliğinde veri sızıntısıyla ilgili en büyük zorluklar nelerdir? Bu zorlukları ele aşmak için hangi stratejiler izlenebilir?

26. Bir kuruluş, bulut bilişim hizmetlerini kullanmaya başlamadan önce hangi güvenlik önlemlerini almalıdır? Bu önlemler hangi riskleri azaltmaya yardımcı olabilir?

VERİ TABANI SİSTEMLERİ VE GÜVENLİĞİ

8. ÖĞRENME BİRİMİ

KONULAR

8.1. VERİ TABANI YÖNETİM SİSTEMLERİ

8.2. VERİ TABANI YÖNETİM SİSTEMLERİNİN KURULUMU

8.3. VERİ TABANI YÖNETİM SİSTEMLERİNİN KULLANIMI

8.4. SQL ve NoSQL UYGULAMALARI

8.5. VERİ TABANI GÜVENLİĞİ

NELER ÖĞRENECEKSİNİZ?

- Veri tabanı kavramını açıklama
- Veri tabanı yönetim sistemlerini tanımlama
- Veri tabanı yönetim sistemlerinin kurulumunu yapma
- Veri tabanı yönetim sistemi arayüzünü açıklama
- Tablo oluşturma
- Tablolar arası ilişkileri oluşturma
- SQL kavramını açıklama
- NoSQL kavramını açıklama
- SQL ve NoSQL arasındaki farkları açıklama
- Veri tabanı yönetim sistemini kullanarak SQL uygulaması yapma
- Veri tabanı yönetim sistemini kullanarak NoSQL uygulaması yapma
- Veri tabanı yönetim sistemi yapılandırmalarını yapma
- Veri tabanı güvenliği temel ilkelerini açıklama
- Güvenli veri tabanı için gerekli yapılandırmaları yapma

ANAHTAR KELİMELER

Alan, güvenlik, güvenlik önlemleri, kayıt, kullanılabilirlik, optimizasyon, ölçeklenebilirlik, veri, veri tabanı, veri tabanı tasarımı, veri tabanı yönetim sistemi (DBMS), tablo, yedekleme, yönetim

HAZIRLIK ÇALIŞMALARI

1. Kişiler, verilerini neden saklar ve onları korumak için hangi güvenlik tedbirlerini alır? Düşüncelerinizi arkadaşlarınızla paylaşınız.
2. Veri tabanı güvenliği ile ilgili güncel tehditler nelerdir? Düşüncelerinizi sınıfta arkadaşlarınızla tartışınız.

8.1. VERİ TABANI YÖNETİM SİSTEMLERİ

Veri tabanı yönetim sistemleri, verilerin hızlı ve kullanıcı ihtiyaçlarına uygun formatta saklanmasını ve alınmasını sağlar. Bu sistemler, güç kapatıldığında verileri kaybetmeden disk gibi kalıcı bir ortamda depolar.

Veri tabanları ve veri tabanı sistemleri hayatın her alanında yer alır. Veri tabanı sistemleri, bilgi toplumunun temelini oluşturur. Veri tabanları olmadan büyük miktarda veri içeren herhangi bir faaliyeti yürütmek zordur. Bankacılık işlemleri, hava yolları, üniversiteler, telekomünikasyon, finans işlemleri, online alışveriş işlemleri, günlük aktivitelerin bulunduğu veri tabanlarının kullanımına örnek olarak gösterilebilir.

İlişkisel veri tabanı, birbiriyle ilişkili verilerin bir koleksiyonudur. Örneğin bir üniversite veri tabanı; öğrenciler, öğretim üyeleri ve idari personel hakkındaki verileri düzenler. Bu veri tabanı; verilerin verimli bir şekilde alınmasına, eklenmesine ve silinmesine yardımcı olur. Bir başka veri tabanı örneği ise kütüphanedir. Kütüphane, farklı türlerden kitap koleksiyonu içerir. Bu örnekte kütüphane veri tabanıdır ve kitaplar da veridir. Modern veri tabanları, bir veri tabanı yönetim sistemi kullanılarak yönetilir.

Veri tabanı, bilgisayar ortamında saklanan düzenli bilgiler topluluğudur. Bilgisayarda sistematik erişim imkânı olan ve yönetilebilir, güncellenebilir, taşınabilir, birbirleri arasında tanımlı ilişkiler bulunan bilgiler kümesidir. Veri tabanının bileşenleri şunlardır:

Veri: Veri tabanı yönetim sisteminin en temel unsurudur. Kaydedilebilir bilinen gerçeklere **veri** denir. Bir diğer ifadeyle veri; belirli bir anlamı olmayan gerçekler, rakamlar ve istatistiklerdir. Veri kaynakları; fiziksel sistemler, analog veya dijital ölçümler, sayısal ifadeler, nüfus, maaş, deney sonucu vb. olabilir. Veri kaynağına örnek olarak 1, Ali, 25 gibi veriler verilebilir.

Bilgi: Veri işlendiğinde, organize edildiğinde, yapılandırıldığında veya belirli bir bağlamda sunulduğunda bilgi hâline gelir. Bilgi, kararların ve eylemlerin dayandığı işlenmiş verilerdir. Bilgi, anlamlı değerler sağlamak için düzenlenmiş ve sınıflandırılmış veriler olarak da tanımlanabilir. Bilgiye örnek olarak "Ali'nin yaşı 25'tir." cümlesi verilebilir.

Kayıt: Birbiriyle ilgili verilerin toplanmasıdır. Kayıt, tablodaki bir satırdır. Her satır, tablonun sütun olarak bölünen verilerini içerir. Örnekte verilen üç veri ögesinin (1, Ali, 25) hiçbir anlamı yoktur ancak bunlar şu şekilde düzenlenirse toplu olarak anlamlı bilgileri temsil eder:

Örnek

Bir veri tabanı için örnek tablo:

Sıra	İsim	Yaş
1	Ali	25
2	Veli	18

Tablo veya İlişki: Birbirleriyle ilişkili verilerin depolandığı ve ilgili kayıtların toplandığı yapılardır. Bu tablonun sütunlarına **alanlar** veya **nitelikler,** satırlarına ise **kayıt** denir.

Veri Tabanı: Mantıksal olarak ilişkili verilerin kalıcı bir koleksiyonudur. Bir veri tabanı, birden çok kullanıcı tarafından paylaşılan, biçimlendirilmiş bir şekilde depolanan, bir kuruluşun ilgili verilerinin organize edilmiş koleksiyonudur.

Örnek

Bir veri tabanı için tablo koleksiyonu şu şekilde oluşturulur:

1. Öğrenci Tablosu

Sıra	İsim	Yaş
1	Ali	25
2	Veli	18

2. Adres Tablosu

Sıra	Adres
1	Ankara
2	Bursa

3. Yıl Tablosu

Sıra	Yıl
1	I
2	II
3	I

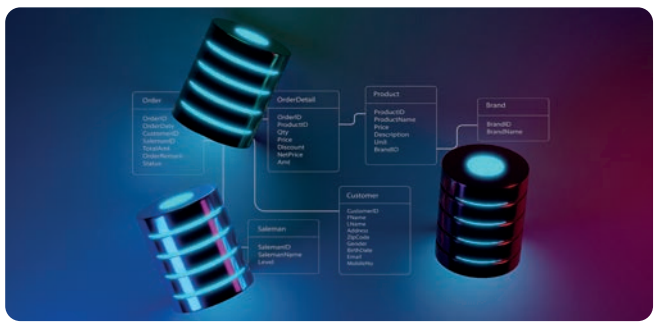
4. Yurt Tablosu

Yıl	Yurt
I	A Yurdu
II	B Yurdu

Dört tablodan oluşan bir veri tabanı elde edilir. Bunlara **ilgili koleksiyon** denilebilir çünkü seçilen bir tablo çiftinde bazı ortak özelliklerin mevcut olduğu açıkça görülür. Bu ortak özellikler nedeniyle bir öğrencinin tüm ayrıntılarını bulmak için iki veya daha fazla tablonun verileri bir araya getirilebilir. "B yurdunda kalan öğrenci hangi şehirde yaşıyor?", "En küçük öğrenci hangi yurtta kalıyor?" gibi yaş, şehir ve yurt özellikleriyle sorular, veriler farklı tablolarda olmasına rağmen cevaplandırılabilir.

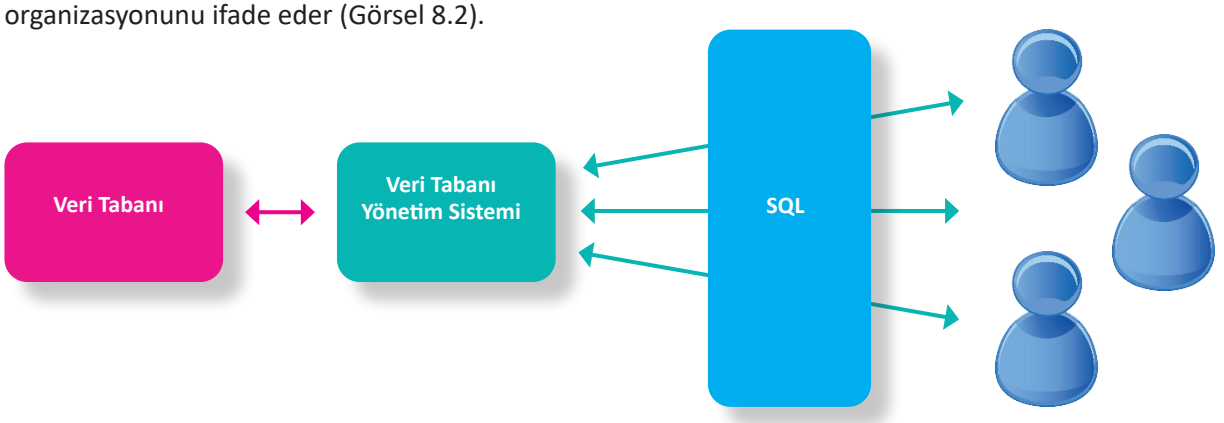
Anahtarlar: Veri tablosundaki kayıtları benzersiz bir şekilde tanımlayan ve ilişkilendiren ögelerdir. Tablolardaki verileri eşleştirmek için kullanılır.

- **Birincil Anahtar (Primary Key):** Benzersiz değerlerdir ve tablolardaki verilerin tutarlılığını sağlar. Genellikle otomatik artan bir sayısal değer (auto-increment) veya benzersiz bir alan olarak kullanılabilir.
- **İkincil Anahtar (Foreign Key):** Bir tablodaki alanın, başka bir tablodaki birincil anahtarla ilişkilendirilmesini sağlar. İki tablo arasında ilişkiler kurmak için kullanılır. Bir tablodaki bir alanın değeri, başka bir tablodaki birincil anahtarın değeriyle eşleşmelidir (Görsel 8.1).



Görsel 8.1: Veri tabanı bileşenleri

Veri Tabanı Yönetim Sistemi (Database Management System-DBMS); veri tabanlarının oluşturulmasını, yönetilmesini ve kullanılmasını sağlayan bir yazılımdır. DBMS; bir veri tabanı ortamında verilerin toplanmasını, depolanmasını, yönetimini, kullanımını tanımlayan ve düzenleyen bileşenlerin organizasyonunu ifade eder (Görsel 8.2).



Görsel 8.2: Veri tabanı ve veri tabanı yönetim sistemi ilişkisi

Veri tabanı yönetim sistemi; donanım, yazılım, kullanıcılar, prosedürler ve veriler olmak üzere beş ana bölümden oluşur:

- **Donanım:** Bilgisayarlar, depolama aygıtları, ağ aygıtları, yazıcılar ve diğer aygıtlar olmak üzere sistemin tüm fiziksel cihazlarını ifade eder.
- **Yazılım:** Veri tabanı yönetim sisteminin tam olarak çalışmasını sağlamak için işletim sistemi yazılımı, DBMS yazılımı, uygulama programları ve yardımcı programlar olmak üzere üç tür yazılıma ihtiyaç vardır.
- **Kullanıcılar:** Veri tabanı sisteminin tüm kullanıcılarını içeren bileşendir. Bu kullanıcılar; sistem yöneticileri, veri tabanı yöneticileri, veri tabanı tasarımcıları, sistem analistleri ve programcıları ile son kullanıcılardır (Görsel 8.3).

Görsel 8.3: Veri tabanıyla ilgili çalışan görevliler

- **Prosedürler:** Veri tabanı yönetim sisteminin tasarımını ve kullanımını yöneten talimatlar ve kurallardır. Prosedürler, sistemin kritik bir bileşenidir. Prosedürler bir şirkette önemli rol oynar. Bir organizasyonda müşterilerle işlerin yürütülmesine ilişkin standartları belirler.
- **Veriler:** Veri tabanının ana bileşeni veridir. Veriler; sayıları, karakterleri, sembolleri, görüntüleri içeren bir dizi bilgidir.

Bir DBMS şu ögeleri yönetir:

- **Fiziksel Veri Tabanı:** Verileri bir klasördeki dosya veya dosyalar olarak saklar.
- **Veri Tabanı Motoru:** Veriye erişim, düzenleme ve benzeri işlevleri sağlar.
- **Veri Tabanı Şeması:** Verilerin mantıksal olarak yapılandırılmasını sağlar.

DBMS; yedekleme, kurtarma, performans izleme, etkinlik izleme gibi görevlerden de sorumludur. Böylece daha az hata oluşur, üretkenlik ve verimlilik artar, manuel yapılması uzun sürebilecek işlemler kısa zamanda tamamlanır (Görsel 8.4).

Görsel 8.4: Veri tabanı yönetim sistemleri

DBMS'ler şu türlere ayrılır:

- **İlişkisel (Relational) Veri Tabanı Yönetim Sistemi (RDBMS):** En yaygın kullanılan veri tabanı yönetim sistemi türüdür. RDBMS'ler, verileri tablolar hâlinde depolar. Tablolar, sütunlar ve satırlardan oluşur. Sütunlar, tablodaki verileri ifade eder. Satırlar ise tablodaki verileri satır olarak temsil eder.
- **Nesne Tabanlı (Object Oriented) Veri Tabanı Yönetim Sistemi (OODBMS):** Verileri nesneler hâlinde depolar. Nesneler, veri ve işlevlerden oluşur. Veriler, nesnelerin özelliklerini ifade eder. İşlevler ise nesnelerin davranışlarını temsil eder.
- **NoSQL (Not Only SQL) Veri Tabanı Yönetim Sistemi:** İlişkisel olmayan verileri depolamak için tasarlanmıştır. Veriler, tablolar hâlinde değil ağaç şeklindeki (gövde ve dallar) veri yapılarında depolanır.

Sıra Sizde

Birincil anahtar olarak nitelendirilen alanlara örnekler veriniz. Bu alanların neden birincil anahtar olarak nitelendirildiğini arkadaşlarınızla sınıfta tartışınız.

8.1.1. Veri Tabanı Yönetim Sistemlerinin İşlevleri

Veri tabanı yönetim sistemlerinin işlevleri şunlardır:

- **Depolama:** Verileri ölçeklenebilir bir şekilde depolamak için çeşitli teknikler kullanır.
- **Yönetim:** Verileri tutarlı bir şekilde yönetmek için doğrulama ve onaylama yöntemleri kullanır.
- **Sorgulama:** Verileri hızlı bir şekilde sorgulamak için teknikler kullanır.
- **Raporlama:** Verileri görsel bir şekilde raporlamak için çeşitli grafik ve tablolar kullanır.

8.1.2. Veri Tabanı Yönetim Sistemlerinin Avantajları

Veri tabanı yönetim sistemlerinin geleneksel bilgisayar dosya işleme yaklaşımıyla karşılaştırıldığında bir takım avantajları vardır. DBA, veritabanlarını tasarlarken, DBMS'yi koordine ederken ve izlerken bu faydaları veya yetenekleri göz ününde tutulmalıdır. Veri tabanı yönetim sistemlerinin avantajları şunlardır:

- **Fazlalıkların Azaltılması:** Verilerin veri tabanı yöneticisi tarafından merkezî kontrolü, verilerin gereksiz kopyalanmasını önler.
- **Verilerin Paylaşımı:** Bir veri tabanı, kontrolü altındaki verilerin herhangi bir sayıda uygulama programı veya kullanıcı tarafından paylaşılmasına olanak tanır.
- **Veri Bütünlüğü:** Veri tabanında yer alan verilerin hem doğru hem de tutarlı olduğu anlamına gelir.
- **Veri Güvenliği:** Veri tabanındaki veriler, kimlik doğrulama ve hassas verilere erişime izin verilmeden önce erişim prosedürlerinin takip edilmesini sağlar.
- **Çatışma Çözümü:** Çeşitli kullanıcı ve uygulamaların gereksinimlerindeki çatışmayı çözer.
- **Veri Bağımsızlığı:** Fiziksel veri bağımsızlığı, kavramsal görünümde veri tabanını kullanan uygulama programlarında değişikliğe gerek kalmadan fiziksel depolama cihazlarında işlem yapılmasına olanak tanır. Mantıksal veri bağımsızlığı ise kavramsal şemanın, mevcut haricî şemayı veya herhangi bir uygulama programını etkilemeden değiştirilebileceğini belirtir.

8.1.3. Veri Tabanı Yönetim Sistemlerinin Dezavantajları

Geleneksel veri işleme sistemine göre verilere erişen kullanıcı sayısının daha fazla olması nedeniyle veri tabanı yönetim sistemi birtakım riskler taşıyabilir. Veri tabanı yönetim sistemlerinin dezavantajları şunlardır:

- DBMS yazılım ve donanım maliyeti yüksektir.
- Verilerin güvenliğinin, bütünlüğünün ve paylaşımının uygulanması için DBMS tarafından yapılan işlem yükü vardır.
- Merkezî veri tabanının kontrolü zordur.
- Veri tabanı yönetim sistemlerinin kurulumu daha fazla bilgi, para, beceri ve zaman gerektirir.
- Veri tabanının karmaşıklığı, performansın düşmesine yol açar.

Sıra Sizde

Tablo oluşturabileceğiniz elektronik tablo programları varken neden veri tabanı yönetim sistemlerini kullanırsınız? Düşüncelerinizi sınıfta arkadaşlarınızla paylaşınız.

Veri tabanı yönetim sistemlerinin örnekleri Tablo 8.1'de verilmiştir. Bu sistemlere SAP HANA, Amazon Redshift, Google Cloud Bigtable, Microsoft Azure eklenebilir.

Tablo 8.1: Veri Tabanı Yönetim Sistemleri Örnekleri

DBMS	Açıklamalar
MySQL	Ücretsiz ve açık kaynaklı bir ilişkisel veri tabanı yönetim sistemi türüdür. C ve C++ dillerinde yazılmıştır. Küçük web siteleri ve işletmelerden büyük kurumsal uygulamalara kadar çok çeşitli uygulama için uygundur. Verileri bir veya daha fazla tabloda saklayan ve düzenleyen bir veri tabanıdır. Windows, Solaris, Linux, macOS, FreeBSD gibi birçok işletim sistemini destekler.
PostgreSQL	Nesne ilişkisel bir veri tabanı yönetim sistemidir. C dilinde yazılmıştır. Açık kaynaklı, ilişkisel bir veri tabanı yönetim sistemidir. MySQL'e benzer özelliklere sahiptir ancak daha güçlü ve ölçeklenebilirdir. Linux, OpenBSD, Windows, macOS ve FreeBSD dâhil olmak üzere birçok işletim sistemini destekler.
Oracle	C, Assembly ve C++ dillerinde yazılmıştır. Ticari bir ilişkisel veri tabanı yönetim sistemidir. Büyük ölçekli işletmeler için tasarlanmıştır, yüksek performans ve güvenilirlik sunar.
MongoDB	Açık kaynaklı, NoSQL, belge odaklı bir veri tabanı yönetim sistemidir. Python, C++ ve JavaScript ile yazılmıştır. Dizilere, gömülü belgelere izin verir ve tek bir kayıt kullanarak hiyerarşik ilişkileri temsil eder. Belgede tanımlanan anahtarlar şema içermediği için sabit değildir. Linux, Windows Vista, Solaris, FreeBSD, Mac OS X gibi işletim sistemlerini destekler.
MS Access	C++ ile yazılmıştır. İlişkisel bir veri tabanını grafik kullanıcı arayüzü ile birleştirir. Bilgisayar sistemindeki verileri veya bilgileri toplar. Tablo şeklinde saklanan veriler birbiriyle ilişkilidir. Visual Basic adı verilen üst düzey bir programlama dilini de destekler.
Microsoft SQL Sunucusu	Verileri sistematik bir şekilde depolayan RDBMS'di. C++ ve C dillerinde yazılmıştır. Ticari bir ilişkisel veri tabanı yönetim sistemidir. Oracle ile benzer özelliklere sahiptir ancak Windows platformu için tasarlanmıştır.
IBM DB2	Assembly, C++, C ve Java dillerinde yazılmıştır. İlişkisel modeli, nesne ilişkisel ve ilişkisel olmayan yapıları destekler. Hızlı ve güvenlidir. Bir kullanıcı, veri tabanına her erişmeye çalıştığında yetkilendirmeyi kontrol eder. Küçük ve orta ölçekli kuruluşlar için uygundur. Kullanıcıların SQL ile veri tabanlarına erişmesine, oluşturmasına, güncellemesine, silmesine ve bakımını yapmasına olanak tanır. Unix, Linux, Windows, OS/2 gibi işletim sistemlerini destekler.

8.2. VERİ TABANI YÖNETİM SİSTEMLERİNİN KURULUMU

Veri tabanı yönetim sistemi ile ilgili işlemleri yapabilmek için önce kurulum gerçekleştirilir. Kurulum için gerekli yazılımların ilgili internet sitesinden indirilmesi gereklidir.

1. Uygulama

İşlem adımlarına göre veri tabanı yönetim sisteminin kurulumunu gerçekleştiriniz.

1. Adım: **SQL Server Express** kurulum dosyasını indirdikten sonra kurulumu **Basic** seçeneği ile başlatınız.

2. Adım: Lisans sözleşmesini onayladıktan sonra kurulum klasörü belirleyiniz.

3. Adım: **SQL Server Express** kurulumu tamamlandıktan sonra **SQL Server Management Studio** kurulumunu başlatınız.

4. Adım: **SQL Server Management Studio** kurulumunu tamamlayınız.

Sıra Sizde

MySQL veri tabanı yönetim sistemini kurunuz ve gerekli yapılandırmaları yaparak, veri tabanı yönetim sisteminde tablo oluşturup bu tabloya veri girişi sağlayınız.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. Kurulum dosyasını indirdi.		
2. Kurulum türünü belirledi.		
3. Kurulumu başlattı.		
4. Kullanıcı hesabını oluşturdu.		
5. Yapılandırma işlemini gerçekleştirdi.		
6. Veri tabanı yönetim sistemi üzerinde veri tabanı oluşturdu.		
7. Veri tabanı tabloları oluşturdu.		
8. Oluşturduğu veri tabanı tablolarına veri girişi yaptı.		

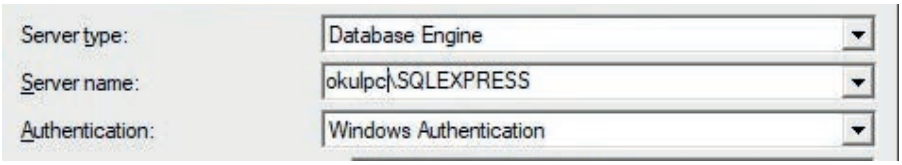
"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.

8.3. VERİ TABANI YÖNETİM SİSTEMLERİNİN KULLANIMI

Veri tabanı yönetim sistemi yüklendikten sonra veri tabanı bağlantı arayüzü ile veri tabanı üzerinde tablo oluşturma, tablolara veri girme, tablolar arasında ilişki oluşturma gibi işlemler yapılır.

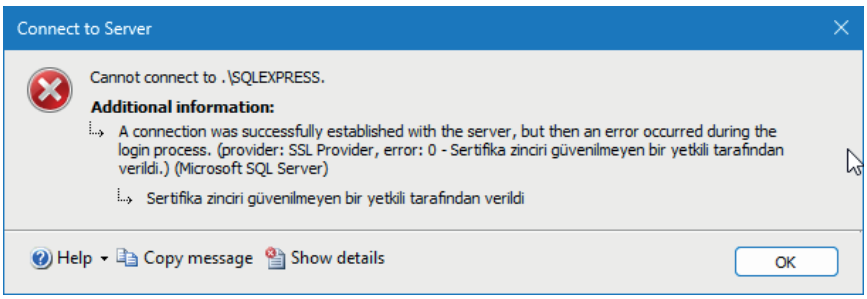
8.3.1. Veri Tabanı Yönetim Sistemi Arayüzü

Veri tabanı yönetim sistemi arayüzü açıldığında karşılaşılan bağlantı penceresinde **Server Name** (Sunucu Adı) kutucuğuna **okulpc\SQLEXPRESS** gibi önce sunucu adı, "\" ve yanına örnek adı birleştirilerek yazılır. Bir başka deyişle "." karakteri ile bilgisayardaki veri tabanı sunucusuna bağlanılır. Yerel veya uzak sunucu ismi de girilebilir (Görsel 8.5).



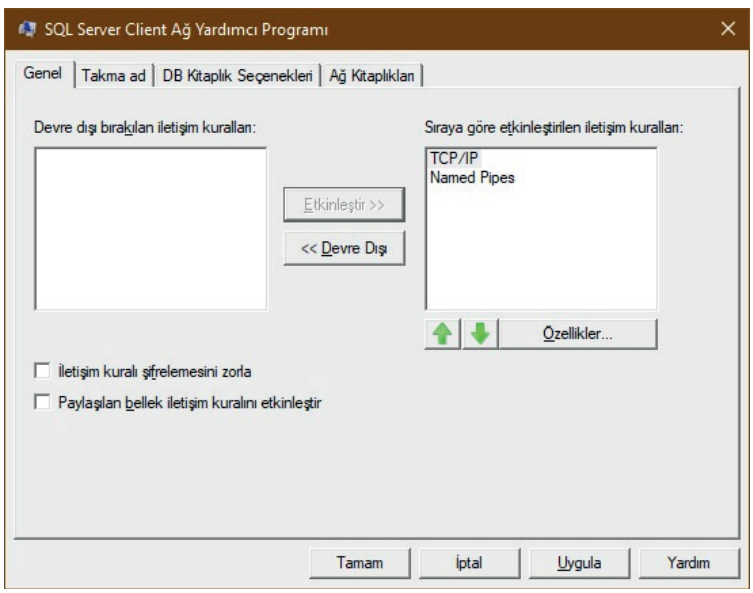
Görsel 8.5: Sunucu bağlantısı

Connect (Bağlan) tıklandığında Görsel 8.6'daki gibi bir hata ile karşılaşılırsa şu işlemler yapılır:



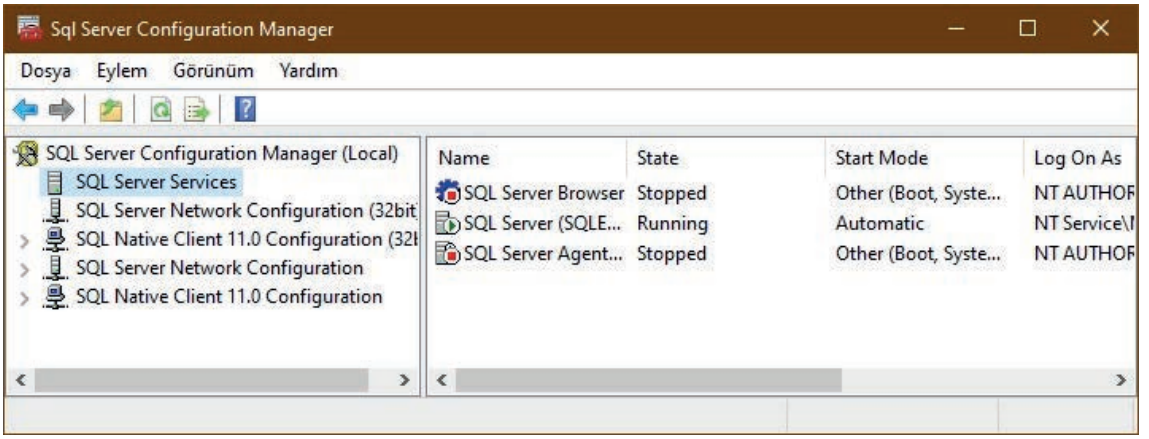
Görsel 8.6: Bağlantı hatası

a. **Başlat>Çalıştır>cliconfg.exe** iletişim kuralları TCP/IP üstte olacak şekilde kaydedilir (Görsel 8.7).



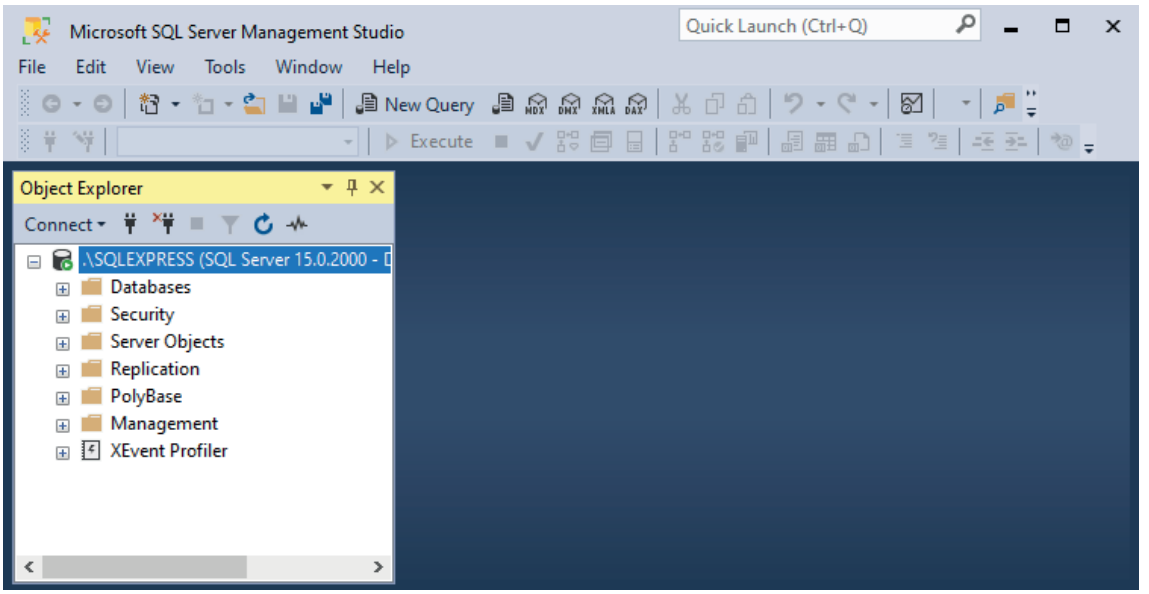
Görsel 8.7: TCP/IP etkinleştirilmesi

b. Ağ ayarı yapıldıktan sonra **Başlat>SQL Server Configuration Manager** ile açılan pencereden ilgili sunucu yeniden başlatılır (Görsel 8.8).



Görsel 8.8: Sunucu hizmetinin yeniden başlatılması

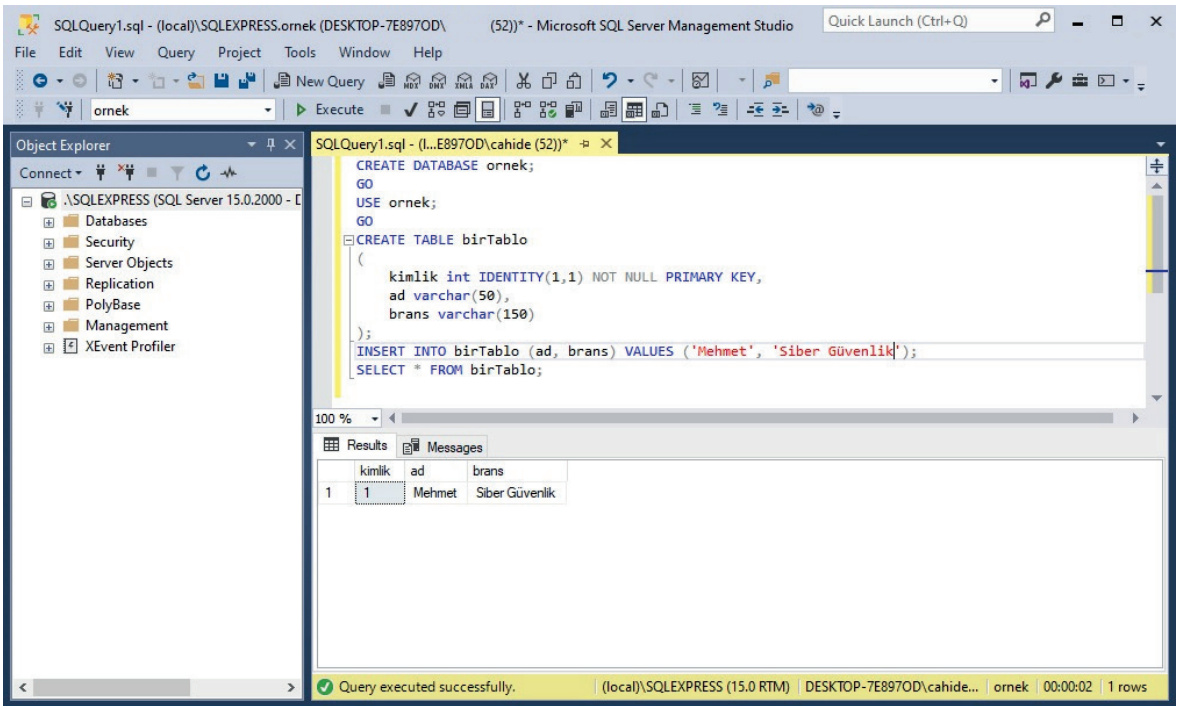
Sunucuya başarılı bir şekilde bağlanıldığında ekranda veri tabanlarının görüntülendiği Görsel 8.9'daki gibi boş bir pencere açılır.



Görsel 8.9: Boş bir SQL sunucusu çalışma ekranı

Örnek

Bir veri tabanı ve bir tablo oluşturma örnek kodunu **File>New>Query with Current Connection** veya **Ctrl+N>New** kısayolu ile açılan pencereye yazıp çalıştırınız. Bu kodu çalıştırma kısayolu **F5 (Execute)** tuşudur. Kod çalıştırıldığında sırasıyla bir veri tabanı oluşturulur, bir tablo oluşturulur, tabloya örnek veri eklenir ve ekranda tablo listelenir (Görsel 8.10).



Görsel 8.10: Örnek kodun çalıştırılması

Notlar

- Belli bir alan seçilerek F5 tuşuna basıldığında sadece seçili satırlar çalıştırılabilir.
- Örnek kod birden fazla çalıştırılırsa hata ile karşılaşılır.

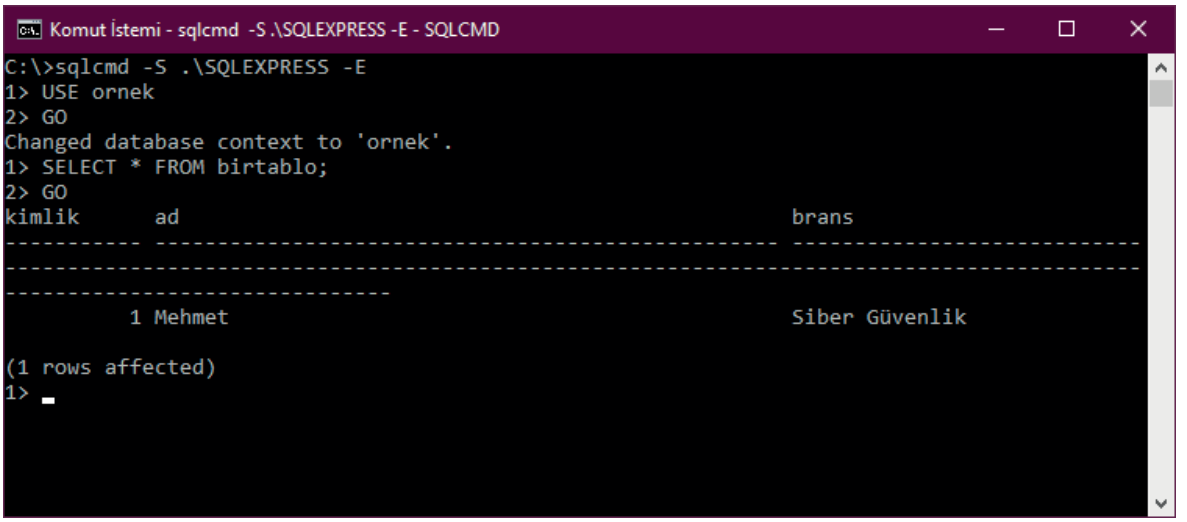
Hem **ornek** veri tabanı hem de **birtablo** nesnesi önceden oluşturulduğu için CREATE komutu hatasız yazılmışsa bir kere çalıştırılabilir. **ornek** veri tabanı silinerek tekrar örnek kod çalıştırılabilir.

```
USE master;
GO
DROP DATABASE ornek;
```

USE master komutu ile başka bir veri tabanı seçilir, DROP DATABASE ile istenen veri tabanı soru sorulmadan silinir. DROP ve DELETE komutu sonrasında silinen unsurların geri alınma imkânı yoktur. Sadece veri tabanının yedeği varsa silinen unsurlar geri yüklenebilir veya manuel olarak tekrar oluşturulur.

8.3.2. Veri Tabanı Yönetim Sistemi Yapılandırmaları

Tasarım (Design) penceresi veya komut kullanılarak iki yöntem ile tablo oluşturulabilir. Geliştiriciler her zaman tasarım penceresine erişemeyebilir. Örneğin uzaktan erişim ile sunucuya girilerek yönetilmesi gereken durumlarda konsol ekranı kullanılır. Konsol ekranı adı verilen komut isteminden SQL komutlarının tümü çalıştırılabilir (Görsel 8.11).

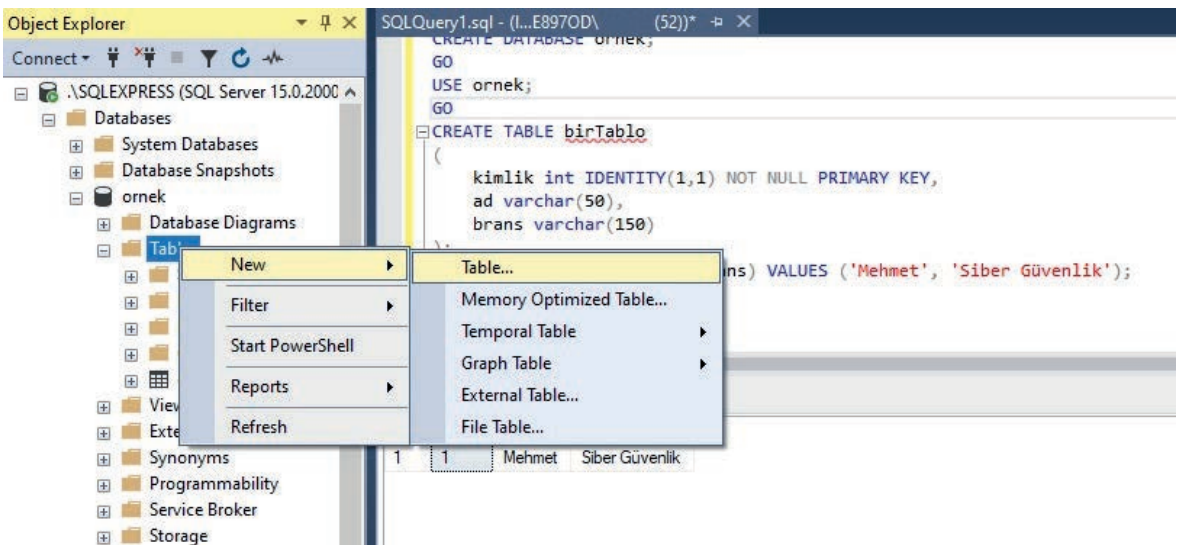


Görsel 8.11: Komut istemine girilerek SQL komutlarının çalıştırılması

2. Uygulama

İşlem adımlarına göre veri tabanı yönetim sistemini kullanarak veri tabanına tablo oluşturma ve tabloya veri ekleme işlemlerini yapınız.

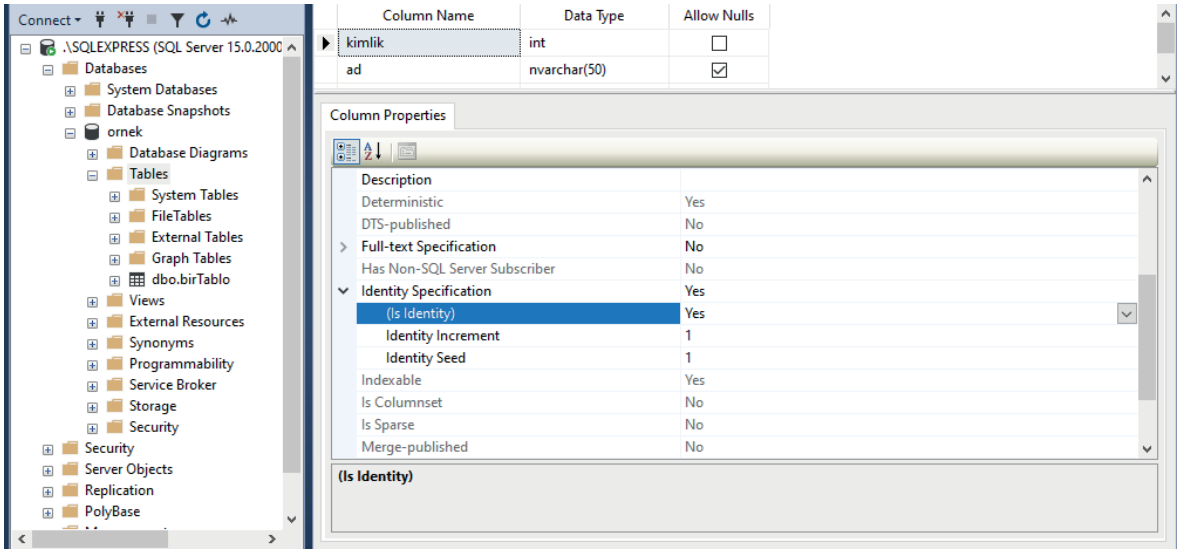
1. Adım: Tablo eklemek için Görsel 8.12'deki gibi **Tables** klasörüne sağ tıklayıp **New>Table**... komutunu veriniz.



Görsel 8.12: Yeni tablo eklenmesi

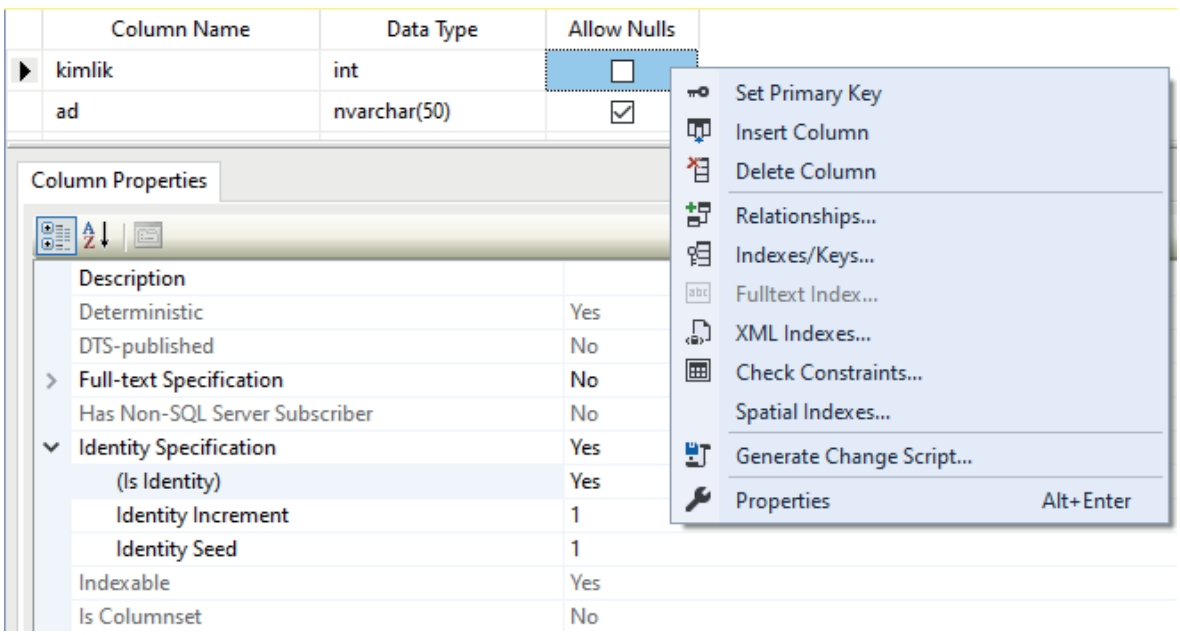
Not Her tabloda genellikle ilk sütun, **birincil anahtar** olarak belirlenir. Birincil anahtar, boş geçilemeyen ve tekil değerler içermesi gereken bir alandır.

2. Adım: Kimlik adındaki sütunda sayısal (int-tam sayı türü) ve otomatik artan özelliği olarak **(Is Identity)** kutucuğunu seçiniz (Görsel 8.13).



Görsel 8.13: Kimlik alanının otomatik artan sayı olarak belirlenmesi

3. Adım: Kimlik alanının birincil anahtar olması için alan üzerinde sağ tıklayıp **Set Primary Key** seçiniz (Görsel 8.14).



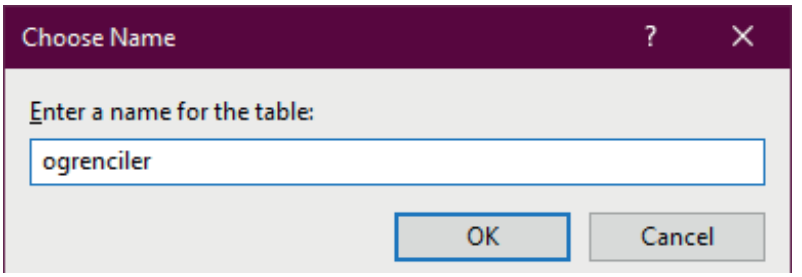
Görsel 8.14: Kimlik alanının birincil anahtar yapılması

4. Adım: Tabloya ad (veri türü olarak nvarchar(50)-metin) ve soyad (veri türü olarak nvarchar(50)-metin) alanlarını ekleyiniz (Görsel 8.15).

Column Name	Data Type	Allow Nulls
kimlik	int	☐
ad	nvarchar(50)	☑
soyad	nvarchar(50)	☑
		☐

Görsel 8.15: Ad ve soyad alanlarının eklenmesi

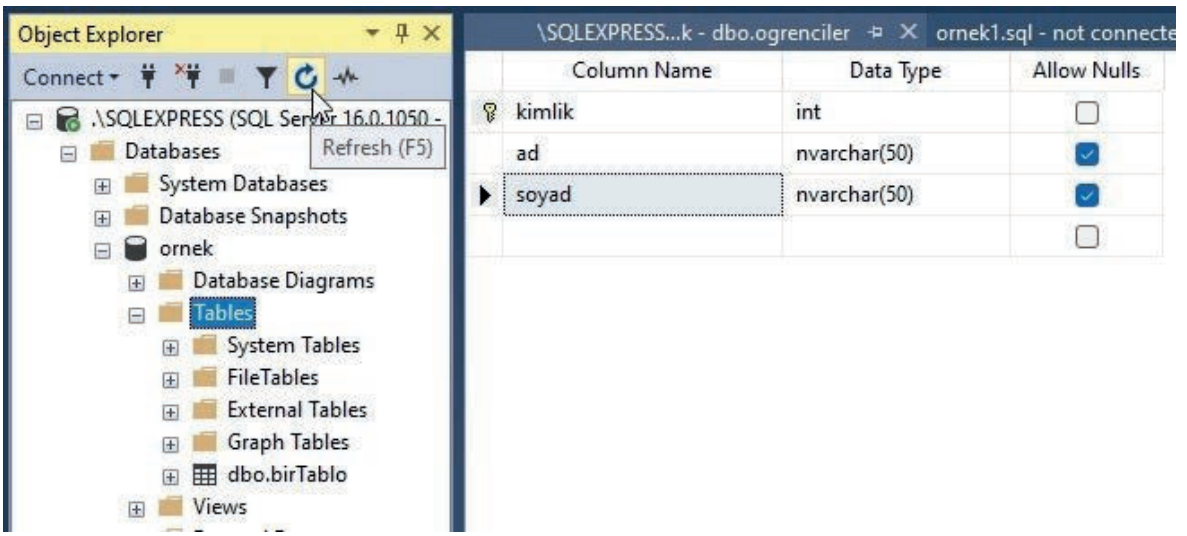
5. Adım: Tablonun kaydedilmesi için **Ctrl+S (Save-Kaydet)** komutunu verip, tablo adını **ogrenciler** olarak kaydediniz (Görsel 8.16).



Görsel 8.16: Tabloya isim verilmesi

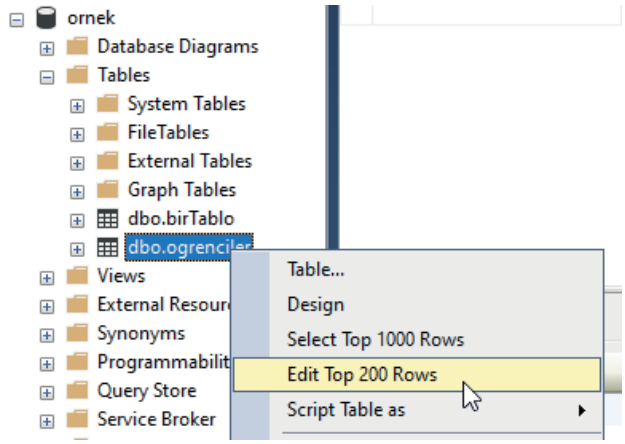
Not Tablo ve alan adlarında Türkçe karakterler, boşluk ve özel karakterler kullanmamaya çalışınız.

6. Adım: Tablo ismi, sol taraftaki Object Explorer'da (Nesne Gezgini) görülmezse Görsel 8.17'deki gibi Refresh (Yenile) düğmesine basınız.



Görsel 8.17: Yeni tablonun görülmesi

7. Adım: Tabloya veri girmek için Görsel 8.18'deki gibi sağ tıklayıp **Edit Top 200 Rows** komutunu veriniz.



Görsel 8.18: Tabloya kayıt eklenmesi

8. Adım: Tabloya örnek veriler girerek tablodaki verilerde silme ve değiştirme işlemleri yapınız (Görsel 8.19). Tablodaki değişiklikler otomatik olarak kaydedilir.

	kimlik	ad	soyad
	1	Ahmet	K
	2	Mehmet	T
▶*	NULL	NULL	NULL

Görsel 8.19: Verilerin düzenlenmesi

Sıra Sizde

Kütüphane veri tabanında Kitap tablosunu oluşturarak Kitap Numarası ve Kitap Adı alanlarını oluşturunuz. Tabloyu oluşturduktan sonra birincil anahtarı belirleyip tabloya veri girişi sağlayınız.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. Birincil anahtarı belirledi.		
2. Tablonun alanlarını oluşturdu.		
3. Tabloya isim verdi.		
4. Tabloyu kaydetti.		
5. Tabloya veri girişi yaptı.		
"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.		

8.3.3. Veri Tabanı Normalizasyonu

Veri tabanı normalizasyonu, verileri düzenli ve tutarlı bir şekilde düzenlemek için kullanılan bir veri tabanı tasarım ilkesidir. Normalleştirme kuralları, büyük tabloları daha küçük tablolara böler ve ilişkileri kullanarak bunları birbirine bağlar. SQL'de normalleştirmenin amacı, tekrarlayan verileri ortadan kaldırmaktır. Veri tabanı yöneticileri; birincil, yabancı ve bileşik anahtarları kullanarak bu ilişkileri sağlayabilirler. Normalizasyon kurallarına uygun tasarım yapılmadığında kayıt güncelleme, kayıt bulma ve yeni kayıt işlemlerinde sorunlarla karşılaşılabilir. Normalizasyon yapılırken uyulması gereken kuralların her birine **normal form** adı verilir. Normalleştirme kurallarından bazıları şunlardır:

Birinci Normal Form (1NF)

Birinci normal formda, veri tabanında bulunan tablolar ilişkilendirilebilir bir şekilde tasarlanmalıdır. Birden fazla bilgi tek bir sütunda olmamalıdır. Bir alan içindeki bilgi, özel karakterlerle ayrılarak tutulmuş olmamalıdır. Bir veri tabanı bu kurallara uyuyorsa 1NF, bir başka deyişle birinci normal formdadır.

Örnek

Veri tabanı normalizasyon örneği için Tablo 8.2'deki bilgiler, 1NF kurallarına göre incelenir. **Ders_kodu** ve **Sınav_S** sütunlarındaki her bir alanda birden çok veri tutulur. Veriler, virgüllerle ayrılmış durumdadır. Bu tabloya ilk normalizasyon kuralı uygulandığında elde edilen değerler Tablo 8.3'te verilmiştir.

Tablo 8.2: Normalizasyon Uygulanmayan Tablo

ogrenci_no	Ad Soyad	Bolum_kodu	Bolum	Ders_kodu	Sınav_S
20231101	Hasan O.	BTT	Bilgisayar	B101, B102	87, 65
20231102	Eda B.	BTT	Bilgisayar	B102, B103	45, 85
20231103	Ahmet B.	MTL	Metal	M201	60

Tablo 8.3: 1NF Kuralları Uygulanmış Tablo

ogrenci_no	Bolum_kodu	Ad Soyad	Bolum	Ders_kodu	Sınav_notu
20231101	BTT	Hasan O.	Bilgisayar	B101	87
20231101	BTT	Hasan O.	Bilgisayar	B102	65
20231102	BTT	Eda B.	Bilgisayar	B102	45
20231102	BTT	Eda B.	Bilgisayar	B103	85
20231103	MTL	Ahmet B.	Metal	M201	60

1NF sonucu elde edilecek tabloda tekrarlı satırlar bulunacağı için bu tabloya satır ekleme, satır silme ve satır güncellemede sorunlar yaşanabilir.

İkinci Normal Form (2NF)

1NF'de karşılaşılan güncelleme sorununu çözmek için tablo, 2NF kuralına uygun hâle getirilir. 1NF'yi 2NF'ye dönüştürmede uygulanacak kurallar şunlardır:

- Bir tablo içinde tanımlı ancak anahtar olmayan uygun sütunlar, anahtar olarak tanımlanmalı ve tanımlı birincil anahtar sütunlara bağlanmalıdır. Örneğin öğrenci bilgilerinin tutulduğu bir tabloda not ve ders bilgisinin olması gereksizdir. Bunu çözmek için öğrenci bilgileri ve not bilgileri ayrı tablolarda tutulmalıdır.
- Anahtar sütun birden fazla sütunun birleşiminden oluşuyorsa tabloda yer alacak veriler iki sütuna da bağımlı olmalıdır. Veriler tek sütuna bağımlı ise ayrı bir tabloda tutulmalıdır. Örneğin bir ders farklı bölümlerde veriliyorsa ders kodu, bölüm koduyla beraber bir anahtar olarak tanımlanmalıdır.

2NF uygulanan Tablo 8.4'te **Ogrenci_no** birincil anahtar olarak tanımlanmalı ve Tablo 8.5'te **Dersler** tablosundaki yabancı anahtar, tanımlanacak **Ogrenci_no** sütunu ise özlük tablosundaki **Ogrenci_no** alanı ile ilişkilendirilmelidir.

Tablo 8.4: 2NF Kuralları Uygulanmış Özlük Tablosu

Ogrenci_No	Ad Soyad	Bolum_kodu	Bolum
20231101	Hasan O.	BTT	Bilgisayar
20231102	Eda B.	BTT	Bilgisayar
20231103	Ahmet B.	MTL	Metal

Tablo 8.5: 2NF Kuralları Uygulanmış Dersler Tablosu

Ogrenci_No	Ders _kodu	Sınav_notu
20231101	B101	87
20231101	B102	65
20231102	B101	45
20231102	B103	85
20231103	M201	60

2NF sonucunda güncelleme sorunu çözülür ancak kayıt ekleme ve kayıt silme sorunu devam eder.

Üçüncü Normal Form (3NF)

3NF'de bir tablo içinde anahtar olmayan bir sütun, başka bir tablonun anahtar sütunu veya bulunduğu tablonun sütunlarıyla ilgili olmalıdır. Ayrıca veri tabanındaki ilişkiler, 2NF kuralına uymalıdır. Öğrenci tablosu için Bolum>Bolum_kodu geçişli bağımlılığı mevcuttur. Öğrenci tablosundaki sorunlardan kurtulmak için bu bağımlılığın da kaldırılması gereklidir. Bağımlılığı kaldırmak için bölümler, ayrı bir tablo olarak oluşturulmalıdır. Bu düzenlemeden sonra Öğrenci Özlük Tablosu, Öğrenci Tablosu ve Bölümler Tablosu şeklinde iki ayrı tablo hâline getirilebilir. Tablo 8.6'daki Bolum_kodu ve Tablo 8.7'deki Bolum_k arasında oluşturulacak ilişkiyle veri bütünlüğü sağlanır.

Tablo 8.6: 3NF Kuralları Uygulanmış Öğrenci Tablosu

Ogrenci_No	Ad_Soyad	Bolum_kodu
20231101	Hasan O.	BTT
20231102	Eda B.	BTT
20231103	Ahmet B.	MTL

Tablo 8.7: 3NF Kuralları Uygulanmış Bölümler Tablosu

Bolum_k	Bolum
BTT	Bilgisayar
MTL	Metal

3. Uygulama

İşlem adımlarına göre veri tabanı yapılandırması için Tablo 8.8'deki Proje bilgilerinin tutulduğu Proje Tablosuna normalizasyon kurallarını uygulayınız.

Tablo 8.8: Proje Tablosu

OgrenciNo	OgrenciAdı	Sehir	ProjeNo	Kurs	icerik
Ogrenci1	Erdal U.	Ankara	P1	Programlama	C++, Java, C
Ogrenci2	Emel B.	Bursa	P2	WEB	HTML, PHP, ASP

1. Adım: Tekrarlanan alanlardan oluşan normalleştirilmemiş Tablo 8.8'e 1NF kuralını uygulayınız.

Ogrenci Tablosu

OgrenciNo	OgrenciAdı	Sehir
Ogrenci1	Erdal U.	Ankara
Ogrenci2	Emel B.	Bursa

Proje Tablosu

OgrenciNo	ProjeNo	ProjeAdi	Kurs	Icerik
Ogrenci1	P1	Sağlık	Programlama	C++
Ogrenci1	P1	Sağlık	Programlama	Java
Ogrenci1	P1	Sağlık	Programlama	C
Ogrenci2	P2	Sosyal	WEB	HTML
Ogrenci2	P2	Sosyal	WEB	PHP
Ogrenci2	P2	Sosyal	WEB	ASP

2. Adım: 1NF uygulanmış tabloların kısmi bağımlılıklarını kurtararak 2NF normal formda olmasını sağlayınız.

Ogrenci Tablosu

OgrenciNo	OgrenciAdı	Sehir
Ogrenci1	Erdal U.	Ankara
Ogrenci2	Emel B.	Bursa

Proje Tablosu

OgrenciNo	ProjeNo	ProjeAdi	KursNo
Ogrenci1	Proje1	Sağlık	Kurs1
Ogrenci1	Proje1	Sağlık	Kurs2
Ogrenci1	Proje1	Sağlık	Kurs3
Ogrenci2	Proje2	Sosyal	Kurs4
Ogrenci2	Proje2	Sosyal	Kurs5
Ogrenci2	Proje2	Sosyal	Kurs6

Kurs Tablosu

KursNo	Kurs	Icerik
Kurs1	Kütüphane Sistemi	C++
Kurs2	Kütüphane Sistemi	Java
Kurs3	Kütüphane Sistemi	C
Kurs4	Otel Sistemi	HTML
Kurs5	Otel Sistemi	PHP
Kurs6	Otel Sistemi	ASP

Sıra Sizde

Üçüncü uygulamadaki 2NF kuralı uygulanmış tablolarda geçişli bir bağımlılığa neden olan Sehir alanının öğrenci kimliği alanıyla ilişkisini ortadan kaldırarak tablolara 3NF kuralını uygulayınız.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. Sehir tablosunu oluşturdu.		
2. Sehir tablosunun birincil anahtarını belirledi.		
3. Ogrenci tablosundaki Sehir sütununu sildi.		
"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.		

8.3.4. Tablolar Arası İlişkiler Oluşturma

Bir web veya mobil uygulamada genellikle birden fazla tablo kullanılır. Tek tabloda tüm verileri saklamak hem performansı düşürür hem de büyük bir tabloda işlem yapmak zorlaşır. Veri tabanı içinde veri tekrarlarının ve hatalı veri girişlerinin olmaması gerekir. Tablolar arasındaki ilişkiler; veri tutarlılığını korumaya, veri tabanı performansını iyileştirmeye ve verileri daha verimli bir şekilde sorgulamaya yardımcı olur.

Tablolar arasındaki ilişkiler, tabloların sütunları arasında ortak bir **anahtar (key)** sütunuyla tanımlanır. Bir anahtar sütun, her tablodaki benzersiz bir sütundur ve tablolar arasındaki ilişkiyi tanımlamak için kullanılır. İlişkisel veri tabanlarında dört tür temel tablo ilişkisi vardır:

1. **Bire Bir (1→1):** Bir tablodaki her satır, başka bir tablodaki tek bir satırla eşleşir. Örneğin bir kullanıcının yalnızca bir adresi olabilir ve bir adres yalnızca bir kullanıcıya aittir.

Kullanıcı Tablosu

PK

Sıra	İsim	Yaş
1	Ali	25
2	Ahmet	35
3	Emel	45
5	Ayşe	55

Adres Tablosu

PK PK

Sıra	Sokak	Şehir
1	1 Lale Sokak	Ankara
2	2 Gül Sokak	İzmir
5	3 Gonca Sokak	Konya

2. **Birden Çoğa (1→∞):** Bir tablodaki bir satır, başka bir tablodaki birden fazla satırla eşleşir. Örneğin bir inceleme yalnızca bir kitaba aittir. Bir kitabın birçok incelemesi vardır.

Kitap Tablosu

PK

Sıra	Başlık	Yazar	Basım Yılı
1	SQL Server	Meryem P.	1999
2	Türkçe	Ali Y.	2000
3	Matematik	Demet Ç.	1978

İnceleyen Tablosu

PK FK

Sıra	Kitap Sıra	İnceleyen	Değerlendirme
1	1	Can T.	4
2	2	Can T.	5
3	2	Demet Ç.	1

3. **Çoktan Çoğa (∞→∞):** Bir tablodaki her satır, başka bir tablodaki birden fazla satırla eşleşir. Örneğin bir kullanıcı birçok kitaba göz atabilir. Bir kitap, birçok kullanıcı tarafından (zaman içinde) ödünç alınabilir.

Kullanıcı Tablosu

PK FK **FK**

Sıra	Kullanıcı_No	Kitap _No
1	11	1
2	11	2
3	27	2
4	51	3

Kitap Tablosu

4. Çoktan Bire (∞→1): Bir tablodaki her satır, başka bir tablodaki tek bir satırla eşleşir.

Kullanıcı Tablosu

Sıra	İsim	Yaş
1	Ali	27
2	Ahmet	35
3	Emel	40
5	Ayşe	52

İnceleme Tablosu

Sıra	Kullanici_No	Kitap_No	İnceleme_tarihi	Yayın Tarihi
1	1	1	2012	2003
2	1	2	2022	2002
3	2	2	2023	2000
4	5	3	2000	2000

Kitap Tablosu

Sıra	Başlık	Yazar	Yayın Tarihi	ISBN
1	SQL Server	Meryem P.	2012	9814830291
2	Türkçe	Ali Y.	2022	8573009237
3	Matematik	Demet Ç.	2023	5231209678
5	SQL Server	Meryem P.	2000	5685698712

Sıra Sizde

ogrenci tablosu ile ders tablosu arasında odev tablosu yardımıyla çoktan çoka ilişki oluşturunuz.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. ogrenci tablosunu oluşturdu.		
2. ders tablosunu oluşturdu.		
3. odev tablosunu oluşturdu.		
4. Tabloların birincil anahtarlarını belirledi.		
5. Tablolar arasında ilişki oluşturdu.		

"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.

4. Uygulama

İşlem adımlarına göre veri tabanı yönetim sisteminde hazırlanan tablolar arasında ilişki oluşturunuz.

1. Adım: SQL kodunu çalıştırarak iki tablo ve bu tablolar içinde veriler olmasını sağlayınız.

```
CREATE DATABASE veritabani;
GO
USE veritabani;
GO
CREATE TABLE kategoriler(
        kimlik int PRIMARY KEY IDENTITY(1,1) NOT NULL,
        kategoriAdi nvarchar(50)
)
INSERT INTO kategoriler (kategoriAdi) VALUES ('Meyveler'), ('Sebzeler'), ('Baharatlar');

CREATE TABLE urunler(
        kimlik int PRIMARY KEY IDENTITY(1,1) NOT NULL,
        urunAdi nvarchar(50),
        kategoriKimlik int FOREIGN KEY REFERENCES kategoriler(kimlik)
)
INSERT INTO urunler (urunAdi, kategoriKimlik) VALUES
('Ayva, 1), ('Kavun, 1),('Salatalık, 2),('Domates, 2),('Kekik, 3);

SELECT * FROM kategoriler;
SELECT * FROM urunler;
```

Kodlar çalıştırıldığında ekrana Görsel 8.20'deki kategori ve ürün bilgileri gelir.

	kimlik	kategoriAdi
1	1	Meyveler
2	2	Sebzeler
3	3	Baharatlar

	kimlik	urunAdi	kategoriKimlik
1	1	Ayva	1
2	2	Kavun	1
3	3	Salatalık	2
4	4	Domates	2
5	5	Kekik	3

Görsel 8.20: İki tablodan gelen veri listeleri

2. Adım: Verilerin çoğalmasıyla birlikte her kategorinin kimlik bilgilerini akılda tutmak zorlaşır. Diğer tablodan ürünlerin isminin yanına **kategori adı** bilgisini getiriniz (Görsel 8.21).

```
SELECT * FROM urunler INNER JOIN kategoriler ON urunler.kategoriKimlik =kategoriler.kimlik
```

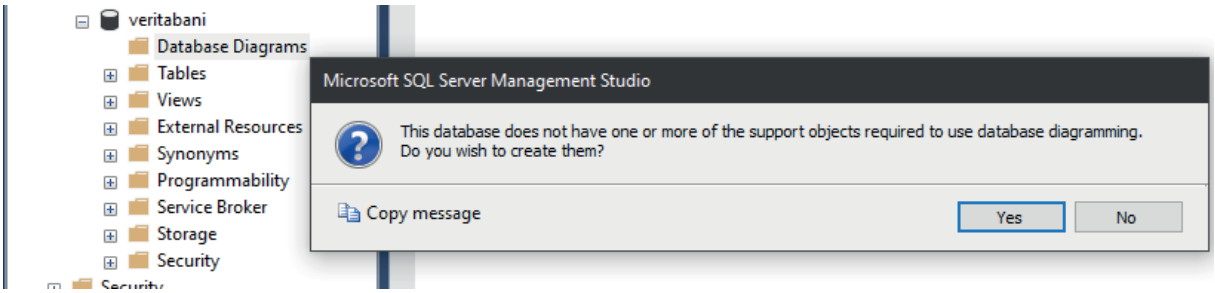
100 %

Results | Messages

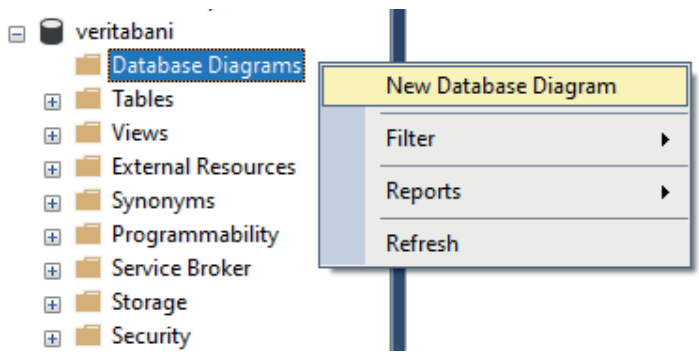
	kimlik	urunAdi	kategoriKimlik	kimlik	kategoriAdi
1	1	Ayva	1	1	Meyveler
2	2	Kavun	1	1	Meyveler
3	3	Salatalık	2	2	Sebzeler
4	4	Domates	2	2	Sebzeler
5	5	Kekik	3	3	Baharatlar

Görsel 8.21: Ürünlerle birlikte kategori adı sütunun ekranda listelenmesi

3. Adım: Diğer yöntem olarak tabloları ilişkilendirmek için **Database Diagrams** özelliğini kullanınız (Görsel 8.22, Görsel 8.23).

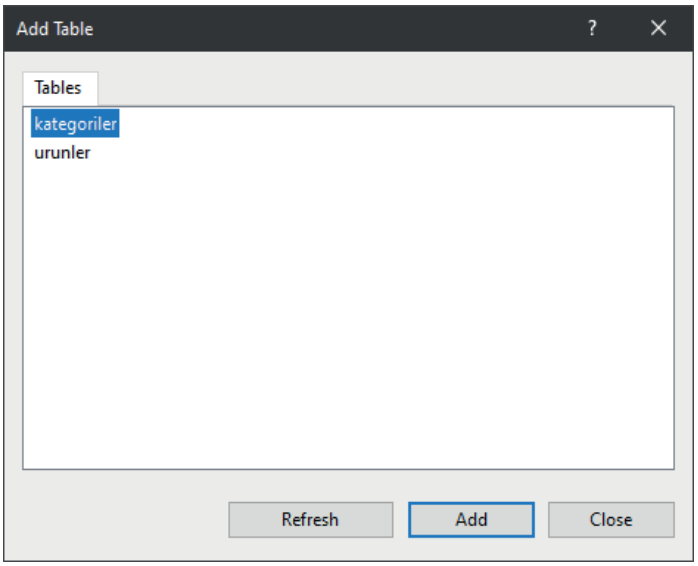


Görsel 8.22: Yeni veri tabanı şemasının eklenmesi



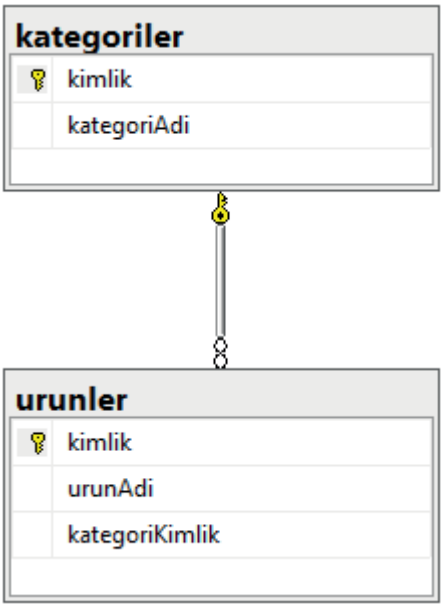
Görsel 8.23: Database Diagrams yöntemiyle yeni şema eklenmesi

4. Adım: Shift tuşu ile iki tabloyu seçip, Add tuşuna basarak tabloları ekleyiniz (Görsel 8.24).



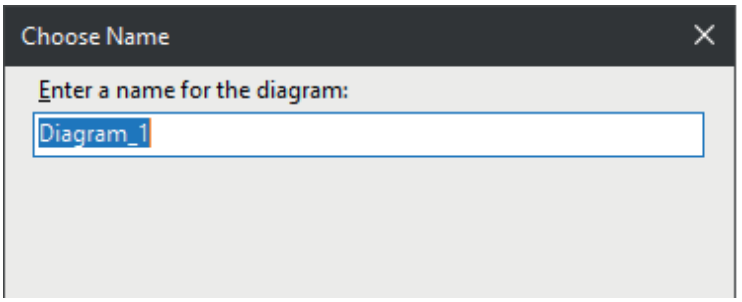
Görsel 8.24: Add tuşuna basılması

5. Adım: Tablolar arasındaki bağı görüntüleyiniz (Görsel 8.25).



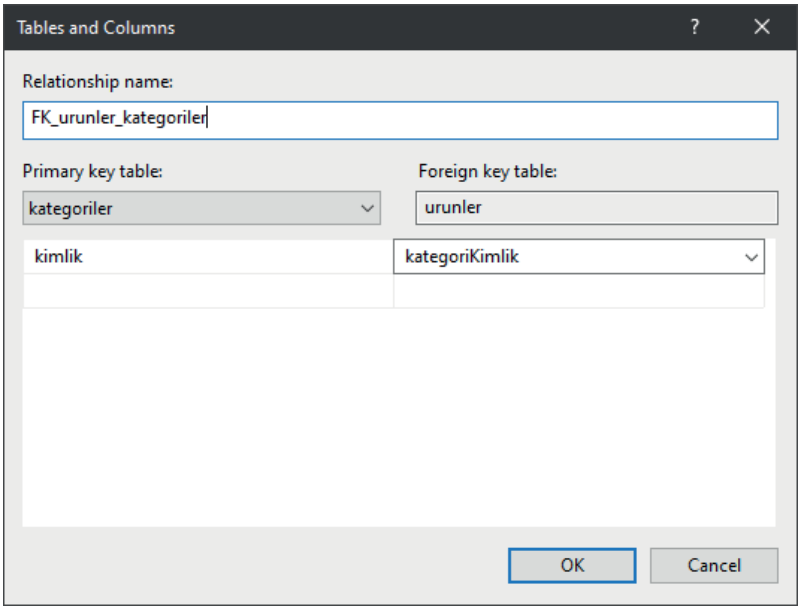
Görsel 8.25: Tablolar arası bağın görüntülenmesi

6. Adım: Diyagramı Ctrl+S ile kaydediniz (Görsel 8.26).

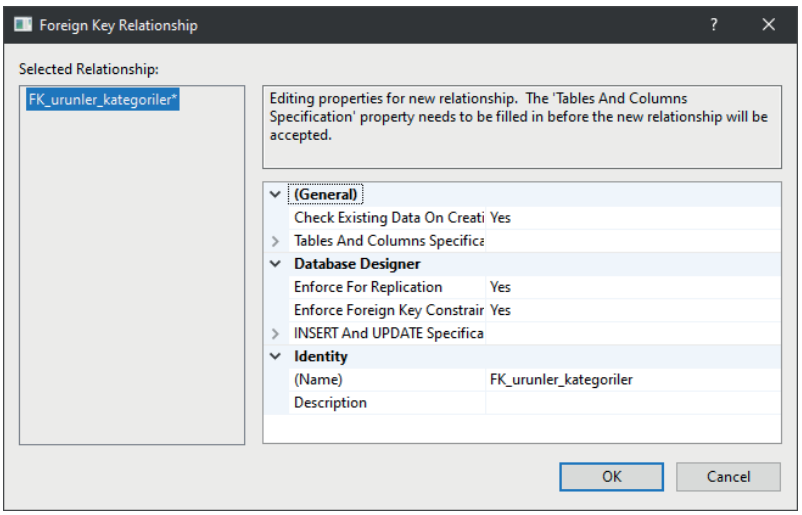


Görsel 8.26: Diyagram kaydedilmesi

7. Adım: Görsel 8.27 ve Görsel 8.28'deki ürünler tablosunda yer alan **kategoriKimlik** alanını, kategoriler tablosundaki **kimlik** alanına sürükleyerek diyagramın özelliklerini inceleyiniz.



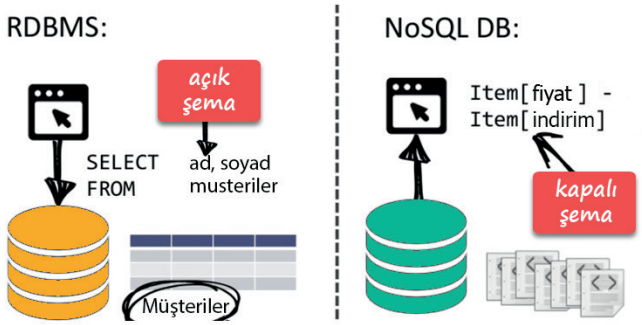
Görsel 8.27: Yeni ilişki tanımlama penceresi



Görsel 8.28: İkincil anahtar ilişki özelliklerinin düzenlenmesi

8.4. SQL ve NoSQL UYGULAMALARI

Veri tabanı sistemleri; büyük miktarda veriyi organize eden, yöneten ve uzun süre depolayan güçlü araçlardan oluşan uygulamalardır. Bu uygulamalarda karmaşık sistemler aracılığıyla çeşitli veriler üretilmiştir. Bu durum; büyük miktarda veri, veri tabanı yapısı, ölçeklenebilirlik, verilerin kullanılabilirliği gibi sorunlara yol açarak NoSQL terimini ortaya çıkarmıştır. Farklı sorunları çözmek için farklı veri tabanları tasarlanmıştır. Tüm gereksinimler için tek bir veri tabanı motorunun kullanılması genellikle başarısız çözümlere yol açmıştır. İlişkisel veri tabanı, çok miktarda veri açısından web uygulamalarının büyümesine neden olduğu için yapılandırılmamış verilere uygun bir veri tabanı sistemi değildir (Görsel 8.29).



Görsel 8.29: SQL ve NoSQL veri tabanları

İlişkisel veri tabanı, **SQL sistemleri;** ilişkisel olmayan veri tabanı ise **NoSQL sistemleri** olarak adlandırılır. NoSQL veri tabanı, ilişkisel veri tabanlarında kullanılan tablo ilişkilerinden farklı yöntemlerle modellenen verilerin depolanması ve alınması için bir mekanizma sağlar. NoSQL sistemlerinin avantajları arasında tasarımın basitliği, yatay ölçeklendirme ve kullanılabilirlik üzerinde daha hassas kontrolün yer alması sıralanabilir.

NoSQL veri tabanları, büyük veri ve gerçek zamanlı web uygulamalarında daha fazla kullanılır. Bu veri tabanları, daha yüksek ölçeklenebilirlik elde etmek için veri modeliyle ilgili konularda ilişkisel veri tabanları kadar katı kurallar içermez ve şemasız olmayı amaçlar.

SQL; ilişkisel veri tabanlarında verileri depolamak, yönetmek ve sorgulamak için kullanılan bir sorgulama dilidir. Web, mobil, IoT, masaüstü gibi yazılımların altyapısında kullanılabilir.

NoSQL, ilişkisel olmayan bir veri tabanı türüdür. İlişkisel veri tabanlarından farklı olarak verileri tablolar hâlinde depolamaz. Veriler, anahtar-değer çiftleri, belgeler, grafikler, bulut bilişim hizmetleri kullanılarak saklanır (Görsel 8.30). Geleneksel RDBMS, daha fazla bilgi edinmek amacıyla verileri depolamak ve almak için SQL sözdizimini kullanır. NoSQL veri tabanı sistemi; yapılandırılmış, yarı yapılandırılmış, yapılandırılmamış ve çok biçimli verileri depolayabilen çok çeşitli veri tabanı teknolojilerini kapsar.

Görsel 8.30: NoSQL veri tabanı

NoSQL veri tabanlarında tanımlı bir şema bulunmaz. Kayıtlar, NoSQL veri tabanlarında bütün ilişkiyi içeren bir satır olarak tutulur. NoSQL veri tabanı yapıları dağıtık biçimde çalıştırılır. NoSQL veri tabanı sistemleri SQL kullanmaz. Veri kümeleri üzerinde çalışır, tutarlılık ve dağıtım açısından kullanıcıya farklı seçenekler sunar.

NoSQL veri tabanları, ilişkisel veri tabanlarından daha esnektir ve büyük miktarda veriyi daha hızlı şekilde işleyebilir. Genellikle sosyal medya platformları, web ve büyük veri uygulamalarında kullanılır (Görsel 8.31).

Görsel 8.31: Büyük veri bileşenleri

NoSQL veri tabanı kavramı, büyük miktarda veriyle uğraşan Google, Facebook, Amazon gibi internet sitelerinin verilerinin artmasıyla önem kazanmıştır. Bu tür siteler, büyük hacimli veriler için RDBMS kullandığında sistemin yanıt süresi yavaşlar. Bu sorunu çözmek için mevcut donanımların yükseltilerek ölçeklendirilmesi veya veri tabanı yükünün birden fazla ana bilgisayara dağıtılması gerekir. NoSQL veri tabanı ilişkisel değildir. NoSQL veri tabanı , web uygulamaları göz önünde bulundurularak tasarlandığı için ilişkisel veri tabanlarından daha iyi ölçeklenir.

NoSQL veri tabanlarının bazı avantajları şunlardır:

- Esneklik
- Yüksek performans
- Büyük miktarda veri
- Ölçeklenebilirlik

NoSQL veri tabanlarının bazı dezavantajları şunlardır:

- İlişkisel veri tabanları kadar güçlü olmayabilir.
- Karmaşık sorgular için uygun olmayabilir.
- Veri güvenliği konusunda daha az güvenli olabilir.

En yaygın NoSQL veri tabanı türleri şunlardır:

Doküman Tabanlı Veri Tabanları: JSON gibi belgelere dayalı depolama modeline sahiptir. Veriler genellikle belirli bir belge koleksiyonu içinde saklanır ve bu belgeler, hiyerarşik bir yapıya sahip olabilir.

Anahtar ve Değer Tabanlı Veri Tabanları: Benzersiz anahtarlarla değerlerin ilişkilendirildiği basit bir yapıya sahiptir. Hızlı erişim sağlamak için anahtarlar kullanılır ve bunlar genellikle dağıtık bellek tabanlı (distributed memory-based) verilerdir.

Sütun Tabanlı Veri Tabanları: Veriler sütunlar hâlinde saklanır. Bu sayede okuma işlemleri yalnızca ilgili sütunlarla sınırlı kalır. Sütun tabanlı veri tabanları, büyük veri analitiği ve raporlama için kullanışlıdır.

Grafik Tabanlı Veri Tabanları: Veriler, düğümler ve kenarlardan oluşan grafik yapısına dayalı şekilde saklanır. İlişkisel verilerin ağ yapısını temsil etmek için kullanılır.

8.4.1. SQL Uygulamaları

SQL'in belli bir kod dizimi vardır ve sorgu yazılırken bu dizime uyulmalıdır. Bu dizimde bir yanlışlık yapılırsa sorgu çalışmaz, hata oluşur ve herhangi bir veri dönmez.

Örnek SQL Kod Örneği

```
SELECT alanAdi
        FROM tabloAdi
WHERE kosul
        ORDER BY
        DESC;
```

SQL kodunda önce veri tabanından belirtilen tabloyu çeken FROM işlemi çalışır. İkinci olarak filtreleme koşullarının yazıldığı WHERE işlenir. Veriler çekilip filtrelendikten sonra veri tabanına hangi alanların döndürüleceği SELECT ile belirlenir. Gerekli tüm satır ve sütunlar oluşunca ORDER BY ile sıralama işlevi gerçekleştirilir. Görsel 8.32'deki örnek tablolar kullanılarak SQL sorgu komutları incelenebilir.

Results Messages

	kimlik	kategoriAdi
1	1	Meyveler
2	2	Sebzeler
3	3	Baharatlar

	kimlik	urunAdi	kategoriKimlik
1	1	Elma	1
2	2	Karpuz	1
3	3	Domates	2
4	4	Kıvırcık	2
5	5	Karabiber	3

Görsel 8.32: Örnek tablolar

SELECT

SELECT, SQL'in en temel sorgulama komutudur. Tablolar ve aralarındaki ilişkileri göz önüne alarak istenen sorgulara göre tablolardan ve istenen veri alanlarından veriler toparlayıp raporlayan komuttur. SELECT komutunun kullanımı şu şekildedir:

```
SELECT <alanAdi> FROM <tabloAdi>
```

Örnek

Veri tabanındaki **kategoriler** tablosunda bulunan **kategoriAdi** sütunları Görsel 8.33'te görüldüğü gibi listelenir.

```
SELECT kategoriAdi FROM kategoriler
```

100 %

Results | Messages

	kategoriAdi
1	Meyveler
2	Sebzeler
3	Baharatlar

Görsel 8.33: SELECT komutunun kullanılması

Örnek

Veri tabanındaki **urunler** tablosunda bulunan tüm kayıtlar Görsel 8.34'te görüldüğü gibi listelenir.

```
SELECT * FROM urunler
```

100 %

Results | Messages

	kimlik	urunAdi	kategoriKimlik
1	1	Elma	1
2	2	Karpuz	1
3	3	Domates	2
4	4	Kıvırcık	2
5	5	Karabiber	3

Görsel 8.34: Tüm kayıtların listelenmesi

WHERE

WHERE anahtar sözcüğü ile sadece belirlenen kurala uygun kayıtların listelenmesi sağlanır.

Örnek

Veri tabanındaki **urunler** tablosunda **urunAdi** sütunu **Elma** olan kayıtlar Görsel 8.35'te görüldüğü gibi listelenir.

```
SELECT * FROM urunler WHERE urunAdi='Elma'
```

100 %

Results | Messages

	kimlik	urunAdi	kategoriKimlik
1	1	Elma	1

Görsel 8.35: WHERE komutunun kullanılması

WHERE ile kullanılan operatörler Tablo 8.9'da verilmiştir.

Tablo 8.9: SQL Sorgularında WHERE Komutuyla Kullanılan Operatörler

Operatör	Açıklama
=	Eşittir.
<>	Eşit değil. Bazı versiyonlarda "!=" kullanılabilir.
>	Büyüktür.
<	Küçüktür.
>=	Büyük ve eşittir.
<=	Küçük ve eşittir.
BETWEEN	Arasındadır.
LIKE	Örüntü aramada kullanılır.
IN	Bir sütun için birden çok olası değeri belirtmede kullanılır.

AND ve OR

AND operatörü 1. koşul ve 2. koşulun doğru olması durumunda çalışır. OR operatörü ise 1. koşul veya 2. koşulun doğru olması durumunda çalışır.

Örnek Veri tabanındaki **urunler** tablosunda **kategoriKimlik** sütunu **1** ve **urunAdi** sütunu **Karpuz** olan veriler Görsel 8.36'da görüldüğü gibi listelenir.

```
SELECT * FROM urunler
WHERE kategoriKimlik='1'
AND urunAdi='Karpuz'
```

100 %

Results | Messages

	kimlik	urunAdi	kategoriKimlik
1	2	Karpuz	1

Görsel 8.36: AND komutunun kullanılması

Örnek Veri tabanındaki **urunler** tablosunda **kategoriKimlik** sütunu **1** veya **3** olan veriler Görsel 8.37'de görüldüğü gibi listelenir.

```
SELECT * FROM urunler
WHERE kategoriKimlik='1'
OR kategoriKimlik='3'
```

100 %

Results | Messages

	kimlik	urunAdi	kategoriKimlik
1	1	Armut	1
2	2	Karpuz	1
3	5	Karabiber	3

Görsel 8.37: OR komutunun kullanılması

ORDER BY

ORDER BY, varsayılan olarak artan düzende kayıtları sıralar. Azalan kayıtları sıralamak için DESC anahtar sözcüğü kullanılabilir.

Örnek Veri tabanındaki **urunler** tablosunda bulunan kayıtlar, **urunAdi** sütununa göre azalan düzende Görsel 8.38'de görüldüğü gibi sıralanır.

```
SELECT * FROM urunler
ORDER BY urunAdi DESC
```

100 %

Results | Messages

	kimlik	urunAdi	kategoriKimlik
1	4	Kıvırcık	2
2	2	Karpuz	1
3	5	Karabiber	3
4	3	Domates	2
5	1	Amut	1

Görsel 8.38: DESC komutunun kullanılması

INSERT INTO

Veri tabanındaki tablolara kayıt eklemek için kullanılır.

Örnek Veri tabanındaki **urunler** tablosuna Görsel 8.39'da görüldüğü gibi bir kayıt eklenir.

```
INSERT INTO urunler(urunAdi, kategoriKimlik)
VALUES ('Kekik',3)

SELECT * FROM urunler
```

100 %

Results | Messages

	kimlik	urunAdi	kategoriKimlik
1	1	Amut	1
2	2	Karpuz	1
3	3	Domates	2
4	4	Kıvırcık	2
5	5	Karabiber	3
6	9	Kekik	3

Görsel 8.39: INSERT INTO komutunun kullanılması

UPDATE

Veri tabanındaki tabloların kayıtları üzerinde değişiklik ve güncelleme yapmak için kullanılır.

Örnek

Veri tabanında **urunler** tablosundaki **urunAdi Elma** olan alan Görsel 8.40'ta görüldüğü gibi **Armut** yapılır.

```
UPDATE urunler
SET urunAdi='Armut'
WHERE urunAdi='Elma'
```

100 %

Messages

```
(0 rows affected)

Completion time: 2023-10-23T12:40:46.6680466+03:00
```

Görsel 8.40: UPDATE komutunun kullanılması

DELETE

Veri tabanındaki tablolardan kayıt silmek için kullanılır.

Örnek

Veri tabanında **urunler** tablosundaki **urunAdi Domates** olan alan Görsel 8.41'de görüldüğü gibi silinir.

```
DELETE FROM urunler
WHERE urunadi='Domates'

SELECT * FROM urunler
```

100 %

Results | Messages

	kimlik	urunAdi	kategoriKimlik
1	1	Armut	1
2	2	Karpuz	1
3	4	Kıvırcık	2
4	5	Karabiber	3
5	9	Kekik	3

Görsel 8.41: DELETE komutunun kullanılması

5. Uygulama

İşlem adımlarına göre veri tabanındaki **ogrenciler** tablosunda SQL sorgu komutlarını yazınız.

ogrenci_no	ogrenci_ad	ogrenci_soyad	sinif	dogum_tarihi	dogum_yeri
1	Ayşe	S.	12	2000-05-12	Ankara
2	Emel	G.	11	2000-05-30	İzmir
3	Emre	T.	11	0200-05-08	Ankara
4	Gökhan	T.	12	2000-05-09	İstanbul
5	Eda	K.	11	0200-08-07	Çankırı

1. Adım: Veri tabanındaki ogrenciler tablosunda 12. sınıf öğrencilerinin listesini sorgulayan SQL komutunu çalıştırınız (Görsel 8.42).

```
SELECT * FROM ogrenci WHERE sinif=12
```

100 %

Results | Messages

	ogrenci_no	ogrenci_ad	ogrenci_soyad	sinif	dogum_tarihi	dogum_yeri
1	1	Ayşe	S.	12	2000-05-12	Ankara
2	4	Gökhan	T.	12	2000-05-09	İstanbul

Görsel 8.42: On ikinci sınıf öğrenci listesinin SQL komutlarıyla sorgulanması

2. Adım: Veri tabanındaki **ogrenciler** tablosunda doğum yeri Ankara olan öğrencileri listeleyen SQL komutunu çalıştırınız (Görsel 8.43).

```
SELECT * FROM ogrenci WHERE dogum_yeri='Ankara'
```

100 %

Results | Messages

	ogrenci_no	ogrenci_ad	ogrenci_soyad	sinif	dogum_tarihi	dogum_yeri
1	1	Ayşe	S.	12	2000-05-12	Ankara
2	3	Emre	T.	11	0200-05-08	Ankara

Görsel 8.43: Doğum yeri Ankara olan öğrencilerin SQL komutlarıyla sorgulanması

8.4.2. NoSQL Uygulamaları

Günümüzde internet kullanımının artması ve sosyal medyanın hayata girmesi, insanların sanal dünya için oluşturduğu verinin büyümesine yol açmıştır. NoSQL, internete yüklenen ve gün geçtikçe artan bu veriyi depolayabilmek, yüksek sayıdaki kullanıcıların anlık taleplerine cevap vermek amacıyla ortaya çıkmıştır. **NoSQL**, yatay olarak ölçeklendirilebilen veri tabanı sistemlerinin genel adıdır. NoSQL içinde büyük verinin analizinin ve sorgulamalarının yapılması için gerekli elemanlar bulunur.

NoSQL veri tabanı, sabit bir şema gerektirmeyen ve ilişkisel olmayan bir DBMS'dir. Big Data ve gerçek zamanlı web uygulamalarında NoSQL'den yararlanılır. NoSQL'de sorgu yazmak için SQL kullanılmaz. NoSQL'e en iyi örnek MongoDB'dir. MongoDB, yüksek hacimli veri depolama için kullanılan belge odaklı bir NoSQL veri tabanıdır. Belgeler, MongoDB'deki temel veri birimi olan anahtar-değer çiftlerinden oluşur. Koleksiyonlar, ilişkisel veri tabanı tablolarının eş değeri olan belge ve işlev kümelerini içerir.

6. Uygulama

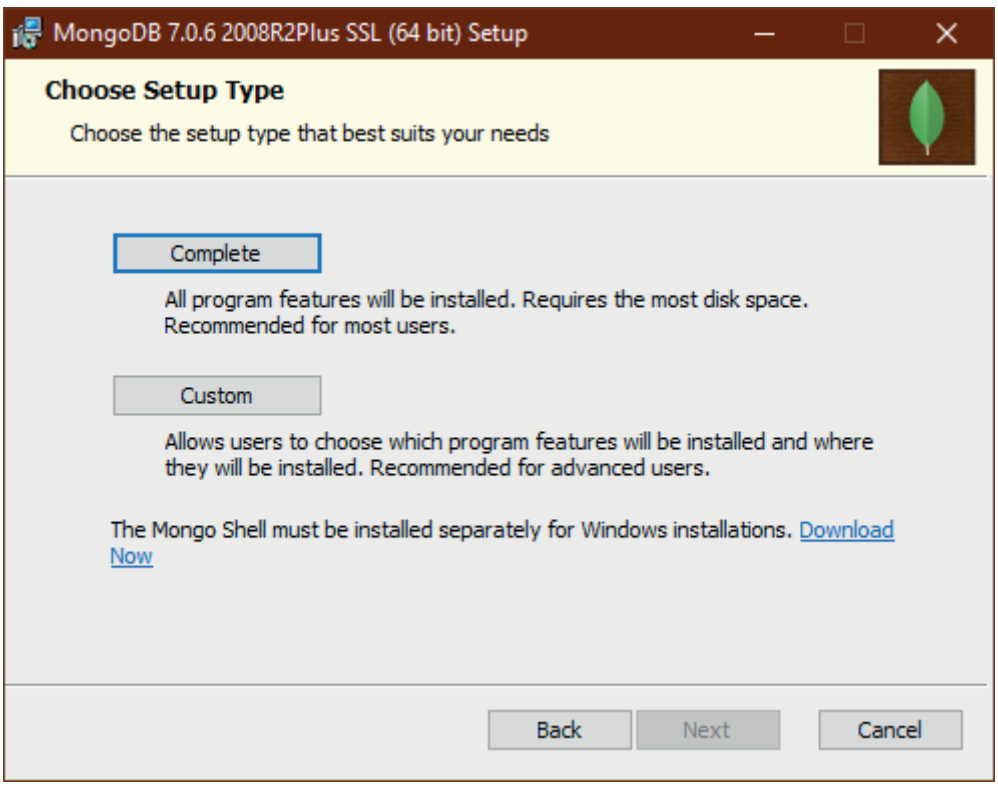
İşlem adımlarına göre veri tabanı oluşturmak, oluşturulan veri tabanını silmek ve veri tabanına veri girmek için NoSQL veri tabanı yönetim sisteminin kurulumunu yapınız.

1. Adım: İnternetten **MongoDB Community Server** kurulum dosyasını indirerek yüklemeyi başlatınız.

2. Adım: Tüm bileşenleri yüklemek için **Tam kurulum (Complete)** düğmesine tıklayınız (Görsel 8.44).

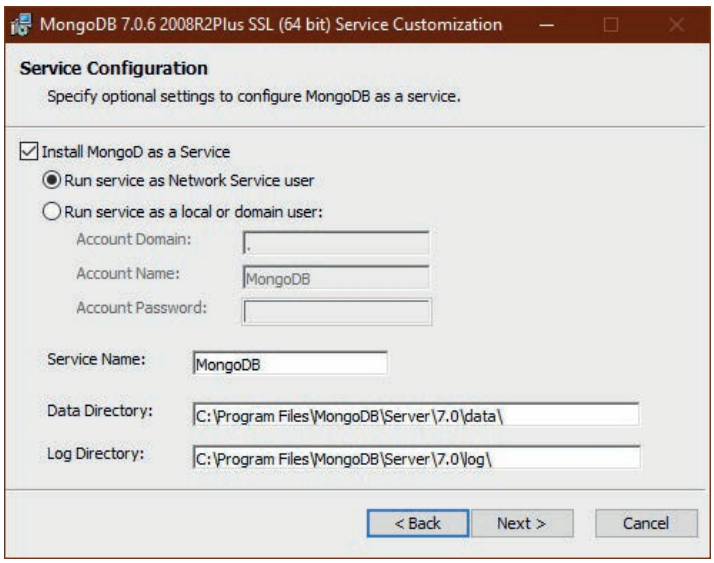
Not

Özel seçenek (Custom), belirli bileşenleri kurmak veya kurulumun konumunu değiştirmek için kullanılabilir.



Görsel 8.44: Kurulum şeklinin belirlenmesi

3. Adım: **Hizmeti şebeke hizmeti kullanıcısı olarak çalıştır (Run service as Network Service user)** seçeneğini seçiniz ve veri dizinini not ederek **Next>** düğmesine tıklayınız (Görsel 8.45).



Görsel 8.45: Kurulum konumunun belirlenmesi

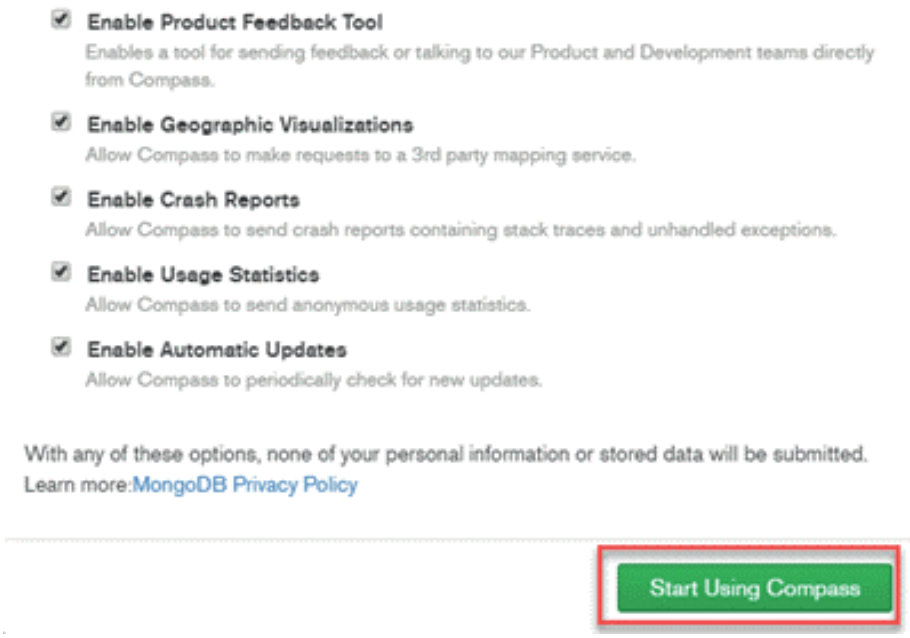
4. Adım: Kurulumu tamamladıktan sonra **Bitir (Finish)** düğmesine tıklayınız.

7. Uygulama

İşlem adımlarına göre veri tabanı yönetim işlemlerini gerçekleştirmek için NoSQL veri tabanı yönetim aracını yükleyiniz.

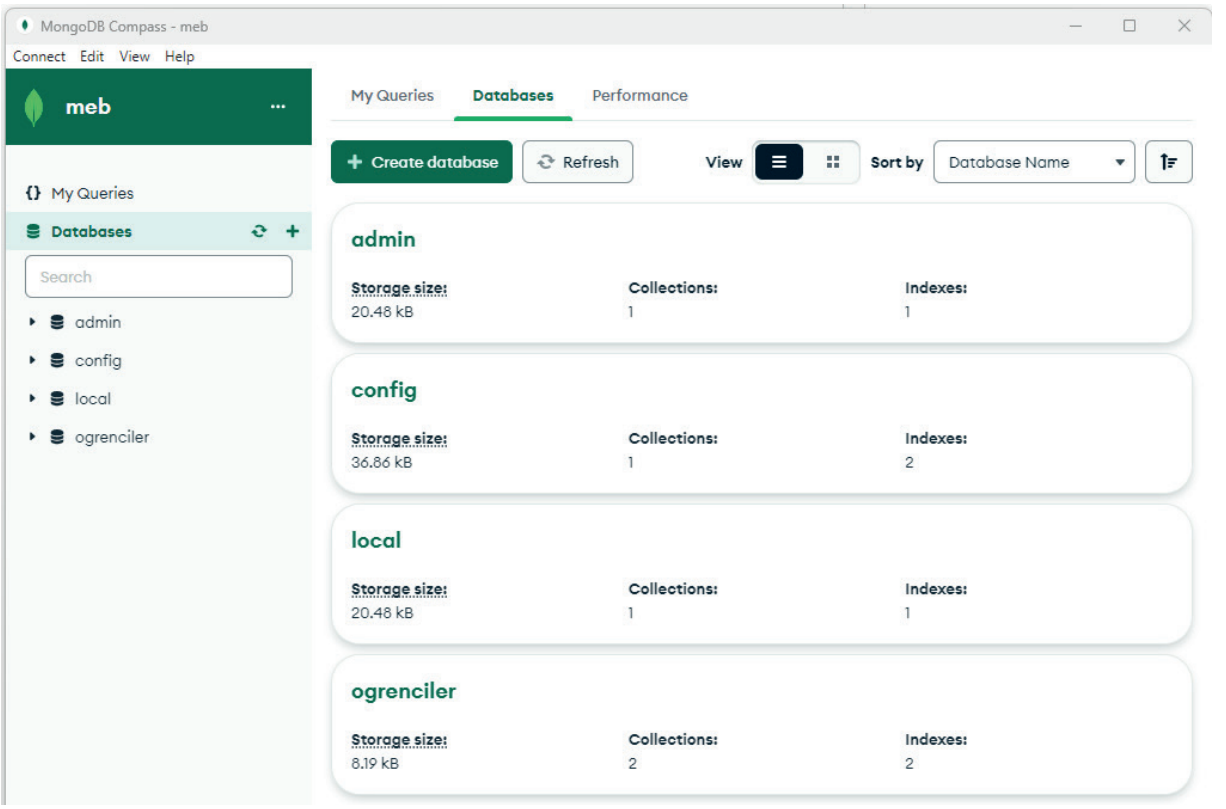
1. Adım: **MongoDB Compass** için kurulum dosyasını internetten indirerek kurulumu başlatınız.

2. Adım: Gizlilik ayarlarını varsayılan olarak bırakarak **Start Using Compass** düğmesine tıklayınız (Görsel 8.46).



Görsel 8.46: MongoDB Compass aracının indirilmesi

3. Adım: Mevcut veri tabanlarının listesini içeren ana ekranı görüntüleyiniz (Görsel 8.47).



Görsel 8.47: MongoDB ana ekranının görüntülenmesi

8.4.3. NoSQL Veri Tabanı ve Koleksiyon Oluşturma

NoSQL veri tabanında temel adım, bir veri tabanına ve koleksiyona sahip olmaktır. Veri tabanı, tüm koleksiyonları depolamak için kullanılır. Koleksiyon ise tüm belgeleri depolamak için kullanılır. Belgeler de ilgili alan adları ve değerlerini içerir.

```
{     “ogrenciNo” : 1,
      “ogrenciAdi” : “Ali”
}
```

Alan adları **Alan değerleri**

Belgenin alan adları **ogrenciNo** ve **ogrenciAdi**, alan değerleri ise sırasıyla **1** ve **Ali** girilmiştir. Bu şekilde hazırlanan bir grup belge, NoSQL veri tabanında bir koleksiyon oluşturur.

8. Uygulama

İşlem adımlarına göre NoSQL veri tabanı yönetim sistemi yapılandırması için veri tabanı ve koleksiyon oluşturma, veri tabanındaki belgeleri gösterme ve silme, bilgileri listeleme işlemlerini gerçekleştiriniz.

1. Adım: **data.csv** adında bir **CSV** dosyası oluşturup verileri içine kaydediniz.

ogrenciNo	ogrenciAdi
1	Eda
2	Ayse
3	Oya

2. Adım: Verilerin MongoDB'ye aktarılması için **mongoimport** komutunu kullanınız (Görsel 8.48).

Not İlk satır, koleksiyonda başlık satırı olarak adlandırılır. Bu nedenle **--headerline** seçeneği belirtilir. Daha sonra **data.csv** dosyası yazılır.

```
C:\Program Files\MongoDB\Server\7.0\bin>mongoimport --db ogrenciler --type csv --headerline --file ogrenci.csv
2023-12-06T09:59:35.908+0300    no collection specified
2023-12-06T09:59:35.935+0300    using filename 'ogrenci' as collection
2023-12-06T09:59:36.575+0300    connected to: mongodb://localhost/
2023-12-06T09:59:36.627+0300    3 document(s) imported successfully. 0 document(s) failed to import.
```

Görsel 8.48: MongoDB içe aktarma komutunun kullanılması

3. Adım: **use** komutunu kullanarak veri tabanını oluşturunuz (Görsel 8.49).

```
> use ogrenciler
< switched to db ogrenciler
>
ogrenciler>
```

Görsel 8.49: MongoDB use komutunun kullanılması

4. Adım: **insert()** komutunu kullanarak koleksiyon oluşturunuz (Görsel 8.50).

```
> db.ogrenci.insertOne({"ogrenciNo":4,"ogrenciAdi":"Oya"});
< {
    acknowledged: true,
    insertedId: ObjectId("657023aa8b5a7b31f367e747")
  }
ogrenciler>
```

Görsel 8.50: MongoDB insert komutunun kullanılması

5. Adım: Koleksiyonda bulunan tüm belgeleri gösteriniz (Görsel 8.51).

```
> db.ogrenci.find().forEach(printjson);
< {
    _id: ObjectId("65701bd852eee5874e1e1852"),
    'ogrenciNo;ogrenciAdi': '2;Emir'
  }
< {
    _id: ObjectId("65701bd852eee5874e1e1853"),
    'ogrenciNo;ogrenciAdi': '1;Ali'
  }
ogrenciler>
```

Görsel 8.51: MongoDB'deki tüm belgelerin gösterilmesi

6. Adım: ogrenciNo'su 1'den büyük olan tüm çalışanları listeleyiniz (Görsel 8.52).

```
> db.ogrenci.find({ogrenciNo:{$gt:1}}).forEach(printjson);
< {
    _id: ObjectId("657025338b5a7b31f367e748"),
    ogrenciNo: 3,
    ogrenciAdi: 'Oya'
  }
ogrenciler>
```

Görsel 8.52: MongoDB'de kimliği ikiden büyük tüm öğrencilerin listelenmesi

7. Adım: Koleksiyondaki tüm belgeleri döndüren sıralama işlevini azalan olarak listeleyiniz (Görsel 8.53).

```
> db.ogrenci.find().sort({ogrenciNo:-1}).forEach(printjson);
< {
    _id: ObjectId("657025338b5a7b31f367e748"),
    ogrenciNo: 3,
    ogrenciAdi: 'Oya'
  }
< {
    _id: ObjectId("65701bd852eee5874e1e1852"),
    'ogrenciNo;ogrenciAdi': '2;Emir'
  }
< {
    _id: ObjectId("65701bd852eee5874e1e1853"),
    'ogrenciNo;ogrenciAdi': '1;Ali'
  }
ogrenciler>
```

Görsel 8.53: MongoDB'de tüm belgelerin azalan sırada listelenmesi

8. Adım: ogrenciNo'su 3 olan belgeleri kaldırınız (Görsel 8.54).

```
> db.ogrenci.remove({ogrenciNo:3})
< {
    acknowledged: true,
    deletedCount: 1
  }
ogrenciler>
```

Görsel 8.54: MongoDB'den belge silinmesi

9. Adım: **ogrenciNo'su** 1 olan belgenin ogrenciAdi değerini değiştiriniz (Görsel 8.55).

```
> db.ogrenci.update({"ogrenciNo":3},{$set:{"ogrenciAdi":"Hatice"}});
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

Görsel 8.55: MongoDB'de belge güncellenmesi

Sıra Sizde

NoSQL veri tabanı yönetim sistemi üzerinde veri tabanı oluşturarak ve takım tablosuna ad, soyad, doğum tarihi bilgilerini girerek en küçük oyuncu ismini sorgulayınız.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. Veri tabanını oluşturdu.		
2. Takım tablosunu oluşturdu.		
3. Tablo alanlarına veri girişini yaptı.		
4. Tablo üzerinde sorgulama işlemini gerçekleştirdi.		

"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.

8.5. VERİ TABANI GÜVENLİĞİ

Veri tabanı güvenliği, veri tabanını iç ve dış tehditlerden korumak için alınan çeşitli önlemleri ifade eder. Veri tabanı güvenliği; veri tabanının kendini, içerdiği verileri, veri tabanı yönetim sistemini ve ona erişen uygulamaları korumayı içerir. Veri tabanları, siber güvenlik tehditleri ve veri tabanına erişebilen kişiler tarafından kötüye kullanılması gibi kasıtlı saldırılara karşı güvence altına alınmalıdır.

Veri tabanı güvenliği, yalnızca kimliği doğrulanmış kullanıcıların yetkili etkinlikleri gerekli zamanlarda gerçekleştirmesini sağlamaya çalışır. Veri tabanı güvenliği; fiziksel güvenlik, ağ güvenliği, şifreleme ve kimlik doğrulamanın yanı sıra çok çeşitli güvenlik konularını içerir. Veri tabanı güvenliği; verilerin yetkisiz girişimlere karşı gizliliği veya korunması, bütünlük veya yetkisiz veri erişiminin önlenmesi ve donanım ile yazılım hatalarının veya veri kullanılabilirliğinin reddine yol açan kötü niyetli etkinliklerden kurtarılması olmak üzere üç yapıyı kapsar (Görsel 8.56).

Görsel 8.56: Veri tabanı güvenliği

8.5.1. Veri Tabanı Yönetim Sistemlerinin Yapılandırmaları

Veri tabanı yönetim sistemleri için en yaygın yapılandırmalar şunlardır:

- **Veri Tabanı Kullanıcıları:** Yalnızca belirli kullanıcıların erişimine izin verilerek veriler güvence altına alınabilir.
- **Veri Tabanı Rolleri:** Kullanıcıların izinlerini gruplandırmak için kullanılabilir.
- **Veri Tabanı Kısıtlamaları:** Verilerin doğruluğunu ve tutarlılığını korumak için kullanılabilir.
- **Veri Tabanı İndeksleri:** Veri tabanlarında arama ve sıralama performansını iyileştirmek için kullanılabilir.
- **Veri Tabanı Yedekleme ve Kurtarma:** Bir arıza durumunda veri tabanı geri yüklenebilir, geri yükleme noktaları oluşturulabilir ve geri yükleme noktalarının durumu görüntülenebilir.

8.5.2. Veri Tabanı Güvenliği Temel İlkeleri

Veri tabanı güvenliği, hassas verilerin kaybolmasını veya izinsiz olarak açığa çıkmasını önlemeyi amaçlar. Elektronik olarak toplanan, saklanan ve paylaşılan veri miktarı arttıkça veri tabanı güvenliğine ihtiyaç da artar. Veri tabanı güvenliği; veri tabanının içeriğine kontrollü, korumalı erişim sağlanması ve aynı zamanda bütünlüğünün, tutarlılığının ve genel yapısının korunması gerektiğini belirtir.

Fiziksel Veri Tabanı Güvenliği

Verilerin saklandığı, korunduğu fiziksel donanımı güvende tutmak çok önemlidir. Fiziksel güvenlik ister kurum içinde ister bulut üzerinden erişilsin, veri tabanı sunucusunun bulunduğu odanın kilitlenmesini içerir. Ayrıca güvenlik ekiplerinin bu donanıma fiziksel erişiminin denetlenmesi gerekir.

Fiziksel bir felaket durumunda veri tabanı yedekleme ve felaket kurtarma önlemlerinin alınması gereklidir. Web sunucularını ve uygulamalarını, kurumun güvenliğini sağlamak istediği veri tabanıyla aynı sunucuda barındırmamak da önemlidir. Bir diğer husus da verilerin şifrelenmesidir. Böylece bir sistemin fiziksel depolaması çalınırsa veya güvenliği ihlal edilirse verilerin güvenliği sağlanır.

Web Uygulamaları ve Güvenlik Duvarları

Web uygulamaları ve güvenlik duvarları, veri tabanı güvenliği için en iyi uygulamalardır. Bir veri tabanı güvenlik duvarı, yetkisiz kişilerin ağa erişmesini engeller. Güvenlik duvarları, hassas bilgileri saldırılardan korumak için çok önemli bir ön koşuldur. Veri tabanlarıyla etkileşime giren web uygulamaları, uygulama erişim yönetimi yazılımı tarafından korunabilir. Veri tabanı güvenlik önlemi, erişim kontrol listelerine benzer ve web uygulamalarına kimlerin, nasıl erişebileceğini belirler.

Veri Tabanı Şifreleme

Veri şifreleme, verilerin veri tabanında olduğu yerde uygulandığı için en etkili veri tabanı güvenlik uygulamalarından biridir. Şifrelenmiş veriler, uygun anahtarlarla şifresi çözülmediği sürece anlamsız görünecek şekilde dönüştürülür. Bu nedenle yetkisiz kullanıcılar, şifrelenmiş verilere erişse de bu veriler onlar için anlamsız olacaktır.

Parolaların ve İzinlerin Yönetimi

Parolaları ve izinleri yönetmek, veri tabanı güvenliğini sağlamak için kritik öneme sahiptir. Bu görev genellikle güvenlik çalışanları tarafından denetlenir. Parolaları yönetmek için çift veya çok faktörlü kimlik doğrulama önlemleri ile kullanıcılara kimlik bilgilerini girmeleri için sınırlı bir süre vermek gibi birçok farklı uygulama yapılabilir. Bu uygulama, erişim ve izin listelerinin sürekli olarak güncellenmesini gerektirir.

Hassas Veri Tabanlarını İzole Etme

Hassas veri tabanları izole edilirse veri tabanı güvenliğini aşmak çok zordur. İzolasyon tekniklerinin nasıl uygulandığına bağlı olarak yetkisiz kullanıcılar, hassas bilgilerin varlığından da haberdar olmayabilir.

Değişim Yönetimi

Değişim yönetimi, değişiklik sırasında veri tabanlarını korumak için hangi prosedürlerin uygulanması gerektiğini ana hatlarıyla belirtmeyi gerektirir. Değişiklik örnekleri arasında birleşmeler, satın almalar veya yalnızca farklı kullanıcıların çeşitli bilgisayar kaynaklarına erişim kazanması yer alır.

Veri Tabanı Denetimi

Veri tabanı denetimi genellikle veri tabanları ve uygulamaları için günlük dosyalarının düzenli olarak okunmasını gerektirir. Bu bilgiler; kimin, hangi uygulamaya ne zaman eriştiğini ve orada ne yaptığını ortaya çıkarır. Verilere yetkisiz erişim varsa zamanında yapılan denetimler, veri tabanı yöneticilerini uyararak ihlallerin genel etkisini azaltmaya yardımcı olabilir.

8.5.3. Veri Tabanı Sistemindeki Tehdit Türleri

Veri tabanı sistemlerinin güvenliğini etkileyen çeşitli tehditler vardır. Bu tehditler; veri gizliliğini, bütünlüğünü ve erişilebilirliğini tehlikeye atar. Veri tabanı sistemindeki tehdit türleri şunlardır:

Şifre Kırma

Şifre kırma, veri tabanı sistemi testi yapılırken en önemli kontroldür. Bilgisayar korsanları, kritik bilgilere erişmek için bir şifre kırma aracı kullanabilir, ortak bir kullanıcı adı veya şifreyi tahmin edebilir. Bu ortak şifreler, internette kolayca bulunabildiği gibi şifre kırma araçları da mevcuttur. Bu nedenle test sırasında şifre politikasının sistemde korunup korunmadığını kontrol etmek gerekir. Herhangi bir bankacılık ve finans uygulamasında, tüm kritik bilgi veri tabanı sistemlerinde sıkı bir şifre politikasının belirlenmesine ihtiyaç vardır.

Kaba kuvvet saldırısı, olası tüm harf kombinasyonlarının denenerek veya en olası parolaları içeren bir sözlük kullanılarak SQL oturum açma parolalarını kırmanın bir yoludur. Bu nedenle şifrelerin test edilmesi oldukça önemlidir.

Araştırma

Şifre kırma için kullanılan araçları araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

9. Uygulama

İşlem adımlarına göre veri tabanı giriş bilgilerini elde etmek için SQL kimlik bilgilerine yönelik kaba kuvvet saldırısını engelleme uygulamasını yapınız.

1. Adım: Erişim politikaları tarafından zorunlu tutulmayan SQL şifrelerini tanımlayınız.

```
SELECT name, is_disabled
FROM sys.sql_logins
WHERE is_policy_checked = 0
ORDER BY name;
```

2. Adım: Bir şifre politikası uygulamak için önceden oluşturulmuş Ali kullanıcısının oturum açma bilgilerini değiştiriniz.

```
ALTER LOGIN Ali WITH CHECK_POLICY = ON,
CHECK_EXPIRATION = ON;
```

Not Bu kod, şifre politikasını ve son kullanma tarihini zorunlu kılmak için Ali oturum açma bilgisini değiştirir.

3. Adım: Parola süresinin ne zaman dolacağını görüntüleyiniz.

```
SELECT LOGINPROPERTY('Ali', 'DaysUntilExpiration');
```

4. Adım: Oturum açmada MUST_CHANGE seçeneğiyle kullanıcının şifresini değiştirmesini sağlayınız.

```
ALTER LOGIN Ali WITH PASSWORD = 'sifreyi değiştirmeniz gerekiyor !' MUST_CHANGE,
CHECK_POLICY = ON, CHECK_EXPIRATION = ON;
```

5. Adım: Gerekli tüm oturum açma bilgilerini görüntüleyiniz.

```
SELECT 'ALTER LOGIN ' + QUOTENAME(name) + ' WITH PASSWORD = ''Parolanızı değiş-
tirmeniz gerekiyor.'' MUST_CHANGE, CHECK_POLICY = ON, CHECK_ EXPIRATION = ON; '
FROM sys.sql_logins WHERE is_policy_checked = 0 ORDER BY name;
```

Parolaları kaba kuvvet saldırılarına karşı korumanın en iyi yolu, Windows parola ilkelerini ve geçerlilik bitişini zorunlu kılmaktır. Kaba kuvvet saldırı araçlarıyla şifrelerin düzenli olarak test edilmesi önerilir. Kaba kuvvet saldırıları, SQL Server hata günlüğünde ve Windows olay günlüğünde izler bırakır.

SQL oturum açma parolaları veya şifreleme anahtarları herhangi bir sistem tablosunda saklanmaz, parolanın karma değeri tutulur. Bu durum, şifrenin çözülerek geri alınmasının hiçbir yolu bulunmadığı anlamına gelir.

Şifrelerin test edilmesi için bir kelime listesi tablosu kullanılarak SQL Server içinde hızlı bir sözlük saldırısı gerçekleştirilebilir.

Araştırma

Burp Suite ve Damn Vulnerable Web uygulamasını araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

10. Uygulama

İşlem adımlarına göre veri tabanı giriş bilgilerini elde etmek için kaba kuvvet saldırısı yapınız.

1. Adım: Güvenlik testleri için kullandığınız Damn Vulnerable Web uygulamasını açarak, **Güvenlik** ayarını **Orta** olarak belirleyiniz.

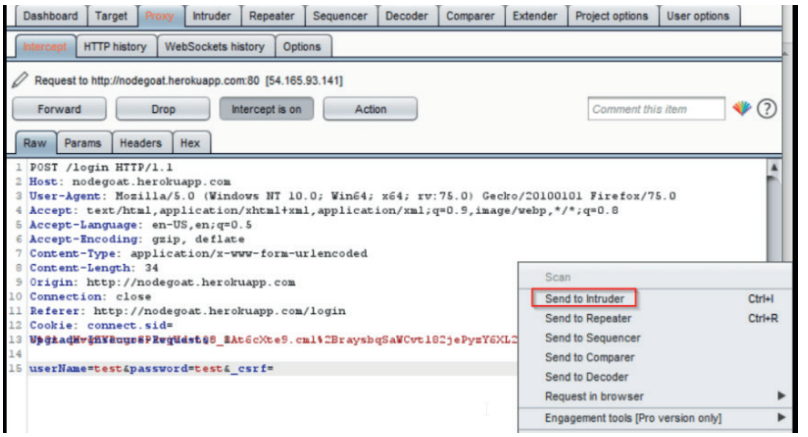
2. Adım: Damn Vulnerable Web uygulamasınında **Brute Force** seçeneğini tıklayarak kaba kuvvet saldırısı yapmak için bir şifre giriniz.

3. Adım: Burp Suite uygulamasını açarak **Proxy** sekmesinden **Intercept** sekmesini açınız.

4. Adım: **Intercept** sekmesinde **Intercept is on** seçeneğini tıklayınız.

5. Adım: Ele geçirilen istek için sağ tıklayarak **Send to Intruder** seçeneğini seçiniz (Görsel 8.57).

.



Görsel 8.57: Burp suite Intercept ekranı

6. Adım: **Intruder** sekmesinde ayarlanmış ana bilgisayar ve bağlantı noktasının bulunduğu **Positions** alt sekmesine geçiniz (Görsel 8.58).

Görsel 8.58: Burp suite positions ekranı

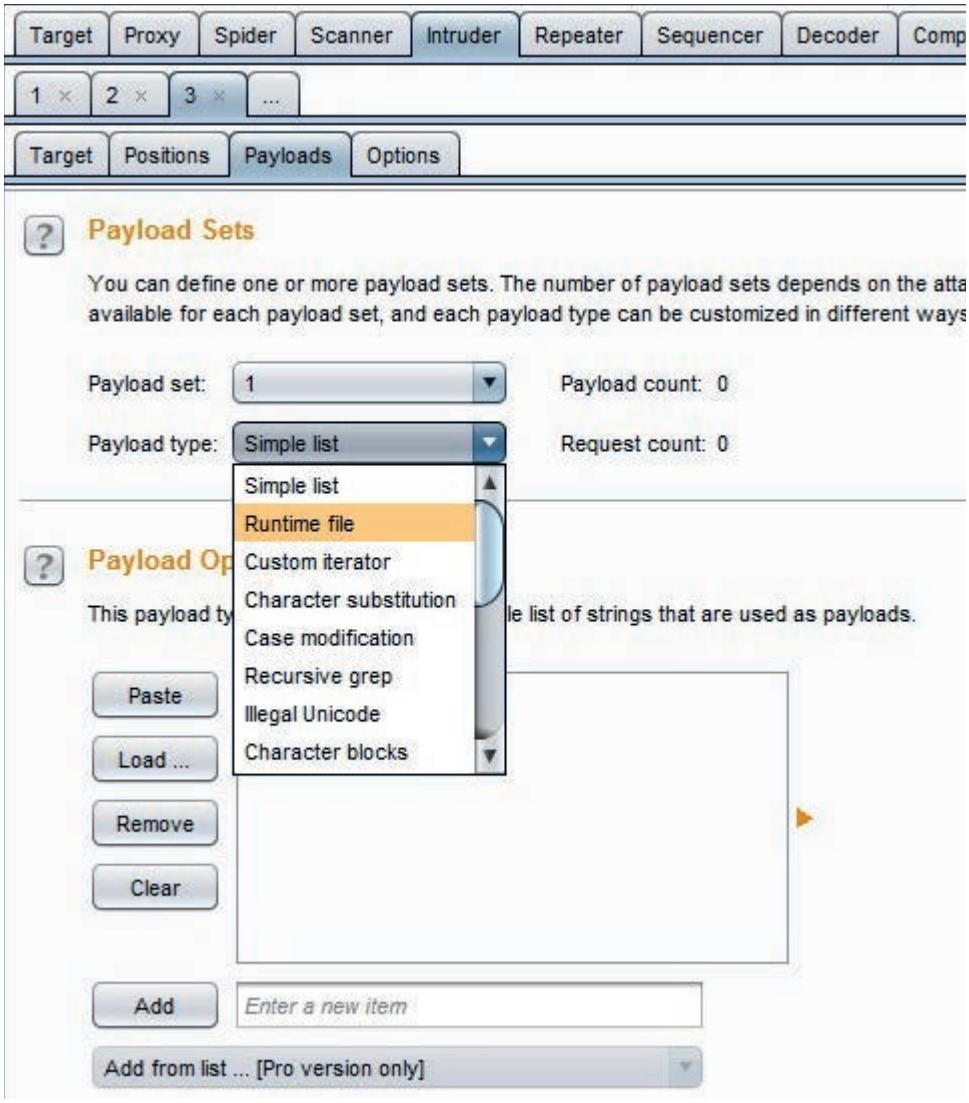
7. Adım: **Positions** sekmesinde önce **Clear** butonu ile **username** ve **password** alanlarını silerek **Add** butonuna tıklayınız.

8. Adım: Saldırı tipini **Cluster Bomp** olarak seçiniz.

Araştırma

Burp Suite uygulamasındaki saldırı türlerini araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

9. Adım: **Payloads** sekmesinde işaretlenen **username** ve **password** alanları için **Runtime File** seçiniz (Görsel 8.59).



Görsel 8.59: Burp suite payloads ekranı

10. Adım: **username** ve **password** değerlerini içeren dosyaları yükleyiniz.

11. Adım: Kaba kuvvet saldırısının hangi **username** ve **password** değerlerinde başarılı olduğunu görmek için **Options** sekmesindeki **Grep-Match** ayarını yapınız.

12. Adım: **Intruder** menüsünden **Start attack** seçeneğini seçerek kaba kuvvet saldırısını başlatınız.

Araştırma

Veri tabanı şifrelerinin gücünü denetleyen şifre kırıcı araçları araştırınız. Edindiğiniz bilgileri sınıfta arkadaşlarınızla paylaşınız.

Veri Tabanında Ayrıcalık Yükselişi

Kullanıcı, veri tabanı sisteminde erişime sahiptir ve bazı yetkisiz faaliyetler gerçekleştirebilmek için yalnızca bu erişimi daha üst seviyeye çıkarmaya çalışır.

Hizmet Reddi

Saldırgan, bir veri tabanı sistemini veya uygulama kaynağını meşru kullanıcıların kullanımına kapalı hâle getirir.

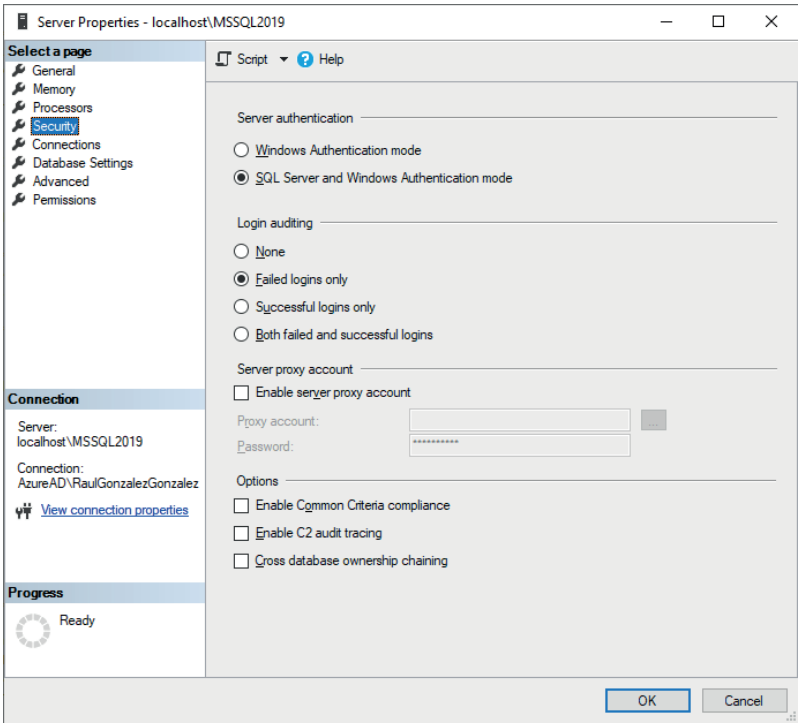
Verilere Yetkisiz Erişim

Başka bir saldırı türü, bir uygulama veya veri tabanı sistemi içindeki verilere yetkisiz erişim sağlamaktır.

11. Uygulama

İşlem adımlarına göre SQL Server'da kaba kuvvet saldırılarına önlem alınız (Görsel 8.60).

1. Adım: SQL Server kurulumu sırasında veya Görsel 8.60'ta görüldüğü gibi **Server Properties (Sunucu Özellikleri)** iletişim kutusu altındaki **Security (Güvenlik)** sayfasında **Server authentiarion (Sunucu kimlik doğrulama)** yöntemini değiştirerek etkinleştiriniz.



Görsel 8.60: Sunucu özellikleri

2. Adım: [sa]'yı devre dışı bırakınız (Görsel 8.61).

```
USE [master]
GO
ALTER LOGIN [sa] DISABLE
GO
SELECT principal_id,name, sid, type, type_desc, is_disabled,default_database_name, credential_id,
owning_principal_id, is_fixed_role
FROM sys.server_principals
WHERE sid = 0x01
GO
```

100 %

Results | Messages

	principal_id	name	sid	type	type_desc	is_disabled	default_database_name	credential_id	owning_principal_id	is_fixed_role
1	1	sa	0x01	S	SQL_LOGIN	1	master	NULL	NULL	0

Görsel 8.61: [sa]'nın devre dışı bırakılması

3. Adım: [sa]'yı yeniden adlandırınız (Görsel 8.62).

```
USE [master]
GO
ALTER LOGIN [sa] WITH NAME = [ww]
SELECT principal_id
, name, sid, type_desc, is_disabled
, default_database_name, credential_id
, owning_principal_id
FROM sys.server_principals
WHERE sid = 0x01
GO
```

100 %

Results | Messages

	principal_id	name	sid	type_desc	is_disabled	default_database_name	credential_id	owning_principal_id
1	1	ww	0x01	SQL_LOGIN	1	master	NULL	NULL

Görsel 8.62: [sa]'nın yeniden adlandırılması

Kimlik Sahtekârlığı

Kimlik sahtekârlığında bir bilgisayar korsanı; ağ ana bilgisayarlarına saldırılar başlatmak, verileri çalmak veya veri tabanı sistemine erişim kontrollerini atlamak için bir kullanıcının veya cihazın kimlik bilgilerini kullanır.

Veri Manipülasyonu

Veri manipülasyonu saldırısında bilgisayar korsanı bir avantaj elde etmek veya veri tabanı sahiplerinin imajına zarar vermek için verileri değiştirir.

Veri Tabanı Giriş Denetimi

Veri tabanı girişlerini denetlemek için yaygın olarak kullanılan en iyi uygulamalardan biri, ana programdaki **syslogins** tablosunu kullanmaktır. Bu tablo; kullanıcının oturum açma adını, kimliğini, hesap türünü ve varsayılan veri tabanını içerir.

SQL Enjeksiyon Saldırıları

SQL enjeksiyonu, veri tabanı sistemine kötü amaçlı SQL ifadelerinin eklenmesi ve veri tabanı sisteminden kritik bilgilerin çalınması için yapılan bir saldırı türüdür. Bu saldırıyı önlemek için kullanıcı giriş alanlarının şu şekilde düzenlenmesi gerekir:

- SQL için dinamik sorgular kullanılmamalıdır.
- Kötü amaçlı değerleri tespit etmek için sistem girişleri doğrulanmalıdır.
- NoSQL veri tabanları için giriş türleri, bilinen türlere göre de doğrulanmalıdır.
- Uygulama hesaplarına yönetici tipi erişim hakkı atanmamalıdır.
- İşletim sistemi hesabının ayrıcalıkları en aza indirilmelidir.

SQL enjeksiyon, uygulama alanlarındaki kullanıcı girişlerinin kontrol edilmesini içerir. Bir web uygulamasındaki herhangi bir metin kutusuna "," veya ";" gibi özel bir karakterin girilmesine izin verilmemelidir. Bu durum, kullanıcı girişinin bir sorguya eklendiği ve daha sonra web uygulama tarafından yürütüldüğü anlamına gelir. Böyle bir durumda web uygulama, SQL enjeksiyonuna karşı savunmasız kalır. Parantez, virgül ve tırnak işaretleri için SQL enjeksiyon testi yapılmalıdır.

Basit bir kimlik doğrulama örneği şu şekildedir:

```
SELECT * FROM kullanicilar
WHERE kimlik_no = 'kullanıcının kimlik bilgisi'
AND sifre = 'kullanıcının şifresi'
```

SELECT ifadesinin kullanıldığı bu kimlik doğrulama örneğinde, veri tabanında eşleşen bir kimlik ve parola kaydı varsa ilgili tüm kullanıcı verilerini döndürür. Kullanıcı, geçersiz bir kimlik numarası veya şifre girerse sorgu boş bir veri kümesi döndürür. Bir geliştirici, kullanıcının oturum açıp açamayacağını kontrol etmek için bu basit sorguyu kullanabilir. Kötü niyetli bir bilgisayar korsanı, **kullanıcı tarafından belirlenen kimlik numarası** ve **şifre** değerini elde edebilir. Sonuç olarak veri tabanına gönderilen sorgu dizesi şu şekildedir:

```
SELECT * FROM kullanicilar
WHERE kimlik_no = 'admin'--' AND sifre = ''
```

Tek tırnak (') işareti, **kimlik_no** atamasını tamamlar ve çift çizgi (--) işareti ise SQL ifadesinin geri kalan şifre kontrolünün bir yorum olarak değerlendirilmesine neden olur. Bu nedenle uygulama şu sorguyu yürütür:

```
SELECT * FROM kullanicilar
WHERE kimlik_no = 'admin'
```

Sorgu çalıştırıldığında başarılı bir SQL enjeksiyonunu temsil eder. Yönetici için tüm kullanıcı verilerini döndürür ve bilgisayar korsanının yönetici olarak uygulamaya yetkisiz erişimini sağlar.

12. Uygulama

İşlem adımlarına göre basit bir SQL kodu ile uygulamalardan kullanıcı adı ve şifre elde etmek için SQL enjeksiyon testini gerçekleştiriniz.

1. Adım: Örnek SQL kodunu MySQL Online Editör yardımıyla çalıştırınız (Görsel 8.63).

```
1 CREATE TABLE ogrenci (
2 numara INT NOT NULL AUTO_INCREMENT,
3 eposta VARCHAR(45) NULL,
4 sifre VARCHAR(45) NULL,
5 PRIMARY KEY (numara));
6
7 insert into ogrenci (eposta,sifre) values ('meb@edu',md5('123'));
```

Execute Share MySQL

Görsel 8.63: SQL editör ekranı

2. Adım: **SELECT * FROM ogrenci** SQL kodunu çalıştırarak SQL komutunun sonucunu görüntüleyiniz (Görsel 8.64).

```
1 CREATE TABLE ogrenci (
2 numara INT NOT NULL AUTO_INCREMENT,
3 eposta VARCHAR(45) NULL,
4 sifre VARCHAR(45) NULL,
5 PRIMARY KEY (numara));
6
7 insert into ogrenci (eposta,sifre) values ('meb@edu',md5('123'));
```

Execute Share MySQL

Görsel 8.64: SQL komutu sonucu

3. Adım: **SELECT * FROM ogrenci WHERE eposta = 'xxx@xxx.xxx' OR 1=1**; SQL ifadesini çalıştırarak kullanıcıların şifrelerini görüntüleyiniz.

NoSQL Enjeksiyon Saldırıları

NoSQL enjeksiyonları ortak bir sorgu diline dayanmadığı için belirli bir enjeksiyon güvenlik açığı yalnızca belirli bir NoSQL veri tabanı türünü etkiler. Bunun dışında NoSQL enjeksiyon saldırıları, geleneksel SQL enjeksiyon saldırılarına benzer. Saldırgan, kullanıcı girişinde formlar veya HTTP istekleri gibi kötü amaçlı bir veri sağlar ve bu giriş, NoSQL veri tabanına temizlenmeden iletilirse veri tabanının bozulmasına neden olabilir.

Kimlik doğrulamayı gerçekleştirmek için MongoDB veri tabanına erişen basit bir örnek şu şekildedir:

```
$username = $_POST['username'];
$password = $_POST['password'];
$connection = new MongoDB\Client('mongodb://localhost:27017');
if($connection) {
	$db = $connection->test;
	$users = $db->users;
	$query = array(
		"user" => $username,
		"password" => $password
	);
	$req = $users->findOne($query);
}
```

Görüldüğü gibi kimlik doğrulama için kullanılan kullanıcı adı ve şifre parametreleri bir POST isteğinden alınır ve doğrudan sorguya eklenir. Diğer enjeksiyon türlerine benzer şekilde NoSQL enjeksiyon, kötü niyetli bir kullanıcının verilere erişimini sağlar.

Başarılı bir MongoDB enjeksiyonu gerçekleştirmek için bir POST isteğinde şu kötü amaçlı giriş verilerinin sağlanması yeterlidir:

```
kullanıcı adi[$eq]=admin&sifre[$ne]=sifre
```

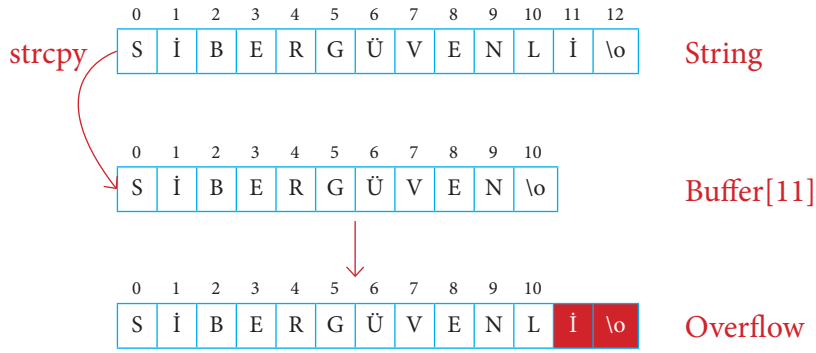
[$ne] sorgu operatörü **eşit değil** anlamına gelir. Dolayısıyla ortaya çıkan sorgu, kullanıcı adının admin ve şifrenin **sifre** olmadığı ilk kaydı bulur. Kimlik doğrulama için böyle bir kod kullanılıyorsa ve yönetici kullanıcı mevcutsa saldırgan bu kullanıcı olarak oturum açar.

Arabellek Taşması İstismarları

Arabellek taşması istismarları, sabit uzunluklu bir bellek blokunda tutmasına izin verilenden daha fazla veri yazmaya çalışıldığında meydana gelir. Saldırganlar, bitişik bellek adreslerinde depolanan fazla verileri saldırı başlatmak için bir temel olarak kullanabilir.

Basit bir arabellek taşması örneği şu şekildedir (Görsel 8.65).

```
#include <stdio.h>
İnt main (int argc, char * * argv)
{
Char Buffer[11]="SIBERGUVEN";
Strcpy(Buffer,"SIBERGUVENLI");
Printf("% \n",Buffer);
Return 0;
}
```



Görsel 8.65: Arabellek taşması

Sıra Sizde

Bir işletmenin verilerini içeren bir veri tabanı oluşturarak müşteri bilgilerinin gizliliği, yetkilendirme, erişim kontrolü ve güvenli iletişim için gerekli güvenlik önlemlerini alınız.

Değerlendirme

Çalışmanız aşağıda yer alan kontrol listesi kullanılarak değerlendirilecektir. Çalışmanızı yaparken kontrol listesindeki değerlendirme ölçütlerini dikkate alınız.

Kontrol Listesi

Değerlendirme Ölçütleri	Evet	Hayır
1. Veri tabanı şemasını oluşturdu ve gizlilik için önemli alanları belirledi.		
2. Kullanıcı rollerini ve erişim izinlerini tanımladı.		
3. Güvenli SQL kullanımını öğrendi ve örnek sorgular yazdı.		
4. Güvenlik duvarlarını yapılandırdı ve şifre politikalarını uyguladı.		

"Hayır" olarak işaretlenen değerlendirme ölçütleri için ilgili konu ve uygulamaları tekrar ediniz.

ÖLÇME VE DEĞERLENDİRME

A. Aşağıdaki parantezlerin içine cümlelerde verilen bilgiler doğru ise "D", yanlış ise "Y" yazınız.

1. ( ) Veri tabanı tablolarında bir alana birden fazla veri girişi olabilir.

2. ( ) Herhangi bir tablonun tekrarlı bilgiler içerdiği duruma 2NF denir.

3. ( ) Tablodaki benzersiz değerler içeren her bir sütun veya sütunlar grubu aday anahtar olarak adlandırılır.

4. ( ) İlişkisel veri tabanındaki tablolarda birbirleriyle tamamen aynı olan iki kayıt kullanılabilir.

5. ( ) Birincil anahtarlar, birbiriyle aynı olan değerler içerebilir.

B. Aşağıdaki cümlelerde boş bırakılan yerlere doğru sözcükleri yazınız.

6. Bir veri tabanında ..... olması için aynı tablo içinde tekrarlayan sütunlar olmamalı, sütunlarda yalnızca bir değer olmalı ve her satır bir eşsiz anahtara sahip olmalıdır.

7. ....., sabit bir şema gerektirmeyen ve ilişkisel olmayan bir veri tabanı yönetim sistemidir.

8. Olası tüm harf kombinasyonlarını deneyerek SQL oturum açma parolalarını kırmanın bir yolu ..... dır.

C. Aşağıdaki soruları dikkatlice okuyarak doğru cevabı işaretleyiniz.

9. Aşağıdakilerden hangisi SQL'in en önemli temel sorgulama komutudur?

A) Delete
B) Having
C) Insert into
D) Update
E) Select

10. Veri tabanlarını oluşturmak, verilere erişim izni olan kullanıcıları tanımlamak, işletmek, güncellemek ve veri tabanı elemanları ile ilgili her türlü yönetsel gereksinimleri karşılamak için tasarlanmış sistemlere ne ad verilir?

A) Veri Tabanı Kuramı
B) Veri Tabanı Tasarımı
C) Veri Tabanı Yönetim Sistemleri
D) Veri Tanımlama Dili
E) Yapılandırılmış Sorgulama Dili

11. Aşağıdaki özelliklerden hangisi herhangi bir veri kaydı üzerinde aynı anda farklı iki kullanıcı tarafından işlem yapılmak istendiğinde DBMS tarafından kullanılır?

A) Güncelleme
B) Sorgulama
C) Soyutlama
D) Yalıtım
E) Yedekleme

12. Veri çeşitliliği, yapılacak iş sürecinin kapsamı, kullanıcı sayısı arttıkça veri tabanı tasarım ve işletiminin zorlaşması, DBMS'nin dezavantajlarından hangisi ile açıklanır?

A) Altyapı gereksinimi
B) Yüksek maliyetli olması
C) Personel ihtiyacı
D) Donanım gereksinimi
E) Yapı karmaşıklığı

13. Aşağıdakilerden hangisi INSERT komutunun kullanılma amacıdır?

A) Ekleme
B) Güncelleme
C) Kopyalama
D) Sıralama
E) Silme

14. Aşağıdakilerden hangisi ogrenci tablosundaki tüm kayıtları listeleyen SQL komutudur?

A) select * from ogrenci
B) select ograd,ogrsoyad,sinif from ogrenci
C) select all from ogrenci
D) select ogrenci from *
E) select öğrenci_no from ogrenci

15. Aşağıdakilerden hangisi NoSQL veri tabanının avantajlarındandır?

A) Sıkı veri modelleme
B) Sınırlı ölçeklenebilirlik
C) Sınırlı veri depolama gelişimi
D) Kolay şema gelişimi
E) Sınırlı sorgu yetenekleri

16. Aşağıdakilerden hangisi birincil anahtar alanı için uygun değildir?

A) Yaş
B) T.C. kimlik bilgisi
C) Öğrenci numarası
D) Müşteri numarası
E) Pasaport numarası

17. Veri tabanı güvenlik yönetimi aşağıdakilerden hangisini ifade eder?

A) Verilerin güvenliğini sağlamak için yönetimin kurumsal sorumluluğunu ayrıntılarıyla anlatır.
B) Verileri, veri tabanı sistemlerini korumak için kullanılan süreç ve prosedürlerin toplamıdır.
C) Veri tabanı kullanıcı erişimini yönetmek için kullanılan teknikleri içerir.
D) Güvenlik ihlallerini tespit etmek için kullanılan süreç ve prosedürlerdir.
E) Veri tabanındaki verileri şemasız olarak kaydetmek için kullanılan sistemdir.

18. Aşağıdakilerden hangisi SQL enjeksiyonu tanımlar?

A) Verileri veri tabanına kaydeden SQL sorguları
B) Orijinal sorguları düzenleyerek yürütülen SQL sorguları
C) Bir web sistemine giriş yaparak yürütülen SQL sorguları
D) Tabloların veri tekrarını en aza indiren SQL sorguları
E) Verileri web sistemine kaydeden SQL sorguları

D. Aşağıdaki soruların cevaplarını ilgili alana yazınız.

19. Veri tabanı yönetim sistemlerinin avantajları nelerdir?

20. NoSQL veri tabanlarının avantaj ve dezavantajları nelerdir?

CEVAP ANAHTARI

1. ÖĞRENME BİRİMİ

A	**1.** Yanlış	**2.** Doğru	**3.** Doğru	**4.** Yanlış	**5.** Doğru
B	**6.** beyaz şapkalı	**7.** 6698	**8.** delil	**9.** imajını	**10.** fiziksel
C	**11.** C	**12.** A	**13.** E	**14.** B	**15.** D

2. ÖĞRENME BİRİMİ

A	**1.** Doğru	**2.** Yanlış	**3.** Doğru	**4.** Doğru
B	**5.** Demilitarized Zone-DMZ	**6.** Saldırı Tespit Sistemleri	**7.** ISO: 27001	
C	**8.** D	**9.** A	**10.** A	

3. ÖĞRENME BİRİMİ

A	**1.** Doğru	**2.** Doğru	**3.** Yanlış	**4.** Doğru	**5.** Yanlış
B	**6.** Simetrik şifreleme	**7.** diffie-hellman anahtar değişimi	**8.** Yerine koyma	**9.** steganografi	**10.** eliptik eğri algoritması
C	**11.** D	**12.** C	**13.** B	**14.** C	**15.** B

4. ÖĞRENME BİRİMİ

A	**1.** Doğru	**2.** Yanlış	**3.** Yanlış	**4.** Doğru	**5.** Yanlış
B	**6.** Şifreleme	**7.** Güçlü şifreler	**8.** İki faktörlü	**9.** Erişim kontrol	
C	**10.** D	**11.** D	**12.** B	**13.** A	**14.** B
	15. E	**16.** C	**17.** A	**18.** C	**19.** D
	20. B	**21.** E			

5. ÖĞRENME BİRİMİ

A	**1.** Yanlış	**2.** Yanlış	**3.** Doğru		
B	**4.** Çerez	**5.** Pozitif girdi	**6.** 404 hata kodu		
C	**7.** E	**8.** A	**9.** C	**10.** A	**11.** E
	12. A	**13.** D	**14.** B	**15.** C	

6. ÖĞRENME BİRİMİ

A	**1.** Doğru	**2.** Doğru	**3.** Doğru	**4.** Yanlış	**5.** Doğru
B	**6.** Nesnelerin İnterneti (Internet of Things-IoT)	**7.** DDoS	**8.** TEMPEST	**9.** Ubuntu	**10.** AMQP
C	**11.** D	**12.** C	**13.** A	**14.** A	**15.** E

7. ÖĞRENME BİRİMİ

A	**1.** Doğru	**2.** Yanlış	**3.** Yanlış	**4.** Doğru	**5.** Yanlış
B	**6.** İnternet	**7.** Güvenli bir şekilde	**8.** e-postalar ve web siteleri	**9.** Veri bozulması	
C	**10.** E	**11.** C	**12.** E	**13.** D	**14.** B
	15. C	**16.** B	**17.** C	**18.** C	**19.** A
	20. B	**21.** D	**22.** D	**23.** C	**24.** C

8. ÖĞRENME BİRİMİ

A	**1.** Yanlış	**2.** Yanlış	**3.** Doğru	**4.** Yanlış	**5.** Yanlış
B	**6.** 1.Normal Form	**7.** NoSQL veri tabanı	**8.** Kaba kuvvet saldırıları		
C	**9.** E	**10.** C	**11.** D	**12.** E	**13.** A
	14. A	**15.** D	**16.** A	**17.** B	**18.** D

KAYNAKÇA

- MEB (2022). **Siber Güvenlik Alanı Çerçeve Öğretim Programı.** Ankara: T.C. MEB.
- TDK (2012). **Türk Dil Kurumu Yazım Kılavuzu.** Ankara: Türk Dil kurumu yayınları
- Abuagoub, A. M. (2019). IoT security evolution: challenges and countermeasures review. International Journal of Communication Networks and Information Security, 11(3), 342-351.
- Ahmid, M., & Kazar, O. (2023). A comprehensive review of the internet of things security. Journal of Applied Security Research, 18(3), 289-305.
- Ashima Bhatnagar Bhatia, & Vaibhav Bansal. (2015). Database Management System. Alpha Science Internation Limited.
- Avcı İ., (2022). "Akıllı evlerde IoT teknolojileri ve siber güvenlik", Avrupa Bilim ve Teknoloji Dergisi, (34), 226-233.
- Bray, S. W. (2020). Implementing Cryptography Using Python. John Wiley & Sons.
- Burma, Z. A., (2009).Veritabanı ve uygulamaları. Seçkin Yayıncılık.
- Dr Janusz R. Getta, (2023).NoSQL Database Systems, School of Computing and Information Technology -University of Wollongong, Created by Janusz R. Getta, CSCI235 Database Systems.
- Dr. Sampath Jayarathna. (2019). Introduction to Data Science & Analytics, Lecture 6- NoSQL. Old Dominion University. C J Date.
- Dr. Subasish Mohapatra.Database Engineering Lecture Notes. (2019).Department of Computer Science and Application College of Engineering and Technology, Bhubaneswar,Biju Patnaik University of Technology, Odisha
- Kaya Y. TekinR. (2007). Fundamentals of Database Systems5th Edition, 2007, Papatya Yayıncılık Eğitim.
- Klima, R. E., Klima, R., Sigmon, N. P., & Sigmon, N. (2018). Cryptology: classical and modern. Chapman and Hall/CRC.
- Paar, C., & Pelzl, J. (2009). Understanding cryptography: a textbook for students and practitioners. Springer Science & Business Media.
- Ramez Elmasri, ShamNavathe. (2007). An IntroductiontoDatabase Systems, 8th Edition. Pearson/ AddisonWesley.
- Rekha, S., Thirupathi, L., Renikunta, S., & Gangula, R. (2023). Study of security issues and solutions in Internet of Things (IoT). Materials Today: Proceedings, 80, 3554-3559.
- Sağıroğlu, Ş. (2019). Siber Güvenlik ve Savunma Standartlar ve Uygulamalar. Ankara.
- Sağıroğlu, Ş., & Akleylek, S. (2020). Siber Güvenlik ve Savunma Biyometrik ve Kriptografik Uygulamalar. Aralık.
- Sağıroğlu, Ş., & Akleylek, S. (2021). Siber Güvenlik ve Savunma Blok Zincir ve Kriptoloji. Ankara.
- Sağıroğlu, Ş., & Akleylek, S. (2022). Siber Güvenlik ve Savunma Siber Güvenlik Ontolojisi, Tehditler ve Çözümler. Ankara.
- Sağıroğlu, Ş., & Alkan, M. (2018). Siber Güvenlik ve Savunma Farkındalık ve Caydırıcılık. Ankara.

- Sağıroğlu, Ş., & Şenol, M. (2019). Siber Güvenlik ve Savunma Problemler ve Çözümler. Ankara.
- Schneier, B. (2007). Applied cryptography: protocols, algorithms, and source code in C. john wiley & sons.
- Serengil, Sefik. (2011). Attacking Turkish Texts Encrypted by Homophonic Cipher. DOI: 10.13140/RG.2.1.3706.7608.
- Shiu, H. J., Yang, C. T., Tsai, Y. R., Lin, W. C., & Lai, C. M. (2023). Maintaining Secure Level on Symmetric Encryption under Quantum Attack. Applied Sciences, 13(11), 6734.
- Siber Güvenlik Alanı Çerçeve Öğretim Programı, Ankara, 2023.
- Smart, P. N. (2016). Cryptography made simple. Springer International Publishing Switzerland.
- Stallings W. (2006). Cryptography and Network Security : Principles and Practices. 4th ed. Upper Saddle River N.J: Pearson/Prentice Hall; 2006.
- Stinson, D.R. Cryptography - theory and practice. Discrete mathematics and its applications series. CRC Press. 1995
- Türkiye Büyük Millet Meclisi. (2016). 6698 sayılı "Kişisel Verilerin Korunması Kanunu".
- V. Vidhya, G. Jeyaram, & K. R. Ishwarya. (2016). Database Management Systems. Alpha Science Internation Limited.
- Wenbo M. (2003), Modern Cryptography: Theory and Practice. Publisher: Prentice. Hall PTR. ISBN: 0-13-066943-1.
- Wobst, R. (2007). Cryptology unlocked. John Wiley & Sons.
- Yaacoub, J. P. A., Noura, H. N., Salman, O., & Chehab, A. (2023). Ethical hacking for IoT: Security issues, challenges, solutions and recommendations. Internet of Things and Cyber-Physical Systems, 3, 280-308.
- Yarımağan, Ü. (2000). Veri tabanı sistemleri. Akademi & Türkiye Bilişim Vakfı.
- Yılmaz, H. (2014). TS ISO/IEC 27001 Bilgi Güvenliği Yönetimi Standardı Kapsamında Bilgi Güvenliğinin Yönetim Sisteminin Kurulması ve Bilgi Güvenliği Risk Analizi. DENETİŞİM, 45-59.

Kaynakça APA'ya göre düzenlenmiştir.

GENEL AĞ KAYNAKÇASI VE GÖRSEL KAYNAKÇASI

http://kitap.eba.gov.tr/karekod/Kaynak.php?KOD=3436

NOT