HACKLENMİŞ LINUX SİSTEM ANALİZİ

Yazar: Uygar Köroğlu – Samet Sazak
Mentor: Huzeyfe ÖNAL
Baskı: 2017

İÇİNDEKİLER

Giriş

İnternet ortamında altyapı, ağ ve sunucu hizmetlerinde kullanılan işletim sistemleri %70 gibi büyük oranda UNIX/Linux tabanlı sistemlerden oluşmaktadır. [1] Linux sistemlerin yoğun kullanımı saldırganların da hedeflerini bu sistemlere yöneltmesine sebep olmuştur. Hacklenen sitelerin kaydını tutan Zone-H istatistiklerine göre son yıllarda en çok Linux sistemler hedef oluyor ve ele geçiriliyor.

Operative System	Year 2010
Linux	1.126.987
Windows 2003	197.822
FreeBSD	46.992
Win 2008	15.083
F5 Big-IP*	14.000
Unknown	7.840
Win 2000	6.097
Solaris $^{9}/_{10}$	2.373
MacOSX	1.038
Citrix Netscaler*	232
Win NT9x	221
Win XP	196
NetBSDOpenBSD	99
HP-UX	73
IRIX	47
SCO UNIX	22

Linux sistemler genellikle komut satırı kullanarak yönetildiği için sistem üzerindeki anormallikler Windows işletim sistemindeki gibi kolay anlaşılmayabilir.

Bu yazıda internet üzerinde hizmet veren Linux sistemlere yönelik gerçekleştirilen saldırılar sonrası sistemlerin nasıl inceleneceği, canlı analiz yöntemleri, saldırılar ve anormalliklerden erken haberdar olmak için gerekli adımlara değinilecektir. Burada dikkat edilmesi gereken bir husus da yapılan analizin adli delil olarak kullanımına uygun şekilde elde edilecek verilerin *delil* niteliğini korumasıdır. Bunun için de analiz işlemi öncesinde yedek/imaj alınması önem arz etmektedir.

Analiz Çalışmasına Başlamadan Kontrol Edilmesi Gereken Ön Hususlar

Zaman Bilgisi

İncelenecek sistemin saat bilgisinin kontrol edilmesi, saat bilgisi hatalı olan sistem inceleme sonuçlarının da sağlıksız sonuçlanmasına sebep olacaktır. "*date*" komutu kullanılarak sisteme ait zaman bilgileri elde edilebilir.

```
root@bgapentest:~# date
```

İşletim Sistemi ve Kernel(Çekirdek) Sürüm Bilgisi

İşletim sistemi Kernel(çekirdek) sürümü ve mimari bilgileri oldukça önemlidir. Aşağıdaki komut yardımı ile bu bilgiler edinilebilir.

```
root@bgapentest:~# uname -a
Linux bgapentest 3.12-kali1-amd64 #1 SMP Debian 3.12.6-2kali1 (2014-01-06) x86_64 GNU/Linux
```

Dosyalara Ait Bütünlük Hash(Özet) Değerlerinin Kontrol Edilmesi

Herhangi bir canlı sistem üzerinde çalışma yapılacaksa sistemin yedeğinin ve tüm dosyaların hash(özet) bilgilerinin önceden alınmış olması önemlidir. Hash değerleri dosyaların parmak izi gibidir, bir dosya değiştiğinde bu özet de değişmektedir. Sistemdeki tüm dosyaların "hash" değerleri ve boyut bilgisi, sahiplik (file owner) gibi özelliklerinin olası bir sızmaya karşı yedeklenmesi önemlidir. (Linux üzerinde "find" komutu kullanarak "/etc/" dizini içerisindeki dosyaları ve sahipliklerini gösterir)

```
root@bgapentest:~# find /etc -type f -ls
1588664   4 -rw-------   1 root     root        1716 Aug 15 20:05 /etc/shadow-
1570743   4 -rw-r--r--   1 root     root         578 Mar 12  2012 /etc/darkstat/init.cfg
1570628   4 -rw-r--r--   1 root     root         356 Dec 30  2012 /etc/bindresvport.blacklist
1572069   4 -rwxr-xr-x   1 root     root         249 Mar  7  2013 /etc/resolvconf/update-libc.d/avahi-
daemon
1571499   4 -rwxr-xr-x   1 root     root        1806 Nov 14  2011 /etc/network/if-post-down.d/ifenslave
1571500   4 -rwxr-xr-x   1 root     root         795 Sep 30  2007 /etc/network/if-post-down.d/vlan
1571501   4 -rwxr-xr-x   1 root     root        1070 Dec 18  2009 /etc/network/if-post-down.d/wireless-
tools
1571495   4 -rwxr-xr-x   1 root     root         332 Mar 14  2013 /etc/network/if-down.d/upstart
1571494   4 -rwxr-xr-x   1 root     root         172 Jun 18  2013 /etc/network/if-down.d/openvpn
1571518   4 -rwxr-xr-x   1 root     root        1483 Mar 14  2013 /etc/network/if-up.d/upstart
1571512   4 -rwxr-xr-x   1 root     root        1699 Nov 14  2011 /etc/network/if-up.d/ifenslave
1571517   4 -rwxr-xr-x   1 root     root         173 Jun 18  2013 /etc/network/if-up.d/openvpn
1571516   4 -rwxr-xr-x   1 root     root         849 Feb  8  2013 /etc/network/if-up.d/openssh-server
1571510   4 -rwxr-xr-x   1 root     root         484 Mar  7  2013 /etc/network/if-up.d/avahi-daemon
1571515   8 -rwxr-xr-x   1 root     root        4490 Aug 19  2013 /etc/network/if-up.d/mountnfs
1571511   4 -rwxr-xr-x   1 root     root        1675 Apr 28  2012 /etc/network/if-up.d/ethtool
1571514   4 -rwxr-xr-x   1 root     root         866 Oct 21  2012 /etc/network/if-up.d/miredo
1571513   4 -rwxr-xr-x   1 root     root         578 Sep 30  2007 /etc/network/if-up.d/ip
1571946   4 -rw-r--r--   1 root     root         671 May 24 13:03 /etc/network/interfaces
1571505   8 -rwxr-xr-x   1 root     root        6157 Nov 14  2011 /etc/network/if-pre-up.d/ifenslave
```

Hash değerleri

Md5sum = (Message-Digest algorithm 5) girilen veriye ait 128 bitlik bir özet değeri üretir.
Sha1sum = SHA-1 (Secure Hash Algorithm 1) girilen veriye ait 160 bitlik bir özet değeri üretir
Linux üzerinde bunu sha1sum ve sha256sum gibi komutlarla bu hash değerlerini alabilirsiniz. Bu hash komutları genellikle birçok dağıtımla birlikte gelmektedir. Hash değerlerinin güvenilirliği açısından da birden fazla hash algoritması ile değer elde etmek faydalı olmaktadır. Örnek hash değerleri aşağıdaki şekilde incelenebilir.

```
root@bgapentest:~# find /etc -type f |xargs md5sum |head
709f91514828bd175ad096bb1437f5fa  /etc/shadow-
caa6f5bd68ec9ecaf917e4f3e572fdca  /etc/darkstat/init.cfg
d2d1b996add35b65f64a22e9c8413632  /etc/bindresvport.blacklist
2cf53ff5a00f9d1fed653a2913de5bc7  /etc/resolvconf/update-libc.d/avahi-daemon
8cc5418ae8f2c12eccc9ee1435a2c35f  /etc/network/if-post-down.d/ifenslave
af8ff6431490f05395844386b1c71210  /etc/network/if-post-down.d/vlan
1f6530d0aee88247fe5001fe2f5f50d0  /etc/network/if-post-down.d/wireless-tools
1a0205ddbc1446782a8d4d818e97d8a5  /etc/network/if-down.d/upstart
6f027552ae527133e46efb9201e0b9fc  /etc/network/if-down.d/openvpn
```

Bunlara ek olarak sistemde "md5deep" yazılımı kurulu ise aşağıdaki komutları kullanılarak da alt dizinlerle birlikte (recursive olarak) tüm dosyalara ait hash bilgileri sistemden alınabilir.

```
root@bgapentest:~# md5deep  -r /etc/
709f91514828bd175ad096bb1437f5fa  /etc/shadow-
caa6f5bd68ec9ecaf917e4f3e572fdca  /etc/darkstat/init.cfg
d2d1b996add35b65f64a22e9c8413632  /etc/bindresvport.blacklist
2cf53ff5a00f9d1fed653a2913de5bc7  /etc/resolvconf/update-libc.d/avahi-daemon
4c82dbf7e1d8c5ddd70e40b9665cfeee  /etc/network/if-post-down.d/wpasupplicant
8cc5418ae8f2c12eccc9ee1435a2c35f  /etc/network/if-post-down.d/ifenslave
af8ff6431490f05395844386b1c71210  /etc/network/if-post-down.d/vlan
1f6530d0aee88247fe5001fe2f5f50d0  /etc/network/if-post-down.d/wireless-tools
6dbf1a91ab420a99d1205972d6401e67  /etc/network/if-post-down.d/avahi-daemon
1a0205ddbc1446782a8d4d818e97d8a5  /etc/network/if-down.d/upstart
…
```

Canlı sistem üzerinde analiz yaparken kullanılacak komutlar sistem üzerindeki komut ve programlar olmamalıdır. Canlı sistem zaten şüphelenilen sistem olduğu için kullanılan komutlar güvenilir değildir. Benzeri bir sistem üzerinde temiz olduğu kesin olan programlar kayıt edilemez/yazılamaz bir aygıt aracılığı ile sisteme bağlanarak kullanılmalıdır. Canlı sistem üzerinden komutlar çalıştırılacaksa bu komutların "hash" değerleri kontrol edilerek arka kapı ve olası bir zararlı yazılımın olmadığından emin olunmalıdır.

Olası Sızma Yöntemleri ve Giriş Noktalarının Tespiti

Analiz çalışmasına başlamadan önce sistemi hızlı bir şekilde gözden geçirerek, saldırgan gözüyle sisteme yaklaşarak, "Hangi yol ve yöntemlerle bu sisteme sızılmış olabilir?" sorusunun cevabını bulmak yerinde olacaktır. Aksi halde yoğun kullanıma sahip bir Linux sistemde analiz yapmak: "samanlıkta saman rengi bir iğne" aramaya benzeyecektir.

Not: Bu aşamada sisteme girilen yolun tespit edilebilmesi için "Nmap, Nessus, OpenVAS" gibi yazılımlar ilk aşamada çalıştırılarak standart, bilinen bir yollarla sistem ele geçirilmişse hızlı bir şekilde bilgi edinilebilir.

Genel olarak bir Linux/UNIX sisteme yapılabilecek sızma yöntemleri aşağıdaki başlıklar altında sınıflandırabilir.

- Brute-force Parola Saldıları ile Sisteme Erişilmesi
    - HTTP, FTP, TELNET, SSH gibi servislere kaba kuvvet saldırıları.
- Web uygulama Zafiyetleri Kullanılarak Sisteme Erişilmesi
    - Çalışan bir web uygulama sunucusunun sahip olduğu bir zafiyetin istismar edilmesi.
- Çalışan Servislerin Hedef Alınması
    - Tomcat, Distcc gibi Linux sunucular üzerinde bolca kullanılan servislerin istismar edilmesi.
- Fiziksel Erişim Yöntemleri ile Sisteme Erişilmesi
    - Fiziksel olarak erişilen sunucuların istismar edilmesi.
- Ağ Seviyesinde Yapılan Saldırılar
    - Man-in-the-middle, Sniffing, DDOS, DNS saldırıları gibi tekniklerin kullanılması.
- Kernel Seviyesinde Yetki Yükseltme Saldırıları ile Sisteme Erişilmesi
    - Linux çekirdeklerinin önceki sürümlerinde tespit edilmiş zafiyetlerin istismar edilmesi ile sisteme zarar verilerek erişilebilir olması.

Son bir yılda BGA Bilgi Güvenliği olarak incelediğimiz olayların tamamına yakınının sonuçları Linux sistemlere gerçekleştirilen sızma olaylarının "WEB tabanlı zafiyetlerden" kaynaklandığını göstermektedir.

Bu yazıda da web tabanlı sızma sonrası sistem analizine özellikle değinilecektir.

Linux Sistem Analizi

İşletim sistemi analiz etmek için birden fazla nokta vardır ve genellikle saldırı sonrası giriş yolu kesin belli değilse, analizi yapacak kişi ilk olarak hangi dosyaya, hangi log dosyasına bakacağına karar veremez. Bu tip durumlar için analiz yapacak uzmanın önceden hazırlanmış bir kontrol listesinin olmasında fayda vardır. Doğrudan bu kontrol listesi üzerinden ilerleyerek adım adım sistem analizi gerçekleştirilebilir.

Temel olarak incelenmesi gereken maddeler başlıklar halinde aşağıda verilmiştir.

- Yerel ve uzak bağlantı detayları
- Olay anına ve sonrasına ait bellek dökümü incelemesi
- Sistemde aktif olarak işlemde bulunan dosyaların analizi
- Son 3-5 günde değiştirilmiş, yeni eklenmiş dosyaların analizi
- Dosya izinleri 777 olan dosyaların incelenmesi
- Sisteme yapılmış başarılı, başarısız giriş denemeleri
- Sistemde anlık olarak bulunan kullanıcıların ve bağlandıkları ip adreslerinin incelenmesi
- Sistem üzerinde aktif olarak çalışan ağ servislerinin ve bunları çalıştıran kullanıcıların bilgileri
- SSH anahtarlarının incelenmesi
- Zamanlanmış görevlerin incelenmesi
- Sistem üzerinde paylaşımda dosya olup olmadığının incelenmesi
- Tüm aktif kullanıcılara ait komut geçmişlerinin incelenmesi
- Aktif tüm process ve alt processlerin incelenmesi
- Sistemde kurulu olan ve hizmet veren servislere ait yapılandırma dosyalarının incelenmesi
- Sistemde kurulu olan ve hizmet veren servislere ait log/kayıt dosyalarının incelenmesi
- Disk incelemesi
- Zararlı yazılım analizi

Yerel ve uzak bağlantı detayları & Ağ Servisleri

Linux sistem üzerinde aktif bağlantıların tespit edilmesi için "netstat" ve "lsof" kullanılabilir. Bu bağlantıların kaydedilmesi daha sonraki incelemelerde de faydalı olacaktır.

"ss & netstat" Kullanılarak Network İstatistiklerinin Okunması

Bu istatistikler network protokollerine ait sistemdeki aktif olarak kullanılan konfigürasyonları bilgilerini de vermektedir. Örneğin "*IP forwarding=1"* olması sistemin kendi üzerine gelen trafiği yönlendirebilmesi anlamına gelmektedir.

```
$ netstat -s
```

```
$ ss -s
```

Örnek çıktı:

```
Ip:
    Forwarding: 1
    1537930 total packets received
    0 forwarded
    0 incoming packets discarded
    1537561 incoming packets delivered
    1104636 requests sent out
    24 outgoing packets dropped
    103 dropped because of missing route
Icmp:
    823 ICMP messages received
    0 input ICMP message failed
    ICMP input histogram:
        destination unreachable: 823
    768 ICMP messages sent
    0 ICMP messages failed
    ICMP output histogram:
        destination unreachable: 768
IcmpMsg:
        InType3: 823
        OutType3: 768
Tcp:
    3790 active connection openings
    0 passive connection openings
    13 failed connection attempts
    1130 connection resets received
    17 connections established
    1521722 segments received
    1092307 segments sent out
    1048 segments retransmitted
```

```
   25 bad segments received
   4570 resets sent
...
```

“lsof” Kullanılarak Aktif Bağlantıların Listelenmesi

“netstat ve ss” komutları Tcp, Udp bağlantılarını, soketlerin durumlarını öğrenmek için kullanılabilir. Paketleri detaylı bir şekilde görebilir ve aynı zamanda kaydedebilirsiniz.

“watch” komutu ise aktif çıktıyı yenilemek içindir.

```
$ watch -d -n1 lsof -i
```

“netstat” Kullanılarak Aktif Bağlantıların Listelenmesi

```
$ watch -d -n1 'netstat -anp | grep -i stream'
```

Netsat &Ss kullanılarak processlerin açtığı/kullandığı socketler listelenmesi

```
$ ss -nap
```

```
$ netstat -tulpn
```

Örnek çıktı:

```
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 192.168.124.1:53        0.0.0.0:*               LISTEN      1396/dnsmasq
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN      2108/cupsd
tcp6       0      0 ::1:631                 :::*                    LISTEN      2108/cupsd
udp        0      0 0.0.0.0:5353            0.0.0.0:*                           2323/chrome
udp        0      0 0.0.0.0:5353            0.0.0.0:*                           976/avahi-daemon: r
udp        0      0 0.0.0.0:54670           0.0.0.0:*                           976/avahi-daemon: r
udp        0      0 192.168.124.1:53        0.0.0.0:*                           1396/dnsmasq
udp        0      0 0.0.0.0:67              0.0.0.0:*                           1396/dnsmasq
udp        0      0 0.0.0.0:68              0.0.0.0:*                           9686/dhclient
udp        0      0 127.0.0.1:323           0.0.0.0:*                           1017/chronyd
```

Olay anına ve sonrasına ait bellek dökümü incelemesi

Olay esnasında veya sonrasında saldırgan tespit edilmemek için izlerini silmek isteyecektir. Bunun için de dosyaları, logları vb. arkada bıraktığı izleri silecektir. Bunun yanında disk üzerinde sildiği veriler geri getirilemeyecek durumda olabilir. Bu durumda bellek(RAM) olayı aydınlatmak adına birçok bilgi verebilir. Bellek, üzerine veri yazılana kadar sonlandırılan proseslere ait bilgileri bulundurmaktadır. Ayrıca bellek geçici olarak veri bulundurduğu için sistem kapatılmamalıdır. Bellek analizi için yapılacak işlemler de bellek üzerinde yapılacağından bu işlem elde edilmek istenen verilerin üzerine yazabilir. Bu olası kayıp ihmal edilebilir olsa da belleği analiz edebilmek için bu işlem gereklidir. Bellek üzerinden elde edilmek istenen bu verilerin kaybını azaltmak için en iyi çözüm belleğin imajının alınıp ayrıca incelenmesi olacaktır.

İmaj dosyası binary (ikili) olduğundan üzerinde yapılacak analiz için eskiden *strings* ve *grep* gibi komutlar kullanılarak bilgiler elde edilmeye çalışılırken günümüzde *volatility* gibi araçlar ile çalışan prosesler, aktif ağ bağlantıları, process ağaçları, sonlandırılmış prosesler gibi bu işlemlerden çok daha fazlası elde edilebilmektedir. Elde edilecek bilgilerin adli delil niteliği taşıması isteniyorsa, bellek imajı almak ve imaj üzerinde analiz yapmak için kullanılacak uygulamaların genel kurullarca kabul edilir olmasında fayda olacaktır.

Olay esnasında ve olay sonrasında alınacak iki farklı bellek imajı da kıyaslanarak sistem üzerinde gerçekleşen değişiklikler hakkında bilgi sahibi olunabilir. Bu işlem yapılan saldırının aydınlatılması açısından analizi gerçekleştiren kişiye bilgi sağlayacaktır. Çalışmakta olan ve olay sonrasında sonlanan prosesleri tespit etmek veya olay sonrası oluşan prosesleri tespit etmek ya da değişmiş olabilecek ağ bağlantılarını elde etmek gibi olayın aydınlatılması açısından çok faydalı bilgiler sunacaktır.

Sistemde aktif olarak işlemde bulunan dosyaların analizi

Unix/Linux üzerinde her şey dosyadan ibarettir, tüm pipelar, dizinler, socketler, aygıtlar… Herhangi bir process'in eriştiği tüm bu dosyalar lsof ile tespit edilebilir. Açık olan tüm network socketleri ve aktif bağlantıları listelemek için -i parametresi kullanılmaktadır. Zararlı olarak nitelendirilen bir process'in hangi dosyayı kullandığı bilgisi bize hangi socket üzerinden hangi bağlantıyı kurduğuna kadar bilgi sağlayabilir.

```
$ lsof -i
```

Örnek çıktı:

```
avahi-dae  976  avahi  15u  IPv4  22814    0t0  UDP *:mdns
avahi-dae  976  avahi  16u  IPv6  22815    0t0  UDP *:mdns
avahi-dae  976  avahi  17u  IPv4  22816    0t0  UDP *:54670
avahi-dae  976  avahi  18u  IPv6  22817    0t0  UDP *:48925
chronyd   1017 chrony   1u  IPv4  24681    0t0  UDP bgapentest:323
chronyd   1017 chrony   2u  IPv6  24682    0t0  UDP localhost:323
dnsmasq   1396 nobody   3u  IPv4  29085    0t0  UDP *:bootps
dnsmasq   1396 nobody   5u  IPv4  29088    0t0  UDP bgapentest:domain
dnsmasq   1396 nobody   6u  IPv4  29089    0t0  TCP bgapentest:domain (LISTEN)
cupsd     2108   root   9u  IPv6  39168    0t0  TCP localhost:ipp (LISTEN)
cupsd     2108   root  10u  IPv4  39169    0t0  TCP bgapentest:ipp (LISTEN)
```

Aktif bağlantılar ve dinlenilen portlar (LISTEN) durumunda gözükmektedir.

```
$ lsof -Pni
```

-Pni parametresi ile "hostname" çözümlemesi yapmadan çıktı alınabilir. Hostname çözümlemesi yapmadan sonuç almak, saldırganın olası analiz işleminden haberi olmaması açısından önemlidir. Aksi durumda DNS sorgulaması gerçekleştirileceğinden analiz işlemi yapıldığının farkına varılabilir.

```
COMMAND    PID   USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
avahi-dae  976  avahi  15u  IPv4  22814    0t0  UDP *:5353
avahi-dae  976  avahi  16u  IPv6  22815    0t0  UDP *:5353
avahi-dae  976  avahi  17u  IPv4  22816    0t0  UDP *:54670
avahi-dae  976  avahi  18u  IPv6  22817    0t0  UDP *:48925
chronyd   1017 chrony   1u  IPv4  24681    0t0  UDP 127.0.0.1:323
chronyd   1017 chrony   2u  IPv6  24682    0t0  UDP [::1]:323
dnsmasq   1396 nobody   3u  IPv4  29085    0t0  UDP *:67
dnsmasq   1396 nobody   5u  IPv4  29088    0t0  UDP 192.168.124.1:53
dnsmasq   1396 nobody   6u  IPv4  29089    0t0  TCP 192.168.124.1:53 (LISTEN)
cupsd     2108   root   9u  IPv6  39168    0t0  TCP [::1]:631 (LISTEN)
```

Sadece TCP ve UDP bağlantılarını listelemek için -iTCP ve -iUDP parametrelerini kullanabiliriz.

Son Günlerde Değiştirilmiş ve Yeni Eklenmiş Dosyaların Analizi

Ele geçirilmiş bir Linux sunucu üzerinde saldırganın değişiklik yaptığı, yapmak zorunda kaldığı ya da yanlışlıkla değiştirdiği dosyaları tespit etmek bu süreçte yardımcı olabilir. Örneğin bir zararlı indirilip, önemli olmayan bir dizinde kullanılmış ve kendisini bir başka çalışan process'e migrate etmiş olabilir. Bu durumda orijinal dosyayı bulmak işimize yarayabilir. Ya da kullanılan bir sistem dosyasını değiştirip manipüle etmiş olabilir. Linux üzerinde son günlerde düzenlenmiş, yeni eklenmiş dosyaları "find" komutunu kullanarak tespit etmek oldukça kolaydır.

Son zamanlarda değiştirilmiş/düzenlenmiş tüm dosyaları tespiti:

Aşağıdaki komut ile "/etc" dizini ve tüm alt dizinlerde değiştirilmiş dosyaları listeler.
Zaman formatı: Yıl/Ay/Gün:Saat şeklindedir.

```
$ find /etc -type f -printf '%TY-%Tm-%Td %TT %p\n' | sort -r
```

-type f : sadece dosyaları listeler, dizinleri de dahil etmek istiyorsanız çıkartın.

sort -r : Reverse sıralama yapar.

```
2017-08-02 11:33:02.7185808290 /etc/hostname
2017-08-02 10:26:09.6337759860 /etc/resolv.conf
2017-08-02 00:31:50.8860337160 /etc/ld.so.cache
2017-08-01 23:48:56.0225050230 /etc/cups/subscriptions.conf
2017-08-01 18:21:13.9269096490 /etc/cups/subscriptions.conf.O
2017-08-01 16:41:36.5675491120 /etc/gshadow
2017-08-01 16:41:36.3115482640 /etc/group
2017-08-01 16:38:32.1949386750 /etc/udev/rules.d/60-vboxdrv.rules
2017-08-01 16:33:05.9052330670 /etc/dconf/db/ibus
2017-07-31 13:32:23.6215468630 /etc/hosts
...
```

Son 60 dakika içerisinde değiştirilmiş dosyaların tespiti:

Aşağıdaki komut ile son 60 dakika içerisinde belirlenen dizinde eklenmiş, düzenleme yapılmış tüm dosyaları tespit edebilirsiniz. Dizinleri de dahil etmek için -type f parametresini kaldırın.

```
$ find /hedef_dizin -type f -mmin -60
```

Son 2 gün içerisinde değiştirilmiş dosyaların tespiti:

Aşağıdaki komut ile son 2 gün içerisinde belirlenen dizinde eklenmiş, düzenleme yapılmış tüm dizinleri tespit edebilirsiniz.

```
$ find /hedef_dizin -type f -mtime -2
```

"-mtime" parametresi ile "modify time" yani düzenleme zamanı belirlenir. Son 20 gün için "-mtime -20" yazabilirsiniz. Bu tüm dizinler ve alt dizinler için geçerli olacaktır.

Son 1 Hafta içinde "düzenlenmiş" ama son 3 gün içerisinde "düzenlenmemiş" dosyaların tespiti:

Son 1 hafta içerisinde düzenlenmiş dosyaları araştırıyor olabilirsiniz fakat son 2 gün içerisinde bazı değişikliklerin yapıldığını biliyorsunuz. O halde bu durumu atlamak için aşağıdaki şekilde kullanabilirsiniz.

```
$ find /hedef_dizin -type f -mtime -7 ! -mtime -3
```

Dosyaların daha detaylı bir şekilde listelenmesi:

Yukarıda kullanılan komutların tamamı değişiklik yapılmış dosyaların adreslerini vermektedir. Daha detaylı bir inceleme yapmak için, dosyanın izinleri, sahipliği gibi bilgilere de bakılmalıdır. O zaman "exec" kullanarak daha detaylı çıktı alabiliriz.

```
$ find /hedef_dizin -type f -mmin -60 -exec ls -al {} \;
```

Alternatif olarak "xargs" komutu da kullanılabilir.

```
-r--r-----+ 1 root sys 32 Aug  2 13:15 /run/cups/certs/0
-rw-r--r--. 1 root root 162 Aug  2 13:03 /run/systemd/inhibit/99
-rw-------. 1 root root 149 Aug  2 13:05 /run/systemd/journal/streams/9:452091
-rw-------. 1 root root 149 Aug  2 13:05 /run/systemd/journal/streams/9:452090
-rw-------. 1 root root 152 Aug  2 12:23 /run/systemd/journal/streams/9:426227
-rw-------. 1 root root 152 Aug  2 12:23 /run/systemd/journal/streams/9:426226
```

Dosya izinleri 777 olan dosyaların incelenmesi

Chmod, Unix/Linux sistemlerde dosya ve dizinlerin yetkilerini belirleyen ve "change mod" anlamına gelen komuttur. "777" izni hem okuma hem yazma hem de çalıştırma izni demektir. Bu dosyalar sistemde en tehlikeli dosyalar haline gelebilmektedir; çünkü saldırgan herhangi bir kullanıcı hesabıyla bu dosyayı düzenleyip, çalıştırabilir.

Dosya izinleri 777 olan tüm dosyaları tespit etmek için:

Daha önce kullandığımız gibi bu dosyaları tespit etmek için de "find" komutu kullanabiliriz. Find komutunun -perm parametresini kullanacağız. Dizinleri de aramak istiyorsanız -type f parametresini kaldırın.

```
$ find / -type f -perm 0777
```

Sisteme yapılmış başarılı, başarısız giriş denemeleri

Olası bir saldırıda en önemli şeylerden bir tanesi sisteme kimin giriş yaptığı/yapamadığıdır. Bunu Linux sistemler üzerinde tespit etmek için birkaç farklı yol vardır.

"Last" komutu ile sisteme giriş denemelerini tespit etmek

Last komutu uzaktan veya local olarak sisteme giriş yapmış kullanıcıların listesini /var/log/temp dizinindeki dosyayı okuyarak bize gösterir.

```
$ last -f /var/log/wtmp
```

Örnek çıktı:

```
user  tty2        /dev/tty2       Tue Aug  1 16:23   still logged in
reboot   system boot  4.11.11-300.fc26 Tue Aug  1 19:23   still running
user  tty2        /dev/tty2       Mon Jul 31 02:42 - crash (1+16:40)
reboot   system boot  4.11.8-300.fc26. Mon Jul 31 02:41   still running

wtmp begins Mon Jul 31 02:41:17 2017
```

Herhangi bir kullanıcının sisteme ne zaman girişlerini listelemek için:

```
$ last [username]
```

Herhangi Bir TTY için:

```
# last [TTY]
```

Sistemin reboot edilme zamanlarını listelemek için:

```
$ last reboot
```

Sistemde Aktif olarak bağlı kullanıcıları listelemek için:

Aktif kullanıcıları who komutu ile tespit edilebilir.

```
$ who -aH
```

-aH parametresi ile tüm istatistikler elde edilebilir.

```
NAME     LINE        TIME            IDLE        PID COMMENT  EXIT
         system boot  2017-08-01 19:23
         run-level 5  2017-08-01 16:23
user  + tty2        2017-08-01 16:23  old        1715 (/dev/tty2)
```

Journalctl ile Sisteme girişleri tespit etmek

Eğer kullanılan dağıtım “Systemd” kullanıyor ise /var/log/ dizinini karıştırmaktansa alternatif olarak journalctl kullanılabilir. Journal günlük anlamına gelmektedir.

```
journalctl -u 'systemd-logind'  --since "today" --until "tomorrow"
```

"/var/log/auth.log" Kullanılarak Girişleri tespit etme:
Birçok Linux sistemde auth.log dosyası bulunmaktadır, bu dosyayı okuyarak tüm authentication(kimlik doğrulama) kayıtlarına ulaşabilirsiniz.

```
$ less /var/log/auth.log
```

Örnek çıktı:

```
May  3 18:20:45 localhost sshd[585]: Server listening on 0.0.0.0 port 22.
May  3 18:20:45 localhost sshd[585]: Server listening on :: port 22.
May  3 18:23:56 localhost login[673]: pam_unix(login:session): session opened fo
r user root by LOGIN(uid=0)
May  3 18:23:56 localhost login[714]: ROOT LOGIN  on '/dev/tty1'
Sep  5 13:49:07 localhost sshd[358]: Received signal 15; terminating.
Sep  5 13:49:07 localhost sshd[565]: Server listening on 0.0.0.0 port 22
```

Sistemde anlık olarak bulunan kullanıcıların ve bağlandıkları ip adreslerinin incelenmesi

Unix/Linux sistemlerde "ProcFS" adı verilen özel bir dosya sistemi bulunmaktadır. Bu dosyalar process'ler ait bilgiler içermektedir ve işletim sistemi boot edilirken "/proc" dizinine mount edilmektedir.

"w" komutu bu dosya sisteminin bir parçası olarak çalışır ve açıklaması "kim sisteme bağlı ve şu anda ne yapıyor" anlamına gelmektedir. Aşağıda "w" komutu ile sisteme bağlı olan kullanıcıları ve aktif olarak ne iş yaptıklarını gösteren komutları bulabilirisiniz.

```
$ w
```

Örnek Çıktı:

```
user2    pts/0    192.168.1.6     14:10    3:24m  2.15s  0.00s dbus-launch --auto
root     pts/1    192.168.1.6     14:51    1:41m  0.16s  0.00s pager -s
user1 pts/2    192.168.1.6     14:52   13:07   0.41s  0.02s vi /etc/passwd
root     pts/3    192.168.1.6     17:21    3.00s  0.12s  0.01s w -h
```

1. USER – Kullanıcı adı.
2. TTY – PTS/0 veya Konsol.
3. FROM – Uzak kullanıcı veya IP adresi.
4. LOGIN@ – Giriş zamanı.
5. IDLE – Idle zamanı.
6. JCPU – The JCPU time is the time used by all processes attached to the tty.
7. PCPU – The PCPU time is the time used by the current process displayed in WHAT field.
8. WHAT – Kullanıcın aktif olarak hangi işle uğraştığı

Bunun dışında yukarıda anlatılan netstat, ss, lsof komutları ile bağlı olan kullanıcı denetlenebilir.

SSH anahtarlarının incelenmesi

Linux bir sunucuda SSH anahtarlarını incelemek önemlidir. SSH uzak bağlantısı ile sunucuya parola girişi ile bağlantı sağlanması yerine bir açık anahtar kullanılabilir. Bu durumda kullanıcı özel anahtarını çaldırmış ve başkası tarafından kullanıyor olabilir. Bu kısımda SSH anahtarlarını ve SSH loglarını inceleyeceğiz.

Authorized_Keys dosyasının incelenmesi

Linux sistemlerde yetki verilmiş SSH açık anahtarları "/home/username/.ssh/authorized_keys" dosyasında bulunmaktadır. Bu dosyanın içerisindeki açık anahtarlar bu sisteme özel anahtarı ile bağlanabilecek olan kullanıcıları gösterir. Bu dosya incelenerek kimlerin erişim izni olduğu öğrenilebilir.

SSH loglarının incelenmesi ve Bağlı kullanıcıların incelenmesi

Eğer sistemde Rsyslog/Syslog kurulu ise /var/log/auth.log dosyası incelenebilir. Yanlış parola giriş denemeleri için:

```
grep sshd.\*Failed /var/log/auth.log | less
```

Çıktı:

```
Aug 18 11:00:57 testvps sshd[5657]: Failed password for root from 95.58.255.62 port 38980 ssh2
Aug 18 23:08:26 testvps sshd[5768]: Failed password for root from 91.205.189.15 port 38156 ssh2
Aug 18 23:08:30 testvps sshd[5770]: Failed password for nobody from 91.205.189.15 port 38556 ssh2
```

Başarısız bağlantılar için:

```
grep sshd.*Did /var/log/auth.log | less
```

Çıktı:

```
Aug  5 22:19:10 testvps sshd[7748]: Did not receive identification string from 70.91.222.121
Aug 10 19:39:49 testvps sshd[1919]: Did not receive identification string from 50.57.168.154
Aug 13 23:08:04 testvps sshd[3562]: Did not receive identification string from 87.216.241.19
Aug 17 15:49:07 testvps sshd[5350]: Did not receive identification string from 211.22.67.238
```

Journalctl kullanarak SSHD loglarına erişmek

Journalctl kullanarak ssh daemon'ı sshd loglarına bakarak bilgi edinilebilir.

```
journalctl -u sshd |tail -100
```

Zamanlanmış görevlerin incelenmesi

Crontab, Unix/Linux sistemlerde zamana bağlı olarak çalışan bir görev planlayıcısıdır. Yapılmak istenilen işi belirli zaman aralıklarında çalıştırma, kontrol etme gibi işlemleri yaptırabileceğiniz bir uygulamadır.

Sızılmış bir sistemde kullanılan aktif görevler manipüle edilmiş olabilir veya yeni eklenmiş bir görev olabilir. Bu durumda zaman planlı görevleri kontrol etmek yerinde olacaktır.

Planlanmış görevler "/etc/crontab" adresinde bulunan dosyanın içerisindedir. Bu dosya kontrol edilebilir ya da crontab komutunu kullanabiliriz.

```
$ crontab -l
```

Örnek çıktı:

```
30 08 10 06 * /home/user/full-backup
```

Sistem üzerinde paylaşımda dosya olup olmadığının incelenmesi

SAMBA, SMB protokolünün yeniden implement edilmiş versiyonudur. Unix/Linux sistemlerde NFS alternatif olarak kullanılabilir. NFS ve Samba ile paylaşımda olan dosya/dizinler var ise bunları tespit etmek olası bir saldırı da çok önemli bir yere sahip olabilir. Saldırganın bu erişimler sayesinde diğer sunucu ve cihazlara erişmesi mümkündür.

NFS ile Aktif Paylaşımda Bulunan Dizin ve Dosyaların Tespit Edilmesi

NFS(Network File System), Linux dünyasında oldukça fazla kullanılan bir dosya paylaşım sistemidir. NFS ile uzak bilgisayara bağlanmış bir disk, dizin ve dosyaları tespit etmek önemlidir.

Df komutunu kullanarak incelemek:

Df komutu, Linux sistemlerde disk alanı hakkında bilgi almak için kullanılır. Sisteme mount(bağlamak) edilmiş dizinleri de tespit etmek için kullanılabilir.

```
$ df -h
```

Örnek çıktı:

```
Filesystem            Size  Used Avail Use% Mounted on
devtmpfs              5.8G     0  5.8G   0% /dev
tmpfs                 5.8G  127M  5.7G   3% /dev/shm
tmpfs                 5.8G  2.0M  5.8G   1% /run
tmpfs                 5.8G     0  5.8G   0% /sys/fs/cgroup
/dev/mapper/fedora-root  196G  6.4G  180G   4% /
tmpfs                 5.8G  3.2M  5.8G   1% /tmp
/dev/sda3             976M  160M  749M  18% /boot
```

Bağlı olan bir NFS dosya sistemi olsaydı bunu da boyut bilgisiyle bize gösterebilir.

Mount komutu ile bağlı NFS dosya sistemlerini tespit etmek

Sistemin yüklü olduğu dosya sisteminin dışında ikinci bir disk, CDROM, NFS ne olursa olsun, bu parçaya erişmeden önce, sistemde var olan bir dizine iliştirilmesi gerekir. Bu iliştirmeye bölüm eşleme denir. Mount komutu ile bu eşleşmeleri görebiliriz.

```
$ mount
```

Mount komutu ile NFS dosya sistemlerini listelemek için:

```
mount -l | grep 'type nfs' | sed 's/.* on \([^ ]*\) .*/\1/'
```

Eğer sistemde çalışan bir SAMBA varsa:

Samba ile birlikte gelen 2 tane uygulama vardır. Bunlardan birisi “smbtree” bir diğeri ise “smbstatus” bu araçlar Samba ile paylaşılan dosyaları görüntülemeye yardımcı olacaktır. [3]

```
smbclient -L localhost
```

Örnek çıktı:

```
Domain=[WORKGROUP] OS=[Unix] Server=[Samba 4.1.6-Ubuntu]

Sharename       Type      Comment
---------       ----      -------
print$          Disk      Printer Drivers
IPC$            IPC       IPC Service (host-name server (Samba, Ubuntu))
hp1320          Printer   Hewlett-Packard hp LaserJet 1320 series
HP-LaserJet-1200 Printer   HP LaserJet 1200
Public          Disk
Domain=[WORKGROUP] OS=[Unix] Server=[Samba 4.1.6-Ubuntu]

Server               Comment
```

```
---------          -------
                host-name server (Samba, Ubuntu)

Workgroup          Master
---------          -------
WORKGROUP          HOST-NAME
```

Tüm aktif kullanıcılara ait komut geçmişlerinin incelenmesi

Unix/Linux sistemlerde "Bash" geçmişleri de kayıt edilir. Bu dosya her kullanıcının kendi dizininde "bash_history" adındaki dosyada tutulmaktadır. Root kullanıcısı da buna dahildir. Eğer çalıştığınız sistemde çok fazla kullanıcı var ise "find" komutu ile bu dosyaları bulabilirsiniz.

Tüm history kayıtlarını bulmak için:

```
find /home/ -name '*_history*'
```

Daha sonra bu dosyalar incelenebilir ve kullanıcının geçmiş komut kayıtlarına bakılabilir.

Aktif tüm proses ve alt processlerin incelenmesi

Linux sistemlerde "ps" komutu tüm çalışan prosesleri ve alt proseslerin gösterilmesi için kullanılan komuttur. Ps komutu da "w" komutu gibi bu bilgiyi ProcFS üzerinden almaktadır.

Tüm processleri listelemek için:

```
$ ps aux
```

Ağaç şeklinde görüntülemek için:

```
$ ps auxf
```

Örnek çıktı:

```
root     916  0.0  0.0     0     0 ?      S    Aug01   0:03  \_ [jbd2/dm-3-8]
root     917  0.0  0.0     0     0 ?      S<   Aug01   0:00  \_ [ext4-rsv-conver]
root     939  0.0  0.0     0     0 ?      S<   Aug01   0:00  \_ [rpciod]
root     940  0.0  0.0     0     0 ?      S<   Aug01   0:00  \_ [xprtiod]
root   26253  0.0  0.0     0     0 ?      S<   Aug01   0:00  \_ [iprt-VBoxWQueue]
root   26254  0.0  0.0     0     0 ?      S    Aug01   0:00  \_ [iprt-VBoxTscThr]
root   26393  0.0  0.0     0     0 ?      S<   Aug01   0:00  \_ [dio/dm-3]
root    9640  0.0  0.0     0     0 ?      S    10:26   0:00  \_ [irq/45-mei_me]
root   16850  0.0  0.0     0     0 ?      S    13:54   0:02  \_ [kworker/u16:1]
```

```
root    17105  0.0  0.0    0    0 ?      S   14:07   0:01  \_ [kworker/3:2]
root    18839  0.0  0.0    0    0 ?      S   15:24   0:00  \_ [kworker/u16:5]
root    19249  0.0  0.0    0    0 ?      S   15:37   0:00  \_ [kworker/u16:4]
root    19478  0.0  0.0    0    0 ?      S   15:53   0:00  \_ [kworker/u16:3]
```

Strace Kullanılarak Processlerin İncelenmesi

Strace sistem çağrılarını ve sinyalleri görüntülemek için kullanışlı bir araçtır. Yazılımın kaynak kodunun elinizde bulunmadığı süreçlerde debugging yapmanıza yardımcı olacaktır. Strace binary'nin başlangıçtan sonuna kadar çalışma süreciyle ilgili bilgi verir.

```
$ strace cp
```

Cp komutunu strace ile çalıştırdığımızda binary hakkında sistem çağrılarını görüyoruz.

```
execve("/usr/bin/cp", ["cp"], 0x7ffe165736a0 /* 57 vars */) = 0
brk(NULL)                               = 0x563330807000
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7f34f2f26000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
open("/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=92978, ...}) = 0
mmap(NULL, 92978, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f34f2f0f000
close(3)                                = 0
open("/lib64/libselinux.so.1", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0`d\0\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0755, st_size=158320, ...}) = 0
mmap(NULL, 2258128, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
0x7f34f2ada000
mprotect(0x7f34f2aff000, 2093056, PROT_NONE) = 0
mmap(0x7f34f2cfe000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x24000) = 0x7f34f2cfe000
mmap(0x7f34f2d00000, 5328, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f34f2d00000
close(3)
...
```

Strace ile özel bir sistem çağrısını takip etmek

```
$ strace -e open cp
```

Cp komutuna ait, "open" sistem çağrısını takip etmek için kullanılır.

```
$ strace -e trace=open, read cp -o output.txt
```

Birden fazla sistem çağrısını yukarıdaki parametre ile görebilirsiniz. -O parametresi çıktıyı dosyaya kaydeder.

Strace ile Process Takip Etmek

```
$ps -C firefox-bin
```

Yukarıdaki komut ile firefox-bin prosesine ait PID sayısını öğrenebiliriz. Daha sonra bu PID'i strace ile takip edeceğiz.

```
$ strace -p 1725 -o firefox_trace.txt
```

Bu komut ile Firefox yazılımının kullandığı tüm sistem çağrıları ve işlemleri firefox_trace.txt dosyasına kaydetmiş olduk. Yazılımın hem zararlı hem zararlı olmadığı durumlarda Strace bilgi almak açısından çok önemli bir araçtır.

Disk incelemesi

Linux sistemlerde diskleri takip etmek için birçok araç bulunmaktadır. Disk istatistiklerine erişmek bazı ön bilgiler sağlayabilir. Bu başlıkta disk araçlarını inceleyeceğiz.

Iostat ile Disk İstatistikleri

Iostat ile diskin yazma/okuma oranı ve sayıları sürekli olarak takip edilebilir. Önce disk verisini alıp bir süre bekleyip daha sonra aradaki farklarla birlikte sunar.
Iostat kullanabilmek için sysstat paketinin sistemde kurulu olması gerekmektedir.

```
$ iostat -y 5
```

Her 5 saniyede bir istatistikleri yeniden düzenleyecektir, CPU istatistikleri, IOWait, Idletime gibi istatistikleri görebilirsiniz. Yazma/Okuma oranları “kB_read/s and kB_write/s” şeklinde belirlenmiştir.

Örnek Çıktı:

```
Linux 4.11.11-300.fc26.x86_64 (machine)         08/03/2017      _x86_64_        (4 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          7.29    0.02    2.08    2.27    0.00   88.33

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda              18.79       114.31       142.45    1151970    1435560
dm-0             26.35       113.19       142.45    1140705    1435548
dm-1              8.53       103.19        26.84    1039937     270464
dm-2              0.01         0.23         0.00       2292          0
dm-3             15.99         9.73       115.61      98016    1165084
```

Analiz Amaçlı Sistem Log Dosyalarının İncelenmesi

Linux log ağacı oldukça detaylıdır. Herhangi bir kriz durumunda bu logların incelenmesi günü kurtarabilir. Linux'a ait loglar "/var/log" dizini altında bulunur. Öncelikle bu dosyaları ve kısa bir özetle ne işe yaradıklarını anlatalım.

- **/var/log/messages** : Global sistem loglarının tutulduğu dosyadır, sistem başlangıcı ile ilgili loglar, mail, cron, kernel, auth benzeri logları da barındırır.
- **/var/log/dmesg**: "Dmesg" komutu ile de erişilebilen bu dosyada kernel üzerinden gelen mesajlar doğrudan okunabilir. Kernel ring buffer bilgisi içermektedir. Sistem boot edildiğinden itibaren aygıtlar ile alakalı tüm mesajları içerir.
- **/var/log/auth.log:** Sistem authentication(kimlik doğrulama) loglarını bulundurur. Kullanıcı girişleri, başarısız giriş denemeleri buradan öğrenilebilir. Brute-force saldırıları buradan tespit edilebilir.
- **/var/log/boot.log** : Sistem boot sürecine ait tüm loglar bu dosya içerisinde bulunur.
- **/var/log/daemon.log:** Aktif, durmuş, başarısız tüm daemon'lar hakkında bilgilerin bulunduğu log dosyasıdır.
- **/var/log/dpkg.log:** Yüklenen, güncellenen, silinen tüm paket loglarının bulunduğu dosyadır.
- **/var/log/kern.log:** Kernel loglarını içerir, eğer özelleştirilmiş bir kernel kullanılıyor ise çok daha önemli bir dosya haline gelmektedir.
- **/var/log/lastlog:** Son kullanıcı girişlerini ve zaman bilgisini içeren log dosyasıdır. Ancak bu dosya ASCII formatında değildir. Okumak için "lastlog" komutunu kullanmanız gerekmektedir.
- **/var/log/mail.log & /var/mail/maillog:** Sistem üzerinde çalışan mail sunucusuna ait loglar bu dosyada bulunmaktadır. Örneğin sendmail loglarının tamamı burada bulunabilir.
- **/var/log/user.log:** Linux sistem üzerinde kullanıcı seviyesine ait loglar bu dosyada bulunmaktadır.
- **/var/log/Xorg.x.log:** X ile alakalı tüm loglar bu dosyada bulunmaktadır.
- **/var/log/cups:** Sistemdeki tüm printer ve bu printerdan alınmış çıktılara ait logların bulunduğu dosya.
- **/var/log/cron:** Zamanlanmış görevlere(cron) ait tüm loglar bu dosyada bulunmaktadır. Örneğin her 1 saatte bir çalışan bir script tanımlanmış ise cron loglarına bakarak olası düzensizlik tespit edilebilir.
- **/var/log/secure:** Kimlik doğrulama, yetkiler ve güvenlik ile alakalı tüm logların tutulduğu dosyadır. Bu dosya daha sonra anlatacağımız OSSEC ve benzeri yazılımlar aracılığıyla incelenebilir ve olası anormallikler tespit edilebilir.
- **/var/log/httpd/ (or) /var/log/apache2:** HttpD ve Apache web sunucularının loglarının bulunduğu dosyalardır. Bu dosyalar daha sonraki başlıklarda detaylı bir şekilde incelenecektir.
- **/var/log/audit/ :** Linux Audit daemon tarafından oluşturulan tüm logların bulunduğu dizindir.
- **/var/log/setroubleshoot/**: SElinux güvenlik mekanizmasına ait loglar için setroubleshootd daemon loglarının bulunduğu dizindir. SElinux logları olası bir saldırı için önem arz etmektedir.
- **/var/log/sa/:** Sysstat paketi ile birlikte gelen "sar" yazılımına ait logların tutulduğu dizindir. Sar komutu ile sisteme ait birçok istatistik kontrol edilebilir.

- **/var/log/samba:** SMB protokolonün Linux implementasyonuna ait logların bulunan dosyadır. Buradan paylaşılmış dosya ve dizinlere ait bilgi edinilebilir.

Web Server Apache Loglarının ve Diğer Logların İncelenmesi

Bilişim sistemlerine yönelik saldırıları belirlemek ve engellemek için aktif ve pasif olmak üzere temelde iki yöntem vardır. Aktif saldırı belirleme ve engelleme sistemleri genellikle ağ/host tabanlı çalışır NIPS/HIPS (Network / Host Intrusion Prevention System) olarak adlandırılır ve anlık ağ trafiği ya da işletim sistemi fonksiyonlarını kullanarak engelleme işlemi gerçekleştirir.

Pasif belirleme sistemleri çok farklı olabilmektedir. Bunlardan biri de sistemin loglarını inceleyerek gerçekleşmiş saldırıları belirlemektir. Saldırıların büyük çoğunluğu log(kayıt) dosyalarındaki –eğer yeterli loglama altyapısı var ve sağlıklı çalışıyorsa- anormallikler incelenerek belirlenebilir.

Konunun detayına girmeden bu yöntemin (log analizi->saldırı inceleme) ciddi eksikliklerinin bulunduğunu belirtmek gerekir. Log analizi yöntemiyle sadece sıradan saldırılar konusunda bulgular elde edilebilir. Genel web sunucu altyapısının eksikliği nedeniyle karmaşık saldırılar sadece web sunucu log analizinden bulunamaz. Ortamda paketleri olduğu gibi gören ve kaydeden başka bileşenlere ihtiyaç vardır. (IDS, FPL gibi)

Mesela web sunucular POST isteklerinin detaylarını loglamaz ve eğer saldırganın gerçekleştirdiği atak POST detayında gizli ise sunucu logunda şüpheli bir işlem olarak gözükmeyecektir. Yine saldırgan çeşitli encoding yöntemlerini kullanarak aranacak kelimelerin farklı şekillerde log dosyasında saklamasını sağlayabilir. Burada log analizi gerçekleştiren uzmanın konu hakkında etraflıca bilgi sahibi olması önemlidir.

Log Analizi

Genellikle iki şekilde log analizi gerçekleştirilir.

- Hazır araçlar kullanarak,
- UNIX/Linux sistemlerdeki cat, awk, grep, cut... gibi basit araçlar kullanarak.

Hazır araç kullanmak işlemleri hızlandırsa da false positive oranı yüksek olduğu için çıkan sonucun tekrar gözden geçirilmesi gerekmektedir. Bu yazıda hem otomatize araç hem de elle yapılan ve toplamda 100.000.000'dan fazla satır içeren yoğun bir sunucuya ait gerçekleştirilen analize dair notlar bulacaksınız.

Apache Loglarında Saldırı İmzası Arama – Log Tabanlı IDS Gerçekleştirilen her saldırı arkasında mutlaka bir iz bırakır. Bu iz bazı durumlarda saldırının gerçekleştirildiği sistem üzerinde olur bazı durumlarda -saldırganın teknik bilgi seviyesine bağlı olarak- aradaki IDS/IPS gibi pasif sistemlerde olur. Apache, IIS gibi web sunucu yazılımlarının loglarını analiz ederek saldırı imzası arayan çeşitli yazılımlar vardır. Bu yazılımların ortak özelliği kayıtlı loglar arasında daha önceden tanımlanmış belirli kelime/kelime gruplarını aramak ve buna göre uyarı vermektir.

Bu tip yazılımları kullanırken unutulmaması gereken en önemli konu Apache ve diğer web sunucular ön tanımlı olarak POST isteklerinde gelen değerleri loglamazlar. Bunun için mod_forensic gibi ya da mod_security gibi ek bileşenler kullanılmalıdır ya da POST üzerinden gerçekleştirilecek saldırıları yakalamak için WAF, Load Balancer, IPS gibi ürünlerin loglarına başvurmak gerekir.

HTTP GET / POST İstekleri

Aşağıdaki iki farklı sistem tarafından alınmış bir POST istek detayı bulunmaktadır. Bunlardan ilki web sunucu kayıtlarından alınmış, diğeri web sunucuya gidip gelen trafiği dinleyen bir sniffer tarafından alınmıştır.

POST isteğinin web sunucu logundaki çıktısı

```
127.0.0.1 – – [04/Mar/2012:02:10:10 -0500] “POST
/dvwa/login.php HTTP/1.1” 302 454 “http://localhost/dvwa/login.php”
“Mozilla/5.0 (X11; Linux i686; rv:5.0.1) Gecko/20100101 Firefox/5.0.1”
```

POST isteğinin sniffer aracılığıyla gösterimi

```
T 127.0.0.1:47635 -> 127.0.0.1:80 [AP]
POST /dvwa/login.php HTTP/1.1.
Host: localhost.
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:5.0.1) Gecko/20100101 Firefox/5.0.1.
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8.
Accept-Language: en-us,en;q=0.5.
Accept-Encoding: gzip, deflate.
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7.
Connection: keep-alive.
Referer: http://localhost/dvwa/login.php.
Cookie: security=high; XpLiCo=i68o2ejvm6jnp3b9pv083i4mi7;
PHPSESSID=uvn4olrlfd6sckjhe4eea4jno4.
Content-Type: application/x-www-form-urlencoded.
Content-Length: 49.
username=admin&password=hatali_parola&Login=Login
```

Görüleceği gibi POST isteğini detaylı olarak incelendiğinde (Network üzerinden) hangi kullanıcı adı ve parola bilgilerinin girildiği ortaya çıkmaktadır. Bu detay web sunucu loglarında gözükmeyecektir. Web sunucu loglarında sadece hangi URL’e istek yapıldığı bilgisi kayıt altına alınır.

SQLi Denemesinin Web Sunucu Logu ve Sniffer Üzerinden Analizi

Web sunucu logu:

```
127.0.0.1 – – [04/Mar/2012:02:10:10 -0500] “POST
/dvwa/login.php HTTP/1.1” 302 454 “http://localhost/dvwa/login.php”
“Mozilla/5.0 (X11; Linux i686; rv:5.0.1) Gecko/20100101 Firefox/5.0.1”
```

Sniffer üzerinden alınan çıktı

```
T 127.0.0.1:47632 -> 127.0.0.1:80 [AP]
POST /dvwa/login.php HTTP/1.1.
Host: localhost.
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:5.0.1) Gecko/20100101 Firefox/5.0.1.
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8.
Accept-Language: en-us,en;q=0.5.
Accept-Encoding: gzip, deflate.
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7.
Connection: keep-alive.
Referer: http://localhost/dvwa/login.php.
Cookie: security=high; XpLiCo=i68o2ejvm6jnp3b9pv083i4mi7;
PHPSESSID=uvn4olrlfd6sckjhe4eea4jno4.
Content-Type: application/x-www-form-urlencoded.
Content-Length: 234.
.
username=ECYH%252C%2528SELECT%2520%2528CASE%2520WHEN%2520%25282167%253D2167%2
529%2520THEN%2520ECYH%2520ELSE%25
202167%252A%2528SELECT%25202167%2520FROM%2520INFORMATION_SCHEMA.CHARACTER_SET
S
%2529%2520END%2529&password=parola&Login=Login
```

URL encode edilmiş veri decode edilirse aşağıdakine benzer bir SQL sorgusu olduğu ortaya çıkacaktır.

```
ECHO SELECT …  CASE … WHEN … THEN ECYH ELSE SELECT FROM
INFORMATION_SCHEMA.CHARACTER_SETS AND password=parola&Login
```

Bu tip POST isteği kullanılarak gerçekleştirilen saldırılar Ağ tabanlı IPS/IDS sistemleri ya da WAF/Load Balancer sistemler kullanarak da belirlenebilir. Hazır Araçlar Kullanarak Log Analizi, Scalp Web sunucu loglarından denenmiş web saldırılarını log analizi yöntemiyle bulmaya çalışır. Saldırı analizinde kullanacağı değişkenleri de PHPIDS projesinden almaktadır.

Komut satırı parametrlerini görme:

```
root@bt:/home/huzeyfe# python scalp-0.4.py
Scalp the apache log! by Romain Gaucher – http://rgaucher.info
usage:  ./scalp.py [–log|-l log_file] [–filters|-f filter_file] [–period time-frame] [OPTIONS] [–attack
a1,a2,..,an]
[–sample|-s 4.2]
–log      |-l:  the apache log file ‘./access_log’ by default
–filters   |-f:  the filter file          ‘./default_filter.xml’ by default
–exhaustive|-e:  will report all type of attacks detected and not stop
at the first found
–tough  |-u:  try to decode the potential attack vectors (may increase
the examination time)
–period|-p:  the period must be specified in the same format as in
the Apache logs using * as wild-card
ex: 04/Apr/2008:15:45;*/Mai/2008
if not specified at the end, the max or min are taken
–html    |-h:  generate an HTML output
–xml     |-x:  generate an XML output
–text     |-t:  generate a simple text output (default)
–except|-c:  generate a file that contains the non examined logs due to the
main regular expression; ill-formed Apache log etc.
–attack |-a:  specify the list of attacks to look for
list: xss, sqli, csrf, dos, dt, spam, id, ref, lfi
the list of attacks should not contains spaces and comma separated
ex: xss,sqli,lfi,ref
–output           |-o:  specifying the output directory; by default, scalp will try to write
in the same directory as the log file
–sample           |-s:  use a random sample of the lines, the number (float in [0,100]) is
the percentage, ex: –sample 0.1 for 1/1000
```

…çalıştırıldığında aşağıdaki gibi hata alınabilir.

```
# python scalp-0.4.py  –log access.101223.log –filters default_filter.xml  -e  –html
Loading XML file ‘default_filter.xml’…
The rule ‘(?:unions*(?:all|distinct|[(!@]*)?s*[([]*s*select)|(?:w+s+likes+”)|(?:likes*”%)|(?:”s*like
W*[“d])|(?:”s*(?:n?and|x?or|not ||||&&)s+[sw]+=s*w+s*having)|(?:”s**s*w+W+”)|(?:”s*
[^?ws=.,;)(]+s*[(@”]*s*w+W+w)|(?:selects*[[]()sw.,”-]+from)|(?:find_in_sets*()’ cannot be compiled
properly
```

Çözümü:

http://code.google.com/p/apache-scalp/issues/list

Belirli Tipteki Saldırıları Belirleme Sadece belirli tipteki saldırıları aratmak için –a sqli, xss gibi parametre kullanılabilir. Diretct…

```
85.95.238.173 – – [11/Feb/2012:00:50:50 -0600] “GET
/nessus……winntwin.ini HTTP/1.1” 404 21816 “-”
“Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)”
Reason: “Detects basic directory traversal”
85.95.238.173 – – [11/Feb/2012:00:50:50 -0600] “GET
/exchweb/bin/auth/owalogon.asp?url=http://12345678910 HTTP/1.1” 404
21816 “-” “Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1;
Trident/4.0)”
Reason: “Detects specific directory and path traversal”
85.95.238.173 – – [11/Feb/2012:00:50:50 -0600] “GET
/%80../%80../%80../%80../%80../%80../windows/win.ini HTTP/1.1” 404 21816
“-” “Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)”
Reason: “Detects basic directory traversal”
85.95.238.173 – – [11/Feb/2012:00:50:51 -0600] “GET
/%80../%80../%80../%80../%80../%80../winnt/win.ini HTTP/1.1” 404 21816
“-” “Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)”
Reason: “Detects basic directory traversal”
85.95.238.173 – – [11/Feb/2012:00:50:52 -0600] “GET
/%c0.%c0./%c0.%c0./%c0.%c0./%c0.%c0./%c0.%c0./windows/win.ini HTTP/1.1”
404 21816 “-” “Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1;
Trident/4.0)”
Reason: “Detects basic directory traversal”
```

…scalp log satırı yüksek olan log analizlerinde genellikle yüksek oranda false positive ürettiği için gerçek ortamlarda kullanılmayacak bir yazılımdır. Ortalama %10-%15 civarında false positive ürettiği göz önüne alınırsa log analizi yapan uzmanın logları analiz ettiği kadar sonuçları da analiz etmesini gerek kılacaktır.

Basit UNIX Araçlarıyla Log Analizi

Bu yöntemde önemli olan logların arasından ne tip özellikte olanlarını bulmak istediğimizi belirlemektir. Zira milyonlarca satır log arasında ne aradığını bilmeyen birinin samanlıkta iğne arayandan farkı kalmayacak ve zamanı boşa geçirecektir. Saldırı Olarak Değerlendirilebilecek Durumlar Saldırı yapılan sunucuya özel bazı dizin/dosyaların istenmesi. Mesela, WordPress gibi sistemlerde genellikle /wp-admin gibi dizinler ya da wp-login.php gibi dosyalara yönelik brute force denemeleri gerçekleştirilir.

Saldırı imzası olarak /wp-admin ve wp-login.php gibi kelimeleri arattırırsak saldırı yapanların bir kısmı belirlenmiş olunur. Sunucuya bağlantı kuran ip adresleri ve bağlantı sayıları Sunucu üzerinde deneme gerçekleştiren ip adreslerinin normalin üzerinde bağlantı sayısına sahip

olması beklenir. Aşağıdaki komutla Apache loglarında hangi ip adresi kaç adet bağlantı gerçekleştirmiş (top 10) ortaya çıkarılabilir.

```
# cat siber-access_log |awk -F " " '{print $1}'|sort -n|uniq -c|sort -nr|head
3556 9.6.2.2
1527 9.2.4.1
1142 1.1.2.8
1055 193.2.2.1
1046 9.1.2.1
```

Directory Traversal Denemelerini Bulma

Web üzerinden gerçekleştirilebilecek önemli saldırı yöntemlerinden birisi web üzerinden sistemdeki dosyaları okuma olarak tanımlayabileceğimiz LFI (Local File Inclusion) saldırılarıdır. Web üzerinden gerçekleştirilen LFI vs saldırılarını loglardan yakalamak için ara bileşen olarak kullanılan ../.. gibi özel ifadeleri aratmak yeterli olacaktır. Yine burada hatırlanması gereken önemli nokta bu karekterler GET isteği üzerinden taşındığı zaman web sunucu loglarında yer bulacaktır.

```
13.22.1.129 – – [01/Dec/2010:02:20:55 +0200] "GET
/imprimer.asp?no=/../../../../../../../../etc/passwd|44|80040e14|[Microsoft][ODBC_SQL_Server_Drive
r][SQL_Server]Line_1:_Incorrect_syntax_near_&#039;/&#039;.
HTTP/1.1" 404 210 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US;
rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3"
19.22.1.1 – – [01/Dec/2010:02:20:55 +0200] "GET
/mailview.cgi?cmd=view&amp;fldrname=inbox&amp;select=1&amp;html=../../../../../../etc/passwd
HTTP/1.1" 404 210 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US;
rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3"
13.22.1.19 – – [01/Dec/2010:02:20:55 +0200] "GET
/modif_infos.asp?n=../../../../../../../../../etc/passwd%00 HTTP/1.1"
404 213 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.1.3)
Gecko/20090824 Firefox/3.5.3"
13.22.1.19 – – [01/Dec/2010:02:20:55 +0200] "GET
/modif_infos.asp?n=/../../../../../../../../../../../../../../../../../../../../boot.ini
HTTP/1.1" 404 213 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US;
rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3"
13.22.1.19 – – [01/Dec/2010:02:20:55 +0200] "GET
/modif_infos.asp?n=/../../../../../../../../../etc/passwd HTTP/1.1" 404
213 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.1.3)
Gecko/20090824 Firefox/3.5.3"
13.22.1.19 – – [01/Dec/2010:02:20:56 +0200] "GET /pm/lib.inc.php
HTTP/1.1" 404 212 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US;
rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3"
13.22.1.19 – – [01/Dec/2010:02:20:56 +0200] "GET
/productcart/pc/Custva.asp?|-|0|404_Object_Not_Found HTTP/1.1" 404 223
"-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.1.3)
Gecko/20090824 Firefox/3.5.3"
13.22.1.19 – – [01/Dec/2010:02:20:56 +0200] "GET
/ProductCart/pc/msg.asp?|-|0|404_Object_Not_Found HTTP/1.1" 404 220 "-"
```

```
“Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.1.3)
Gecko/20090824 Firefox/3.5.3”
13.22.1.19 – – [01/Dec/2010:02:20:56 +0200] “GET
/rubrique.asp?no=../../../../../../../../../etc/passwd%00|55|80040e14|[Microsoft][ODBC_SQL_Server_
Driver][SQL_Server]Line_1:_Incorrect_syntax_near_&#039;/&#039;.
HTTP/1.1” 404 210 “-” “Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US;
rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3”
13.22.1.19 – – [01/Dec/2010:02:20:56 +0200] “GET
/rubrique.asp?no=/../../../../../../../../../../../../../../../../../../../../../../../boot.ini|55|80040e14|[Microsoft][OD
BC_SQL_Server_Driver][SQL_Server]Line_1:_Incorrect_syntax_near_&#039;/&#039;.
HTTP/1.1” 404 210 “-” “Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US;
rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3”
13.22.1.19 – – [01/Dec/2010:02:20:56 +0200] “GET
/rubrique.asp?no=/../../../../../../etc/passwd|55|80040e14|[Microsoft][ODBC_SQL_Server_Driver][SQ
L_Server]Line_1:_Incorrect_syntax_near_&#039;/&#039;.
HTTP/1.1” 404 210 “-” “Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US;
rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3”
13.22.1.19 – – [01/Dec/2010:02:20:56 +0200] “GET
/rubrique.asp?no=/../../../etc/passwd|55|80040e14|[Microsoft][ODBC_SQL_Server_Driver][SQL_Serv
er]Line_1:_Incorrect_syntax_near_&#039;/&#039;.
HTTP/1.1” 404 210 “-” “Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US;
rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3”
13.22.1.19 – – [01/Dec/2010:02:20:56 +0200] “GET
/shoutbox/expanded.php?conf=../../../../../../etc/passwd%20 HTTP/1.1”
404 219 “-” “Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US;
rv:1.9.1.3) Gecko/20090824 Firefox/3.5.3”
```

13.22.1.19 ip adresinin sisteme yönelik normal olmayan bir şeyler denediği ortadadır. Bu IP adresinin başka neler denediğine bakarak saldırı imzası olarak aranacak kelime grupları genişletilebilir. Burada önemli olan bu tip isteklere web sunucunun döndüğü cevaptır . Web sunucu 404 değil 300 veya 200 lü cevap dönüyorsa saldırının başarılı olmuş olma ihtimali vardır. 404 alınıyorsa bu saldırganın denediği atakların başarılı olmadığı, sunucu tarafında bulunmadığı anlamına gelir.

SQL Injection Denemelerini Yakalama

SQLi denemelerini web sunucu loglarından yakalamak için genellikle tercih edilen yöntem sqli için kullanılan kelimeler ve evasion amaçlı kullanılan özel karekterlerin web sunucu loglarından aratılmasıdır.

SQLi aratmak için Concat, char, union, select, order by, group by gibi komutlar denenebilir. Bu kelimeleri log dosyasında aratmak false positive sonuçlar çıkarabileceği için çıkan sonuçların teker teker incelenmesi gerekir. Burada yine sadece GET üzerinden denenen sqli sldırılarının loglarından anlamlı bir şeyler çıkacağını belirtmemiz gerekiyor.

Concat denemeleri:

```
127.0.0.1 – – [28/Feb/2012:05:03:40 -0500] “GET
/dvwa/vulnerabilities/sqli/?id=9&Submit=Submit%27%29%20AND%20%28SELECT
%204770%20FROM%28SELECT%20COUNT%28%2A%29%2CCONCAT
%28CHAR%2858%2C118%2C106%2C115%2C58%29%2C%28SELECT%20%28CASE%20WHEN
%20%284770%3D4770%29%20THEN%201%20ELSE%200%20END%29%29%2CCHAR%2858%2C101%2
C111%2C102%2C58%29
%2CFLOOR%28RAND%280%29%2A2%29%29x%20FROM%20
INFORMATION_SCHEMA.CHARACTER_SETS%20GROUP%20BY%20x%29a%29%20AND%20%28%27JrPI
%27%3D%27JrPI
HTTP/1.1” 200 4660 “-” “sqlmap/1.0-dev (r4009)
(http://sqlmap.sourceforge.net)”
127.0.0.1 – – [28/Feb/2012:05:03:40 -0500] “GET
/dvwa/vulnerabilities/sqli/?id=9&Submit=Submit%27%20AND%20%28SELECT%204770%20FROM%28
SELECT%20COUNT%28%2A%29%2CCONCAT%28
CHAR%2858%2C118%2C106%2C115%2C58%29%2C%28SELECT%20%28CASE%20WHEN%20%284770
%3D4770%29%20
THEN%201%20
ELSE%200%20END%29%29%2CCHAR%2858%2C101%2C111%2C102%2C58%29%2CFLOOR%28RAND%
280%29%2A2%29%29x%20
FROM%20INFORMATION_SCHEMA.CHARACTER_SETS%20GROUP%20BY%20x%29a%29%20AND%20%
27AwiQ%27%3D%27AwiQ
HTTP/1.1” 200 4660 “-” “sqlmap/1.0-dev (r4009)
(http://sqlmap.sourceforge.net)”
```

Not: İleri seviye sql injection denemelerini yakalamak için PHPIDS’deki düzenli ifadeler kullanılmalıdır.

Command Execution Denemeleri

Eğer saldırgan başarılı bir şekilde sisteme sızmayı başardıysa ilk işi Linux sistemde çalıştırılacak temel komutları id, whoami, wget, .etc.passwd vs denemek olacaktır. Bu komutları sistemde çalıştırarak saldırganın sisteme erişim sağlayıp sağlayamadığı belirlenebilir.

Ossec Kullanarak Pasif Log Analizi

Ossec, açık kaynak kodlu aktif log analiz ve alarm yazılımıdır. Daha önceden belirtilmiş log dosyalarını aktif olarak izleyerek belirli durumlar (şüpheli durumlar, saldırılar) oluştuğunda log üretip e-posta gönderebilen ve aksiyon alabilen bir yapıdadır. OSSEC aynı zamanda "Forensic" amaçlı da kullanılabilir. Sistem logları OSSEC'e girdi olarak verilebilir ve bir simülasyon gibi kendi kurallarından süzerek olası alarmları raporlar. Bunu bir örnek olarak açıklayabiliriz.

OSSEC, normalde canlı sistemde aktif olarak log analizi için kullanılan bir yazılım olmasına rağmen içerisindeki "ossec-logtest" yazılımı aslında bir logun kuralları tetikleyip tetiklemediğini test etmek için oluşturulmuştur.

"/var/log/secure" dosyasını ossec-logtest ile inceleyelim.

```
# cat /var/log/secure | /var/ossec/bin/ossec-logtest -a
```

Ossec-logtest -a parametresi ile alarmları oluşturmasını sağlar.

var/ossec/logs/alerts.log çıktısı:

```
** Alert 1264788284.11: – syslog,sshd,authentication_success,
2010 Jan 29 14:04:44 enigma->stdin
Rule: 5715 (level 3) -> 'SSHD authentication success.'
Src IP: a.b.2.15
User: dcid
Jan 15 10:25:01 enigma sshd[17594]: Accepted password for dcid from a.b.2.15 port 47526 ssh2

** Alert 1264788284.12: – syslog,sshd,authentication_success,
2010 Jan 29 14:04:44 enigma->stdin
Rule: 5715 (level 3) -> 'SSHD authentication success.'
Src IP: 127.0.0.1
User: dcid
Jan 15 11:19:20 enigma sshd[18853]: Accepted publickey for dcid from 127.0.0.1 port 6725 ssh2
```

Görüldüğü üzere ossec-logtest OSSEC'in içerisinde bulunan tüm kurallardan süzerek, alerts.log dosyasına alarmları bastı. Bu alarmları raporlanmış bir şekilde okumak için: "ossec-reported" aracı kullanılabilir.

```
cat /var/log/secure | /var/ossec/bin/ossec-logtest -a |/var/ossec/bin/ossec-reported
```

Ossec-reported çıktısı:

```
Report completed. ==
————————————————————
->Processed alerts: 522
->Post-filtering alerts: 522
Top entries for ‘Source ip’:
————————————————————
89.200.169.170 |41 |
127.0.0.1 |33 |
83.170.106.142 |20 |
204.232.206.109 |16 |
..
Top entries for ‘Username’:
————————————————————
root |247 |
Top entries for ‘Level’:
————————————————————
Severity 5 |406 |
Severity 3 |41 |
Severity 10 |32 |
Top entries for ‘Group’:
————————————————————
syslog |522 |
sshd |509 |
authentication_failed |369 |
invalid_login |146 |
Top entries for ‘Rule’:
————————————————————
5716 – SSHD authentication failed. |223 |
5710 – Attempt to login using a non-existent.. |146 |
5715 – SSHD authentication success. |41 |
5702 – Reverse lookup error (bad ISP or atta.. |37 |
```

Örneğin sadece Brute-force saldırılarını görmek istiyorsanız (authentication_failed):

```
$ cat /var/log/secure | /var/ossec/bin/ossec-logtest -a |/var/ossec/bin/ossec-reported -f group
authentication_failures
```

Linux Audit Loglarının İncelenmesi

Linux Audit sistemi kernel(çekirdek) tarafından sisteminizde oluşan çağrı ve olayların hemen hemen hepsi hakkında süreç işlemlerinin loglanmasıdır. Syslog/Rsyslog loglarından farklıdır; çünkü Audit logları bazı temel değişmeyen alt yapısı mevcuttur. Syslog logları üzerinde kullanıcı değişiklikler yapabilir. RHEL tabanlı dağıtımlarda Auditd sistem çekirdeğine gömülü olarak gelmektedir. Tüm dağıtımlarda durum bu şekilde değildir.

Auditd konfigürasyonları /etc/audit/auditd.conf dosyasında ve kurallar /etc/audit/rules.d/ dizininde bulunmaktadır. Audit logları /var/log/audit/audit.log dosyasında tutulmaktadır. Örnek bir Audit logu aşağıdaki gibi gözükmektedir.

```
type=USER_AUTH msg=audit(1364475353.159:24270): user pid=3280 uid=500 auid=500 ses=1
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 msg='op=PAM:authentication
acct="root" exe="/bin/su" hostname=? addr=? terminal=pts/0 res=failed'
```

Yukarıdaki log uid=500 olan bir kullanıcının root olmaya çalıştığını fakat başarısız olduğunu gösteren logdur.

Linux audit logları ilk bakışta okunması zor görünebilir; fakat audit araçları olan “ausearch” ve “aureport” kullanarak daha kolay okunabilir hale getirilebilir ve rapor alınabilir.

Ausearch Kullanarak Audit Loglarının Analizi

Örneğin tüm başarısız girişleri listelemek için şu şekilde bir komut kullanılır.

```
ausearch --message USER_LOGIN --success no --interpret
```

--message : aranılan mesaj, --success: işlemin başarılı/başarısız olması, --interpret: uid’leri kullanıcı hesabına çevirir.

Tüm kullanıcılar ve gruplar üzerinde rol değişiklikleri hakkında audit loglarını incelemek için:

```
ausearch -m ADD_USER -m DEL_USER -m ADD_GROUP -m USER_CHAUTHTOK -m DEL_GROUP -m
CHGRP_ID -m ROLE_ASSIGN -m ROLE_REMOVE -i
```

Örnek çıktı:

```
----
type=ADD_GROUP msg=audit(07/31/2017 08:07:29.378:497) : pid=30906 uid=root auid=testuser
ses=3 subj=unconfined_u:unconfined_r:groupadd_t:s0-s0:c0.c1023 msg='op=add-group id=vboxusers
exe=/usr/sbin/groupadd hostname=? addr=? terminal=? res=success'
----
```

Tek bir kullanıcıya ait tüm aksiyonların auditd logları incelemek için:

auid = 1000(kullanıcı login id'si)

```
ausearch -ua 1000 -i
```

-i = interpret (nümerikleri yazıya çevirir (uid to text))

```
type=CRED_DISP msg=audit(08/03/2017 11:41:59.636:345) : pid=4641 uid=root auid=tesuser ses=2
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 msg='op=PAM:setcred
grantors=pam_env,pam_fprintd acct=root exe=/usr/bin/sudo hostname=? addr=?
terminal=/dev/pts/0 res=success'
```

Aureport Kullanarak Audit Raporu Oluşturmak

Raw audit loglarını okumak zorlaşabilir, aureport kullanarak audit raporlarının bir özeti alınabilir. Aureport'u parametresiz olarak çalıştırmak kısaca bir özet verir fakat parametreler ile bu raporları detaylandırabiliriz.

```
$ aureport
```

Örnek Çıktı:

```
Summary Report
======================
Range of time in logs: 07/30/2017 23:41:25.380 - 08/03/2017 12:23:23.903
Selected time for report: 07/30/2017 23:41:25 - 08/03/2017 12:23:23.903
Number of changes in configuration: 603
Number of changes to accounts, groups, or roles: 17
Number of logins: 5
Number of failed logins: 33
Number of authentications: 39
Number of failed authentications: 34
Number of users: 4
Number of terminals: 8
Number of host names: 4
Number of executables: 15
Number of commands: 8
Number of files: 2
Number of AVC's: 2
Number of MAC events: 20
Number of failed syscalls: 0
Number of anomaly events: 17
Number of responses to anomaly events: 0
Number of crypto events: 0
Number of integrity events: 0
```

```
Number of virt events: 0
Number of keys: 1
Number of process IDs: 264
Number of events: 2452
```

Başarısız işlemlerin Audit Raporunu görmek için:

```
aureport --failed
```

Çalıştırılabilir dosyaların Audit Raporlarını görmek için:

```
aureport -x --summary
```

Sistem çağrıları ve kullanıcı isimleri ile erişilmiş dosyaların raporu için:

```
$aureport -f -i
```

Örnek Çıktı:

```
File Report
============================================
# date time file syscall success exe auid event
============================================
1. Monday 15 June 2015 08:27:51 /etc/ssh/sshd_config open yes /usr/bin/cat sammy 135496
2. Tuesday 16 June 2015 00:40:15 /etc/ssh/sshd_config getxattr no /usr/bin/ls root 147481
```

“open” , “getaxttr” sistem çağrılarıdır.

Autrace Kullanarak Process Analizi

Autrace bir process tarafından kullanılan sistem çağrılarını takip etmek için kullanılır. Bu araç sistemde eğer bir trojan veya problemli bir proses varsa tespit etmekte oldukça işe yarayabilir. Autrace çalıştırıldıktan sonra aureport ile görüntülenebilir çıktı vermektedir.

Uyarı: Autrace bazı sebeplerden ötürü özelleştirilmiş Audit kurallarını silmektedir!

Örneğin “date” komutuyla ilgili autrace sorgusu yapalım.

```
$ autrace /bin/date
```

```
Waiting to execute: /bin/date
Thu Aug  3 12:33:24 +03 2017
Cleaning up...
Trace complete. You can locate the records with 'ausearch -i -p 5474'
```

Ausearch kullanarak bu logları görebileceğimi söylüyor. Aureport’a göndererek raporlanmış sonuç alabiliriz.

```
ausearch  -p 5474 --raw | aureport -f -i
```

```
File Report
===============================================
# date time file syscall success exe auid event
===============================================
1. 08/03/2017 12:33:24 /bin/date execve yes /usr/bin/date testuser 381
2. 08/03/2017 12:33:24 /etc/ld.so.preload access no /usr/bin/date testuser 384
3. 08/03/2017 12:33:24 /etc/ld.so.cache open yes /usr/bin/date testuser 385
4. 08/03/2017 12:33:24 /lib64/libc.so.6 open yes /usr/bin/date testuser 389
5. 08/03/2017 12:33:24 /usr/lib/locale/locale-archive open yes /usr/bin/date testuser 406
6. 08/03/2017 12:33:24 /etc/localtime open yes /usr/bin/date tesuser 410
```

“Date” komutuna kimlerin eriştiği, hangi sistem çağrısını kullandığını, zamanını belirlemiş olduk. Olası bir zararlı yazılım proseslerini incelediğimiz de çok önemli bir veri olacaktır.

Bunların dışında Tüm audit kayıt tiplerine şuradan ulaşabilirsiniz:

https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Security_Guide/sec-Audit_Record_Types.html

Sistemde Arka Kapı Kontrolü

Web tabanlı arka kapılar fark edilmez, istenildiğinde port açmak içindir. Kullanılan web tabanlı arka kapıları tespit edebilmek için aşağıdaki yöntemler kullanılabilir. İşini iyi yapan, özelleştirilmiş yazılımlar kullanıldığı durumlarda tespit etmek oldukça zor olacaktır. [4]

Linux Üzerinde Zararlı Yazılım Tespiti

Rootkit Nedir?

Rootkit, adından da anlaşılacağı gibi iki parçadan oluşmaktadır. "Root" unix/linux sistemlerde en yetkili kullanıcı iznine sahip olan kullanıcıdır, "Kit" ise bu yetki sahibi olabilmek için kullanılan araç kutusu şeklinde ifade edilebilir. Rootkit'ler her yerden buluşabilir. Kaynak koddan derlediğiniz ve nereden geldiğini bilmediğiniz programlardan bulaşma imkanı yüksektir. En temiz yolu kullanacağınız programı kendi web adreslerinden indirmek ve kullanmaktır. Rootkit özel olarak hazırlanmış zararlı bir yazılımdır. Rootkit'ler virüsler gibi bilgisayarlara zarar vermek için sızmazlar yaptıkları sisteme sızıp bilgisayarı tam yetki ile kontrol etmektir.

Rootkit'in gerçekte hangi dosyaları değiştirdiği, kernel'a hangi modülü yüklediği, dosya sisteminin neresinde kayıtlı olduğu, hangi ağ servisi üzerinden dinleme yaparak uygun komutla harekete geçeceğini tespit etmek güçtür. Yine de, belli zamanlarda en temel komutların ve muhtemel rootkit bulaşma noktalarının öz değerlerinin saklanarak bunların daha sonra kontrol edilmesi vb. metotlar kullanılabilir.

Linux dünyasında rootkit yazılımlarının sistemdeki varlığını kontrol eden rootkit tespit yazılımları bulunmaktadır. Bunlar Chkrootkit ve Rkhunter yazılımlar bu işi yapan araçlardan en popülerleri arasındadır.

Bu yazılımların kurumları oldukça basittir. Aşağıdaki size uygun komut ile iki yazılım arasından birini sisteme kurabilirsiniz.

```
[Rhel/Centos ve Fedora]
root# yum install rkhunter
root# yum install chkrootkit

[Ubuntu/Linux ve Debian]
root# apt-get install rkhunter
root# apt-get install chkrootkit
```

Rkhunter ve Chkrootkit Kullanımı

Rootkit tespit araçlarının kullanımı oldukça basittir. Aşağıdaki komutları çalıştırdıktan sonra sisteminizi tarayacaktır ve sonuçları ekrana yazacaktır. Bu iki araç ellerinde önceden tespit ettikleri rootkit 'leri tanımaktadır ve onlar sisteme bulaştılarsa bulabilirler. Yeni yazılan ve henüz kimse tarafından bulunmayan bir rootkit sisteme bulaştıysa bunu bulamazlar. Böyle bir durumda sizin elle kontrol etmeniz gerekir.

Chkrootkit aracının çalıştırılması için aşağıdaki komut kullanılabilir.

```
root# chkrootkit

ROOTDIR is `/'
Checking `amd'...                    not found
Checking `basename'...               not infected
Checking `biff'...                   not found
Checking `chfn'...                   not infected
Checking `chsh'...                   not infected
Checking `cron'...                   not infected
Checking `crontab'...                not infected
Checking `date'...                   not infected
Checking `du'...                     not infected
Checking `dirname'...                not infected
Checking `echo'...                   not infected
Checking `egrep'...                  not infected
Checking `hdparm'...                 not found
Checking `su'...                     not infected
Checking `ifconfig'...               not infected
Checking `inetd'...                  not infected
Checking `named'...                  not found
Checking `passwd'...                 not infected
Checking `pidof'...                  not infected
Checking `pop2'...                   not found
Checking `pop3'...                   not found
Checking `ps'...                     not infected
```

Rkhunter aracının çalıştırılması için aşağıdaki komut kullanılabilir.

```
root# rkhunter --update

root# rkhunter --check

Checking for rootkits...

  Performing check of known rootkit files and directories

    55808 Trojan - Variant A                [ Not found ]
    ADM Worm                                [ Not found ]
    AjaKit Rootkit                          [ Not found ]
    Adore Rootkit                           [ Not found ]
    aPa Kit                                 [ Not found ]
    Apache Worm                             [ Not found ]
    Ambient (ark) Rootkit                   [ Not found ]
    Balaur Rootkit                          [ Not found ]
    BeastKit Rootkit                        [ Not found ]
    Dreams Rootkit                          [ Not found ]
    Duarawkz Rootkit                        [ Not found ]
    Enye LKM                                [ Not found ]
    Li0n Worm                               [ Not found ]
    Lockit / LJK2 Rootkit                   [ Not found ]
    Mood-NT Rootkit                         [ Not found ]
    MRK Rootkit                             [ Not found ]
    Ni0 Rootkit                             [ Not found ]
    Ohhara Rootkit                          [ Not found ]

Checking the network...

  Performing checks on the network ports
    Checking for backdoor ports             [ None found ]
    Checking for hidden ports               [ None found ]

  Performing checks on the network interfaces
    Checking for promiscuous interfaces     [ None found ]

……..
```

Rkhunter ve Chkrootkit 'in Test Edilmesi

Bu aşamada azazel adlı zararlı yazılımının bulaşmış olduğu bir Linux sunucusunu "rkhunter" ve "chrootkit" araçları ile tarayıp/test edip sonuçlarına bakıyor olacağız.

Azazel test ortamına kurulmuştur. Kurulumdan sonra "chkrootkit" ve "rkhunter" araçları çalıştırılıp tespit edilmeye çalışılmıştır. Azazel adlı tehlikeli Rootkiti kullandığınız bilgisayarda çalıştırmayınız!

Sonuç: İki rootkit tespit aracı (chkrootkot-rkhunter) azazel adlı virüsü tespit edememiştir. [2]

Bellek Dökümü Alma ve Bellek Analizi

Bir Linux sistemin bellek analizinin yapılabilmesi için bellek dökümü/imajı alınmalıdır. Linux sistemlerde bellek dökümü almak için fmem, LIME(Linux Memory Extractor), memdump gibi uygulamalar kullanılmaktadır. Aşağıda fmem kullanılarak Ubuntu14.04.5 sürümüne ait bellek dökümü alınmıştır.

```
root # wget http://hysteria.cz/niekt0/fmem/fmem_current.tgz
--2017-08-07 06:51:53--  http://hysteria.cz/niekt0/fmem/fmem_current.tgz
Resolving hysteria.cz (hysteria.cz)... 77.78.111.10, 2001:1528:162:20::10
Connecting to hysteria.cz (hysteria.cz)|77.78.111.10|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 12566 (12K) [application/x-gzip]
Saving to: 'fmem_current.tgz'

100%[==================================================>] 12.566      856B/s   in 15s

2017-08-07 06:52:08 (856 B/s) - 'fmem_current.tgz' saved [12566/12566]

root # tar -zxvf fmem_current.tgz
fmem_1.6-0/
fmem_1.6-0/debug.h
fmem_1.6-0/README
fmem_1.6-0/ChangeLog
fmem_1.6-0/COPYING
fmem_1.6-0/AUTHORS
fmem_1.6-0/TODO
fmem_1.6-0/run.sh
fmem_1.6-0/lkm.c
fmem_1.6-0/Makefile

root # cd fmem_1.6-0/

root # make
rm -f *.o *.ko *.mod.c Module.symvers Module.markers modules.order \.*.o.cmd \.*.ko.cmd \.*.o.d
rm -rf \.tmp_versions
make -C /lib/modules/`uname -r`/build SUBDIRS=`pwd` modules
```

```
make[1]: Entering directory `/usr/src/linux-headers-3.13.0-24-generic'
  CC [M]  /root/fmem_1.6-0/lkm.o
  LD [M]  /root/fmem_1.6-0/fmem.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /root/fmem_1.6-0/fmem.mod.o
  LD [M]  /root/fmem_1.6-0/fmem.ko
make[1]: Leaving directory `/usr/src/linux-headers-3.13.0-24-generic'

root # ./run.sh
Module: insmod fmem.ko a1=0xc105c9b0 : OK
Device: /dev/fmem
----Memory areas: -----
reg00: base=0x000000000 (    0MB), size= 2048MB, count=1: write-back
reg01: base=0x080000000 ( 2048MB), size= 1024MB, count=1: write-back
reg02: base=0x100000000 ( 4096MB), size= 4096MB, count=1: write-back
reg03: base=0x200000000 ( 8192MB), size= 8192MB, count=1: write-back
reg04: base=0x400000000 (16384MB), size=16384MB, count=1: write-back
reg05: base=0x800000000 (32768MB), size=32768MB, count=1: write-back
----------------------
!!! Don't forget add "count=" to dd !!!

root # dd if=/dev/fmem of=/media/root/USBDisk/Ubuntu14045Image bs=10MB count=2048
214+0 records in
214+0 records out
2140000000 bytes (2,1 GB) copied, 436,367 s, 4,9 MB/s
```

Yukarıdaki işlemlerde öncelikle *fmem* uygulaması ilgili bağlantıdan indirilmiştir. Sonrasında arşivden çıkarılıp sisteme dahil edilmiştir. Burada sisteme dahil edilen *fmem.ko* isimli dosya kernel(çekirdek) modülüdür ve bellek dökümü almayı sağlamaktadır. Son olaral *dd* komutu kullanılarak bellek dökümü harici bir belleğe aktarılmıştır.

Bu noktadan sonra alınan bellek dökümünün herhangi bir analiz uygulaması ile incelemesi yapılmalıdır. Analiz işlemi için de en çok kullanılan uygulamalardan *Volatility Framework* tercih edilecektir. Analiz işleminde Volatility'den daha verimli sonuçlar alabilmek için bellek dökümü alınan sistemin profil dosyası oluşturmak gerekir. Profil dosyası ilgili kernel(çekirdek) sürümüne ait map dosyası ve veri yapıları dosyasının arşiv halidir. Profil dosyasını oluşturmak için sisteme indirilen Volatility için **tools/linux** dizininde *make* komutunu çalıştırmak yeterlidir.

Eğer Volatility, sisteme paket yöneticisi ile kurulmuş ise bu durumda **/usr/share/volatility/tools/linux** dizini altında *make* komutunu çalıştırmak gerekir.

```
root:/usr/share/volatility/tools/linux# make
make -C //lib/modules/3.13.0-24-generic/build CONFIG_DEBUG_INFO=y
M="/usr/share/volatility/tools/linux" modules
make[1]: Entering directory '/usr/src/linux-headers-3.13.0-24-generic'
make[1]: Entering directory `/usr/src/linux-headers-3.13.0-24-generic'
  Building modules, stage 2.
```

```
  MODPOST 1 modules
make[1]: Leaving directory '/usr/src/linux-headers-3.13.0-24-generic'
dwarfdump -di module.ko > module.dwarf
make -C //lib/modules/3.13.0-24-generic/build M="/usr/share/volatility/tools/linux" clean
make[1]: Entering directory '/usr/src/linux-headers-3.13.0-24-generic'
make[1]: Entering directory `/usr/src/linux-headers-3.13.0-24-generic'
  CLEAN   /usr/share/volatility/tools/linux/.tmp_versions
  CLEAN   /usr/share/volatility/tools/linux/Module.symvers
make[1]: Leaving directory '/usr/src/linux-headers-3.13.0-24-generic'
```

Bu komut çalıştıktan sonra aynı dizinde **module.dwarf** isimli bir dosya oluşacaktır. Bu dosya ile birlikte */boot* dizini altında bulunan kernel(çekirdek) versiyonuna ait map dosyası gereklidir. Map dosyası **System.map-3.13.0-24-generic** gibidir. Burada sistemde kullanılan kernel versiyonuna ait dosyanın seçilmesi önemlidir. Volatility profili oluşturmak için de bu iki dosya zip dosyası haline getirilerek bellek dökümünün analizinin yapılacağı sistemde kurulu olan Volatility'nin **/usr/lib/python2.7/dist-packages/volatility/plugins/overlays/linux** dizini altına koyulmalıdır.

```
root # zip /root/Ubuntu14045.zip /usr/share/volatility/tools/linux/module.dwarf /boot/System.map-3.13.0-24-generic
```

Profil dosyası Volatility'nin ilgili dizinine yerleştirildikten sonra aşağıdaki komut çıktısında profil bölümünde görüldüğünden emin olunabilir.

```
root # volatility --info
```

Profil dosyasının varlığından emin olduktan sonra bellek dökümü üzerinden bilgiler toplanmaya başlanır. Örneğin prosesler ile ilgili bilgilerin karşılaştırmalı halde bilgilerini alalım.

```
root # volatility -f Ubuntu14045Image --profile=LinuxUbuntu14045x86 linux_psxview
Volatility Foundation Volatility Framework 2.6
Offset(V)  Name                    PID pslist psscan pid_hash kmem_cache parents leaders
---------- -------------------- ------ ------ ------ -------- ---------- ------- -------
INFO    : volatility.debug    : SLUB is currently unsupported.
0x32fe8cf0 console-kit-dae       1611 True   True   True     False      False   True
0x2e6e19e0 gvfsd-fuse            1831 True   True   True     False      False   True
0x2cca0000 rtkit-daemon          1942 True   True   True     False      False   True
0x35ba19e0 firefox               3059 True   True   True     False      False   True
0x34684da0 ksmd                    27 True   True   True     False      False   True
0x33958000 at-spi-bus-laun       2114 True   True   True     False      True    True
0x35d799e0 scsi_eh_4              116 True   True   True     False      False   True
0x2e7c26d0 mate-settings-d       1920 True   True   True     False      False   True
0x35ba6780 scsi_eh_19             150 True   True   True     False      False   True
0x353a99e0 gvfs-gphoto2-vo       2010 True   True   True     False      False   True
0x2cd44da0 upowerd               2048 True   True   True     False      False   True
0x353a8cf0 udisksd               1986 True   True   True     False      False   True
```

```
0x339a0cf0 system-tools-ba      2259 True   True   True    False     False   True
0x32d633c0 dbus-daemon           540 True   True   True    False     False   True
0x35d3c0b0 scsi_eh_26            163 True   True   True    False     False   True
0x32f4cda0 NetworkManager        861 True   True   True    False     True    True
...
```

Ağ bağlantılarını tespit etmek için de aşağıdaki gibi bir komut kullanılır.

```
root # volatility -f Ubuntu14045Image --profile=LinuxUbuntu14045x86 linux_netscan
Volatility Foundation Volatility Framework 2.6
e009c580 UDP     192.168.107.142 :35980 52.86.243.173   :    0
e009cdc0 UDP     192.168.107.142 :42397 91.189.88.162   :   80
e009db80 UDP     127.0.0.1       :46857 127.0.1.1       :   53
e009de40 UDP     0.0.0.0         :49923 0.0.0.0         :    0
e009e100 UDP     127.0.0.1       :60875 127.0.1.1       :   53
e009ec00 UDP     127.0.0.1       :42517 127.0.1.1       :   53
e009eec0 UDP     0.0.0.0         :16819 0.0.0.0         :    0
e009f180 UDP     192.168.107.142 :37665 68.235.39.11    :   80
e0100000 TCP     192.168.107.142 :45692 52.0.216.165    :  443 CLOSE
e0100540 TCP     192.168.107.142 :39593 52.7.34.147     :  443 CLOSE
e0100a80 TCP     192.168.107.142 :39595 52.7.34.147     :  443 CLOSE
e0100fc0 TCP     192.168.107.142 :39596 52.7.34.147     :  443 CLOSE
e0101500 TCP     192.168.107.142 :39597 52.7.34.147     :  443 CLOSE
e0101a40 TCP     192.168.107.142 :45696 52.0.216.165    :  443 CLOSE
e0101f80 TCP     192.168.107.142 :55495 52.72.241.171   :  443 CLOSE
e01024c0 TCP     192.168.107.142 :55496 52.72.241.171   :  443 CLOSE
e019b480 TCP     192.168.107.142 :60502 52.25.203.53    :  443 CLOSE
e019b9c0 TCP     192.168.107.142 :50277 192.168.107.140 : 4444 ESTABLISHED
e019bf00 TCP     192.168.107.142 :42317 208.77.20.11    :   80 CLOSE
e019c440 TCP     192.168.107.142 :58289 52.86.243.173   :  443 CLOSE
f2acc380 TCP     192.168.107.142 :46948 91.189.91.26    :   80 CLOSE
f2acc8c0 TCP     192.168.107.142 :58287 52.86.243.173   :  443 CLOSE
f2e7c000 UDP     ::              : 5353 ::              :    0
f2e7c300 UDP     ::              :60155 ::              :    0
f2e7c600 UDP     ::              :36701 ::              :    0
f2e7c900 UDP     ::ffff192.168.107.142:42793 ::ffff91.189.88.162:   80
f2ec8000 TCP     0.0.0.0         :  445 0.0.0.0         :    0 LISTEN
f2ec8540 TCP     0.0.0.0         :  139 0.0.0.0         :    0 LISTEN
f2ec8a80 TCP     127.0.1.1       :   53 0.0.0.0         :    0 LISTEN
f2ec8fc0 TCP     0.0.0.0         :   22 0.0.0.0         :    0 LISTEN
f2ec9500 TCP     127.0.0.1       :  631 0.0.0.0         :    0 LISTEN
f2ec9a40 TCP     192.168.107.142 :47719 140.90.101.207  :  443 CLOSE
f2ec9f80 TCP     192.168.107.142 :43859 195.175.113.51  :   80 CLOSE
f2eca4c0 TCP     192.168.107.142 :41837 34.206.14.85    :  443 CLOSE
f2ecaa00 TCP     192.168.107.142 :38950 93.184.220.29   :   80 CLOSE
f2ecaf40 TCP     192.168.107.142 :56564 52.222.157.157  :  443 CLOSE
...
```

Linux sistemden alınmış bir bellek dökümü üzerinde yapılabilecek tüm işlemleri görebilmek için aşağıdaki komutu çalıştırarak gerekli parametre ve açıklaması görülebilir.

```
root # volatility --info | grep linux
Volatility Foundation Volatility Framework 2.6
linux_apihooks          - Checks for userland apihooks
linux_arp               - Print the ARP table
linux_aslr_shift        - Automatically detect the Linux ASLR shift
linux_banner            - Prints the Linux banner information
linux_bash              - Recover bash history from bash process memory
linux_bash_env          - Recover a process' dynamic environment variables
linux_bash_hash         - Recover bash hash table from bash process memory
linux_check_afinfo      - Verifies the operation function pointers of network protocols
linux_check_creds       - Checks if any processes are sharing credential structures
linux_check_evt_arm     - Checks the Exception Vector Table to look for syscall table hooking
linux_check_fop         - Check file operation structures for rootkit modifications
linux_check_idt         - Checks if the IDT has been altered
linux_check_inline_kernel  - Check for inline kernel hooks
linux_check_modules     - Compares module list to sysfs info, if available
linux_check_syscall     - Checks if the system call table has been altered
linux_check_syscall_arm    - Checks if the system call table has been altered
linux_check_tty         - Checks tty devices for hooks
linux_cpuinfo           - Prints info about each active processor
linux_dentry_cache      - Gather files from the dentry cache
linux_dmesg             - Gather dmesg buffer
linux_dump_map          - Writes selected memory mappings to disk
linux_dynamic_env       - Recover a process' dynamic environment variables
linux_elfs              - Find ELF binaries in process mappings
linux_enumerate_files   - Lists files referenced by the filesystem cache
linux_find_file         - Lists and recovers files from memory
linux_getcwd            - Lists current working directory of each process
linux_hidden_modules    - Carves memory to find hidden kernel modules
linux_ifconfig          - Gathers active interfaces
linux_info_regs         - It's like 'info registers' in GDB. It prints out all the
linux_iomem             - Provides output similar to /proc/iomem
linux_kernel_opened_files  - Lists files that are opened from within the kernel
linux_keyboard_notifiers   - Parses the keyboard notifier call chain
linux_ldrmodules        - Compares the output of proc maps with the list of libraries from libdl
linux_library_list      - Lists libraries loaded into a process
linux_librarydump       - Dumps shared libraries in process memory to disk
linux_list_raw          - List applications with promiscuous sockets
linux_lsmod             - Gather loaded kernel modules
linux_lsof              - Lists file descriptors and their path
linux_malfind           - Looks for suspicious process mappings
linux_memmap            - Dumps the memory map for linux tasks
linux_moddump           - Extract loaded kernel modules
linux_mount             - Gather mounted fs/devices
linux_mount_cache       - Gather mounted fs/devices from kmem_cache
linux_netfilter         - Lists Netfilter hooks
linux_netscan           - Carves for network connection structures
```

```
linux_netstat            - Lists open sockets
linux_pidhashtable       - Enumerates processes through the PID hash table
linux_pkt_queues         - Writes per-process packet queues out to disk
linux_plthook            - Scan ELF binaries' PLT for hooks to non-NEEDED images
linux_proc_maps          - Gathers process memory maps
linux_proc_maps_rb       - Gathers process maps for linux through the mappings red-black tree
linux_procdump           - Dumps a process's executable image to disk
linux_process_hollow     - Checks for signs of process hollowing
linux_psaux              - Gathers processes along with full command line and start time
linux_psenv              - Gathers processes along with their static environment variables
linux_pslist             - Gather active tasks by walking the task_struct->task list
linux_pslist_cache       - Gather tasks from the kmem_cache
linux_psscan             - Scan physical memory for processes
linux_pstree             - Shows the parent/child relationship between processes
linux_psxview            - Find hidden processes with various process listings
linux_recover_filesystem - Recovers the entire cached file system from memory
linux_route_cache        - Recovers the routing cache from memory
linux_sk_buff_cache      - Recovers packets from the sk_buff kmem_cache
linux_slabinfo           - Mimics /proc/slabinfo on a running machine
linux_strings            - Match physical offsets to virtual addresses (may take a while, VERY verbose)
linux_threads            - Prints threads of processes
linux_tmpfs              - Recovers tmpfs filesystems from memory
linux_truecrypt_passphrase - Recovers cached Truecrypt passphrases
linux_vma_cache          - Gather VMAs from the vm_area_struct cache
linux_volshell           - Shell in the memory image
linux_yarascan           - A shell in the Linux memory image
```

Elde edilmek istenen verilere uygun şekilde yukarıdaki ilgili parametreler kullanılarak bilgiler bellek dökümü üzerinden elde edilir.

Linux Sistemlerde Ağ Trafiği Analizi Ağ Trafiği İnceleme

DDOS İhtimaline Karşı İnceleme

DOS saldırılarında temel amaç hedefi işlevsiz kılmaktadır. Bunun için yapılacak iki şey vardır. Hedef sistemin sahip olduğu Bandwith miktarından daha fazla trafik göndermek ya da hedef sistemin kaynaklarını kullanıcıların kullanamayacağı şekilde sömürmek.

DDOS saldırıları yapılış şekillerine göre birçok çeşiti olmasına rağmen günümüzde en çok SYNFlood yöntemidir. Amaç hedef sisteme kapasitesinden fazla SYN bayraklı TCP paket gönderilip sistem kaynaklarını hizmet veremez hale getirmektir.

Saldırının asıl kaynağı IPv4 protokolünden kaynaklanmaktadır. İstemcilerin herhangi bir doğrulama (bağlantı yapanın gerçekten ilgili IP adresine sahip olduğu bilgisi) mekanizmasından geçmemesidir.

Linux sistemlerde SYNFlood saldırısını tespit etmek

SynFlood saldırısı basitçe açık bir port hedef sistemin kapasitesinden fazla gönderilecek SYN paketleriyle gerçekleştirilir. Buradaki kapasite tanımı teknik olarak "Backlog Queue" olarak bilinmektedir.

Backlog Queue = İşletim sistemi aldığı her SYN paketine karşılık üçlü el sıkışmanın tamamlanacağı ana kadar bellekten bir alan kullanır. Half-open olarak tanımlayabileceğimiz bu bağlantılar "Backlog queue" tarafından belirlenir. Bu alanın dolması yeni gelen bağlantıların reddedilmesine sebep olur.

Netstat aracını kullanarak bağlantı durumlarına bakarak olası bir Synflood olup olmadığını anlayabiliriz.

En fazla bağlantı kurulan IP adresleri

```
$ netstat -ntu | awk '{print $5}' | cut -d: -f1 | sort | uniq -c | sort -n
```

Bağlantı durumlarını gösterir

```
$ netstat -nat | awk '{print $6}' | sort | uniq -c | sort -n
```

SYN_RECV durumundaki bağlantılarını görmek için

```
netstat -n | grep :80 | grep SYN |wc –l
```

“SYN_RECV” durumundaki bağlantıların belirli bir sayının üzerinde olması olası bir DDOS saldırısının göstergesidir.

SynFlood Saldırısı Sırasında Yapılacaklar

İlk olarak gelen ve giden tüm paketleri incelemek için bir port mirroring yapılarak kayıt altına alınması gerekmektedir. Bunun için tcpdump aracını kullanabiliriz.

```
#tcpdump -ttttnn -s0 tcp -w SYNFLOOD.pcap
```

Tcpdump ile sadece SYN bayraklı TCP paketlerini görmek için:

```
$ tcpdump 'tcp[13] & 2 != 0' -i eth0 -nnn
```

MITM İhtimaline Karşı İnceleme

MITM - Ortadaki Adam Saldırısı- yapılarak Linux sistemden internete yönelik giden ve dönen trafik içindeki hassas bilgiler elde edilerek sisteme erişim sağlanabilir. Yaşanmış bir olaydan örnek verilecek olursa MITM saldırılarının nasıl kullanılacağı daha iyi anlaşılmış olur.

Linux Sistem üzerinde yüklü olan "Wordpress" sistemine erişim sağlarken MITM yapılarak Wordpress erişim bilgileri elde edilmiştir. Bu bilgiler kullanılarak Wordpress'e giriş yapılmış ve editor özelliği kullanılarak sisteme php tabanlı bir arka kapı bırakılmıştır. Ardından bu arka kapı kullanılarak sistemde yetki yükseltme yapılarak Linux sistem root yetkileri ile ele geçirilmiştir.

Arp -a komutu kullanarak hedef sistemin o anlık "Gateway" bilgisi elde edilebilir. Ek olarak "dmesg" komutu kullanarak değişmiş olan aygıt durumları kontrol edilmelidir.

Bunun için mümkünse sisteme bir usb bellek takıp tüm çalışmaları onun üzerineden halletmek gerekir. İdeal durumda dosyaların değişikliğine karşı önlem alınmak istense de çoğunlukla hedef sisteme hangi yollarla girildiği, sistemden nelerin alındığı (veritabanı, önemli dosyalar vs.) ve sisteme sonradan erişim için kullanılabilecek arka kapıların tespiti beklenmektedir. Bunun için sistem üzerinde açık port/servisler belirlenmelidir.

"netstat -anlp" komutu kullanarak açık portlar ve bu portları dinleyen servislerin isimleri elde edilebilir. Buradan yola çıkarak sistemin SSH veya Web kullanılarak ele geçirildiği konusunda şüphelenebiliriz.

Bir başka yöntem de sisteme müdahale edildikten sonra ilk iş olarak son bir iki günde değişmiş dosyaların belirlenmesi çalışmasıdır. Saldırgan sisteme girdikten sonra veya sisteme girmek icin dosya yapısında mutlaka bir şeyleri kullanmış, eklemiş olmalıdır. Özellikle web tabanlı saldırılarda hacker mutlaka sisteme erişim sağladıktan sonra arka kapı bırakmaktadır.

Aktif olarak hackerın sistem üzerinde olduğu düşünülüyorsa yapılması gerekecek aktivitelerden ilgili tcpdump kullanarak tüm paketleri kaydetmek. Ardından bu paketleri kullanarak saldırgana ait tüm detaylar -eğer şifreli iletişim kanalı kullanılmamışsa- ortaya çıkartılabilir. Aktif Ağ Bağlantılarının İncelenmesi yukarıdaki başlıklarımızda anlatılmıştır.
Aktif olan bağlantılarına kontrol ettikten sonra ek olarak uzaktan "Nmap" yazılımı kullanarak

da sistem üzerindeki hizmet veren portlar tespit edilerek bir önceki işlem doğrulanmalıdır.

Tcpdump, ngrep ve buradaki anormal trafik SecurityOnion ortamı Suricata/Snort, BRO IDS gibi yazılımlarla ve Virustotal gibi sitelere incelenmesi için upload edilebilir.

Sisteme başarılı sızma gerçekleştiren saldırgan klasik yöntemlerle uzaktan tekrar erişim denemesi yerine "icmpshell" gibi ileri seviye araç ve yöntemler tercih etmişse netstat gibi sadece udp/tcp bağlantıları inceleyen araçlar yetersiz kalacaktır. Bu durumda sistemi tcpdump ile paket kaydı yapıp bir müddet sonra Wireshark ve benzeri programlarla incelemek yerinde olacaktır.

Kaynaklar

[1] http://en.wikipedia.org/wiki/Usage_share_of_operating_systems
[2] Kaynak : http://dfir.org/research/omfw.pdf , https://www.sans.org/reading-room/whitepapers/linux/linux-rootkits-beginners-prevention-removal-901
[3] https://askubuntu.com/questions/198501/list-of-all-shared-folders
[4] http://www.garage4hackers.com/showthread.php?t=987
http://resources.infosecinstitute.com/checking-out-backdoor-shells/

BGA Bilgi Güvenliği A.Ş. Hakkında

BGA Bilgi Güvenliği A.Ş. 2008 yılından bu yana siber güvenlik alanında faaliyet göstermektedir. Ülkemizdeki bilgi güvenliği sektörüne profesyonel anlamda destek olmak amacı ile kurulan BGA Bilgi Güvenliği, stratejik siber güvenlik danışmanlığı ve güvenlik eğitimleri konularında kurumlara hizmet vermektedir.

Uluslararası geçerliliğe sahip sertifikalı 50 kişilik teknik ekibi ile, faaliyetlerini Ankara ve İstanbul ve USA'da sürdüren BGA Bilgi Güvenliği'nin ilgi alanlarını "*Sızma Testleri, Güvenlik Denetimi, SOME, SOC Danışmanlığı, Açık Kaynak Siber Güvenlik Çözümleri, Büyük Veri Güvenlik Analizi ve Yeni Nesil Güvenlik Çözümleri*" oluşturmaktadır.

Gerçekleştirdiği başarılı danışmanlık projeleri ve eğitimlerle sektörde saygın bir yer edinen BGA Bilgi Güvenliği, kurulduğu günden bugüne alanında lider finans, enerji, telekom ve kamu kuruluşlarına 1.000'den fazla eğitim ve danışmanlık projeleri gerçekleştirmiştir.

BGA Bilgi Güvenliği, kurulduğu 2008 yılından beri ülkemizde bilgi güvenliği konusundaki bilgi ve paylaşımların artması amacı ile güvenlik e-posta listeleri oluşturulması, seminerler, güvenlik etkinlikleri düzenlenmesi, üniversite öğrencilerine kariyer ve bilgi sağlamak için siber güvenlik kampları düzenlenmesi ve sosyal sorumluluk projeleri gibi birçok konuda gönüllü faaliyetlerde bulunmuştur.

BGA Bilgi Güvenliği AKADEMİSİ Hakkında

BGA Bilgi Güvenliği A.Ş.'nin eğitim ve sosyal sorumluluk markası olarak çalışan Bilgi Güvenliği AKADEMİSİ, siber güvenlik konusunda ticari, gönüllü eğitimlerin düzenlenmesi ve siber güvenlik farkındalığını arttırıcı gönüllü faaliyetleri yürütülmesinden sorumludur. Bilgi Güvenliği AKADEMİSİ markasıyla bugüne kadar "Siber Güvenlik Kampları", "Siber Güvenlik Staj Okulu", "Siber Güvenlik Ar-Ge Destek Bursu", "Ethical Hacking yarışmaları" ve "Siber Güvenlik Kütüphanesi" gibi birçok gönüllü faaliyetin destekleyici olmuştur.