T.C.
GEBZE TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ADAPTİF SIZMA TESTİ MODELİ

VOLKAN DÖRTKARDEŞ
YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
SİBER GÜVENLİK PROGRAMI

GEBZE
2020

T.C.
GEBZE TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ADAPTİF SIZMA TESTİ MODELİ

VOLKAN DÖRTKARDEŞ
YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
SİBER GÜVENLİK PROGRAMI

DANIŞMANI
PROF. DR. İBRAHİM SOĞUKPINAR

GEBZE
2020

T.R.

GEBZE TECHNICAL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

ADAPTIVE PENETRATION TESTING MODEL

VOLKAN DÖRTKARDEŞ A THESIS SUBMITTED FOR THE DEGREE OF MASTER OF SCIENCE DEPARTMENT OF COMPUTER ENGINEERING CYBER SECURITY PROGRAM

THESIS SUPERVISOR

PROF. DR. İBRAHİM SOĞUKPINAR

GEBZE

2020

GEBZE TEKNİK ÜNİVERSİTESİ	YÜKSEK LİSANS JÜRİ ONAY FORMU

GTÜ Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 26/02/2020 tarih ve 2020/13 sayılı kararıyla oluşturulan jüri tarafından 01/05/2020 tarihinde tez savunma sınavı yapılan VOLKANDÖRTKARDEŞ' in tez çalışması Bilgisayar Mühendisliği Anabilim Dalında YÜKSEK LİSANS tezi olarak kabul edilmiştir.

JÜRİ

ÜYE
(TEZ DANIŞMANI) :Prof. Dr. İbrahim SOĞUKPINAR

ÜYE :Doç. Dr. Mehmet GÖKTÜRK

ÜYE :Doç. Dr. A. Gökhan YAVUZ

ONAY

Gebze Teknik Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun
…..…/…..…/..….…tarih ve ……/……... sayılı kararı.

İMZA/MÜHÜR

ÖZET

Bu çalışmada Manuel ve Otomatize Sızma Testi Yöntemleri karşılaştırılarak Adaptif Sızma Testi Modeli incelenmiştir. Kaynak olarak kullanılmak üzere birbirinden farklı zafiyetlere sahip 7 ortam testlerin gerçekleştirilip sonuçların kıyaslanması amacıyla kullanılmıştır. Otomatize Sızma Testlerini gerçekleştirmek için OWASP 10'daki en önemli zafiyetler baz alınarak bir araç seti geliştirilmiştir. Bu tezde, birinci aşamada manuel sızma testi gerçekleştirilmiştir. İkinci aşamada ise otomatize sızma testi aracı kullanılarak test sonuçları incelenmiştir. Birinci aşamada manuel olarak belirlenen ortamlar üzerinde analizler yapılmış ve elde edilen sonuçlarda, sızma testinin belirli aşamalarında çok fazla zaman gerekliliği tespit edilmiştir. İkinci aşamada otomatize sızma testi aracı kullanılarak daha az zamanda daha fazla işlem gerçekleştirildiği ve sistemden daha fazla çıktı elde edildiği gözlemlenmiştir. Yapılan testlerin sonuçlarında elde edilen çıktılar, testlerin yapıldığı ortamların bulundurduğu zararlılar ile karşılaştırılmış ve tutarlı olduğu gözlenmiştir.

Anahtar Kelimeler: Zayıflık Analizi, Adaptif Sızma Testi, Otomatize Uygulama Güvenliği, Bilgi Güvenliği ve Etiği.

SUMMARY

In this study, adaptive penetration test model was examined by comparing manual and automated penetration test methods. 7 environments with different vulnerabilities for use as resources were used to perform tests and compare the results. To perform automated penetration tests, a toolkit has been developed based on the most important vulnerabilities in OWASP 10. In this thesis, a manual penetration test was carried out in the first phase. In the second phase, the test results were examined using automated penetration test tool. In the first phase, analyses were performed on manually determined environments and the results determined that there was a need for too much time in certain stages of penetration testing. In the second phase, it was observed that more operations were performed in less time using the automated penetration test tool and more output was obtained from the system. The results of the tests were compared and consistent with the pests contained in the environments in which the tests were conducted.

Key Words: Vulnerability Analysis, Adaptive Penetration Testing, Automated Application Security, Computer Security and Computer Ethics.

TEŞEKKÜR

Başta, yüksek lisans eğitimimde ve akademik hayatımda desteğini ve yardımlarını hiçbir zaman esirgemeyip bilgisi ile bu çalışmanın oluşmasının yolunu açan danışmanım Prof. Dr. İbrahim Soğukpınar'a,

Göstermiş oldukları desteklerinden dolayı sevgili aileme en içten teşekkürlerimi sunarım.

İÇİNDEKİLER

	Sayfa
ÖZET	v
SUMMARY	vi
TEŞEKKÜR	vii
İÇİNDEKİLER	viii
SİMGELER ve KISALTMALAR DİZİNİ	ix
ŞEKİLLER DİZİNİ	x
TABLOLAR DİZİNİ	xi
1. GİRİŞ	1
1.1. Tezin Amacı, Katkısı ve İçeriği	1
2. SIZMA TESTİ TEMELLERİ	3
2.1.Sızma Testi Yaklaşımları ve İlgili Çalışmaların Değerlendirmesi	3
2.2. Bilgi Güvenliği Çerçeveleri ve Standartları	9
2.2.1. OWASP	10
2.2.2. Bilgi Sistemleri Güvenlik Değerlendirme Çerçevesi (ISSAF)	10
2.2.3. Açık Kaynak Güvenlik Test Metodolojisi Kılavuzu (OSSTMM)	12
2.2.4. CREST Sızma Testi Kılavuzu	13
3. ADAPTİF SIZMA TESTİ YAKLAŞIMI	14
3.1. Adaptif Sızma Testi Aracının Mimarisi	14
3.1.1. Adaptif Sızma Testi Aracının Bileşenleri	16
3.1.2. Çalışmanın Matematiksel Modeli	22
3.1.2.1. CRAWLER Fonksiyonunun Kimlik Doğrulamasız Program Modeli	23
3.1.2.2. CRAWLER Fonksiyonunun Kimlik Doğrulamalı Program Modeli	25
4. GERÇEKLEME ve ÖZELLİKLERİ	26
4.1. Test Ortamları ve Özellikleri	27
4.2. Gerçekleştirilen Testler	28
4.3. Değerlendirme	45
5. SONUÇLAR ve YORUMLAR	46

KAYNAKLAR 47
ÖZGEÇMİŞ 49
EKLER 50

SİMGELER ve KISALTMALAR DİZİNİ

Simgeler ve Kısaltmalar		**Açıklamalar**
BYOD	:	Kendi cihazını getir
OWASP	:	Açık Web Uygulama Güvenliği Projesi
PT	:	Sızma Testi
NIST	:	Ulusal Standartlar ve Teknoloji Enstitüsü
XXS	:	Siteler arası komut dosyası oluşturma
XXE	:	XML harici varlığı
SSRF	:	Sunucu Tarafı Talep Sahteciliği
CORS	:	Kökler Arası Kaynak Paylaşımı
HTTP	:	Üst metin transfer protokolü
CSRF	:	Siteler arası istek sahteciliği
GTÜ	:	Gebze Teknik Üniversitesi

ŞEKİLLER DİZİNİ

Şekil No :		Sayfa
2.1:	Otomatize Sosyal mühendislik aracıyla kurbanların bulunması.	6
2.2:	Julian karakteri ile hedefin gerçek insanları arasındaki bir konuşmanın sonuçları.	7
2.3:	Anna karakteriyle hedef kullanıcılar için mesajlaşma sonuçları.	8
3.1:	Adaptif Sızma Testi Kalite Modeli.	15
3.2:	Adaptif Sızma Testi Kullanım Durumu Diyagramı.	16
3.3:	Adaptif Sızma Testi Aracının Sınıf Diyagramı.	17
3.4:	Adaptif Sızma Testi aracının Arayüzü.	19
3.5:	Kimlik Doğrulamasız Model.	24
3.6:	Kimlik Doğrulamalı Model.	25
4.1:	Manuel olarak CSS zayıflığının tespiti ve çalıştırılması ardından elde edilen ekran görüntüleri.	29
4.2:	Adım 1'deki komutların çalıştırılması ve sürecin başlangıcı.	30
4.3:	Otomatize Sızma Testi aracıyla CSS Zafiyeti tarama sonuçları.	31
4.4:	Manuel olarak SQL Enjeksiyonu Zafiyetinin zafiyetin tespiti ve sistemin tepkisi.	32
4.5:	Otomatize araç ile SQL Enjeksiyon zafiyetinin tespiti.	32
4.6:	Manuel Kayıp Güvenlik Başlığı Zafiyetinin Tespiti.	33
4.7:	Otomatize olarak Kayıp Güvenlik Başlığının Tespiti.	34
4.8:	Alt Alan adı devralma zafiyetinin manuel olarak tespiti.	35
4.9:	Alt Alan adı devralma zafiyetinin Otomatize olarak tespitinde elde edilen çıktılar.	36
4.10:	XXE Zafiyetinin Otomatik olarak tespitinde elde edilen çıktılar.	37
4.11:	Hassas Veri Sızıntısı zararlısının adaptif sızma testi aracında çalıştırılması ve sonucu.	38
4.12:	Manuel CORS zafiyetinin uygulanması ve çıktıları.	39
4.13:	Otomatize olarak CORS Zafiyeti tespiti.	40
4.14:	Hassas Veri Kaçağı zafiyetinin otomatize olarak tespiti ve çıktıları.	41
4.15:	CSS Zafiyetinin otomatize olarak tespiti.	42
4.16:	SQL Enjeksiyon zafiyetinin otomatize araç ile tespiti.	43

4.17: Hassas Veri Kaçağının otomatize araç ile tespiti. 43
4.18: Kayıp Güvenlik Bayrağı zafiyetinin otomatize araç yardımıyla tespiti. 44
4.19: Hassas Veri Kaçağı zafiyetinin otomatize araç ile tespiti ve elde edilen çıktılar. 44

TABLOLAR DİZİNİ

Tablo No :		Sayfa
4.1:	Manuel ve Otomatik Testlerin sonucunda tespit edilen zararlılar (TestphpVulnweb ortamı).	28
4.2:	Manuel ve Otomatik Testlerin sonucunda tespit edilen zararlılar (DemoTestfire).	35
4.3:	Manuel ve Otomatik Testlerin sonucunda tespit edilen zararlılar (ZeroWebappsecurity ortamı).	37
4.4	Manuel ve Otomatik Testlerin sonucunda tespit edilen zararlılar (Testhtml5Vulnweb ortamı).	39
4.5:	Manuel ve Otomatik Testlerin sonucunda tespit edilen zararlılar (BWAPP ortamı).	40
4.6:	Manuel ve Otomatik Testlerin sonucunda tespit edilen zararlılar (DVA ortamı).	42

1. GİRİŞ

Sistemlerin karmaşıklığı her geçen gün artmaktadır. Bu, sistemlerde daha fazla güvenlik açıklarına yol açar. Saldırganlar, hedefin sistemini kullanmak için bu güvenlik açıklarını kullanır. Saldırganın yapmadan önce bu güvenlik açıklarını önceden bulmak daha iyidir. Güvenlik açığı değerlendirmesinin gücü genellikle göz ardı edilir. Sızma testi, güvenlik açıklarını tanımlamak ve kullanmak için üstlenilen bir dizi faaliyettir. Uygulanan güvenlik önlemlerinin etkinliğini veya etkisizliğini doğrulamaya yardımcı olur. Mevcut sızma testi yaklaşımları ve yöntemleri incelendiğinde gözden kaçan bir diğer konu, tüm adımların sızma testi sürecinde gerektiği kadar yeterli bir şekilde gerçekleştirilmemesidir. Sızma testlerinin yapılacağı ortamlarda, saldırı vektörlerinin sayısı ancak Sızma testi uzmanının yeteneklerine bağlı olarak ilerleme kaydedilmesi ve / veya bazı önemli parçaların unutulması veya yetenek eksikliği nedeniyle eksik olması durumunda sınırlı olabilir. Bu yazıda, siber savunma teknolojisinde kapsamlı sızma testi yazılımının nasıl kullanılacağı, daha fazla saldırı vektörünün nasıl uygulanacağı ve daha fazla sonuç, testler sırasında yeni bir perspektifle nasıl elde edilebileceği ve otomatik bir uygulama aracı ile sosyal mühendislik zayıflıklarının tespitinin nasıl otomatikleştirileceği üzerine çalışmalarım birlikte yer almaktadır. Otomatik test sonuçları, 7 farklı ortamda yapılan manuel testlerin sonuçları ile karşılaştırıldı ve üstün parçalar sonuçlar bölümünde belirtilmiştir. Ayrıca hangi araçların hangi amaçlar için ve hangi aşamalarda kullanılacağı hakkında konuştum ve birkaç adım sızma testinin otomatize hale getirilmesi yer almaktadır.

1.1. Tezin Amacı, Katkısı ve İçeriği

Sızma Testleri günümüzde birçok alanda yaygın olarak kullanılmaktadır. Kurumların Bilgi Teknolojileri hakkında detaylı sonuçlar ve zafiyetler ortaya çıktığı için büyük önem taşımaktadır.

Sızma Testi Yöntemleri, bilgi güvenliği sistemlerindeki zayıflıklar hakkında sonuçlar çıkarmaya yarayan ve farklı yaklaşımları benimseyen yapılardır.

Sitemler üzerinde gerçekleştirilecek olan testlerin kapsamının çok fazla olması ve manuel olarak gerçekleştirildiğinde insan faktörüne çok fazla bağlı olması nitelikli ve yeterli sonuçların alınmasını zorlaştırmaktadır. Bu zorluğun üstesinden gelebilmek için çeşitli yöntemler geliştirilmiştir. Bu tez çalışmasında da kullanılacak olan Adaptif Sızma Testi yöntemi bu yöntemler arasında en fazla çıktı üreten ve zamandan en fazla tasarruf sağlayan yöntemdir.

Bu tez kapsamında, öncelikle ikinci bölümde Sızma Testi Yöntemlerinin türleri ve bu türlerin çeşitli problemler için yapılmış benzetim çalışmaları anlatılmıştır. Üçüncü bölümde, Adaptif Sızma Testi aracının mimarisi hakkında bilgiler verilmiş ve bileşenleri ile birlikte hangi zayıflıkları tespit etmeye yaradığı anlatılmıştır. Dördüncü kısımda ise manuel ve otomatik olarak gerçekleştirilen testler ve sonuçları ile bu ortamların özellikleri hakkında ve yöntemin uygulanması hakkında geniş bilgiler verilmiştir. Beşinci ve son bölümde ise elde edilen sonuçlar ile ilgili yorumlar ve gelecek önerileri konu başlıkları altında anlatılmıştır.

2. SIZMA TESTİ TEMELLERİ

Sızma testi, zayıflıklar için bir sistemin incelenmesini, bu zayıflıkların kötüye kullanımın etkisi olarak tanımlanmasını ve son olarak sistemin sahipleri için bir rapor hazırlanmasını içeren bir dizi faaliyettir. Sızma testinin yaşam döngüsü aşağıdaki adımlardan oluşur [2]:

- Kapsam
- Keşif
- Güvenlik açığı tespiti
- Bilgi analizi ve planlama
- Sızma testinin uygulanması
- Ayrıcalık yükseltme
- Sonuç analizi
- Raporlama
- Kalan izlerin temizlenmesi

2.1. Sızma Testi Yaklaşımları ve İlgili Çalışmaların Değerlendirilmesi

Sızma testi teknikleri üç tiptir ve her tipin kendine özgü özellikleri vardır.

Beyaz kutu test modelinde, testi yapan güvenlik uzmanı, test ağının ve test/sistem ağının ağ yapılandırması hakkında tam bilgiye sahiptir. Bu test genellikle dahili ağdan yapılır. Beyaz kutu testi, test ağının veya sisteminin derinlemesine anlaşılmasını gerektirir ve daha iyi sonuçlar verir.

Kara Kutu Test Modelinde, testi gerçekleştiren uzmanın ağ mimarisi veya test ağının sistemleri hakkında önceden bilgisi yoktur. Kara kutu testi harici ağlardan dahili ağlara kadar yapılır. Testi yapan kişi uzmanlığını ve becerisini kullanmalıdır.

Gri kutu Test Modelinde, bu testi gerçekleştiren kişinin ağ mimarisi hakkında ayrıntılı bilgisi yoktur, ancak ağ ve sistem yapılandırmasını test etme hakkında bazı bilgileri bilir. Aslında önceki iki yöntemin bir karışımı. Hem iç ağdan hem de dış

ağdan gerçekleştirilebilir. Bu testleri gerçekleştirirken, uygulama olarak aşağıdaki işlemler gerçekleştirilir:

i) Bilgi toplama,
ii) Ağ Haritalama,
iii) Güvenlik Açığı Taraması,
iv) "Performansı sömürme" sistemine giriş,
v) Yetkilendirme Yükseltmesi,
vi) Diğer Ağlara Giriş,
vii) Erişimi Korumak,
viii) Web Tabanlı Uygulamalara Yapılan Saldırılar,
ix) Sosyal Mühendislik Saldırıları,
x) Ayak İzlerini Temizleme,
xi) Raporlama.

Statik analiz, uygulama kodunda güvenlik hatalarını bulmak için kullanılan yaygın tekniklerden biridir. Sızma testinden farkı, bir sistemin kaynak kodunu analiz ederek ve güvenlik açıklarını belirleyerek beyaz bir kutuda çalışmasıdır. Her iki teknik de farklı özelliklerinden dolayı bulgularını farklı rapor etmektedir [3]. Mariano ve Riccardo' nun çalışmasında, statik analiz tabanlı bir güvenlik aracına dayanan bir rapor sızma testine dayanan bir raporla karşılaştırılmış ve raporlamanın nasıl daha yararlı olabileceği konusunda senaryoya dayalı çalışmalar yapılmıştır.

Sosyal mühendislik, sosyal topluluklar için ciddi bir tehdit olarak ortaya çıkmış ve bilgi sistemlerine saldırmak için etkili bir araçtır. Bugünün bilgi çalışanları tarafından kullanılan hizmetler, sofistike sosyal mühendislik saldırılarının önünü açıyor. BYOD (kendi cihazınızı getirin) politikalarına yönelik artan eğilim ve özel / iş ortamlarında çevrimiçi iletişim ve sosyal medya kullanımı sorunu daha da kötüleştiriyor. Küresel olarak faaliyet gösteren şirketlerde, ekipler artık coğrafi olarak birlikte çalışmıyor, ancak tam zamanlı çalışıyor. Kişisel etkileşimdeki azalmaya rağmen, iletişim için kullanılan çok sayıda aracın ortaya çıkması (e-posta, IM, Skype, Dropbox, LinkedIn, Lync, vb.) sosyal mühendislik için yeni saldırı vektörleri oluşturur. New York Times ve RSA gibi şirketlere yapılan son saldırılar, hedeflenen spearfishing saldırılarının sosyal mühendislik saldırılarında etkili ve

evrimsel bir adım olduğunu göstermiştir. [5] Katharina ve meslektaşları, bilgi işçisine yönelik gelişmiş sosyal mühendislik saldırılarının kapsamlı bir bakışının yanı sıra, sosyal mühendislik saldırılarının tanınmış bir taksonomisi sağlar.

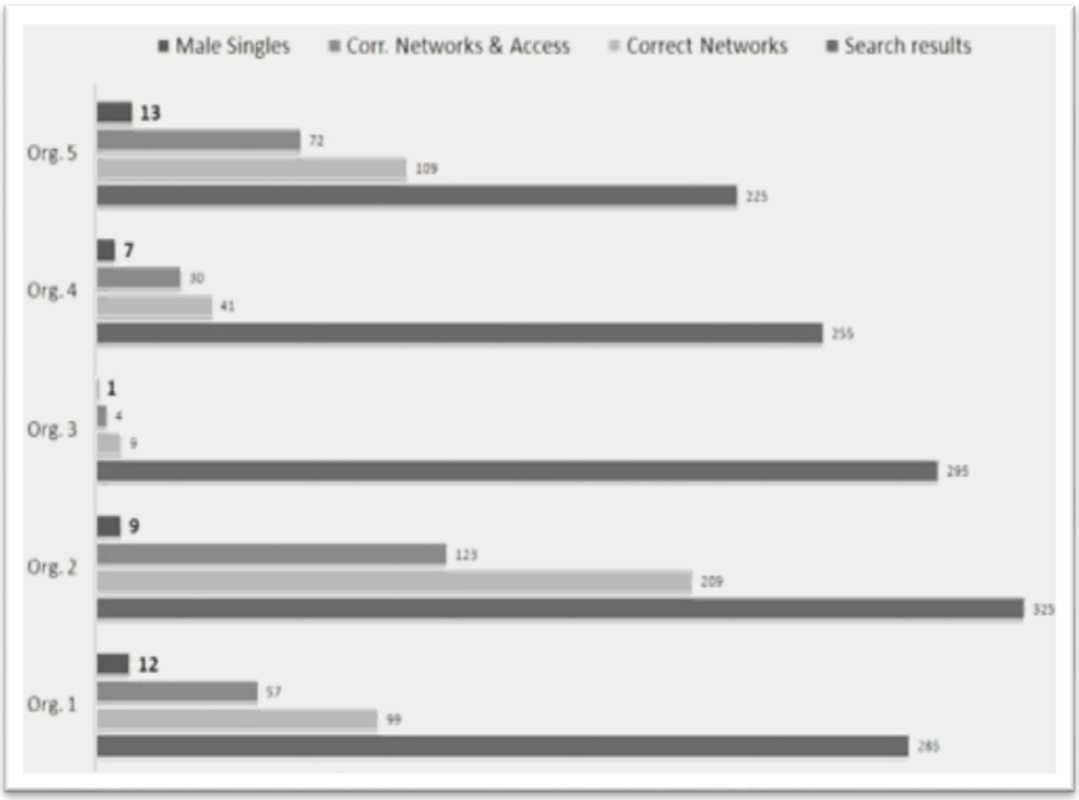
Bu çalışma, sızma test süreçlerini ve sistemlerini, özellikle de güvenlik açığı değerlendirmesi (güvenlik açığı analizi) ve Sızma testlerini otomatikleştirmeyi ve optimize etmeyi amaçlayan uzun bir dizi uygulamalı araştırma çalışmasından yararlandı. [13.15]. Bu konuya en önemli katkılar arasında, burada benimsenen yaklaşımlar ve katkılara odaklanan önceki araştırmanın bir özetini sunuyoruz. Başlangıçta, araştırmacılar planlama aşamasına ilgi duyuyorlardı. Bazı çalışmalar endüstriyel Sızma test sistemleri ve çerçevelerinde uygulanırken, Diğerleri araştırma fikirleri olarak kalmıştır [13,14].

İnsanlar sosyal ağ sitelerini birbirleriyle iletişim kurmak için kullanamasa da, sağlanan hizmetlerin bir avantaj olarak kabul edilmesine rağmen, aslında kullanıcının bilgilerinin gizliliği el konmakta ve sıklıkla ihmal edilmektedir [4]. Markus ve meslektaşlarının çalışmada ortaya koyduğu yaklaşım, klasik sosyal mühendisliğin Otomasyonu ile bir adım daha ileri götürüyor. Önerilen saldırı döngüsü ve prototip uygulamalarını (ase bot) değerlendirmek için iki deney gerçekleştirdiler. İlk girişimlerinde, teknelerinin bilgi toplama yeteneğini inceliyorlar. İkinci değerlendirmelerinde Turing testi yaptılar. Değerlendirmelerinin umut verici sonuçları, otomatik sosyal mühendislik botları kullanarak etkili ve etkili sosyal mühendislik saldırıları olasılığını vurgulamaktadır.

Markus ve meslektaşlarının çalışmalarında yarattıkları otomatik sosyal mühendislik aracıyla yaptıkları saldırılar çok çarpıcı. Birincisi, Sızma testi çalışmasını yürüttükleri beş başarılı büyük İsveç merkezli çok uluslu şirkette otomatik sosyal mühendislik çalışmasıdır. Bu kurumlar:

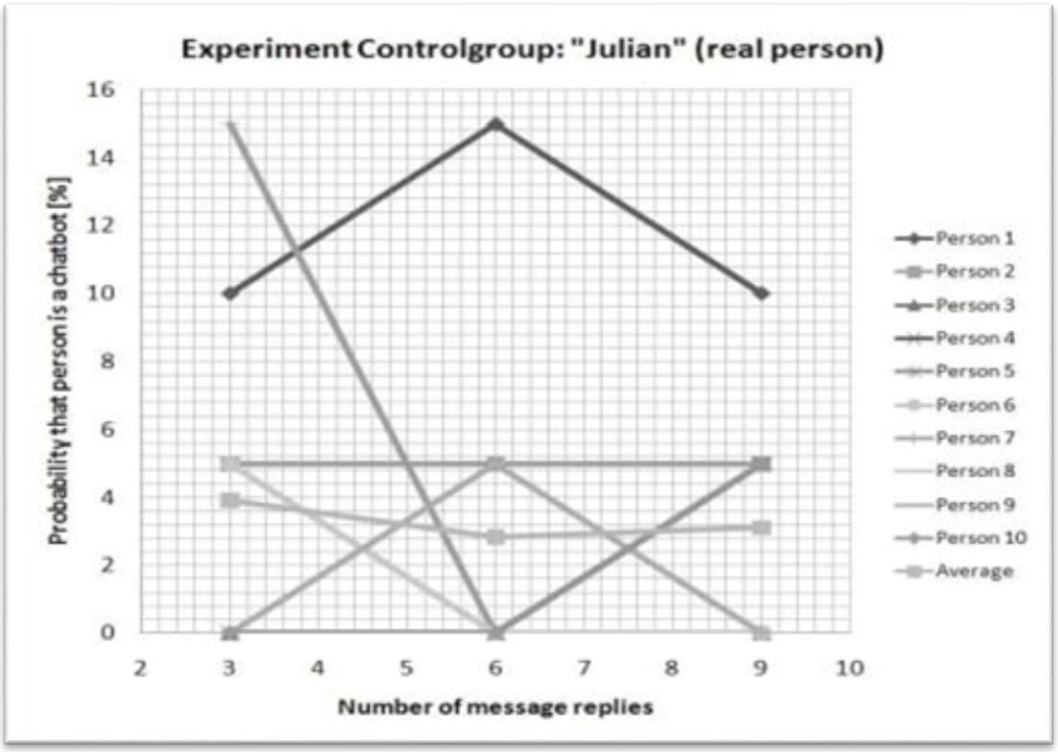
i) Yüksek teknoloji şirketi
ii) BT şirketi
iii) İskandinav finans şirketi
iv) Endüstri Mühendisliği şirketi
v) Telekomünikasyon şirketi.

Aşağıdaki grafik, bu firmalarda sosyal mühendislik yoluyla kurban bulma vakalarını göstermektedir.

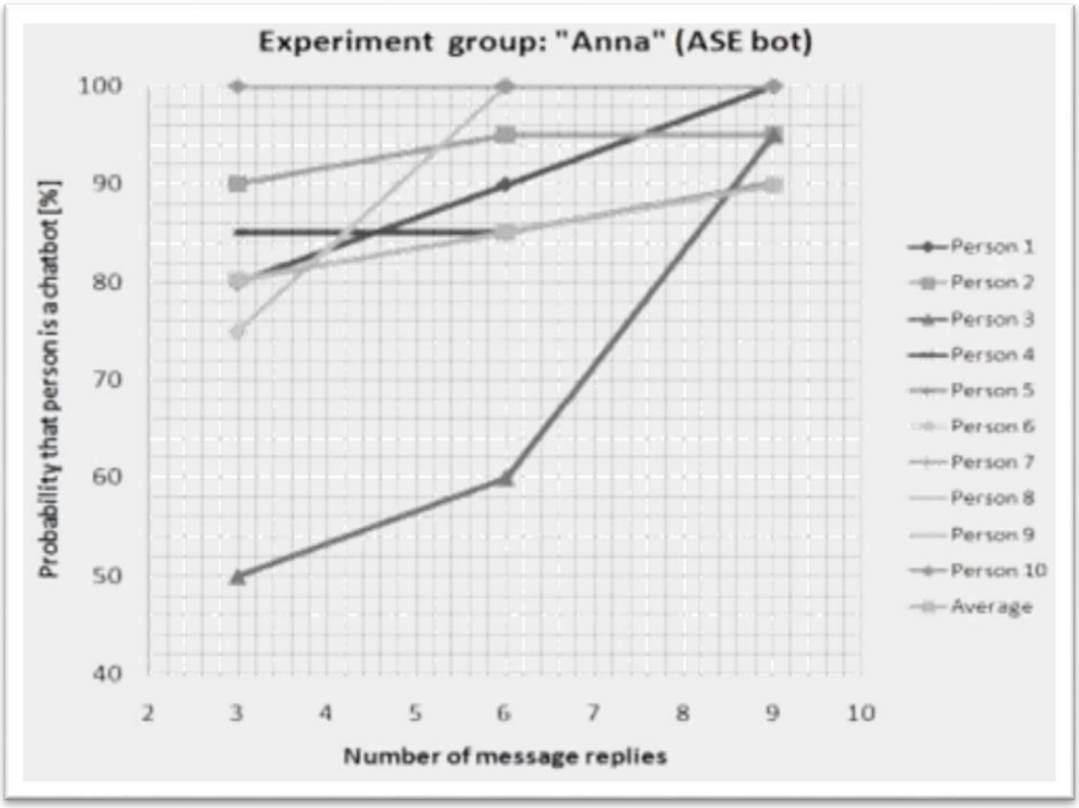


Şekil 2.1 : Otomatize Sosyal mühendislik aracıyla kurbanların bulunması.

Bilgi toplama sürecinin 16 dakika (organizasyon 3), 65 dakika (organizasyon2) ve organizasyon başına ortalama 44 dakika sürdüğü gözlenmiştir. Aşağıdaki iki grafikte, hedef gruplarla mesajlaşmaları, bu otomatik araçla oluşturulan farklı profiller kullanılarak sayısal olarak ifade edilir.



Şekil 2.2 : Julian karakteri ile hedefin gerçek insanları arasındaki bir konuşmanın sonuçları.



Şekil 2.3 : Anna karakteriyle hedef kullanıcılar için mesajlaşma sonuçları.

Bu çalışmalardan da görülebileceği gibi, otomatikleştirme girişimleri genellikle sosyal mühendislik saldırılarını otomatikleştirmeye odaklanır. Benim çalışmam, sızma testinin aşamalarından sadece bir tanesini içermeyip; Bilgi Toplama, Sosyal mühendislik: Ağ tarama ve Güvenlik Açığı Değerlendirme sonuçları ile OWASP 10 sıralamasında var olan zayıflıkların taramasını yaparak daha fazla çıktı üreterek daha fazla ipucu sağlar ve Bilgi Güvenliği uzmanı bu ipuçlarını kullanarak daha fazla atak vektör senaryosu oluşturabilir. Ortaya çıkan sonuçlara bakıldığında hem benzer çalışmaların varlığında hem de başarı oranları oluşturmayı düşündüğüm otomatik sızma testi aracıyla gerçekleştirdiğim testler ve sonuçları, yenilikçi ve etkili bir araç olduğu fikrini desteklemektedir.

Otomatik sistemler, Sızma Testi uzmanı bir insanın kalıcı kontrolünü gerektirir ve genellikle tüm ağı kapsayacak şekilde gereken önemli sayıda operasyon nedeniyle orta ve büyük boyuttaki sistemler bağlamında kabul edilebilir sonuçlar üretemez [9,10,11,12]

Mohamed Ghanem ve C. Thomas M. Chen, özellikle ağ perspektifi ile ilgili sızma testi uygulaması yapılan ve makine öğrenme ile sızma testi uygulamalarını akıllı ve verimli hale getirmek için Destekli Öğrenme teknikleri üzerinde durmuşlardır.[9] Bu araştırmada, sızma testi aşamalarının olasılıksal çıktısı (tarama, parmak izi, istismar), geçişler ve gözlemler için gerçek dünyadaki sızma testi uygulamasını yansıtacak yeterli olasılıkları tahsis etmeyi düşünmeleri önemli bir faktördü. Bu nedenle, güvenilir bir çevrimiçi Katalog (CVSS [30] ve ortak güvenlik açıkları ve istismarları (CVE) [31] oluşturan herkes için bilinen NIST Ulusal Güvenlik Açığı veri tabanı, seçtikleri yöntemi standart ve çapraz doğrulama olarak kullanan iki köklü kaynaktır. İşletim sistemleri kanıtlanmış güvenlik açıkları Puanlama fonksiyonu ve ilişkili değerler: bu tür kaynakların kullanımı, zengin içerik, kolay erişim ve düzenli güncellemeler ve hesaplama gibi cvssv3 ile ilgili, yazılım ve farklı uygulamalar mevcut mekanizma tarafından motive edilir. Her geçit veya gözlem [30.31] ' de detaylandırılmıştır.

2.2. Bilgi Güvenliği Çerçeveleri ve Standartları

Birçok kuruluş, devlet tarafından zorunlu, endüstriye özgü ve uluslararası siber güvenlik düzenlemelerinin bir karışımına uymalıdır. Ulusal, hatta küresel olarak ticaret yapan bir organizasyon için böyle bir meydan okuma çok önemlidir.

Çerçeve, mevcut yönergelere ve kuruluşların siber güvenlik riskini daha iyi yönetmesi ve azaltması için uygulanan gönüllü bir kılavuzdur. İş dünyası ve hükümet arasındaki koordineli çaba ile geliştirilen Kasıtlı Çerçeve, zorunlu vakfın güvenliğini göstermek için önlemler, kurallar ve uygulamalardan oluşur. Çerçevenin organize, uyarlanabilir, tekrarlanabilir ve etkili yaklaşımı, sistem sahiplerine ve kritik vakıf yöneticilerinin siber güvenlikle ilgili tehlikeyi denetlemelerine yardımcı olur.

Çeşitli türdeki işletmelerin güvenlik açıklarına karşı korunmasına yardımcı olabilecek bir dizi ortak güvenlik kontrol çerçevesi vardır ve doğru çerçeveyi seçmek MSP'lere ve müşterilerine özgü bir dizi faktöre bağlıdır. Burada, en yaygın BT güvenliği çerçevelerini ve her birinin artılarını ve eksilerini tartışacağız.

2.2.1. OWASP

OWASP, geliştiriciler ve web uygulaması güvenliği için bir farkındalık standart belgesidir. Bilgi Güvenliği Kurumlarının, web uygulamalarına yönelik en kritik güvenlik riskleri konusunda geniş bir fikir birliğini temsil eder. Şirketler bu belgeyi kabul etmeli ve web uygulamalarının bu riskleri en aza indirmesini sağlama sürecini başlatmalıdır. OWASP Top 10'u kullanmak, kuruluşunuzdaki yazılım geliştirme kültürünü daha güvenli kod üreten bir kültür olarak değiştirmek için belki de en etkili ilk adımdır. OWASP 10 ‘da işaret edilen en yaygın zayıflıklar:

i) Enjeksiyon
ii) Bozuk Kimlik Doğrulama
iii) Hassas Verileri Açığa Çıkarma
iv) XML Harici Varlıkları (XXE)
v) Bozuk Erişim Kontrolü
vi) Güvenlik Yanlış Yapılandırması
vii) Siteler Arası Komut Dosyası Oluşturma
viii) XSS Güvensiz Patlatma
ix) Bilinen Güvenlik Açıklarına Sahip Bileşenleri
x) Kullanma Yetersiz Kayıt ve İzleme

2.2.2. Bilgi Sistemleri Güvenlik Değerlendirme Çerçevesi (ISSAF)

ISSAF, aktif bir topluluk olmayan Bilgi Sistemleri Güvenlik değerlendirme Çerçevesi (ISSAF) olsa dahi, Sızma testinin çok iyi bir referans kaynağıdır. Kapsamlı adım adım Sızma testi teknik rehberlik sağlar. Aşağıdaki alanları kapsar.

- Proje Yönetimi
- Yönergeler ve En İyi Uygulamalar- Ön Değerlendirme, Değerlendirme ve Değerlendirme Sonrası
- Değerlendirme Metodolojisi

- Bilgi Güvenliği Politikası ve Güvenlik Organizasyonunun Gözden Geçirilmesi
- Risk Değerlendirme Metodolojisinin Değerlendirilmesi
- Teknik Kontrol Değerlendirmesi
- Teknik Kontrol Değerlendirmesi- Metodoloji
- Şifre Kırma Stratejileri
- Unix / Linux Sistem Güvenlik Değerlendirmesi
- Windows Sistem Güvenlik Değerlendirmesi
- Novell Netware Güvenlik Değerlendirmesi
- Veri tabanı Güvenliği Değerlendirmesi
- Kablosuz Güvenlik Değerlendirmesi
- Anahtar Güvenlik Değerlendirmesi
- Yönlendirici Güvenlik Değerlendirmesi
- Güvenlik Duvarı Güvenlik Değerlendirmesi
- Saldırı Tespit Sistemi Güvenlik Değerlendirmesi
- VPN Güvenlik Değerlendirmesi
- Anti-Virüs Sistemi Güvenlik Değerlendirmesi ve Yönetimi Stratejisi
- Web Uygulaması Güvenlik Değerlendirmesi
- Depolama Alanı Ağı (San) Güvenliği
- İnternet Kullanıcısı Güvenliği
- 400 Güvenlik Olarak
- Kaynak Kod Denetimi
- İkili Denetim
- Sosyal mühendislik
- Fiziksel Güvenlik Değerlendirmesi
- Olay Analizi
- Kayıt / İzleme ve Denetim Süreçlerinin Gözden Geçirilmesi
- İş Sürekliliği Planlaması ve Olağanüstü Durum Kurtarma
- Güvenlik Bilinci ve Eğitimi
- Dış Kaynak Güvenliği Endişeleri
- Güvenlik Değerlendirme Projelerinin Hukuki Boyutları

- Gizlilik Sözleşmesi (NDA)
- Güvenlik Değerlendirme Sözleşmesi
- Teklif Talebi Şablonu
- Masaüstü Güvenlik Kontrol Listesi- Windows
- Linux Güvenlik Kontrol Listesi
- Solaris İşletim Sistemi Güvenlik Kontrol Listesi
- Varsayılan Bağlantı Noktaları- Güvenlik Duvarı
- Varsayılan Bağlantı Noktaları- IDS / IPS
- Bağlantılar
- Sızma Testi Laboratuvarı Tasarımı

2.2.3. Açık Kaynak Güvenlik Test Metodolojisi Kılavuzu (OSSTMM)

OSSTMM, fiziksel konumların, iş akışının, insan güvenliği testlerinin, fiziksel güvenlik testlerinin, kablosuz güvenlik testlerinin, telekomünikasyon güvenlik testlerinin, veri ağlarının güvenlik testlerinin ve uygunluğunun operasyonel güvenliğini test etmek için kullanılan bir metodolojidir. OSSTMM, uygulamalı sızma testi kılavuzu yerine IOS 27001 referansını destekliyor olabilir. OSSTMM aşağıdaki anahtar bölümleri içerir:

- Operasyonel Güvenlik Metrikleri
- Güven Analizi
- İş Akışı.
- İnsan Güvenliği Testi
- Fiziksel Güvenlik Testi
- Kablosuz Güvenlik Testi
- Telekomünikasyon Güvenlik Testi
- Veri Ağları Güvenlik Testi
- Uyumluluk Düzenlemeleri
- STAR ile Raporlama (Güvenlik Test Denetim Raporu)

2.2.4. CREST Sızma Testi Kılavuzu

CREST Sızma Testi Kılavuzu, sızma testi programının uygulanması ve yönetimi hakkında pratik önerilerde bulunarak kuruluşların teknik güvenlik güvencesi çerçevesinin bir parçası olarak etkin, ödedikleri paranın karşılığını alarak sızma testi yapmalarına yardımcı olur. Kuruluşların sızma testlerine hazırlanmalarını, gerçek testleri tutarlı ve yetkin bir şekilde yapmalarını ve testleri etkin bir şekilde takip etmelerini sağlamak için tasarlanmıştır.

Kılavuz, iyi yönetilen sızma testleri yapmak için anlaşılması gereken, sızma testinin ne olduğunu ve ne olmadığını açıklayan, güçlü yönlerini ve sınırlamalarını açıklayan ve bir kuruluşun neden ve hangi özelliklere sahip bir çalışanı görevlendireceğini açıklar.

Kılavuz, üç aşamalı faydalı bir yaklaşım sunar ve aşağıdakiler için gerekli önlemlerin nasıl alınacağı konusunda tavsiye ve rehberlik sağlar:

Teknik güvenlik güvencesi çerçevesinin bir parçası olarak sızma testine hazırlanmak; uygun bir Sızma testi yönetim yapısı tarafından yönetilir; test etmenin nedenlerini, testin amacını ve hedef ortamları dikkate alarak ve testler yapmak için uygun tedarikçiler tayin eder.

Test stilini ve türünü onaylayarak kurumsal çapta Sızma testleri gerçekleştirmek; test kısıtlamalarını değerlendirme, test sürecini yönetme, testleri etkin bir şekilde planlama ve gerçekleştirme, güvenlik açıklarını belirleme, patlatma ve düzeltme

Uygun takip faaliyetlerini yürütmek, zayıflıkları düzeltmek, iyileştirme planını sürdürmek ve üzerinde anlaşmaya varılmış bir eylem planı sunmak.

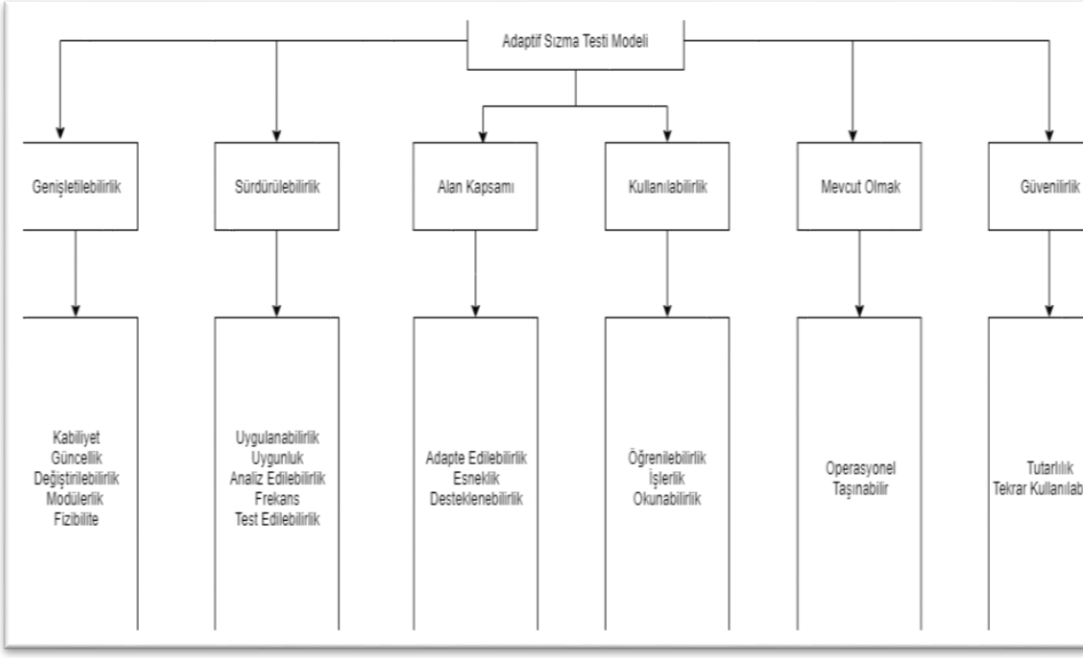
3. ADAPTİF SIZMA TESTİ YAKLAŞIMI

Uygulama ve gerçek dünya uygulaması, Sızma testlerinin nasıl etkili bir şekilde yapılacağını öğrenmek için önemlidir. Adaptif Sızma Testi, derinlemesine güvenlik değerlendirmeleri yapmak için pratik deneyim ve sağlam bir çerçeve sağlayacak kapsamlı bir yaklaşımdır. Başarılı Sızma test cihazlarının çeşitli kurumsal ortamlarda kapsamlı ve verimli güvenlik değerlendirmeleri sağlamak için kullandıkları taktikleri, teknikleri ve prosedürleri (TTP' ler) ele alır. Sunulan yöntemler Sızma test cihazlarımızın operasyonel deneyimi tarafından sürekli olarak iyileştirilen TTP' lere dayanmaktadır.

İş için doğru aleti kullanmak genellikle etkili bir Sızma testi için fark yaratır. Saldırı vektörlerini tanımlamak ve kurumsal ortamlara sızmak için çeşitli ticari ve açık kaynaklı araçlarda kullanılabilir ancak geliştirilen araç ile size örnekler sunulmuştur. Hem ağ hem de web test araçlarını ve çerçeveleri ele alınacaktır. Bu araçlar, değişken hedef ortamlara karşı etkili ve etkili bir şekilde Sızma testleri gerçekleştirmenizi sağlar. Adaptif Sızma Testi, engelleri aşacak, modern saldırı tekniklerini uygulayarak ve Sızma testlerini zorlamak için gelişmiş taktiklerin nasıl kullanılacağını öğretir.

3.1. Adaptif Sızma Testi Aracının Mimarisi

ISO / IEC 25010: 2013 kalite standardı, yazılım özelliklerini sekiz önemli özellikten oluşan alt özelliklere ayırarak bir ürün modeli tanımlar. Bu ürün, metodolojisine veya çerçevesine bakılmaksızın OWASP 10'un zayıf yönleri dikkate alınarak geliştirilmiştir. Burada, söz konusu aracın tam sızıntı testiyle ilgilenip ilgilenmediğini araştırıldı ve Uyarlanabilir sızma testi yaklaşımını değerlendirmek yaklaşımını desteklemek için bir araç oluşturulmuştur. Uyarlanabilir sızma testi aracı, standartları akılda tutarak yaratılmıştır.



Şekil 3.1 : Adaptif Sızma Testi Kalite Modeli.

Bir Web uygulamasının sızma testi zaman kazanmak ve manuel sızma testi kıyasla daha geniş bir kapsam kapsayacak şekilde bir web sitesi ile otomatik hale getirilebilir. Bu araç, web sitesini tarar, yani web sitesindeki tüm sayfaları bir kerede ziyaret etmeye başlar ve her sayfada karşılaştığı tüm benzersiz URL'nin adresini saklar ve bir dahaki sefere ziyaret etmek için bir deque olarak saklar. Araç aşağıdaki zayıflıkları bulacak kabiliyette geliştirilmiştir

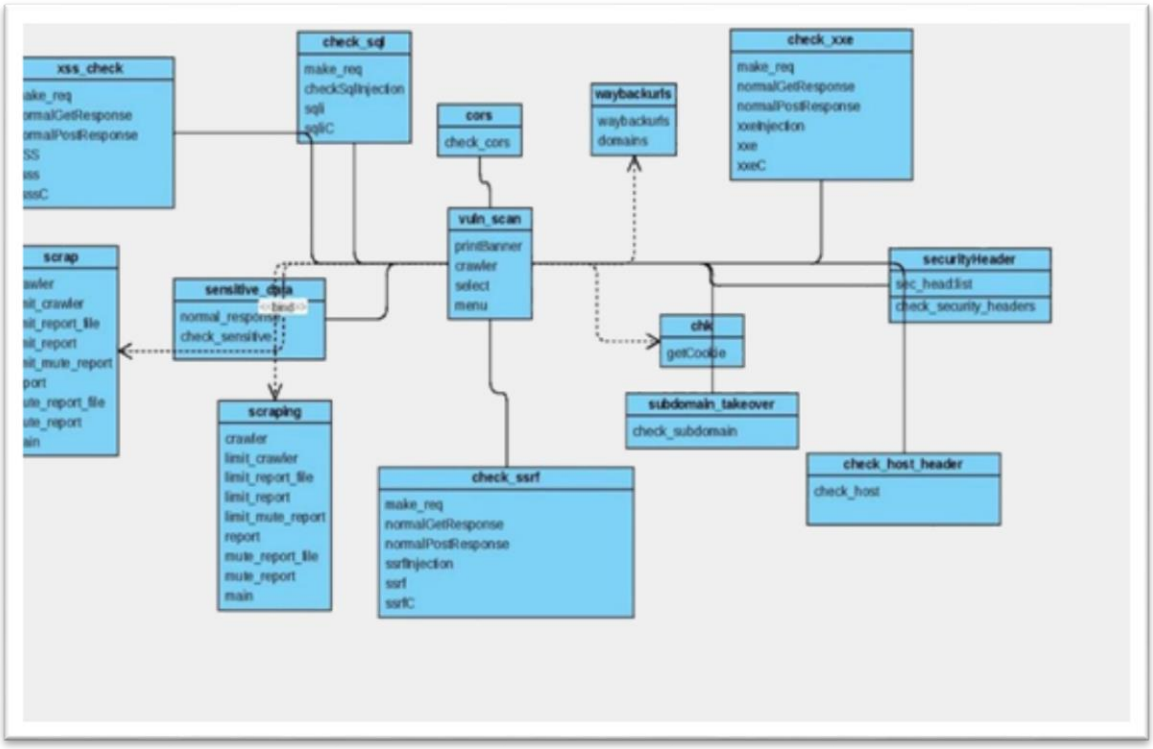
- XSS
- SQL enjeksiyon
- XXE
- SSRF
- Alt Alan Devralma
- Eksik güvenlik başlıkları
- Ana Başlık Enjeksiyonu CORS
- Hassas Veri Sızıntısı

3.1.1. Adaptif Sızma Testi Aracının Bileşenleri

Şekil 3.2 : Adaptif Sızma Testi Kullanım Durumu Diyagramı.

Yukarıdaki diyagram, bir web sitesinin Sızma testinin tüm sürecini gösterir. Bir Sızma testi gerçekleştirmek için bizim takip etmemiz gereken 3 adım:

i) Tarama
ii) Menü Yönlendirme
iii) Zayıflık testi.



Şekil 3.3 : Adaptif Sızma Testi Aracının Sınıf Diyagramı.

Bilgi Akışı: vul_scan varlıklarından veriler hurdaya çıkarılacak varlık olacaktır. Burada hurdaya atılan varlık, web sitesini taramak ve bir rapor dosyası oluşturmak için tarayıcı modülünü çağırır. VuL_scanner ve hurdaya ayırma arasındaki ilişki hurdaya ayırma kısmı kullanıcıya bağlı isteğe bağlı bir araçtır. Kullanıcı web sitesini taramak istiyorsa, yalnızca bu modül çalışmaz.

Veri kazıma sonra chk ve wayback_URLs varlıklarına gidiyor. Chk varlık tarayıcı uygulamanın çerezleri alır. Vul_scan varlığı ile ilişki isteğe bağlıdır.

Bundan sonra veriler kullanıcının etkileşimi ile tek tek vul_scan varlıktan diğer tüm varlıklara gider. Seri, tarayıcının kullanıcısına bağlıdır. Biri önce xxs kontrolü gerçekleştirebilir, diğer taraftan SQL_Check ile başlayabilir.Tamamen kullanıcıya bağlıdır.

Çalışma yordamı: Check_SQL varlık: (Vul_scan varlık ile ilişkisi bir çoktur.) Sunucuya bir istekte bulunun ve sonra sunucudan yanıt alın. Daha sonra form girişine normal SQL enjeksiyonu enjekte edin. Çalışmazsa, modüller SQLi, SQLblind enjeksiyon moduna gider. Modüller tarama yaparken herhangi bir güvenlik açığı bulursa kullanıcıya bir mesaj gösterir.

-XSS kontrolü: bire çok ilişkisi

Önce modül bağlanmak için sunucuya bir istek gönderir. Ardından normal alma ve gönderme isteğini kontrol edin. Sonra URL veya form girişi için XSS enjeksiyon saldırısı başlatın. Savunmasız herhangi bulursa kullanıcı için bir uyarı gösterir.

check__SSRF: bire çok ilişki Önce modül bağlanmak için sunucuya bir istek gönderir. Ardından normal alma ve gönderme isteğini kontrol edin. Ardından URL'ye SSRF enjeksiyon saldırısını başlatın veya girdi oluşturun. Herhangi bir güvenlik açığı bulursa kullanıcıya bir uyarı gösterir.

check__XXE: bire çok ilişki Önce modül sunucuya bağlanmak için bir istek gönderir. Ardından normal alma ve gönderme isteğini kontrol edin. Ardından URL'ye SSRF enjeksiyon saldırısını başlatın veya girdi oluşturun. Herhangi bir güvenlik açığı bulursa kullanıcıya bir uyarı gösterir.

Sensitive_data: Bu bölümde bu modüller robot.txt, password.txt dosyalarını arıyor.

Alt alan adı alımı: ana ip adresinin altında alt alan adını bulun. cors: sunucuya bir parodi isteği göndererek cors hatasını kontrol edin.

check_host üstbilgisi ve güvenlik üstbilgisi: başlıkları inceleyerek bu güvenlik açığını bulun ve imza veritabanlarıyla eşleştirin. Bu uygulamanın 3 temel kısmı bulunmaktadır.

Tarama (CRAWLER) fonksiyonu ilk adımdır ve komut dosyası başlar başlamaz en önemli adımdır, scraping.py içindeki tarayıcı işlevi, web sitesi adını ve taranan URL'lerin sahip olduğu dosyayı iki argüman alarak kaydeder. Kimlik doğrulamaya dayalı iki tarama işlevi vardır; Web sitesini kimliği doğrulanmamış kullanıcı olarak tarama (İşlenen URL'leri kaydetmek için iki bağımsız değişken web sitesi adı ve dosya adı alır) ve Web sitesini kimliği doğrulanmış kullanıcı olarak tarama (Üç bağımsız değişken web sitesi adı, işlenen URL'leri kaydetmek için dosya adı ve kimliği doğrulanmış kullanıcının çerezi alır)Tarayıcı, web sitesini dizin sayfasından ziyaret etmeye başlar ve src özniteliğine sahip tüm bağlayıcı ve HTML etiketlerini ve bunu web sitesinden veri kazımak için kullanılan beautifulSoup kütüphanesini kullanarak javascript'teki bağlantıları bulur. Tüm bağlantıları ziyaret ettikten sonra dahili olarak demofile.txt dosyasına kaydeder ve ayrıca kimliği doğrulanmış kullanıcı için oturumu kaybetmemesine dikkat eder, bu nedenle

oturumlarında çıkış veya çıkış anahtar kelimeleri içeren URL'leri ziyaret etmekten kaçınır. Tüm URL'yi ziyaret ettiğinden emin olmak için bir komut dosyası waybackURLs.py kullanır, hedef web sitesine ait olan URL'leri esas olarak başlangıçtan yani web sitesinin şu ana kadar oluşturulduğu zamana kadar olan URL'leri arar. Genellikle manuel bir sızma testi unutulabilecek URL'ler için çok zengin bir kaynaktır. Bu unutulan URL'ler web sitesi tarafından başlangıçta kullanılan ve web sitesinden kaldırılması unutulunca ve bazı saldırılara karşı savunmasız kalmasına sebebiyet verebilir.

Menü, kullanıcı için arayüzü, araç ile yapılabilecekler konusunda kullanıcıya yardımcı olur ve kullanıcı için kullanmayı çok kolay hale getirir, kullanıcının web sitesinde test etmek istediği zayıflığı seçmesine yardımcı olur. Menü sağlamaz ve burada aksine sonuç üretmek için çok uzun zaman alır birçok araç vardır burada çok zaman tasarrufu ve zaman tasarrufu ve hataların kontrol listesi için bir liste verir sadece belirli bir hata kontrol edebilirsiniz. Bir kontrol listesi olmayan manuel bir test cihazı bazı hataları test etmek için kaçırabilir. Linux terminalindeki animasyonlu metne, animasyonlu bir görünüm vermek için bir linux yardımcı programı olan figlet varlığı kullanılır.

Şekil 3.4 : Adaptif Sızma Testi aracının Arayüzü.

Belirli Güvenlik Zafiyetlerinin Test Sürecinde ise çalışmanın başında belirttiğimiz gibi, OWASP'NİN en bilinen zayıflıklarına odaklandık ve bu bölümde 9 zayıflık yer aldı. Bu zayıflıkların her biri için ayrı bir modül oluşturuldu ve çalışma felsefeleri aşağıda listelenmiştir.

- XSS: Menüden seçim yapabileceğiniz ilk seçenek, Siteler Arası Komut Dizisi anlamına gelen XSS, modern web uygulamasındaki en yaygın hatadır, çok büyük işletmeler de bu hata türüne karşı savunmasızdır. XSS' de bir saldırgan, kurbanın tarayıcısında javascript'i çalıştırabilir. 1. seçeneğe girilen komut dosyası xxs_check.py dosyasındaki XSS işlevini başlatır. Web sitesi adı ve varsa çerez değeri iki argüman alır ve sonra form etiketli tüm URL'leri ve girişlerini almak için form_input.py dosyasındaki tür adlarını kullanır. Tüm URL'leri ve giriş türlerini listede sakladıktan sonra, tek bir URL ve onun verilerini almak için döngüyü kullanır ve URL'nin XSS' ye karşı savunmasız olup olmadığını test eder. Girişte batman gibi bir benzersiz göndererek tek bir URL'nin savunmasız olduğunu söyler ve yanıtın yanıtta aynı dizeyi içerip içermediğini kontrol eder, dize varsa girişte <> / () gönderir ve bkz. şifrelenmiş olup olmadıklarının yanıtını gönderir. Aksi halde, istekte sunduğu bir XSS çokgen listesi vardır ve dizenin terminalde yazdırdığı dizede varsa yanıtta yansıtılıp yansıtılmadığını kontrol eder; URL, saldırıda kullanılan yük ile XSS' e karşı savunmasızdır.
- SQL Enjeksiyonu: bu menüde ikinci seçenekte yer alan ve web uygulamalarında bulunan çok eski zayıflık olan ve günümüzde Wordpress gibi CMweb çerçeveleri kullanarak geliştirilen Web Uygulamalarında çok daha az yaygındır. Bir saldırganın herhangi bir kullanıcının verilere erişmek ve hatta verileri değiştirebilmesine olanak tanıyan çok ciddi bir hatadır. 2 seçeneği seçme komut dosyasında SQL'i işlevi present olarak adlandırılır check_SQL.py sağlanırsa web sitesi adı ve çerez iki argüman alır ve giriş ve URL'leri XSS olarak alır.
- XXE Enjeksiyonu: Menüdeki üçüncü seçenek, web sitesinde bulunan ve kullanıcının verilerini depolamak için XML dosyasını içeren veya kullanıcıya talep edilen ile cevap vermek için XML kullanan çok yeni bir zayıflıktır. Bu

nedenle, check_XXE.py dosyasında XXE işlevini çağırdığımız komut dosyasını kullanarak XXE'yi bulmak için, web sitesi adı ve varsa çerez iki argüman alır. Burada, her istek için application / XML isteğinin içerik türünü değiştirir ve normal verilerle isteğin yanıtının uzunluğu ve durum kodu ile XXE yükü ile istek ve fark olasılığını gösterip göstermediği arasındaki farkı kontrol eder. XXE güvenlik açığı belirli bir yük ile belirli bir URL bulduğunda onu işaretleyen XXE saldırısı. Yük olarak XXE poliglotları kullanır.

- SSRF: bu menüde dördüncü seçenek ve aynı zamanda çok yeni bir hata türü olan bu zayıflık, saldırganın sunucu ve port üzerindeki özel dosyaya erişim elde edebilmesinden yararlanarak mümkün olan sunucuların mevcut olduğu ağda tarama işlemi yapar. Bu hatayı komut dosyasında bulmak için SSRF işlevini çağırılır check_SSRF.py dosya sağlanırsa web sitesi ve çerez olarak iki argüman alır. Bu noktada normal istek ve kötü amaçlı yük arasındaki farkı geliştirilir. Durum kodu ve yanıt uzunluğu arasındaki fark varsa, SSRF hatası belirli yükü belirli yük ile bulduğunda işaretlenir. Yük olarak SSRF polyglots kullanılır.
- CORS: Web sitesinde bir istek göndermek ve saldırganın değiştirebileceğinden daha kötü niyetli bir kökene yanıt vermek zorunda olduğu alan adından farklı bir kaynağa izin verilir, menüdeki beşinci seçenektir, bu da oldukça yeni bir zayıflık türüdür. Kurbanın isteği için başlangıç başlığı ve zararlı sunucusunda yanıt alınması ve javascipt değişiklikler yapılması kullanıcıyı tehlikeye atar. Bu zayıflığı bulmak için, her isteğin Kökeni üstbilgisini değiştiririz ve AccessControl'e İzin Ver üstbilgisinin * veya boş olup olmadığını veya web sitesinin adının kökenini belirleyip işaretleriz ve CORS URL bulduğunda terminalde yazdırılır.
- Eksik Güvenlik Başlıkları: Bir web sitesi yanıtının saldırılara karşı güvenli hale getirmesi gereken tüm güvenlik başlıklarının mevcut olup olmadığını kontrol eden menüdeki altıncı seçenektir. Komut dosyasını bulmak için security_header.py dosyasında check_security_header işlevi çağırılır. Varsa veya yoksa X-Frame-Options, X-XSS-Protection ve daha birçok başlığı kontrol eder, savunmasız olarak işaretler ve eksik özel güvenlik başlığı olarak terminale yazdırır.

- Alt alan adı ele geçirme: Bu, menüdeki yedinci seçenektir ve saldırgan bir alt alan adı talep edebilir. Bu işlevi bulmak için subdomain_takeover.py dosyasında check_subdomain çağrılır. Komut dosyası, hedef etki alanının alt etki alanlarını aramak için https://crt.sh/?q=%25 kullanır ve ardından bulunan tüm alt etki alanını isterse respose durum kodu 404 veya Bulunamadı dizesi mevcutsa, komut dosyası alt etki alanı devralma olarak bulur ve uçbirime yazdırır.
- Hassas Veri Sızıntısı: Menüde sekizinci seçenek, geliştiricilerin kaldırmayı unuttuğu veya varsayılan olarak sunucuda bulunan hassas dosyaya erişebilmesidir. Bu hatayı bulmak için, varsayılan olarak sunucuda bulunan ve hassas dosya içeren bir dosya listesi kullanılır. Belirli bir dosya için talep ettiğimizden ve yanıt durumu 200 olduğundan, belirli bir URL'de hassas veri sızıntısı olmasını ve terminalde yazdırılır.
- Ana Bilgisayar Üstbilgisi Enjeksiyonu: Bu menüdeki dokuzuncu seçenek, bir saldırganın yalnızca ana bilgisayar başlığını düzenleyerek korsanın istenen çıktısını üretmek için kodu manuel olarak yönlendirebilir. Büyük olasılıkla web sunucuları, ana bilgisayar üstbilgisini, uygun bir yeniden düzenleme olmadan listedeki ilk sanal ana bilgisayara geçirecek şekilde yapılandırır. Böylece, keyfi sanal bilgisayar üstbilgileriyle HTTP isteklerini ilk sanal ana bilgisayara göndermek mümkündür. Bu durumda, geçersiz bir Ana Bilgisayar belirtirsek, web sunucusunun bunu işlediği ve geçersiz ana bilgisayar üstbilgisini listedeki ilk sanal ana bilgisayara geçirdiği anlamına gelir. Bu hatayı bulmak için check_host_header.py dosyasındaki check_host işlevini kullanırız, bu istekte Host ve X ForwardedHost üstbilgisi ile URL ister ve kötü amaçlı ana bilgisayar bulunursa, URL ile bulunan ana bilgisayar üstbilgisi enjeksiyonu olarak işaretler.

3.1.2. Çalışmanın Matematiksel Modeli

Bir CRAWLER, bir veya daha fazla tohum URL'si verildiğinde, bu URL'lerle ilişkili web sayfalarını indiren, içinde bulunan herhangi bir köprüyü ayıklayan ve yinelemeli olarak bu köprülerle tanımlanan web sayfalarını indirmeye devam eden

bir programdır. CRAWLER, web arama motorlarının önemli bir bileşenidir ve burada arama motoru tarafından dizine eklenen web sayfalarının yapısını toplamak için kullanılırlar. Ayrıca, web veri madenciliği, karşılaştırma alışveriş motorları vb. gibi çok sayıda web sayfasını işleyen diğer birçok uygulamada kullanılırlar.

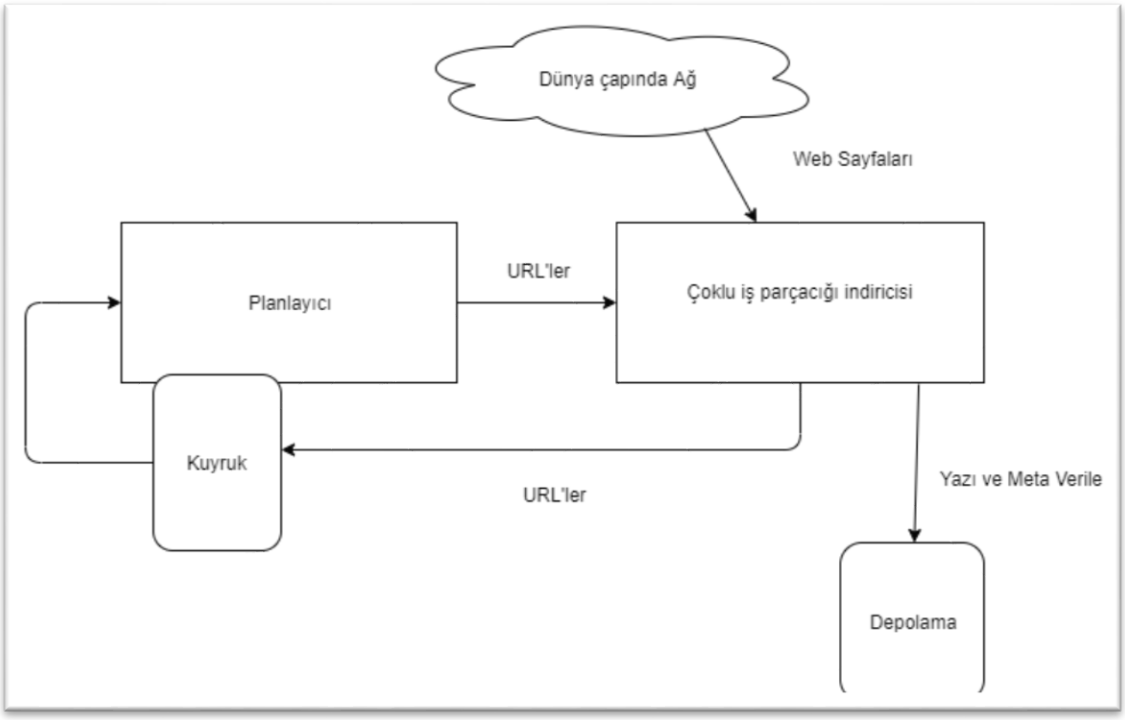
Kavramsal basitliklerine rağmen, yüksek performanslı CRAWLER uygulanması, Web' in 3462W Web içeriği çıkarma ölçeği nedeniyle büyük mühendislik zorlukları oluşturmaktadır. 'Yüzeysel web' in önemli bir kısmını makul bir süre içinde taramak için, CRAWLER saniyede binlerce sayfa indirmeli ve genellikle onlarca veya yüzlerce bilgisayara dağıtılmalıdır. İki ana veri yapısı – henüz taranacak URL'lerin 'sınır' kümesi ve keşfedilen URL'ler kümesi-genellikle ana belleğe sığmaz, bu nedenle verimli disk tabanlı gösterimlerin kullanılması gerekir.

Bir web sayfasından bir köprü çıkarıldıktan sonra, tarayıcı, aynı URL'nin birden çok örneğini bekleyen URL kümesine eklemekten kaçınmak için bu URL'nin daha önce karşılaşılıp karşılaşılmadığını test etmelidir. Bu, karma tablo gibi set üyelik testini destekleyen bir veri yapısı gerektirir. CRAWLER fonksiyonun mimarisi w3463 W kullanılan karma işlevin çarpışmaya dayanıklı olduğuna ve karma değerlerin yeterince büyük olduğuna dikkat edilmelidir (bir dizi n URL'yi korumak, her biri $\log_2 n^2$ bitli karma değerleri gerektirir).

Bu büyük veri yapılarına ek olarak, çoğu web ölçekli CRAWLER fonksiyonu, DNS arama sonuçları için önbellekler gibi bazı yardımcı veri yapılarını da korur. Yine, bu veri yapıları tarama makineleri arasında bölümlenmiş olabilir. CRAWLER, dizine eklenecek sayfaları toplamak için kullanılan web arama motorlarının önemli bir bileşenidir. Tarayıcıların genel aramanın ötesinde, örneğin web veri madenciliğinde (örneğin, Attributor, telif hakkı ihlalleri için Web'i mayınlayan bir hizmet veya bir fiyat karşılaştırma hizmeti olan shopwiki) birçok uygulaması vardır.

3.1.2.1. CRAWLER Fonksiyonunun Kimlik Doğrulamasız Program Modeli

CRAWLER fonksiyonunun işlevi program modelinin ana hedefi ve görevi, şemanın söylediği gibidir:



Şekil 3.5 : Kimlik Doğrulamasız Model.

Model 1: CRAWLER işleminin kimlik doğrulama gerekmeden yapılacağını varsayalım. Bu durumu matematiksel olarak ifade etmek için geliştirilen eşitlik [13].

$$W \rightarrow M(W) = I \rightarrow Q(I) = \sum_{I}^{N} W_i \rightarrow S\left(\sum_{i}^{N} W_i\right) = [\text{Veri Matrisi}] \tag{3.1}$$

Bu süreç tekrar tekrar gerçekleşiyor.

W, Zayıflık taraması yapılacak uygulamanın tüm verilerini temsil eder.

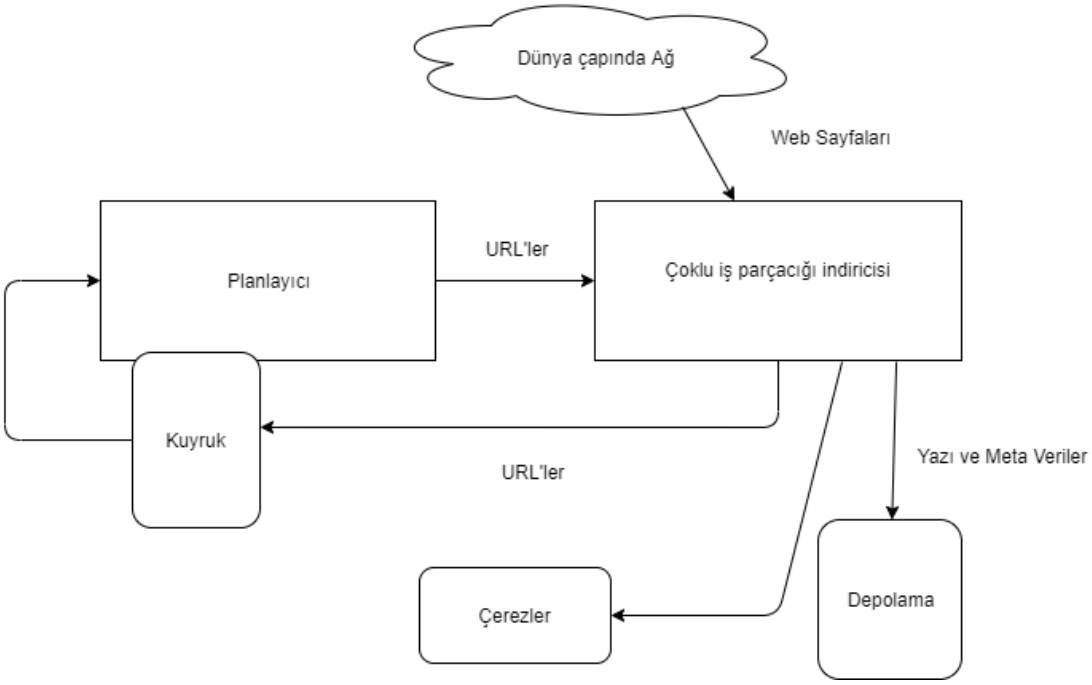
M, W ile ilgili tüm bilgileri alan ve algoritma için kullanılabilir bilgileri indiren fonksiyondur.

Q, toplamın bölümlerine ayrılabilmesi için verileri düzenleyen işlevdir.

S işlevi veya Zamanlayıcı yeniden test edilmesi gereken son veri kümesini verir, bu eylem tekrarlanır.

3.1.2.2. CRAWLER Fonksiyonunun Kimlik Doğrulamalı Program Modeli

Kimlik doğrulamalı CRAWLER işlevi yapıldığı durumlarda ise matematiksel model aşağıdaki gibidir.



Şekil 3.6 : Kimlik Doğrulamalı Model.

Model 2: Bu tür tarama işlevinin, tarama birimi verilerinin kaydedilmesini sağlayan bir kimlik doğrulama işlemi olması gerekir. Aşağıdaki şema, verilerin iki bölüme ayrılacağını söylüyor

i) Biri dosya tasarruflarına gidiyor
ii) İkincisi çerez dosyalarına gidiyor.

Bunun matematiksel ifadesini aşağıdaki şekilde gerçekleştiriyoruz. [13].

$$\mathrm{W} \rightarrow \mathrm{M(W)} = \mathrm{I} + \mathrm{I_C} \rightarrow \mathrm{Q(I)} = \sum_{I}^{N} \mathrm{W_i} \rightarrow \mathrm{S}\left(\sum_{\mathrm{i}}^{\mathrm{N}} \mathrm{W_i}\right) = [\text{Veri Matrisi } (\textit{Ikı dosya}\ )] \tag{3.2}$$

4. GERÇEKLEME VE ÖZELLİKLERİ

4.1. Test Ortamları ve Özellikleri

Otomatikleştirilmiş uygulamayı 7 farklı ortamda yeterince farklı parametrelerle test ettim ve sonuçları manuel olarak yaptığım testlerle karşılaştırdım. Test ortamları aşağıda yer almaktadır:

- TestphpVulnweb Ortamı, bu ortam, web saldırılarına karşı kasıtlı olarak zayıflık içeren bir örnek PHP uygulamasıdır. Acunetix'i test etmenize yardımcı olmak için tasarlanmıştır. Ayrıca, geliştirici hatalarından ve kötü yapılandırmadan faydalanabilecek birisinin web sitenize girmesine nasıl izin vereceğini anlamanıza yardımcı olur. Ayrıca becerilerinizi test etmek için diğer otomatik araçları veya manuel güvenlik testlerini de kullanabilir. İpucu: potansiyel SQL enjeksiyonlarını, siteler arası komut dosyalarını (XSS) ve siteler arası istek sahteciliğini (CSRF) ve daha fazlasını aranabilir.
- DemoTestfire/Altoroj Ortamı, Bir örnek bankacılık J2EE web uygulamasıdır. Web uygulamaları, uygulama güvenliği göz önünde bulundurularak değil, uygulama işlevselliği göz önünde bulundurularak yazıldığında ne olacağını gösterir. Gerçek hayattaki uygulama güvenliği sorunları hakkında daha fazla bilgi edinmek ve öğrenmek için basit ve düzenli bir platformdur. AltoroJ, herhangi bir ek çerçeve kullanmadan standart Javave JSP işlevlerini kullanır. Gerçek hayattaki uygulamaların çoğunda çerçeveler kullanılırken, her iki durumda da aynı uygulama güvenliği ilkeleri geçerlidir. Belirli bir aşinalığı olmayan biri için çerçeveleri anlamak ve öğrenmek zor olabilir. Altoroj'a çok benzeyen birçok büyük ve karmaşık "eski" Java web uygulaması vardır (ancak elbette birçoğu neredeyse tekrar tekrar daha karmaşıktır).
- ZeroWebapp Security Ortamı, Ücretsiz Çevrimiçi Banka web sitesi olan bu ortam Micro FocusFortify tarafından yalnızca Micro Focusfortify'ninWebInspect ürünlerinin Web uygulaması güvenlik açıklarını tespit etme ve bildirmedeki işlevselliğini ve etkinliğini göstermek için yayınlanmıştır. Bu site gerçek bir bankacılık sitesi değildir ve üçüncü taraf

ürünlere ve / veya web sitelerine benzerlikler tamamen rastlantıdır. Bu site açık veya zımni herhangi bir garanti olmaksızın "olduğu gibi" sunulmaktadır. Micro FocusFortify, bu web sitesini kullanımınızla ilişkili herhangi bir riski kabul etmemektedir. Bu web sitesinin kullanılması, Micro Focusfortify'nin Kullanım koşullarını okuduğunuzu ve kabul ettiğinizi gösterir.

- Testhtml5Vulnweb Ortamı/Bu, tasarımdan kötü amaçlı tasarıma sahip bir HTML5 uygulamasıdır. Bu uygulama, Acunetix, diğer araçlar veya manuel sızma testi becerilerinin test edebilmesi için oluşturuldu. Uygulama kodu, Siteler Arası Komut Dosyası Oluşturma (XSS) ve XML harici varlığı (XXE) gibi saldırılara karşı zayıflık içerir. Bu sitede sağlanan bağlantılar siteye bağlı değildir ve burada sadece örnek olarak bulunmaktadır.
- Tayfun Güvenlik Açığı Makinesi Ortamı, Typhoon savunmasız VM, siber güvenlik becerilerini geliştirmek isteyen araştırmacılar için laboratuvar ortamı sağlayan çeşitli güvenlik açıklarıyla birlikte gelen sanal bir makinedir. Typhoon VM v1, Prisma CSI ekibi tarafından şirket tarafından sunulan pratik Sızma Testi Eğitimi için bir mini laboratuvar ortamı sağlamak üzere geliştirilmiştir. Ayrıca, sanal makineyi sisteminize indirip kurulabilir, bu alanda bazı pratik beceriler kazanma şansını verir.
- Damn Güvenlik Açığı Web Uygulaması Ortamı, Damn zararlı Web uygulaması (DVWA), savunmasız olan bir PHP / MySQL web uygulamasıdır. BT güvenlik uzmanlarının temel ortamdaki beceri ve araçları test etmek, web geliştiricilerinin web uygulamalarını güvence altına alma sürecini daha iyi anlamalarına yardımcı olmak ve sınıf ortamındaki öğretmenlere / öğrencilere nasıl web uygulaması güvenliğini öğretmeyi / yardım etmeyi öğrenmelerini sağlamak için temel hedeflere yöneltir.
- Bwapp Ortamı, bWAPP veya bir buggy web uygulaması, kasıtlı olarak güvenliksiz oluşturulmuş bir web uygulamasıdır. Güvenlik meraklılarının, geliştiricilerin ve öğrencilerin web güvenlik açıklarını keşfetmelerine ve önlemelerine yardımcı olur. bWAPP başarılı Sızma testi ve etik güvenlik kırma projeleri gerçekleştirmeye hazırlanmaya olanak sağlar. Bwapp'ı benzersiz yapan şey 100 web güvenlik açığına sahip olmasıdır. OWASP, ilk 10 projenin tüm riskleri dahil olmak üzere bilinen tüm büyük web hatalarını

kapsar. bWAPP, MySQL veri tabanı kullanan bir PHP uygulamasıdır. Linux / Windows üzerinde Apache / IIS ve MySQL ile barındırılabilir. Ayrıca WAMP veya XAMPP ile de kurulabilir. Başka bir olasılık, bwapp ile önceden kurulmuş özel bir Linux VM olan bee-box'ı indirmektir.

4.2. Gerçekleştirilmiş Testler

Tüm ortamlarda gerçekleştirilen testler ve sonuçları komple belirtilmiş olup; her ortamda sadece belirli testlerin gerçeklemeleri raporda yer almaktadır.
TestphpVulnweb Ortamında gerçekleştirilen manuel ve otomatik testler ile ilgili bulgular aşağıda yer almaktadır. Bu ortamda aşağıdaki testler hem manuel hem de otomatize araç kullanarak uygulanmıştır:

i) XSS
ii) SQL Enjeksiyonu
iii) Kayıp Güvenlik Başlıkları
iv) Alt Alan adı Devralma
v) Hassas Bilgi Sızıntısı
vi) Ana Bilgisayar Başlık Enjeksiyonu

Tablo 4.1: Manuel ve Otomatik Testlerin sonucunda tespit edilen zararlılar.

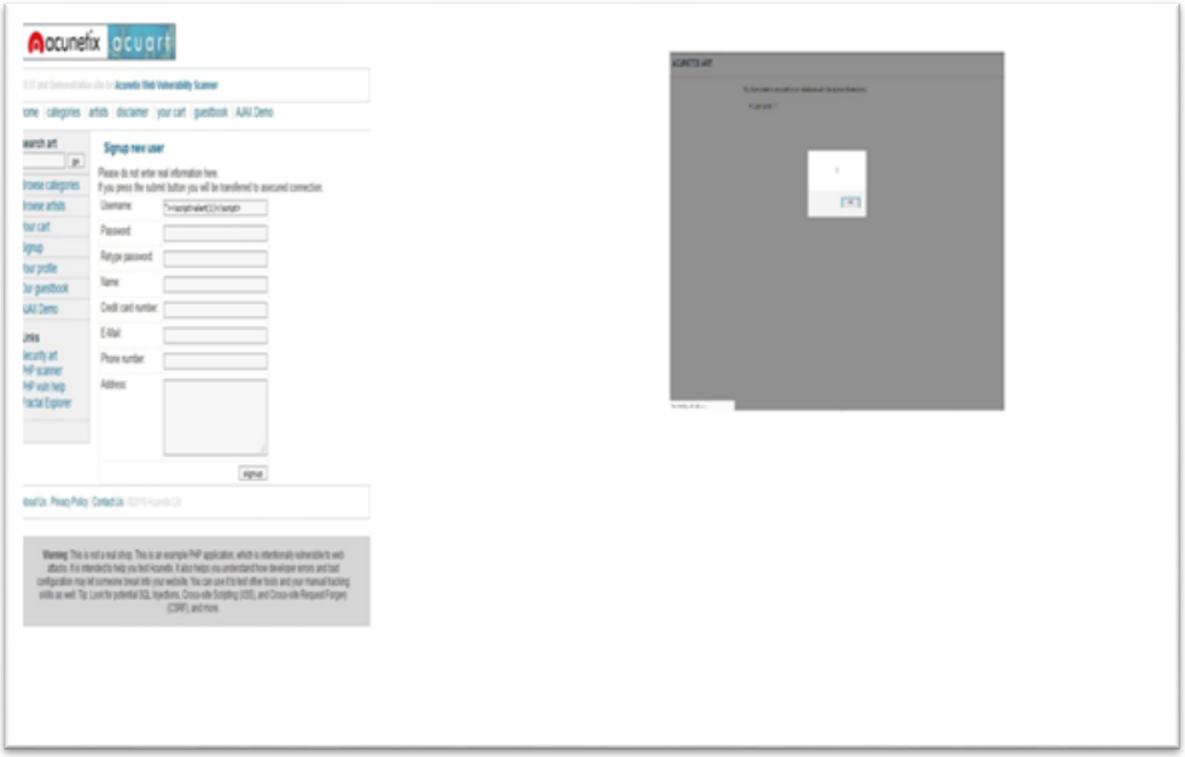
Zayıflık Adı	**Manuel Tarama**	**Otomatik Tarama**
XSS	Tespit Edildi	Tespit Edildi
SQL Enjeksiyonu	Tespit Edildi	Tespit Edildi
Kayıp Güvenlik Başlıkları	Tespit Edildi	Tespit Edildi
Alt Alan adı Devralma	Tespit Edildi	Tespit Edilmedi
Hassas Bilgi Sızıntısı	Tespit Edildi	Tespit Edildi
Ana Bilgisayar Başlık Enjeksiyonu	Tespit Edildi	Tespit Edilmedi

Manuel CSS Testi için şu adımlar uygulanmıştır:

i) Bu URL ziyaret edilir Web 10.

ii) uuname (kullanıcı adı) parametresine faydalı yükü (“><script>alert (1) </script>) eklenir.

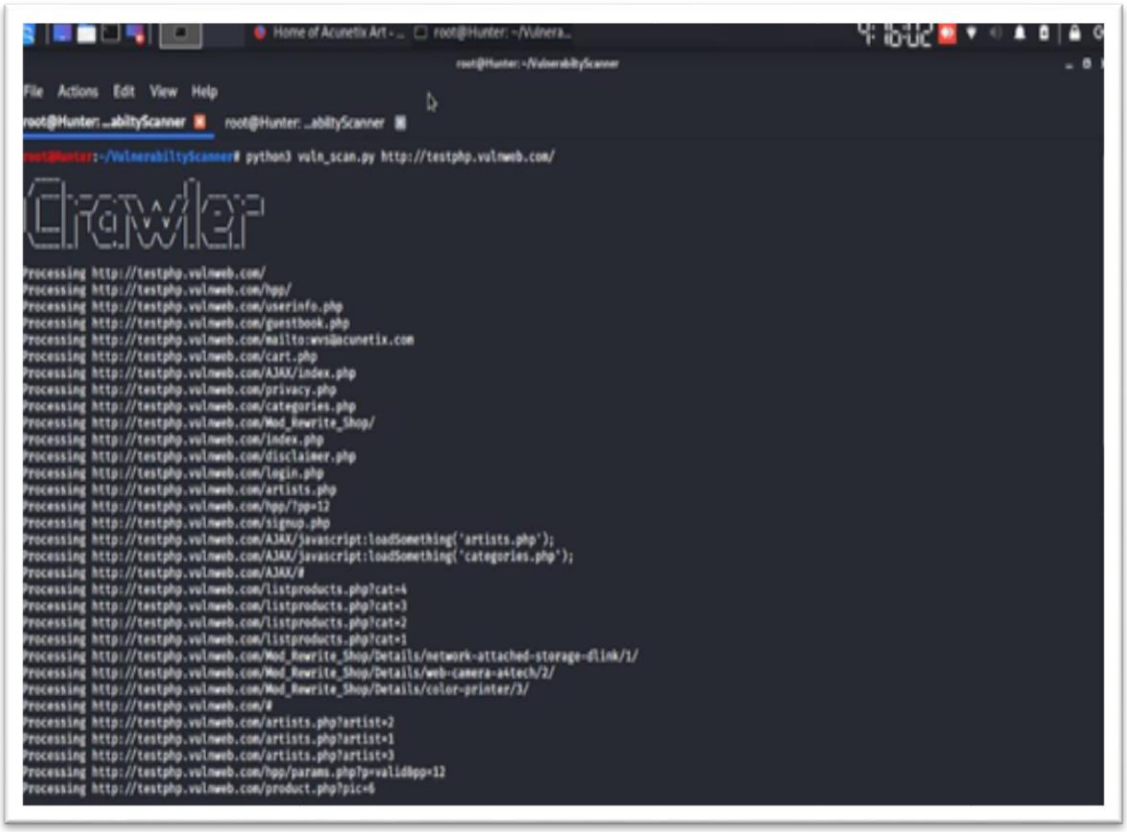
iii) Gönderildiğinde, çalıştığı görülür.



Şekil 4.1: Manuel olarak XSS zayıflığının tespiti ve çalıştırılması ardından elde edilen ekran görüntüleri.

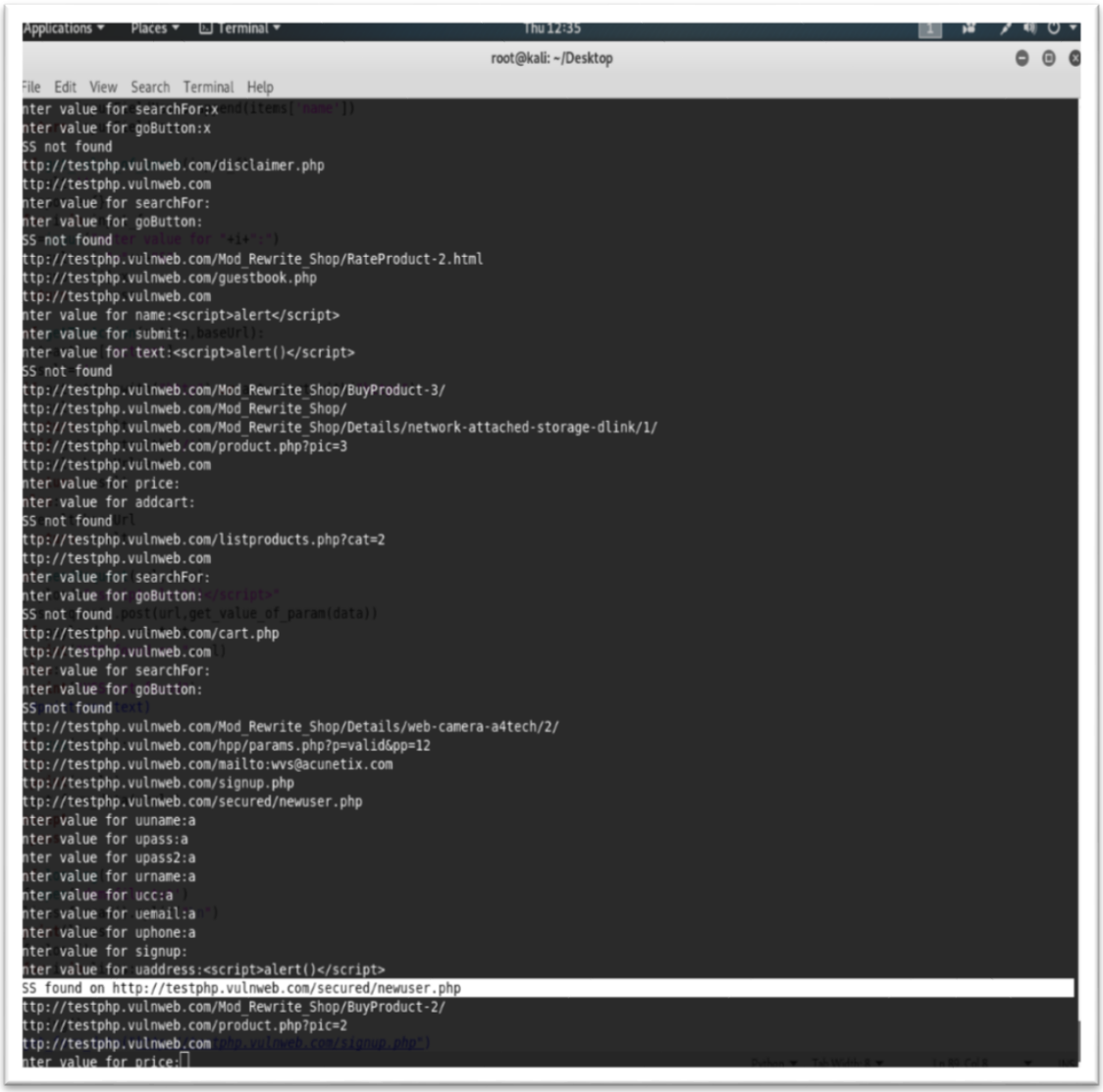
CSS Zafiyetinin Otomatize olarak test edilmesi gereken işlemler aşağıda yer almaktadır.

i) Terminalde aşağıdaki komutu çalıştırın
python3 attaxscan.pyhttp:Web 9



Şekil 4.2: Adım 1'deki komutların çalıştırılması ve sürecin başlangıcı.

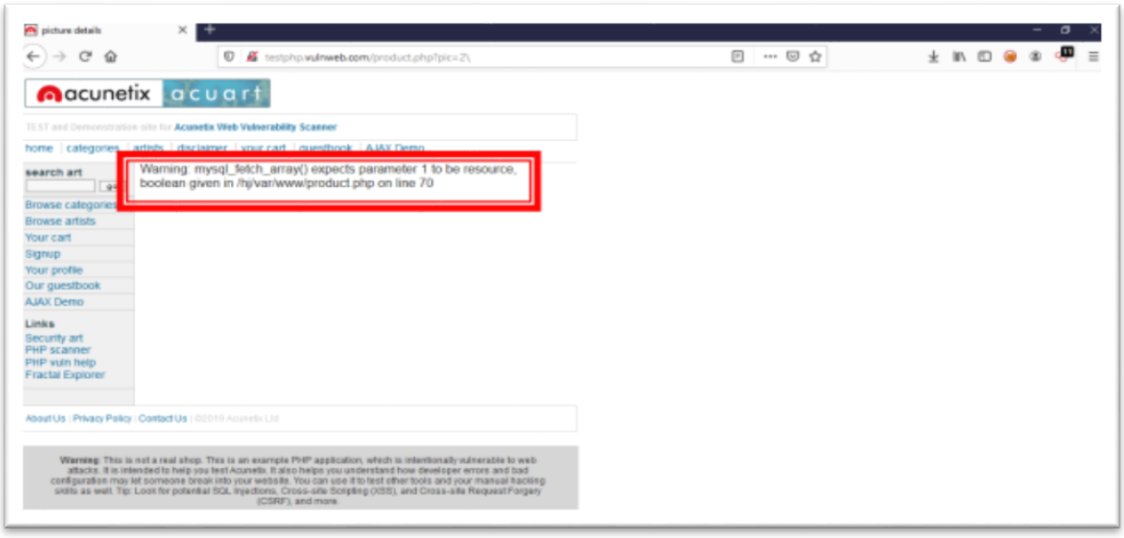
ii) Tarama bölümünü tamamladıktan sonra hangi tarayıcıyı çalıştırmak istediğinizi sorar (bu durumda 1 numara). Seçtiğinizde yürütülür ve sonuç ekranımızda olur.



Şekil 4.3: Otomatize Sızma Testi aracıyla XSS Zafiyeti tarama sonuçları.

SQL Enjeksiyonu Zafiyetinin Manuel olarak tespit edilmesi için gerekenler şu şekildedir

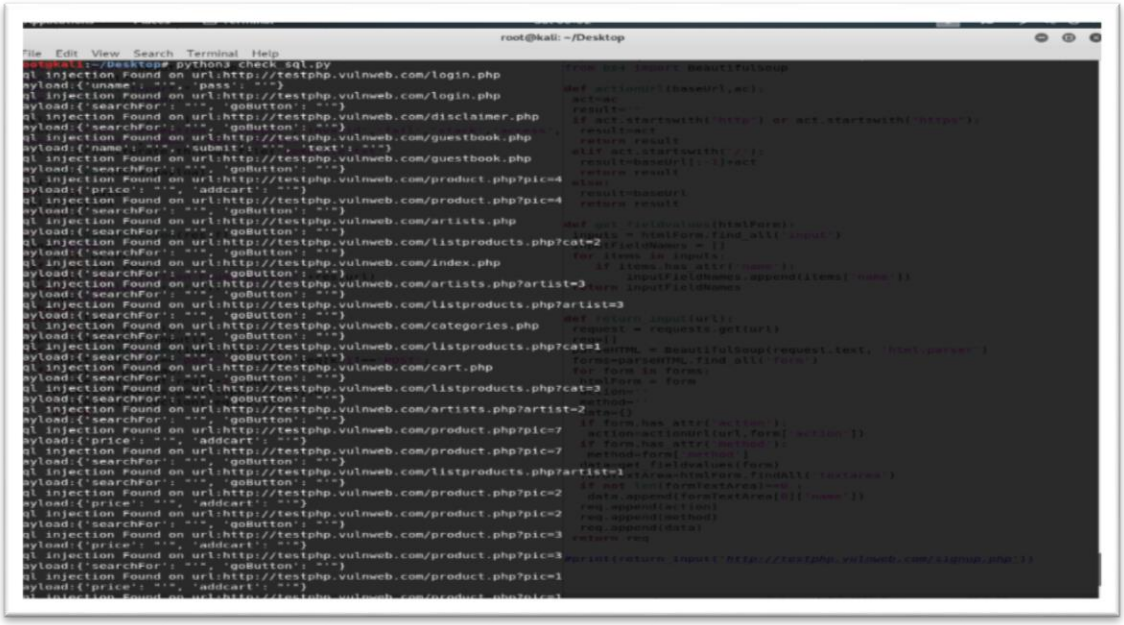
i) Bu URL ziyaret edilir Web 11.
ii) Ters eğik çizgi eklenir (\).
iii) SQL sözdizimini gösterecektir.



Şekil 4.4: Manuel olarak SQL Enjeksiyonu Zafiyetinin zafiyetin tespiti ve sistemin tepkisi.

SQL Enjeksiyon Zafiyetinin Tespiti' nin Otomatize edilmesi için gerekenler:

i) Crawling bölümünü tamamladıktan sonra hangi tarayıcının çalıştırılmak istendiğini sorar (bu durumda 2 numara). Seçildiğinde yürütülür ve sonuç ekranda gözükür.



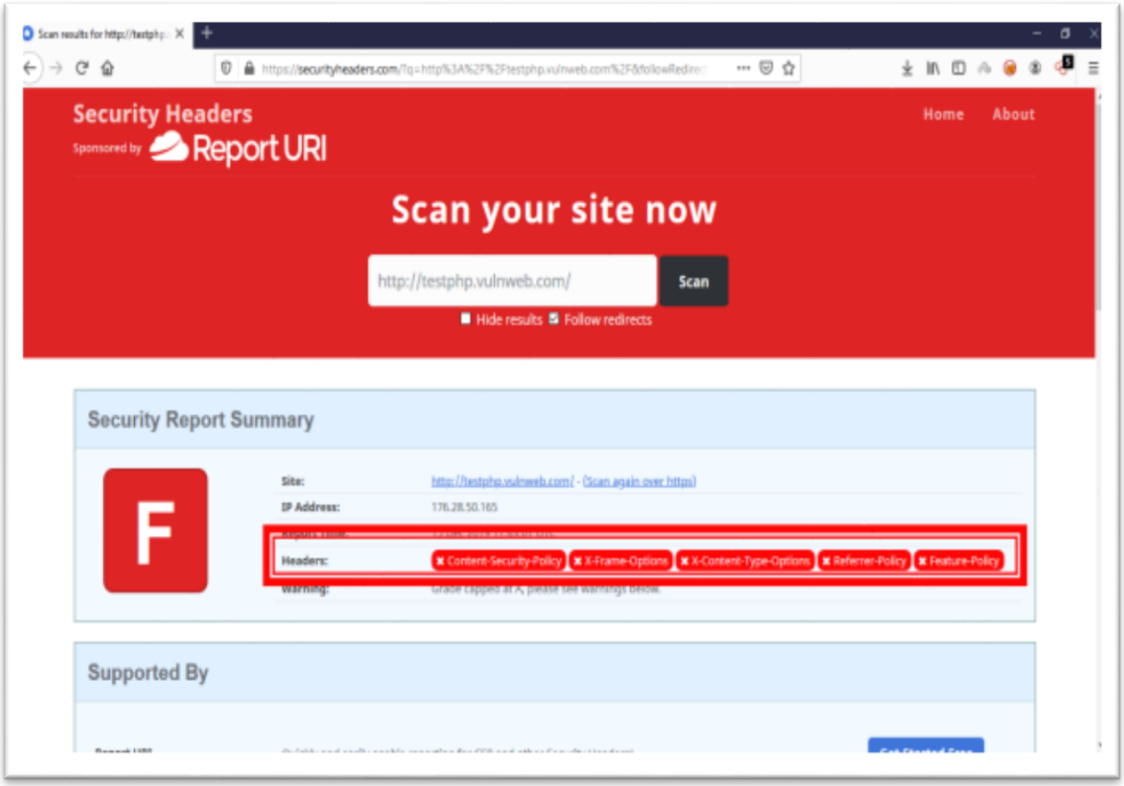
Şekil 4.5 : Otomatize araç ile SQL Enjeksiyon zafiyetinin tespiti.

Kayıp Güvenlik Başlığı Zafiyetinin Manuel olarak tespiti için gerekenler:

i) Bu URL ziyaret edilir Web 12.

ii) Orada "web sitenizi tarayın" bölümünü görülmektedir, web sitesi URL'i girilir ve tarama başlatılır.

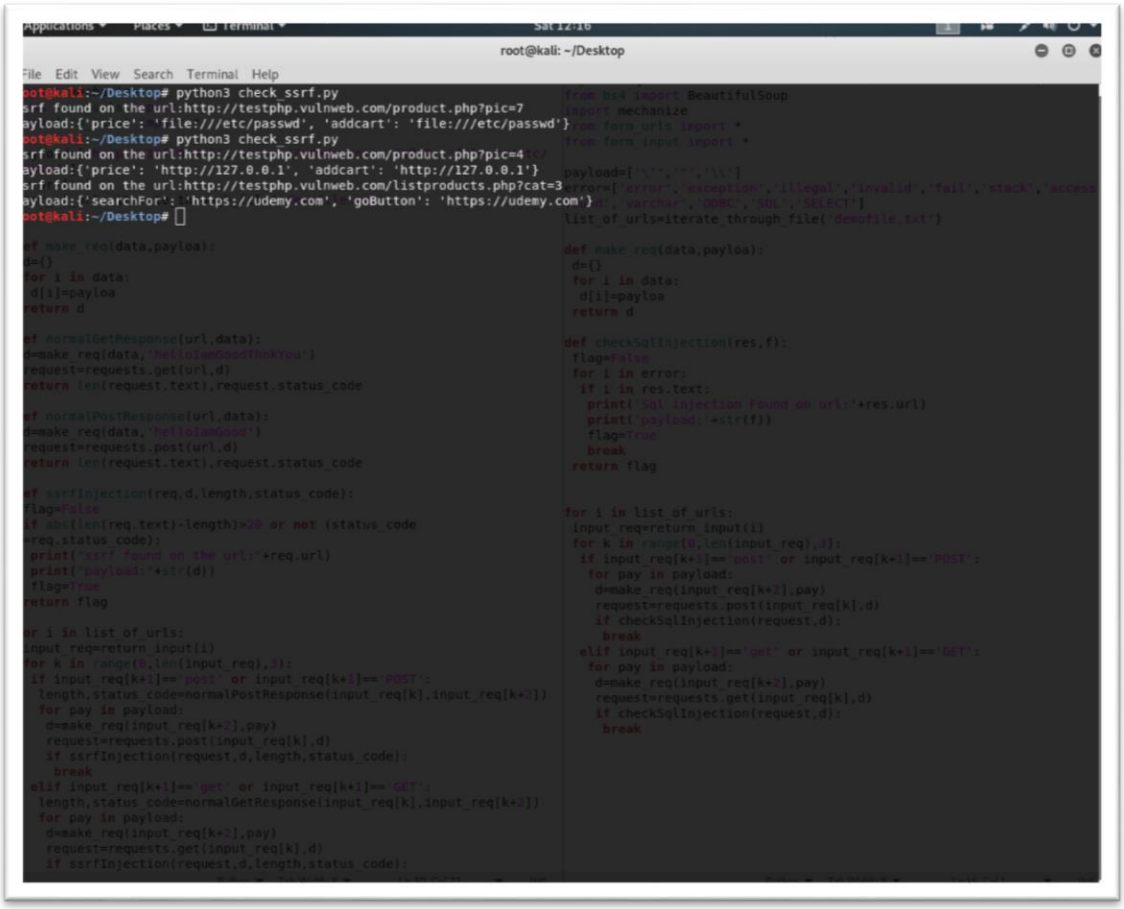
iii) Birkaç dakika sonra sonuçlar gözükecektir.



Şekil 4.6 : Manuel Kayıp Güvenlik Başlığı Zafiyetinin Tespiti.

Otomatize olarak Kayıp Güvenlik Başlığının Tespiti için gerekenler:

i) Aşağıdaki komutu python3 terminalinde çalıştırmak gerekir vuln_scan.py Web 9



Şekil 4.7 : Otomatize olarak Kayıp Güvenlik Başlığının Tespiti.

Demo Testfire Ortamında, aşağıdaki testler hem manuel hem de otomatize araç kullanarak uygulanmıştır:

i) XSS
ii) SQL Enjeksiyonu
iii) Kayıp Güvenlik Başlıkları
iv) Alt Alan adı Devralma
v) Hassas Bilgi Sızıntısı
vi) Ana Bilgisayar Başlık Enjeksiyonu

Tablo 4.2: Manuel ve Otomatik Testlerin sonucunda tespit edilen zararlılar.

Zayıflık Adı	Manuel Tarama	Otomatik Tarama
XSS	Tespit Edildi	Tespit Edildi
SQL Enjeksiyonu	Tespit Edildi	Tespit Edildi
Kayıp Güvenlik Başlıkları	Tespit Edildi	Tespit Edildi
Alt Alan adı Devralma	Tespit Edildi	Tespit Edildi
Hassas Bilgi Sızıntısı	Tespit Edildi	Tespit Edildi
Ana Bilgisayar Başlık Enjeksiyonu	Tespit Edildi	Tespit Edildi

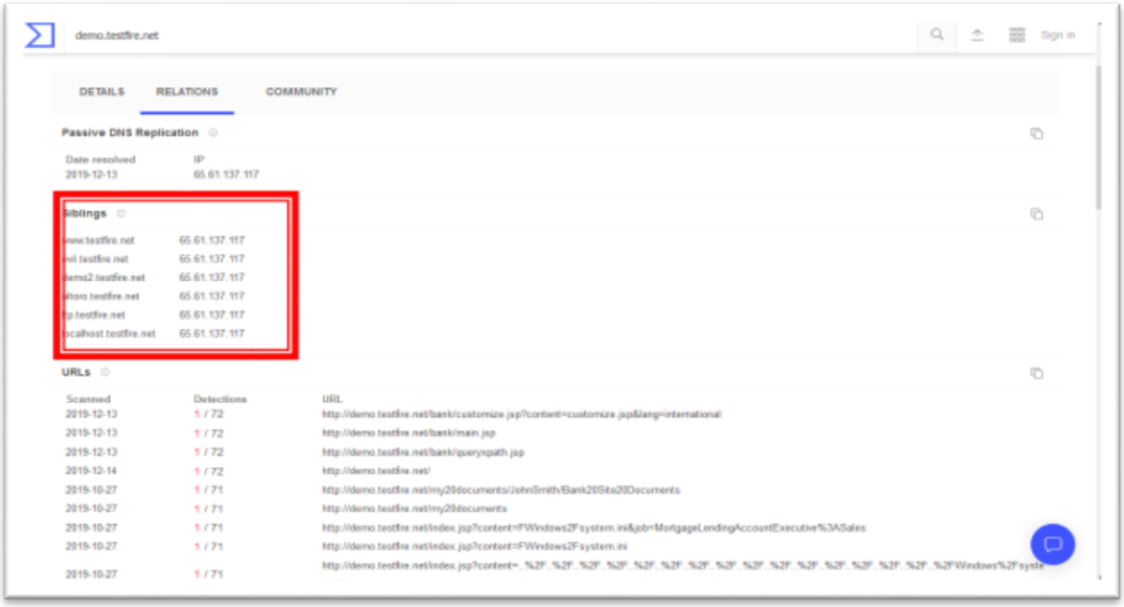
Alt Alan adı Devralma Zafiyeti'nin manuel ve otomatize araç ile gerçeklenmesi ve sonuçları aşağıda gösterilmektedir.

Manuel olarak yürütülen işlemler şu şekildedir:

i) Bu URL ziyaret edilir Web 13.

ii) Orada "web sitenizi tarayın" bölümünü görülür, web sitesi URL'si girilir ve arama yapılır.

iii) Sonuçlardan Web 14 URL'sini tıklanır.

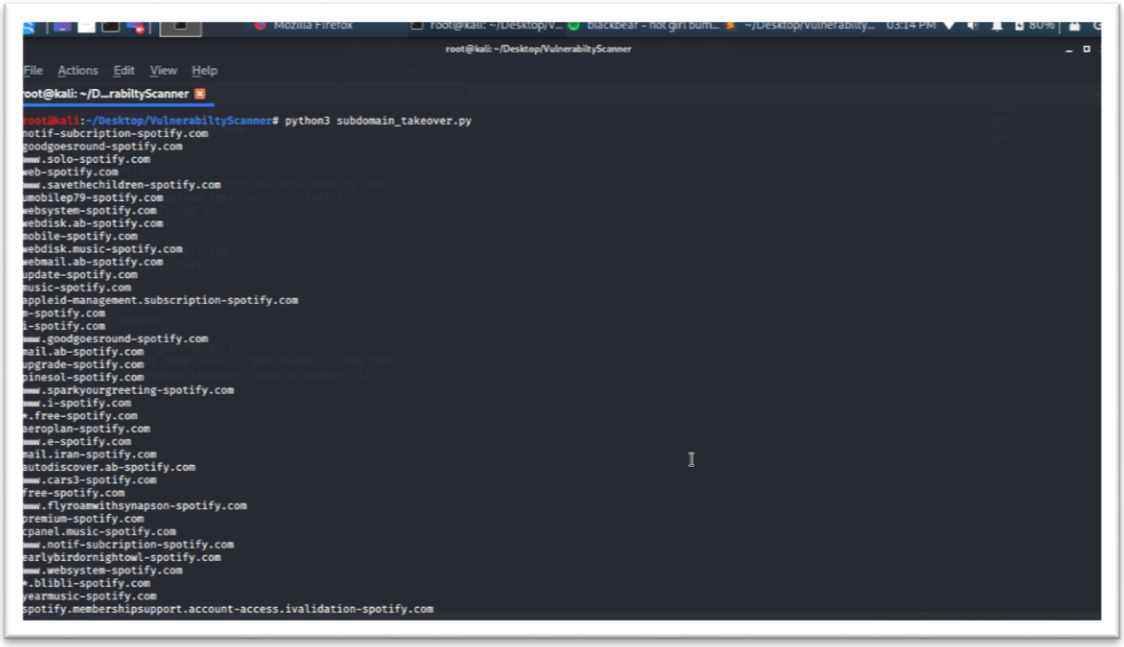


Şekil 4.8: Alt Alan adı devralma zafiyetinin manuel olarak tespiti.

Alt Alan adı devralma zafiyetinin Otomatize olarak tespit edilmesi için gereken adımlar şu şekildedir:

i) Terminalde bu komut çalıştırılır.

python3 vuln_scan.py Web 14



Şekil 4.9: Alt Alan adı devralma zafiyetinin Otomatize olarak tespitinde elde edilen çıktılar.

ZeroWebapp Security Ortamında aşağıdaki testler hem manuel hem de otomatize araç kullanarak uygulanmıştır:

- XSS
- SQL Enjeksiyonu
- XML Harici Varlık
- Kökenler arası kaynak paylaşımı
- Hassas Bilgi Sızıntısı
- Kayıp Güvenlik Başlıkları

Tablo 4.3: Manuel ve Otomatik Testlerin sonucunda tespit edilen zararlılar.

Zayıflık Adı	**Manuel Tarama**	**Otomatik Tarama**
XSS	Tespit Edildi	Tespit Edildi
SQL Enjeksiyonu	Tespit Edildi	Tespit Edildi
XML Harici Varlık	Tespit Edilmedi	Tespit Edildi
Kökenler arası kaynak paylaşımı	Tespit Edildi	Tespit Edildi
Hassas Bilgi Sızıntısı	Tespit Edilmedi	Tespit Edildi
Kayıp Güvenlik Başlıkları	Tespit Edildi	Tespit Edildi

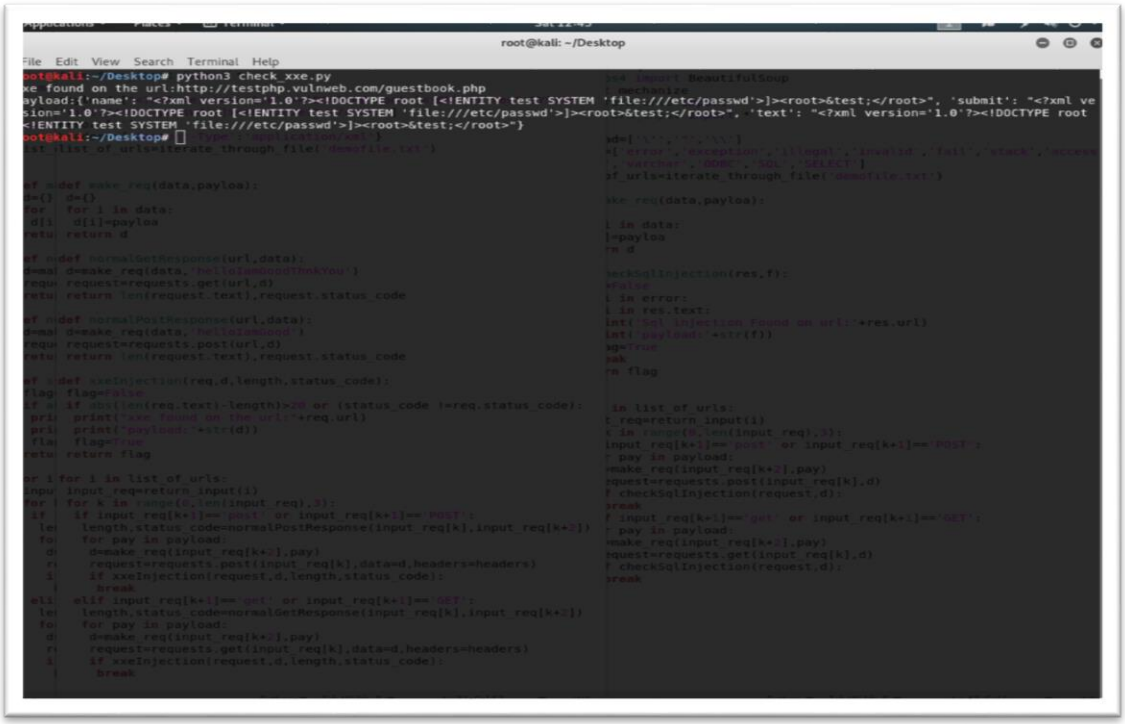
XXE Zafiyetinin Manuel olarak tespiti için yapılan işlemler esnasında Sonuç sağlanamamıştır. Manuel olarak tespit edilememiştir.

XXE Zafiyetinin Otomatik olarak tespiti için gerekenler

i) Aşağıdaki komutu çalıştırırız.

python3 attax_scan.py Web 15 "JSESSIONID = D437107A"

ii) 3 seçeneği “XXE” seçilmelidir.



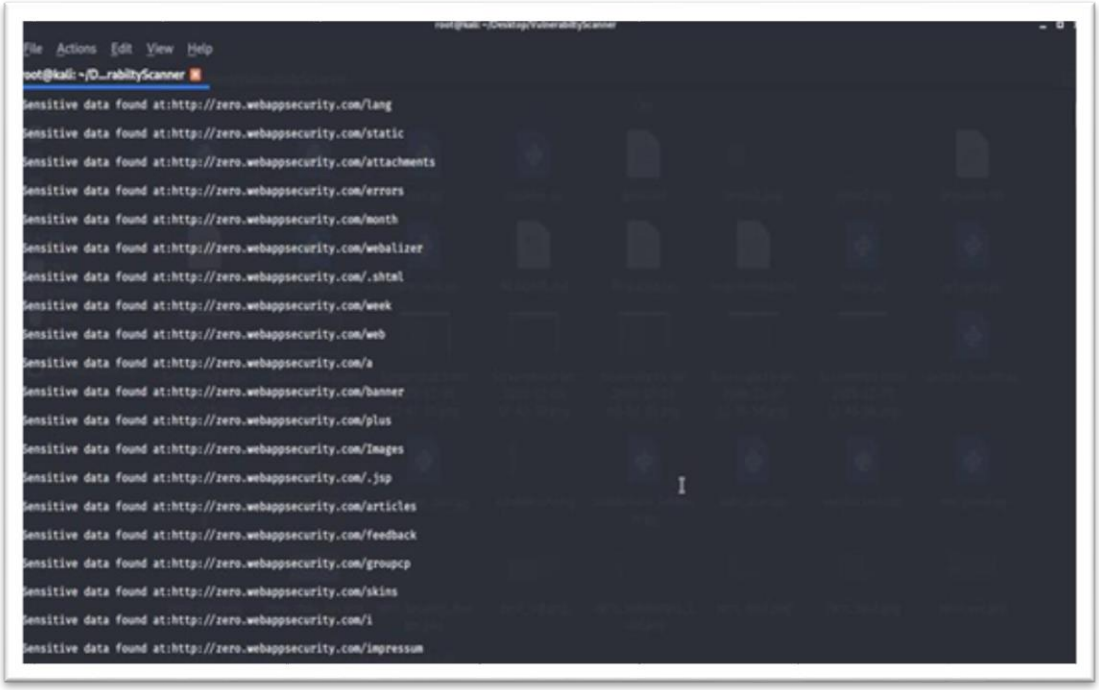
Şekil 4.10 : XXE Zafiyetinin Otomatik olarak tespitinde elde edilen çıktılar.

Hassas Veri Sızıntısı saldırısının gerçeklenmesi ve sonuçları aşağıda gösterilmektedir. Bu zayıflığın tespiti manuel olarak bu ortamda sağlanamamıştır.
Bu zayıflığın adaptif sızma testi aracıyla tespiti için gereken prosedür:

i) Aşağıdaki komut çalıştırılır:

python3 vuln_scan.py Web 15 "JSESSIONID=D437107A"

ii) 8. Basamakta yer alan “Sensitive Data Leak” seçilir ve tarama başlatılır.



Şekil 4.11 : Hassas Veri Sızıntısı zararlısının adaptif sızma testi aracında çalıştırılması ve sonucu.

Testhtml5Vulnweb Ortamında aşağıdaki testler hem manuel hem de otomatize araç kullanarak uygulanmıştır:

- XSS
- Kayıp Güvenlik Başlığı
- XML Harici Varlık
- Kökenler arası kaynak paylaşımı
- Hassas Bilgi Sızıntısı
- Ana Bilgisayar Başlık Enjeksiyonu

Tablo 4.4: Manuel ve Otomatik Testlerin sonucunda tespit edilen zararlılar.

Zayıflık Adı	**Manuel Tarama**	**Otomatik Tarama**
XSS	Tespit Edildi	Tespit Edildi
Ana Bilgisayar Başlık Enjeksiyonu	Tespit Edilmedi	Tespit Edildi
XML Harici Varlık	Tespit Edildi	Tespit Edildi
Kökenler arası kaynak	Tespit Edildi	Tespit Edildi
Paylaşımı Hassas Bilgi Sızıntısı	Tespit Edilmedi	Tespit Edildi
Kayıp Güvenlik Başlıkları	Tespit Edildi	Tespit Edildi

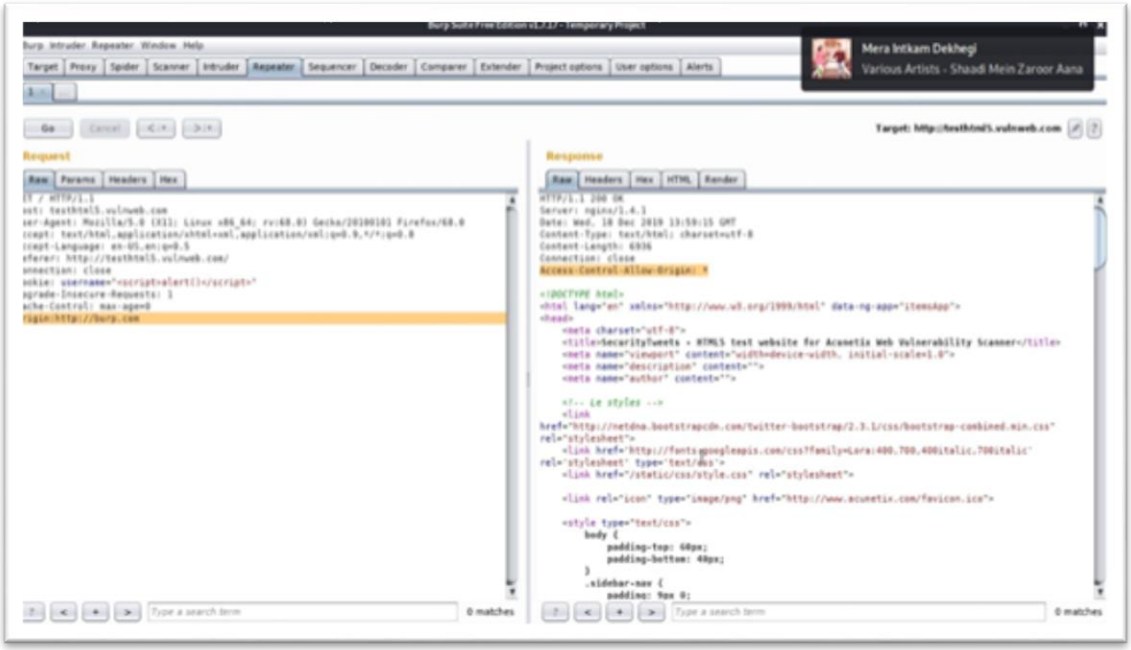
İlaveten CORS Zafiyetinin gerçeklenmesi şu şekilde sağlanmıştır.

Manuel olarak CORS Zafiyetinin Tespiti için :

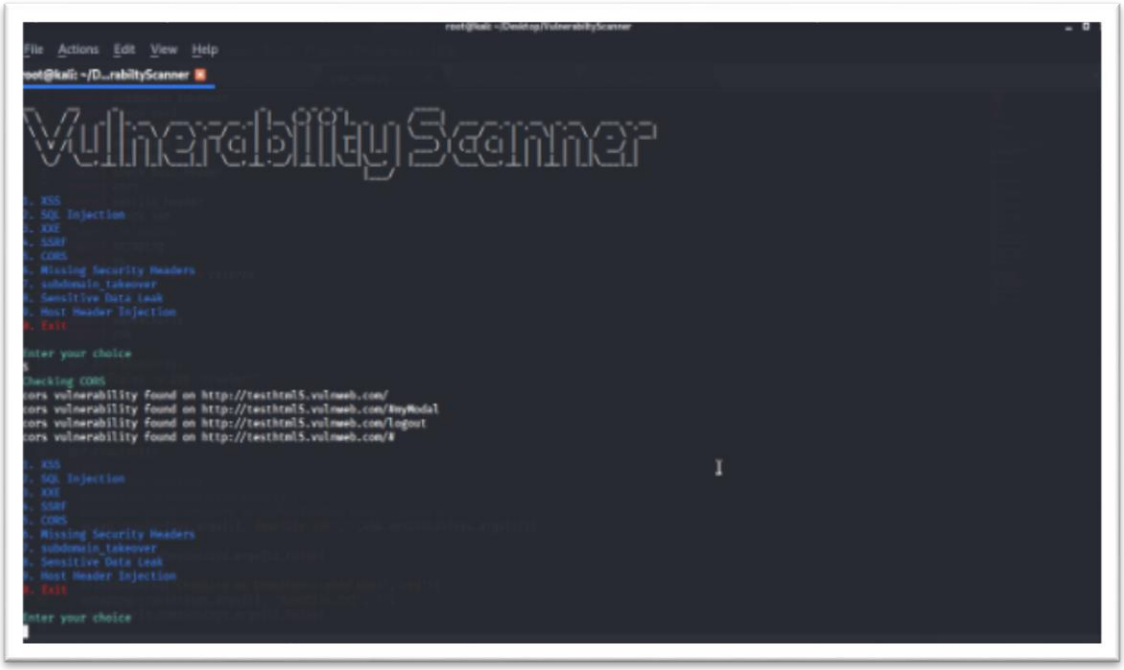
i) Web siteye giriş sağlanır.

ii) Bu URL ziyaret edilir Web 16 ve “burp” isteği tıklandı ve tekrarlayıcıya gönderildi.

iii) Başlangıç başlığı Web 17 olarak değiştirilir.



Şekil 4.12 : Manuel CORS zafiyetinin uygulanması ve çıktıları.



Şekil 4.13 : Otomatize olarak CORS Zafiyeti tespiti.

BWAPP Ortamında aşağıdaki testler hem manuel hem de otomatize araç kullanarak uygulanmıştır:

- XSS
- SQL Enjeksiyonu
- Kayıp Güvenlik Başlıkları
- Hassas Bilgi Sızıntısı
- Ana Bilgisayar Başlık Enjeksiyonu

Tablo 4.5: Manuel ve Otomatik Testlerin sonucunda tespit edilen zararlılar.

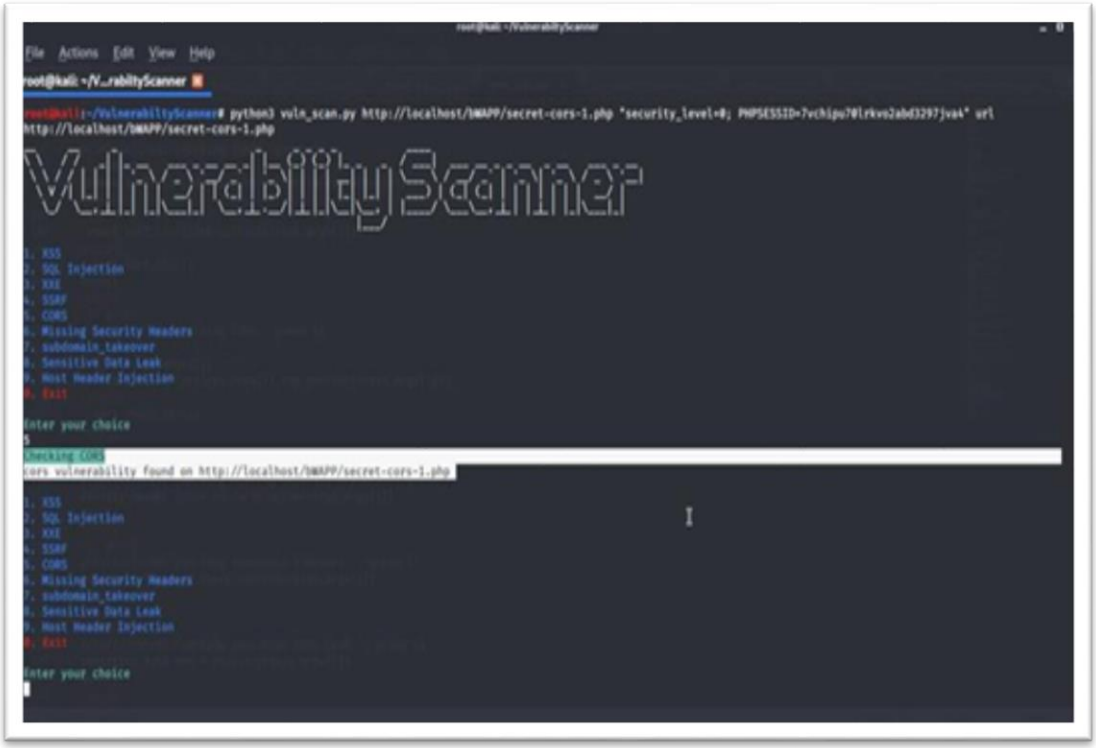
Zayıflık Adı	**Manuel Tarama**	**Otomatik Tarama**
XSS	Tespit Edildi	Tespit Edildi
SQL Enjeksiyonu	Tespit Edilmedi	Tespit Edildi
Kayıp Güvenlik Başlıkları	Tespit Edildi	Tespit Edildi
Ana Bilgisayar Başlık Enjeksiyonu	Tespit Edildi	Tespit Edildi
Hassas Bilgi Sızıntısı	Tespit Edilmedi	Tespit Edildi

Otomatize olarak tespit edilmesi için gerekenler

i) Savunmasız URL: Web 18

ii) Terminalde aşağıdaki komut çalıştırılır.
python3 vuln_scan.py Web 18 PHPSESSID=7vchipu70lrkvo2abd3297jva4" URL

iii) 3- 8 numaralı opsiyon seçildiğinde sonuçlar ekrana yansıyacaktır.



Şekil 4.14 : Hassas Veri Kaçağı zafiyetinin otomatize olarak tespiti ve çıktıları.

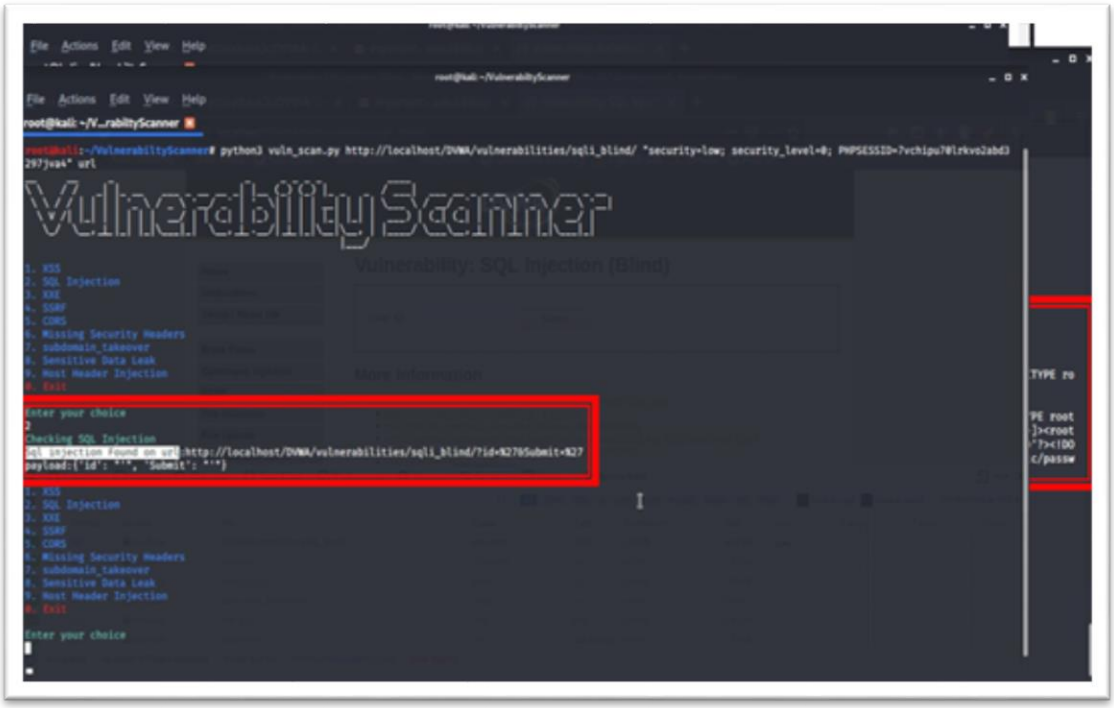
Damn Zararlı Web Uygulaması Ortamında, aşağıdaki testler hem manuel hem de otomatize araç kullanarak uygulanmıştır:

- XSS
- SQL Enjeksiyonu
- Kayıp Güvenlik Başlıkları
- Hassas Bilgi Sızıntısı
- Ana Bilgisayar Başlık Enjeksiyonu

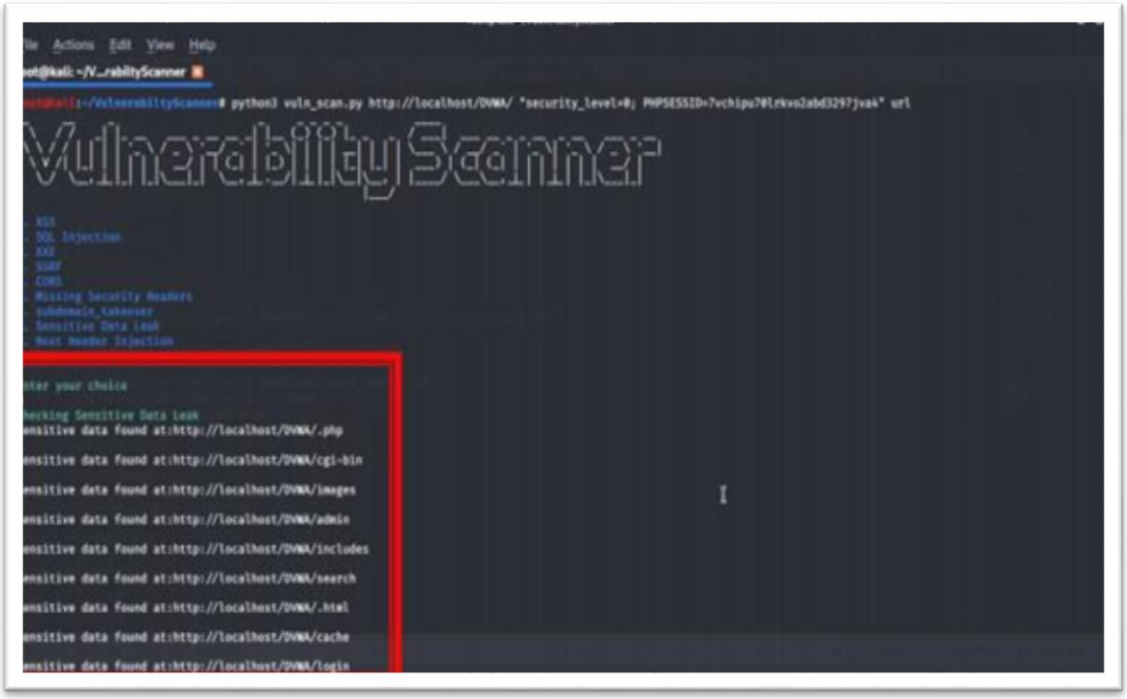
Tablo 4.6 : Manuel ve Otomatik Testlerin sonucunda tespit edilen zararlılar (DVA ortamı).

Zayıflık Adı	**Manuel Tarama**	**Otomatik Tarama**
XSS	Tespit Edildi	Tespit Edildi
SQL Enjeksiyonu	Tespit Edilmedi	Tespit Edildi
Kayıp Güvenlik Başlıkları	Tespit Edildi	Tespit Edildi
Hassas Bilgi Sızıntısı	Tespit Edilmedi	Tespit Edildi

Şekil 4.15 : XSS Zafiyetinin otomatize olarak tespiti.



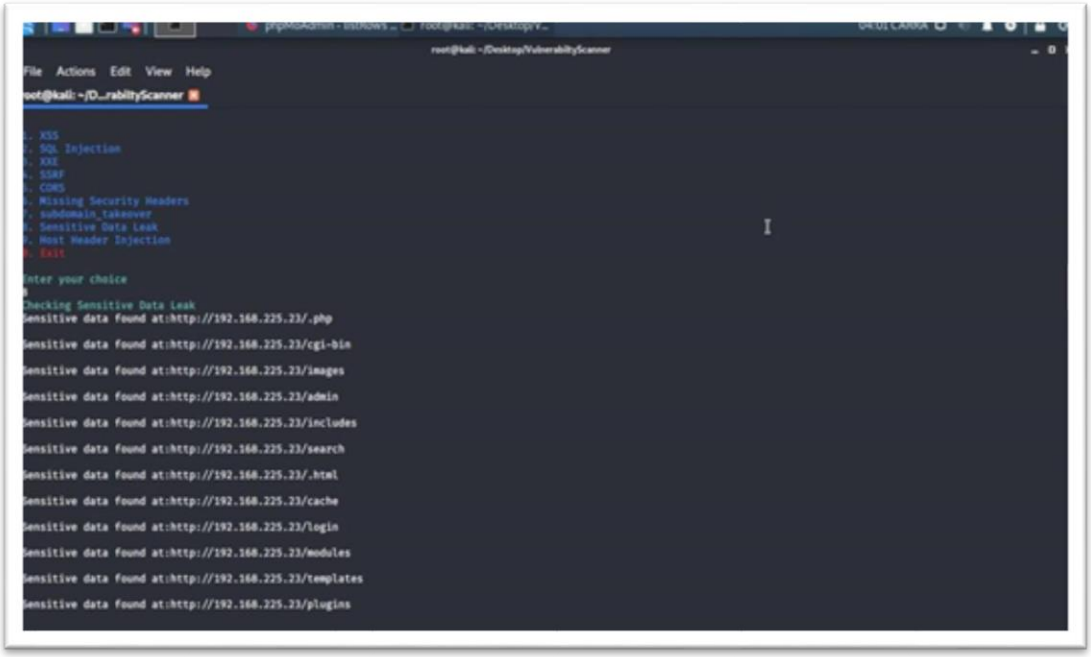
Şekil 4.16 : SQL Enjeksiyon zafiyetinin otomatize araç ile tespiti.



Şekil 4.17 : Hassas Veri Kaçağının otomatize araç ile tespiti.

Typhoon Zararlı Sanal Makinesi Ortamında, Kayıp Güvenlik Bayrağı ve Hassas Veri Kaçağı zayıflıklarının testleri hem manuel hem de otomatik olarak gerçekleştirilmiştir. Her iki türlüde tespit sağlanmıştır.

Şekil 4.18 : Kayıp Güvenlik Bayrağı zafiyetinin otomatize araç yardımıyla tespiti.



Şekil 4.19 : Hassas Veri Kaçağı zafiyetinin otomatize araç ile tespiti ve elde edilen çıktılar.

4.3. Değerlendirme

Testin yapıldığı ortamların çeşitliliği göz önüne alındığında, acil öneme sahip açıklıklar olarak anlatılabilecek varlıklar, vasıfsız saldırganların uzaktan gerçekleştirdiği ve sistemin tamamen ele geçirilmesiyle sonuçlanan açıklıklardır. Örneğin, XSS, SQL enjeksiyonu, müşteri bilgilerinin açığa çıkmasına neden olabilecek açıklık vektörleri gibi bankacılık uygulamalarında bu kategoriye girer. Özellikle, bu açıklıklar her ortamda yüksek verim ile tanımlanmıştır. Bilgi panelinde, görüntü değiştirme işlevinde bir zayıflık vardır. Yüklenen dosya gerekli denetimi geçmediğinden, kullanıcının profil alanına kötü amaçlı yazılım içeren bir dosyanın yüklenebileceği gözlenmiştir. Algılanan XSS güvenlik açıkları, uygulamalarda dış bilgiler için yeterli giriş ve çıkış denetimleri olmadığında ve kötü niyetli bir kullanıcı hedef kişilerin oturum bilgilerini çalmak ve istedikleri zaman hedef kişilerin tarayıcısını yeniden yönlendirmek için javascript kodu yürütebilir. Yakalanan kurbana, tarayıcıyı kullanarak dahili ağda port tarama, medya kaydı ve video kaydı yapabilir.

XSS' yi kontrol etmek için otomatik tarama manuel taramadan daha iyidir. Manuel taramada Sızma test cihazı tüm giriş alanından geçmeli ve daha sonra komut dosyasını manuel olarak enjekte etmelidir, bu da zaman alıcıdır. Ancak otomatik tarayıcı hepsini kısa sürede yapar. Manuel test, A'dan Z'ye kadar her şeyi kapsayamaz. Bu, zaman ve beceriler gibi bariz nedenlerden dolayı daha zordur, oysa otomatik araçlar bunu biraz insan müdahalesi ile yapabilir. Otomatik araçlar, tek bir test için gerekebilecek olan bin farklı taşıma yükü ile bile testi gerçekleştirebilmek amacıyla çok sayıda taşıma yükü ile daha fazla saldırı için bir hedefi test etmek için mükemmel bir seçimdir. Bu nedenle, otomatik araçlar geniş kapsamlı geliştirilebilir. Güvenlik açığı bulunan bir SQL Enjeksiyonu güvenlik açığı, bir saldırganın uygulama parametreleri aracılığıyla gönderilen bilgiler düzgün denetlenmediği için arka planda gönderilen bir veri tabanına sorgu eklemesine olanak verebilecek bir güvenlik açığıdır.

5. SONUÇLAR VE GELECEK ÇALIŞMALAR

Farklı araştırmacıların çalışmalarının ve sonuçlarının, otomatik sızma testi yazılımı geliştirirken kullanmayı düşündüğüm teknolojileri desteklendiği gözlenmiştir. Buna ek olarak, mimaride kullanılan python diline ait birçok güvenlik kütüphanesinin varlığı, yazılımın uygulama süresi üzerinde olumlu bir etkiye sahip olacaktır. Buna ek olarak, KALI işletim sistemindeki birçok araç göz ardı edilmemelidir. Çalışmanın bir sonraki aşaması, test işlemleri için geliştirilen tüm bu araçların entegre edilmesi olacaktır. Çalışma boyunca gösterilen test sonuçlarından elde edilen daha fazla çıktı ve kazanılan avantajlar aşağıdaki ifadelerin temelini oluşturmaktadır:

i) Otomatize Sızma testleri ile “daha az ve daha fazla” yapması gereken maliyet bilincine sahip bilgi güvenliği müdürleri.
ii) Katmanlı güvenlik sağlamak ve birden çok kaynaktan sonuçları doğrulamak için gereken uygulama güvenlik ekipleri.
iii) Geleneksel araçlarla ilişkili yanlış pozitiflerin sayısını azaltmak için uygulama güvenliğine ihtiyaç duyan devops ekipleri.
iv) Uygulamalardaki mevcut vurgulanan güvenlik açıklarının ayrıntılı bir “yol haritasından” yararlanacak olan "kırmızı takımlar".

KAYNAKLAR

[1] Epling l., Brandon H., HU Y., (2015), “Penetration Testing in a Box”, InfoSec '15: Proceedings of the 2015 Information Security Curriculum Development Conference, 1-4, Kennesaw Georgia, USA, 10-11 October.

[2] Goel J. N., B.M. Mehtreb, (2015), “Vulnerability Assessment & Penetration Testing as a Cyber Defence Technology”, 3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015), 710-715, Ghaziabad, Hindistan, 12-13 March.

[3] Ceccato M., Scandarito R., (2016), “Static Analysis and Penetration Testing from the Perspective of Maintenance Teams”, Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, 1450344275, 9781450344272, Association for Computing Machinery.

[4] Huber M., Kowalski S., Nohlberg M., Tjoa S., (2009), “Towards Automating Social Engineering Using Social Networking Sites”, CSE '09: Proceedings of the 2009 International Conference on Computational Science and Engineering, 2159-2165, Vancouver, Canada, 29-31 August.

[5] K. Krombholz, H. Hobel, M. Huber E. Weippl , (2014), “Advanced social engineering attacks ”, Journal of Information Security and Applications, 22, 113 – 122.

[6] Web 1, 2015, https://docs.python.org/2/library/os.html , (Erişim Tarihi: 15/5/2015).

[7] Web 2, 2015, https://www.python.org//, (Erişim Tarihi: 16/5/2015).

[8] Ghanem M.C., Chen, T. M., (2020), “Reinforcement Learning for Efficient Network Penetration Testing”, Information, 11(1), 6.

[9] Qiu, X.; Jia, Q., Wang, S., Xia, C.; Shuang, L., (2014), “Automatic generation algorithm of penetration graph in penetration testing”, Ninth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, 531-537, Guangdong, China, 8-10 November.

[10] Heinl C., (2014), “Artificial (intelligent) agents and active cyber defence: Policy implications”. In Proceedings of the 6th International Conference On Cyber Conflict (CyCon 2014), 53-66, Tallinn, Estonia, 3–6 June.

[11] Web 3, (2019), https://arxiv.org/abs/1307.8182, (Erişim Tarihi: 20 Aralık 2019).

[12] Web 4, (2017), https://arxiv.org/abs/1705.05088, (Erişim Tarihi: 23 Aralık 2019).

[13] Almubairik, N.; Wills, G., (2016),"Automated penetration testing based on a threat model", In Proceedings of the 11th International Conference for Internet Technologies and Secured Transactions, ICITST, 413-414, Barcelona, Spain, 5–7 December.

[14] Web 5, (2013), https://arxiv.org/abs/1306.4044, (Erişim Tarihi: 14 Nisan 2019)

[15] Web 6, (2019), https://arxiv.org/abs/1705.05088, (Erişim Tarihi: 21 Mayıs 2019)

[16] Web 7, (2019), https://nvd.nist.gov , (Erişim Tarihi: 18 Aralık 2019).

[17] Web 8, (2019), https://cve.mitre.org , (Erişim Tarihi: 18 Aralık 2019).

[18] Web 9, (2018), http://testphp.vulnweb.com//, (Erişim Tarihi: 24/11/2018).

[19] Web 10, (2018), http://testphp.vulnweb.com/signup.php, (Erişim Tarihi: 24/11/2018).

[20] Web 11, (2018), http://testphp.vulnweb.com/product.php?pic=2, (Erişim Tarihi: 24/11/2018).

[21] Web 12, (2018), https://securityheaders.com//, (Erişim Tarihi: 27/11/2018).

[22] Web 13, (2018), https://www.virustotal.com//, (Erişim Tarihi: 30/11/2018).

[23] Web 14, (2018), https://demo.testfire.net//, (Erişim Tarihi: 30/11/2018).

[24] Web 15, (2019), http://zero.webappsecurity.com/bank/accountsummary.html, (Erişim Tarihi: 21/05/2019).

[25] Web 16, (2019), http://testhtml5.vulnweb.com//, (Erişim Tarihi: 23/05/2019).

[26] Web 17, (2019), www.bing.com//, (Erişim Tarihi: 13/0/2019).

[27] Web 18, (2019), http://localhost/bwapp.php "security_level=0 //,(Erişim Tarihi: 21/06/2019).

[28] Web 19, (2020), https://crt.sh/?q=%25 , (Erişim Tarihi: 24/06/2019)

ÖZGEÇMİŞ

Volkan DÖRTKARDEŞ 1986 yılında Makedonya-Üsküp'te doğdu. İstanbul Aydın Üniversitesi Bilgisayar Mühendisliği Bölümünü Bölüm 3.sü olarak yüksek başarıyla tamamlayarak aynı yıl Gebze Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı Siber Güvenlik Yüksek Lisans programına başladı. 2009 yılından bu yana Bilgisayar Mühendisi olarak uluslararası finansal kuruluşlarda Ağ ve Uygulama Güvenliği Uzmanı olarak çalışmıştır. 2019 yılından bu yana kendi Siber Güvenlik şirketinde büyük kuruluşlara servis sağlamaktadır.

EKLER

Ek A: Tez Çalışması Kapsamında Yapılan Yayınlar

İbrahim SOGUKPINAR, Volkan DORTKARDES, (2020), “Adaptive Penetration Test Method”, International Journal of Innovative Science and Research Technology, 5th Volume, 1295-1304.