Networked Software Systems Lab
Electrical Engineering | Technion

Technion
Israel Institute of Technology

Department of Electrical Engineering
Electronics
Computers
Communications

Vulnerabilities in IPv6
Project ID: 6778

Supervisor: Mordechai Hagiz

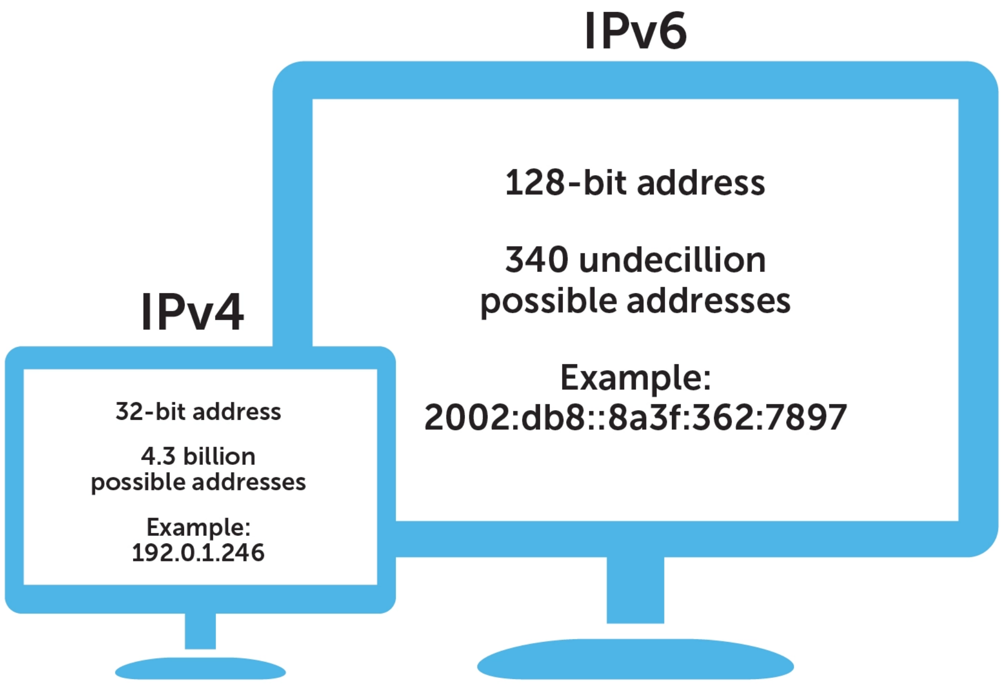
Students: Jamal Tannous
Salah Kadry

Table of Contents:

Table of Figures:

ABSTRACT

IPv6 is the most promising solution when it comes to dealing with the problem of the lack of **IPv4** addresses.
But as in the common case of new technologies, a lot of vulnerabilities and attack scenarios pop up.
In this project we will discuss some well known and not so well known attacks that target IPv6's vulnerabilities and propose some ways through which we can mitigate these attacks.
While working on the project we got to know some useful tools, platforms and libraries that helped us gain better knowledge regarding IPv6 and its vulnerabilities.


IPv6
128-bit address
340 undecillion possible addresses
Example: 2002:db8::8a3f:362:7897
IPv4
32-bit address
4.3 billion possible addresses
Example: 192.0.1.246

MOTIVATION

The advancement of Internet technologies has raised various security issues including the limitation of IP address space, due to drawbacks in the IPv4 protocol.
The most prominent solution proposed by IETF is IPv6.
The “new” protocol brought with it a lot of new features and advancements such as such as very large address space, simpler header format, autoconfiguration, builtin security features, as well as extensibility of IPv6 extension header.
But as in the common case with new technologies, unpredicted vulnerabilities pop up where the developers least expect them.
It is also worth mentioning that IPv6 inherited some of the vulnerabilities that were present in IPv4.
In the following pages, we will discuss the work that has been done in our project, mainly we will focus on attacks that target IPv6’s vulnerabilities.
We also propose ways in which we can mitigate these attacks.

<u>Discussing IPv6</u>

IPv6 is the next generation Internet Protocol (IP) standard intended to eventually replace IPv4, the protocol many Internet services still use today. Every computer, mobile phone, and any other device connected to the Internet needs a numerical IP address in order to communicate with other devices. The original IP address scheme, called IPv4, is running out of addresses.
IPv6 has been proposed in the beginning as a solution to the problem of the lack of sufficient IPv4 addresses in the world. But the new version of the internet is not solely an expansion of the address space.
Along the introduction of IPv6 came different protocols, address assignments, routing policies...
IPv6 also posed a gateway to new vulnerabilities, which can be exploited by newly crafted attacks, also some well known attacks from the previous version of the internet are yet to be dealt with.

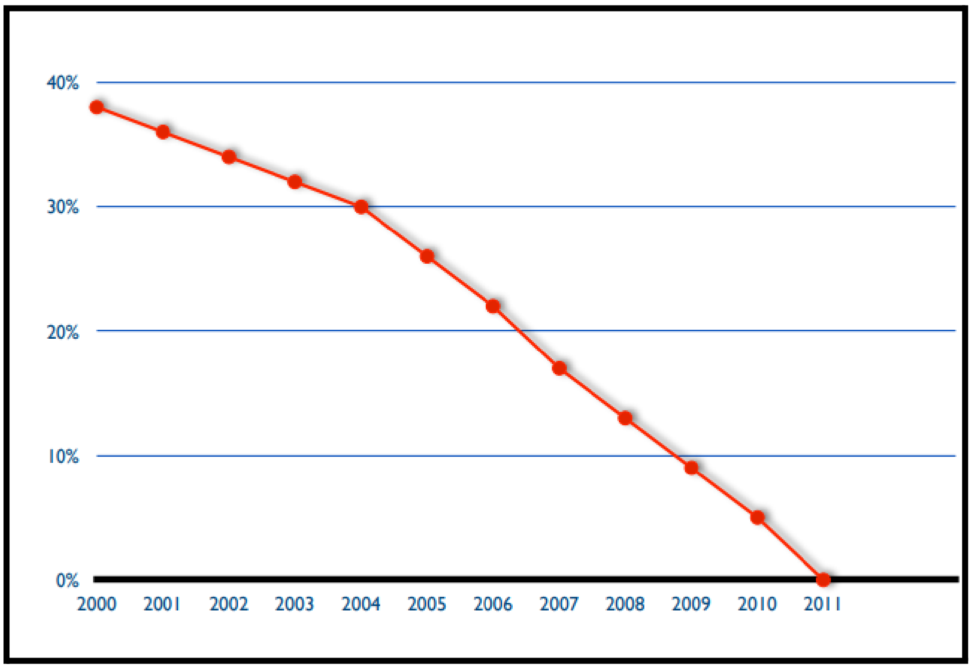


Figure 1: IANA IPv4 pool through the years

➢ The Structure of the IPv6 Protocol

We will shortly discuss the most important IPv6 features:

<u>Fixed header length:</u>

IPv6 header is twice as long (40 bytes) as IPv4 header without options (20 bytes).

Optional headers are daisy-chained.

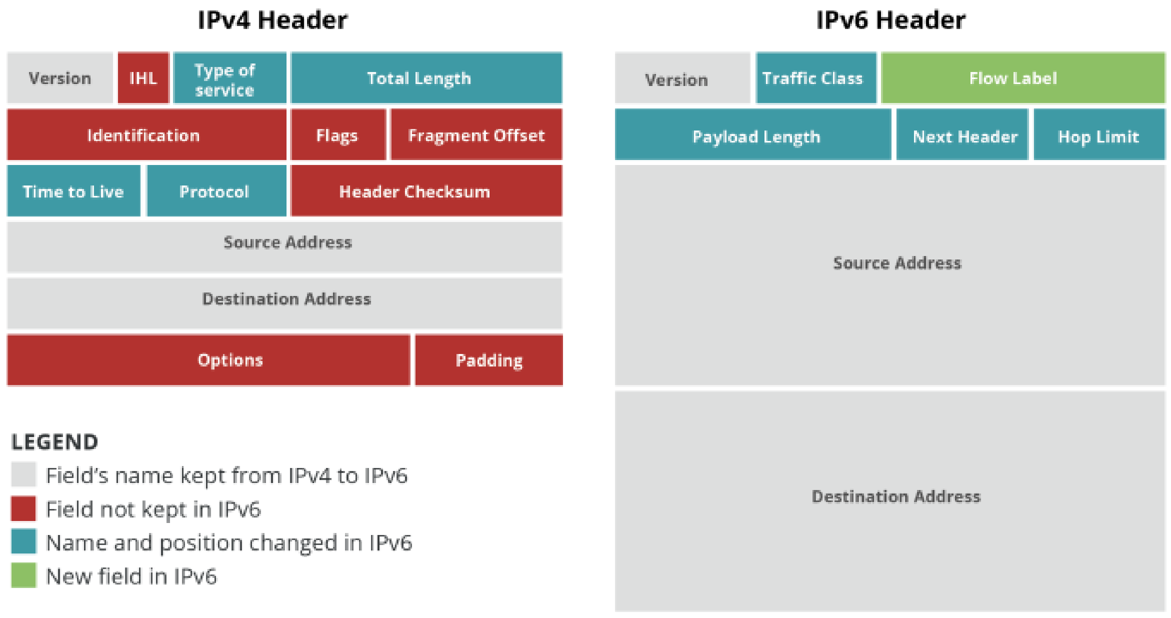


Figure 2: IPv4 header vs IPv6 Header

IPv6 introduces what is called "extension headers", this is a part of the IPv6 payload.

IPv6 extension headers contain supplementary information used by network devices (such as routers, switches, and endpoint hosts) to decide how to direct or process an IPv6 packet. The length of each extension header is an integer multiple of 8 octets. This allows subsequent extension headers to use 8-octet structures.

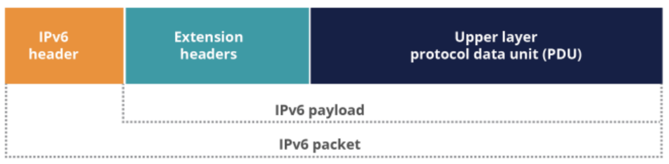


Figure 3: IPv6 Extension Header

FRAGMENTATION

Routers are no longer required to perform packet fragmentation and reassembly, resulting in dropped packets larger than the router's interface MTU. Instead, IPv6 hosts perform PMTU (Path MTU) to determine the maximum packet size for the entire path. When a packet hits an interface with a smaller MTU, the routers send back an ICMPv6 type 2 error, known as Packet Too Big, to the sending host. The sending host receives the error message, reduces the size of the sending packet, and tries again.

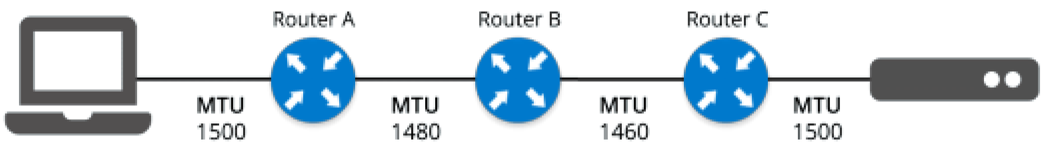


Figure 4: Path MTU Discovery

DHCPv6 (Dynamic Host Configuration Protocol)

DHCPv6 is a network protocol for configuring Internet Protocol version 6 (IPv6) hosts with IP addresses, IP prefixes, default route, local segment MTU, and other configuration data required to operate in an IPv6 network.

DHCPv6 runs between a client and a server. Similar to DHCP for IPv4, DHCPv6 clients and DHCPv6 servers exchange DHCPv6 packets using the User Datagram Protocol (UDP). In IPv6, packets cannot be broadcast; therefore, DHCPv6 uses multicast packets. In this case, DHCPv6 clients do not need to be configured with IPv6 addresses of DHCPv6 servers.

Among the things that it offers is SLAAC.

SLAAC (Stateless address autoconfiguration) - allows an IPv6-aware device to be plugged into a network, and given an IPv6 address prefix without manual configuration.

A normal DHCPv6 message exchange involves the following messages:

1. **Solicit** - sent by a DHCPv6 Client to locate DHCPv6 Servers.
2. **Advertise** - sent by a DHCPv6 server to a DHCPv6 Client in answer to the solicit message as an affirmative message that DHCPv6 Server services are available to a DHCPv6 Client.
3. **Request** - sent by a DHCPv6 Client to a DHCPv6 Server to request configuration parameters.
4. **Reply** - sent by a DHCPv6 Server to a DHCPv6 Client with configuration information.
5. **Renew** - sent by a DHCPv6 Client to a DHCPv6 Server requesting an extension to the address lifetime.

- An IPv6 address may be assigned to the DHCPv6 Client for a limited or unlimited time. If the address lifetime is limited, it has a preferred lifetime and a (generally longer) valid lifetime.

➢ The **IPv6** protocol provides a huge address space formed by 128-bit IPv6 addresses that require proper and efficient assignment and management policies.
Currently, the following methods are available to allocate IPv6 addresses:

- Manual configuration: You can manually configure IPv6 addresses, prefixes, and other network configuration params.
- Stateless address autoconfiguration: Hosts generate a link-local address based on the interface ID and automatically configure IPv6 addresses.
- Stateful autoconfiguration,DHCPv6 allocation has the following two methods:
    - DHCPv6 stateful autoconfiguration: DHCPv6 servers automatically provide IPv6 addresses.
    - DHCPv6 stateless autoconfiguration: IPv6 addresses are generated based on RA packets. A DHCPv6 server does not provide IPv6 addresses but provides other configuration parameters.

➢ **DHCPv6** involves the following roles:

- DHCPv6 client - A DHCPv6 client applies to a DHCPv6 server for IPv6 addresses, prefixes, and network configuration parameters to complete its address configuration.
- DHCPv6 relay - A DHCPv6 relay agent relays DHCPv6 packets between a DHCPv6 client and a DHCPv6 server to help the DHCPv6 client complete its address configuration.
- DHCPv6 server - A DHCPv6 server processes requests of address allocation, address lease extension, and address release from a DHCPv6 client or a DHCPv6 relay agent, and assigns IPv6 addresses and other network configuration parameters to the DHCPv6 client.

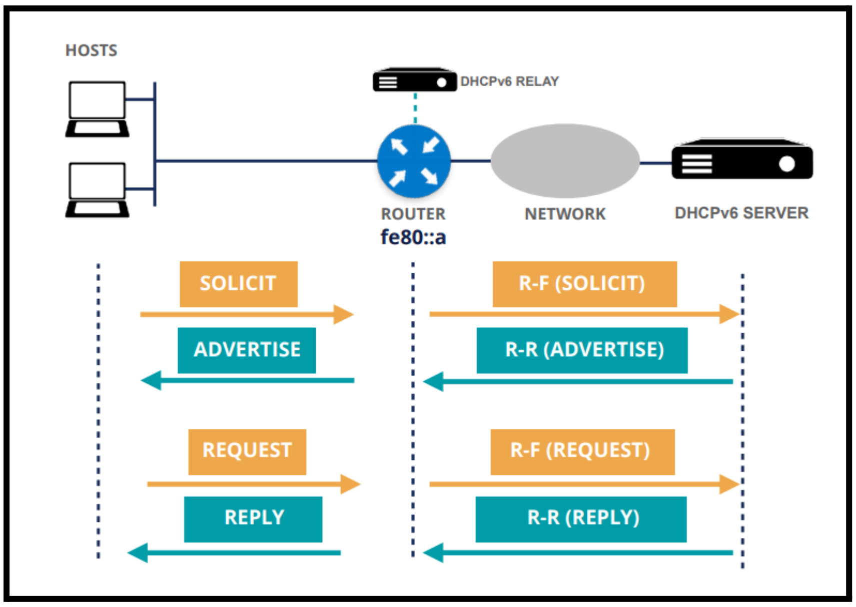


Figure 5: DHCPv6: Flow from Solicit until Reply

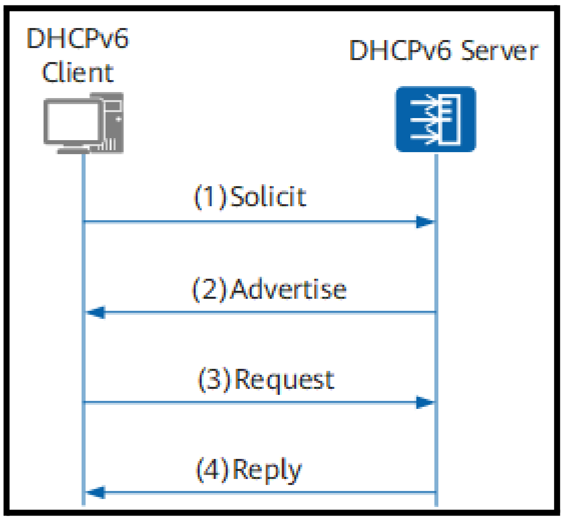
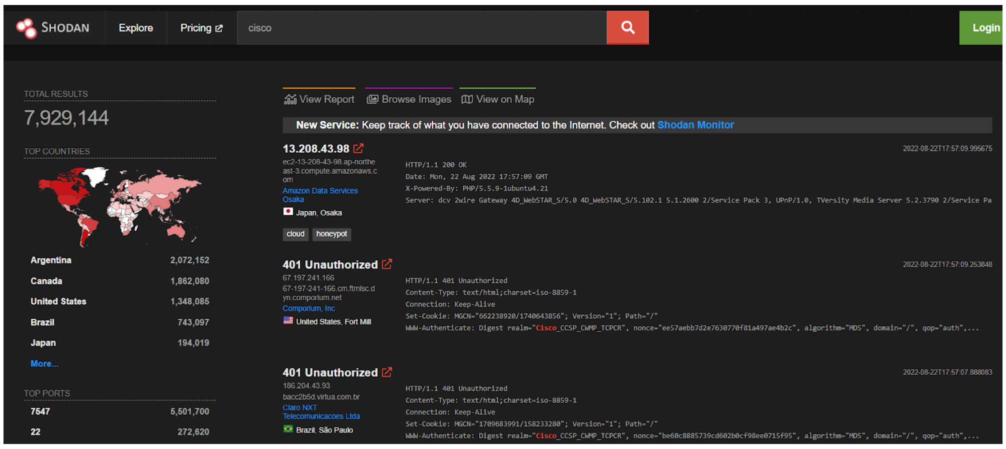


Figure 6: Process of address allocation using four-message exchange

<u>Project's Challenges</u>

The project provided us with a fair number of challenges.
First of all, the project deals with an enormous field that is still under development and far from being optimal.
IPv6 is harder and more complex than IPv4, that's why it's harder to find "good quality" studies and explanations regarding it, compare this to IPv4, where the academic courses are abundant and studies have been conducted for decades now, It's also harder to simulate attacks that exploit IPv6's vulnerabilities.
An example for the hardships in simulating an IPv6 attack is the lack of support Mininet provides, until this day Mininet does not support IPv6, that's why it wasn't possible to simulate attacks using Mininet.
Another interesting thing that we found is the fact that <u>shodan.io</u> has been crawling IPv6 for several years but until now it wasn't possible to search for specific IPv6 network ranges.
So in case shodan finds a way to search some specific IPv6 ranges, this could yield a whole new wave of possible attacks targeting IPv6 hosts.



TOTAL RESULTS
7,929,144

TOP COUNTRIES

Argentina	2,072,152
Canada	1,862,080
United States	1,348,085
Brazil	743,097
Japan	194,019
More...	

TOP PORTS

7547	5,501,700
22	272,620

New Service: Keep track of what you have connected to the Internet. Check out Shodan Monitor

13.208.43.98 2022-08-22T17:57:09.995675
ec2-13-208-43-98.ap-northeast-3.compute.amazonaws.com
Amazon Data Services Osaka
Japan, Osaka
cloud honeypot

```
HTTP/1.1 200 OK
Date: Mon, 22 Aug 2022 17:57:09 GMT
X-Powered-By: PHP/5.5.9-1ubuntu4.21
Server: dcv 2wire Gateway 4D_WebSTAR_S/5.0 4D_WebSTAR_S/5.102.1 5.1.2600 2/Service Pack 3, UPnP/1.0, TVersity Media Server 5.2.3790 2/Service Pa
```

401 Unauthorized 2022-08-22T17:57:09.253848
67.197.241.166
67-197-241-166.cm.ftmlsc.dyn.comporium.net
Comporium, Inc
United States, Fort Mill

```
HTTP/1.1 401 Unauthorized
Content-Type: text/html;charset=iso-8859-1
Connection: Keep-Alive
Set-Cookie: MGCN="662238920/1740643856"; Version="1"; Path="/"
WWW-Authenticate: Digest realm="Cisco_CCSP_CWMP_TCPCR", nonce="ee57aebb7d2e7630770f81a497ae4b2c", algorithm="MD5", domain="/", qop="auth",...
```

401 Unauthorized 2022-08-22T17:57:07.888083
186.204.43.93
bacc2b5d.virtua.com.br
Claro NXT Telecomunicacoes Ltda
Brazil, São Paulo

```
HTTP/1.1 401 Unauthorized
Content-Type: text/html;charset=iso-8859-1
Connection: Keep-Alive
Set-Cookie: MGCN="1709683991/158233280"; Version="1"; Path="/"
WWW-Authenticate: Digest realm="Cisco_CCSP_CWMP_TCPCR", nonce="be60c8885739cd602b0cf98ee0715f95", algorithm="MD5", domain="/", qop="auth",...
```

Figure 7: Example of shodan search results

Tools and references

While working on the project we encountered a variety of tools and looked up some very useful theoretical information from references and academic studies.
Here we present some of the tools that we used and explain shortly about them.

- **Scapy:**
  Scapy is a powerful interactive packet manipulation program. It is able to forge or decode packets of a wide number of protocols, send them on the wire, capture them, match requests and replies, and much more. It can easily handle most classical tasks like scanning, tracerouting, probing, unit tests, attacks or network discovery.
  It also performs very well at a lot of other specific tasks that most other tools can't handle, like sending invalid frames, injecting your own frames, combining techniques..
  We used Scapy to perform two attacks, namely MITM and TCP Syn Flood (which will be discussed further on in the project).
  These attacks were performed using Kali Linux based virtual machines, as one of the machines was the attacker and the others were the victims.

- **THC:**
  (The hacker's choice) is a group of international hackers.
  It does IT security work, which is publicly accessible.
  It also researches tools and academic papers to expose IT security tools and platforms that claim to be safe but actually aren't.
  THC also publishes tools to enhance the IT security movement.

In our project we specifically used THC's IPv6 attack toolkit.
In this toolkit there is a variety of attacks that exploit IPv6's weaknesses, ranging from "fake_router6" : an attack through which the attacker announces itself as a router on the network, with the highest priority, to "too_big6" : an attack through which the attacker manages to decrease the MTU in the network.

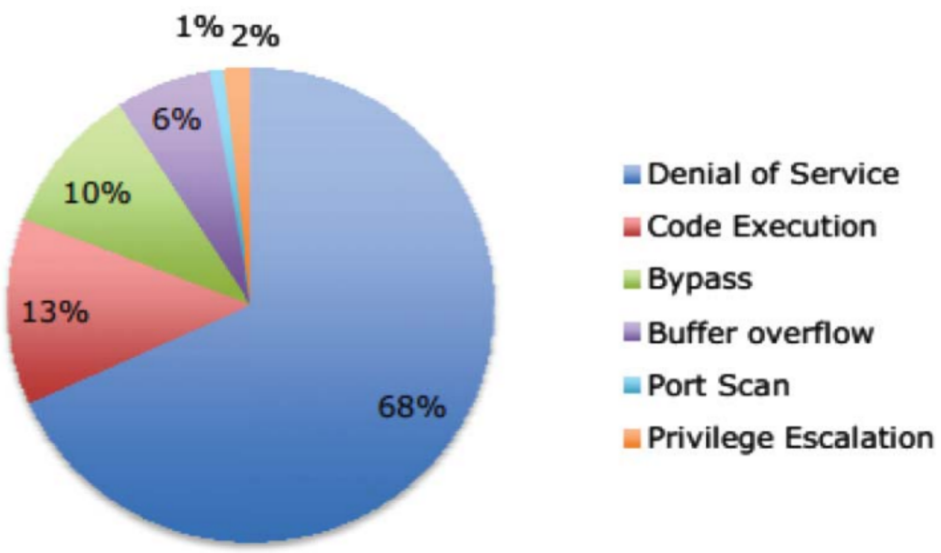


Figure 8: IPv6 Vulnerability Classes

In the coming pages, we will discuss some interesting attacks spanning over a variety of fields and vulnerability classes.
Some of these attacks are "classic" and well known, some of them also got inherited from IPv4.

TCP SYN Flood and SYN Cookies:

Let's first start with a classic DoS/DDoS attack - TCP SYN Flood.
This is a type of attack that exploits a weakness specifically in TCP protocol, namely by consuming resources on the targeted server and rendering it unresponsive through exploiting the three-way handshake process of the protocol.
A normal TCP three-way handshake procedure occurs in this manner:

1) Client requests connection by sending **SYN** (synchronize) message to the server.
2) Server acknowledges by sending **SYN-ACK** (synchronize-acknowledge) message back to the client.
3) Client responds with an **ACK** (acknowledge) message, and the connection is established.

A SYN flood attack is the procedure through which the attacker repeatedly sends SYN packets to every port on the targeted server (victim) (often using a fake IP address). The server, unaware of the attack, receives multiple, apparently legitimate requests to establish communication.

Essentially, with SYN flood DDoS, the offender sends TCP connection requests faster than the targeted machine can process them, causing network saturation.

It responds to each attempt with a SYN-ACK packet from each open port.

The attacker does not send the expected ACK, or it never receives the SYN-ACK if the IP was spoofed.

The damage is done in the fact that the server can't close the connection with the attacker, because it should wait for acknowledgement of its SYN-ACK packet for some time.

The server should wait a certain timeout until it can close the connection, but the problem is that during this timeout the attacker sends another SYN packet to the server, and thus it keeps the connection “half-open”.

As a result of this attack, the server’s connection overflow tables fill, service to legitimate clients will be denied, and the server may even malfunction or crash.

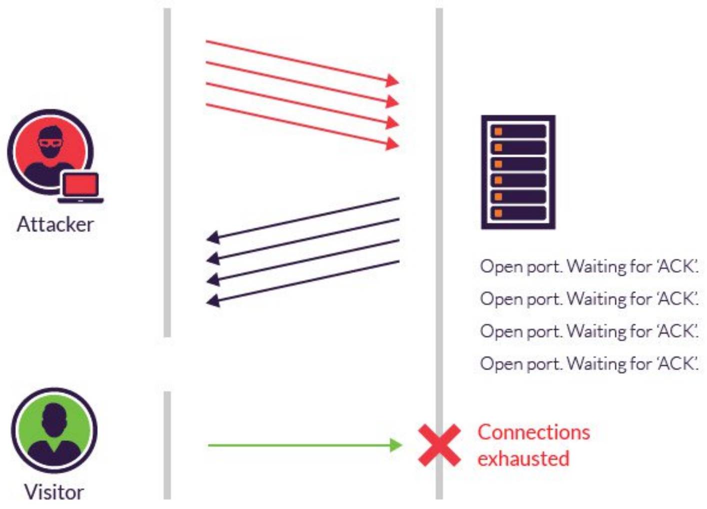

Attacker
Open port. Waiting for 'ACK'.
Open port. Waiting for 'ACK'.
Open port. Waiting for 'ACK'.
Open port. Waiting for 'ACK'.
Visitor
Connections exhausted

Figure 9: Illustration of TCP SYN Flood Attack

<u>Mitigation techniques</u>

There are several techniques for mitigating this attack, we will discuss **RST cookies**:

This technique is very simple yet very effective:

For the client’s first request, the server intentionally sends an <u>invalid SYN-ACK</u>. This should result in the client generating an RST packet. If the server indeed receives an RST packet from this client, then it’s of high probability that the client is legitimate and not an attacker, and as a result, the server will accept the request (log the client) and start accepting subsequent incoming connections from it.

Another mitigation technique is **TCP SYN Cookies**, which uses cryptographic hashing; the server sends a SYN-ACK along a sequence number consisting of the client's IP address, port number and possibly other unique identifying info. If the client sends the ACK along with this hash, then and only then, the server will allocate the necessary memory for the connection.

But where does IPv6 come into play?

For now, we've spoken about an attack that targets TCP. In other words, the attacker does not care whether the server is using TCP over IPv4 or IPv6.
But we found a very interesting encounter on Linux Mint forum stating scenarios with both IPV4/IPV6 TCP SYN Flood traffic using network simulation tools towards targets which have SYN Cookie enabled.
The results were as follows:

> Observed legitimate users able to access the target properly when there is IPv4 TCP SYN Flood attack from random sources. But when it comes to IPv6, the server was unreachable from legitimate users and it's CPU was high while there is IPv6 a TCP SYN Flood attack.
> This issue was for Linux Kernel version 4.19.81, and very recent, namely April 2022.

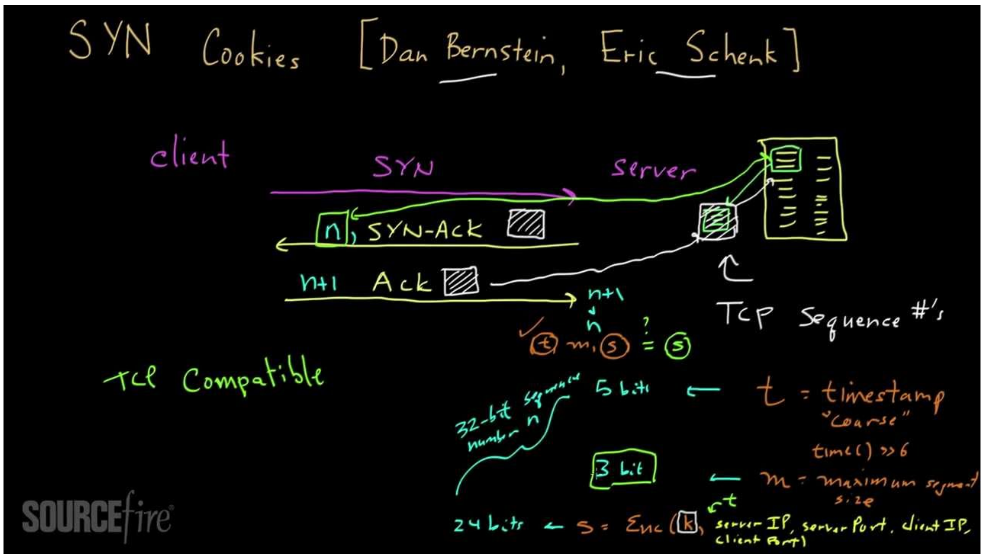


Figure 10: TCP SYN Cookies Illustration

Here is the link to our simulation with TCP Synflood while Syn_Cookies are off: Video

IPv6's Neighbor Discovery Protocol:

In order to clearly discuss some of the following attacks, firstly we need to present a new feature unique to IPv6 -
Neighbor Discovery Protocol.
This protocol presents one of the most significant differences between IPv4 and IPv6.
This is a mechanism through which new devices can connect to the network and get an IPv6 address assigned to them.
The ND protocol uses five types of ICMPv6 (Internet Control Message Protocol version 6).

These messages are: Router Solicitation (RS), Router Advertisement (RA), Neighbor Solicitation (NS), Neighbor Advertisement (NA), and Redirect.

Here is the flow, through which a new device can acquire an IPv6 address.

- RS is sent by IPv6 hosts to discover neighboring routers on an attached link.
- RA is sent by IPv6 routers periodically or in response to a RS message.
- NS is sent by IPv6 nodes to resolve a neighbor's IPv6 address to its link-layer address (MAC address) or to verify if an IPv6 node is still reachable.
- NA is sent by IPv6 nodes in response to a NS message or to propagate a link-layer address change.
- Redirect messages are sent by IPv6 routers to inform hosts of a better first-hop for a destination.

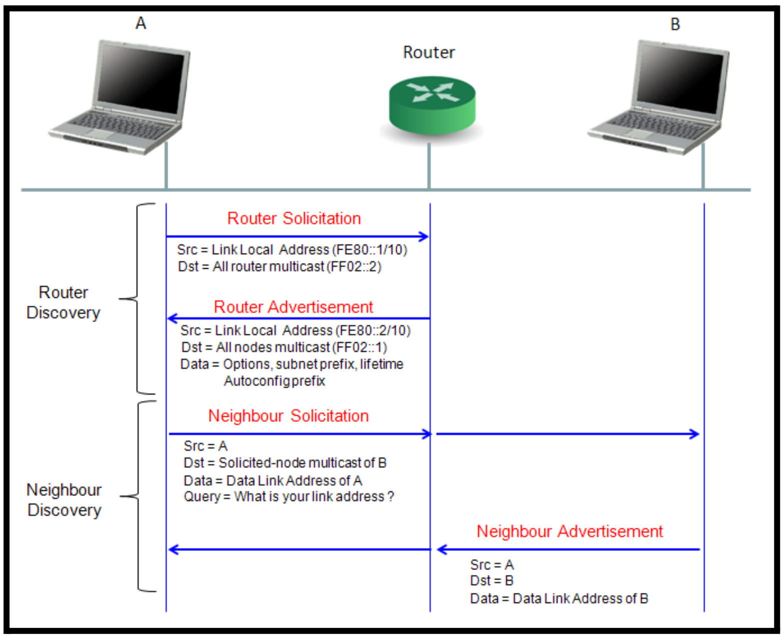

A
Router
B
Router Discovery
Router Solicitation
Src = Link Local Address (FE80::1/10)
Dst = All router multicast (FF02::2)
Router Advertisement
Src = Link Local Address (FE80::2/10)
Dst = All nodes multicast (FF02::1)
Data = Options, subnet prefix, lifetime
Autoconfig prefix
Neighbour Discovery
Neighbour Solicitation
Src = A
Dst = Solicited-node multicast of B
Data = Data Link Address of A
Query = What is your link address ?
Neighbour Advertisement
Src = A
Dst = B
Data = Data Link Address of B

Figure 11: Neighbor Discovery Protocol Flow

ND messages consist of an ND message header, ICMPv6 header, some ND message-specific data, and with zero or more ND options.

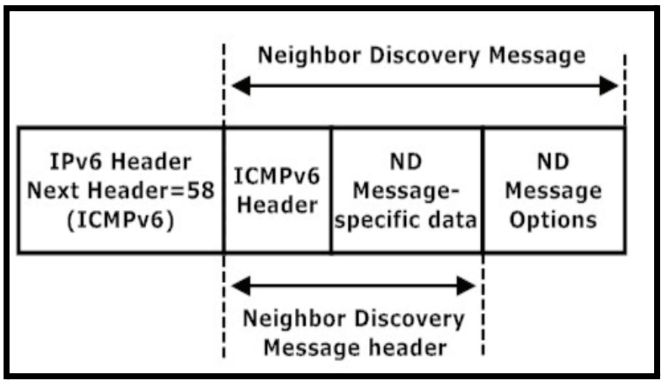

Neighbor Discovery Message
IPv6 Header Next Header=58 (ICMPv6)
ICMPv6 Header
ND Message-specific data
ND Message Options
Neighbor Discovery Message header

Figure 12: ND Message Struct

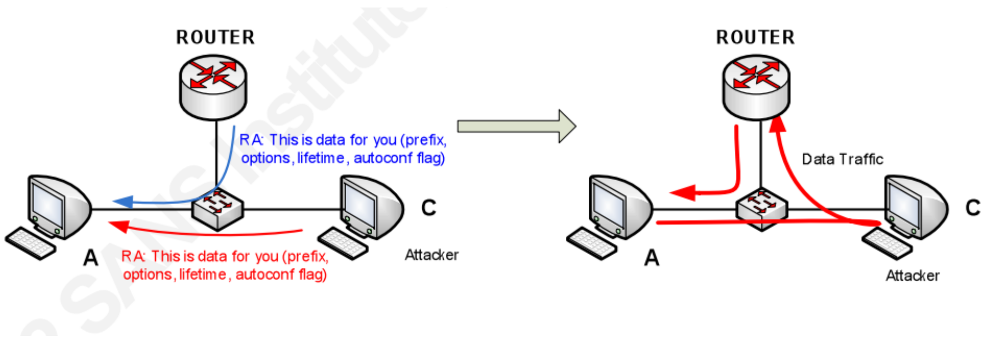
MITM with Spoofed ICMPv6 Router Advertisement:

- Neighbor Discovery (ND), is responsible for gathering various information required for network communication.

This is an attack that exploits a weakness in ND protocol.
One significant drawback of the ND protocol is the fact that it's a **"trusting"** protocol.
This means that each legitimate device connected on the network trusts that all other devices that are on the same network are also legitimate and don't intend to cause harm or violate their privacy.
This is where this specific attack comes into play, it exploits the "naiveness" of other devices to spoof the router advertisement and claim to be the router with the highest priority on the network, which redirects all of the traffic to pass through it and thus, having the ability to violate the privacy of other devices connected on the network.
Here is how the attack is performed: Attacker connects to the network.
Whenever the attacker hears a router advertisement on the network it also sends its own router advertisement, announcing itself as the router on the network with the highest priority, this causes all devices to reconfigure their routing tables and set their default gateway to the attacker (thinking that it's the router with the highest priority on the network).
The attacker on its side can redirect the flow passing through it to a router on the network (thus viewing all datagrams passing through it), or it can also throw each datagram it receives, rendering all of the devices on the network unable to exchange data with the internet.

➢ So this attack can be used in two different ways:

1) Violate the privacy of devices on the network.
2) Function as a type of DoS attack to render the network non functioning.



ROUTER
RA: This is data for you (prefix, options, lifetime, autoconf flag)
A
RA: This is data for you (prefix, options, lifetime, autoconf flag)
C
Attacker
ROUTER
Data Traffic
A
C
Attacker

Figure 13: MITM With Spoofed ICMPv6 Router Advertisement Illustration

Here is the link to our simulation of MITM With Spoofed ICMPv6 Router Advertisement: Video

Ways of mitigation

One very interesting mitigation strategy that we found is called **RA-Guard**.
This is a mitigation attack which uses a P4 switch together with a learning phase to determine legitimate router advertisement messages in order to help in denying a scenario in which the MITM with fake router advertisement attack occurs.
The system design consists of a control plane and a data plane, which communicate with a P4 Runtime API.
The main purpose of the control plane is to maintain the forwarding and filter tables and monitor incidents.

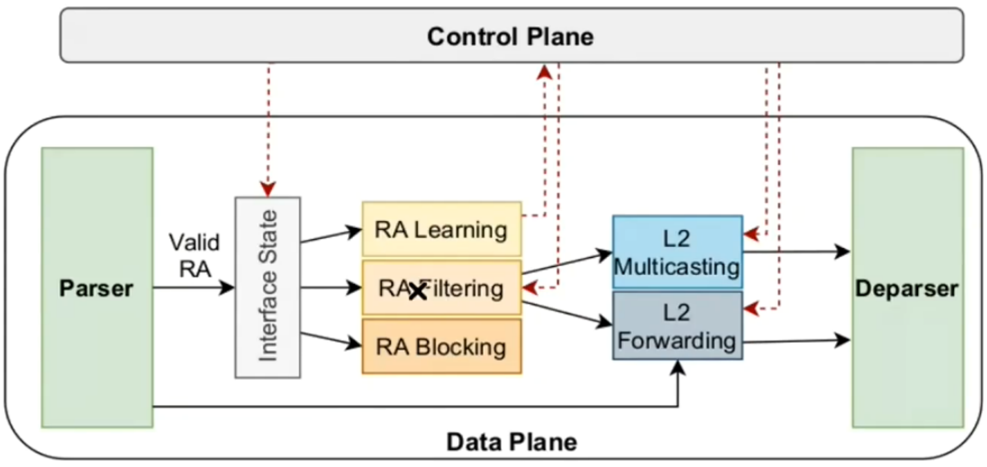


Figure 14: RA-Guard Design

This method filters fake RA (Router Advertisement) messages by detecting the RA extension header, in case the RA extension header are not fully in the ICMPv6 packet, this RA gets filtered.

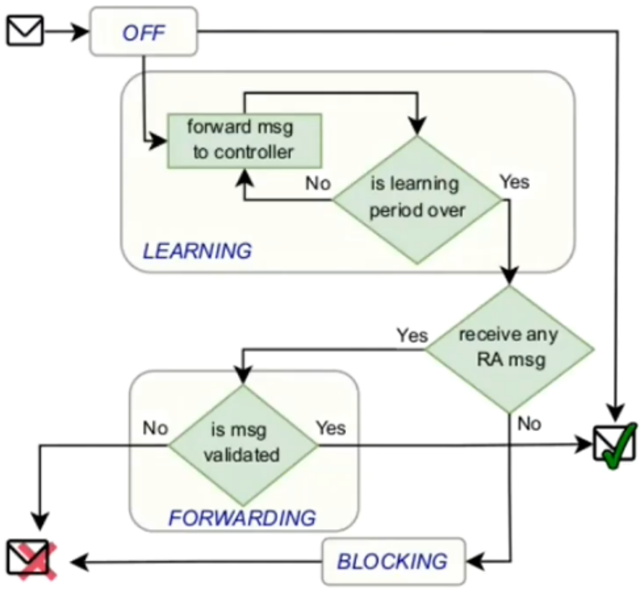


Figure 15: Flow of receiving an RA message

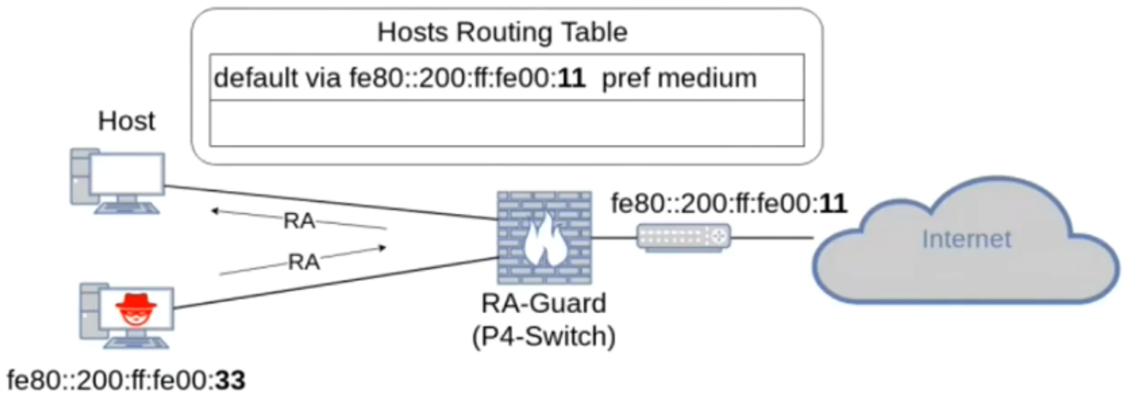


Figure 16: Attack scenario mitigation using RA-Guard

We haven’t tried this solution but in case you are curious, we have added it to our project’s codebase.

Security Issues in Duplicate Address Detection

A unique mechanism to IPv6 that helps in ensuring that each connected device on the network has its unique IPv6 address is **Duplicate Address Detection**.
Simply the procedure dictates that in order to verify that the desired IPv6 address is unique, a newly connected host must "make sure" that no device on the network has the desired IPv6 address, this can be described in the following diagram:

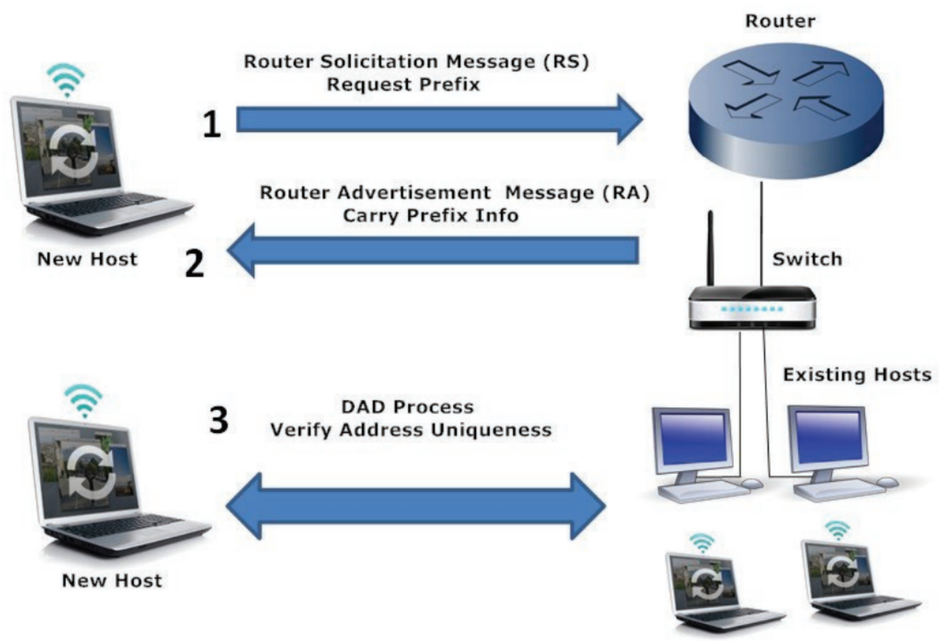

Router
Router Solicitation Message (RS)
Request Prefix
1
Router Advertisement Message (RA)
Carry Prefix Info
New Host
2
Switch
Existing Hosts
3
DAD Process
Verify Address Uniqueness
New Host

Figure 17: DAAD Protocol Flow

The problem is that During unique address verification process, Duplicate Address Detection (DAD) assumes that all the nodes on the local link are trusty to each other.
However, if the Neighbor Solicitation (NS) message is replied by a malicious node continuously, it will stop link local nodes to generate unique addresses thus will prevent them from address configuration.

Some attacks found in THC's IPv6 attacks:

We have found a lot of interesting attacks in THC's library, and we would like to briefly explain about "smurf6".
A smurf attacks is a DDoS attack through which the attacker carries the following steps:

1) Smurf malware is used to generate a fake Echo request containing a spoofed source IP, which is actually the target server address.
2) The request is sent to an intermediate IP broadcast network.
3) The request is transmitted to all of the network hosts on the network.
4) Each host sends an ICMPv6 response to the spoofed source address.
5) With enough ICMPv6 responses forwarded, the target server is brought down.

Turns out that this attack can be run very easily using ICMPv6.

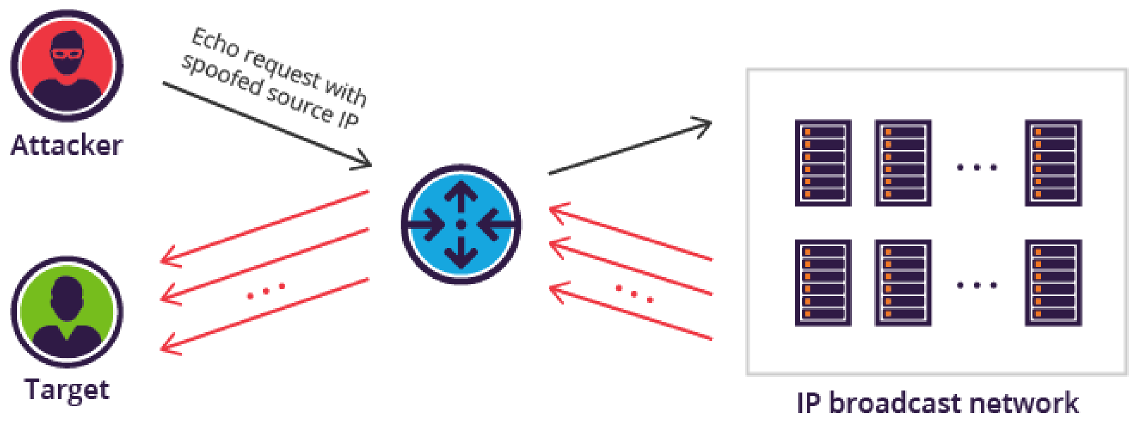


Figure 18: Smurf Attack

Copycat Attacks in IPv6 - Based Low Power and Lossy Networks:

This is one of the most interesting attacks that we found while working on the project.
This attack targets IoT devices which are connected over Low Power and Lossy Networks, which are arranged in a topology called DODAG (Destination Oriented Directed Acyclic Graph).
We found this attack to be extra interesting because of the fact that the widespread use of IoT devices was one of the main catalysts to integrating IPv6 as fast as possible.
In the following chapter we will shortly discuss specific terms and topics related to this attack.
We will also present a way of mitigation for this attack.

Low Power and Lossy Networks

Low-power and lossy networks (LLNs) are comprised of thousands of embedded networking devices that can be used in a variety of applications, such as smart grid automated metering infrastructures (AMIs) and wireless sensor networks. Connecting these LLNs to the Internet has even greater potential, leading to the emerging concept of the Internet of Things (IoT). With the goal of integrating LLNs into IoT, the IETF has recently standardized RPL and 6LoWPAN to allow the use of IPv6 on LLNs.
Low power and Lossy Networks (LLNs) consist of largely constrained nodes (with limited processing power, memory, and sometimes energy when they are battery operated or energy scavenging). These routers are interconnected by lossy links, typically supporting only low data rates, that are usually unstable with relatively low packet delivery rates. Another characteristic of such networks is that the traffic

patterns are not simply point-to-point, but in many cases point-to-multipoint or multipoint-to-point. Furthermore such networks may potentially comprise up to thousands of nodes.

RPL and DODAG

The functioning of the **RPL** is based on the construction of a Directed Acyclic Graph (DAG), which consists of one or more DODAGs (Destination Oriented DAGs), for each sink/root a DODAG.
The position of an individual node in a DODAG is determined by the rank of the node.
Routing and forwarding is implemented at the network layer, according to the IP architecture.
RPL routes are optimized for traffic to or from a root that acts as a sink/root for the topology.

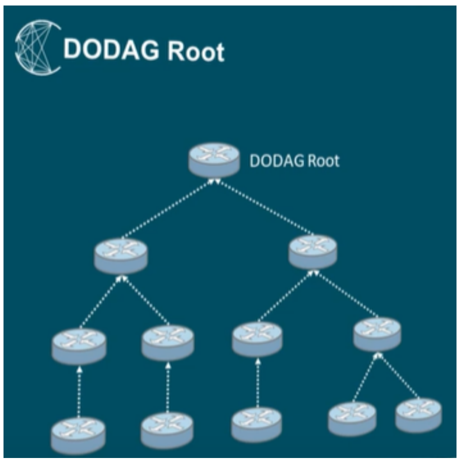


Figure 19: DODAG Example

RPL Control -messages:

There are 5 Control Messages, they form the Spanning tree:

1. DODAG information Object (DIO) : This message is Multicasted downwards . A given node In a DODAG may multicast this message , which lets other nodes know about it, about its state, and it announces other nodes "if they are interested to join , Please Let Me know. "
2. DODAG Information Solicitation (DIS) : When no announcement is heard, and if a node wants to join a DODAG it sends a control message , for that it wants to know If any DODAG exists. So the message which it sends is Like " Is there any DODAG ? "
3. DODAG advertisement Object (DAO) : It is a request sent by a child to parent or root. This message requests to allow the child to join a DODAG.
4. DAO-ACK : It is a response sent by a root or parent to the child , this response can either be a Yes or No.
5. Consistency check : deals with security.

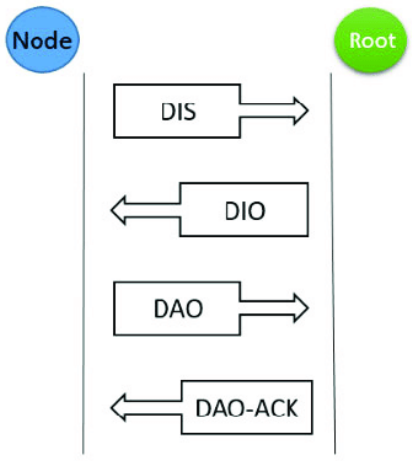


Figure 20: RPL Control Messages Example

<u>Copycat Attacks in IPv6-Based LLNs:</u>

This is where our attack scenario comes into play.

The main goal of the copycat attack is to degrade the routing performance of RPL based networks so that the Quality of Service (QoS) of real-time applications is affected.
Launching a Copycat attack:

1) Attacker listens to DIO messages from neighbors.
2) After capturing messages, the attacker re-sends those DIO messages again (multicast) (after modifying the source IP of ICMPv6 packet containing DIO message.) many times with a fixed replay interval.
3) The modification of the IP source address forces receiving (victim neighbors) nodes to believe that the packet is from a legitimate sender and makes them perform unnecessary routing related operations.

This kind of attack Leads in decrease in Packet Delivery Ratio (PDR) and increases Average End-to-End Delay (AE2ED) of the underlying network.

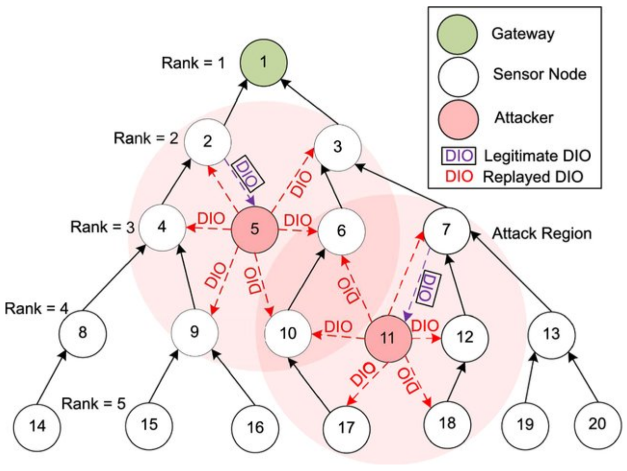


Figure 21: Copycat Attack Example

We simulated this attack using a C++ simulation (Program) that mimics the behavior and transportation of an RPL network.
There are two types of messages in the simulation: TCP and DIO messages.
We chose the following topology:

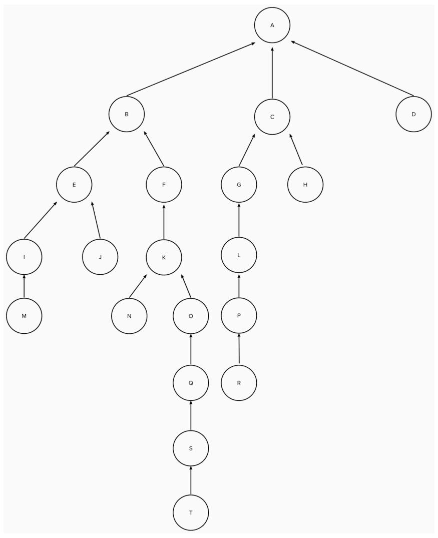


Figure 22: Simulation Network Topology

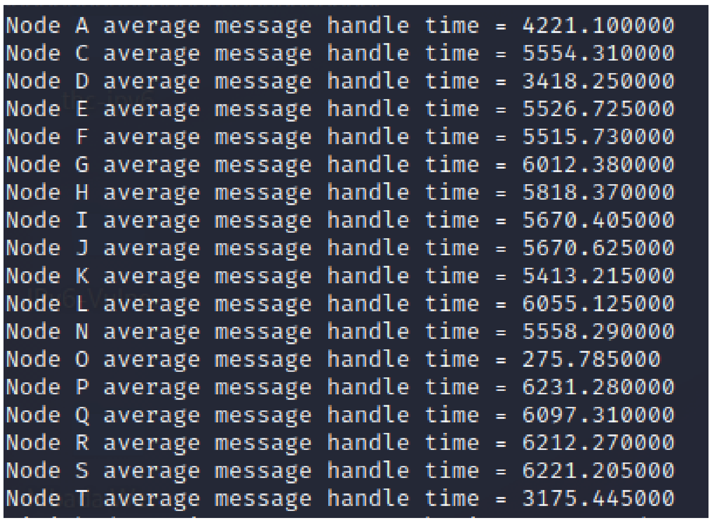
<u>**Discussing a mitigation method:**</u>

We noticed while running the simulations that the attacker node has a unique characteristic compared to the other nodes on the network. Namely the ratio between TCP messages and DIO messages is much lower in the attacker node than in regular nodes.
So we came up with a mitigation algorithm that goes recursively from the root to the children.
Each child chooses its child with the lowest ratio between TCP and DIO messages (in case the child is a root of a subtree, then it calls recursively), the suspect to be hacker is “bubbled” up to the parent and all the way until it reaches the root.

This mechanism is done multiple times in the simulation, until the attacker is found.
The attacker is marked and blocked after three consecutive checks that yielded the same suspicious node.
Now we present the results of our simulation.
We ran the simulation in three ways:

1) Regular simulation without a Copycat attack
2) Simulations with a Copycat attacking node which sends 5 consecutive DIO messages whenever it receives one (each simulation the attacker chooses a different parent).
3) Same simulations as (2) but with the mitigation method.

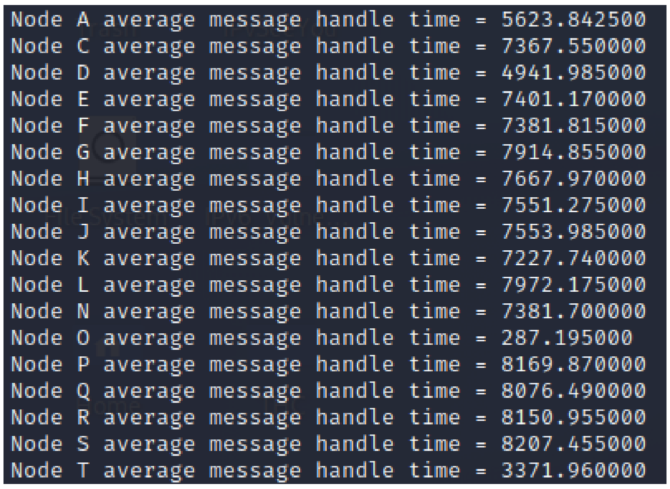
➢ **We got the following result for simulation (1):**



```
Node A average message handle time = 4221.100000
Node C average message handle time = 5554.310000
Node D average message handle time = 3418.250000
Node E average message handle time = 5526.725000
Node F average message handle time = 5515.730000
Node G average message handle time = 6012.380000
Node H average message handle time = 5818.370000
Node I average message handle time = 5670.405000
Node J average message handle time = 5670.625000
Node K average message handle time = 5413.215000
Node L average message handle time = 6055.125000
Node N average message handle time = 5558.290000
Node O average message handle time = 275.785000
Node P average message handle time = 6231.280000
Node Q average message handle time = 6097.310000
Node R average message handle time = 6212.270000
Node S average message handle time = 6221.205000
Node T average message handle time = 3175.445000
```

Figure 23: Results for Simulation 1

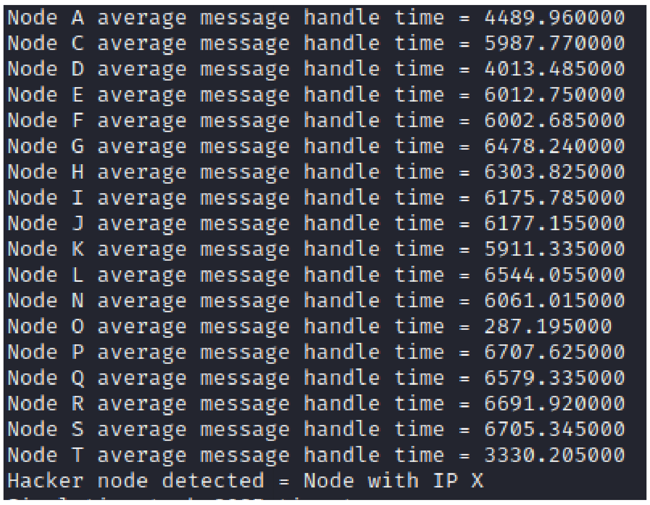
➢ **We got the following result for simulation (2) with attacker as a direct child of the root:**



```
Node A average message handle time = 5623.842500
Node C average message handle time = 7367.550000
Node D average message handle time = 4941.985000
Node E average message handle time = 7401.170000
Node F average message handle time = 7381.815000
Node G average message handle time = 7914.855000
Node H average message handle time = 7667.970000
Node I average message handle time = 7551.275000
Node J average message handle time = 7553.985000
Node K average message handle time = 7227.740000
Node L average message handle time = 7972.175000
Node N average message handle time = 7381.700000
Node O average message handle time = 287.195000
Node P average message handle time = 8169.870000
Node Q average message handle time = 8076.490000
Node R average message handle time = 8150.955000
Node S average message handle time = 8207.455000
Node T average message handle time = 3371.960000
```

Figure 24: Results for Simulation 2

➢ **We got the following result for simulation (3) with attacker as a direct child of the root:**



```
Node A average message handle time = 4489.960000
Node C average message handle time = 5987.770000
Node D average message handle time = 4013.485000
Node E average message handle time = 6012.750000
Node F average message handle time = 6002.685000
Node G average message handle time = 6478.240000
Node H average message handle time = 6303.825000
Node I average message handle time = 6175.785000
Node J average message handle time = 6177.155000
Node K average message handle time = 5911.335000
Node L average message handle time = 6544.055000
Node N average message handle time = 6061.015000
Node O average message handle time = 287.195000
Node P average message handle time = 6707.625000
Node Q average message handle time = 6579.335000
Node R average message handle time = 6691.920000
Node S average message handle time = 6705.345000
Node T average message handle time = 3330.205000
Hacker node detected = Node with IP X
```

Figure 25: Results for Simulation 3

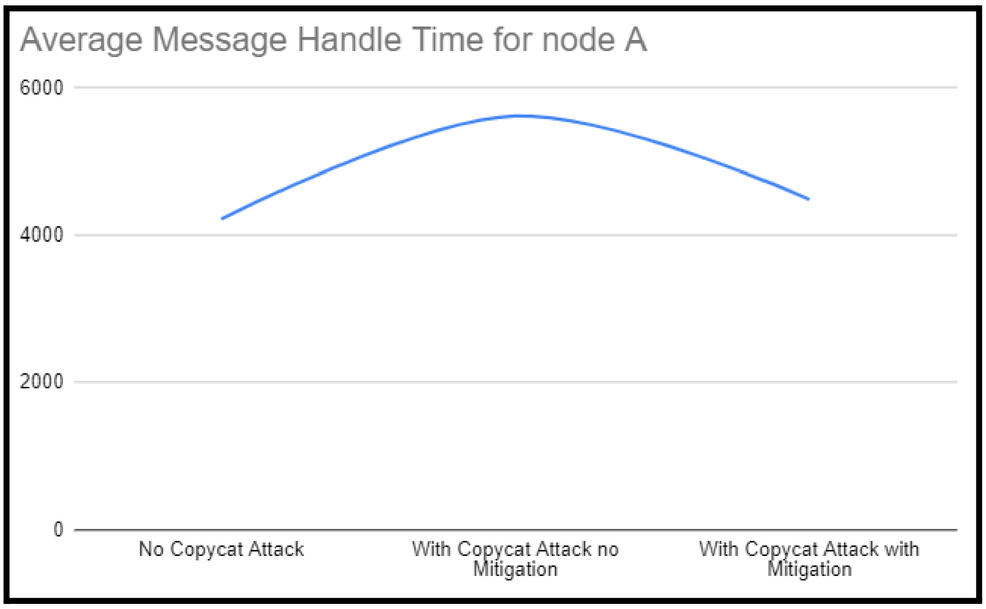


Figure 25: Average message handle time for node A

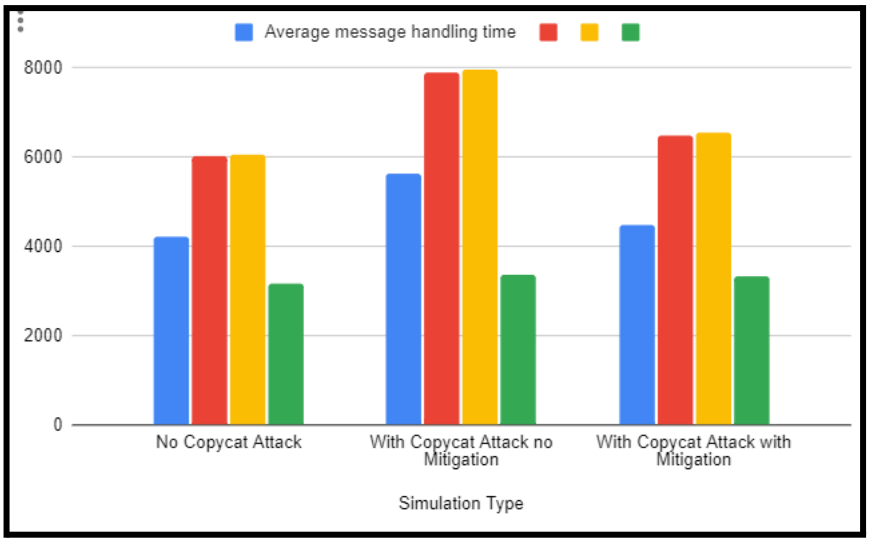


Figure 26: Average message handle time for different nodes

We can clearly see from the results that the mitigation mechanism works very efficiently in excluding the attacker node from the network and finding it as the outlier in the topology.

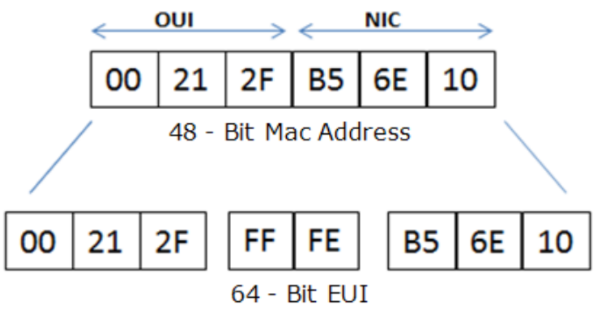
IPvSeeYou: Exploiting Leaked Identifiers in IPv6 for Street-Level Geolocation

This is an attack that targets residential low cost routers that are deployed widely.
These routers use a form of legacy IPv6 addressing.
It turns out that anyone using ping6 or traceroute6 can exploit this addressing weakness and find these routers' physical geolocation with street level precision.
In this chapter we will discuss some relevant terms and topics related to this attack and describe how the attack is performed.

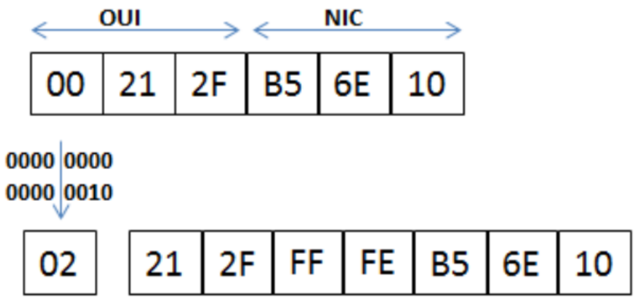
IPv6 EUI-64 Bit Address

Extended Unique Identifier (EUI) allows a host to assign itself a unique 64-Bit IP Version 6 interface identifier (EUI-64). This feature is a key benefit over IPv4 as it eliminates the need of manual configuration or DHCP as in the world of IPv4. The IPv6 EUI-64 format address is obtained through the 48-bit MAC address.
The MAC address is first separated into two 24-bits, with one being OUI (Organizationally Unique Identifier) and the other being NIC specific. The 16-bit oxFFFE is then inserted between these two 24-bits for the 64-bit EUI address.

Here is an example showing how a MAC Address is used to generate EUI.



Next, the seventh bit from the left, or the universal/local (U/L) bit, needs to be inverted. This bit identifies whether this interface identifier is universally or locally administered. If 0, the address is locally administered and if 1, the address is globally unique.



Once the above is done, we have a fully functional EUI-64 format address.

It is worth mentioning that this kind of address allocation exposes the MAC address of the device which on itself poses an array of hazard and possible vulnerabilities.

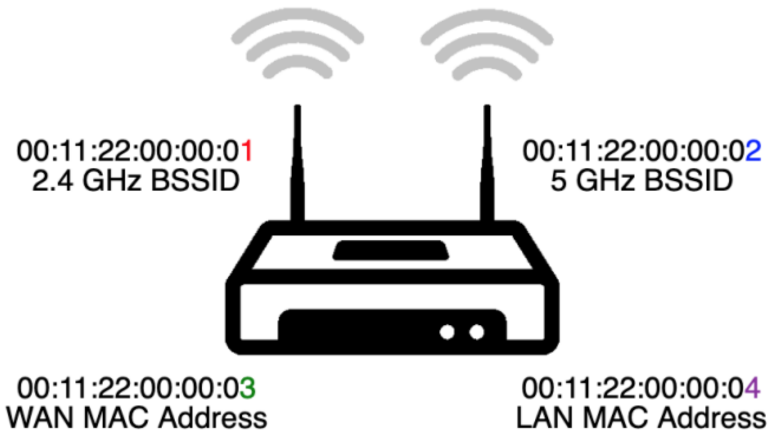
Mapping WAN MAC to WiFI BSSID

Most of the cheap customer premise equipment (CPE) routers are all-in-one CPE devices, e.g. cable modem with built-in WiFi.

Many of them are System-on-a-Chip (SoC) designs where all radios are made by one company.

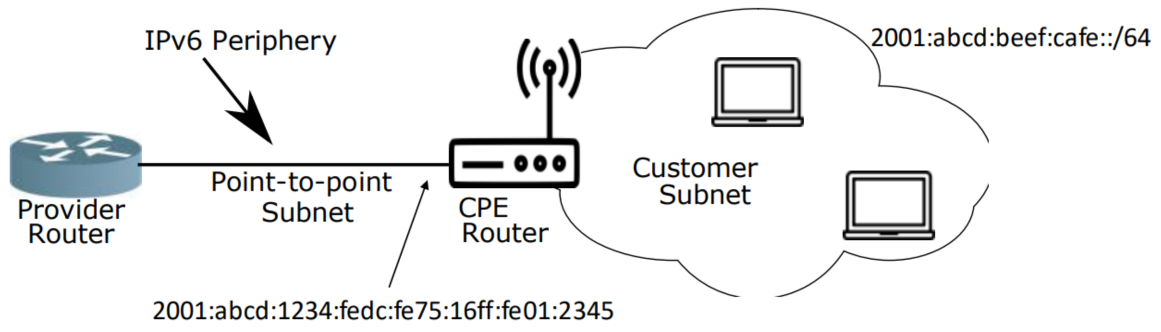
Each interface gets its own MAC address.

These MAC addresses are normally related in these devices, for example, +/- 1.



This fact would prove to be of great significance to the attack which we will discuss.

<u>The IPvSeeYou attack</u> combines EU-I64 addresses with WiFi geolocation data.
Most cheap routers allocate their IPv6 addresses using EUI-64, these routers are called customer premise equipment (CPE) routers.



We know that the smallest IPv6 address is a /64,
so in case we traceroute6 to a random address inside a network, most likely that the target won't exist.
But in this attack we are not actually attempting to hit the target using traceroute6, but we are hoping that one of its packet reaches the CPE router as its hop limit expires, which causes the CPE router to send and "ICMPv6 time exceeded message", and hopefully this CPE router will have an EUI-64 address.

In order to implement the attack efficiently, a corpus of >60 millions EUI-64 derived WAN addresses through active probing of random IPv6 addresses.
The other data source that was used/gathered is the contents of WiFi geolocation services.

These services map a WiFi BSSID to its corresponding geolocation.
The attack amassed a corpus of about 450 million BSSID-geolocation pairs.

The question that comes is how to combine these two separate data sources to the same device!

And here the above mentioned relation between the related MAC addresses of different interfaces in cheap CPE routers comes into play.
Because the EUI-64 address includes the WAN MAC address in it, then through extracting it and modifying it slightly, we can get the BSSID address which then can be queried in the corpus which maps the BSSIDs to geolocations.

And thus the attack can be done as follows:

1) Traceroute6 to a random IPv6 address and hope that an "ICMPv6 time exceeded message", and hopefully this CPE router will have an EUI-64 address.
2) Lookup the CPE router using IPvSeeYou tool which matches the EUI-64 address to a possible BSSID and searches for this BSSID's geolocation.

Here is an example of an attack:

1) Solicited volunteers with CPE using EUI-64 IPv6 addresses (They divulged internal subnet,eg 2003:ab::/56).
2) We traceroute to a random address in an internal network and obtained WAN EUI-64 IPv6 address.
3) Used IPvSeeYou to infer BSSID and geolocate IP addresses.

4 out of 5 volunteer devices geolocated < 50m geolocation accuracy. 5th device used EUI-64 IPv6 addressing but non-sequential WAN/BSSID MACs

The optimal mitigation mechanism is to stop manufacturing CPE routers that use EUI-64 IPv6 addresses.

Simulation of the attack can be found here: Video

Conclusion:

In this project we have dealt with a “hot” and fascinating topic that has significant effects on our day to day life.
The topic of IPv6 is a hugely elaborate one and is far from being sufficiently explored in terms of weaknesses and features.
In our project we managed to have a peek into this field and learn a lot of new topics.
Our study was heavily focused on the weaknesses in IPv6, and we found them to be plenty.
We got to know new vulnerabilities unique to IPv6 and others that got inherited from IPv4.
We also managed to explore two of the most fascinating vulnerabilities:

1) The first attack is Copycat Attacks on RPL, this was extra interesting because IoT devices were the main catalysts to the rapid integration of IPv6.
   We managed to propose and simulate a mitigation technique that proved to be very effective in reducing the effects of the attack.
2) The other attack that we discussed was IPvSeeYou, which targets the weaknesses of CPE routers found on most households today, which makes this vulnerability very severe and the solution to it very urgent.

Overall, we found this project very enriching and beneficial to our degree and to our fields of study.

Bibliography

Knowledge in IPv6:

1) ripe.net's Basic IPv6 Training Course
2) IPv6 Flow-Based Processing Overview
3) DHCPV6 Feature Overview Guide
4) Shodan blog
5) THC IPv6 Attack Toolkit
6) Scapy
7) RPL and DODAG for Low Power and Lossy Networks
8) TCP Over RPL

Vulnerabilities and attacks in IPv6:

1) Imperva's SYN Flood
2) Imperva's Anycast Defense Mechanism
3) Linux Mint's Forum Regarding IPv6 SYN Flood Attack
4) SANS: IPv6 MITM via fake router advertisements
5) IPv6 Security: Attacks and Countermeasures in a Nutshell
6) IPv6 MITM with RA Spoofing Attack and Mitigation Using P4
7) CURRENT STATE OF IPV6 SECURITY IN IOT
8) Understanding IPv6's EUI-64
9) IPvSeeYou: Exploiting Leaked Identifiers in IPv6 for Street-Level Geolocation
10) IPvSeeYou BlackHat Conference
11) IPvSeeYou Github
12) IPvSeeYou Youtube Video
13) New Ways of IPv6 Scanning
14) New ways of IPv6 Scanning Youtube Video
15) DODAG Youtube lecture
16) Smurf Attack