Aktif Tarama – Bilgi Toplama(ACTİVE SCAN)

Aktif tarama olarak Türkçeye çevirebileceğimiz tarama türü, hedef hakkında daha fazla bilgi alabilmek için etkileşime girilerek uygulanan bir tarama yöntemidir. Yapılan taramada hedef cihazla etkileşime girildiği için yapılan taramaların kayıtları hedef cihazda tespit edilebilmektedir. Bu tarama türünü iki teknik ile gerçekleştirilebilir. Bunlar IP blok taraması ve Güvenlik açığı taramalarıdır.

IP blok taraması (Scanning IP Block) hedef sistemde hangi bilgisayarların çalıştığını/aktif olduğu ve IP adresini belirlemede kullanılan taramadır. Genellikle bir saldırganın saldırıdaki ilk adımıdır. Host keşfinde yaygın olarak kullanılan iki protokol vardır. Bunlar ICMP ve ARP protokolleridir. ARP protokolünün asıl amacı IP adresine karşılık gelen MAC adresini keşfetmekti. Bu protokolü bulunduğu ağın, alt ağ bilgisine göre (aktif olabilecek cihaz sayısının belirlenmesi için gerekli) alınabilecek IP adreslerini içeren ARP istek paketlerini yerel alan ağına yayınlaması ile gerçekleştiriyor. ARP cevap paketi gönderen cihazların aktif olduğu anlaşılıyor. ICMP protokolü ise iki cihaz arasındaki bağlantı durumunu kontrol etmek için kullanılıyordu. Bu protokolü ICMP yankı (type 8) paketi gibi birkaç ICMP paket türü ile gerçekleştirilebiliyor. Gelen yanıta göre cihazın aktif olup olmadığı belirleniyor. Cevap alınmadığı durumlarda ise paketin hedef cihazın aktif olmadığı veya güvenlik duvarın tarafından engellenmiş olabileceği göz önünde bulundurulur. Bu tarama türünü gerçekleştirmek için yaygın olarak kullanılan araçlara örnek olarak nmap, nping, IP range scanner ve advanced IP scanner gibi araçları verebiliriz.

Güvenlik açığı taraması, hedef kurumun ağındaki cihazların olası saldırılara karşılaşabilecekleri güvenlik açıklarını raporlamak için kullanılan yazılım araçları kullanılarak gerçekleştirilir. Taramada hedef ile ilgili ayrıntıları karşılaştırmak için veri tabanı kullanırlar. Bu veri tabanı bilinen kusurlara, kodlama hatalarına, varsayılan yapılandırmalara ve saldırgan tarafından ele geçirilebilecek hassas verilere yönelik olası yollara başvurur. Hedef cihazdaki olası güvenlik açıklarını kontrol ettikten sonra bir tarama raporu oluşturur. Bu rapor sonucu analiz edilerek sistemdeki güvenlik açıklarının kapatılması sağlanır. Saldırgan tarafında ise bu açıkları belirleyerek saldırı haritasını oluşturmasında yardımcı olur. Yanlış anlamadıysam port taramaları da bu tarama türüne giriyor. Cihazdaki portlara TCP oturum açma istekleri göndererek test ediliyor. TCP isteklerine karşılık TCP + ACK paketi gönderen porta RST bayrağı göndererek bağlantı sonlandırılıyor. Bu şekilde açık portlar belirleniyor. Daha fazlasını aşağıdaki örnek taramalarda inceleyebilirsiniz. Bu taramayı gerçekleştirebilmek için yaygın kullanılan araçlara örnek olarak nikto, Netsparker, Nessus, opevas ve namp gibi araçları verebiliriz.

Bu tarama türlerine bir örnek olarak https://www.vulnhub.com/ sitesinden rastgele seçtiğim “pownos :2.0” adlı sanal makinayı virtualbox üzerinde kurarak gerçekleştirmeye çalışacağım.

NOT: Buradaki uygulamalar gerçekleştirmek isterseniz öncelikle sanal makinanın network ayarlarını gerçekleştirmeniz gerekiyor. İlk olarak indirmiş olduğum sanal makinamın (pwnOs:2.0) ip adresi sorun çıkarmaması için iki sanal makinada da network ayarlarını “internal network” olarak seçiyoruz. İndirdiğimiz sanal makinanın IP adresi statik olarak atanmış olduğu için sadece atanan IP adresinin alt ağ bilgisine göre bir IP adresi atamamız gerekiyor. Bunları göz önünde bulundurarak DHCP sunucumuz olmadığı için kali cihazımızda IP adresini kendimiz veriyoruz (IP=10.10.10.98 SUBNET=/24 GATEWAY=10.10.10.15).

İlk olarak kurduğumuz sanal makinayı başlattığımızda dahil olduğu ağdaki IP adresini öğrenebilmek için IP block taraması gerçekleştiriyoruz. Taramayı başlatmadan önce tarama yapacağım ama taramayı başlatmadan önce ağdaki trafik akışını yakalamak için wireshark aracını çalıştırıyorum. Tarama için öncelikle dahil olduğu ağı bilmem gerekiyor. Bunun için sanal makinayı indirdiği sitede her ne kadar IP adresi verilmiş de olsa ben dahil olduğu ağdaki bütün aktif cihazları görebilmek için nmap aracında "sudo nmap -sP 10.10.10.1/24" komutunu kullanıyorum (255.255.255.0 alt ağında olduğu için /24 koyduk). Tarama sonucu çıktı aşağıdaki gibidir.

```
kali@kali:~$ sudo nmap -sP 10.10.10.1/24
Starting Nmap 7.80 ( https://nmap.org ) at 2021-02-22 11:06 EST
Nmap scan report for 10.10.10.100
Host is up (0.0015s latency).
MAC Address: 08:00:27:AE:75:BA (Oracle VirtualBox virtual NIC)
Nmap scan report for 10.10.10.98
Host is up.
Nmap done: 256 IP addresses (2 hosts up) scanned in 33.54 seconds
kali@kali:~$
```

Taramanın arka planda oluşturduğu trafiğe baktığımızda aşağıda görüldüğü gibi 1-254 IP adres aralığında ARP istek paketlerini ağa bırakarak (ARP istek paketi broadcast yayın yapıyor) dönüş yapan cihazlara göre aktif olan bilgisayarların durumlarını belirliyordu. Burada da 10.10.10.100 IP adresli sanal cihazım kendi IP adresini içeren ARP istek paketine karşılık olarak ARP cevap paketi göndererek MAC adresini bildiriyor.

No.	Time	Source	Destination	Protocol	Length	Info
367	7.306155643	PcsCompu_5c:65:26	Broadcast	ARP	42	Who has 10.10.10.99? Tell 10.10.10.98
368	7.306166262	PcsCompu_5c:65:26	Broadcast	ARP	42	Who has 10.10.10.100? Tell 10.10.10.98
369	7.306220908	PcsCompu_5c:65:26	Broadcast	ARP	42	Who has 10.10.10.101? Tell 10.10.10.98
370	7.306233954	PcsCompu_5c:65:26	Broadcast	ARP	42	Who has 10.10.10.102? Tell 10.10.10.98
371	7.306454606	PcsCompu_ae:75:ba	PcsCompu_5c:65:26	ARP	60	10.10.10.100 is at 08:00:27:ae:75:ba
372	7.308713117	PcsCompu_5c:65:26	Broadcast	ARP	42	Who has 10.10.10.109? Tell 10.10.10.98
373	7.308736580	PcsCompu_5c:65:26	Broadcast	ARP	42	Who has 10.10.10.116? Tell 10.10.10.98

```
▸ Frame 371: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface eth0, id 0
▾ Ethernet II, Src: PcsCompu_ae:75:ba (08:00:27:ae:75:ba), Dst: PcsCompu_5c:65:26 (08:00:27:5c:65:26)
  ▸ Destination: PcsCompu_5c:65:26 (08:00:27:5c:65:26)
  ▸ Source: PcsCompu_ae:75:ba (08:00:27:ae:75:ba)
    Type: ARP (0x0806)
    Padding: 000000000000000000000000000000000000
▾ Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    Sender MAC address: PcsCompu_ae:75:ba (08:00:27:ae:75:ba)
    Sender IP address: 10.10.10.100
    Target MAC address: PcsCompu_5c:65:26 (08:00:27:5c:65:26)
    Target IP address: 10.10.10.98
```

Aktif cihazlarımızı belirledikten sonra artık açık portlarını ve bu portların sürümlerini öğrenmemiz gerekiyor. Bu bilgileri elde etmek için yine nmap aracında "sudo nmap -sC -sV 10.10.10.100" komutunu kullanıyorum. Çıktısı aşağıdaki gibidir. Görüldüğü gibi 22/SSH portunun ve TCP/80 portunun açık olduğunu ve sürüm bilgileri belirlendi.

```
kali@kali:~$ sudo nmap -sC -sV 10.10.10.100
[sudo] password for kali:
Starting Nmap 7.80 ( https://nmap.org ) at 2021-02-22 12:46 EST
Nmap scan report for 10.10.10.100
Host is up (0.00053s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 5.8p1 Debian 1ubuntu3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 85:d3:2b:01:09:42:7b:20:4e:30:03:6d:d1:8f:95:ff (DSA)
|   2048 30:7a:31:9a:1b:b8:17:e7:15:df:89:92:0e:cd:58:28 (RSA)
|_  256 10:12:64:4b:7d:ff:6a:87:37:26:38:b1:44:9f:cf:5e (ECDSA)
80/tcp open  http    Apache httpd 2.2.17 ((Ubuntu))
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
|_http-server-header: Apache/2.2.17 (Ubuntu)
|_http-title: Welcome to this Site!
MAC Address: 08:00:27:AE:75:BA (Oracle VirtualBox virtual NIC)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 20.26 seconds
kali@kali:~$
```

Bu çıktının arkasında oluşturduğu trafik aşağıdaki gibiydi. Gördüğüm kadarıyla 1-65535 aralığındaki portlara belirli aralıklarla öncelikle SYN paketleri yollayarak oturum açmaya çalışıyor. Bu sayede cevap veren portlara göre hangi portların açık olduğunu da belirliyor. Cevap gönderen (SYN+ACK) portlar için tekrar RST paketi göndererek bağlantıyı koparıyor. Örnek olarak aktif olan 80 portu için gerçekleşen işlemler aşağıdaki görselde verilmiştir.

	Source	src port	Destination	dst port	Protocol	Length	Info
57301	10.10.10.100	111	10.10.10.98	54241	TCP	60	111 → 54241 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
51277	10.10.10.98	54241	10.10.10.100	3389	TCP	58	54241 → 3389 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
13801	10.10.10.98	54241	10.10.10.100	8888	TCP	58	54241 → 8888 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
60368	10.10.10.100	5900	10.10.10.98	54241	TCP	60	5900 → 54241 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
60641	10.10.10.100	554	10.10.10.98	54241	TCP	60	554 → 54241 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
60823	10.10.10.100	1720	10.10.10.98	54241	TCP	60	1720 → 54241 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
66349	10.10.10.98	54241	10.10.10.100	25	TCP	58	54241 → 25 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
18103	10.10.10.98	54241	10.10.10.100	80	TCP	58	54241 → 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
58847	10.10.10.98	54241	10.10.10.100	110	TCP	58	54241 → 110 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
10120	10.10.10.98	54241	10.10.10.100	3306	TCP	58	54241 → 3306 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
89760	10.10.10.100	587	10.10.10.98	54241	TCP	60	587 → 54241 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
90048	10.10.10.100	3389	10.10.10.98	54241	TCP	60	3389 → 54241 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
90274	10.10.10.100	8888	10.10.10.98	54241	TCP	60	8888 → 54241 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
70892	10.10.10.100	25	10.10.10.98	54241	TCP	60	25 → 54241 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
71145	10.10.10.100	80	10.10.10.98	54241	TCP	60	80 → 54241 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1460
71318	10.10.10.100	110	10.10.10.98	54241	TCP	60	110 → 54241 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
71519	10.10.10.100	3306	10.10.10.98	54241	TCP	60	3306 → 54241 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
10938	10.10.10.98	54241	10.10.10.100	80	TCP	54	54241 → 80 [RST] Seq=1 Win=0 Len=0
24065	10.10.10.98	54241	10.10.10.100	993	TCP	58	54241 → 993 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
52179	10.10.10.98	54241	10.10.10.100	995	TCP	58	54241 → 995 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
08013	10.10.10.98	54241	10.10.10.100	256	TCP	58	54241 → 256 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
49382	10.10.10.98	54241	10.10.10.100	8652	TCP	58	54241 → 8652 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
98382	10.10.10.98	54241	10.10.10.100	37	TCP	58	54241 → 37 [SYN] Seq=0 Win=1024 Len=0 MSS=1460

Bütün portları denedikten sonra açık olduğunu belirlediği portlara tekrar bağlantı isteği göndererek oturum açıyor. Anladığım kadarıyla "nmap-service-probes" adında kullanılan veri tabanında çeşitli sorgulama için araştırmaların ve bu sorguların yanıtlarını ayrıştırmak ve tanımlamak için ifade eşleştirmelerini içeriyormuş. Daha detaylı bilgi için https://nmap.org/book/man-version-detection.html ve https://linuxhint.com/nmap_version_scan/#:~:text=To%20enable%20service%20detection%20the,such%20as%20ssh%20or%20http. sitesindeki yazıları okuyabilirsiniz.

Açık olan 80 portunun versiyon bilgisi için ise sanırım yukarıda açıkladığımız gibi farklı http istekleri göndererek sürümünü belirlemeye çalışıyor.

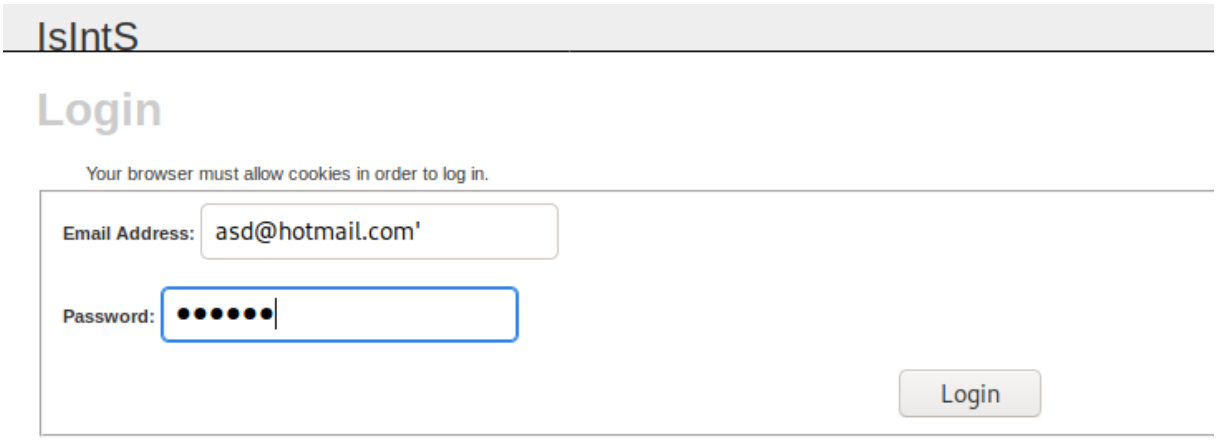
No.	Time	Source	src port	Destination	dst port	Protocol	Length	Info
2027	19.320419057	10.10.10.98	47438	10.10.10.100	80	HTTP	84	GET / HTTP/1.0
2029	19.322240471	10.10.10.100	80	10.10.10.98	47438	HTTP	1325	HTTP/1.1 200 OK (text/html)
2078	19.366004445	10.10.10.98	47442	10.10.10.100	80	HTTP	226	GET /robots.txt HTTP/1.1
2079	19.366026139	10.10.10.98	47444	10.10.10.100	80	HTTP	240	GET /nmaplowercheck1614016034 HTTP/1.1
2080	19.366040919	10.10.10.98	47446	10.10.10.100	80	HTTP	278	OPTIONS / HTTP/1.1
2081	19.366062899	10.10.10.98	47448	10.10.10.100	80	HTTP	282	GET / HTTP/1.1 , NTLMSSP_NEGOTIATE
2082	19.366087749	10.10.10.98	47450	10.10.10.100	80	HTTP	682	POST /sdk HTTP/1.1
2083	19.366118902	10.10.10.98	47452	10.10.10.100	80	HTTP	225	GET /.git/HEAD HTTP/1.1
2084	19.366119907	10.10.10.98	47454	10.10.10.100	80	HTTP	231	PROPFIND / HTTP/1.1
2085	19.366138196	10.10.10.98	47456	10.10.10.100	80	HTTP	220	OPTIONS / HTTP/1.1
2086	19.366154198	10.10.10.98	47458	10.10.10.100	80	HTTP	220	OPTIONS / HTTP/1.1
2087	19.366169578	10.10.10.98	47460	10.10.10.100	80	HTTP	374	POST / HTTP/1.1 (application/x-www-form-urlencoded)
2088	19.366187161	10.10.10.98	47462	10.10.10.100	80	HTTP	84	GET / HTTP/1.0
2089	19.366204239	10.10.10.98	47466	10.10.10.100	80	HTTP	231	PROPFIND / HTTP/1.1
2102	19.366735283	10.10.10.100	80	10.10.10.98	47442	HTTP	555	HTTP/1.1 404 Not Found (text/html)
2105	19.367009685	10.10.10.100	80	10.10.10.98	47444	HTTP	569	HTTP/1.1 404 Not Found (text/html)
2108	19.367745300	10.10.10.100	80	10.10.10.98	47446	HTTP	1325	HTTP/1.1 200 OK (text/html)
2111	19.368421706	10.10.10.100	80	10.10.10.98	47448	HTTP	1325	HTTP/1.1 200 OK (text/html)
2114	19.368562305	10.10.10.100	80	10.10.10.98	47450	HTTP	548	HTTP/1.1 404 Not Found (text/html)
2117	19.368854497	10.10.10.100	80	10.10.10.98	47452	HTTP	554	HTTP/1.1 404 Not Found (text/html)
2158	19.436141500	10.10.10.100	80	10.10.10.98	47458	HTTP	1325	HTTP/1.1 200 OK (text/html)
2161	19.437503972	10.10.10.100	80	10.10.10.98	47460	HTTP	1325	HTTP/1.1 200 OK (text/html)
2164	19.438912722	10.10.10.100	80	10.10.10.98	47462	HTTP	1325	HTTP/1.1 200 OK (text/html)
2166	19.438912898	10.10.10.100	80	10.10.10.98	47456	HTTP	1325	HTTP/1.1 200 OK (text/html)
2170	19.440600464	10.10.10.100	80	10.10.10.98	47466	HTTP	1325	HTTP/1.1 200 OK (text/html)
2173	19.440804704	10.10.10.100	80	10.10.10.98	47454	HTTP	1325	HTTP/1.1 200 OK (text/html)
2198	19.447412556	10.10.10.98	47468	10.10.10.100	80	HTTP	226	GET /evox/about HTTP/1.1
2199	19.447548698	10.10.10.98	47470	10.10.10.100	80	HTTP	277	OPTIONS / HTTP/1.1
2209	19.448371919	10.10.10.100	80	10.10.10.98	47468	HTTP	555	HTTP/1.1 404 Not Found (text/html)

IP blok ve versiyon taramaları sonrasında, hedef sistemde güvenlik açığı taraması gerçekleştirerek sistemdeki belirli güvenlik açıklarını belirleyelim. Bu tarama için “nikto” aracını kullanarak gerçekleştirdim. Makinamda yazdığım komut “nikto -h 10.10.10.100” şeklindedir. Çıktısı aşağıdaki gibidir.

```
kali@kali:~$ nikto -h 10.10.10.100
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          10.10.10.100
+ Target Hostname:    10.10.10.100
+ Target Port:        80
+ Start Time:         2021-02-23 14:12:00 (GMT-5)
---------------------------------------------------------------------------
+ Server: Apache/2.2.17 (Ubuntu)
+ Retrieved x-powered-by header: PHP/5.3.5-1ubuntu7
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ Cookie PHPSESSID created without the httponly flag
+ Apache/2.2.17 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.
+ Uncommon header 'tcn' found, with contents: list
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. See http://www.wisec.it/sectou.php?id=4698ebdc59
  following alternatives for 'index' were found: index.php
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-12184: /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY
.
+ OSVDB-12184: /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY
.
+ OSVDB-12184: /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY
.
+ OSVDB-12184: /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY
.
+ OSVDB-3268: /includes/: Directory indexing found.
+ OSVDB-3092: /includes/: This might be interesting ...
+ /info/: Output from the phpinfo() function was found.
```

Tarama sonrası oluşturduğu trafik de nmap aracının versiyon taramasına benzer şekilde 80 portuna belirli istekler göndererek çıkarımlar yapmaya çalışıyordu.

Görüldüğü gibi hedef sistemde olabilecek güvenlik açıklarını listeliyor. Özetle bir XSS zafiyeti olabileceğini gösteriyor. Bu durumu anlamak için tarayıcımda siteyi açarak denemeler yapıyorum. Bizi “Home” sayfası karşılıyor. “login” sayfasında birkaç deneme yaptıktan sonra e posta kısmına “ ’ ” işareti koyduğumda veri tabanına gönderilen sorguda hata oluşturduğunu görüyoruz. XSS hakkında fikir edinmek için https://portswigger.net/web-security/cross-site-scripting yazısına göz atabilirsin. İlk olarak XSS zafiyetine sahip olduğundan emin olmak adına giriş kısımlarına veri tabanına gönderilen sorguda hata oluşturabilecek işaretler kullanarak denemeler yaptım. “ ‘ ” işareti sonrası hatayı aldım.



(Giriş sayfası)

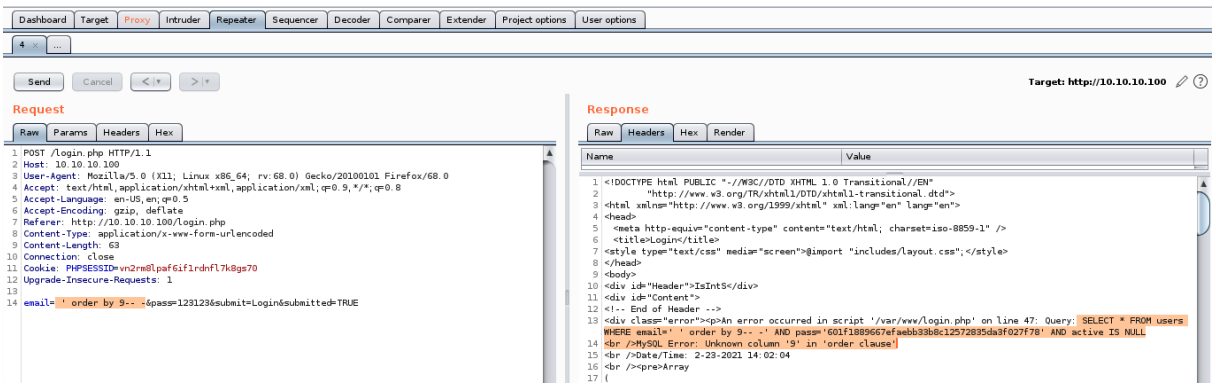
```
            [server_version] => 50154
            [sqlstate] => 42000
            [protocol_version] => 10
            [thread_id] => 83
            [warning_count] => 0
        )

    [e] => asd@hotmail.com'
    [p] => 123asd
    [q] => SELECT * FROM users WHERE email='asd@hotmail.com'' AND pass='ec4c8836db96b8aca8381c7c64bb095ba46d5e28' AND active IS NULL
    [r] =>
)
```

(Verilen hata)

Sayfada istekler POST metodunu kullanarak gönderiliyordu. Yukarıdaki çıktıyı gördükten sonra “login” sayfasındaki girişleri manipüle edebilmek için “Burpsuite” aracını açıyorum (Tarayıcının Proxy ayarlamaları düzenlendi). Proxy açtıktan sonra “login” sayfasında tekrar girişleri yaparak gönderilen giriş bilgilerini düzenlemeye başladım. Yukarıdaki görselde de görüldüğü gibi “user” tablosundaki bilgileri sorguluyor. Bu durumda ilk olarak ”user” tablosunda satır sayısını bulabilmek için aşağıdaki sorguları sırasıyla gönderiyoruz.

“email=” kısmından sonra “&” işaretine kadar olan kısmına sayıyı bulabilmek için “ ‘ order by 1-- -” komutuyla denemeler yapıyoruz. Bu sitede 9 olarak gönderilen içerikte aşağıdaki gibi “bilinmeyen sütun” hatası alıyoruz. Bu bize tabloda 8 kayıt olduğunu gösteriyor.



Yazının fazla uzamaması için makine çözümünü burada bırakıyorum. Çözümü için kendiniz uğraşabilir veya site üzerindeki çalışmalara göz atabilirsiniz. Taramanın saldırıdaki yerine bir örnek verdikten sonra şimdi de bu saldırıları nasıl tespit edebileceğimize bakalım.

NASIL TESPİT EDİLEBİLİR?

IP blok ve güvenlik açığı taramalarında, örneklerden de dikkatinizi çekebileceği gibi aynı IP adreslerinden çok kısa zaman aralıklarıyla hedef ağa veya hedef cihaza istekler gönderiliyor. Bu durum ağda anlık yüksek trafiğe neden olur. Her ne kadar belirli bir IP adresinden gelen isteklerin birim zamandaki sayısına bakarak tespit edilebilir görünse de saldırganlar bu saldırıları belirli zaman periyotlarında gerçekleştirerek bu tespit yöntemini atlatabiliyorlar. Aynı zamanda güvenlik açığı tarayıcıları bazı sistemlerde daha fazla bilgi edinebilmek için sisteme erişim sağlamaya çalışıyorlardı. Host tabanlı IDS sistemler bu erişimi tespit edebilmek için cihazın olay günlüklerini, alınan hataları ve ağ trafiği gibi birçok durumu kontrol ederler. Burada IDS sistemin tespit edebilmesi için belirli kuralların tanımlaması gerekir. Örnek olarak https://resources.infosecinstitute.com/topic/snort-network-recon-techniques/ buradan ve kendi gerçekleştirdiğim uygulama için buradan uygulamalarını inceleyebilirsiniz. Bu ve bu gibi birçok tespit yöntemi vardır. Bu konu hakkında daha fazla bilgi için https://www.ciscopress.com/articles/article.asp?p=469623&seqNum=5 bağlantısını inceleyebilirsiniz.

NASIL ÖNLEMLER ALINABİLİR?

“Nasıl tespit edilir” kısmında da bahsedildiği gibi host tabanlı IDS-IPS sistemler kullanarak bu tür taramalar için yapılandırmak (örnek olarak kullanılması tavsiye edilmeyen bir yöntem ana bilgisayarımıza gönderilen ICMP paketlerini IPS sistem ile engellemek), güvenlik duvarları yapılandırılarak bu taramalar için önlemler alınabilir.

Bu kısımda gerçekleştirdiğim uygulamaların oluşturduğu ağ trafiğini ve olay günlük dosyalarına buradan ulaşabilirsiniz.

Kaynaklar:

https://www.esecurityplanet.com/networks/vulnerability-scanning-what-it-is-and-how-to-do-it-right/

https://www.balbix.com/insights/what-to-know-about-vulnerability-scanning-and-tools/

https://www.extrahop.com/company/blog/2016/how-to-recognize-malicious-network-scanning-port-scanning/

https://www.varonis.com/blog/port-scanning-techniques/

https://www.ciscopress.com/articles/article.asp?p=469623&seqNum=5