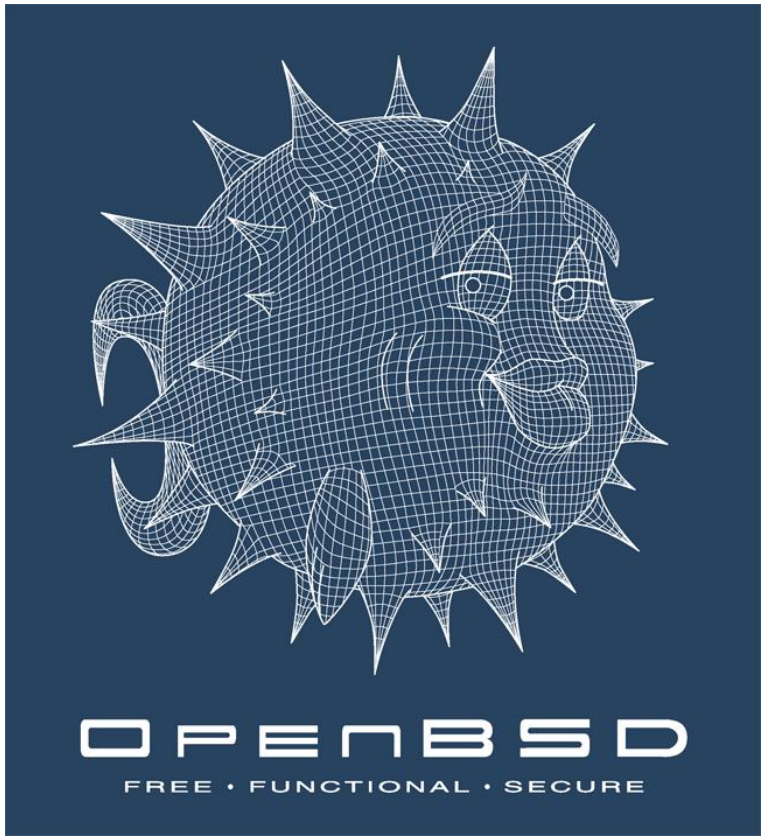
OpenBSD Packet Filter DDoS Koruma Özellikleri



OpenBSD ?

- *BSD ailesinin güvenliğe önem veren asi üyesi
- Alternatiflerine göre:
  - Daha zordur
  - Daha basitdir
  - Daha temizdir(dökümantasyon, kod vs)
  - Daha kanaatkardır(güvenliğe uymayan ve lisansı kabul edilemez yazılımar yoktur)
  - Daha güvenlidir!
- Firewall, IPS, VPN, Load Balancer, DDoS Engelleme amaçlı rahatlıkla kullanılabilir

OpenBSD Packet Filter

- *BSD dünyasının de fakto güvenlik duvarı yazılımı
- Piyasadaki tüm ticari-açık kod güvenlik duvarlarının teknik kabiliyetlerinin üzerindedir
  - 100Mb hat %3 CPU kullanımı (100.000 session)
- Kullanımı UNIX/Linux sistemlerin tersine çok kolaydır
  - İngilizce yazar gibi kural yazma kolaylığı
  - Pass in on $ext_if proto tcp from 1.1.1.1 to 1.1.1.2 port 80

Packet Filter Firewall Özellikleri

- IP başına session başına limit koyma özelliği
- SYN Proxy özelliği
  - TCP authentication özelliği
- HA(Yüksek bulunurluk) özelliği
- Anormal paketleri (port tarama, işletim sistemi saptama, traceroute vs) engelleme özelliği

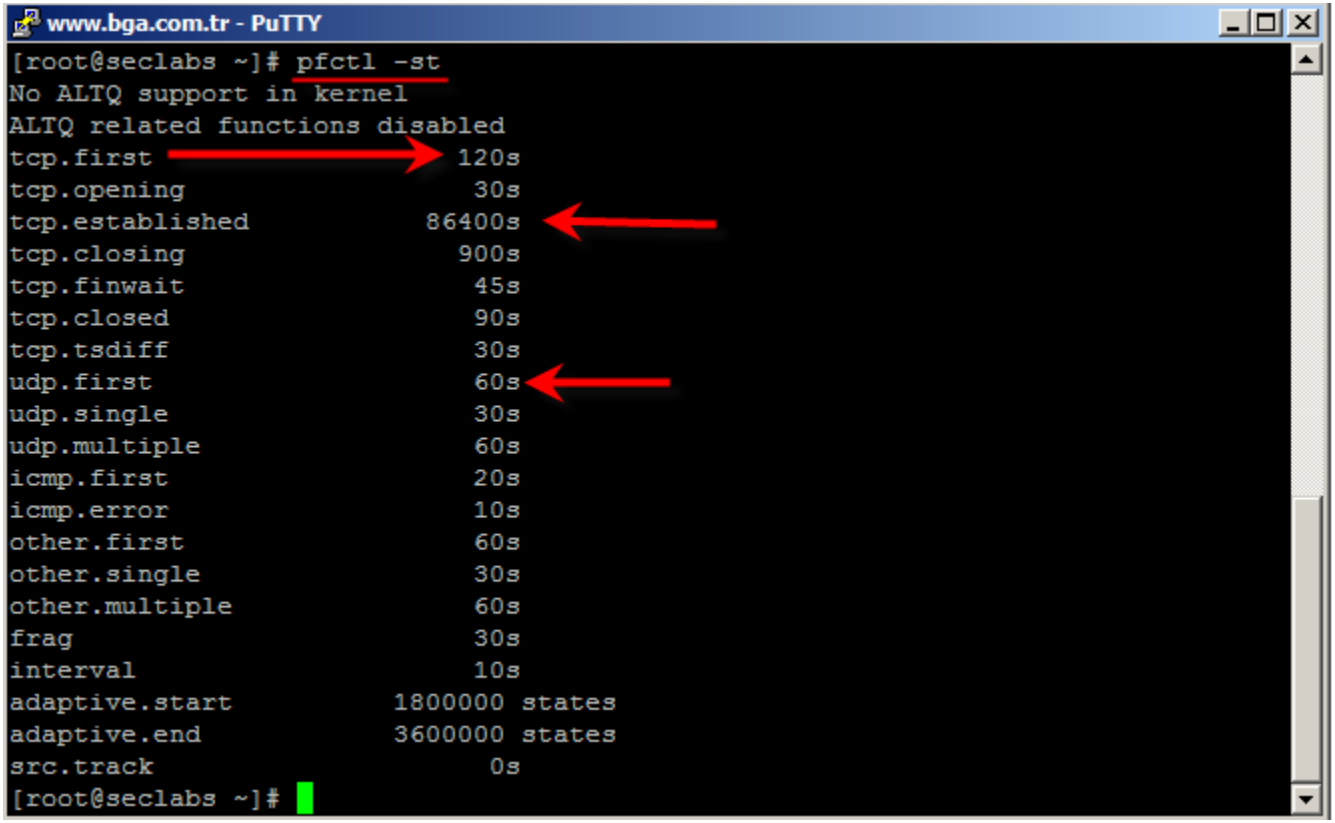
Firewall Yapılandırması

- Tüm ayarlar pf.conf dosyasından yapılır
- Pfctl komutu ile yönetilir
- #pfctl –e // Sistemi aktif hale getirir
- #pfctl –d // Sistemi pasif hale getirir
- #pfctl –f /etc/pf.conf // Yeni kuralları yükler
- #pfctl –sr //Yüklü kural ailesini gösterir
- ##pfctl –ss //State tablosunu gösterir

DDoS Koruma:TimeOut Değerleri

- Güvenlik duvarı, IPS, DDoS engelleme gibi inline çalışan ve state tutan tüm sistemler paketleri belirli zaman için tutarlar
- Bu zaman değerleri öntanımlı olarak yüksektir ve saldırıların başarılı olmasında etkindir
- Örnek:
- Gelen bir SYN paketi için bellekte 120 saniye son ACK cevabının gelmesi beklenir!

OpenBSD Öntanımlı Zaman Aşımı



Zaman Aşımı Sürelerini Düşürme

```
set timeout {tcp.first 10, tcp.opening 10 tcp.established 96000, tcp.closing 33, tcp.finwait 10, tcp.closed 20}
```

```
tcp.opening                10s
tcp.established         96000s
tcp.closing                33s
tcp.finwait                10s
tcp.closed                 20s
tcp.tsdiff                 30s
udp.first                  60s
udp.single                 30s
udp.multiple               60s
icmp.first                 20s
icmp.error                 10s
other.first                60s
other.single               30s
other.multiple             60s
frag                       30s
interval                   10s
adaptive.start        6300000 states
adaptive.end         12600000 states
src.track                  0s
You have new mail in /var/mail/root
```

DDoS Koruma:Anormal Paketler

- TCP bağlantılarında ilk paket mutlaka SYN olmalıdır
- FIN flood, ACK flood, PUSH flood gibi saldırılarda ilk paket SYN değildir

PF:Anormal Paketleri Eleme

- match in all scrub (no-df)

no-df
Clears the dont-fragment bit from a matching IP packet. Some operating systems are known to generate fragmented packets with the dont-fragment bit set. This is particularly true with NFS. Scrub will drop such fragmented dont-fragment packets unless no-df is specified.

Unfortunately some operating systems also generate their dont-fragment packets with a zero IP identification field. Clearing the dont-fragment bit on packets with a zero IP ID may cause deleterious results if an upstream router later fragments the packet. Using the random-id modifier (see below) is recommended in combination with the no-df modifier to ensure unique IP identifiers.

min-ttl <number>
Enforces a minimum TTL for matching IP packets.

max-mss <number>
Enforces a maximum MSS for matching TCP packets.

random-id
Replaces the IP identification field with random values to compensate for predictable values generated by many hosts. This option only applies to packets that are not fragmented after the optional fragment reassembly.

fragment reassemble
Using scrub rules, fragments can be reassembled by normalization. In this case, fragments are buffered until they form a complete packet, and only the completed packet is passed on to the filter. The advantage is that filter rules have to deal only with complete packets, and can ignore fragments. The drawback of caching fragments is the additional memory cost. But the full reassembly method is the only method that currently works with NAT. This is the default behavior of a scrub rule if no fragmentation modifier is supplied.

fragment crop
The default fragment reassembly method is expensive, hence the option to crop is provided. In this case, pf(4) will track the

Engellenen Paketlere Cevap Dönmeme

- Firewall bir paketi engellediğinde iki tür aksiyon alabilir
  - Paketi engelle ve geriye cevap dön
  - Paketi engelle ve geriye cevap dönme
- DDoS saldırılarında kesinlikle geriye cevap dönülmemeli
- Her gelen FIN paketine RST dönülürse iki kat performans harcanır

- set block-policy drop

State Tablosu Belirleme

- Toplamda kaç adet session(durum) tutulacağını belirler
  - Fiziksel ram miktarıyla doğru orantılıdır
  - set limit states 10500000
- Kaç adet kaynak IP adresine ait bilgi tutulacağını belirler
  - set limit src-nodes 5000000

Durum Tablosu Görüntüleme

```
Status: Enabled for 145 days 20:56:40          Debug: Urgent

Interface Stats for em0                IPv4              IPv6
  Bytes In                   5332984625231          29739690
  Bytes Out                 14370657365878                 0
  Packets In
    Passed                   13580471481            222550
    Blocked                    788635885                 0
  Packets Out
    Passed                   20182481531                 0
    Blocked                    150586291                 0

State Table                          Total             Rate
  current entries                     8908
  searches                      67593686319          5363.1/s
  inserts                         353680459            28.1/s
  removals                        353671551            28.1/s
Counters
  match                         35025741018          2779.1/s
  bad-offset                            0             0.0/s
  fragment                           4244             0.0/s
  short                           3728791             0.3/s
  normalize                             0             0.0/s
  memory                            52669             0.0/s
  bad-timestamp                         0             0.0/s
  congestion                            0             0.0/s
  ip-option                         75916             0.0/s
  proto-cksum                       85428             0.0/s
  state-mismatch                   828435             0.1/s
  state-insert                      16403             0.0/s
  state-limit                           0             0.0/s
  src-limit                       2418414             0.2/s
  synproxy                      472858114            37.5/s
```

FIN/ACK/PUSH Flood Engelleme

- TCP connection flood engelleme özelliği
- Temel mantık:İlk gelen paketin SYN olma zorunluluğu
- pass in log(all) on $ext_if proto tcp to $WEB_SUNUCU port 8000 **flags S/SA keep state**

Syn Flood Engelleme

- Packet Filter syn flood engelleme yöntemlerinden syn cookie değil syn proxy'i kullanır
- Syn proxy sesssion tuttuğu için sistemdeki fiziksel ram miktarı önemlidir

Packet Filter SynProxy Kullanımı

- pass in log(all) on $ext_if proto tcp to $web_server port {80 443} ***flags S/SA*** **synproxy state**

HTTP GET/POST Flood Engelleme

- HTTP GET/POST flood saldırılarında IP spoofing yapılamaz
- Saldırının başarılı olması için binlerce IP adresinden onlarca HTTP GET paketi gönderilmelidir
- OpenBSD PF kullanarak bir IP adresinden eş zamanlı veya belirli süree gelebilecek paket sayısını kısıtlayabiliriz
  - Rate limiting özelliği
- Belirli seviyenin üzerinde paket gönderenler engellenir!

HTTP GET Flood Engelleme-II

- HTTP GET paketi boyutu 400 Byte
- TCP SYN paketi boyutu 60 Byte
- HTTP GET flood saldırılarında sahip olduğumuz bandwith'in 8 katına kadar DDoS saldırılarını başarıyla engelleyebiliriz
  - Nasıl mı?

Packet Filter HTTP Flood Engelleme

- pass in log(all) quick on $ext_if proto tcp\

to $web_server port {80 443} flags S/SA\

synproxy state (max-src-conn 400, max-src-conn-rate 90/3, overload <ddos_host> flush global)

- max-src-conn *number* : Limit the maximum number of simultaneous TCP connections which have completed the 3-way handshake that a single host can make.
- max-src-conn-rate *number / interval* : Limit the rate of new connections to a certain amount per time interval.
- overload <*table*> :Put an offending host's IP address into the named table.
- flush [global]: Kill any other states that match this rule and that were created by this source IP. When global is specified, kill all states matching this source IP, regardless of which rule created the state.

Rate Limiting

- IP başına max bağlantı sayısını 100 ile limitle
- IP başına saniyede gönderilecek paket sayısını 10 ile limitle
- Bu kurallara uymayan IP adreslerini ddos tablosuna ekle
- Bu kural tarafından oluşturulan state tablosunu boşalt!

```
table <ddos> persist
block in quick from <ddos>

pass in on $ext_if proto tcp to $web_server \
    port 80 flags S/SA keep state \
    (max-src-conn 100, max-src-conn-rate 50/5, overload <ddos>
flush)
```

TCP Authentication

- Rate limiting uygulamalarının en önemli problemi IP spoofingdir
- Akıllı bir saldırgan rate limiting yapan bir sistemi spoof edilmiş IP adresleriyle kandırarak
- İstediği IP adreslerini engelletebilir
  - Root dns sunucular, Türkiye IP blokları vs
- OpenBSD PF TCP bağlantıları için rate limiting yapmadan önce IP adresinin gerçek olup olmadığını belirler!
- IP adresi gerçek değilse syn proxy'den geri döner
- IP adresi gerçekse rate limiting özellikleri devreye alınır

Probability Özelliği

- Gelen –giden paketlerin belirli oranını kabul etme/engelleme
- UDP flood engelleme amaçlı kullanılabilir
  - İlk gelen UDP paketini engelle
  - Eğer gönderen gerçekse tekrar gönderecektir
  - Eğer gönderen gerçek değilse tekrarlamayacaktır
- block in log quick on $Ext_If from <DDOSers> to any probability 50%

Grey Listing ÖZelliği

- SPAM engelleme için kullanılan bir özelliktir
- SMTP için kullanılır
- Amaç spam gönderen ve normal kullanıcıları ayırt etmektir
  - Spam gönderenler bir kere maili gönderir mail sunucudan dönecek cevaba bakmaz
  - Normal smtp bağlantılarında smtp bağlantısı kurulur mail gönderilir eğer bir hata dönerse sunucudan belirli müddet sonra tekrar gönderilir!
- Spamleri ek bir sisteme ihtiyaç duymadan %70 oranında engeller!

Packet Filter GrayListing

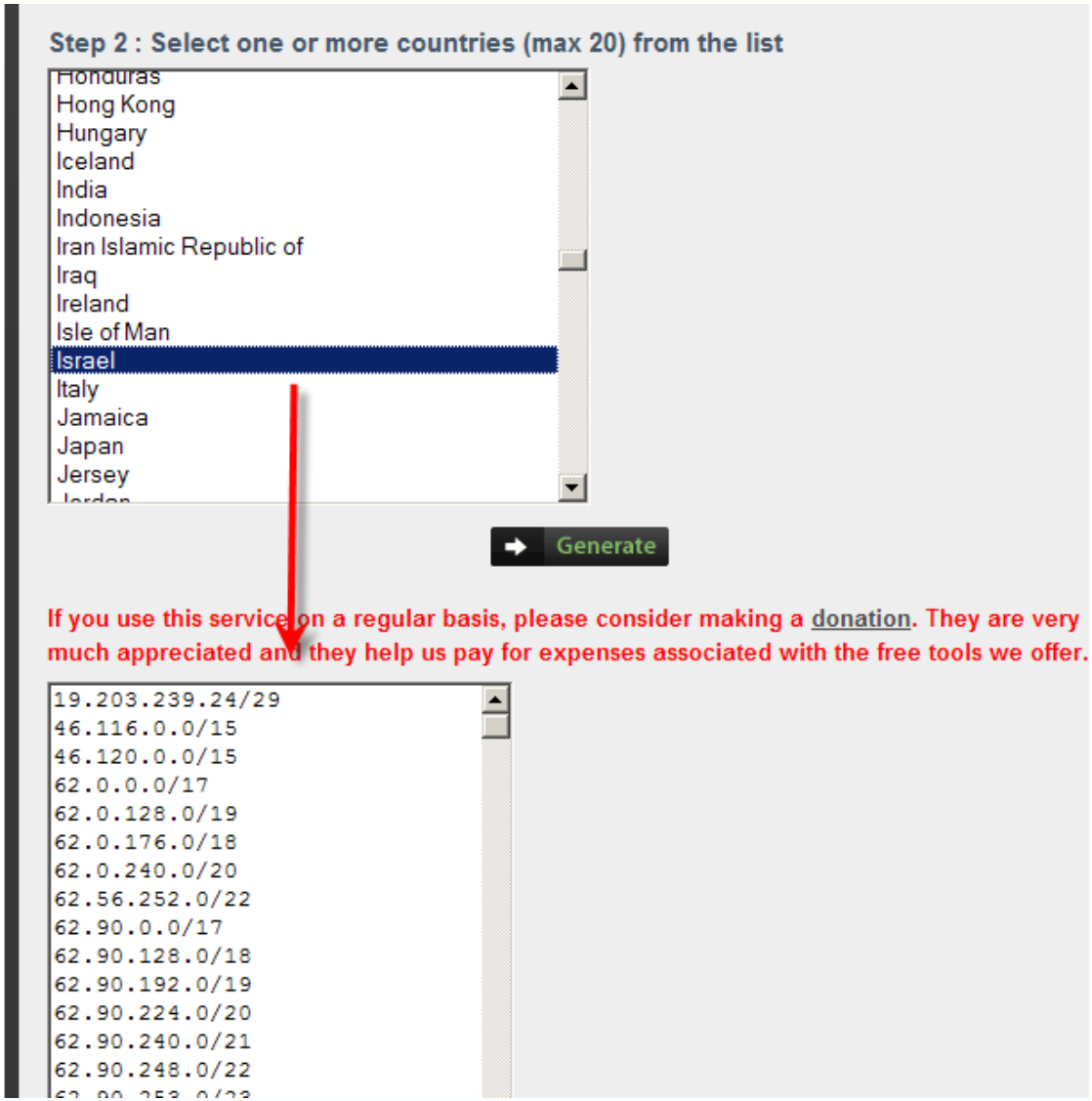
```
pf=YES
spamd_flags="-v -G 5:4:864"
spamd_grey=YES
```

```
ext_if="fxp0"
table <spamd-white> persist
rdr pass on $ext_if proto tcp from !<spamd-white> to
port smtp -> 127.0.0.1 port spamd
```

Ülkelere Göre IP Adresi Engelleme

- Özellikle spoof edilmeiş IP kullanılan saldırılarda trafik yoğun olarak bir ülkeden geliyorsa o ülkeye ait ip blokları tümden engellenebilir!
  - O ülkeden glecek ziyaretcilere farklı bir sayfa gösterilmelidir!

Ülke IP Aralıkları



Packet Filter Ülkeye Göre Bloklama

```
table <Turkiye> persist file "/etc/TR"
table <israil> persist file "/etc/Israil"
table <Cin> persist file "/etc/Cin"
table <Rusya> persist file "/etc/Rusya"
table <Usa> persist file "/etc/Usa"
table <India> persist file "/etc/India"
```

```
block in quick log on $ext_if from <israil>
```

```
[root@seclabs ~]# more /etc/israil
19.203.239.24/29
46.116.0.0/15
46.120.0.0/15
62.0.0.0/17
62.0.128.0/19
62.0.176.0/18
62.0.240.0/20
62.56.252.0/22
62.90.0.0/17
62.90.128.0/18
62.90.192.0/19
62.90.224.0/20
62.90.240.0/21
62.90.248.0/22
62.90.253.0/23
```

IP Spoofing Engelleme

- ISP'lerin mutlaka alması gereken önlemlerden
- Amaç: İç ağdan dışarı spoof edilmiş IP adreslerinin çıkmasını engellemek
- Standart olarak URPF kullanılır

Unicast Reverse Path Forwarding

- IP spoofing yapılmasını engelleme amaçlı standart bir özelliktir.

```
block in quick from urpf-failed label uRPF
```