

T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

BSM 498 BİTİRME ÇALIŞMASI

**AKILLI ŞEBEKELER İÇİN GÜVENLİ YAZILIM
GELİŞTİRME TEMELLERİNE UYGUN
BİR UTM ARAYÜZ TASARIMI**

**G191210303 – Barış YILMAZ
G191210387 – Cemal ASLAN**

**Fakülte Anabilim Dalı : BİLGİSAYAR MÜHENDİSLİĞİ
Tez Danışmanı : Dr. Öğr. Üyesi Musa BALTA**

2022-2023 Bahar Dönemi

T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

**AKILLI ŞEBEKELER İÇİN GÜVENLİ YAZILIM
GELİŞTİRME TEMELLERİNE UYGUN
BİR UTM ARAYÜZ TASARIMI**

BSM 498 - BİTİRME ÇALIŞMASI

**G191210303 – Barış YILMAZ
G191210387 – Cemal ASLAN**

Fakülte Anabilim Dalı : BİLGİSAYAR MÜHENDİSLİĞİ

Bu tez .. / .. / ... tarihinde aşağıdaki jüri tarafından oybirliği / oyçokluğu ile kabul edilmiştir.

.....
Jüri Başkanı

.....
Üye

.....
Üye

ÖNSÖZ

Günümüzde teknolojinin önemi oldukça artmakta ve bununla birlikte teknolojik, uzaktan takibi yapılmayan ürün ve süreçler için çalışmalar artmaktadır. Bu tarz uzaktan erişim bulunan ve izleme sistemleri gerektiren gelişmeler beraberinde güvenlik endişelerini getirmektedir. Güvenlik endişelerini mümkün olduğunda yok etmek, minimuma indirmek projelerin ana fikri kadar mühimdir.

İÇİNDEKİLER

ÖNSÖZ	iii
İÇİNDEKİLER	iv
SİMGELER VE KISALTMALAR LİSTESİ	vi
ŞEKİLLER LİSTESİ	vii
ÖZET.....	ix
BÖLÜM 1. GİRİŞ.....	1
BÖLÜM 2. GÜVENLİ YAZILIM GELİŞTİRME KİLAVUZU	2
2.1. Yazılım Güvenliğinin İlkeleri	2
BÖLÜM 3. UTM ARAYÜZÜ	6
3.1. Sektördeki Ticari Ürünlerin UTM Arayüz Örnekleri.....	6
BÖLÜM 4. TASARIM	12
4.1. Ana Sayfa Görünümü	12
4.2. Giriş	14
4.2.1. Authorization.....	15
4.2.2. Authentication	16
4.3. Kullanıcı	22
4.3.1. Parola Geçerlilik Süresi.....	23
4.3.2. Oturum Geçerlilik Süresi	24
4.4. Rol	24
4.4.1. Admin.....	24
4.4.2. Süper Admin.....	25
4.4.2.1. Kullanıcı Listesi	25
4.4.2.2. Veritabanı İşlemleri	28
BÖLÜM 5. MODÜLLER.....	29

5.1.	Log Yönetimi.....	29
5.1.1.	Cihaz Logları.....	29
5.1.2.	Kullanıcı Logları	32
5.2.	Zararlı Yazılım	32
5.2.1.	Solucan (Worm)	34
5.2.2.	Casus Yazılım (Spyware)	35
5.2.3.	Reklam Yazılımı (Adware).....	35
5.2.4.	Rootkit.....	35
5.3.	Anomaly Detection.....	35
5.3.1.	Point Anomalies	37
5.3.2.	Contextual Anomalies	37
5.3.3.	Collective Anomalies	37
5.4.	Tehdit İstihbaratı	38
5.5.	Zararlı Yazılım Analizi.....	40
BÖLÜM 6.	SONUÇLAR VE ÖNERİLER.....	42
KAYNAKLAR.....		43
ÖZGEÇMİŞ		44
BSM 498 BİTİRME ÇALIŞMASI DEĞERLENDİRME VE SÖZLÜ SINAV TUTANAĞI		45

SİMGELER VE KISALTMALAR LİSTESİ

UTM	: Unified Threat Management (Birleşik Tehdit Yönetimi)
SQL	: Structured Query Language
BT	: Bilgi Teknolojileri
ISO	: International Organization for Standardization
URL	: Uniform Resource Locator
VPN	: Virtual Private Network
ERD	: Entity-Relationship Diagram
SQL	: Structured Query Language
API	: Application Programming Interface
HTML	: Hyper-Text Markup Language
HTTP	: Hyper-Text Transfer Protocol

ŞEKİLLER LİSTESİ

Şekil 3.1. UTM Yapılandırması	6
Şekil 3.2. ManageEngine Log360	7
Şekil 3.3. WatchGuard Firebox.....	7
Şekil 3.4. Exosphere.....	8
Şekil 3.5. AlienVault.....	9
Şekil 3.6. Comodo Dome Firewall.....	9
Şekil 3.7. Security Management Summary Dashboard by Carole Fennelly.....	10
Şekil 3.8. STMBugShield	10
Şekil 3.9. Mandiant Advantage.....	11
Şekil 3.10. Anomali.....	11
Şekil 4.1. Ana Sayfa Görünümü	13
Şekil 4.2. Giriş Ekranı.....	14
Şekil 4.3. Maksimum Giriş Denemesi Ayarı Kodu	15
Şekil 4.4. Hesap Kapatıldı Sayfası.....	15
Şekil 4.5. Authorization Require Kodu.....	15
Şekil 4.6. Erişim Reddedildi Sayfası.....	16
Şekil 4.7 Authenticator Resource Filter	17
Şekil 4.8. Authenticator Resource Filter Devamı	17
Şekil 4.9. Authenticator Resource Filter Etkinleştirme	17
Şekil 4.10. Authenticator Kurulum Sayfası	18
Şekil 4.11. Authenticator Kurulumu Gerçekleştı Sayfası	19
Şekil 4.12. Authenticator Kurtarma Kodu Üretme Sayfası.....	19
Şekil 4.13. Authenticator Kurtarma Kodları Sayfası	19
Şekil 4.14. 2fa Doğrulama Kodu.....	20
Şekil 4.15. 2fa Kurtarma Kodu	21
Şekil 4.16. Google Authenticator Doğrulama Kodu	21
Şekil 4.17. AkilliSayacUser Sınıfı	22
Şekil 4.18. Kullanıcı Paneli.....	22
Şekil 4.19. Password Resource Filter.....	23

Şekil 4.20. Password Resource Filter Devamı	23
Şekil 4.21. Password Resource Filter Etkinleştirme	23
Şekil 4.22. Maksimum Oturum Süresi Ayarı Kodu.....	24
Şekil 4.23. Maksimum Oturum Süresi Cookie Görüntüsü	24
Şekil 4.24. Süper Admin Barı	25
Şekil 4.25. Kullanıcı Listesi Sayfası	25
Şekil 4.26. Hesap Durumu Aktif Butonu	26
Şekil 4.27. Hesap Durumu İnaktif Butonu.....	26
Şekil 4.28. Kullanıcı Kayıt Sayfası	27
Şekil 4.29. Veritabanı İşlemleri Sayfası.....	28
Şekil 4.30. Veritabanı Yedeği Dosyası	28
Şekil 5.1. Log Yönetimi ERD	29
Şekil 5.2. Dosyaları İçे Aktarma ve Veritabanına Ekleme	30
Şekil 5.3. Log Tiplerini Veritabanına Ekleme	30
Şekil 5.4. Log Tiplerini Veritabanına Ekleme Devamı.....	31
Şekil 5.5. Cihazları Veritabanına Ekleme	31
Şekil 5.6. Cihaz Logları Tablosu.....	32
Şekil 5.7. Kullanıcı Logları Sayfası	32
Şekil 5.8. Zararlı Yazılım Tiplerini Veritabanına Ekleme	33
Şekil 5.9. Zararlı Yazılım Grafiği	33
Şekil 5.10. Zararlı Yazılım Tablosu	34
Şekil 5.11. Zararlı Yazılım ERD	34
Şekil 5.12. Anomali Tiplerini Veritabanına Ekleme.....	36
Şekil 5.12. Anomali Grafiği	36
Şekil 5.13. Anomali ERD.....	37
Şekil 5.14. Tehdit İstihbaratı Actor Activity.....	38
Şekil 5.15. Tehdit İstihbaratı Son Aktiviteler	38
Şekil 5.16. Tehdit İstihbaratı En Aktif Zararlı Yazılımlar Top10.....	39
Şekil 5.17. Tehdit İstihbaratı En Aktif Güvenlik Açıkları Top10.....	39
Şekil 5.18. Zararlı Yazılım Analizi Türe Göre Göstergeler.....	40
Şekil 5.19. Zararlı Yazılım Analizi Ülkelere Göre Göstergeler.....	40
Şekil 5.20. Zararlı Yazılım Analizi Kaynağa Göre Göstergeler	41

ÖZET

Anahtar kelimeler: Akıllı Sayaç, Web Uygulama, UTM Dashboard, Güvenli Yazılım

Akıllı sayaçlar kullandıkları haberleşme teknolojileriyle beraber uzaktan anlık takip, kontrol ve izleme, esneklik, güvenilirlik, insan etkisi ile oluşabilecek hatanın ortadan kaldırılması, anlık veri transferi şeklinde sıralanabilecek yenilikler sunmaktadır. Bu yeniliklerin beraberinde uzaktan izleme sistemi geliştirilmesi ve bu geliştirilecek sistemin güvenliğinin maksimum seviyede olması elzemdir.

Bu bitirme çalışması ile Güvenli Yazılım Geliştirme Kılavuzlarına uygun olarak akıllı sayacı uzaktan kontrolünün sağlanabildiği, yalnızca yetkili kişiler tarafından sisteme giriş sağlanabilen ve Anomali Tespiti, Zararlı Yazılım, Log Sistemi, Zararlı Yazılım Analizi ve Tehdit İstihbaratı'nın bulunduğu bir UTM Arayüzü ASP.NET Core 7.0 web uygulaması geliştirilerek takibe imkan sağlanmıştır.

BÖLÜM 1. GİRİŞ

Günümüzde içerisinde internet erişimi bulunan sistemlerde tasarım, erişilebilirlik, ana fikri kadar sistemin güvenliği, erişimi bulunmayan iyi veya kötü niyetli kişilerin sisteme erişememesi de önemlidir. Bu tasarım ile birlikte uzaktan erişim ile izleme yapılacak olan akıllı sayaç için güvenli yazılım geliştirme prosedürlerine uygun olarak bir ASP.NET Core 7.0 web uygulaması geliştirilmiştir.

Yazılımin tehditlerden uzak bir şekilde sürdürülebilmesi ve bu bağlamda güvenliği sağlayabilmek için çeşitli organizasyonlar tarafından güvenli yazılım geliştirmenin standartları belirlenmiştir. Geliştirilecek her yazılımin bu standartlara bağlı olarak geliştirilmesi yazılımin güvenliği için mühimdir. Bu bitirme çalışmasında da bu standartlara bağlı olarak bir uygulama geliştirilmesi sağlanmıştır.

Bu bitirme çalışmasında güvenli yazılım geliştirme standartlarıyla yapılan işlemler, UTM arayüzünün amacı, arayüzde kullanılan modüllerin tanımları ve içerikleri, benzer UTM arayzları, kullanıcı ve rollerle ilgili bilgiler, authorization ve authentication bilgileri ve çalışmaya ait görüntüler bulunmaktadır.

BÖLÜM 2. GÜVENLİ YAZILIM GELİŞTİRME KİLAZUZU

Yazılımın, saldırısı veya tehdit altındayken işlevlerini doğru bir şekilde yerine getirmeye devam edecek şekilde korunmasına yazılım güvenliği denir. Bir yazılımın güvenliği yazılımı ortaya çıkarmak kadar mühimdir. Yazılım güvenliği faaliyetlerinin amacı tüm bilgi güvenliği saldırılara karşı daha dirençli ve hatasız çalışan yazılım üretmektir. Yazılım geliştirirken proje boyunca güvenli yazılım geliştirme kurallarının nasıl ve ne zaman uygulanacağıının belirlenmesi, izlenmesi ve denetimi ve yazılım güvenliğinin sürekliliğinin sağlanması önem arz etmektedir. Bu sürecin yürütülmemesi durumunda saldırılara maruz kalabilir ve yazılımın tamamen durdurulması ya da doğru çalışmasının engellenmesi, istenmeyen amaca hizmet etmesi veya bilgi sızıntılarına engel olunamaz [1].

Tasarımı gizli tutarak güvenlik yaklaşımı, yeni nesil araç ve yöntemlerle (tersine mühendislik, kod analizi araçları vb.) artık geçerliliğini yitirmiştir. Güvenlik tasarımının karşı tarafça bilindiği varsayılarak güvenlik analizlerinin yapılması gerekmektedir.

2.1. Yazılım Güvenliğinin İlkeleri

Çalışmamızda ilkelere göre tamamlananlar derlenmiştir.

- ✓ Her bir prosedür, modül kendi işini bitirmesi için gerekli en az haklarla çalıştırılmalı ve bu süreç esnasında bu haklara gereken en az süre boyunca sahip olmalı
- ✓ Her bir nesneye yapılan erişimlerde yetki kontrolü yapılmalı ve bu kontrol normal durumların dışında başlatma, kapatma veya istek, yanıt aşamalarında da yapılmalı.
- ✓ Gereksiz özellikler eklenmemelidir.
- ✓ Kullanıcılar işini en kolay şekilde ancak en az yetkiyle yapabilmeli.

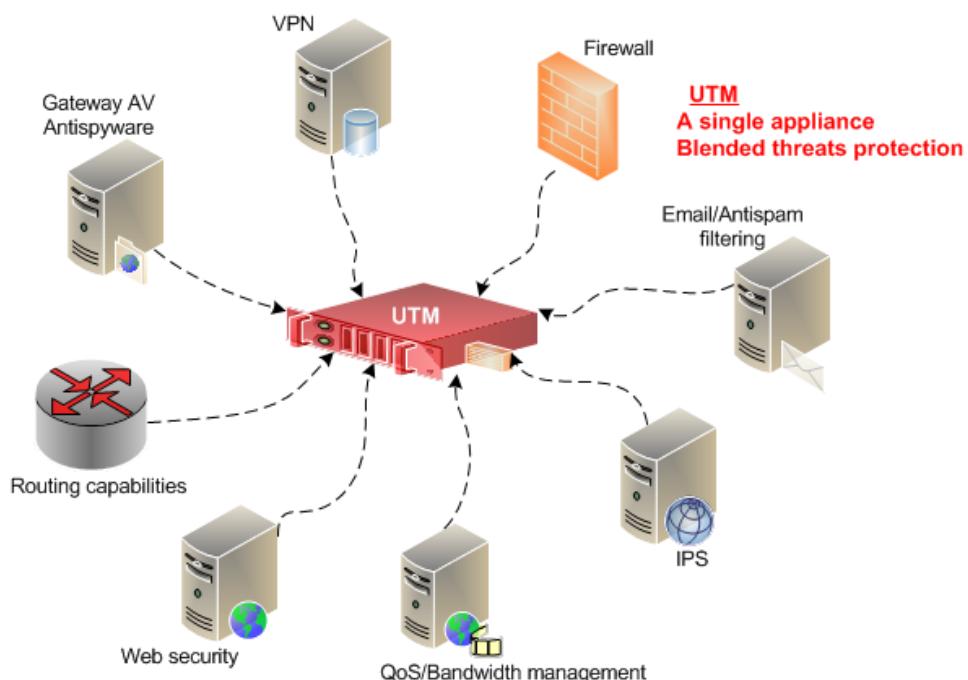
- ✓ Uygulama veritabanında kişisel veri içeren birincil anahtar (kimlik no, e-posta adresi vb.) kullanılmamalıdır.
- ✓ Sistem mimarisi zayıf yönleri veya zayıf noktaları bulmak için saldırganın bakış açısı ile incelenmeli.
- ✓ Güvenli diller, güvenli ve onaylanmış kütüphaneler kullanılmalıdır. Eski ve güvensiz kütüphaneler kullanılmamalıdır.
- ✓ Kasıtlı veya kasıtsız uygulamada güvenlik açığı yatacak kodu kimin yarattığının kontrolü için sürüm kontrolü yapılmalıdır.
- ✓ Mimarideki tüm yazılım bileşenleri tanımlı olmalı ve ihtiyaç duyulmayan bileşenler kaldırılmalıdır.
- ✓ Gereksinimler açık, tutarlı, tam, uygulanabilir, takip edilebilir ve doğrulanabilir olmalıdır.
- ✓ Tüm parola alanlarında kullanıcı giriş yaparken kullanıcının parolası maskelenmeli ve açık olarak görünmemelidir.
- ✓ Kimlik doğrulama başarısız olduğu takdirde güvenli bir duruma geçilmeli ve saldırganların yetkisiz oturum açmaları engellenmelidir.
- ✓ Hesaba yeniden erişebilecek tüm hesap kimlik doğrulama işlevleri (profil güncelleme, parolamı unuttum vb.) en az ana kimlik doğrulama mekanizması kadar saldırılara dayanıklı olmalıdır.
- ✓ Kaynak kodunda veya kaynak kodu depolarında gizli bilgiler, API anahtarları ve parolalar mevcut olmamalıdır.
- ✓ Uygulamanın yönetim arayuzlerine güvenilmeyen taraflarca erişilmesi engellenmelidir.
- ✓ Güvenilmeyen kaynaklardan alınan dosyaların türü doğrulanmalı ve zararlı bir içeriğe sahip olup olmadığı kontrol edilmelidir.
- ✓ Oturumlar belirli bir süre etkinlik olmadığından kendiliğinden sonlanmalıdır.
- ✓ Kimlik doğrulamaya erişilen tüm sayfalardan oturum kapatma işlevine erişilebilmelidir.
- ✓ Oturum kimliğinin URL, hata mesajları ve iz kayıtları içerisinde yer almaması sağlanmalıdır. URL içerisinde oturum kimliğinin yeniden yazılması engellenmelidir.
- ✓ Tüm kimlik doğrulama ve yeniden kimlik doğrulama işlemleri sonucunda yeni bir oturum ve yeni bir oturum kimliği üretilmelidir.

- ✓ Kullanıcı sadece yetkilendirildiği uygulama bileşenlerine ve kaynaklara erişebilmeli ve bunları kullanabilmelidir.
- ✓ Bellekte tutulan önemli veriler gereksinimi sona erdiğinde güvenlik ihlali oluşturamayacak şekilde silinmelidir.
- ✓ Uygulama, hassas veri ve kişisel verileri içeren hata mesajı veya iz kaydı üretmemelidir.
- ✓ Uygulama tarafından, istemci ve sunucu tarafında, kabul edilen her bir veri tipi için girdi doğrulama denetimi yapılmalıdır.
- ✓ Değişen parola fonksiyonu eski parolayı, yeni parolayı ve bir parola onayını kapsamalıdır.
- ✓ Kaba kuvvet saldırıları ya da servis dışı bırakma saldırıları gibi otomatik yapılan yaygın kimlik doğrulama saldırılarını önlemek için istekler azaltılmalıdır. Aşırı kimlik doğrulama denemelerini engellenmeli.
- ✓ Desteklenmeyen, güvensiz veya teknolojisi zaman aşımına uğramış istemci teknolojileri kullanılmamalıdır.
- ✓ Uygulama, ayar ve denetim dosyaları kullanıcı verisiyle aynı konumda depolamamalıdır.
- ✓ Web uygulamalarında oturum cerezlerinde HTTPOnly bayrağı etkin olmalıdır.
- ✓ Tüm rastgele üretilen sayılar, dosya isimleri, global eşsiz değerler (GUID) ve karakter dizilerinin saldırgan için tahmin edilemez olması sağlanmalıdır. Rastgele sayıların yüksek entropiye sahip olarak üretilmelidir.
- ✓ Tüm anahtar ve şifreler kullanımı tamamlandıında, tamamen sıfırlanarak yok edilmelidir.
- ✓ Tüm formlarda istemci tarafında yapılan ön bellekleme işlevselliği önemli veriler için kapatılmalıdır.
- ✓ Uygulama, saklama gereksinimi sona erdikten sonra önemli verileri güvenlik sonunu yaratmayacak şekilde silinmelidir.
- ✓ İz kayıtlarında olayların zaman sıralamasına ilişkin araştırma yapılabilecek şekilde zaman bilgisi yer almalıdır.
- ✓ Uygulama tarafından üretilen iz kayıtları hassas bilgi içermemelidir.
- ✓ Uygulama hassas bilgileri formlarda bulunan gizli alanlarda saklamamalıdır.

- ✓ Uygulama, web servis kimlik doğrulama ve yetkilendirmesi için oturum temelli yapılar kullanacak şekilde tasarlanmalıdır.
- ✓ Uygulama, web servislerinden şifreli olarak paylaşılan verileri yine şifreli olarak saklayacak şekilde tasarlanmalıdır.
- ✓ Uygulama, kişisel veriler üzerinde işlem yapılması ana amaç olmayan durumlarda kişisel verileri maskeleyerek görüntülemeli, aktarmalı veya işlemelidir.
- ✓ Yazılımda kullanılan tüm varsayılan değerler güvenliği artıracak şekilde seçilmeli ve her bir nesnenin varsayılan erişimi hiç olmalı
- ✓ Parolalar için bir en uzun geçerlilik süresi tanımlanmış olmalıdır.
- ✓ Uygulama kullanıcının son başarılı oturum açma tarih ve saatini görüntülemelidir.
- ✓ Yüksek güvenlik gerektiren işlemlerde tek bir koşula bağlı olarak izin verilmemeli
- ✓ Sistem ve yazılım içerisindeki modüller arasında yalıtım sağlanmalı
- ✓ Sunucuda yapılan girdi doğrulama hataları, isteğin reddi ile sonuçlanmalı ve iz kaydı oluşturulmalıdır.
- ✓ Yetkisiz bir kullanıcının yazılım ortamından veri alabileceği, girebileceği veya yetkisiz işlemlerde bulunabileceği noktalar tespit edilmeli, sınırlandırılmalıdır.
- ✓ Hassas işlevler gerçekleştirilmeden önce, yeniden kimlik doğrulama, daha güçlü bir mekanizmayla kimlik doğrulama, ikinci faktör veya işlem imzalama gibi yöntemler uygulanmalıdır.
- ✓ Uygulama, başarısız sistem başlatma, başarısız sonlandırma veya başarısız kapatma gibi işlemlerde güvenli bir duruma geçmelidir.
- ✓ SQL Injection engellemek için bütün veritabanı sorguları, parametre olarak yapılmalı ve veritabanına erişimde kullanılan dile karşı önleyecek denetimler yapılmalıdır.
- ✓ Kullanılan veritabanının dışarıya aktarımı ancak veritabanı yönetim yetkisi olan hesaplarla yapılmalı.

BÖLÜM 3. UTM ARAYÜZÜ

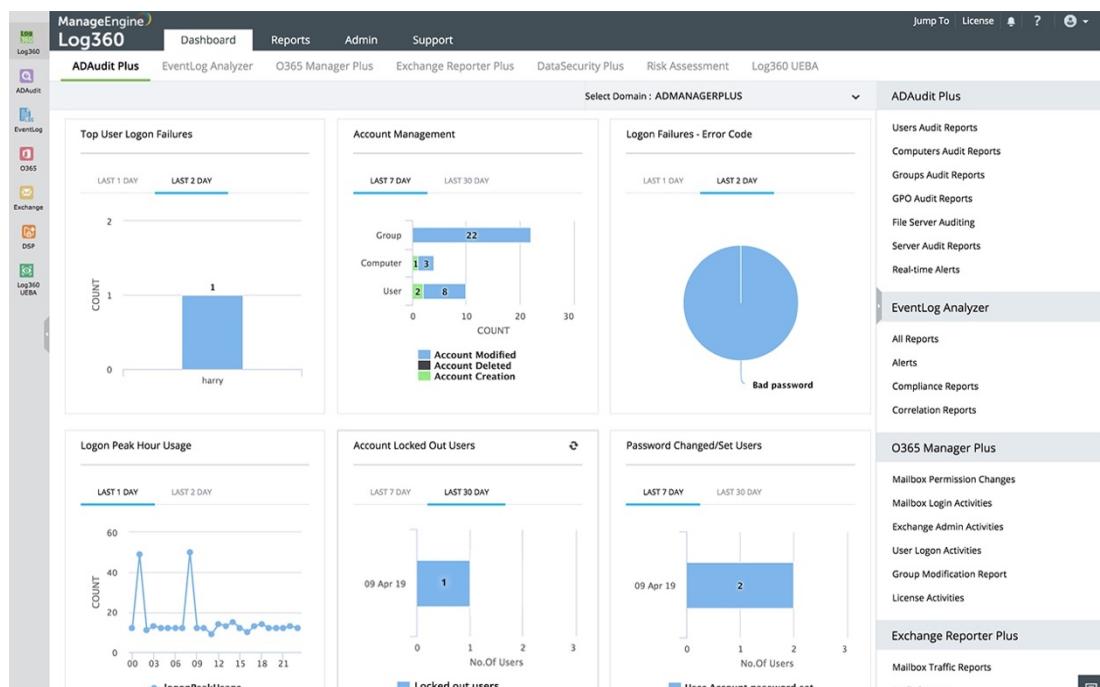
UTM, birden çok güvenlik özelliği ve servislerinin bir cihazda veya hizmette sunularak kullanıcıların basitleştirilmiş bir şekilde korunması ve kontrolünün sağlanmasıdır. Dolayısıyla merkezi bir yönetim sağlanarak kontrol ve organizasyon kolaylaştırılmış olur. İçerisinde Güvenlik Duvarı, VPN, Saldırı Tespit ve Önleme, Web Filtreleme, Antivirüs Gateway, Anti-Spam, İçerik Filtreleme, Trafik Ayarlama gibi özellikler ve daha fazlasını barındırabilir. Bunların yanı sıra yönetim konsolu sayesinde verileri ve olayları raporlayarak bir sorun durumunda bunu giderme aşamasında yöneticilere otomatik olarak rehberlik edebilir ve ayrıca maliyeti de azaltabiliriz [2].



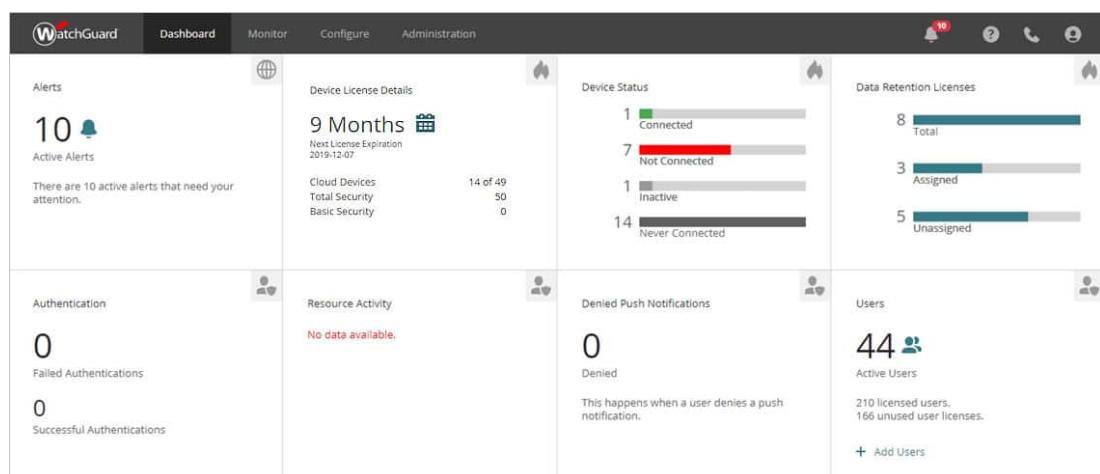
Şekil 3.1. UTM Yapılandırması

3.1. Sektördeki Ticari Ürünlerin UTM Arayüz Örnekleri

Gartner'dan [3] sektördeki ticari ürünlerin arayüzlerini ve modüllerini barındıran gözlemlenebilirliği kolay olan bazı UTM arayızları örnek olmasından açısından aşağıda bulunmaktadır ve bitirme çalışmamız bu arayızlar referans alınarak geliştirilmiştir.



Şekil 3.2. ManageEngine Log360



Şekil 3.3. WatchGuard Firebox

Exosphere

Home > Dashboard

Dashboard

Select Period: 2020-01-23 ~ 2020-01-30

Select Group: Exosphere | Select Sub-Group

Security Scores

Score Range	Persons
0-10	~10
10-20	~10
20-30	~5
30-40	~5
40-50	~5
50-60	~5
60-70	~10
70-80	~10
80-90	~100
90-100	~10

Month	Points
Jan	~95
Feb	~90
Mar	~92
Apr	~94
May	~96
Jun	~93
Jul	~97
Aug	~98
Sep	~95
Oct	~96
Nov	~97
Dec	~94

Outsider Threat Protection

Anti-Malware

189 Disinfected, 2 Quarantined, 0 Skipped

Anti-Ransomware

8 Unknown Process, 11 Blocked Access to Files, 4 Blocked Applications

Web Protection

10 Blocked, 21 Warned

Data Loss Prevention

Device Control

4 Devices Blocked, 2 File Copy Blocked, 11 File Copied

Application Control

2 Applications Blocked, 5 File Transfer Blocked, 2 File Transfer Allowed

Screen Capture Control

11 Blocked

Data Discovery

3 Real Time Detection

Web Control

0 Blocked, 7 Warned

Print Control

0 Blocked, 221 Watermark, 0 Allowed

PC Healthcheck

PC Healthcheck

2 Secured Devices, 13 Weak Devices

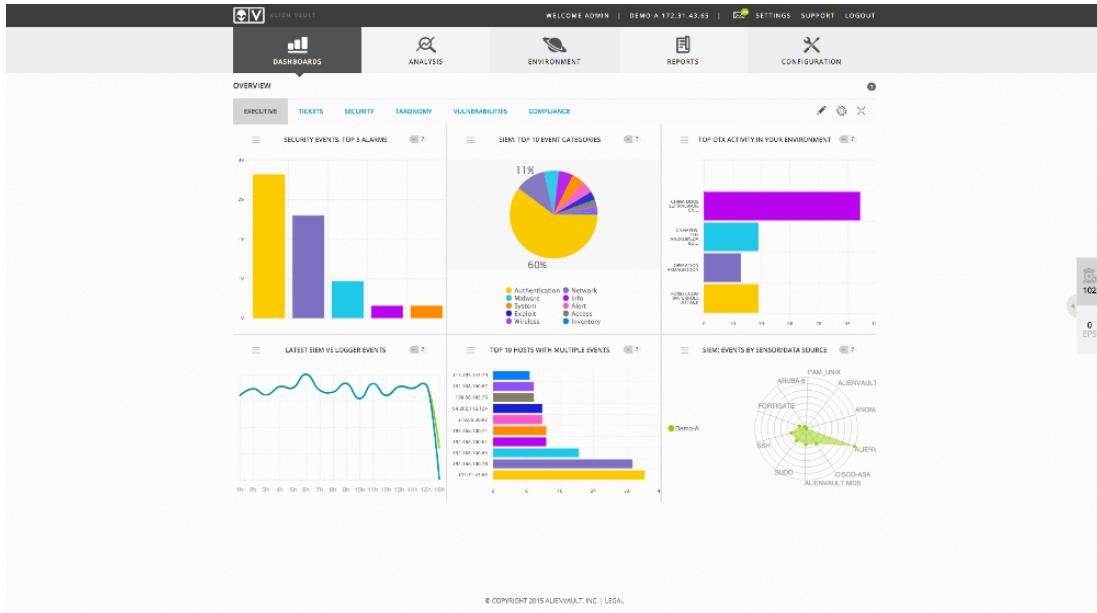
Backup

Backup

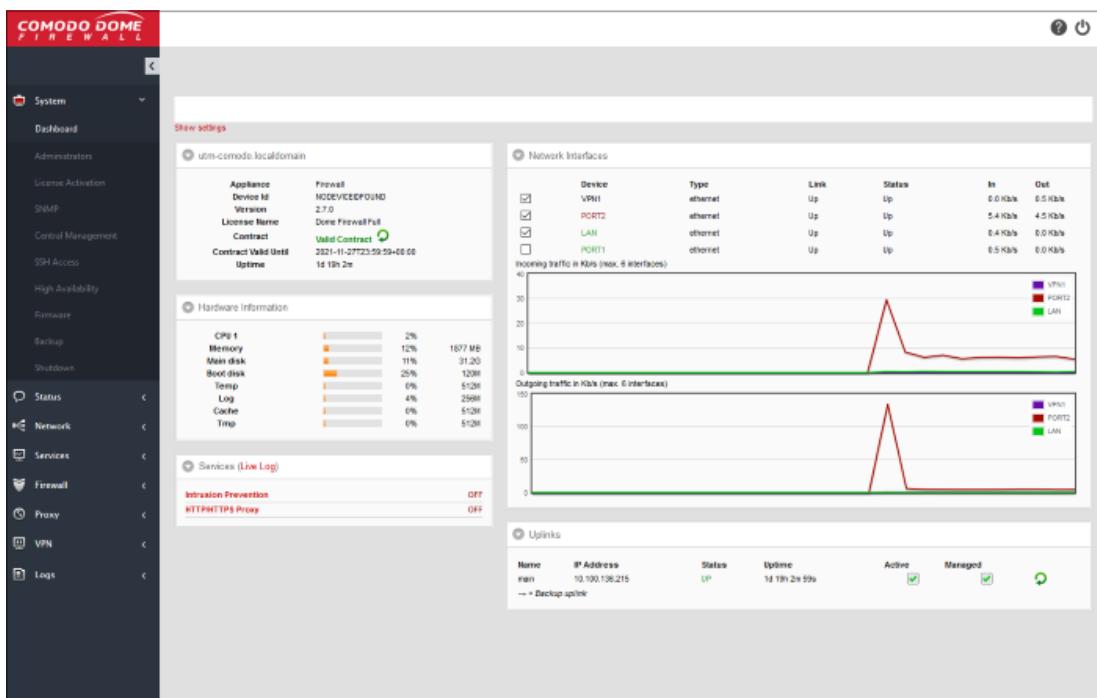
200GB Storage Used, 153K Files Backed Up, 10 Devices Backed Up, 4 Devices Not Backed Up

Copyright Exosphere, Inc. All rights reserved.

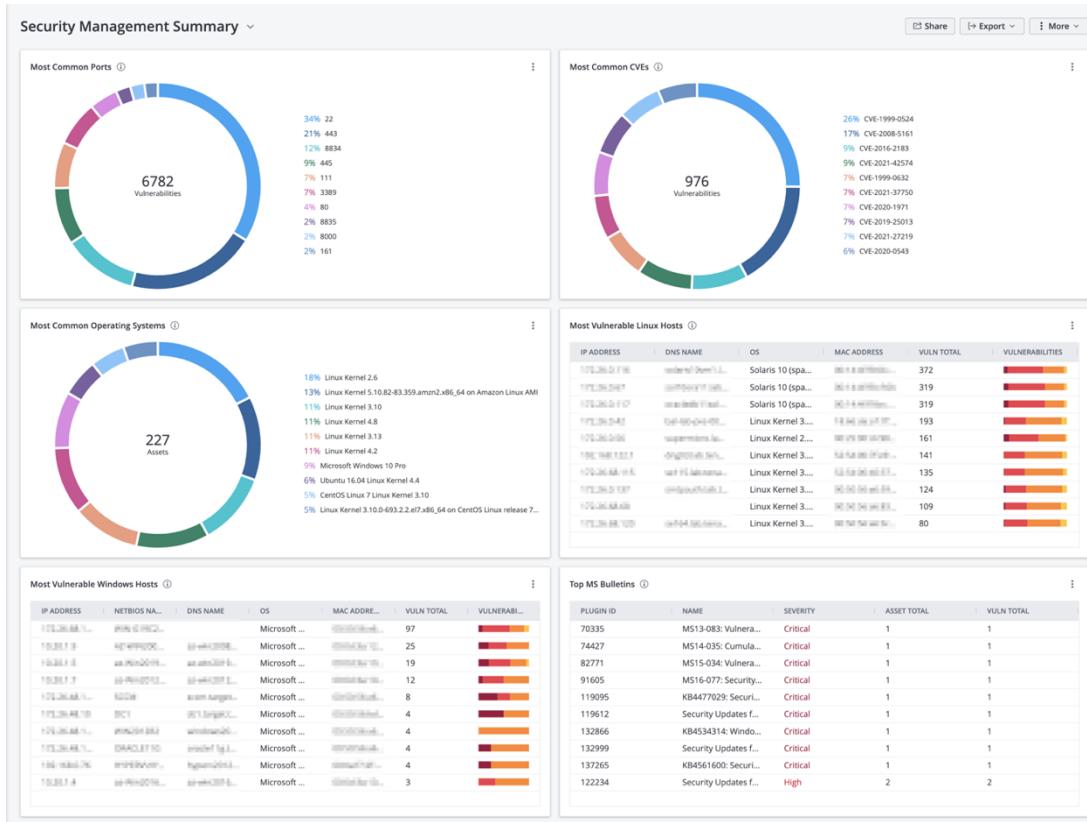
Şekil 3.4. Exosphere



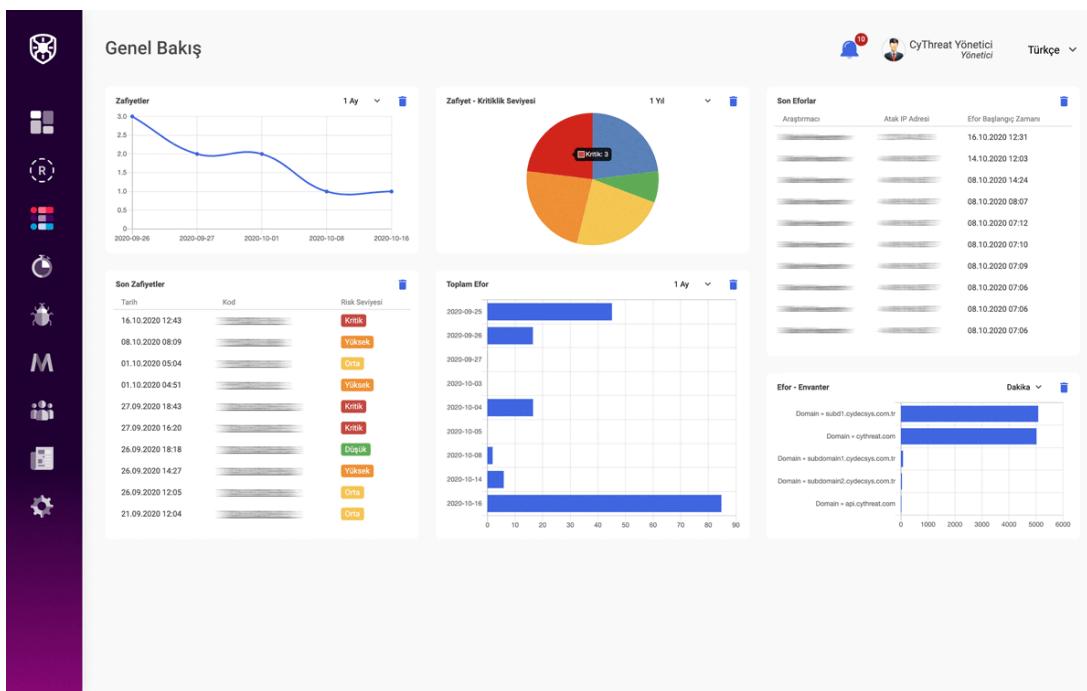
Şekil 3.5. AlienVault



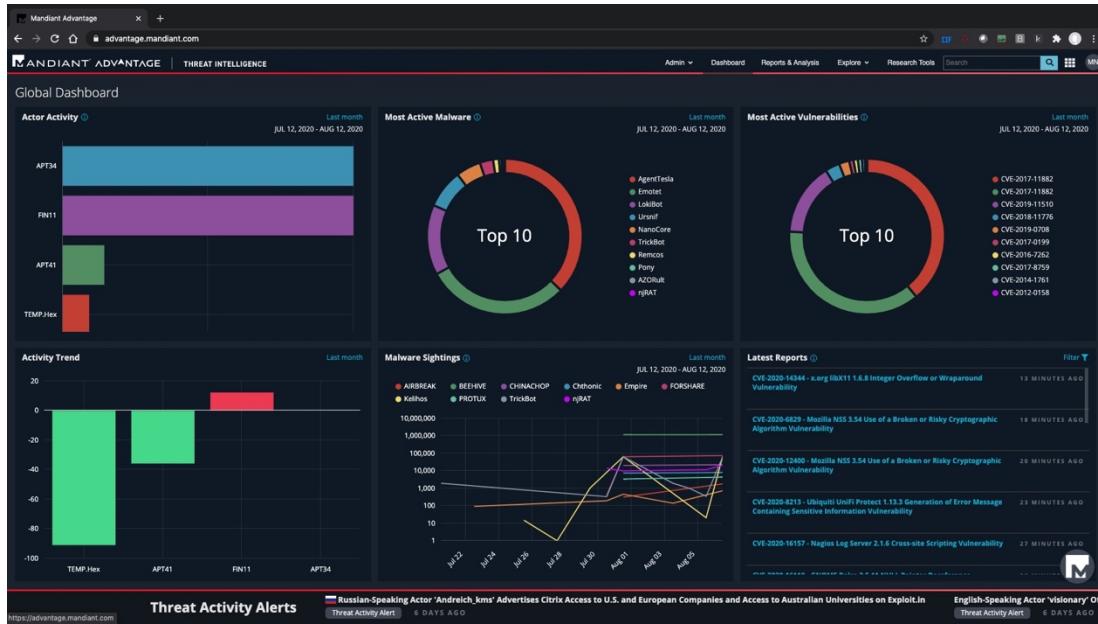
Şekil 3.6. Comodo Dome Firewall



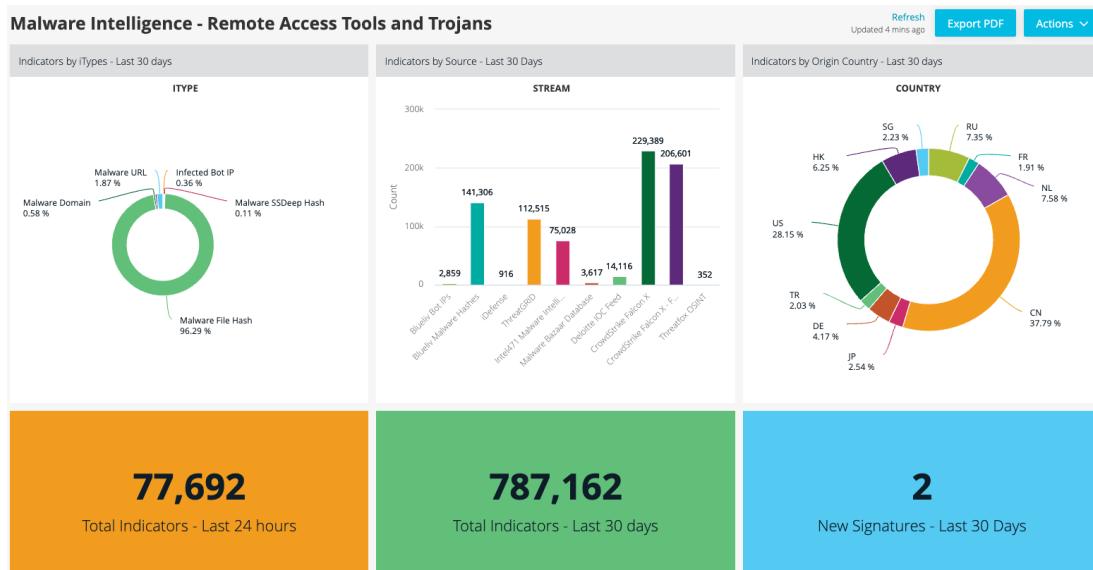
Şekil 3.7. Security Management Summary Dashboard by Carole Fennelly



Şekil 3.8. STMBugShield



Şekil 3.9. Mandiant Advantage



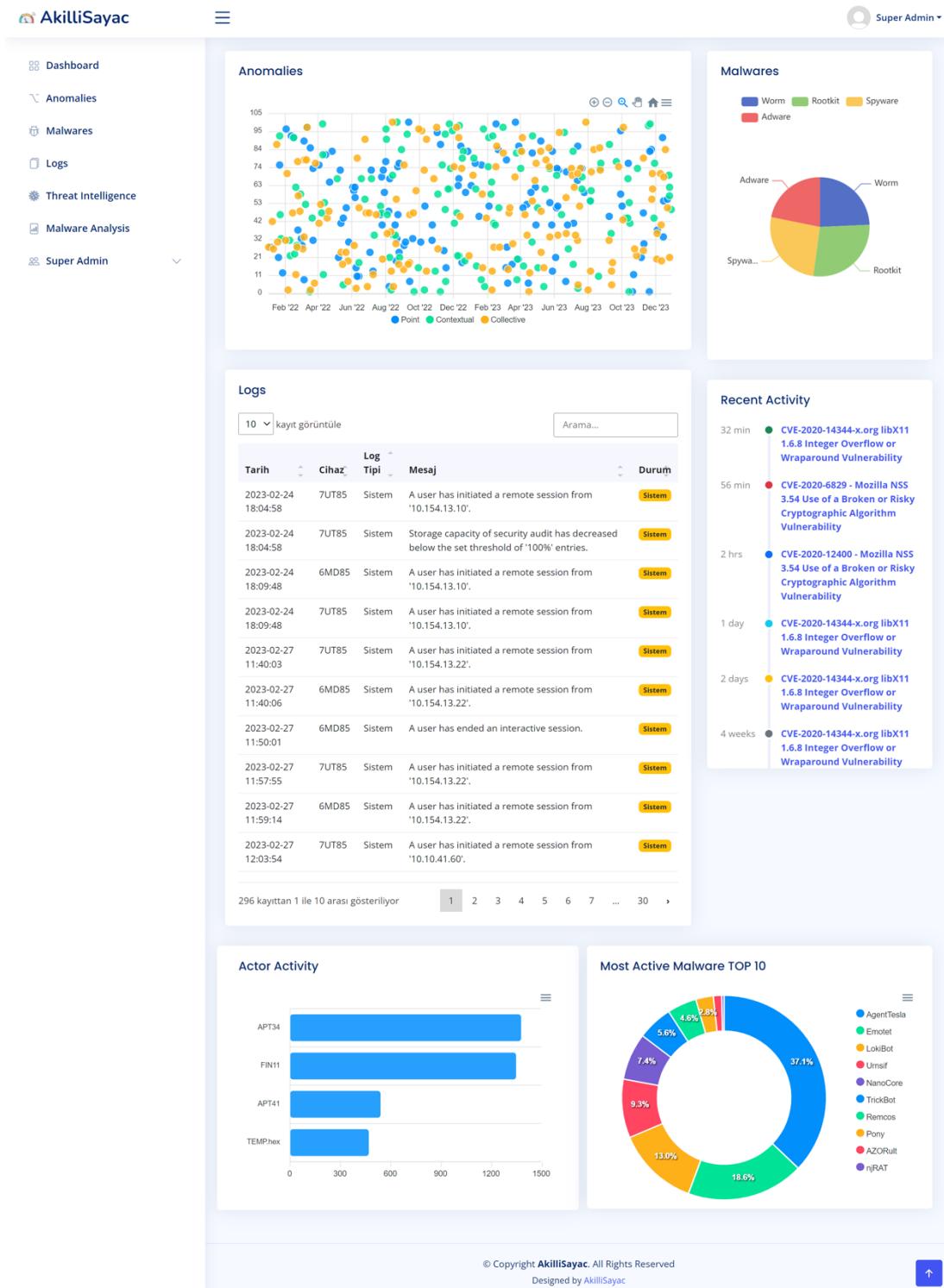
Şekil 3.10. Anomali

BÖLÜM 4. TASARIM

Bitirme çalışmamızda Güvenli Yazılım Geliştirme Kılavuzu, UTM Arayüzü ve entegre edilecek modüller ile ilgili gerekli araştırmaları tamamlayarak yaptığımız web uygulaması ile ilgili bilgi sahibi olduk ve çalışmayı gerçekleştirdik. Güvenli yazılım geliştirme standartlarına uygun olarak güvenli bir ASP.NET Core 7.0 kullanılarak UTM arayüzü sunulmuştur. Proje kapsamında Bootstrap Nice Admin HTML Şablonu [4] kullanılıp ASP.NET'e entegre edilerek geliştirilmesi sağlanmıştır.

4.1. Ana Sayfa Görünümü

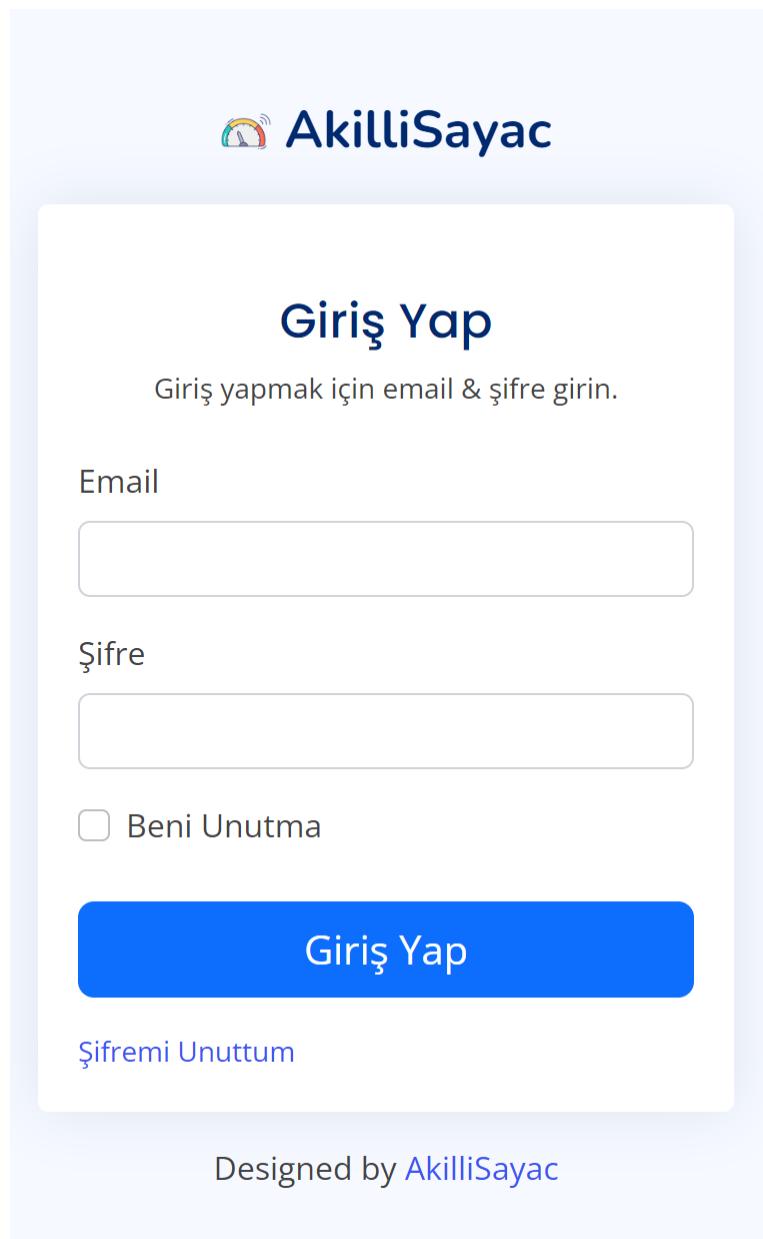
Kullanıcı girişi yapıldıktan sonra çalışmanın ana sayfa görünümü aşağıdaki gibi olup, modüllerin içerisinde bulunan bazı grafiklerini içermektedir. Modülle ilgili daha fazla grafiğe modülün ilgili sayfasından erişilmektedir. Modüllerin grafik içerikleri ilgili modülün başlığı altında inceleneciktir.



Şekil 4.1. Ana Sayfa Görünümü

4.2. Giriş

Web uygulamasına giriş yapıldığında kullanıcının ilk karşılaşacağı ekran giriş ekranıdır. Giriş yapılmadan herhangi bir sayfaya erişim mümkün değildir.



Şekil 4.2. Giriş Ekranı

Kullanıcının 5 defa yanlış giriş denemesinde bulunduğu durumda hesap otomatik olarak inaktif duruma geçmekte ve kapanmaktadır.

```
builder.Services.AddIdentity<AkilliSayacUser, IdentityRole>(options =>
{
    options.SignIn.RequireConfirmedAccount = false;
    options.SignIn.RequireConfirmedEmail = false;
    options.Lockout.MaxFailedAccessAttempts = 5;
    options.Lockout.AllowedForNewUsers = true;
    options.Lockout.DefaultLockout TimeSpan.FromDays(1000);
})
```

Şekil 4.3. Maksimum Giriş Denemesi Ayarı Kodu



Şekil 4.4. Hesap Kapatıldı Sayfası

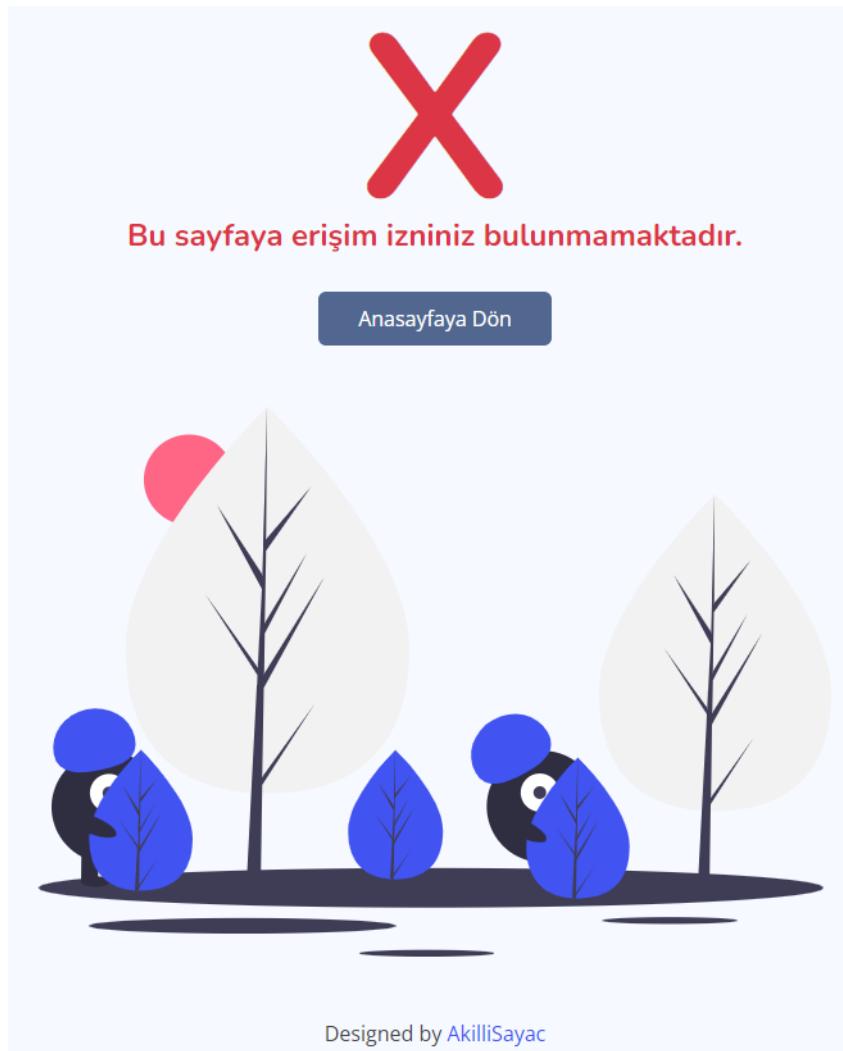
4.2.1. Authorization

Uygulamayı açan kişinin giriş yapmadan herhangi bir sayfaya erişiminin sağlanamaması için authorization gereği duyulmuştur ve Program.cs dosyası içerisinde gerekli tanımlama ve ayrıca controllerlar içerisinde Data Annotations ile authorize işlemleri yapılmıştır.

```
app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Home}/{action=Index}/{id?}")
    .RequireAuthorization();
```

Şekil 4.5. Authorization Require Kodu

Buna ek olarak gerekli role sahip olmayıp yetkilendirilmemiş bir sayfaya erişmeye çalışan kullanıcılar erişim engeli sayfasıyla karşılaşmaktadır.



Şekil 4.6. Erişim Reddedildi Sayfası

4.2.2. Authentication

Yeni kaydı oluşturulmuş ve ilk kez giriş yapan kullanıcıların güvenlik açısından iki adımlı kimlik doğrulamanın etkinleştirilmesi ve kaldırılamaması zorunlu kılınmıştır. Bu işlem de Program.cs içerisindeki işlemlerin yapıldığını sınıfın fonksiyonunun middleware olarak eklenmesi ile sağlanmıştır.

```

public class AuthenticatorResourceFilter2fa : IAsyncResourceFilter
{
    private readonly UserManager<AKilliSayacUser> _userManager;
    private readonly IUrlHelperFactory _urlHelperFactory;

    0 references
    public AuthenticatorResourceFilter2fa(UserManager<AKilliSayacUser> userManager, IUrlHelperFactory urlHelperFactory)
    {
        _userManager = userManager;
        _urlHelperFactory = urlHelperFactory;
    }

    0 references
    public async Task OnResourceExecutionAsync(ResourceExecutingContext context, ResourceExecutionDelegate next)
    {
        var urlHelper = _urlHelperFactory.GetUrlHelper(context);
        var redirectUrl = urlHelper.Content("~/Identity/Account/Manage/EnableAuthenticator");
        var logoutUrl = urlHelper.Content("~/Identity/Account/Logout");
        var currentUrl = context.HttpContext.Request.Path;

        if (redirectUrl != currentUrl && currentUrl != logoutUrl)
        {
            var user = await _userManager.GetUserAsync(context.HttpContext.User);
            if(user!= null)
            {
                if (user.TwoFactorEnabled == false)
                {
                    context.Result = new RedirectResult(redirectUrl);
                    return;
                }
            }
        }
    }

    await next();
}

```

Şekil 4.7 Authenticator Resource Filter

```

        {
            context.Result = new RedirectResult(redirectUrl);
            return;
        }
    }

    await next();
}

```

Şekil 4.8. Authenticator Resource Filter Devamı

```

builder.Services.AddMvc(o =>
{
    o.Filters.Add(typeof(ChangePasswordResourceFilter));
    o.Filters.Add(typeof(AuthenticatorResourceFilter2fa));
});

```

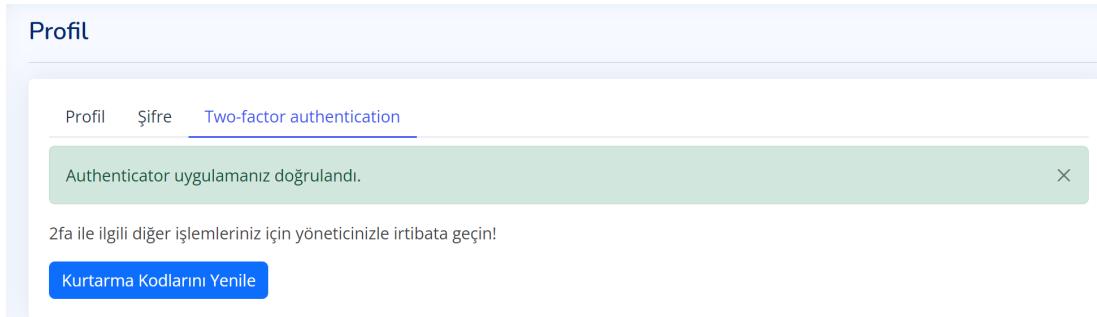
Şekil 4.9. Authenticator Resource Filter Etkinleştirme

Kullanıcı iki adımlı doğrulama kurulum sayfasına yönlendirilmekte ve buradan herhangi bir sayfaya kurulumu yapmadan geçiş yapamamaktadır.



Şekil 4.10. Authenticator Kurulum Sayfası

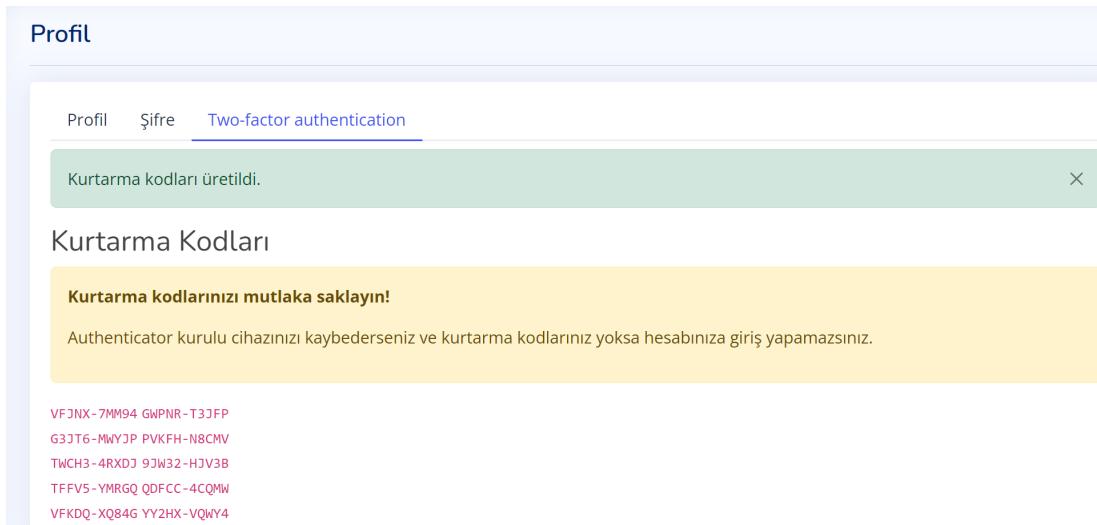
Kurulum başarıyla gerçekleştirilirse profil sayfasına yönlendirilmektedir ve buradan authenticator kurulmuş cihaza erişim olmaması durumunda kullanabilecegi kurtarma kodlarını alabileceği ekran gelmektedir.



Şekil 4.11. Authenticator Kurulumu Gerçekleştirdi Sayfası



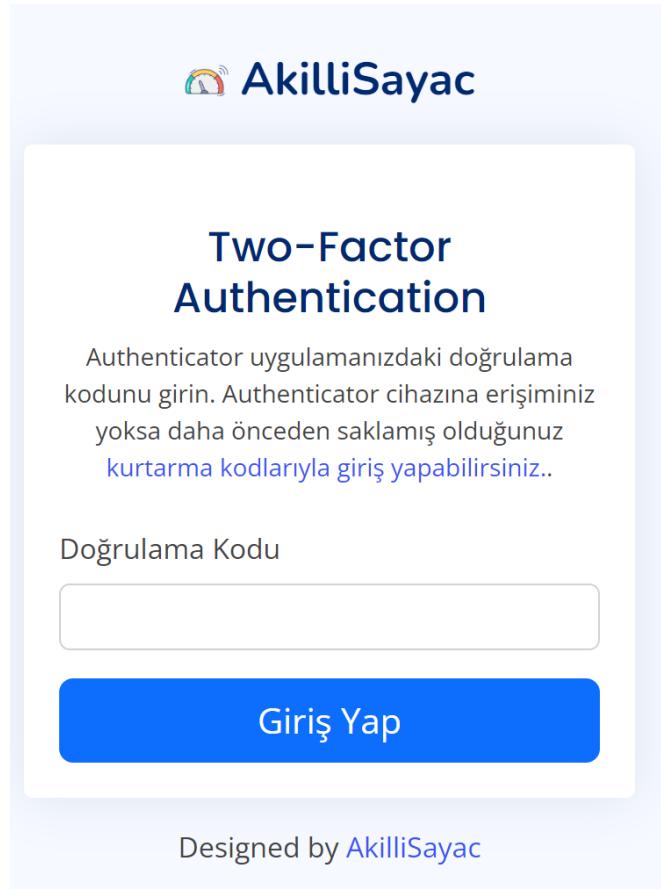
Şekil 4.12. Authenticator Kurtarma Kodu Üretme Sayfası



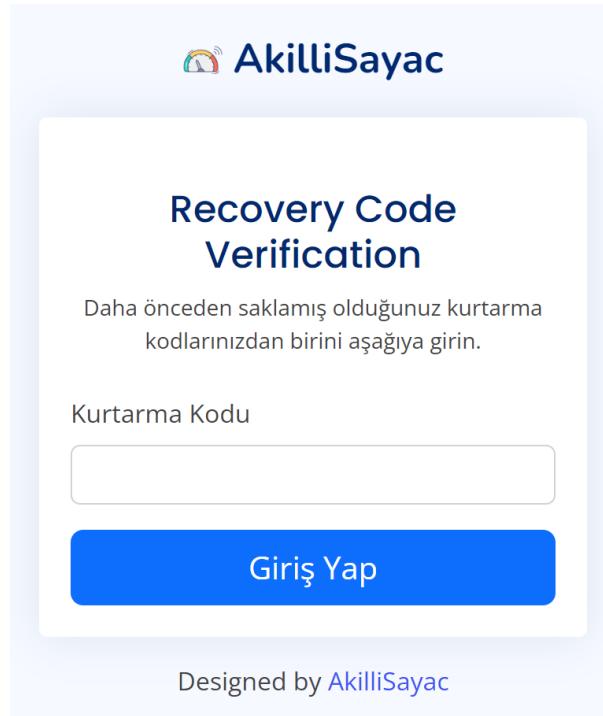
Şekil 4.13. Authenticator Kurtarma Kodları Sayfası

İki adımlı kimlik doğrulama aktifleştirildiğinde artık kullanıcı tekrar giriş yaparken ancak uygulamadaki doğrulama koduyla veya almış olduğu kurtarma kodları ile

bunu sağlayabilmektedir. E-posta ve şifre girişi sonrası direkt olarak iki adımlı kimlik doğrulama sayfası gelmektedir.

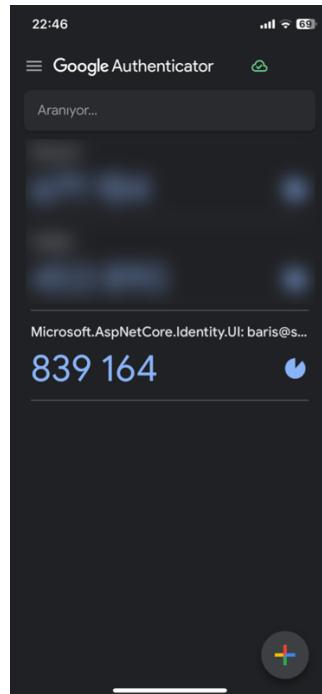


Şekil 4.14. 2fa Doğrulama Kodu



Şekil 4.15. 2fa Kurtarma Kodu

Örneğin Google Authenticator [5] uygulamasından kurulum yapıldığı varsayılırsa, doğrulama kodu sürekli yenilenerek gözükmektedir.



Şekil 4.16. Google Authenticator Doğrulama Kodu

4.3. Kullanıcı

Çalışma içerisinde ASP.NET'in sağlamış olduğu Identity özellikleri Scaffold işlemi ile entegre edilmiştir ve kullanıcı için gerekli olan diğer özellikler IdentityUser sınıfından kalıtım alınarak AkilliSayacUser sınıfı içerisinde eklenmiştir.

```
public class AkilliSayacUser : IdentityUser
{
    [PersonalData]
    public string? FirstName { get; set; }
    [PersonalData]
    public string? LastName { get; set; }

    4 references
    public string? RoleName { get; set; }
    [Required]
    [DataType(DataType.Date)]
    6 references
    public DateTime LastPasswordChangedDate { get; set; }
    [Required]
    [DataType(DataType.Date)]
    4 references
    public DateTime LastLoginDate { get; set; }
}
```

Şekil 4.17. AkilliSayacUser Sınıfı

Son giriş tarihi kullanıcı loglarında ve kullanıcı giriş yaptıktan sonra sağ üstteki kullanıcı panelinde görülmektedir.



Şekil 4.18. Kullanıcı Paneli

4.3.1. Parola Geçerlilik Süresi

Kullanıcıların parola geçerlilik süresi güvenlik sebebiyle 90 gün olarak ayarlanmıştır. Bu işlem de Program.cs içerisindeki gerekli işlemlerin yapıldığı sınıfın fonksiyonunun middleware olarak eklenmesi ile sağlanmıştır. Bu sayede son parola değişikliği 90 günden daha fazla olan kullanıcılar otomatik olarak şifre değiştirme sayfasına yönlendiriliyor ve değişikliği yapmadan herhangi bir işlem yapamıyorlar.

```
public class ChangePasswordResourceFilter : IAsyncResourceFilter
{
    private readonly UserManager<AkilliSayacUser> _userManager;
    private readonly IUrlHelperFactory _urlHelperFactory;

    public ChangePasswordResourceFilter(UserManager<AkilliSayacUser> userManager, IUrlHelperFactory urlHelperFactory)
    {
        _userManager = userManager;
        _urlHelperFactory = urlHelperFactory;
    }

    public async Task OnResourceExecutionAsync(ResourceExecutingContext context, ResourceExecutionDelegate next)
    {
        var urlHelper = _urlHelperFactory.GetUrlHelper(context);
        var redirectUrl = urlHelper.Content("~/Identity/Account/Manage/ChangePassword");
        var logoutUrl = urlHelper.Content("~/Identity/Account/Logout");
        var currentUrl = context.HttpContext.Request.Path;

        if (redirectUrl != currentUrl && currentUrl != logoutUrl)
        {
            var user = await _userManager.GetUserAsync(context.HttpContext.User);
            if(user!= null)
            {
                if (user.LastPasswordChangedDate.AddDays(90) < DateTime.Now)
                {
                    context.Result = new RedirectResult(redirectUrl);
                    return;
                }
            }
        }
        await next();
    }
}
```

Şekil 4.19. Password Resource Filter

```
        context.Result = new RedirectResult(redirectUrl);
        return;
    }
}

await next();
}
```

Şekil 4.20. Password Resource Filter Devamı

```
builder.Services.AddScoped<ChangePasswordResourceFilter>();

builder.Services.AddMvc(o =>
{
    o.Filters.Add(typeof(ChangePasswordResourceFilter));
});
```

Şekil 4.21. Password Resource Filter Etkinleştirme

4.3.2. Oturum Geçerlilik Süresi

Kullanıcıların oturum geçerlilik süresi güvenlik sebebiyle 5 dakika olarak ayarlanmıştır. 5 dakika boyunca inaktif olan kullanıcının oturumu sonlandırılmakta, auth cookie kendini imha etmekte ve kullanıcının tekrar giriş yapması gerekmektedir. Cookie'nin MaxAge'i de belirlediğimiz zaman aşımı süresi kadar olacak şekilde düzenlenmiştir ki oturumdan atıp cookie hala aktif durumda bulunup güvenlik zaafiyetine sebep olamasın. Bunun haricinde HttpOnly bayrağı da set edilmiştir.

```
builder.Services.ConfigureApplicationCookie(options =>
{
    options.Cookie.Name = "AspNetCore.Identity.Application";
    options.ExpireTimeSpan = TimeSpan.FromMinutes(5);
    options.Cookie.MaxAge = options.ExpireTimeSpan;
    options SlidingExpiration = true;
    options.Cookie.HttpOnly = true;
});
```

Şekil 4.22. Maksimum Oturum Süresi Ayarı Kodu

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly
IP_JAR	2023-03-23-15	.gstatic.com	/	2023-04-22T15:25:07.274Z	19	
AspNetCore.Identity.Application	CFD8NZDUVRSDLhGwvDBb0cY2qcOsGu7nVbIvL8_kQwK-9PaZVEUulfOp3tmeDDdpM7...	localhost	/	2023-04-01T15:34:33.410Z	890	✓
AspNetCore.Antiforgery.n-0DVdIMgro	CFD8NZDUVRSDLhGwvDBb0cY2gdX1lgYScc8ld--aRjXpOISFSHMUTB1b-WiOeOvTGEd...	localhost	/Session		190	✓

Şekil 4.23. Maksimum Oturum Süresi Cookie Görüntüsü

4.4. Rol

Proje içerisinde 2 adet yetki tipi bulunmaktadır.

4.4.1. Admin

Admin rolü yalnızca modül sayfalarına erişebilmekte ve grafikleri görüntüleyebilmektedir. Kullanıcılarla ilgili işlemler gibi yetki gerektiren sayfalara erişimi bulunmamaktadır.

4.4.2. Süper Admin

Süper Admin rolüne sahip bir kullanıcı sisteme giriş yaptığımda sayfanın sol panelinde kendisine ait bir bar daha açılmaktadır. Bu sayfalara Admin rolündeki kullanıcılar erişememektedir.



Şekil 4.24. Süper Admin Barı

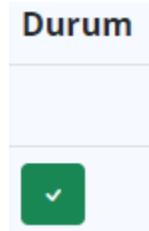
4.4.2.1. Kullanıcı Listesi

Süper Admin rolü sisteme kayıtlı kullanıcılarla ilgili işlemler gerçekleştirebilmektedir. Bu sayfa üzerinden var olan kullanıcının hesap durumunu aktif veya inaktif etme ve kullanıcı silme işlemleri yapılmaktadır.

Kullanıcı Listesi					
Durum	İsim	Soyisim	Email	Kullanıcı Rolü	Son Giriş Tarihi
Super	Admin		baris@sau.edu.tr	SuperAdmin	06.06.2023 22:32:26
✓	baris	yilmaz	baris1@sau.edu.tr	Admin	02.06.2023 18:47:47
 Yeni Kayıt Oluştur					

Şekil 4.25. Kullanıcı Listesi Sayfası

Hesap durumu aktif ise yeşil, inaktif ise kırmızı görülmektedir. Butona basıldığında kullanıcının statüsü değişir.



Şekil 4.26. Hesap Durumu Aktif Butonu



Şekil 4.27. Hesap Durumu İnaktif Butonu

Bunlara ek olarak yeni bir kullanıcı oluşturma işlemi de yalnızca Süper Admin rolüne sahip kullanıcılar tarafından yapılmaktadır.



Kayıt Oluştur

Üye kaydı oluşturuldu.

İsim

Soyisim

Email

Şifre

Şifreyi Doğrula

Rol

Kayıt Oluştur

Designed by [AkilliSayac](#)

Şekil 4.28. Kullanıcı Kayıt Sayfası

4.4.2.2. Veritabanı İşlemleri

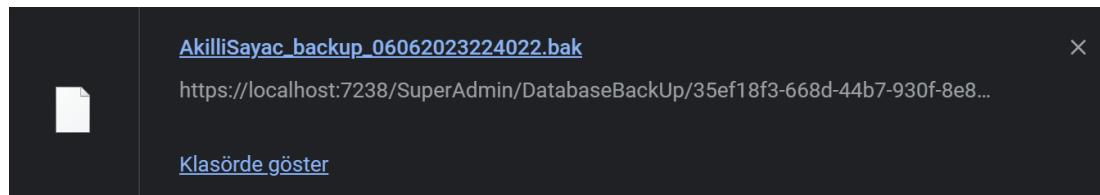
Süper Admin rolüne sahip kullanıcılar veritabanı yedeğini indirebilmektedir. İndiren kullanıcının bilgileri ve tarih raporlanmaktadır.

Veritabanı İşlemleri	
Oluşturan Kişi	Tarih
baris@sau.edu.tr	18-05-2023 01:19:56
baris@sau.edu.tr	18-05-2023 01:31:39
baris@sau.edu.tr	18-05-2023 01:49:26
baris@sau.edu.tr	02-06-2023 22:14:51

[!\[\]\(03756646c5f31ecb5e60d8bb445ffe36_img.jpg\) Veritabanı Yedeği İndir](#)

Şekil 4.29. Veritabanı İşlemleri Sayfası

Yedek indirildiği durumda tarihle beraber bak dosyası inmektedir.



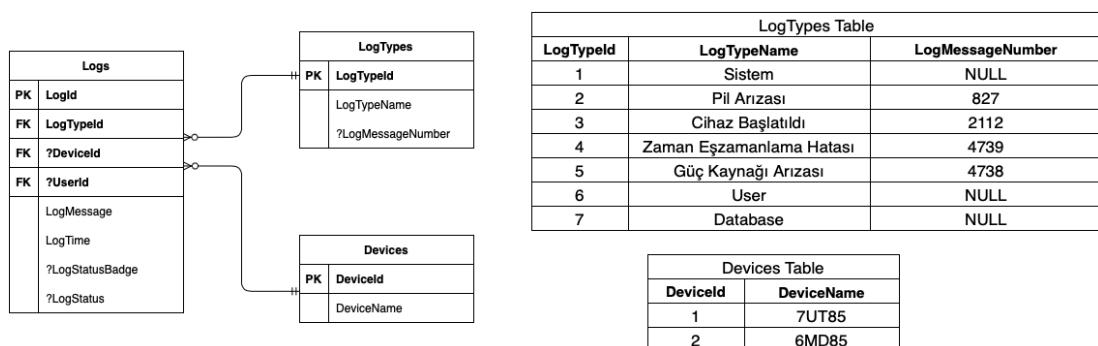
Şekil 4.30. Veritabanı Yedeği Dosyası

BÖLÜM 5. MODÜLLER

UTM arayüzünde kontrolleri yapılan ve izlenen modüller ile alakalı bilgiler ve projedeki tasarımları mevcuttur.

5.1. Log Yönetimi

Sunucular, güvenlik duvarları ve diğer BT ekipmanları dahil olmak üzere şirket sistemlerinde ve ağlarında meydana gelen tüm olayları ayrıntılandıran dosyalardır. Günümüzde güvenlik noktasında önemi artan konuların başında gelmektedir. Log Yönetimi hem yasal zorunluluklar nedeni ile hem de uluslar arası standartlar tarafından (ISO 27001 [6]) önerilmektedir. Projemizde cihaz ve kullanıcı logları olarak 2 log türü bulunmaktadır. İlgili modulün ERD modelini de oluşturduk.



Şekil 5.1. Log Yönetimi ERD

5.1.1. Cihaz Logları

Sahadan alınmış Syslog verileri (.txt) projeye import edilerek tablo üzerinde gösterimi gerçekleştirilmiştir. Bir log dosyasında cihazlara ait sistem logları bulunmakta olup, diğerinde ise bir cihaza ait durum logları bulunmaktadır. Bu dosyalardan loglar çekilerek veri tabanına aktarılmış ve oradan da tablo gösterimleri sağlanmıştır. Uygulama ilk açıldığında Program.cs içerisinde eklenen kodla ilgili parse işlemlerini ve veri tabanına kayıt oluşturan sınıfın fonksiyonu çalışarak işlem

gerçekleştirilmektedir. Program.cs içerisinde eklenen ve açılısta çalışan bu fonksiyonlar diğer modüller için de geçerlidir.

```
using (var scope = app.Services.CreateScope())
{
    var userManager = scope.ServiceProvider.GetService(typeof(UserManager<AkilliSayacUser>));
    var roleManager = scope.ServiceProvider.GetService(typeof(RoleManager<IdentityRole>));
    var hostingEnvironment = scope.ServiceProvider.GetService(typeof(IWebHostEnvironment));
    var db = scope.ServiceProvider.GetService(typeof(ApplicationDbContext));

    ContextSeed.Seed((UserManager<AkilliSayacUser>)userManager, (RoleManager<IdentityRole>)roleManager, (ApplicationDbContext)db);
    LogOperation.GetLogsFromFile((ApplicationDbContext)db, (IWebHostEnvironment)hostingEnvironment);
    AnomalyOperation.GetLogsFromFile((ApplicationDbContext)db, (IWebHostEnvironment)hostingEnvironment);
    MalwareOperation.GetLogsFromFile((ApplicationDbContext)db, (IWebHostEnvironment)hostingEnvironment);
}
```

Şekil 5.2. Dosyaları İçe Aktarma ve Veritabanına Ekleme

Bunun haricinde veritabanında çalışma için gerekli olan log tipleri ve cihazlar otomatik olarak veritabanında oluşturulmaktadır.

```
private static void SeedLogTypes(ApplicationDbContext db)
{
    if (db.LogTypes.Count() == 0)
    {
        LogType logType = new LogType
        {
            LogTypeName = "Sistem",
            LogMessageNumber = null
        };
        db.LogTypes.Add(logType);

        LogType logType1 = new LogType
        {
            LogTypeName = "Pil Arızası",
            LogMessageNumber = 827
        };
        db.LogTypes.Add(logType1);

        LogType logType2 = new LogType
        {
            LogTypeName = "Cihaz Başlatıldı",
            LogMessageNumber = 2112
        };
        db.LogTypes.Add(logType2);

        LogType logType3 = new LogType
        {
            LogTypeName = "Zaman Eşzamanlama Hatası",
            LogMessageNumber = 4739
        };
    }
}
```

Şekil 5.3. Log Tiplerini Veritabanına Ekleme

```

        db.LogTypes.Add(logType3);

        LogType logType4 = new LogType
        {
            LogTypeName = "Güç Kaynağı Arızası",
            LogMessageNumber = 4738
        };
        db.LogTypes.Add(logType4);
        LogType logType5 = new LogType
        {
            LogTypeName = "User",
            LogMessageNumber = null
        };
        db.LogTypes.Add(logType5);
        LogType logType6 = new LogType
        {
            LogTypeName = "Database",
            LogMessageNumber = null
        };
        db.LogTypes.Add(logType6);
    }
    db.SaveChanges();
}

```

Şekil 5.4. Log Tiplerini Veritabanına Ekleme Devamı

```

private static void SeedDevices(ApplicationDbContext db)
{
    if (db.Devices.Count() == 0)
    {
        Device device = new Device
        {
            DeviceName = "7UT85"
        };
        db.Devices.Add(device);

        Device device1 = new Device
        {
            DeviceName = "6MD85"
        };
        db.Devices.Add(device1);
    }
    db.SaveChanges();
}

```

Şekil 5.5. Cihazları Veritabanına Ekleme

Logların aktarımı sonrasında tablo üzerinde gösterimi controllerlar üzerinden databaseden çekilerek ViewData'lar ile gerekli bilgiler view'lere aktarılarak sağlanmaktadır. Tablo üzerinde her özelliğe göre sıralama yapılabilimekte, arama sayesinde örneğin bir log tipine ait logların adedi veya spesifik bir tarih araması ile ince görünlemeler yapılabilmektedir.

Cihaz Logları				
5 kayıt görüntüle	Arama...			
Tarih	Cihaz	Log Tipi	Mesaj	Durum
2023-02-24 18:04:58	7UT85	Sistem	A user has initiated a remote session from '10.154.13.10'.	Sistem
2023-02-24 18:04:58	7UT85	Sistem	Storage capacity of security audit has decreased below the set threshold of '100%' entries.	Sistem
2023-02-24 18:09:48	6MD85	Sistem	A user has initiated a remote session from '10.154.13.10'.	Sistem
2023-02-24 18:09:48	7UT85	Sistem	A user has initiated a remote session from '10.154.13.10'.	Sistem
2023-02-27 11:40:03	7UT85	Sistem	A user has initiated a remote session from '10.154.13.22'.	Sistem

Şekil 5.6. Cihaz Logları Tablosu

5.1.2. Kullanıcı Logları

Kullanıcıların yaptıkları işlemlerin loglarını yine veri tabanı üzerinde kaydolmakta ve tablo üzerinde gösterimi gerçekleştirmektedir. Yöneticinin herhangi bir kullanıcı üzerinde yaptığı işlemde ise Yönetici rozeti görülmektedir.

Kullanıcı Logları				
5 kayıt görüntüle	Arama...			
Tarih	Kullanıcı	Mesaj	Durum	
2023-05-18 01:13:30	baris@sau.edu.tr	Kullanıcı, başarılı giriş yaptı.	Başarılı	
2023-05-18 01:19:41	baris@sau.edu.tr	Kullanıcı, hatalı giriş denedi.	Uyarı	
2023-05-18 01:19:49	baris@sau.edu.tr	Kullanıcı, başarılı giriş yaptı.	Başarılı	
2023-05-18 01:26:30	baris@sau.edu.tr	Kullanıcı, başarılı giriş yaptı.	Başarılı	
2023-05-18 01:31:09	baris1@sau.edu.tr	Kullanıcının hesabı oluşturuldu.	Başarılı	

Şekil 5.7. Kullanıcı Logları Sayfası

5.2. Zararlı Yazılım

Zararlı yazılım, kullanıcıların programlanabilme özelliğine sahip cihazlarına, web sitelerine veya ağlarına zarar veren veya yetkisiz erişim sağlayan herhangi bir yazılımdır. Siber suçlular genellikle bunu, mali kazanç içi kurbanlardan veri elde ederek baskı yapmak üzere kullanır [7]. Projemizde 4 adet zararlı yazılım türüne ait veriler bulunmaktadır ve bu verilere ait sahadan alınmış herhangi bir dosya elimizde

bulunmadığından dolayı ChatGPT'den [8] sahadan alınmış olabilecek zararlı yazılım log mesajlarına benzer her tip için 4'er mesaj taslağı alarak C++ ile mini bir program yazarak sahadan alınmış logların içinde random tip ve mesajlarla doldurarak benzer, 400'er adet log kaydı bulunan log dosyaları oluşturduk ve bunları log modülünde olduğu gibi projeye import ettik ve gerekli, oluşturulması gereken tipleri oluşturduk.

```
private static void SeedMalwareTypes(ApplicationDbContext db)
{
    if (db.MalwareTypes.Count() == 0)
    {
        MalwareType malwareType = new MalwareType
        {
            MalwareTypeName = "Worm"
        };
        db.MalwareTypes.Add(malwareType);

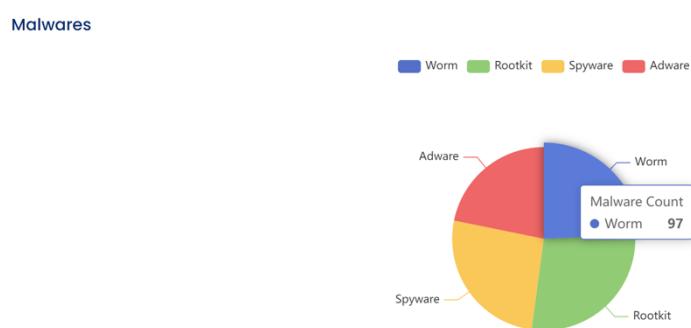
        MalwareType malwareType1 = new MalwareType
        {
            MalwareTypeName = "Rootkit"
        };
        db.MalwareTypes.Add(malwareType1);

        MalwareType malwareType2 = new MalwareType
        {
            MalwareTypeName = "Spyware"
        };
        db.MalwareTypes.Add(malwareType2);

        MalwareType malwareType3 = new MalwareType
        {
            MalwareTypeName = "Adware"
        };
        db.MalwareTypes.Add(malwareType3);
    }
    db.SaveChanges();
}
```

Şekil 5.8. Zararlı Yazılım Tiplerini Veritabanına Ekleme

Logların aktarımı sonrasında grafik görünümünü oluşturduk ve daha da detaylı incelenmesi açısından loglar için bir tablo da ekledik.



Şekil 5.9. Zararlı Yazılım Grafiği

Malwares

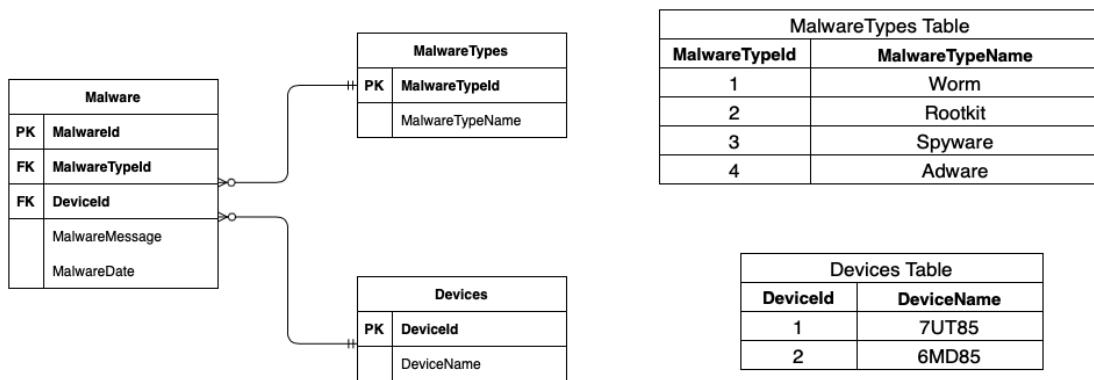
5 kayıt görüntüle	Arama...		
Tarih	Cihaz	Malware Tipi	Mesaj
2022-09-01 00:11:16	7UT85	Rootkit	Sayaç sistemine yerleşen bir rootkit, kullanıcı verilerini çalmaya çalışarak cihazların güvenliğini riske attı.
2023-05-02 03:30:14	6MD85	Worm	Sayaç sistemine bulaşan bir worm, kullanıcı verilerini çalmaya ve kötü amaçlar için kullanmaya çalıştı.
2023-10-26 19:08:28	7UT85	Rootkit	Sayaç sistemine yerleşen bir rootkit, kullanıcı verilerini çalmaya çalışarak cihazların güvenliğini riske attı.
2023-06-05 07:12:51	6MD85	Spyware	Sayaç kullanıcı verilerini toplayan bir spyware, kötü amaçlar için kullanılmak üzere harici bir sunucuya gönderdiği tespit edildi.
2022-09-22 16:55:01	6MD85	Adware	Sayaç kullanıcı arayüzünde istenmeyen reklam içerikleri gösteren bir adware tespit edildi ve kaldırıldı.

399 kayıttan 1 ile 5 arası gösteriliyor

1 2 3 4 5 6 7 ... 80 >

Şekil 5.10. Zararlı Yazılım Tablosu

İlgili modulün ERD modelini de oluşturduk.



Şekil 5.11. Zararlı Yazılım ERD

Projede bulunan zararlı yazılım tipleriyle ilgili bilgiler ise aşağıda bulunmaktadır.

5.2.1. Solucan (Worm)

Kendini bir ağa çoğaltmak ve diğer bilgisayarlara yaymak için tasarlanmış bir tür kötü amaçlı yazılımdır.

5.2.2. Casus Yazılım (Spyware)

Virüs bulaşmış bilgisayardan bilgi çalan ve bunları uzak bir konumdaki kötü amaçlı kişilere gönderen yazılımlardır.

5.2.3. Reklam Yazılımı (Adware)

İnternete bağlandığında reklamları görüntülemek için kullanıcının izni olmadan yüklenen yazılımlardır.

5.2.4. Rootkit

Virüs bulaşmış bir cihaza yönetim erişimi sağlayan ve onu farklı cihaz ve yazılımlara zarar vermek için kullanan kötü amaçlı yazılımdır.

5.3. Anomaly Detection

Anomaly Detection, en basit anlamıyla bir veride beklenmedik durumların veya kalıpların bulunmasını sağlayan bir tekniktir. Bu beklenmedik durumlara literatürde outliers (aykırı değerler), exceptions (istisnai durumlar) veya anomaliler denilmektedir. Anomali tespiti, izinsiz giriş tespiti, dolandırıcılık tespiti, arıza tespiti, sistem sağlığının izlenmesi, sensör ağlarında olay tespiti, ekosistem bozukluklarının tespiti ve makine görüşü kullanarak görüntülerde kusur tespiti gibi çeşitli alanlarda uygulanabilmektedir. Anomali tespiti için Makine Öğrenmesi Temelli Anomali Tespiti, K-Nearest Neighbour ve Density Based Algoritmaları, Kümeleme Tabanlı Algoritmalar, İstatistiksel Metodlarla Anomali Tespiti bulunmaktadır. Anomalilerin 3 adet tipi bulunmaktadır [9]. Bu tipteki verilere ait sahadan alınmış herhangi bir dosya elimizde bulunmadığından dolayı zararlı yazılımda olduğu gibi 400'er adet log kaydı bulunan log dosyaları oluşturduk ve bunları log modülünde olduğu gibi projeye import ettik ve gerekli, oluşturulması gereken tipleri oluşturduk.

```

private static void SeedAnomalyTypes(ApplicationDbContext db)
{
    if (db.AnomalyTypes.Count() == 0)
    {
        AnomalyType anomalyType = new AnomalyType
        {
            AnomalyTypeName = "Point"
        };
        db.AnomalyTypes.Add(anomalyType);

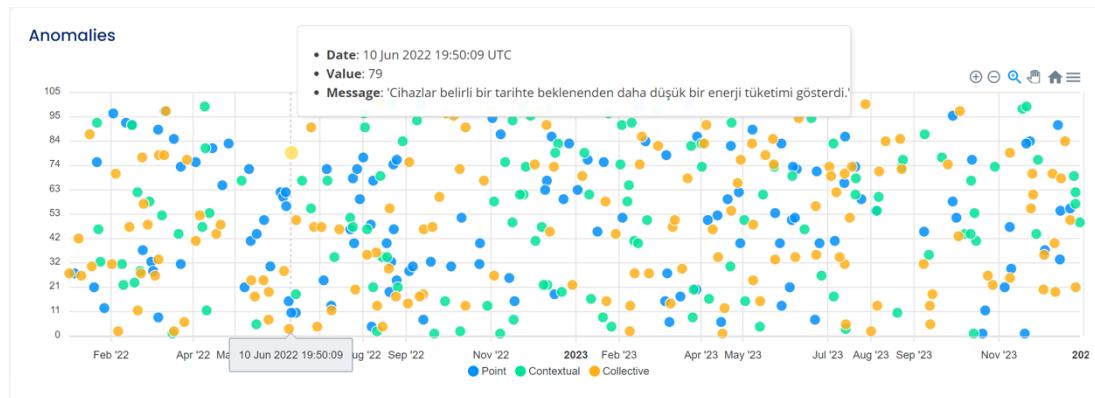
        AnomalyType anomalyType1 = new AnomalyType
        {
            AnomalyTypeName = "Contextual"
        };
        db.AnomalyTypes.Add(anomalyType1);

        AnomalyType anomalyType2 = new AnomalyType
        {
            AnomalyTypeName = "Collective"
        };
        db.AnomalyTypes.Add(anomalyType2);
    }
    db.SaveChanges();
}

```

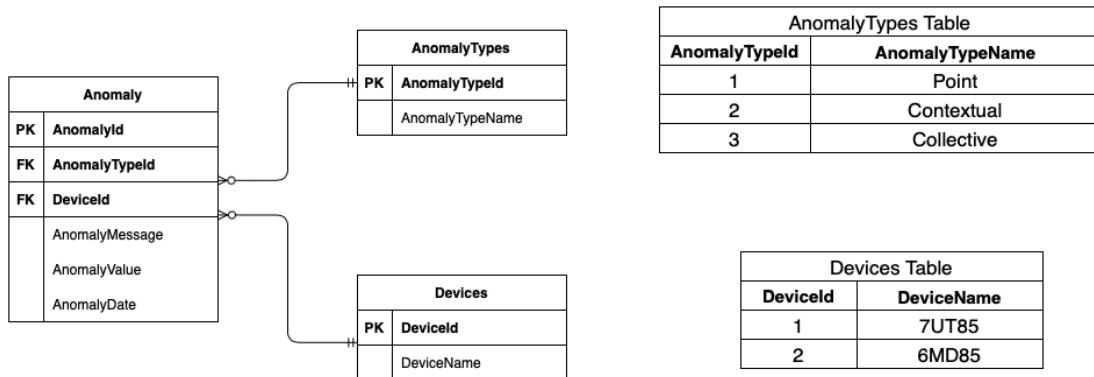
Şekil 5.12. Anomali Tiplerini Veritabanına Ekleme

Logların aktarımı sonrasında grafiğe aktarımını gerçekleştirdik. Grafik üzerinde tarihsel yakınlaşma, uzaklaştırma gibi işlemler veya herhangi bir tipi veya tipleri kapatarak yalnızca diğer tipteki anomalileri görüntüleyebilme işlemleri yapılmaktadır. Bunların yanı sıra aktivitelerin yanı grafikteki baloncukların üzerine gelindiğinde ise anomali ile ilgili bilgiler verilmektedir.



Şekil 5.12. Anomali Grafiği

İlgili modulün ERD modelini de oluşturduk.



Şekil 5.13. Anomali ERD

Projede bulunan zararlı yazılım tipleriyle ilgili bilgiler ise aşağıda bulunmaktadır.

5.3.1. Point Anomalies

Bireysel bir veri örneği eğer diğer normal verilerden uzaktaysa bu bir anomali veridir. Burada bu anomali tespitine point Anomali denmesinin sebebi anomali tespitinin belli bir niteliğe (attribute) bağlı olmasıdır.

5.3.2. Contextual Anomalies

Belirli bir verimiz bazı durumlarda anomaliye işaret ederken diğer durumlarda normal bir veriye işaret ediyorsa, yani özel bir bağlamda (specific context) anomali davranış sergiliyorsa bu context anomaliye bir örnektir.

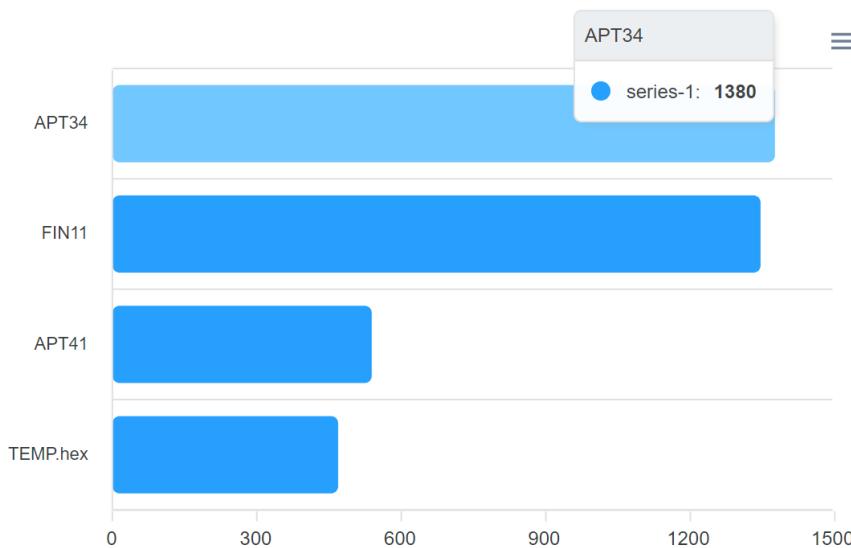
5.3.3. Collective Anomalies

Birbiriyile ilişkili olan veriler tüm verisetinde anomali davranış oluşturuyorsa bu bir collective anomaliye örnektir. Burada ilişkili olan bazı veriler bir araya geldiğinde anomali oluşturabilirken, bu veriler bireysel olarak verisetinde anomali davranış göstermiyor olabilir.

5.4. Tehdit İstihbaratı

Tehdit İstihbaratı, siber uzaydaki zararlı olayları azaltmaya yardımcı olan tehditler ve tehdit aktörleri hakkındaki bilgiler bütünlüğündür. Siber tehdit istihbaratı kaynakları arasında açık kaynak istihbaratı, sosyal medya istihbaratı, insan istihbaratı ve teknik istihbarat bulunur [10]. Projede 4 adet grafiği bulunmaktadır.

Actor Activity



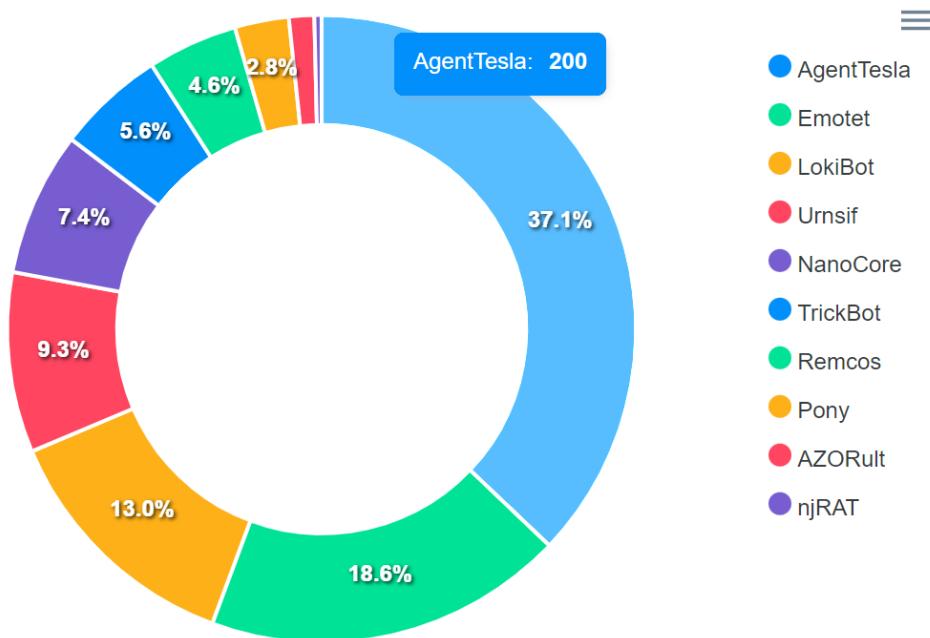
Şekil 5.14. Tehdit İstihbaratı Actor Activity

Recent Activity



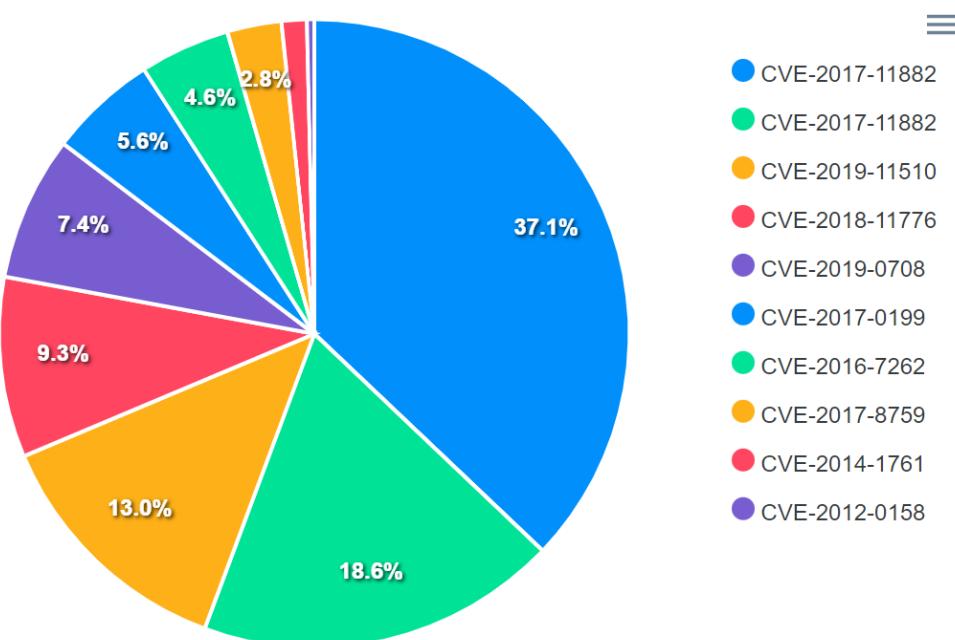
Şekil 5.15. Tehdit İstihbaratı Son Aktiviteler

Most Active Malware TOP 10



Şekil 5.16. Tehdit İstihbaratı En Aktif Zararlı Yazılımlar Top10

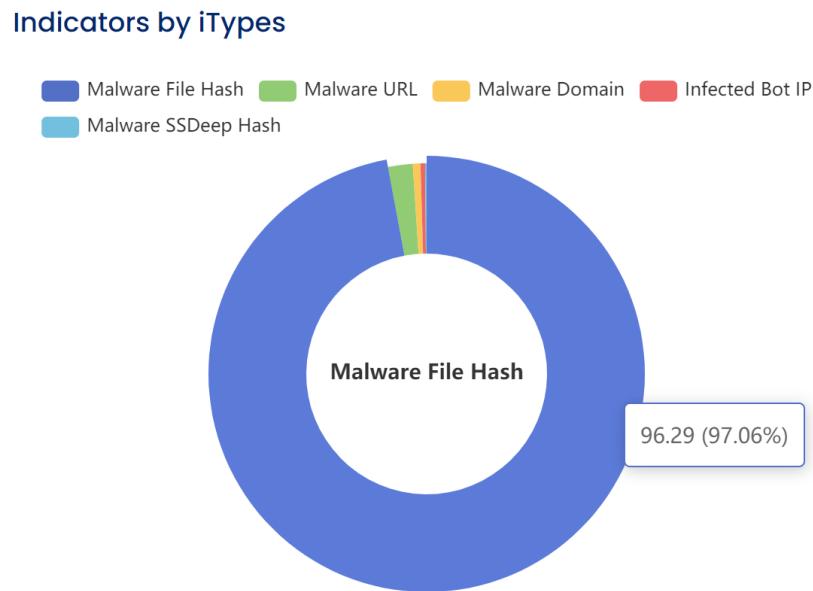
Most Active Vulnerabilities TOP 10



Şekil 5.17. Tehdit İstihbaratı En Aktif Güvenlik Açıkları Top10

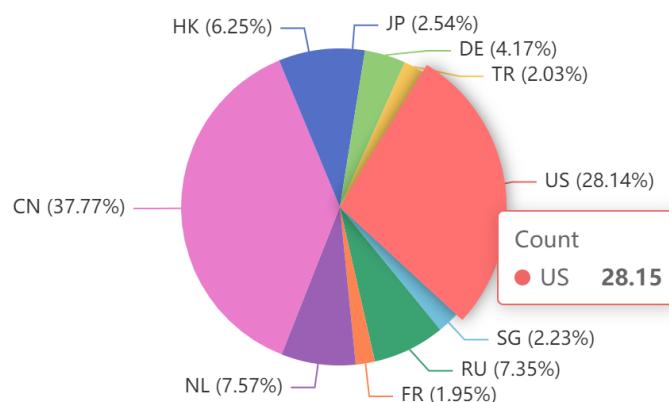
5.5. Zararlı Yazılım Analizi

Zararlı yazılım analizi, virüs, solucan, truva atı, rootkit veya arka kapı gibi belirli bir kötü amaçlı yazılım örneğinin işlevsellliğini, kaynağını ve potansiyel etkisini belirleme çalışması veya sürecidir [11]. Projede 3 adet grafiği bulunmaktadır.



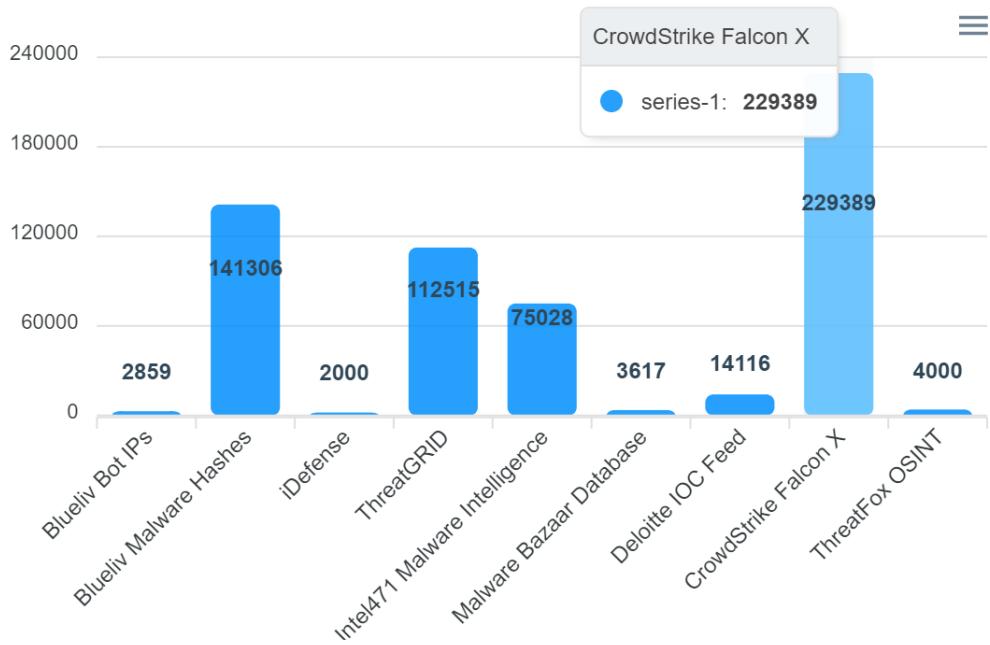
Şekil 5.18. Zararlı Yazılım Analizi Türe Göre Göstergeler

Indicators by Origin Country



Şekil 5.19. Zararlı Yazılım Analizi Ülkelere Göre Göstergeler

Indicators by Source



Şekil 5.20. Zararlı Yazılım Analizi Kaynağa Göre Göstergeler

BÖLÜM 6. SONUÇLAR VE ÖNERİLER

Bu çalışmanın sonucunda Güvenli Yazılım Geliştirme Standartları ve dikkat edilmesi gerekenler, UTM Arayüzü ile ilgili bilgiler ve örnek arayüzler, kullanılacak modüller ile ilgili bilgiler ve örnekleri incelenerek ve araştırılarak bilgi sahibi olunmuş, web uygulaması geliştirilerek tamamlanmıştır. Web uygulamasının gerçekleştirimi esnasında ve tamamlanana kadarki süreçte araştırmalardan faydalananarak güvenli bir uygulama yapmak amaçlanmış ve bunlar dikkate alınarak çalışma bitirilmiştir.

KAYNAKLAR

- [1] NIST Secure Software Development Framework
- [2] UTM Whitepaper, 2012.
<https://fedtechmagazine.com/sites/default/files/utm-whitepaper.pdf>
- [3] Gartner
<https://www.gartner.com/>
- [4] Nice Admin - Bootstrap HTML Template
<https://bootstrapmade.com/nice-admin-bootstrap-admin-html-template/>
- [5] Google Authenticator
<https://apps.apple.com/us/app/google-authenticator/id388497605>
- [6] ISO 27001 Bilgi Güvenliği Yönetim Sistemi
<https://belgelendirme.ctr.com.tr/iso-27001.html>
- [7] Zararlı Yazılım Nedir? Bilinen Zararlı Yazılımlar Nelerdir?, 2022.
<https://berqnet.com/blog/zararli-yazilim>
- [8] ChatGPT
<https://chat.openai.com>
- [9] Anomaly Detection (Anomali Tespiti) Nedir?, 2018.
<https://medium.com/yazilim-bilimi/anomaly-detection-anomali-tepiti-nedir-989a956df7a7>
- [10] Siber Tehdit İstihbaratı(Cyber Threat Intelligence) Nedir?, 2021.
<https://www.siberguvenlik.web.tr/index.php/2021/02/09/siber-tehdit-istihbaraticyber-threat-intelligence-nedir/>
- [11] Malware Analizi Nedir?, 2020.
<https://medium.com/@ishakusak/malware-analizi-nedir-193fffc7eaf9>

ÖZGEÇMİŞ

Barış Yılmaz, 29.01.1998'de İstanbul'da doğdu. İlk, orta ve lise eğitimini Sarıyer'de tamamladı. 2016 yılında Sarıyer Hüseyin Kalkavan Mesleki ve Teknik Anadolu Lisesi'nden mezun oldu. 2016 yılında İstanbul Topkapı Üniversitesi İnternet ve Ağ Teknolojileri Önlisans Bölümü'nü kazandı. 2017 yılında Ced Teknoloji Dan. Bilişim Destek Hiz. Ltd. Şirketinde network stajyeri olarak donanım stajını tamamladı ve 2018 yılında İnternet ve Ağ Teknolojileri bölümünden mezun oldu. 2019 yılında Sakarya Üniversitesi Bilgisayar Mühendisliği Lisans Bölümü'nü Dikey Geçiş Sınavı ile kazandı. 2023 yılında 4. sınıf öğrencisi olarak SAÜ Bilgisayar Mühendisliği Bölümü'nde eğitimini halen sürdürmektedir.

Cemal Aslan, 23.05.1999'da Giresun'da doğdu. İlk, orta ve lise eğitimini Gölcük'de tamamladı. 2017 yılında Gölcük İhsaniye Anadolu Lisesi'nden mezun oldu. 2018 yılında Sakarya Üniversitesi İnşaat Mühendisliği Bölümü'nü kazandı. 2019 yılında Merkezi Yatay Geçiş Sistemi ile Sakarya Üniversitesi Bilgisayar Mühendisliği Bölümü'ne geçiş yaptı. 2023 yılında 4. sınıf öğrencisi olarak SAÜ Bilgisayar Mühendisliği Bölümü'nde eğitimini halen sürdürmektedir.

BSM 498 BİTİRME ÇALIŞMASI
DEĞERLENDİRME VE SÖZLÜ SINAV TUTANAĞI

KONU: Akıllı Şebekeler İçin Güvenli Yazılım Geliştirme Temellerine Uygun Bir UTM
 Arayüz Tasarımı

ÖĞRENCİLER (Öğrenci No/AD/SOYAD):

G191210303 / BARIŞ / YILMAZ

G191210387 / CEMAL / ASLAN

Değerlendirme Konusu	İstenenler	Not Aralığı	Not
Yazılı Çalışma			
Çalışma kluvusa uygun olarak hazırlanmış mı?	x	0-5	
Teknik Yönden			
Problemin tanımı yapılmış mı?	x	0-5	
Geliştirilecek yazılımin/donanımın mimarisini içeren blok şeması (yazılımlar için veri akış şeması (dfd) da olabilir) çizilerek açıklanmış mı?			
Blok şemadaki birimler arasındaki bilgi akışına ait model/gösterim var mı?			
Yazılımin gereksinim listesi oluşturulmuş mu?			
Kullanılan/kullanılması düşünülen araçlar/teknolojiler anlatılmış mı?			
Donanımların programlanması/konfigürasyonu için yazılım gereksinimleri belirtilmiş mi?			
UML ile modelleme yapılmış mı?			
Veritabanları kullanılmış ise kavramsal model çıkarılmış mı? (Varlık ilişkili modeli, noSQL kavramsal modelleri v.b.)			
Projeye yönelik iş-zaman çizelgesi çıkarılarak maliyet analizi yapılmış mı?			
Donanım bileşenlerinin maliyet analizi (prototip-adetli seri üretim vb.) çıkarılmış mı?			
Donanım için gerekli enerji analizi (minimum-uyku-aktif-maksimum) yapılmış mı?			
Grup çalışmalarında grup üyelerinin görev tanımları verilmiş mi (iş-zaman çizelgesinde belirtilibilir)?			
Sürüm denetim sistemi (Version Control System; Git, Subversion v.s.) kullanılmış mı?			
Sistemin genel testi için uygulanan metodlar ve iyileştirme süreçlerinin dökümü verilmiş mi?			
Yazılımin sizme testi yapılmış mı?			
Performans testi yapılmış mı?			
Tasarımın uygulamasında ortaya çıkan uyumsuzluklar ve aksaklıklar belirtilerek çözüm yöntemleri tartışılmış mı?			
Yapılan işlerin zorluk derecesi?	x	0-25	
Sözlü Sınav			
Yapılan sunum başarılı mı?	x	0-5	
Soruları yanıtlama yetkinliği?	x	0-20	
Devam Durumu			
Öğrenci dönem içerisindeki raporlarını düzenli olarak hazırladı mı?	x	0-5	
Diğer Maddeler			
Toplam			

DANIŞMAN (JÜRİ ADINA): DR. ÖĞR. ÜYESİ MUSA BALTA

DANIŞMAN İMZASI: