

ÖZET	2
1. GİRİŞ	3
2. KULLANILAN YÖNTEMLER	4
2.1. MİNİNET VE SANAL MAKİNE KULLANIMI	4
2.2. OPENFLOW	4
2.2.1. <i>OpenFlow Anahtarı</i>	4
2.2.2. <i>Flow ve flow tablosu</i>	7
2.2.3. <i>OpenFlow Ağ Anahtarı Eylemleri</i>	8
2.2.4. <i>POX Denetleyicisi</i>	9
2.3. WIRESHARK.....	9
3. DDOS SALDIRISI.....	10
3.1. DDOS SALDIRISI NEDİR VE NASIL YAPILIR?.....	10
3.2. DDOS SALDIRI ÇEŞİTLERİ	12
3.2.1. <i>Yoğunluk Tabanlı Saldırıları</i>	12
3.2.2. <i>Protokol Kaynaklı Saldırıları</i>	13
3.2.2.1. Internet control message protocol floods	13
3.2.2.2. Ping of Death Saldırısı	18
3.2.2.3. Smurf saldırıları	20
3.2.2.4. TCP SYN flood saldırıları	20
3.2.2.4. UDP Flood saldırıları	27
3.3.3. <i>Uygulama Hedefli Saldırıları</i>	30
3.3.3.1. DNS Amplification saldırıları	30
3.3.3.2 HTTP GET-POST saldırıları	31
4. SONUÇ.....	35
KAYNAKLAR	36

ÖZET

Bu araştırma raporunda DDOS saldırıları tespiti ve benzetimi doğrultusunda ilk olarak yazılım tanımlı ağların ne olduğu ve hangi platform üzerinde çalışıldığı ve bu platformun ne işe yarayıp nasıl çalıştığı aynı zamanda içerisindeki protokoller hakkında detaylı şekilde araştırma yapılmıştır.

Sonrasında DDOS saldırıları hakkında detaylı bir şekilde açıklama yapılmıştır. DDOS saldırıları nelerdir, DDOS saldırıları ne çeşitlerle yapılabilir ve bunların sonucunda neler gerçekleşebilir konusunda açıklamalar mevcuttur. Bu saldırıların birçoğu sanal makine üzerinde denenip analizi yapılarak rapora etki sağlamıştır. Oluşturulan ağlar üzerinde çeşitli DDOS saldırı örnekleri gerçekleştirilip ağ üzerindeki değişimler takip edilmiş ve buna bağlı olarak sonuçlar çıkarılmıştır.

Yapılan saldırıların benzetimi doğrultusunda saldırı istatistikleri göz önüne alınarak saldırının tahmini ve benzetimi sağlanmıştır.

1. GİRİŞ

DDOS saldırıları günümüzde çok fazla kullanılan bir ağ saldırı türüdür. Bu saldırı türü ile hedef makinanın ve sunucuların genel olarak ağ trafiği sekteye uğratılmaya ve çökertilmeye çalışılır.

DDOS saldırıları birçok farklı şekilde yapılabilir. Saldırıyı yapan kişi hedef kişiye sahte ip adresleri (ip spoofing) üzerinde birçok istekte bulunur ve hedef bu isteklere cevap veremeyecek dereceye gelinceye kadar bunu yapar. Ağ trafiğinin son derece artmasından dolayı hedef kişinin veya şirketin ağı çöker. Bazı DDOS saldırıları ise CPU ve RAM üzerinde yoğunluk oluşturarak onları tüketmeye dayalı amaçlarla gönderilirler.

DDOS saldırıları en başta bir yönetici, altındaki bilgisayarlar ve onlara bağlı milyonlarca zombi bilgisayar tarafından yapılabilir. Bu zombi bilgisayarlar normal kişilerin kullandıkları bilgisayarlar olabilir. Kullanıcılar haberleri olmadan zararlı bir yazılım veya program kullandıklarında virüs tehlikesiyle karşılaşabilir ve artık o da bir zombi bilgisayar olabilir. Bilgisayar sürekli ağ üzerinde veri yollayıp durduğu için bilgisayarların internet hızları düşebilir.

2. KULLANILAN YÖNTEMLER

2.1. Mininet ve Sanal Makine Kullanımı

Yapılan proje Mininet platformu kullanılarak gerçekleştirilmiştir. Bu platformu biraz açıklamak gerekirse yazılım tanımlı ağlar oluşturmayı, bu ağlar üzerinde işlemler ve testler yapmayı buna bağlı olarak gerçeklemler yapmayı sağlar. Proje kapsamında Mininet üzerinde çeşitli ağ topolojileri kullanılarak ağ yapıları oluşturulup bu ağ içerisindeki yapılar arasında çeşitli DDOS saldırıları denenip analizi ve benzetimi yapılmıştır.

Linux ve Unix ortamlarında çalışabilen Mininet açık kaynak kodlu bir yazılımdır. Bu platform kullanılarak hem terminal üzerinde hem de Python API kullanılarak SDN (Software Defined Network) yani yazılım tanımlı ağlar oluşturulup anlatılan işlemler gerçekleştirilebilir [1].

Bu platformu çalıştırmak için bir sanal makine programı gerekmektedir. Sanal makine bilgisayar bilimi üzerinde düşündüğümüzde işletim sistemi ve bilgisayar arasında bir sanal ortam yaratarak bu sanal ortam üzerinde sanki başka bir bilgisayar kullanıyor olmamızı sağlayan bir ortamdır. Biz proje kapsamında VirtualBox programını tercih ettik.

2.2. OpenFlow

OpenFlow ağ protokollerini ve yeni tekniklerin kullanımdaki ağlarda geliştirilmesini sağlayan bir açık standarttır [2]. OpenFlow sistem yöneticilerine kurulan bir ağ üzerinde bulunan topoloji ve paket yollama, alma ve filtreleme gibi işlemlerde yönetim olanağı sağlar [3].

2.2.1. OpenFlow Anahtarı

OpenFlow içerisinde en önemli bileşen denetleyici ve OpenFlow ağ anahtarlarıdır. Ağ anahtarları ve yönlendiricilerde paket iletimi (Data Path) ve yüksek seviye yönlendirme karar mekanizması (Control Path) aynı cihazda bulunur. OpenFlow ise Data Path bileşenini fiziksel ortamda bırakır ve Control Path bileşenini bir sunucuya taşır.

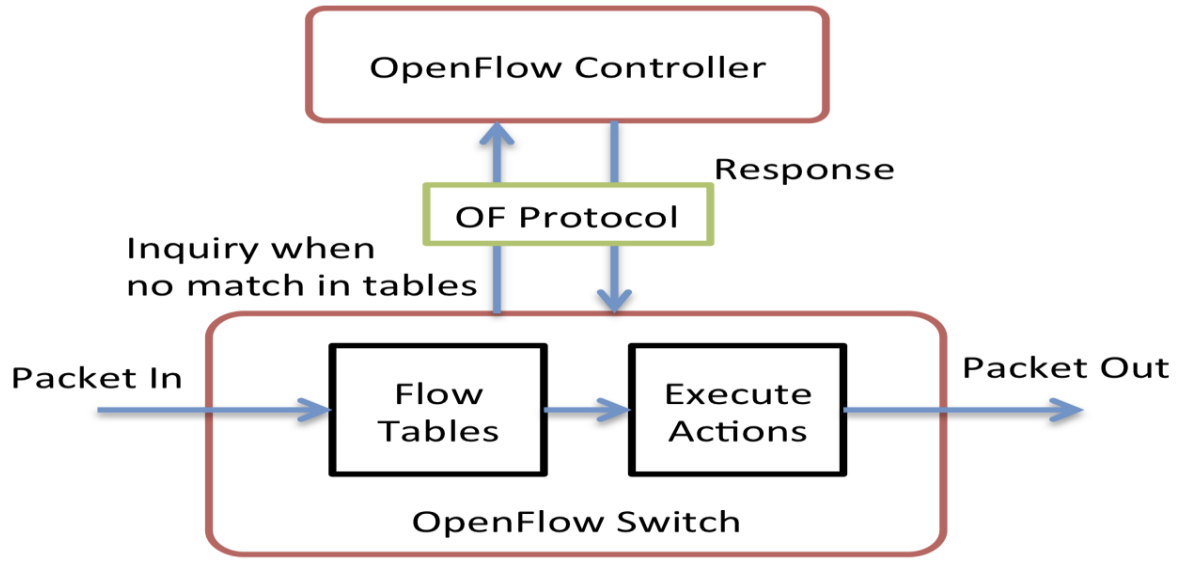
Bir cihazın OpenFlow desteklemesi ve bununla yapılabilecek tüm yönlendirme ve protokolleri çalıştırabiliyor olması anlamına gelir. Routing protokolleri, STP, QoS, Switching eldeki cihaz OpenFlow kullanıyor ise bunun desteklediği özellikleri çalıştırır.

OpenFlow anahtarı ile yerel ağlardan oluşmuş kümeler birbirine bağlanabilir. Genel yapısı aşağıdaki şekilde verilmiştir.

Ağ anahtarlarındaki hat kartları, verinin ağ anahtarlarına giriş ve çıkış yaptığı yerlerdir. Hat kartları, bilgisayar ağını bağlantı örgüsüyle ilişkilendirir.

Seri-paralel dönüşüm, senkronizasyon ve çerçeve işleme gibi fiziksel işlemler hat kartlarında bulunur. Bağlantı örgüsü, giriş ve çıkış hatlarını birbirine bağlar ve anahtarlama işini yapar. Bu işlemde sonra paketler çıkış hat kartlarının tampon belleğinde kuyruğa girer. Çıkış için gönderilecek olan paket çıkış hat kartındaki hizmet kalitesi çizelgeleyici tarafından seçilir. Bu çizelgeleyicinin görevi, farklı trafik kaynaklarına sağlanan hizmet ayrıcalıklarını ve önceliklerini sağlamaktır. Merkezi işlem birimi, adresi tablolarının güncellenmesi, paket istatistiklerinin toplanması gibi genel yönetim ve bakım işlemlerini yapar [4].

Daha genel anlamda bakıldığında ağ anahtarı yapısı iki bölüme ayrılabilir, Veri katı ve kontrol katı. Veri katı, ağ anahtarına gelen ve yönlendirilecek olanların tutulduğu kısımdır. Bunların işlenmesi kontrol katının görevidir. Bu iki katman ile ilgili olarak daha esnek bir yapıya geçmek için OpenFlow switchler tasarlanmıştır. Böylece kullanılagelen ağ anahtarlarında büyük değişiklikler yapılmış ve veri katmanı ile kontrol katmanı birbirinden ayrılmıştır. Bu da veri katmanının bir güvenli bağlantı üzerinden dışarıdan kontrol edilebilmesini getirmiştir.



Şekil 2-1 OpenFlow genel yapı. [5]

OpenFlow anahtarının ana fikri Kontrol katmanını ağ anahtarının dışına almaktır. İki tip OpenFlow ağ anahtarı mevcuttur. Birincisi donanım bazlı olup ticari ağ anahtarlarıdır ve TCAM ile Flow tablosu ve OpenFlow protokolünü kullanırlar.

Diğer tip ağ anahtarları ise yazılım tabanlı olup OpenFlow ağ anahtarı fonksiyonları bir adet UNIX/Linux işletim sistemi kullanarak yerine getirirler. OpenFlow ağ anahtarı en az üç adet bileşenden oluşmaktadır.

Bunlar flow tablosu, güvenli bağlantı ve OpenFlow protokolüdür.

Flow tablosu, anahtara bir flow'u nasıl değerlendirip işleme sokacağını belirten flow tablosu bileşeni içeren veri bütünüdür.

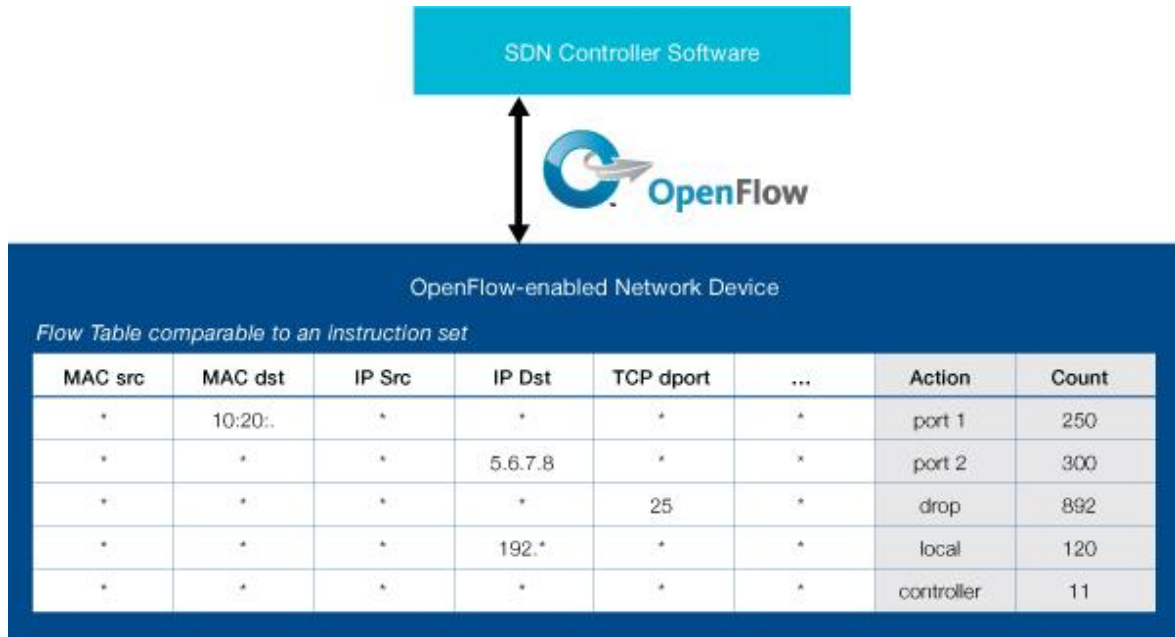
Güvenli bağlantı, anahtarın kullandığı komutlara ait paketlerin controller arasında haberleşmede kullanılır.

OpenFlow protokolü, anahtarla ve controller ile açık ve standart bir bağlantı imkanı sunan yöntemdir.

2.2.2. Flow ve flow tablosu

Flow tanımlanacak olursa aynı kaynak IP, hedef IP, kaynak portu, hedef portu, protokol öğelerini paylaşımına sunan ve ayarlanabilir bir zaman aşımı süresi ile seçilmiş tek yönlü paketler dizisidir [6].

Flow tabloları ise ağ anahtarları, yönlendiriciler gibi günümüz zamanındaki ağ bileşenlerinin çoğunda flow bileşenlerinin bulundurulduğu bir yapıdır. Bu yapı bileşenleri saklamak için kullanılır. Ağ anahtarlarında ve yönlendirici cihazların içinde data-plane içinde bulunur [7].



Şekil 2-2 Flow tablosu [8].

Flow tablosu üzerinde flow'ların tanımı için iki adet tablo bulunur. İlk tablo lineer bir tablo olup flow'ları tanımlayan karakterler içerir Bu tablodaki kayıt girdileri kayıt içindeki bir flow için sadece bazı karakterleri tanımlar (MAC adresi, IP adresi ve port gibi). Bu ilk tablo da flow kaydı içerir.

OpenFlow ağ anahtarı bu tablo içindeki başlık alanlarıyla, gelen flow'un başlık alanlarını karşılaştırır.

- Giriş portu
- Kaynak / hedef MAC adresleri
- Ethernet protokolü
- Kaynak/hedef IP adresi

- Ağ protokolü
- Kaynak/hedef portu

Ayrıca OpenFlow ağ anahtarı tablo verileriyle flow'ları karşılaştırarak işlem yapar [9].

2.2.3. OpenFlow Ağ Anahtarı Eylemleri

OpenFlow ağ anahtarlarını eylemleri iki adet ana bölümde toplanır. Bunlar zorunlu eylemler ve opsiyonel eylemler.

Anlaşılabacağı gibi zorunlu eylemlerin tüm OpenFlow ağ anahtarlarında olması gerekmektedir.

Opsiyonel eylemler ise sonradan geliştirebilir olanlardır. OpenFlow ağ anahtarları iki kategoriye ayrılırlar:

1-Dedicated OpenFlow ağ anahtarları

2-OpenFlow etkin (enabled) ağ anahtarları [10].

OpenFlow ağ Flow tablosunda bulunan flow'larla ilgili Dedicated OpenFlow ağ anahtarlarında 3 adet işlem gerçekleştirir.

Bir diğer dördüncü işlem ise OpenFlow protokolü etkin ağ anahtarlarında gerçekleşmektedir. Bu işlemleri açıklayacak olursak;

1. Flow paketlerinin belirtilen port veya portlara yönlendirilmesi. Bu da bir paketin ağ içinde iletimine imkan verir. Çoğu ağ anahtarında beklenen belli bir oranda bu adımı gerçekleştirmesidir.

2. Flow paketlerini izole eder ve denetleyiciye yollar. Bu paketler güvenli bağlantı ile işleme alınır ve gitmesi gereken yere gönderilirler.

Burada denetleyici, flow paketinin akıbetine karar verir. Flow tablosuna ekleme yapabilir ya da işleme sokulmadan ağ içinde kontrol edebilir.

3. Flow paketlerini siler. Ağa yönelik bir saldırı olduğu durumlarda, denetleyici bu paketi siler.
4. OpenFlow protokolü etkin ağ anahtarlarında, flow, ağ anahtarının normal işlem sırasına sokulur [11].

2.2.4. POX Denetleyicisi

OpenFlow teknolojinin bir ayrı önemli bölümü de denetleyicidir. Denetleyici (Controller) tanımı kısaca uzaktan kontrol işlemi olarak söylenebilir. OpenFlow ağ anahtarlarının flow tablolarının ağ anahtarı içinde olmayan bir bilgisayardan kontrol edilmesini sağlar. Anahtardaki flow tablosunun flow bileşenleri bu controller sayesinde basit olarak eklenip çıkarılabilir. Controller sayesinde, anahtardaki ağ ile ilgili değişkenlikler görülebilir ve anahtarın konfigürasyonundaki bazı düzenlemeler de yapılabilir. POX, python programlama diliyle geliştirilmiştir. Linux, Unix ve Windows işletim sistemleri gibi ortamlarda çalışabilmektedir. Python programlama dili ile yazılmış olmasına bağlı olarak, python runtime destekleyen her işletim sisteminde çalışma imkanı sağlar [11].

2.3. WireShark

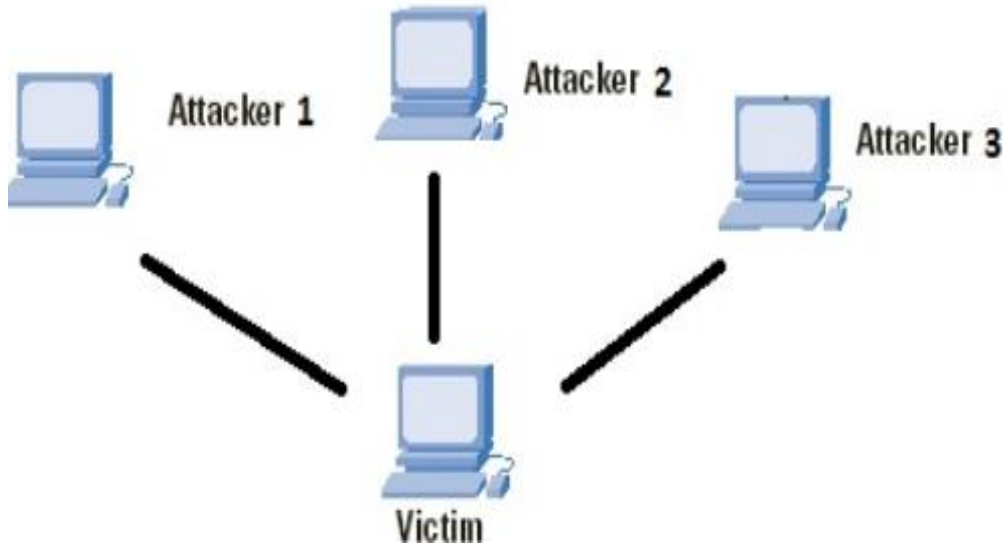
Wireshark ağ trafiğinin, bir grafik arayüz üzerinden izlenmesini sağlayan, pek çok zaman çok yaptığı kritik işler çok büyük öneme sahip bir programdır. Uygulamanın kurulu olduğu bilgisayar üzerinden anlık ağ trafiği izlenebileceği gibi ağ üzerinden gelen saldırıların tespit ve benzetimi gibi işlemlerde yardım olabilir. Ağ üzerinde alınan paketlerin içeriği nerden geldiği ne zaman geldiği ve daha birçok ayrıntısına ulaşmayı sağlayacak bir program olan Wireshark çok gerekli bir programdır. Wireshark daha önce kayıtlı dosyaların incelenmesi amacı ile de kullanılabilir [12].

3. DDOS SALDIRISI

3.1. DDOS Saldırısı nedir ve nasıl yapılır?

Dos saldırısı bir veya daha fazla kişinin bir veya daha fazla makine kullanarak bir kurbanı çeşitli yollar ile saldırdığı saldırı çeşididir.

Bu saldırının hedefi sunucuları sekteye uğratmaktır. Saldırımı yapanın hedefi başta merkezi işlem birimi ve bellek kaynaklarıdır. Saldırı neticesinde cihazın yavaşlaması veya kapanması gibi durumlara yol açabilen dos saldırıları temel olarak ağı çökertmeyi baz alır [13].



Şekil 3-1 DOS atak örnekleme [14].

DDOS saldırıları çok boyutlu da gerçekleşebilir. Bu saldırılar ağ üzerinde ciddi zararlar oluştururlar. Bu saldırılar neticesinde bilgisayarların internet üzerinde kontrolleri alınabilir. Saldırganlar tarafından ağ yönetimi ele geçirilebilir.

Bu saldırılar 2 tipe ayrılabilir.

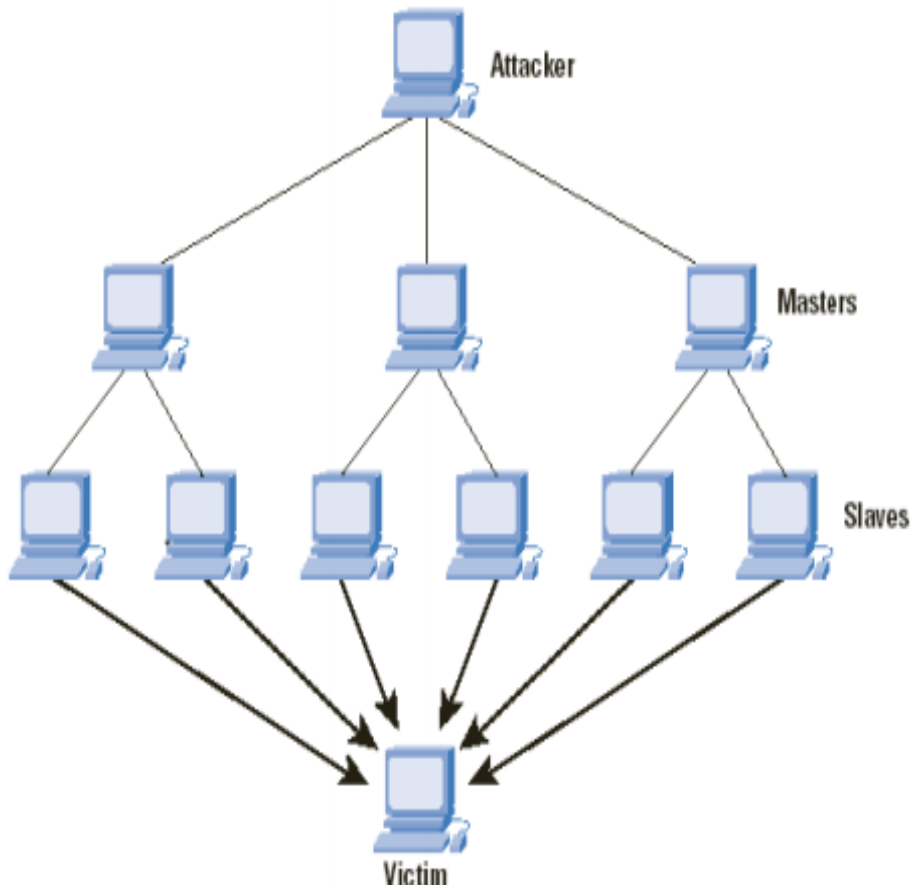
1- Doğrudan Saldırı

2- Reflektör (Yansıtıcı) Saldırı

Doğrudan saldırı yapılırken saldıran kişinin kurbanına kısa süreler içerisinde çok sayıda paket göndermesi ile ağ trafiği oluşur ve bir süre sonra kilitlenmeler meydana gelir ve ağda aksamalar yaşanır.

Diğer tip saldırı da ise zombi bilgisayar olarak da adlandırdığımız çok sayıda bilgisayar üzerinden yapılan saldırılar ile paketler gönderilir ve sistem ele geçirilmeye çalışılır.

Tipik bir DDOS saldırısı 3 unsurdan meydana gelir. Bunlar ilk olarak bir ana cihaz. Daha sonrasında bu ana cihaza bağlı olan bir dizi cihaz ve bu cihazlara bağlı olan çok sayıda zombi makineden oluşan bir sistem vardır . Bu sistemi Şekil 3-2’ de görebilirsiniz.



Şekil 3-2 Kompleks DDOS Saldırı Örnekleme [14].

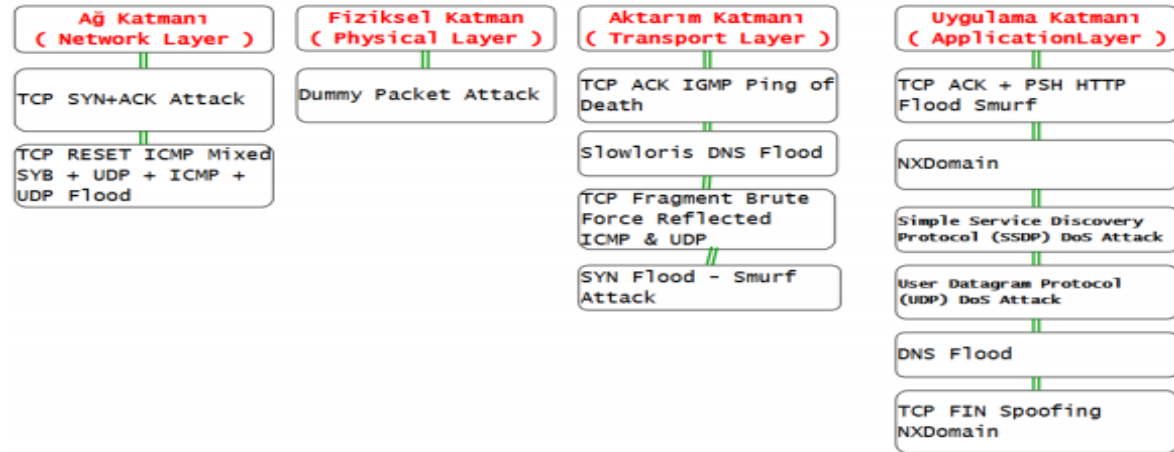
3.2. DDOS Saldırı Çeşitleri

Saldırı kategorileri hakkında genel bir görüş yoktur. Bu saldırıların çoğu kurumsal ve ticaret bazlı şirketleri hedef alır. En çok kullanılan saldırılar arasında SYN flood ve DNS flood gibi saldırılar yer alır. Otoritelerin yaptığı çeşitli sınıflandırmalar mevcuttur. Bu saldırıları şekil üzerinde gösterecek olursak;

Türlerine Göre ;



saldırı şekillerine Göre;

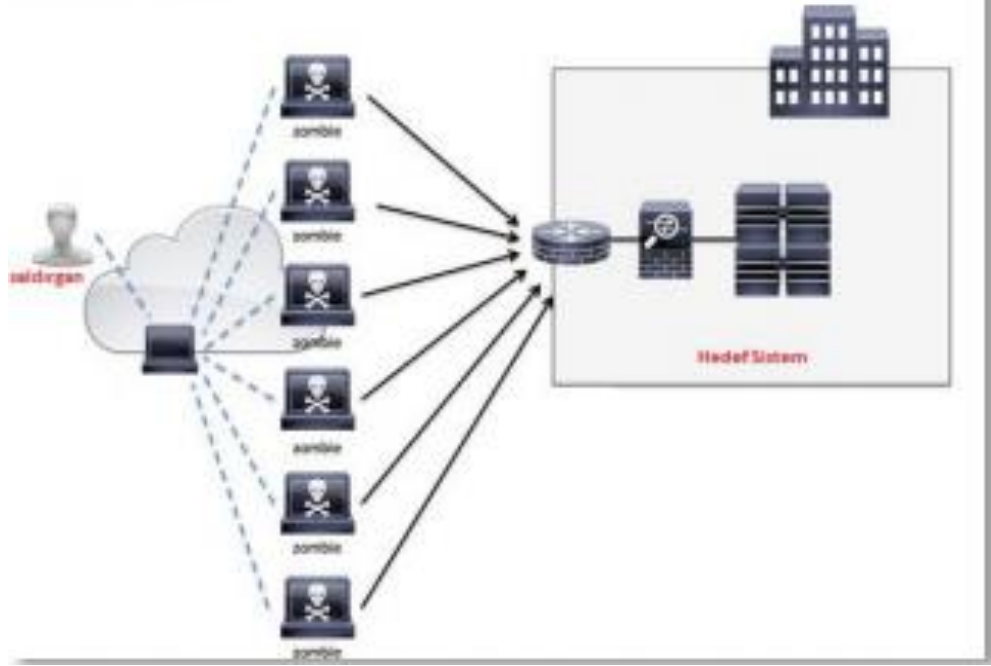


Şekil 3-3 DDOS saldırı çeşitleri [15].

3.2.1. Yoğunluk Tabanlı Saldırıları

Bu saldırılarda hedef saldırılan kaynağın bant genişliğini tüketmeye yöneliktir. Hedefe doğru sürekli çok sayıda yapılan istekler ile çok geniş bant kaynaklarından botnet sistemine dahil olmuş çok sayıda zombi bilgisayar sistemi üzerinden yapılan saldırı çeşididir.

Botnet Örneği



Şekil 3-4 Botnet ağı ve örneği [15].

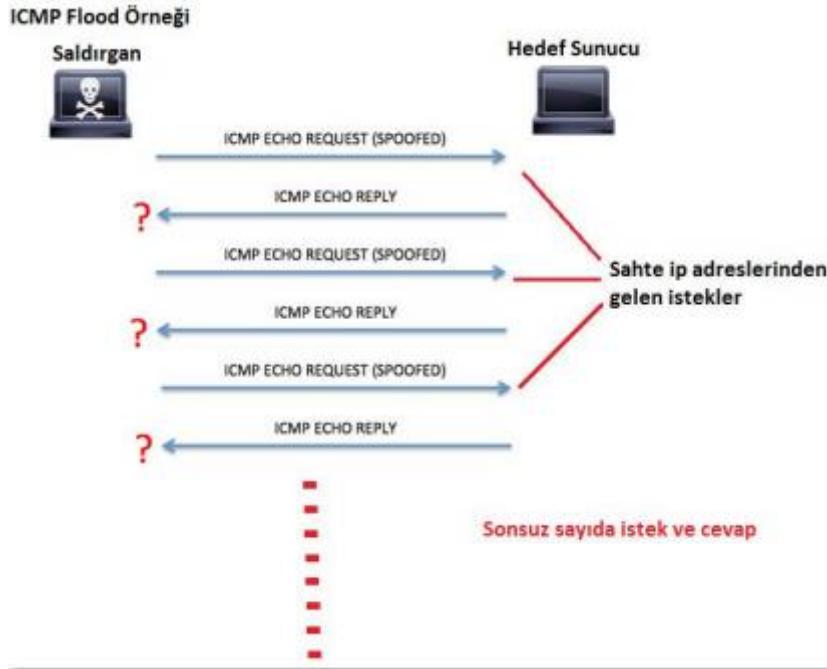
DDOS saldırısı sırasında alınan log'larda en belirgin durum kaynak ip adreslerinin çoğunluğudur. İp spoofing gibi teknikler ile çok fazla sayıda ip adresi görülmesi sağlanabilir. Bu ip adresleri botnet'e dahil olan ve bir merkezden yönetilen bilgisayarlardır. Bu botnet sadece DDOS amaçlı değil çok amaçlı kullanılabilir.

Botnet'e dahil olan çoğu kullanıcı bundan haberdar değildir. Bilgisayara indirilen zararlı bir yazılım veya program ile kullanıcıların bilgisayarları botnet'e dahil olabilir. Buna bağlı olarak botnet'e bağlı bilgisayar internete sürekli paket gönderdiğinden internet hızı yavaşlar.

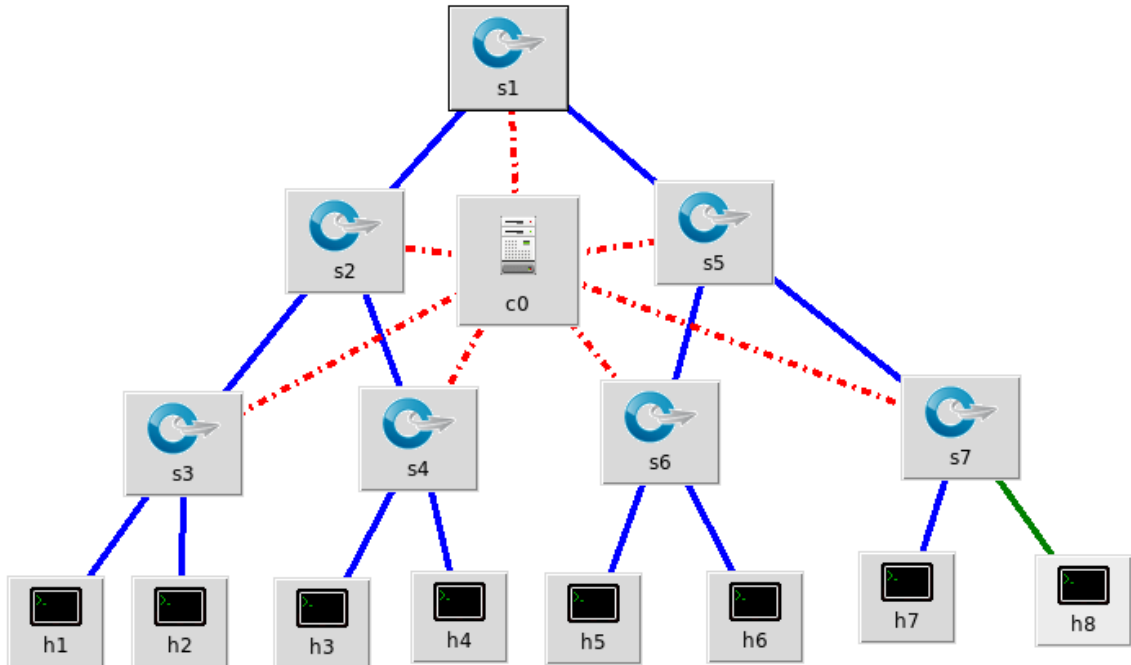
3.2.2. Protokol Kaynaklı Saldırıları

3.2.2.1. Internet control message protocol floods

En eski saldırılardan biridir. TCP/IP nin çalışma mantığında olan soru cevap tekniğini kullanır. ICMP protokolü üzerinde çalışır. Hedefe gönderilen istek ile hedefin açık olup olmadığı bilgisi geri döner. Eğer hedef açık ise sürekli gönderilen istekler ile hedef meşgul edilir. Çünkü hedef kendisine gelen her isteğe cevap vermek zorundadır. Hedefe doğru yapılan sürekli istekler sahte ip adresleri ile gerçekleştirilir.



Şekil 3-5 ICMP Flood örneği [17]



Şekil 3-6 Örnek Saldırı için oluşturulan topolojinin gösterimi.

```

mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8
h2 -> h1 h3 h4 h5 h6 h7 h8
h3 -> h1 h2 h4 h5 h6 h7 h8
h4 -> h1 h2 h3 h5 h6 h7 h8
h5 -> h1 h2 h3 h4 h6 h7 h8
h6 -> h1 h2 h3 h4 h5 h7 h8
h7 -> h1 h2 h3 h4 h5 h6 h8
h8 -> h1 h2 h3 h4 h5 h6 h7
*** Results: 0% dropped (56/56 received)
mininet> links
s1-eth1<->s2-eth3 (OK OK)
s1-eth2<->s5-eth3 (OK OK)
s2-eth1<->s3-eth3 (OK OK)
s2-eth2<->s4-eth3 (OK OK)
s3-eth1<->h1-eth0 (OK OK)
s3-eth2<->h2-eth0 (OK OK)
s4-eth1<->h3-eth0 (OK OK)
s4-eth2<->h4-eth0 (OK OK)
s5-eth1<->s6-eth3 (OK OK)
s5-eth2<->s7-eth3 (OK OK)
s6-eth1<->h5-eth0 (OK OK)
s6-eth2<->h6-eth0 (OK OK)

```

Şekil 3-7 ICMP Flood için oluşturulan topoloji içindeki bağlantıların gösterimi.

Saldırının benzetimi konusuna gelecek olursak bunun için tcpdump ve Wireshark üzerinde inceleme yapıldığında saldırının ICMP protokolü üzerinden geldiği açıktır. Bu da ICMP Flood'a işaret eder.

```
root@mininet-vm:~# tcpdstat icmpFlood.pcab

DumpFile: icmpFlood.pcab
FileSize: 31.44MB
Id: 201706071325
StartTime: Wed Jun 7 13:25:35 2017
EndTime: Wed Jun 7 13:26:33 2017
TotalTime: 57.43 seconds
TotalCapSize: 27.03MB CapLen: 98 bytes
# of packets: 289224 (27.03MB)
AvgRate: 3.95Mbps stddev:0.27M PeakRate: 4.38Mbps

### IP flow (unique src/dst pair) Information ###
# of flows: 6 (avg. 48204.00 pkts/flow)
Top 10 big flow size (bytes/total in %):
 16.8% 16.8% 16.7% 16.7% 16.5% 16.5%

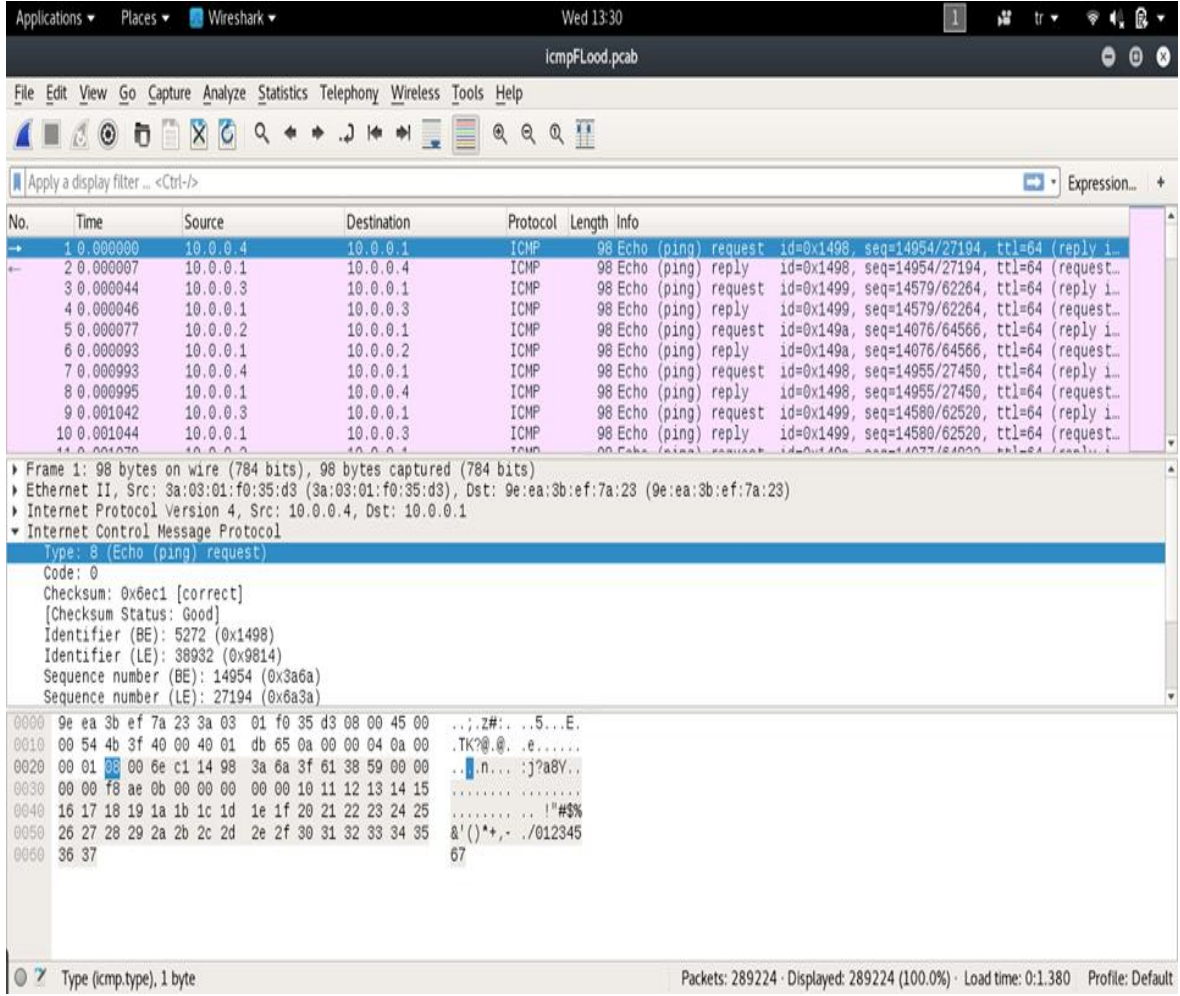
### IP address Information ###
# of IPv4 addresses: 4
Top 10 bandwidth usage (bytes/total in %):
 100.0% 33.7% 33.3% 33.0%

### Packet Size Distribution (including MAC headers) ###
<<<<
[ 32- 63]: 6
[ 64- 127]: 289218
>>>>

### Protocol Breakdown ###
<<<<
protocol                packets                bytes                bytes/pkt
-----
[0] total                289224 (100.00%)      28343616 (100.00%)    98.00
[1] ip                   289218 (100.00%)      28343364 (100.00%)    98.00
[2] icmp                 289218 (100.00%)      28343364 (100.00%)    98.00
>>>>

root@mininet-vm:~#
```

Şekil 3-9 ICMP Flood saldırısının tcpdstat ile incelenmesi.



Şekil 3.10 ICMP Flood saldırısının Wireshark üzerinden takibi.

Saldırı öncesinde 22.0 Gbit/sec olan bant genişliği ICMP Flood saldırısı sonrasında 4.36 Gbit/sec gibi bir değere düşmüştür. Bu düşüş ciddi bir değerdir. Erişilebilirlik konusunda bakıldığında erişim olarak bütün host'lar birbirine erişebilmektedir fakat hızları düşmüştür.

3.2.2.2. Ping of Death Saldırısı

Ping özelliklerinden faydalanılarak maksimum 65535 byte büyüklüğünde IP paketleri yaratılır. Bu paketler sisteme gönderilerek sistemin çökmesi, askıda kalması, işlem yapmaması veya reboot etmesine yol açar.

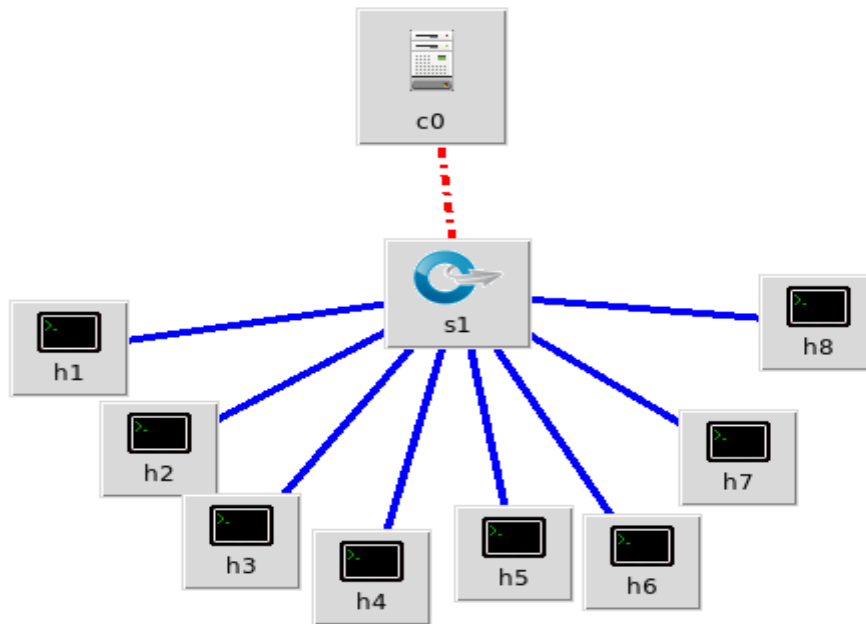
Ping of Death saldırısını denedik. Bunun farklı özelliği ise gönderilen paketlerin boyutlarının oldukça büyük olması ve sık aralıklı paket gönderilmesini sağlamasıdır.

Ping of Death saldırısı için,

Saldırı için öncelikle bir topoloji tanımlandı. Tanımlanan topoloji ve bunun görsel olarak gösterimi aşağıdadır.

```
mininet@mininet-vm:~$ sudo mn --topo single,8
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1) (h4, s1) (h5, s1) (h6, s1) (h7, s1) (h8, s1)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet> iperf
*** Iperf: testing TCP bandwidth between h1 and h8
*** Results: ['22.3 Gbits/sec', '22.3 Gbits/sec']
```

Şekil 3-11 Topoloji tanımlanması ve saldırı öncesi bilgiler.



Şekil 3-12 Ping of Death saldırısı için kullanılan topolojinin görsel hali.

```

*** Results: ['12.7 Gbits/sec', '12.7 Gbits/sec']
mininet> miperf
*** Iperf: testing TCP bandwidth between h1 and h8
*** Results: ['11.7 Gbits/sec', '11.7 Gbits/sec']
mininet> iperf
*** Iperf: testing TCP bandwidth between h1 and h8
*** Results: ['6.82 Gbits/sec', '6.84 Gbits/sec']
mininet> iperf
*** Iperf: testing TCP bandwidth between h1 and h8
*** Results: ['9.63 Gbits/sec', '9.64 Gbits/sec']
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8
h2 -> h1 h3 h4 h5 h6 h7 h8
h3 -> h1 h2 h4 h5 h6 h7 h8
h4 -> h1 h2 h3 h5 h6 h7 h8
h5 -> h1 h2 h3 h4 h6 h7 h8
h6 -> h1 h2 h3 h4 h5 h7 h8
h7 -> h1 h2 h3 h4 h5 h6 h8
h8 -> h1 h2 h3 h4 h5 h6 h7
*** Results: 0% dropped (56/56 received)
mininet>

```

Şekil 3-13 Ping of Death saldırısı sırasında ölçülen bant genişliği değerleri.

3.2.2.3. Smurf saldırıları

Bu saldırı mantığında ise hedef istek göndermekten çok karşı tarafın istek yapmasına dayanır. Bulunduğu yerel ağ içerisinde sürekli kontrolsüz istek yapmış gibi gösterilir. Bu saldırı hedefin ip adresini taklit ederek gerçekleşir [15].

3.2.2.4. TCP SYN flood saldırıları

TCP haberleşmesi, kullanıcı ile başlayan ve sunucu ile yanıtlanan haberleşme verilerinin tamamlanması ile oluşur. Tüm bilgiler bayrak (flag) ile saklanır.

Closed: Bağlantı başlamadan önceki bekleme durumudur.

İstemci hedefe bir SYN paketi gönderir.

Hedef istemciye bir SYN-ACK (acknowledgement) bilgisi döndürür.

İstemci tekrardan ACK cevabı gönderir.

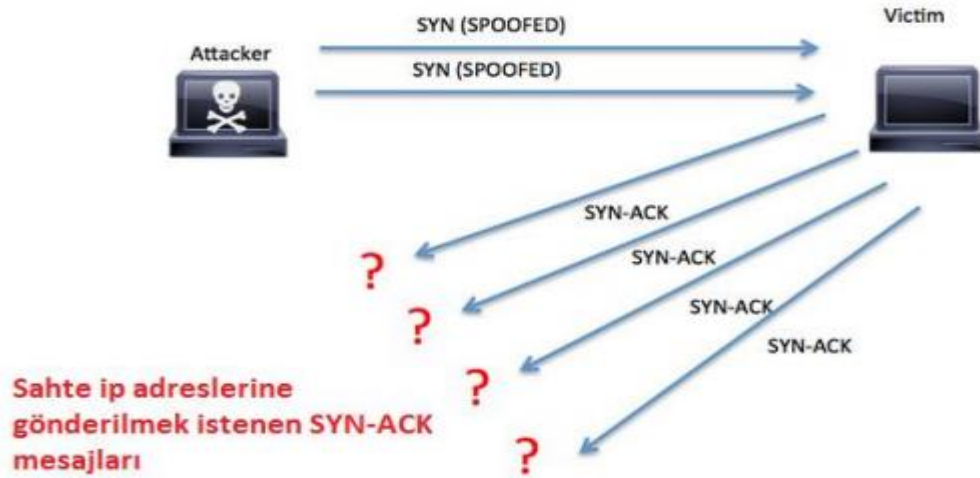
Bu üçlü haberleşmeye TCP three-way handshake denir [18].



Şekil 3-14 Three-way handshake [15].

SYN Flood saldırılarında ise saldırgan hedefe gönderdiği isteklere cevap vermez, spoof edilmiş ip adresleri ile sunucunun sürekli olarak sahte adreslere cevap döndürmesi sağlanır.

SYN Flood Örneği



Şekil 3-15 SYN Flood örnekleme [15].

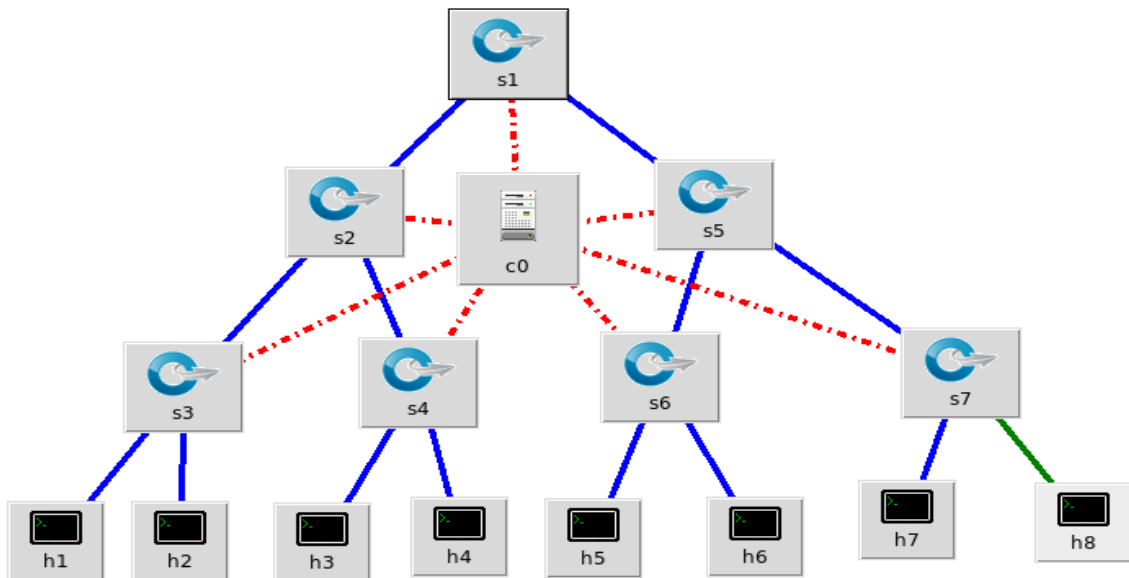
Bir SYN Flood saldırısı örnekleyelim.

```

mininet> iperf
*** Iperf: testing TCP bandwidth between h1 and h8
*** Results: [20.8 Gbits/sec, 20.8 Gbits/sec]
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8
h2 -> h1 h3 h4 h5 h6 h7 h8
h3 -> h1 h2 h4 h5 h6 h7 h8
h4 -> h1 h2 h3 h5 h6 h7 h8
h5 -> h1 h2 h3 h4 h6 h7 h8
h6 -> h1 h2 h3 h4 h5 h7 h8
h7 -> h1 h2 h3 h4 h5 h6 h8
h8 -> h1 h2 h3 h4 h5 h6 h7
*** Results: 0% dropped (56/56 received)
mininet> links
s1-eth1<->s2-eth3 (OK OK)
s1-eth2<->s5-eth3 (OK OK)
s2-eth1<->s3-eth3 (OK OK)
s2-eth2<->s4-eth3 (OK OK)
s3-eth1<->h1-eth0 (OK OK)
s3-eth2<->h2-eth0 (OK OK)
s4-eth1<->h3-eth0 (OK OK)
s4-eth2<->h4-eth0 (OK OK)
s5-eth1<->s6-eth3 (OK OK)
s5-eth2<->s7-eth3 (OK OK)
s6-eth1<->h5-eth0 (OK OK)
s6-eth2<->h6-eth0 (OK OK)
s7-eth1<->h7-eth0 (OK OK)
s7-eth2<->h8-eth0 (OK OK)
mininet>

```

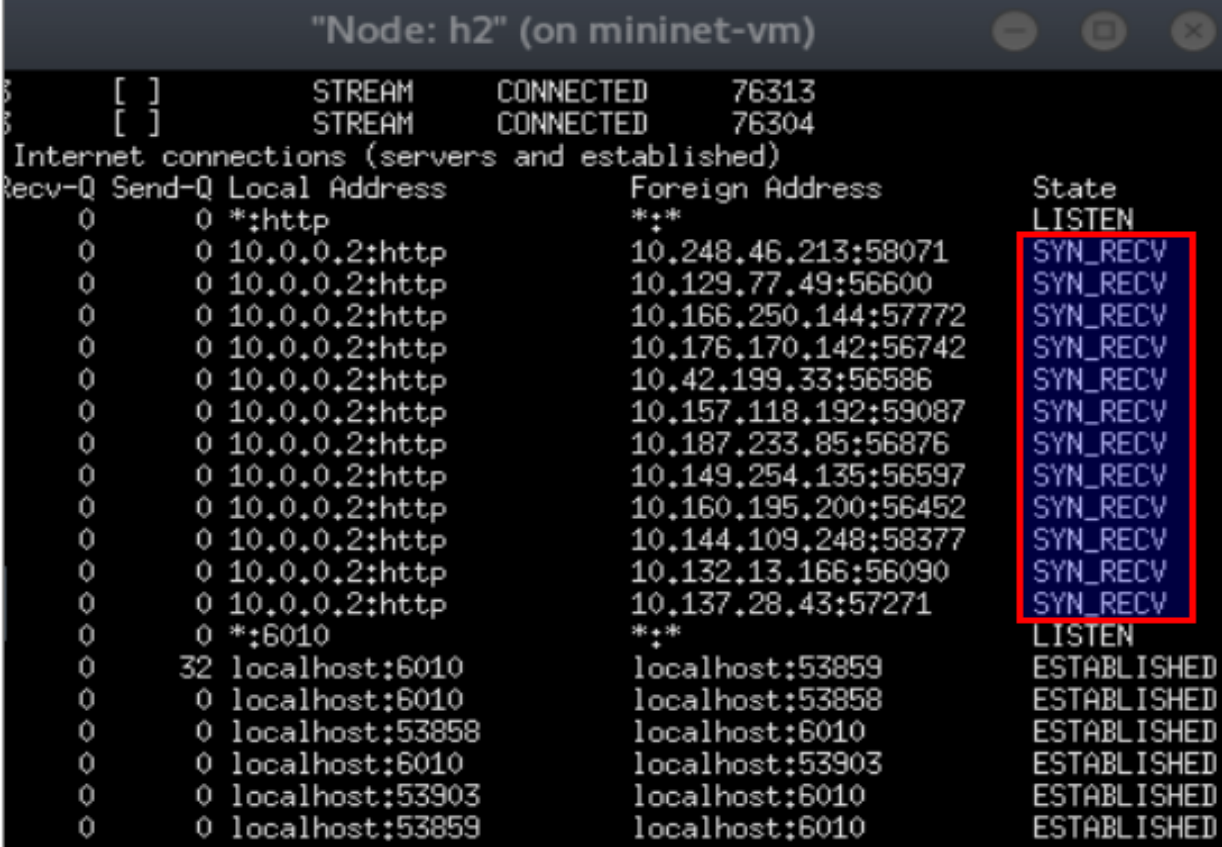
Şekil 3-16 SYN Flood Saldırısı öncesi oluşturulan topoloji yapısı ve bant genişliğini gösteren şekil.



Şekil 3-17 SYN Flood saldırısı için kullanılan topolojinin görsel hali.

Burada saldırı h2 hostu içinde apache2 web server'ına saldırı yapılmıştır.

Örnek bir SYN Flood saldırısı sırasında saldırının yapıldığı tarafta açılan bağlantılar.



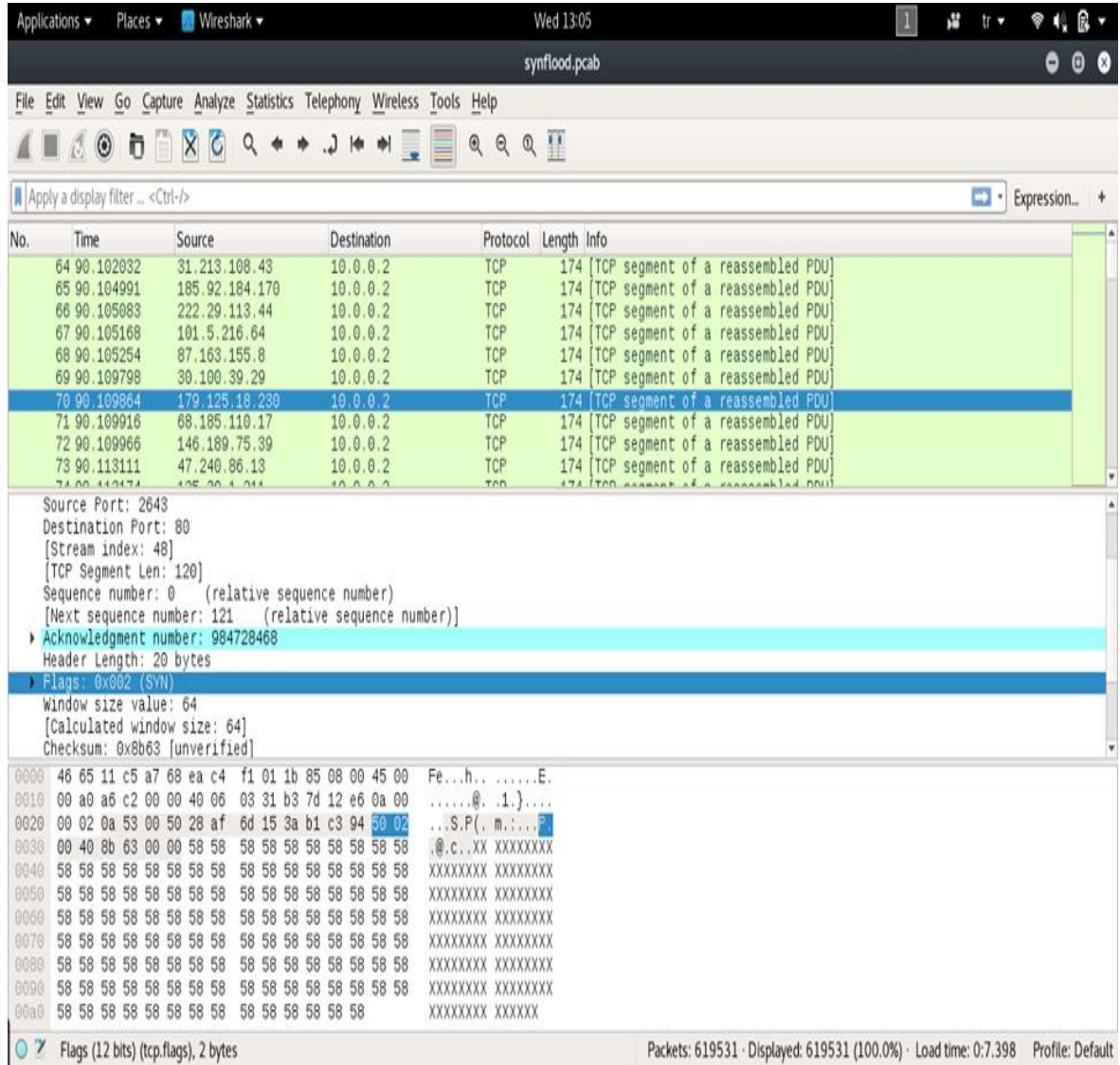
```
"Node: h2" (on mininet-vm)
$ [ ] STREAM CONNECTED 76313
$ [ ] STREAM CONNECTED 76304
Internet connections (servers and established)
Recv-Q Send-Q Local Address Foreign Address State
0 0 *:http *:~ LISTEN
0 0 10.0.0.2:http 10.248.46.213:58071 SYN_RECV
0 0 10.0.0.2:http 10.129.77.49:56600 SYN_RECV
0 0 10.0.0.2:http 10.166.250.144:57772 SYN_RECV
0 0 10.0.0.2:http 10.176.170.142:56742 SYN_RECV
0 0 10.0.0.2:http 10.42.199.33:56586 SYN_RECV
0 0 10.0.0.2:http 10.157.118.192:59087 SYN_RECV
0 0 10.0.0.2:http 10.187.233.85:56876 SYN_RECV
0 0 10.0.0.2:http 10.149.254.135:56597 SYN_RECV
0 0 10.0.0.2:http 10.160.195.200:56452 SYN_RECV
0 0 10.0.0.2:http 10.144.109.248:58377 SYN_RECV
0 0 10.0.0.2:http 10.132.13.166:56090 SYN_RECV
0 0 10.0.0.2:http 10.137.28.43:57271 SYN_RECV
0 0 *:6010 *:~ LISTEN
0 32 localhost:6010 localhost:53859 ESTABLISHED
0 0 localhost:6010 localhost:53858 ESTABLISHED
0 0 localhost:53858 localhost:6010 ESTABLISHED
0 0 localhost:6010 localhost:53903 ESTABLISHED
0 0 localhost:53903 localhost:6010 ESTABLISHED
0 0 localhost:53859 localhost:6010 ESTABLISHED
```

Şekil 3-18 SYN Flood saldırısı sırasında alınan paketler.

```
"Node: h2" (on mininet-vm)
6983, win 64, length 120
12:28:34.949745 IP 135.5.184.124.5095 > 10.0.0.2.http: Flags [S], seq 549625749:54962
5869, win 64, length 120
12:28:34.949769 IP 253.66.103.132.5094 > 10.0.0.2.http: Flags [S], seq 1073245756:107
3245876, win 64, length 120
12:28:34.951340 IP 106.235.195.26.5096 > 10.0.0.2.http: Flags [S], seq 852659038:8526
59158, win 64, length 120
12:28:34.953650 IP 138.252.103.250.5097 > 10.0.0.2.http: Flags [S], seq 1337062865:13
37062985, win 64, length 120
12:28:34.954836 IP 149.34.233.89.5099 > 10.0.0.2.http: Flags [S], seq 319157533:31915
7653, win 64, length 120
12:28:34.954869 IP 170.239.168.95.5098 > 10.0.0.2.http: Flags [S], seq 59603543:59603
663, win 64, length 120
12:28:34.955792 IP 0.250.168.176.5100 > 10.0.0.2.http: Flags [S], seq 2007572805:2007
572925, win 64, length 120
12:28:34.957016 IP 233.114.199.77.5102 > 10.0.0.2.http: Flags [S], seq 1908412401:190
8412521, win 64, length 120
12:28:34.957048 IP 64.90.127.169.5101 > 10.0.0.2.http: Flags [S], seq 868345742:86834
5862, win 64, length 120
12:28:34.957078 IP 101.111.225.224.5103 > 10.0.0.2.http: Flags [S], seq 33346366:3334
6486, win 64, length 120
12:28:34.958014 IP 49.103.151.55.5104 > 10.0.0.2.http: Flags [S], seq 1315457519:1315
457639, win 64, length 120
12:28:34.959108 IP 225.157.88.35.5106 > 10.0.0.2.http: Flags [S], seq 1054962716:1054
962736, win 64, length 120
```

Şekil 3-19 SYN Flood saldırısı anında gelen paketlerin gösterimi.

Şekil 3-19 'da görüldüğü üzere gelen paketlerin flag değerleri S olarak görülmektedir. Bu saldırganın farklı ip değerleri (spoofed ip) üzerinden yaptığı saldırı tcp dump üzerinde kaydedilmektedir. Oluşan bu pcap dosyası Wireshark (Şekil 3-20) ve tcpdstat programlarıyla incelenerek saldırının tipi kesin olarak belirlenir.



Şekil 3-20 SYN FLOOD saldırısının Wireshark üzerinde analizi.

```
mininet@mininet-vm:~$ tcpdstat synflood.pcap
DumpFile: synflood.pcap
FileSize: 111.29MB
Id: 201706071217
StartTime: Wed Jun 7 12:17:02 2017
EndTime: Wed Jun 7 12:28:43 2017
TotalTime: 700.93 seconds
TotalCapSize: 101.83MB CapLen: 11650 bytes
# of packets: 619531 (101.83MB)
AvgRate: 1.75Mbps stddev:0.47M PeakRate: 2.36Mbps

### IP flow (unique src/dst pair) Information ###
# of flows: 605981 (avg. 1.02 pkts/flow)
Top 10 big flow size (bytes/total in %):
0.6% 0.1% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0%

### IP address Information ###
# of IPv4 addresses: 605979
Top 10 bandwidth usage (bytes/total in %):
100.0% 0.7% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0%
### Packet Size Distribution (including MAC headers) ###
<<<<
[ 32- 63]: 12028
[ 64- 127]: 948
[ 128- 255]: 606418
[ 2048- 4095]: 104
[ 8192-16383]: 33
>>>>

### Protocol Breakdown ###
<<<<
protocol      packets      bytes      bytes/pkt
-----
[0] total      619531 (100.00%) 106780915 (100.00%) 172.36
[1] ip         607515 ( 98.06%) 106276243 ( 99.53%) 174.94
[2] tcp        607515 ( 98.06%) 106276243 ( 99.53%) 174.94
[3] ftpdata     21 ( 0.00%) 2214 ( 0.00%) 105.43
[3] ftp         9 ( 0.00%) 1566 ( 0.00%) 174.00
[3] ssh         9 ( 0.00%) 1566 ( 0.00%) 174.00
[3] telnet      9 ( 0.00%) 1566 ( 0.00%) 174.00
[3] smtp        9 ( 0.00%) 1566 ( 0.00%) 174.00
[3] name        9 ( 0.00%) 1566 ( 0.00%) 174.00
[3] dns         9 ( 0.00%) 1566 ( 0.00%) 174.00
[3] http(s)     384 ( 0.06%) 713191 ( 0.67%) 1857.27
[3] http(c)     606599 ( 97.91%) 105471924 ( 98.77%) 173.87
[3] kerb5       9 ( 0.00%) 1566 ( 0.00%) 174.00
[3] pop3       9 ( 0.00%) 1566 ( 0.00%) 174.00
```

Şekil 3-21 SYN Flood saldırısının tcpdstat üzerinde gösterilmesi.

Şekil 3-20 ‘ de görüldüğü üzere gelen paketler TCP protokolünü kullanıyor. Bu gelen paketlerin flag değerlerine bakıldığı zaman SYN Flood saldırısı olduğu net olarak söylenebilir.

Yine aynı şekilde tcpdstat üzerinden bakıldığında saldırısının %98,6 oranında TCP protokolünü kullandığı görülmüştür. Bu da yine SYN Flood saldırısı olduğunu gösterir. Çünkü SYN Flood TCP protokolünün altında yer alan bir saldırı türüdür.

Saldırı sonrası oluşturulan ağ çökmüştür. Bu yüzden saldırı sonrası bant genişliği ölçülememiştir. Aynı zamanda erişilebilirlik olarak bakıldığında bütün bağlantılar işlevsiz hale gelmiştir.

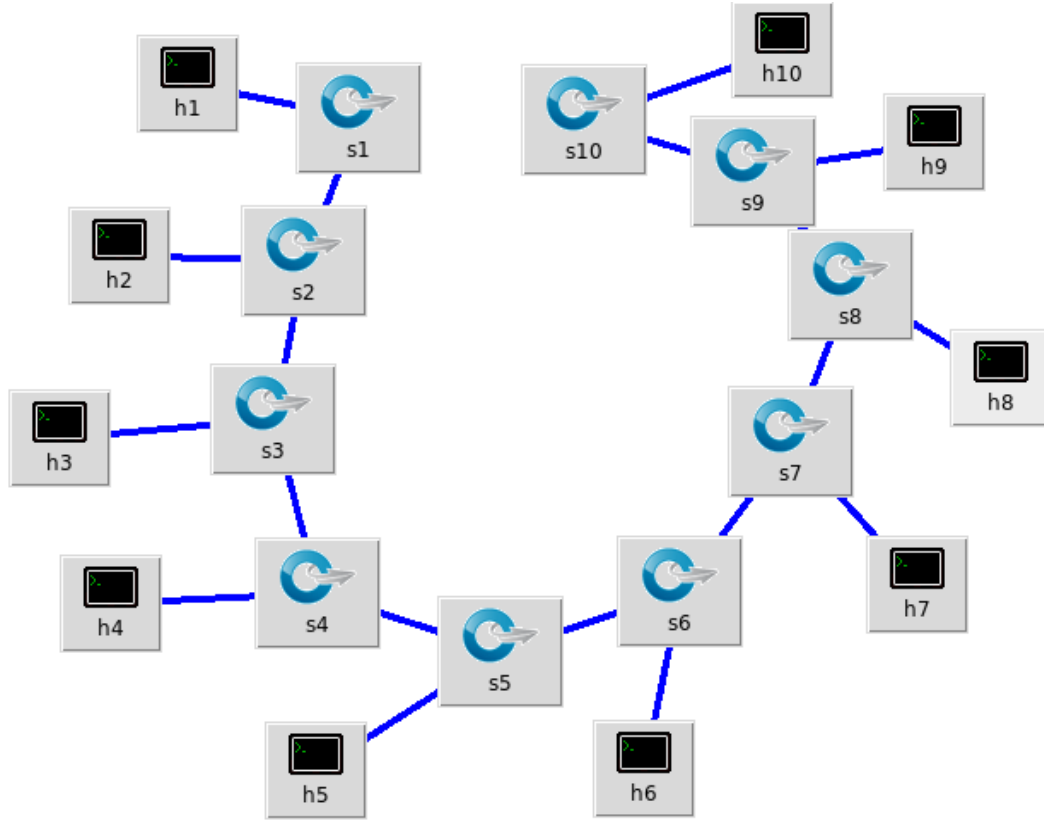
3.2.2.4. UDP Flood saldırıları

UDP bağlantısız bir protokoldür ve veri aktarmak için herhangi bir bağlantı kurulmasına ihtiyaç duymaz. UDP Flood saldırısı, saldırganın kurban sistemdeki rastlantısal bir bağlantı kapısına bir UDP paketi göndermesiyle başlar. Kurban, paketi aldığı anda hedef bağlantı noktasında hangi uygulamanın beklediğini belirleyecektir. Bağlantı noktasında bir uygulamanın beklemediğini fark ettiğinde, sahte adresteki kaynak için, ulaşılamaz hedef ICMP paketi oluşturacaktır. Böylesi tasarlanmış bir yöntemle kurbanın bağlantı noktalarına yeterince UDP paketi gönderilirse, sistem büyük olasılıkla gerçek istemcilere cevap veremeyecek hale gelecektir [18].

Örnek bir UDP Flood saldırısı;

```
*** Iperf: testing TCP bandwidth between h1 and h10
*** Results: ['15.5 Gbits/sec', '15.6 Gbits/sec']
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8 h9 h10
h2 -> h1 h3 h4 h5 h6 h7 h8 h9 h10
h3 -> h1 h2 h4 h5 h6 h7 h8 h9 h10
h4 -> h1 h2 h3 h5 h6 h7 h8 h9 h10
h5 -> h1 h2 h3 h4 h6 h7 h8 h9 h10
h6 -> h1 h2 h3 h4 h5 h7 h8 h9 h10
h7 -> h1 h2 h3 h4 h5 h6 h8 h9 h10
h8 -> h1 h2 h3 h4 h5 h6 h7 h9 h10
h9 -> h1 h2 h3 h4 h5 h6 h7 h8 h10
h10 -> h1 h2 h3 h4 h5 h6 h7 h8 h9
*** Results: 0% dropped (90/90 received)
mininet> links
h1-eth0<->s1-eth1 (OK OK)
h2-eth0<->s2-eth1 (OK OK)
h3-eth0<->s3-eth1 (OK OK)
h4-eth0<->s4-eth1 (OK OK)
h5-eth0<->s5-eth1 (OK OK)
h6-eth0<->s6-eth1 (OK OK)
h7-eth0<->s7-eth1 (OK OK)
h8-eth0<->s8-eth1 (OK OK)
h9-eth0<->s9-eth1 (OK OK)
h10-eth0<->s10-eth1 (OK OK)
s2-eth2<->s1-eth2 (OK OK)
s3-eth2<->s2-eth3 (OK OK)
s4-eth2<->s3-eth3 (OK OK)
s5-eth2<->s4-eth3 (OK OK)
s6-eth2<->s5-eth3 (OK OK)
s7-eth2<->s6-eth3 (OK OK)
s8-eth2<->s7-eth3 (OK OK)
s9-eth2<->s8-eth3 (OK OK)
s10-eth2<->s9-eth3 (OK OK)
mininet>
```

Şekil 3-22 UDP Flood saldırısı için oluşturulan topoloji (linear yani her host'un bir switch'i var).



Şekil 3-23 UDP Flood için kullanılan topolojinin görsel hali.

Saldırı h10 host'una yapılmıştır. Saldırıcı h8 ve h9 host'ları gerçekleştirmiştir.

```

DumpFile: udplinearFlood.pcap
FileSize: 4.01MB
Id: 201706062233
StartTime: Tue Jun 6 22:33:17 2017
EndTime: Tue Jun 6 22:43:00 2017
TotalTime: 583.61 seconds
TotalCapSize: 2.91MB CapLen: 98 bytes
# of packets: 72336 (2.91MB)
AvgRate: 41.46Kbps stddev:24.37K PeakRate: 356.72Kbps

### IP flow (unique src/dst pair) Information ###
# of flows: 24 (avg. 3014.00 pkts/flow)
Top 10 big flow size (bytes/total in %):
 63.7% 34.0% 0.5% 0.3% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0%

### IP address Information ###
# of IPv4 addresses: 10
Top 10 bandwidth usage (bytes/total in %):
 100.0% 64.9% 35.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0%
### Packet Size Distribution (including MAC headers) ###
<<<<
[ 32- 63]: 71972
[ 64- 127]: 364
>>>>

### Protocol Breakdown ###
<<<<

```

	protocol	packets	bytes	bytes/pkt
[0]	total	72336 (100.00%)	3049284 (100.00%)	42.15
[1]	ip	71319 (98.59%)	3006570 (98.60%)	42.16
[2]	udp	70955 (98.09%)	2980110 (97.73%)	42.00
[3]	kazaa	2 (0.00%)	84 (0.00%)	42.00
[3]	mssql-s	3 (0.00%)	126 (0.00%)	42.00
[3]	half lif	2 (0.00%)	84 (0.00%)	42.00
[3]	quake	3 (0.00%)	126 (0.00%)	42.00
[3]	other	70945 (98.08%)	2979690 (97.72%)	42.00
[2]	icmp	364 (0.50%)	26460 (0.87%)	72.69

```

>>>>

```

Şekil 3-24 UDP Flood tcp dump kayıtlarının incelenmesi.

Saldırının sonucu olarak tcpdstat kullanılarak bakıldığında saldırı UDP protokolü üzerinden gerçekleşmiştir. Bu saldırının UDP Flood olduğu söylenebilir.

```

nininet> pingall
*** Ping: testing ping reachability
n1 -> h2 h3 h4 h5 h6 h7 X X X
n2 -> h1 h3 h4 h5 h6 h7 X X X
n3 -> h1 h2 h4 h5 h6 h7 X X X
n4 -> h1 h2 h3 h5 h6 h7 X X X
n5 -> h1 h2 h3 h4 h6 h7 X X X
n6 -> h1 h2 h3 h4 h5 h7 X X X
n7 -> h1 h2 h3 h4 h5 h6 X X X
n8 -> X X X X X X X X X
n9 -> X X X X X X X X X
n10 -> X X X X X X X X X
*** Results: 53% dropped (42/90 received)

```

Şekil 3-25 Saldırı sonrası erişilebilirlik durumları.

Saldırı sonrasında h10 ile diğer hostlar arasındaki bağlantı kesilmiştir.

3.3.3. Uygulama Hedefli Saldırıları

Bu saldırı tipinde belirli uygulamalar hedef alınır. HTTP sunucuları üzerinde GET,POST istekleri gibi örneklenebilir. Bu saldırılar OSI 6. Ve 7. katmanlardadırlar.

Bu saldırının tespiti çok zordur. Çünkü kötü amaçlı istekleri normal isteklerden ayırmak zordur.

3.3.3.1. DNS Amplification saldırıları

Saldırı mantığı olarak smurf saldırılarına benzerler. Saldırgan sahte ip (spoofed ip) ile DNS sunucularına çok sayıda istek gönderir. Hedef bu isteklere cevap döner. Gönderilen istekler küçük paketler halinde listelenir. Böylece sunucunun vermek zorunda olduğu cevap istek paketinin 4-5 katı büyüklüğünde olur.

Kaynaktan yapılan çok sayıda sorgu hedef tarafından yanıtlanmak isteyince ağda bir trafik oluşacak ve hedef cevap veremez duruma gelecektir [19].



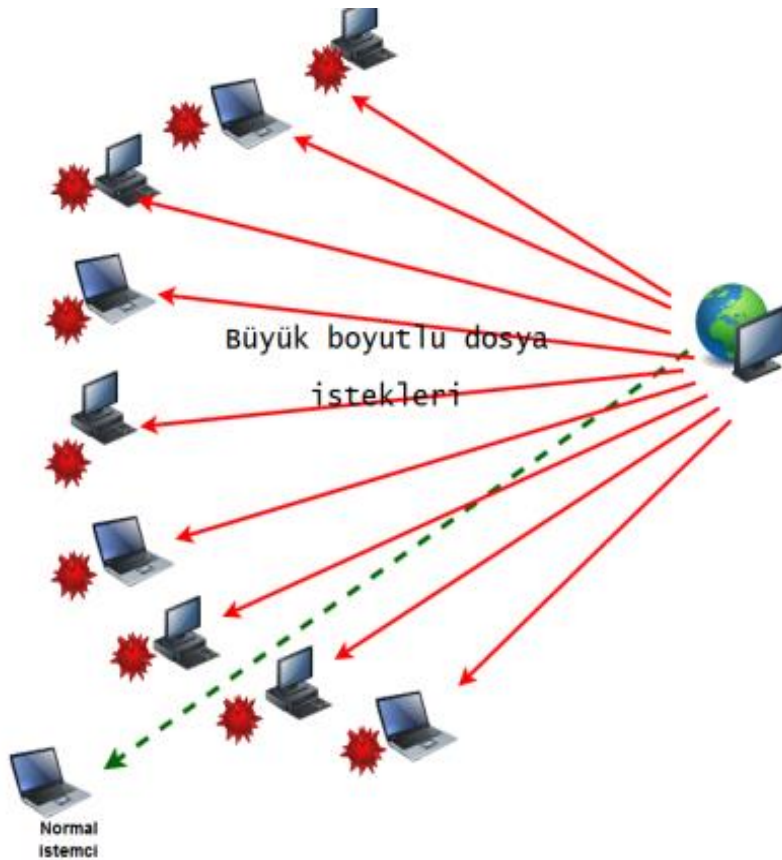
Şekil 3-26 DNS Amplification saldırı şeması [15].

3.3.3.2 HTTP GET-POST saldırıları

Bu saldırıda hedefin kullandığı uygulamaların açıkları hedeflenir. Sunucunun uygulama RAM ve CPU olarak bitirilmesini hedef alır.

HTTP flood saldırıları GET-POST istekleri kullanılarak volumetric attack (ses uygulamalarına saldırı) yapılır. Bu saldırılar OSI üzerinde layer7 de yapılır ve layer7 saldırıları özetlemek gerekirse;

- HTTP GET flood (http get saldırısı).
- HTTP bandwidth consumption (http kullanarak hat tüketimi).
- DNS query flood (DNS istek saldırısı).
- SIP INVITE flood (Ses uygulamalarına yönelik saldırılar) [20].



Şekil 3-27 HTTP GET istekleri [15].

Yapılan HTTP GET istekleri WireShark üzerinden takip edilebilir.

Topolojiler Mininet Python API kullanılarak da oluşturulabilir. HTTP GET saldırısı için oluşturulan topoloji aşağıdaki gibidir.

```
#!/usr/bin/python
import os
import threading

from mininet.topo import Topo
from mininet.net import Mininet
from mininet.node import CPULimitedHost
from mininet.link import TCLink
from mininet.util import dumpNodeConnections
from mininet.log import setLogLevel

class myClass():
    def h2thread(self):
        h2.cmd('./getFlood.sh')

    def h3thread(self):
        h2.cmd('./getFlood.sh')

class SingleSwitchTopo(Topo):
    "Single switch connected to n hosts."

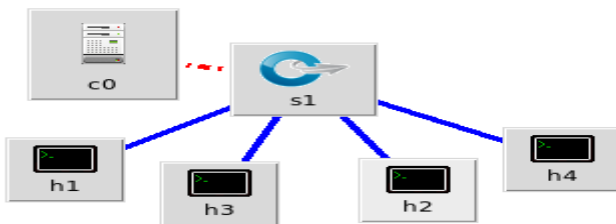
    def build(self, n=2):
        switch = self.addSwitch('s1')
        for h in range(n):
            # Each host gets 50% of system CPU
            host = self.addHost('h%s' % (h + 1),
                                cpu=.5 / n)

            # 10 Mbps, 5ms delay, 2% loss, 1000 packet queue
            self.addLink(host, switch, bw=1, delay='5ms', loss=2,
                          max_queue_size=1000, use_htb=True)

def perfTest():
    "Create network and run simple performance test"
    topo = SingleSwitchTopo(n=4)
    net = Mininet(topo=topo,
                  host=CPULimitedHost, link=TCLink)
    net.start()
    print
    "Dumping host connections"
    dumpNodeConnections(net.hosts)
    print
    "Testing network connectivity"
    net.pingAll()
    print
    "Saldiri Oncesi"
    h1, h4 = net.get('h1', 'h4')
    net.iperf((h1, h4))
    h1.cmd('tcpdump -n -w getflood.pcap')
    h1.cmd('sudo service apache2 start')
    Yep = myClass()
    thread = Thread(target=Yep.h2thread)
    thread2 = Thread(target=Yep.h3thread)
    thread.start()
    thread2.start()
    time.sleep(15)
    print
    "Saldiri Sonrasi h1 ,h4 arasi"
    h1, h4 = net.get('h1', 'h4')
    net.iperf((h1, h4))
    os.system("wget 10.0.0.1")

if __name__ == '__main__':
    setLogLevel('info')
    perfTest()
```

Şekil 3-28 Python API ile HTTP GET Flood için oluşturulan özel topoloji.



Şekil 3-29 Python API ile tanımlanan topolojinin görsel hali.

Bu saldırıda h2,h3 ile h1 host'u üzerine saldırı yapıldı ve bunun h1 h4 arası trafiğe etkisine bakıldı. Performans ölçümü öncesi ve sonrası için gösterildi.

```
*** Iperf: testing TCP bandwidth between h1 and h4
*** Results: ['12.3 Gbits/sec', '12.4 Gbits/sec']
mininet> xterm h1
mininet> xterm h1
mininet> iperf
*** Iperf: testing TCP bandwidth between h1 and h4
*** Results: ['2.90 Gbits/sec', '2.92 Gbits/sec']
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
```

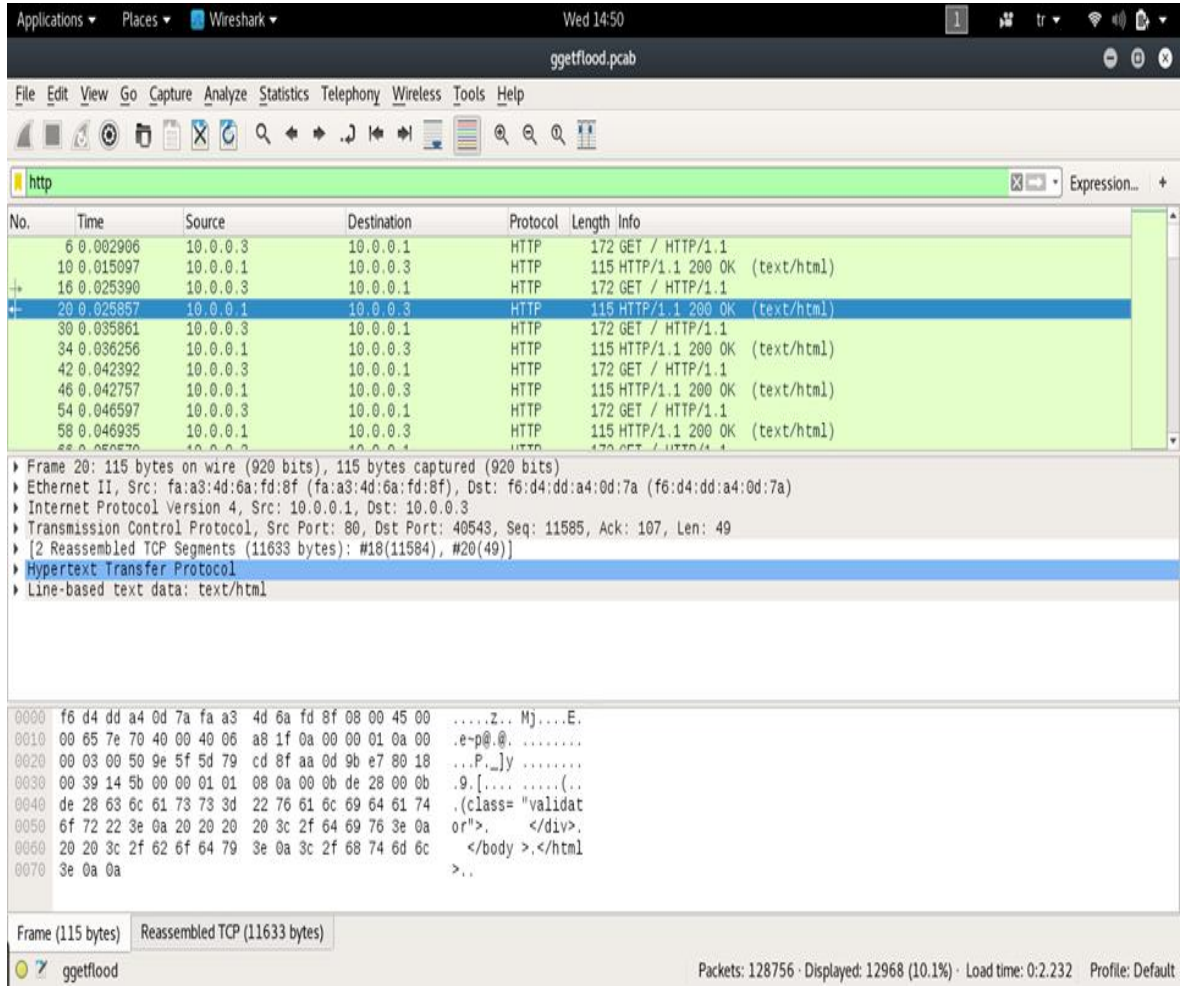
Şekil 3-30 HTTP GET Flood öncesi ve sonrası performans ölçümü.

Ağ üzerindeki performans bant genişliği ile doğru orantılı olup burada 12.3Gbit/sec olan bant genişliği saldırı ile beraber 2.90 Gbit/sec 'e düşmüştür.

```
### IP address Information ###
# of IPv4 addresses: 4
Top 10 bandwidth usage (bytes/total in %):
100.0% 84.0% 8.6% 7.3%
### Packet Size Distribution (including MAC headers) ###
<<<<
[ 32- 63]: 7
[ 64- 127]: 102681
[ 128- 255]: 6486
[ 256- 511]: 4
[ 512- 1023]: 16
[ 1024- 2047]: 6
[ 2048- 4095]: 11012
[ 4096- 8191]: 13
[ 8192-16383]: 3843
[16384-32767]: 285
[32768-65535]: 4403
>>>>

### Protocol Breakdown ###
<<<<
protocol          packets          bytes          bytes/pkt
-----
[0] total          128756 (100.00%) 365025270 (100.00%) 2835.02
[1] ip             128749 ( 99.99%) 365024976 (100.00%) 2835.17
[2] tcp            128737 ( 99.99%) 365023800 (100.00%) 2835.42
[3] http(s)         40772 ( 31.67%) 78164472 ( 21.41%) 1917.11
[3] http(c)         80615 ( 62.61%) 6061044 ( 1.66%) 75.19
[3] other           7350 ( 5.71%) 280798284 ( 76.93%) 38203.85
[2] icmp            12 ( 0.01%) 1176 ( 0.00%) 98.00
>>>>
```

Şekil 3-31 HTTP GET Flood saldırısının tcpdstat ile incelenmesi.



Şekil 3-32 HTTP GET Flood saldırısının Wireshark üzerinden takibi.

Saldırı altında olan sitelerin cevap verme süreleri daha uzundur.

4. SONUÇ

Veri merkezlerinde işlenen verilerin gün geçtikçe artması ile ihtiyaçların kesintisiz ve yerinde karşılanabilmesi için yazılım tanımlı ağlar kullanılmaktadır.

Tabi ki bu ağlar üzerinde baltalama girişimi olarak saldırılar da mevcuttur. Projemiz doğrultusunda yazılım tanımlı ağlar üzerinde gerçekleştirilen DDOS saldırıları üzerinde durulmuştur.

DDOS saldırılarının hedefi sunucuları sekteye uğratmaktır. Saldırıyı yapanın hedefi başta merkezi işlem birimi ve bellek kaynaklarıdır. Saldırı neticesinde cihazın yavaşlaması veya kapanması gibi durumlara yol açabilen dos saldırıları temel olarak ağda aksaklık yaratmayı ve ağı çökertmeyi baz alır.

Yapılan proje kapsam olarak DDOS saldırıları benzetim ve tespiti üzerinedir. Buna bağlı olarak bakıldığında öncelikle DDOS saldırısı nedir, nasıl yapılır, çeşitleri nelerdir gibi bilgiler üzerinde çalışmalar yapılmıştır. Farklı türde saldırılar hakkında araştırmalar ve incelemelerde bulunulmuştur. Saldırı öncesi ve sonrası durumların değerlendirmeleri için saldırıların kullanmış oldukları protokoller ve etki alanları ayrıntılı olarak incelenmiş ve buna bağlı olarak çıkarımlar yapıldı.

Çalışmaların devamında ise Mininet üzerinden sanal makine kullanarak ağ topolojileri oluşturulmuştur. Bu topolojiler oluşturulurken Mininet'in sağladığı hazır topolojiler ve python API kullanılmıştır. Farklı türde oluşturulan topolojilerde farklı türde saldırı çeşitleri analiz edilmiştir. Gerekli protokoller incelenmiş, gereksinimler taranmıştır.

Sonuçları gözle görebilmek ve daha iyi anlayabilmek adına örnek saldırı senaryoları oluşturulmuştur. Yapılan örnek saldırılar için topolojiler oluşturulup bu topolojiler üzerinde farklı saldırılar denenmiş ve ağ performansı ve oluşabilecek aksaklıklar değerlendirilmiştir. Yapılan saldırının benzetimi için bakıldığında uygulamalar üzerinde incelemeler yapılmıştır (tcpdump, Wireshark), sonuçlar değerlendirilmiştir.

KAYNAKLAR

- [1] Team, Mininet. "Mininet: An instant virtual network on your laptop (or other PC)." (2012).
- [2] İnternet Sitesi <http://www.openflow.org/wp/learnmore/>
- [3]İnternet Sitesi OpenFlow: The Next Generation Network interoperability, <http://www.bladenetwork.net/userfiles/file/OpenFlow-WP.pdf>, 2011.
- [4] Doktora Tezi, Yüksek Hızlı Ağlar İçin Zamanlama ve Anahtarlama Mimarilerinin Tasarımı ve Gerçeklenmesi, <http://www.mcu-turkey.com/wpcontent/uploads/2011/11/cpu-turkey-3.pdf>. s. 3, 2011.
- [5] <https://s3f.iti.illinois.edu/usrman/openflow.html>
- [6] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru M. Parulkar, Larry L. Peterson, Jennifer Rexford, Scott Shenker, Jonathan S. Turner, ACM SIGCOMM Computer Communication Review Volume 38, Number 2, April, s 70, 2008.
- [7] Jun Xu, Member, IEEE, and Mukesh Singhal, Fellow, Cost-Effective Flow Table Designs for High-Speed Routers: Architecture and Performance Evaluation, IEEE TRANSACTIONS ON COMPUTERS, VOL. 51, NO. 9, s. 1090, 2002.
- [8]https://www.opennetworking.org/index.php?option=com_content&view=category&id=46&Itemid=158&lang=en
- [9] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru M. Parulkar, Larry L. Peterson, Jennifer Rexford, Scott Shenker, Jonathan S. Turner, ACM SIGCOMM Computer Communication Review Volume 38, Number 2, s. 71, 2008.
- [10] Stefano Avallone, Alessio Botta, Alberto Dainotti, Walter de Donato, and Antonio Pescap'e , D-ITG V. 2.7.0-Beta2 Manual University of Napoli Federico II May 24, s 12, 2009.
- [11]Question & answer: openflow and software-defined networking, http://h17007.www1.hp.com/docs/openflow/openflow_faq.pdf.
- [12] İnternet Sitesi <http://www.karel.com.tr/bilgi/wireshark-nedir-nasil-kullanilir>

- [13] A. Hussain, J. Heidemann, C. Papadopoulos, “Distinguishing between Singnal and Multisource Attacks Using Signal Processing”, Computer Networks, vol. 46, 2004
- [14]<http://www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/table-contents-30/dos-attacks.html>
- [15] Koçaslan, Mehmet Düzgün. “DDOS saldırıları ve korunma yöntemleri üzerine simülasyon uygulamaları.”, Yüksek Lisans Tezi, Beykent Üniversitesi, 2015.
- [16] <http://www.rit.edu/~w-itsnew/07feb/botnet.html>
- [17] <http://www.cisco.com/c/en/us/about/security-center/guide-ddos-defense.html>
- [18] Ünal, Ahmet, “Bilişim Suç Türlerinden Biri Olan Dağıtık Servis Dışı Bırakma (DDOS) Saldırılarının Önlenmesindeki Hukuki ve Teknik Zorluklar”, Yüksek Lisans Programı, İstanbul Bilgi Üniversitesi, 2014.
- [19] Vaughn, R. and Evron, G., “DNS Amplification Attacks, A preliminary release”, March 17, 2006
- [20] <http://resources.infosecinstitute.com%2C16/09/2014>