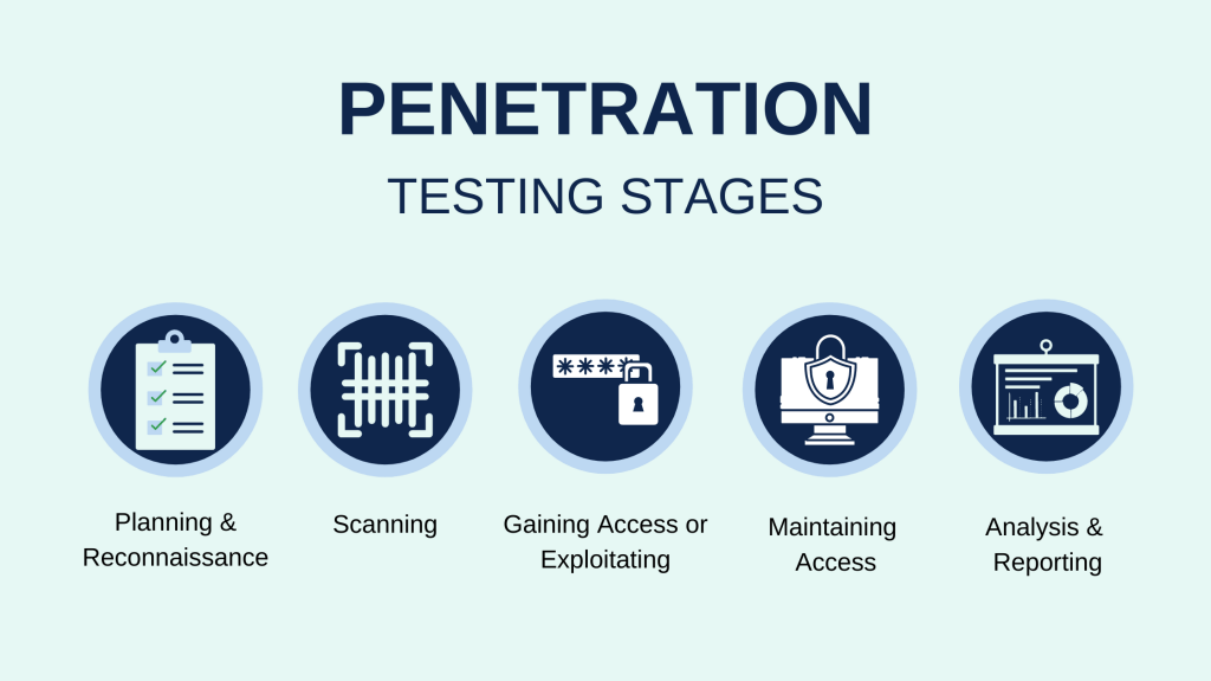
17-18.DÖNEM PENTEST SINAVI/Bilişim Academy

ZEHRA NUR MANGAL /zehranurmangal@gmail.com

SORU1: SIZMA TESTİNİN TANIMINI,AMACINI VE METODOLOJİLERİNİ TANIMLAYINIZ.

Sızma testi, bir kuruluşun güvenlik açıklarını belirlemek için sistematik olarak gerçekleştirilen bir çalışmadır .Bir saldırganın yararlanabileceği yerel ağ, internet ya da web uygulaması gibi muhtemel saldırı yüzeylerinde bulunabilecek güvenlik açıklarını veya bir yazılım uygulamasındaki zayıflıkları ve riskleri ortaya çıkaracak bir tür testtir. sızma testi, bir kuruluşun siber güvenlik açıklarının bulunması ve kuruluşların savunma mekanizmalarının kalitesinin etik olarak test edilmesinin kapsamlı bir yoludur.



1.Kapsam Belirlenmesi
Müşteri, testin yapılmasını istediği hedefi/kapsamı belirler. Testin yaklaşım türüne göre (Black Box, White Box, Gray Box) testi yapacak olan firma ile bilgiler paylaşılır.
2. Bilgi Toplama
Kapsam/Hedef hakkında pasif (sistem ile doğrudan etkileşime geçmeden) ve aktif (sistem ile doğrudan etkileşime geçerek) bilgi toplama işlemi gerçekleştirilir. Bunlara; kullanılan teknoloji, uygulama ve versiyon bilgisi, fonksiyonlar gibi bilgiler örnek gösterilebilir.

3. Güvenlik Açığı Tespiti
Toplanan bilgiler ışığında var olan güvenlik açıklıklarının belirlendiği aşamadır. Otomatize araçlar kullanılarak taranan sistemler, tarama sonrasında/esnasında uzmanlar tarafından manuel olarak test edilir. Bilgi toplama aşamasında tespit edilen servis ve versiyon bilgisi araştırılarak var olan bir güvenlik açığı olup olmadığı kontrol edilir.
4. Bilgilerin Analizi ve Planlama
Tespit edilen güvenlik açıklıklarının sömürülmesi için gerekli araştırmalar yapılarak sömürü kodları, zararlı yazılımlar gibi ofansif araçlar hazırlanır.
5. Sömürü Aşaması
Tespit edilen zafiyetler saldırgan bakış açısı ile sömürülmeye çalışılır ve zafiyetin sistem üzerindeki etkileri incelenir. Saldırgan, sisteme yetkisiz giriş yapabiliyor mu? Servisi durdurabiliyor mu? Gibi sorulara cevap aranır.
6. Yetki Yükseltme/Sömürü Sonrası Aşama
Saldırgan sisteme erişim elde ettikten sonraki aşamada halihazırdaki yetkilerini yükseltebilecek mi? Yetkisi olmayan dosyaları görebilecek mi? Veya sızılan sistem/ler kullanılarak nasıl ilerlenebilir? Ne gibi kritik dosyalara erişim sağlanabilir? Gibi sorulara yanıt aranır. Saldırganın sömürü sonrası yapacağı teknik/taktik/prosedürler simüle edilmeye çalışılır.
7. Temizlik
Test edilen sistemlerde yapılan değişiklikler geri alınır. Test için oluşturulan/yüklenen dosyalar sistemden temizlenir.
8. Raporlama
Yukarıdaki adımların özeti çıkarılır. Var olan veya ileride oluşabilecek potansiyel riskler, alınması gereken önlemler gibi bilgiler raporlanır.

SORU2:FARKLI SIZMA TESTİ TEKNİKLERİNDEN BAHSEDİNİZ.

Penetrasyon testi üç kategoriye ayrılır: kara kutu testi, beyaz kutu testi ve gri kutu testi. Kategoriler farklı saldırı türlerine veya siber güvenlik tehditlerine karşılık gelir.

Kara kutu testi **(BlackBox):** *kaba kuvvet saldırısıyla ilgilidir. Bu senaryoda simülasyon, bir şirketin BT altyapısının karmaşıklığını ve yapısını bilmeyen bir bilgisayar korsanının simülasyonudur. Bu nedenle, bilgisayar korsanı bir zayıflığı tespit edip kullanmaya çalışmak için topyekün bir saldırı başlatacaktır. Penetrasyon testi, testçiye bir web uygulaması, kaynak kodu veya herhangi bir yazılım mimarisi hakkında herhangi bir bilgi vermez. Test uzmanı, BT altyapısındaki güvenlik açıklarının nerede bulunduğunu görmek için "deneme yanılma" yaklaşımını kullanır. Bu tür sızma testleri, gerçek dünya senaryosunu en iyi şekilde taklit eder, ancak tamamlanması uzun zaman alabilir.*

Beyaz kutu **(WhiteBox:)** *penetrasyon testi bu ilk tekniğin tam tersidir. Beyaz kutu testinde test uzmanı, bir web uygulamasının kaynak koduna ve yazılım mimarisine erişimle birlikte BT altyapısı hakkında tam bilgiye sahiptir. Bu onlara sistemin belirli bölümlerine odaklanma ve hedeflenen bileşen testi ve analizini gerçekleştirme yeteneği verir. Kara kutu testinden daha hızlı bir yöntemdir. Ancak beyaz kutu sızma testi, yazılım kodu analizörleri veya hata ayıklama programları gibi daha karmaşık kalem testi araçlarını kullanır.*

Gri kutu testi **(GreyBox):** *, test uzmanının dahili BT altyapısı hakkında kısmi bilgiye sahip olduğu bir senaryoda hem manuel hem de otomatik test süreçlerini kullanır. Test cihazı örneğin yazılım kodunu alabilir ancak sistem mimarisi ayrıntılarını alamayabilir. Gri kutu sızma testi, beyaz kutu ve kara kutu testinin bir melezidir ve kullanıcının manuel çabalarını "güvenlik açıklarını" bulmaya odaklarken, topyekün saldırıda otomatik araçlardan yararlanmasına olanak tanır.*

Bu kapsayıcı penetrasyon testi yöntemleri türleri ayrıca belirli kategorilere ayrılabilir. Diğer penetrasyon testi türleri şunları içerir:

***Sosyal mühendislik testleri**: Kalem testi senaryosu, bir çalışanın veya üçüncü tarafın şifre, iş verileri veya diğer kullanıcı verileri gibi hassas bilgileri açığa çıkarmasını sağlamaya çalışır. Bu, telefon veya internet aracılığıyla yardım masalarını veya satış temsilcilerini hedefleyerek yapılabilir.*

***Web uygulaması testleri**: Kalem testi, web uygulamalarının ve yazılım programlarının güvenlik açıklarını değerlendirmek için yazılım kullanır.*

***Fiziksel sızma testleri:** Çoğunlukla devlet sitelerinde veya diğer güvenli tesislerde kullanılan kalem testi, sahte bir güvenlik ihlaliyle fiziksel ağ cihazlarına ve erişim noktalarına erişmeye çalışır.*

***Ağ hizmetleri testi**: Bu, kullanıcının ağdaki açıklıkları yerel olarak veya uzaktan tanımlamaya çalıştığı en yaygın kalem testi senaryosudur.*

***İstemci tarafı testi**: Bu, bir MSP'nin istemci tarafı yazılım programlarındaki güvenlik açıklarından yararlanmaya çalışmasıdır.*

***Kablosuz güvenlik testi**: Kalem testi açık, yetkisiz veya düşük güvenlikli erişim noktalarını ve Wi-Fi ağlarını tanımlar ve bunlara sızmaya çalışır.*

Her türlü sızma testi, bir BT altyapısının hem iç hem de dış bileşenlerini dikkate almalıdır. Bir kuruluşun siber güvenliğine bütünsel ve düzenli olarak güncellenen bir yaklaşım sağlayacak bir sızma testinin farklı aşamaları vardır.

SORU3: SQL INJECTION'UN NE OLDUĞUNU DETAYLI BİR ŞEKİLDE AÇIKLAYINIZ.VE SORGULARLA ÖRNEKLER VERİNİZ.

SQL Injection (SQLi), bir web uygulamasına veya veritabanına kötü niyetli bir şekilde veri enjekte ederek veya var olan sorguları manipüle ederek veritabanı sorgularını çalıştırmak veya verileri çalmak için kullanılan bir saldırı türüdür. SQL Injection farklı varyasyonlara sahiptir ve bunlardan üçü "In-band SQL Injection,"Inferential (Blind) SQL Injection," ve "Out-of-Band SQL Injection" olarak tanımlanır.

In-Band SQL Injection (Classic SQLi):

In-Band SQL Injection, en yaygın ve basit SQL enjeksiyon türlerinden biridir. Saldırganlar, veritabanına saldırı yapmak için aynı kanalı (band) kullanır. Bu tür saldırıda, saldırganlar genellikle hata mesajları veya uygulamanın yanıtını kullanarak sorguları manipüle ederler.

Örnek:Saldırgan, kullanıcı adı ve şifre girişlerini bozmak için aşağıdaki gibi bir giriş yapabilir:

Kullanıcı Adı: ' OR '1'='1

Şifre: ' OR '1'='1

SELECT * FROM users WHERE username = '' OR '1'='1' AND password = '' OR '1'='1';

Inferential (Blind) SQL Injection:

Inferential SQL Injection, uygulamanın hata mesajları veya yanıtları olmadan çalışır ve saldırganlar sonuçları çıkarır. Bu tür saldırılar, sorgu sonuçlarını dışarıya sızdırmadan yapılır. Salıncağın anahtarı, sorguların sonucunu gözlemlemek ve veri çalmaktır.

Örnek:Saldırgan, bir kullanıcının parolasını tahmin etmek için şu şekilde bir giriş yapabilir:

Kullanıcı Adı: test' AND 1=CONVERT(int, (SELECT @@version))--

Bu giriş, veritabanında SQL sorgularını tetikler ve sonucu gözlemleyerek SQL sorgusu sonuçlarını çıkarır.

Out-of-Band SQL Injection:

Out-of-Band SQL Injection, sorgu sonuçlarını başka bir iletişim kanalı üzerinden alır. Yani, uygulama tarafından verilerin sızdırılması için kullanılan bir kanal ile sorgu sonuçları farklı bir kanal üzerinden alınır. Örnek:Saldırgan, bir web uygulamasının sorguları tetiklemesi için aşağıdaki gibi bir giriş yapabilir:

Kullanıcı Adı: '; EXEC xp_cmdshell('nslookup example.com')--Bu giriş, DNS sorgusunu tetikler ve sonuçları, sızdırılmış veri olarak kullanılabilir.

ÖRNEK:

SELECT * FROM accounts WHERE username="UNION SELECT 1,database(),version(),user(),5,6,7#' AND password="

SELECT * FROM accounts WHERE username=" bu kısım, kullanıcı adı (username) ve şifre (password) alanlarına erişim sağlar ve kullanıcı adı boş olarak belirtilir. Ancak, kullanıcı adının boş bırakılması, asıl amacın bu sorgu üzerinden başka verilere ulaşmak olduğunu gösterir.

'UNION SELECT 1, database(), version(), user(), 5, 6, 7 Bu kısım, SQL Injection'ın ana bölümüdür. UNION SELECT ifadesi, veritabanı sorgusunun sonucuna ek bir sorgu sonucunu ekler. Bu durumda, aşağıdaki verileri getirir:

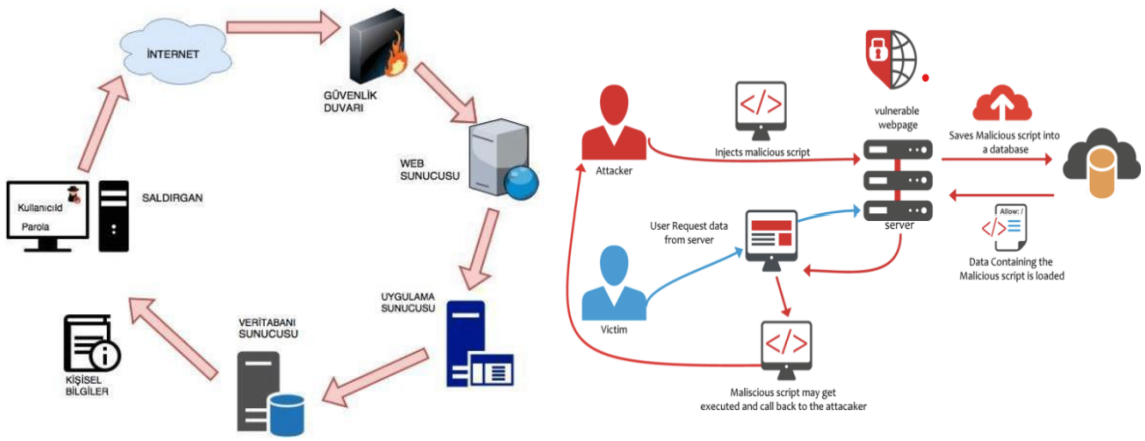
1: İlk kolon olarak sabit bir sayıdır.

database(): Veritabanının adını alır.

version(): Veritabanı sürümünü alır.

user(): Veritabanı kullanıcısının adını alır.

#' Bu işaret, SQL yorumunu sonlandırır. SQL sorgularında # işareti, yorum satırını temsil eder ve sorgu sonrası kısım (örneğin, orijinal şifre sorgusu) görmezden gelinir. **AND password**=" Bu bölüm, şifre alanı için bir koşul ekler, ancak şifre alanı boş olarak bırakılır. Bu, aslında sorgunun şifre kısmını kontrol etme amacını taşır ve burada boş bırakılarak şifreye odaklanılmadığını gösterir.



SORU4:SSL VE TLS ÇALIŞMA MANTIĞINI AÇIKLAYINIZ.

SSL, istemci (client) ile sunucu (server) arasında veri alışverişinin şifreli / güvenli bir şekilde gerçekleştirilebilmesini sağlayan, öntanımlı olarak (default) 443. portu kullanan bir güvenlik katmanıdır.

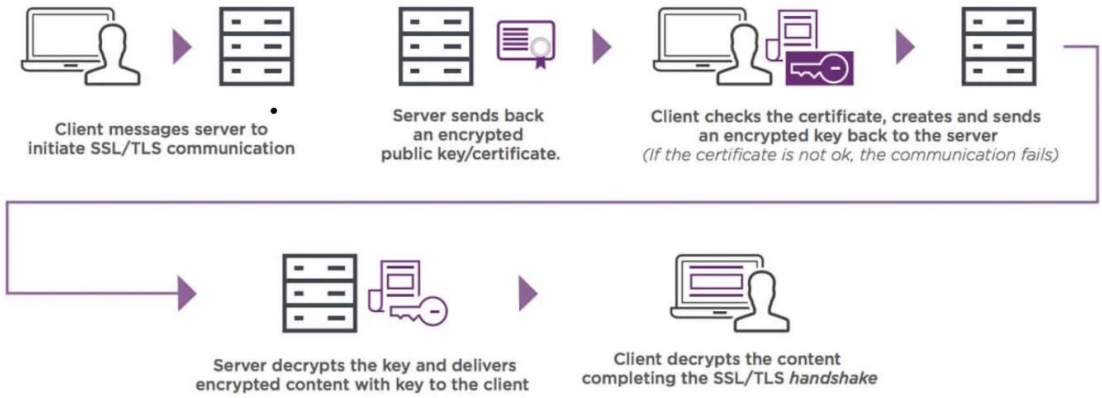
SSL Public Key / Private Key adı verilen anahtarların kullanımına dayalı bir kodlama yöntemine dayalıdır.SSL kodlama için iki adet anahtar bulunmaktadır. Bu anahtarlar, dijital ortamda kodlanmış yazılımlardır. Bir anahtarın kitlemiş olduğu veriyi, sadece diğer anahtar açabilir. Anahtarlarınızı yarattıktan sonra (SSL default olarak bu işlemi yapmaktadır, sizin herhangi bir işlem yapmanıza gerek yoktur), anahtarlardan biri (private key) sizde kalır. Diğer anahtar (public key) ise, bağlantı kurmak istediğiniz kişilere gönderilir.Dışarıdan sizinle iletişime geçmek isteyen kişi, public key'i kullanarak mesajı güvenli bir şekilde size gönderir. Veri, size ulaşmadan, transfer sırasında veriye ulaşılsa bile, şifrenin çözülmesi için sizde bulunan private key gerekecektir. SSL türüne göre 40 bit veya 128 bit şifreleme kullanılmaktadır. Bu karmaşıklıktaki şifrelemenin çözülmesi ileri tekniklerle dahi çok zaman alacaktır

TSL (Transport Layer Security) (IETF) standartlar yolu protokolüdür ve önceki SSL spesifikasyonları esas alınarak SSL'i de kullanıma sunan Netscape tarafından geliştirilmiştir. Bu açıdan değerlendirildiğinde, SSL için TLS'nin öncülü diyebiliriz. Bu nedenle kimi zaman SSL/TLS olarak da adlandırılır. eklenen özellikler ve modifiye edilen, daha gelişmiş ve güvenli hale getirilmiştir..

TLS iki katmandan oluşur:

- TLS Record Protocol (TLS Kayıt Protokolü)
- TLS Handshake Protocol (TLS El Sıkışma Protokolü)

Handshake Protokolü, sunucu ve kullanıcıların kimlik doğrulama işlemlerini gerçekleştirir. Handshake Protocol veri iletişimi öncesine şifreleme algoritmaları ile şifreleme anahtarlarına izin verilirken, Record Protocol ile bağlantının güvenliği sağlanır.



SORU5: SIZMA TESTİ UYGULAMASINDA KULLANILABİLECEK BİR TEHDİT MODELİNİN TEORİK YAPILARINI TANIMLAYIN.

Tehdit modeli, sızma testi sürecini organize etmek ve sistemi değerlendirmek için kullanılan bir çerçevedir. Bu teorik yapılar, sızma testinin adımlarını ve hedeflerini tanımlar.

1. ***Hedef Belirleme:***

Örn:Bir şirketin ya da bir web sitesinin hangi kuruluşun ya da sistemin olduğu belirlenmesi.

2. Bilgi Toplama:

Örn:Aktif ve Pasif tarama yapılarak hedef hakkında bilgi toplamak.Pasif taramalar için archive.org subfinder gibi araçları kullanbilir hedef hakkında pasif bilgi toplayabiliriz.

Aktif taram için nmap maltego gibi araçları kullanbilirz ve hedef hakkında daha detaylı sonuçlar elde ederiz

3. Zafiyet Taraması:

Örn:Hedef sistemin ağlarını tarayabilir ,port taraması yapabilir ve burdan zafiylerini görebiliriz.(Nmap ,Nessus gibi araçlar kullanabiliriz)

4. Zafiyet Analizi:

Örn:Zayifiyetlerin detaylı araştırılması CVE skorlarının incelenmesi ve nasıl kullanılabileciğine karar verilme aşamasıdır.(msfconsole,nessus,exploitdb)

5. Sızma Girişimleri:

- Örn:Bulduğumuz zafiyetler ile hedef bilgisayara ya da sisteme bağlanılmaya çalışılması ve yetki yükseltmeye elde etmeye çalışmak

6. Veri Çalma veya Bozma:

- Örn: hedef sistemlerden veri çalmaya veya verileri bozmaya yönelik girişimleri,Hassas verilere ulaşma çabaları.

7. Saldırıların Belgelemesi:

- Sızma testi sırasında yapılan işlemlerin ve elde edilen sonuçların belgelenmesi.

- Sızma testi raporu hazırlanması.

8. Bildirim ve İzleme:

- Hedef organizasyona sızma testinin gerçekleştirildiğine dair bildirim yapılması.

- İzleme ve reaksiyon planının uygulanması.

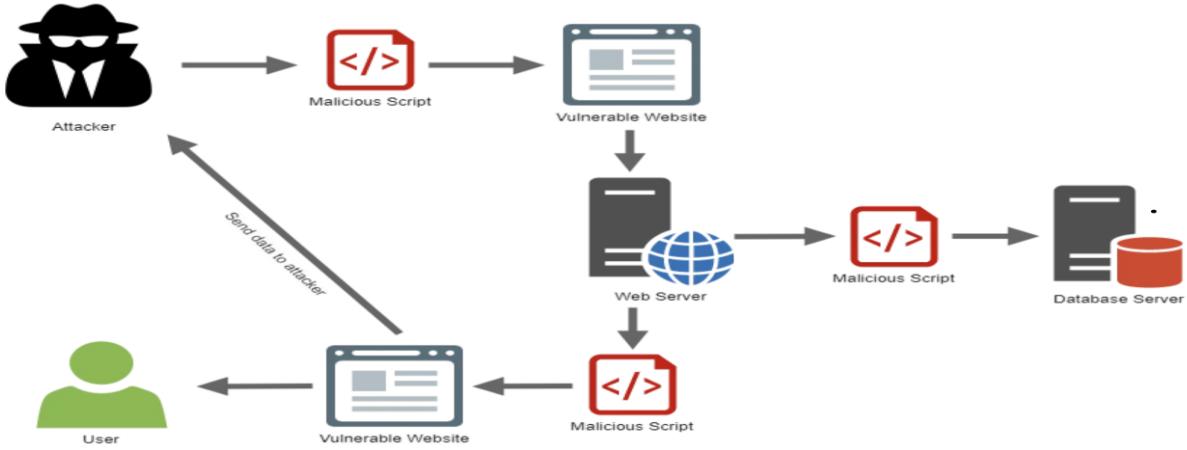
SORU6: ÜÇ TÜR SİTELER ARASI BETİK ÇALIŞTIRMA (XSS) NEDİR?

Cross-site scripting attack (XSS) siteler arası komut dosyası çalıştırma saldırısı, bir bilgisayar korsanının, iyi huylu ve güvenilir olarak görülen bir web sayfasının içeriğine, genellikle istemci tarafı komut dosyası biçiminde kötü amaçlı kod enjekte etmesiyle oluşur. Kötü amaçlı komut dosyası genellikle, JavaScript ve HTML olan istemci tarafı programlama dillerinde yazılır.

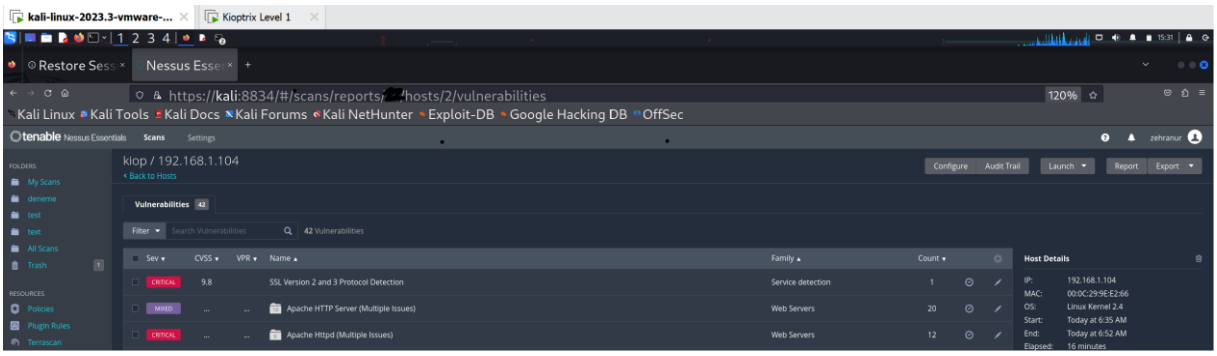
XSS için üç ana saldırı stratejisi vardır. Bunlar DOM XSS, reflected XSS ve stored XSS'dir.

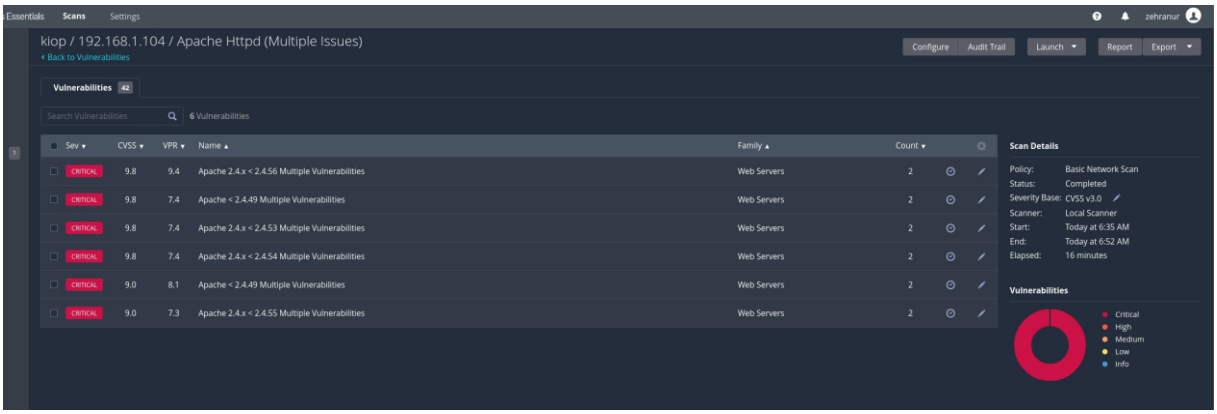
- **Reflected XSS=**Kullanıcıdan aldığı input(girdi) ile yazılan zararlı kodlar sistemin veritabanına kayıt edilmez bu yüzden kalıcı olmayan kodlar çalıştırılabildiği için reflected olarak kullanılır.
- **Stored XSS=**Reflected çeşidinin tam tersi olarak kullanıcıdan alınan girdiler sistemin veritabanına kayıt edilir bunun sonucunda kalıcı bir xss zafiyeti oluşur.Bu zafiyet sizin profilinizi ya da kodu eklediğiniz sayfayı ziyaret eden kullanıcıların maruz kalacağı bir güvenlik zafiyetidir.
- **Dom(Document Object Model) XSS=**Bu türde bütün işlem istemci tarafında gerçekleşir. Bunun diğer türlerden en büyük farkı, Dom XSS

istemci tarafında gerçekleştiği için ana sunucu bunun farkında olmayacaktır.



SORU7:KİOPTRİX ÜZERİNDEKİ APACHE HTTPd (MULTİPLE ISSUES) AÇIĞI HAKKINDA DETAYLI BİR ARAŞTIRMA YAPINIZ. (NASIL SÖMÜRÜLEBİLİR BİR FİKİR SUNUNUZ)





Bir kaç tanesinin içeriği hakkında bilgi vermek istiyorum .En kritik olan CVSS 9.8 açığımız:

HTTP isteğinin mod_rewrite ve mod_proxy ile bölünmesi: Apache HTTP Sunucusu 2.4.0'dan 2.4.55'e kadar olan sürümlerdeki bazı mod_proxy yapılandırmaları, HTTP İstek Kaçakçılığı saldırısına izin verir. Mod_proxy, belirli olmayan bir modelin kullanıcı tarafından sağlanan istek hedefi (URL) verilerinin bir kısmıyla eşleştiği ve daha sonra kullanılarak proxy edilen istek hedefine yeniden eklendiği bir tür RewriteRule veya ProxyPassMatch ile birlikte mod_proxy etkinleştirildiğinde yapılandırmalar etkilenir. değişken ikame. Örneğin, şöyle bir şey: RewriteRule üzerinde RewriteEngine ^/here/(.) http://example.com:8080/elsewhere?$1 http://example.com:8080/elsewhere ; [P] ProxyPassReverse /here/ http://example.com:8080/ http://example.com:8080/ İstek bölme/kaçakçılık, proxy sunucusundaki erişim kontrollerinin atlanmasına ve istenmeyen URL'lerin mevcut kaynak sunuculara proxylenmesine neden olabilir ve önbellek zehirlenmesi. (CVE-2023-25690)*

Bir diğer kritik zafiyetimiz CVSS 7.4 :

- ap_escape_quotes() kötü amaçlı girdi verildiğinde arabelleğin sonunun ötesine yazabilir. Dahil edilen modüllerin hiçbiri bu işlevlere güvenilmeyen verileri iletmez, ancak üçüncü taraf/harici modüller bunu yapabilir. (CVE-2021-39275)

***SEARCHSPLOIT APACHE HTTPD –ID** -W ile ilk önce kullanabilceğimiz herhangi bir modül var mı arama yapabiliriz.*

```
┌──(root㉿kali)-[/home/kali/Desktop]
└─# searchsploit Apache httpd --id -w
 Exploit Title                                                |  URL
Apache - Arbitrary Long HTTP Headers (Denial of Service)      | https://www.exploit-db.com/exploits/360
Apache - Arbitrary Long HTTP Headers Denial of Service        | https://www.exploit-db.com/exploits/371
Apache 0.8.x/1.0.x / NCSA HTTPd 1.x - 'test-cgi' Directo      | https://www.exploit-db.com/exploits/20435
Apache 1.1 / NCSA HTTPd 1.5.2 / Netscape Server 1.12/1.1      | https://www.exploit-db.com/exploits/19536
Apache 1.3.x < 2.0.48 mod_userdir - Remote Users Disclos      | https://www.exploit-db.com/exploits/132
Apache 2.0.44 (Linux) - Remote Denial of Service              | https://www.exploit-db.com/exploits/11
Apache 2.0.45 - 'APR' Crash                                   | https://www.exploit-db.com/exploits/38
Apache 2.0.49 - Arbitrary Long HTTP Headers Denial of Se      | https://www.exploit-db.com/exploits/1056
Apache 2.0.52 - GET Denial of Service                         | https://www.exploit-db.com/exploits/855
Apache 2.4.23 mod_http2 - Denial of Service                   | https://www.exploit-db.com/exploits/40909
Apache 2.x - Memory Leak                                      | https://www.exploit-db.com/exploits/9
Apache HTTP Server 2.4.49 - Path Traversal & Remote Code      | https://www.exploit-db.com/exploits/50383
Apache Httpd mod_proxy - Error Page Cross-Site Scripting      | https://www.exploit-db.com/exploits/47688
Apache Httpd mod_rewrite - Open Redirects                     | https://www.exploit-db.com/exploits/47689
Apache Tomcat mod_jk 1.2.20 - Remote Buffer Overflow (Me      | https://www.exploit-db.com/exploits/16798
NCSA 1.3/1.4.x/1.5 / Apache HTTPd 0.8.11/0.8.14 - Script      | https://www.exploit-db.com/exploits/20595
Shellcodes: No Results
Papers: No Results
```

```
┌──(root㉿kali)-[/home/kali/Desktop]
└─# searchsploit -m 20595
  Exploit: NCSA 1.3/1.4.x/1.5 / Apache HTTPd 0.8.11/0.8.14 - ScriptAlias Source Retrieval
      URL: https://www.exploit-db.com/exploits/20595
     Path: /usr/share/exploitdb/exploits/multiple/remote/20595.txt
    Codes: CVE-1999-0236, OSVDB-1745
Verified: True
File Type: ASCII text, with very long lines (728)
Copied to: /home/kali/Desktop/20595.txt

┌──(root㉿kali)-[/home/kali/Desktop]
└─# searchsploit -m 20435
  Exploit: Apache 0.8.x/1.0.x / NCSA HTTPd 1.x - 'test-cgi' Directory Listing
      URL: https://www.exploit-db.com/exploits/20435
     Path: /usr/share/exploitdb/exploits/cgi/remote/20435.txt
    Codes: CVE-1999-0070, OSVDB-55371
Verified: True
File Type: ASCII text, with very long lines (596)
Copied to: /home/kali/Desktop/20435.txt

┌──(root㉿kali)-[/home/kali/Desktop]
└─# searchsploit -m  19536
  Exploit: Apache 1.1 / NCSA HTTPd 1.5.2 / Netscape Server 1.12/1.1/2.0 - a nph-test-cgi
      URL: https://www.exploit-db.com/exploits/19536
     Path: /usr/share/exploitdb/exploits/multiple/dos/19536.txt
    Codes: CVE-1999-0045, OSVDB-128
Verified: True
File Type: ASCII text, with very long lines (642)
Copied to: /home/kali/Desktop/19536.txt
```

Diğer zafiylerimizde CVE uyuşan modüllerimizi kullanarak bir sömürge oluşturabiliriz.

Diğer bir yöntemimiz şu şekilde olabilir:

Searchsploit mod_ssh 2.4.8 ile versiyon ile ilgili modül var mı araştırma yapabiliriz.

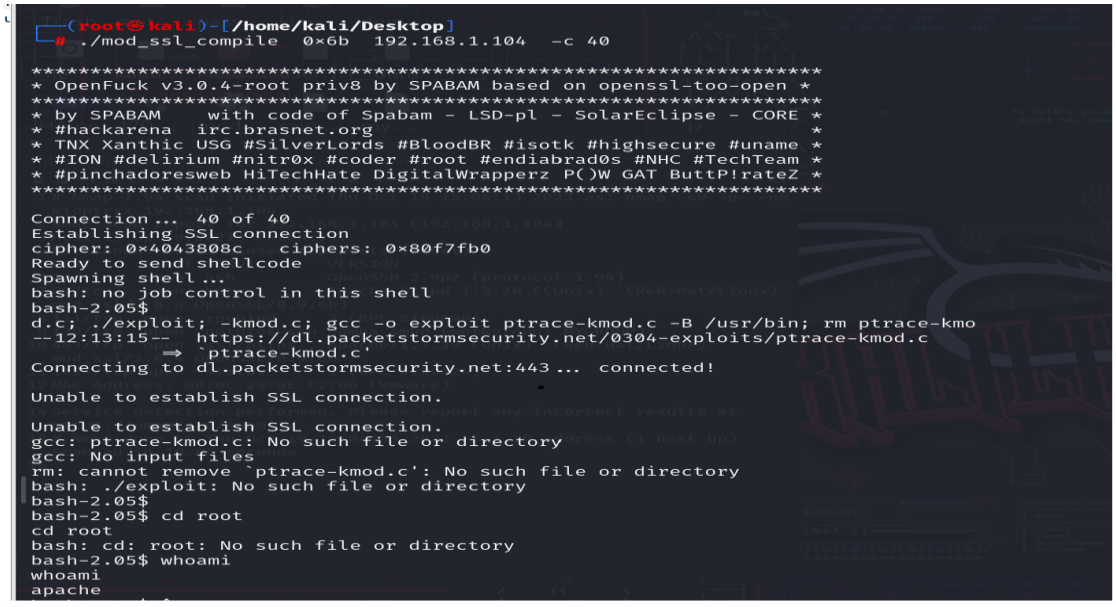
Burada searchsploit -m 47080 kullabilir 80 portu ile erişim sağlamayı deneyebiliriz.Yüksek yetkiye sahip olamayız ama bir bağlantı sağlanabilir.

Searchsploit -m exploit/unix/remote/47080.c

Gcc 470080.c mod_ssl_compile -lcrypto

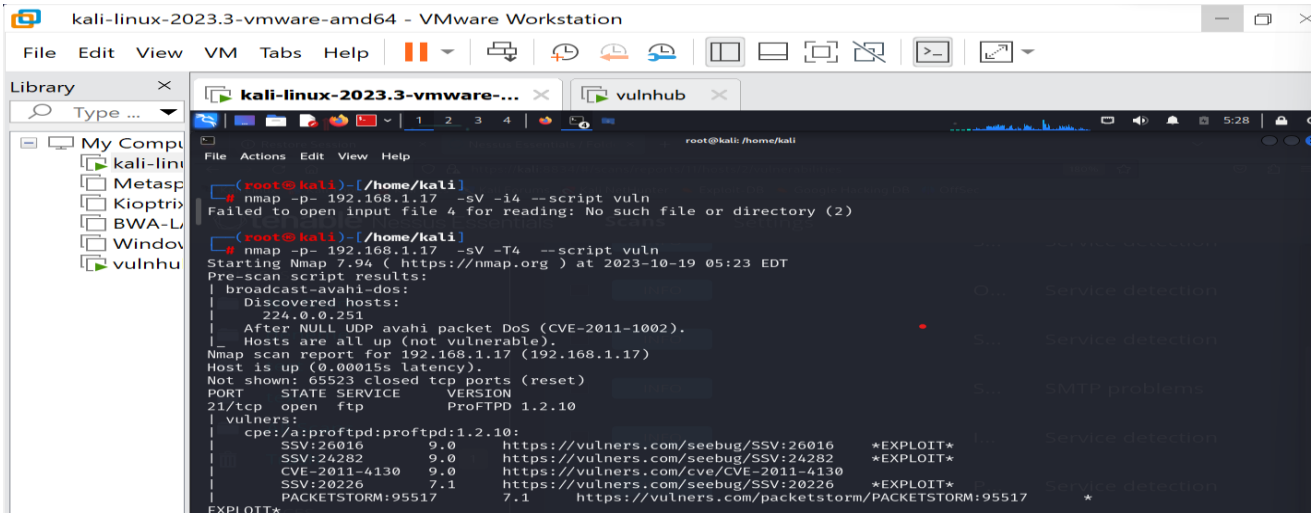
./ mod_ssl_compile

.mod_ssl_compile 0x6b(target) -ip -c 40



---

*SORU8: https://www.vulnhub.com/entry/digitalworldlocal-joy,298/ makinasını indirip vmware'a kurunuz.



ProFTPd versiyonu nedir?

PROFTPD 1.2.10

SORU9: MS17-010 NEDİR AÇIKLAYINIZ.

MS17-010, Microsoft tarafından Mart 2017'de yayınlanan bir güvenlik açığıdır. Bu güvenlik açığı, Windows işletim sistemlerinde SMB (Server Message Block) protokolünün işlendiği bölümün bir eksikliğinden kaynaklanır ve etkilenen sistemlerde kötü amaçlı kişi tarafından kullanılarak, sistemlerin yetkisiz erişimine ve kontrolüne olanak sağlar.SMB, ağdaki cihazların ve bilgisayarların dosya ve yazıcı paylaşımı yapmalarını sağlayan bir ağ protokolüdür. MS17-010 zafiyeti, SMB protokolünde bulunan bir dizi güvenlik açığından kaynaklanmaktadır ve bu açıkları kötü niyetli kişiler, WannaCry fidye yazılımı gibi zararlı saldırılar yapmak için kullanabilirler.

MS17-010 açığını istismar etmek için, bir kötü amaçlı kişi öncelikle etkilenen bir sisteme bir "hedef paket" gönderir. Bu paket, sistemde bir zayıflık tespit eder ve bir "çalışma kodu" yükler. Çalışma kodu, sistemde bir zafiyet yaratarak, bir kötü amaçlı kişinin yetkisiz erişimine ve kontrolüne olanak sağlar

SORU10: VPN VE PROX FARKINI AÇIKLAYINIZ.

Proxy'ler ve VPN'lerin her ikisi de gizlilik sağlasa da bunu farklı şekilde yaparlar. Proxy ile VPN yeteneklerini karşılaştırırken aradaki fark, proxy'lerin kesinlikle internet ile kullanıcılar arasında bir ağ geçidi görevi görmesidir. Öte yandan VPN trafiği şifreli bir tünelden ve kullanıcının cihazından geçerek VPN'leri ağ güvenliğinin sağlanmasında etkili bir çözüm haline getiriyor.

SORU11: METERPRETER VE SHELL FARKINI AÇIKLAYINIZ.

Meterpreter, Metasploit Framework ile kullanılan bir saldırganın kurbanın bilgisayarına görünmez bir kapı/bir iletişim kanalı kurarak kontrol etmesini sağlayan popüler bir penetrasyon testi ve istismar aracıdır. Başlangıçta bir istismar veya ele geçirme gerçekleştikten sonra hedef bir sisteme enjekte edilmek üzere tasarlanmıştır. Meterpreter, bir ele geçirilmiş sistem üzerinde kontrolü sürdürmek için güçlü bir araç olan bir dizi gelişmiş özellik ve yetenek sunar. Meterpreter'ın bazı temel özellikleri şunlardır:

Çeşitlilik: Meterpreter, dosya sistemine erişim, ekran görüntüsü yakalama, tuş kaydı ve ayrıcalık yükseltme gibi bir dizi işlev sunar, bu da onu son işgal görevleri için çok yönlü bir araç yapar.

Şifrelenmiş İletişim: Meterpreter, saldırgan ile ele geçirilen sistem arasında değişen verileri korumak için şifreli iletişim kanalları kullanır.

Oturum Yönetimi: Birden çok oturumu yönetmeyi mümkün kılar, bu da bir saldırganın aynı anda birkaç ele geçirilmiş sistemle etkileşimde bulunmasına imkan tanır.

Betikleme ve Uzantılar: Meterpreter, betikleme ve uzantıları destekler, bu da kullanıcıların özel işlev eklemek için özel modüller geliştirmesine olanak tanır.

Kabuk (Shell):

Kabuk, penetrasyon testi veya hackleme bağlamında, genellikle bir saldırganın ele geçirilen bir sistemi ile etkileşime girmesine izin veren bir komut kabuğu veya komut satırı arabirimini ifade eder. Kabuklar, ters kabuklar ve bağlantı kabukları gibi çeşitli biçimlerde gelir. Kabuklar genellikle bir saldırının ana hedefidir, çünkü saldırganlara hedef sistemin komut satırına erişim sağlar; bu, komutları çalıştırma, betikleri yürütme ve çeşitli görevleri gerçekleştirme amacıyla kullanılabilir. Bir kabuğun temel özellikleri şunlardır:

Komut Yürütme: Kabuklar, saldırganın hedef sistemin komut satırına doğrudan erişim sağlar, bu da saldırganın sistemde fiziksel olarak bulunuyormuş gibi komutları çalıştırmasına olanak tanır.

Sınırlı İşlevsellik: Standart bir kabuk, genellikle Meterpreter'ın kapsamlı işlev setine sahip değildir ve yetkilere ve ele geçirilen kullanıcının veya sistemin erişim düzeyine bağlı olarak işlevselliği değişir.

Dahili Şifreleme Yok: Kabuklar genellikle iletişim için dahili şifreleme sunmaz, bu da saldırgan ile ele geçirilen sistem arasında değişen verilerin çoğu zaman şifrelenmediği anlamına gelir.