Practical Network Defense notes

Matteo Salvino

Contents

1 Network 101 **3**

2 Traffic monitoring **5**

3 IPv6 introduction **8**

3.1 Notation and types 9

3.1.1 Unicast addresses 10

3.1.2 Multicast addresses 12

3.2 Addressing 13

3.2.1 Prefix Delegation process 15

3.3 Comparing the IPv4 and IPv6 headers 16

3.4 Link-local attacks 18

1 Network 101

Internet is in essence a network of networks with a hierarchical structure. The path into the Internet backbone could be wired or wireless. The backbone itself consists of global Internet Service Providers (ISPs) and several regional ISPs that are all interconnected to provide a path from sender to receiver. The communication path may typically contain a variety of switches and routers that facilitate and direct the flow of information through the network. The communication links, regardless of whether they are wired or wireless, are defined by a transmission rate and bandwidth. Access networks are used to connect a host or Local Area Network (LAN) to the Internet. Routers connect LANs, generate routing tables and forward packets of data on their path from source to destination. The Internet backbone, also called network core, is basically a group of routers interconnected by optical fiber as well as DNS servers containing infrastructure name servers, such as root Domain Name Servers (DNSs) employed for naming. The remaining components in the Internet structure that lie outside the network core, are simply access networks. An individual home network can be considered a small network or subnet. The Internet uses a gateway, also known as an edge router, to connect these networks to the edge of the network, also called network edge. The network core is composed of a set of routers and fiber links. The routers work together to determine the most efficient routing path for a packet from source to destination. A distributed algorithm is used that provides the flexibility to adapt to changing conditions, and routing tables are generated and maintained in real time. The core is provided by ISPs that interconnect multiple continents, called Global ISPs or Tier-1 ISPs, whereas the Regional ISPs are known as Tier-2 ISPs. The tier-1 ISPs that form the Internet backbone are interconnected at various access points called **Internet eXchange Points (IXPs)**. It would certainly appear that the intercommunication among computers would require some standardization that would facilitate their successful interactions. There should be some "protocol" that defines the manner in which they talk to each other so that messages are clearly understood. It is this "protocol", documented in a stack that is accomplished through modularization, development and upgrades that support operations such as web surfing, email, etc. By its very structure it is clear that the stack consists of different layers, each of which performs a special function. Modularization of the Internet is accomplished through layering. As a result, the Internet is being developed by many people, and institutions through a divide-and-conquer strategy. There is a strong interaction between layers in that each layer relies on the services of

the layer below and exports services to the layer above. It's the interface between layers that defines the interaction, for example implementation details can be hidden and layers can change without affecting other layers. Access layer is constituted by network with the same end-point, and the protocol used is Ethernet. Each host in a Ethernet network has a Network Interface Card (NIC) with a fixed address. MAC addresses uniquely identify hosts in the network and each of them processes packets intended for it. An Ethernet network constitutes a broadcast domain, where each Ethernet frame is received by all hosts. Actually, switches segment the network to limit the explosion of packets in the network, and only broadcast message are replicated. Switches remembers the source MAC addresses on the different ports. They only replicate the frame on the segment where the destination MAC address replies, with the support of ARP tables (for hosts) and CAM tables (for switches). Ethernet cannot be used for the entire Internet because of its broadcast packets, that would be highly inefficient for large networks. There is the need of a logical division of the networks : while Ethernet provides the Access layer, the Distribution layer is based on **Internet Protocol (IP)**, which is at level of Autonomous Systems (like big enterprises and ISPs). The main difference between Ethernet and IP address is that the first one is a physical addresses, so we cannot change the MAC address of our NICs, and tell who you are but not where you are, the second one is a logical address, so we can change the IP address of our NIC and is used to identify and reach networks and hosts. If two hosts belongs to the same network, then they can communicate using local addresses, otherwise they must use remote addresses. In order to know if one IP is in the same network than you, we can use the subnet mask. There are two version of IP addresses :

- **IPv4** : it defines IP address with 32 bits organized in four octets (8 bits in each)
- **IPv6** : has 128 bits.

For human readability the bits in each octet are separated by dots while writing an IPv4 address and colons in IPv6. IP addresses, regardless its version, are constituted by a network part and an host part, where the subnet mask defines precisely the boundary between them. There are three types of IP addresses :

- **Unicast** : these refer to a single destination host
- **Broadcast** : these refer to every host on a network or subnet

- **Multicast** : these refer to a group of IP addresses in a network, not necessarily all of them.

IP addresses has been classified in the following classes :

- Class A (24 bits for host addresses)
- Class B (16 bits for host addresses)
- Class C (8 bits for host addresses)
- Class D (multicast)
- Class E (reserved).

There are routable and non-routable address ranges. The first one need to be unique on the Internet (public), while the second ones (private, e.g. from 10.0.0.0 to 10.255.255.255, from 172.16.0.0 to 172.31.255.255, from 192.168.0.0 to 192.168.255.255) are defined in RFC 1918. Since each set of 8-bits can hold values from 0-255 we haven't flexibility at all. The idea is to use a Variable Length Subnet Mask in order to use a specific length of the subnet mask based on our needs. Many smaller networks can be grouped using the supernetting technique. In point-to-point links the subnet 31 is allowed. Other ways to reduce the waste of IP addresses in a subnet are NAT and IPv6.

2 Traffic monitoring

It's important to note that activities within the Internet can be approached in a modular fashion and this modularization is accomplished through layering. It is clear that the stack consists of different layers, each of which performs a special function. There is a strong interaction between layers in that each layer relies on the services of the layer below and exports services to the layer above. The task of a layer involves the exchange of messages that follow a set of rules defined by a protocol. When computers are connected with a network, guidelines must be established that support their interaction. The architecture that defines the network functionality is split into layers that collectively form what is commonly known as a protocol stack. Lets talk about **OSI model**. Each layer of the protocol stack may employ several protocols t implement the functionality of that particular layer. In a natural progression up the stack, the physical layer deals with the transmission of bits that are propagating over such media as copper,

fiber or radio. The data link layer aggregates the bits (e.g. in a frame), and performs the data transfer between neighboring network elements using as an example, Ethernet or WiFi. The network layer handles the routing of datagrams, in packet form, from source to destination using routing protocols. The transport layer performs the process-to-process communication using segments, i.e. message transfer using for example TCP. The session layer aggregates connections for efficiency, synchronization and recovery in data exchange. The presentation layer permits applications to deal with coding, encryption, and so on. Finally, we have the application layer which allow the user to invoke protocols for information exchange. Another well known model is **TCP/IP**. It's constituted by four layers : application layer corresponds to the top three layers of the OSI model, transport layer is equivalent to the transport layer of the OSI model, Internet layer is equivalent to the network layer of the OSI model, Datalink is equivalent to the data link layer of the OSI model and physical layer is equivalent to the physical layer of the OSI model. The layer ideal representation is that transport represent the illusion of direct end-to-end connection between processes in arbitrary systems, network transfers data between arbitrary nodes and data link transfers data between directly connected systems (via shared medium or direct cable). Each layer in the stack, with the exception of the physical layer, has a **header**. These headers facilitate the communication of information and are analogous to an envelope that contains both source and destination addresses. The link layer has a header containing MAC addresses, the network layer has a header containing IP addresses and the transport layer has a header containing the port number (i.e. service number). The port range is [0..65535], where the source port is randomly chosen by the OS and the destination port determines the required service. The port range [0 ..1023] contains well-known ports and they are used by servers for standard internet applications (e.g. 25 for SMTP, 80 for HTTP, 143 for IMAP). The ports in the range [1024..49151] can be registered with Internet Application Naming Authority (IANA), whereas ports in the range [49152..65535] are called ephemeral ports. Protocols are classified in two main categories :

- **connection oriented** (like TCP) : these protocols perform a number of very important functions. For example, they govern the movement of packets from source to destination under the specifications of certain standards, take actions that are specified in the packets, manage packet flow and congestion for optimal performance and even recover lost packets. The protocols work in conjunction with one another to accomplish the specified task required by the user.

- **connectionless** (like UDP) : these protocols doesn't provide any guarantee that the datagram is really deliver to the correct destination (i.e. there isn't control on data exchange). Furthermore, there isn't a possibility to recover from errors and manage the flow congestion.

The procedure to open a TCP connection between two hosts, A and B, is called three-way handshake, which is constituted by the following steps :

1. A sends a SYN segment to B with the sequence number field that contains the value x, which specifies the initial sequence number of A.

2. B sends a SYN and ACK segment to A, where the sequence number field contains the value y, which specifies the initial sequence number of B and the ack field contains the received value x increased by one.

3. A sends an ACK segment to B, where the sequence number field contains the value x + 1 and the ack field contains the received value y increased by one.

Packets that flow in the network can be captured using a network traffic dump tool, like dumpcap, wireshark or tcpdump. All of them can visualize and store the captured data, while the last two can also analyze the captured packets. In wireshark data from a network interface are "dissected" in frames, segments and packets, understanding where they begin and end. Then, they are interpreted and visualized in the context of the recognized protocol. When we captures packets, we collect a lot of them, so to make our life easier we can use filters. In fact, they allows us to only focus on requested packets or certain activity by network devices. Filters are divided in : display filters in order to inspect only the packets we want to analyze once the data has been processed, and capture filters in order to limit the amount of network data that goes into processing and is getting saved. In the second one the packets not captured are lost, where in display filters we display only captured packets matching the filter, without discard them. Frames are collected from the interface and passed to several, consecutive, "dissectors" one for each layer (they are passed from the bottom layer to the upper one). Protocols can be detected in two ways :

- **Directly** : if a frame has the field that explicitly states which protocol it is encapsulating.

- **Indirectly** : with tables of protocol/port combinations and heuristics.

We can capture network traffic in different modes :

- **promiscuous mode** : cannot fetch every packet due to network segmentation.
- **physical tap** : device that mirrors traffic passing between two network nodes.
- **port mirroring** : a port of the switch which mirror all the traffic.
- **aggressive approaches** like :
  - **ARP poisoning** : send ARP replies to steal IP addresses
  - **MAC flooding** : fill the CAM table to make switch act like an hub.
  - **DHCP redirection** : exhausts IP addresses of the pool and then pretends to be the default gateway of the network with the new DHCP requests.
  - **ICMP redirection** : use ICMP type 5 message to indicate a better route.

Now, the question is How to prevent packet capture ? We can use two different approaches :

- **Dynamic address inspection** : validates ARP packets with IP-to-MAC inspection by intercepting every ARP request and dropping the invalid ones (placed on switches).
- **DHCP snooping** : always implemented in switches, it distinguishes between trusted and untrusted ports and uses a database of IP-to-MAC. Ports that show rogue activity can also be automatically placed in a disabled state.

3 IPv6 introduction

IPv4 provides for a maximum 4.29 billion 32-bit addresses, which seemed like more than enough addresses when IPv4 became a standard in 1980. The number of IP addresses needed today far exceeds the world's population for several reasons. First of all, IPv4 addresses are often allocated in groups of addresses, as network addresses. Places such as companies, schools and airports are allocated network addresses for their users. But a much larger reason we need so many more IP addresses is the number of devices per

person that are being connected to the Internet. IPv6 provides more addresses than IPv4. In comparison, the 128 bit IPv6 address space provides for 340 undecillion addresses (i.e. approximately 2^{128}). The number of people accessing the Internet is increasing dramatically. Even with short-term solutions like NAT, we are in the final stages of public IPv4 address availability. There has also been extraordinary growth in the number of devices connected to the Internet. IANA assigns IPv4 addresses to the five Regional Internet Registries (RIRs) in /8 address blocks. Using this address space, RIRs then redistribute the IPv4 network addresses to ISPs and other end customers. On January 31, 2011, IANA allocated the last two blocks of IPv4 address space to RIRs. At this point, IANA had now run out of IPv4 addresses. This doesn't mean that IPv4 addresses are no longer available for end customers, but they can still get them from most ISPs. However, many ISPs are severely limited. So, in the early 1990s, IETF began the development of a new version of IP known as **IP Next Generation**, which later became **IPv6**. However, IPv6 was a long-term solution, and a short-term solution was needed immediately. Several short-term solutions were put into place. Two of the most important were **CIDR** and **NAT** with private IPv4 addresses. We know how CIDR allocates IP addresses. NAT along with private IPv4 addresses has been the reason IPv4 has survived all these years. It allows multiple hosts using private IPv4 addresses within their internal network to share one or more public IPv4 addresses when accessing the global Internet. NAT has also some critical points. At the very least, NAT means that our routers, application gateway and other devices must perform extra processing to make NAT work, which also causes latency. Other points are that it breaks peer-to-peer networking and accessing our "hidden" system from other networks. The IETF never intended NAT to be used for security. We want to say that in the late 1970s, a family of experimental protocols, known as **Internet Stream Protocol (ST)**, and later **ST2**, was developed. ST was an experimental resource reservation protocol intended to provide Quality of Service (QoS) for real-time multimedia applications such as video and voice. Although it was never recognized as **IPv5**, when encapsulated in IPv4, ST uses IP version 5.

3.1 Notation and types

IPv6 is much more than just a larger source and destination IP address. Developers of IPv6 took this opportunity to not only improve IP but also many of the protocols and processes related to IP. In fact, the benefits on IPv6 include : larger address space, stateless autoconfiguration, end-to-end

reachability without private addresses and NAT, better mobility support, peer-to-peer networking easier to create and maintain, and services like VoIP and QoS become more robust. An IPv6 address is 128 bits in length and hexadecimal is the ideal number system for representing long strings of bits. Every 4 bits can be represented by a single hexadecimal digit, for a total of 32 hexadecimal values from 0000 to FFFF. AS described in RFC 4291, the preferred form to represent an IPv6 address is x:x:x:x:x:x:x:x. Each x is a 16-bit section that can be represented using up to four hexadecimal digits, with the sections (also called **hextets**) separated by colons. The result is eight 16-bit sections for a total of 128 bits in the address. There exists some rules in order to reduce the notation involved in the preferred format such as :

1. **omit leading 0s** : one way to shorten IPv6 addresses is to omit leading 0s in any hextet. This rule applies only to leading 0s and not to trailing 0s (because omitting both would cause the address to be ambiguous).

2. **omit all 0s hextets** : the second rule for shortening IPv6 addresses is that you can use a double colon (::) to represent any single, continuous string of two or more hextets consisting of all 0s. If there are multiple possible reductions, RFC 5952 states that the longest string of zeroes must be replace with double colon and if they are equal then only the first string of 0s should use the :: representation.

An IPv6 address can be **unicast**, **multicast** and **anycast**. A unicast address uniquely identifies an interface on an IPv6 device. A packet sent to a unicast address is received by the interface that is assigned to that address. Similar to IPv4, a source IPv6 addresses must be a unicast address. In the following sections we will deeply describe the previous types of IPv6 addresses.

3.1.1 Unicast addresses

An unicast IPv6 address in turn can be :

- **Global Unicast Address** : it's a globally routable and reachable address in the IPv6 Internet. They are equivalent to public IPv4 addresses. Its generic structure has three fields :
  - **Global Routing Prefix** : it's the prefix or network portion of the address assigned by the provider to the customer site.

- **Subnet ID** : the subnet id is a separate field for allocating subnets with the customer site. Unlike with IPv4, it's not necessary to borrow bits from the Interface id (host portion) to create subnets. The number of bits in the subnet id falls between where the GRP ends and where the interface id begins.
- **Interface ID** : it identifies the interface on the subnet, equivalent to the host portion of anIPv4 address. The interface id in most cases is 64 bits.

Except under very specific circumstances, all end users will have a global unicast address (or more than one). It's range is from 2000::/64 to 3fff:fff:fff:fff::/64.

- **Link-Local Unicast Address** : a link-local address is an unicast address that is confined to a single link, a single subnet. They needs to be unique on the link and don't needs to be unique beyond the link. Therefore, routers do not forward packets with a link-local address. Link-Local Unicast addresses are in the range of fe80::/10 to febf::/10. The following 54 bits are recommended to be all 0s and 64 bits dedicated to the Interface id. They are created in two ways :
  - **Automatically** : an IPv6-enabled device must have a link-local address. By default, devices automatically create their own link-local unicast addresses. The prefix is typically fe80::/64, followed by a 64 bit interface id that is automatically generated in one of the two ways :
    - **EUI-64** generated : the IEEE defined the EUI process using the interface's Ethernet MAC address to generate a 64-bit interface id. When EUI-64 is used to create a link-local address, the fe80::/64 prefix is prepended to EUI-64 generate interface id. An Ethernet MAC address is 48 bits, a combination of a 24 bit Organizationally Unique identifier (OUI) and a 24 bit Device Identifier, written in hexadecimal. The EUI process is an insertion of 16-bit value of fffe between the 24 bit OUI and the 24 bit device identifier, with the Universal/Local bit flipped (7th bit of the first byte).
    - **Randomly generated** : EUI-64 is a convenient technique for automatically creating a 64 bit interface id from a 48 bit MAC address. However, it introduces a concern for some users : the ability to trace an IPv6 address to the actual

physical device using the 48 bit MAC address. To alleviate this privacy concern, devices can use randomly generated 64 bit interface ids.

- **Static** : the disadvantage of an automatically generated link-local address is that the long interface ID is difficult to recognize. It's much easier to use a simpler, manually configured ink-local address that is easier to identify.

3.1.2 Multicast addresses

Multicast is a technique in which a device sends a single packet to multiple destinations simultaneously (one-to-many), in contrast to a unicast address, which sends a single packet to a single destination (one-to-one). An IPv6 multicast address has the prefix ff00::/8, which defines a group of devices known as a multicast group. It's the IPv4 equivalent of 224.0.0.0/4. A packet sent to a multicast group always has a unicast source address. A multicast address can only be a destination address and can never be a source address. The typical structure of a multicast address is the following :

- ff00::/8 : the first 8 bits are 1-bits(ff), reserved for IPv6 multicast.
- Flags : the next 4 bits are allocated for flags. The fourth flag is the transient flag, which denotes two types of multicast addresses :
  - **Permanent (0)** : these addresses are assigned by IANA and include both well-known and solicited-node multicast.
  - **Non permanent (1)** : these are transient or dynamically assigned, multicast addresses. They are assigned by multicast applications.
- Scope : the scope field defines the range to which routers can forward the multicast packet (0 reserved, 1 interface-local scope, 2 link-local scope, etc.).
- Group id : the next 112 bits represent the group id.

There are two types of multicast addresses :

- **Assigned** : RFC 2375 defines the initial assignment of IPv6 multicast addresses that have permanently assigned Global IDs. IANA maintains the list of well-known IPv6 multicast addresses in a registry called

IPv6 Multicast Address Space Registry. For example when the group id value is one, then the multicast packet is sent to all network devices, when is 2 to all routers in the network, 5 to all OSPF router, and so on.

- **Solicited-Node** : a solicited-node multicast address is automatically created and assigned to an interface for every global unicast address, unique local address, and link-local address on that interface. These multicast addresses are automatically generated using a special mapping of the device's unicast address with the solicited-node multicast prefix ff02:0:0:0:0:1:ff00::/104.

One of the benefits of using a layer 3 multicast address is that it is mapped to a layer 2 Ethernet MAC address. This allows the frame to be filtered by the switch. This means these packets will only be forwarded out ports where there are devices that are members of the multicast group.

3.2 Addressing

Using **ICMPv6 Neighbor Discovery Protocol** Router Solicitation and Router Advertisement messages, hosts determine how to obtain their IPv6 addressing information dynamically. If SLAAC is used, the host can automatically obtain IPv6 addressing information, such as prefix, prefix length, and a default gateway address. The default gateway address is a link-local address, the IPv6 source address of the Router Advertisement message. ICMPv6 ND includes similar processes as in IPv4, such as address resolution, router discovery, and redirect, but also with some significant differences like prefix discovery, Duplicate Address Detection (DAD) and Neighbor Unreachability Detection (NUD). Neighbor Discovery uses five ICMPv6 messages :

- **Router Solicitation** message —
- **Router Advertisement** message — for router-device messages used with dynamic address allocation
- **Neighbor Solicitation** message —
- **Neighbor Advertisement** message — for device-device messages used for address resolution
- **Redirect** message — for router-device messages used for better first-hop selection.

A host sends a Router Solicitation message when it needs to know how to dynamically obtain its addressing information. This typically occurs during startup and is the default on most host operating systems. A Cisco router sends Router Advertisement messages every 200 seconds by default, and it also sends an RA message upon receiving a Router Solicitation message from a device. The RA message is a suggestion to devices on the link about how to obtain their addressing information dynamically. The RA message is sent to all IPv6 devices multicast address. The RA message contains three flag :

- **Address Autoconfiguration** flag (A flag) : when set to 1, which is the default setting, this flag tells the receiving host how to use SLAAC to create its global unicast address. SLAAC allows the host to create its own GUA address by combining the prefix in the RA message with a self-generated interface id (using EUI or randomly generated).
- **SLAAC with Stateless DHCPv6** flag (O flag) : when set to 1 (default setting is 0), this flag tells the host to obtain other addressing information, other than its GUA, from a stateless DHCPv6 server. This information may include DNS server addresses and a domain name.
- **Managed Address Configuration** flag (M flag) : when set to 1 (default setting is 0), this flag tells a host how to use a stateful DHCPv6 server for its GUA and all other addressing information. This is similar to DHCP for IPv4. The only information the host uses from the RA message is from the RA's source IPv6 address, which it uses as the default gateway address.

In the second case after a device generates one or more addresses using SLAAC, it contact a stateless DHCPv6 server for additional information. Remember that a stateless DHCPv6 server doesn't allocate or maintain any IPv6 global unicast addressing information. A stateless server only provides common network information that is available to all devices on the network, such as a list of DNS server addresses or a domain name. In particular, a host after obtained its own GUA sends out a DHCPv6 SOLICIT message to all DHCPv6 servers multicast address. One or more DHCPv6 server respond with a DHCPv6 ADVERTISE message, indicating that they are available for DHCPv6 service. The host responds to the selected server by sending an information REQUEST message, asking for other configuration information. The selected DHCPv6 server responds with a REPLY message that contains the other configuration information.

In the third case, stateful DHCPv6 doesn't utilize SLAAC to generate a GUA. A stateful DHCPv6 server provides IPv6 GUAs to clients and keeps track of which devices have been allocated which IPv6 addresses. A significant difference between stateful DHCPv6 and DHCPv4 is the advertising of the default gateway address. In IPv4, the DHCPv4 server usually provides the default gateway. In IPv6, only the router transmitting the ICMPv6 Router Advertisement can provide the address of the default gateway dynamically. There is no option within DHCPv6 to provide a default gateway address. A host after receiving the RA uses the source IPv6 address of the RA as its default gateway address. Then addressing and other configuration information is available from a stateful DHCPv6 server. So, the host sends out a DHCPv6 SOLICIT message to all DHCPv6 servers multicast address, searching for a DHCPv6 service. One or more DHCPv6 server respond with a DHCPv6 ADVERTISE message, indicating that they are available for DHCPv6 service. The host select one DHCPv6 server by sending to it a DHCPV6 REQUEST message asking for addressing and other configuration information. The selected DHCPv6 server responds with a reply message that contains a GUA and other configuration information. Finally, the host perform DAD on the address received from the stateful DHCPv6 server to ensure that this address in unique.

To ensure unicast address uniqueness, address resolution procedure includes **Duplicate Address Detection (DAD)**. The device sends a Neighbor Solicitation message for its own IPv6 address to detect whether another device on the link is using the same address. If a Neighbor Advertisement message is not received, the device knows its address is unique on the link.

3.2.1 Prefix Delegation process

In the world of IPv4, most internal networks use a private IPv4 address space for internal devices and NAT at the edge router to translate an address to a globally routable public IPv4 address. This is a common mechanism for most home networks and is partly responsible for keeping IPv4 alive for so many years. Avoiding the complications and problems with address translation, IPv6 uses a different technique. One of the methods IPv6 uses is DHCPv6 with the **Prefix Delegation** option, which provides a mechanism for automated delegation of globally routable IPv6 prefix from a provider's router to a customer's router using DHCPv6. In this process are involved two routers :

- **Requesting Router (RR)** : this is the router that acts as the DHCPv6 client, requesting the prefix(es) to be assigned.
- **Delegating Router (DR)** : this is the router that acts as the DHCPv6 server, responding to the requesting router's IPV6 prefix request.

Lets describe in details the Prefix Delegation process. First the RR's ISP facing interface needs an IPv6 address, that can be dynamically obtained using SLAAC, stateless or stateful DHCPv6 server. Then the RR initiates DHCPv6-PD in its SOLICIT message by including a request for an IPv6 prefix. The REPLY message from the DR (the ISP router) includes the IPv6 prefix. This is the prefix that the RR can use for its own internal network (i.e. a prefix that the RR can use to allocate addresses to its client).

3.3 Comparing the IPv4 and IPv6 headers

In comparing the two headers, notice that the IPv6 header takes advantage o 64-bit CPUs, respect to the 23-bit CPUs of the IPv4 header. As a result, all IPv6 fields start on an even 64-bit boundary or a multiple of 64. The advantage is that 64-bit CPUs can read one 64-bit-wide memory word at time. However, this structure doesn't negatively affect 32-bit CPUs because a 64-bit boundary is also a 32-bit boundary. A quick comparison of the two headers reveals that the IPv6 header is a simpler protocol with fewer fields than its IPv4 counterpart. This makes IPv6 a leaner protocol and provides more efficient processing. Both IPv4 and IPv6 begin with the **Version** field, which contains the version number of the IP header. The IPv4 **Internet Header Length (IHL)** field, is the length of the IPv4 header in 32-bit words, including any optional fields or padding (i.e. IPv4 has a variable header length). IPv6 doesn't have an IHL field because the main IPv6 header has a fixed length of 40 bytes, which allows for more efficient processing. The IPv4 **Type of Service (ToS)** field and the IPv6 Traffic class field are identical fields. In particular, they are used to specify what type of treatment the packet should receive from routers. This information helps provide QoS features by offering different degrees of precedence. The IPv6 **Flow Label** field, is a new field used to tag a sequence of flow of IPv6 packets sent from a source to one or more destination nodes. This flow can be used by a source to label sequences of packets for which it requests special handling by the IPv6 routers, such as "real-time" service. The IPv4 **Total Length** field is the length of the entire IPv4 packet, measured in bytes, including the IPv4 header and the data. The IPv6 **Payload Length** field is a 16-bit field that indicates the length in bytes of just the payload

following the main IPv6 header. It does not include the main IPv6 header, but only the payload length and extension headers length. The design of IPv4 accommodates MTU differences by allowing routers to fragment IPv4 packets when an MTU along the path is smaller than the sender's MTU. If a router receives an IPv4 packet that is larger than the MTU of the outgoing interface, this packet can be fragmented, depending on the options in the IPv4 header. Sometimes packets are fragmented into multiple packets at the source. The final destination of the IPv4 packet is responsible for reassembling the fragments into the original full-size IPv4 packet. Fragmentation divides IPv4 packets so that they can be forwarded out a link that doesn't support the size of the original packet. IPv4 **Identification, Flags and Fragment Offset** fields are used for packet fragmentation and reassembly. Unlike in IPv4, an IPv6 router doesn't fragment a packet unless it's the source of the packet. Intermediate nodes do not perform fragmentation. When an IPv6 router receives a packet larger than the MTU of the output interface, the router drops the packet and sends an ICMPv6 Packet Too Big message back to the source. This message includes the MTU size of the link in bytes so that the source can change the size of the packet for retransmission. A device can use **Path MTU Discovery** to determine the minimum link MTU along the path. The IPv4 **Protocol** field indicates the protocol carried in the data portion of the IPv4 packet. IPv6 has a similar field, the **Next Header** field, that specifies the type of header expected after the main IPv6 header. The same values used in the IPv4 Protocol field are used in the IPv6 Next Header field, along with some additional values for IPv6. Some of the most common values for both protocols are 6 for TCP and 17 for UDP. The IPv4 **Time To Live (TTL)** and IPv6 **Hop Limit** fields ensure that packets do not transit between networks for an indefinite period of time, as in the case of a routing loop. These fields are decremented by one each time a router receives the IPv4 or IPv6 packet. When the field contains the value zero, the packet is discarded, an ICMPv4 or ICMPv6 Time Exceeded is sent to the source of the packet. A **checksum** for the IPv4 header is provided for protection against any corruption in transit. Each router along the path verifies and recomputes this field. If the checksum fails, the router discards the packet. There is no checksum field in the IPv6 header. The designers of IPv6 didn't include this field because layer 2 data link technologies such as Ethernet perform their own checksum and error control. Because there is no checksum field in IPv6, the UDP checksum is mandatory in IPv6. The most notable and recognizable difference in IPv6 compared to IPv4 is the expansion of the source and destination address from a 32-bit address in IPv4 to a 128-bit address in IPv6. The IPv4 **Option** field is optional. It's

variable in size and rarely used, so it isn't usually included in most IPv4 packets. The IPv4 **Padding** field is used only if the IPv4 Options field is used and the size of the IPv4 header is no longer a multiple of 32 bits. When this is the case, the IPv4 header is padded with bits that are 0s so that it ends on a 32-bit boundary. Extension headers are an important addition to IPv6. They are optional headers that add flexibility and allow for future enhancements to IPv6. The main IPv6 header include a Next Header filed, which has one of two purposes : to identify the protocol carried in the data portion of the packet or to identify the presence of an extension header. The next header field indicates an additional header known as extension header. Immediately following the mandatory main IPv6 header, there can be zero, one or several extension header. It also allow the main IPv6 header to be a fixed size for more efficient processing. IPv4 uses options and padding fields to accomplish what the extension headers do in IPv6. The next header field is used to chain together multiple IPv6 headers and the data portion of the packet at the end of the chain. RFC 2460 recommends that when multiple extension headers are used in the same packet, those headers should appear in the following order :

1. Main IPv6 header
2. Hop-by-Hop options header
3. Destination option header
4. Routing option header
5. Fragment header
6. Authentication Header (AH)
7. Encapsulating Security Payload (ESP) header
8. Destination Options header
9. Upper-layer protocol header.

3.4 Link-local attacks

In this section we want to explain in details some common types of attacks over link-local addresses.

Network eavesdropping There is somebody within the network that capture packets transmitted by others and read the data content in search of sensitive information (such as password and session tokens). A possible solution to avoid data eavesdropping is to use encryption. In fact, using a strong encryption algorithm, the attacker will be not able to decrypt sensitive information. It's important not only to protect the content of the packets, but also some information that are used to move the packets (in this way we enforce privacy measure), using for example **TOR** or a **VPN**. Network eavesdropping is done by using tools called **network sniffers**, which goal is to focus on really breaking the confidentiality of the data. In particular, they analyze the collected data, filters all the valuable information and provide them at user level. They don't generate any kind of traffic, i.e. work in passive mode. To successful perform an eavesdropping attack, you need to be along the path of the communication. To realize this type of attack, first of all your network interface should be placed in promiscuous mode, i.e. you need to remove from the drivers of your network card the check of your MAC address, in order to accept every network packets. In wireless LAN the things are even worst, in fact you can potentially receive not only the packet of your network but also the packets of other networks that have a station that is in the range of your wireless card. So, we can place a sniffer in several places like :

- **Non-switched LAN (LAN with HUBs)** : it's the ideal case, because the hub duplicates every frame to all the ports.
- **LAN with switches** : in the switch the whole network is segmented, so we try to alter some of the mechanism of the switch, trying to make it to send the packets also in our segment (see MAC flooding). Or we can also use an ARP spoof attack, convincing the source of the packet that to send the packet to its intended destination, must send the packet to us. Lets see in details these two attacks :
  - **MAC flooding** : before considering the switches there were the bridges. The bridges were the first devices that were able to connect two network segments. The idea is to extends the broadcast domain but also reducing the collisions. These devices uses the **store & forward** technique, with which read and regenerate a frame only if needed. If we imagine that the bridge can have multiple port (i.e. a switch), then the mechanism is exactly the same, analyze the packet and understand on which port is connect a segment where the desired destination is connected. Whether

we generate a packet we insert our MAC address in the packet as source MAC address and the switch builds this association between a port, a segment and a list of the hosts in that segment considering the MAC source address. The MAC address is saved in the **CAM Table**, using the content itself to understand where to save that information in memory. What happens when a switch receive a packet in case of destination MAC unknown ? It's clear that it doesn't known on which port is the segment that host the destination MAC, so it replicates the packets on all the ports (flooding on the network). This behavior is used for the **CAM overflow** attack, where the CAM table is filled with useless MAC addresses and then whenever we have no more space to store a MAC address, a cache miss, we have to replicates the packets. Usually switches use hash to place MAC in CAM table,i.e. like hashed lists, where buckets can keep a limited number of values; if the value is the same there are n buckets to place CAM entries, if all n are filled the packet is flooded. In order to avoid this type of attack, we can employ the port security in switches :

  * allows us to specify MAC addresses for each port, or to learn a certain number of MAC addresses per port.
  * upon detection of an invalid MAC the switch can be configured to block only the offending MAC or just shut down the port.

– **ARP spoofing** : An ARP request message should be placed in a frame and broadcast to all computer on the network. Each computer receives the request and examines the IP address. The computer mentioned in the request sends a response, while all other computers process and discard the request without sending any response. We need ARP to resolve an IP address to MAC address. For example, host A wants to communicate with host B. Host A realize that host B is in the same network, so they can communicate directly using Ethernet. So, we need a MAC address of B but A don't know it. Host A sends an ARP request, including the IP address that he's looking for, to all network hosts. Subsequently, host B recognize its own IP address and will answer sending back an ARP-reply to host A. The ARP reply has the MAC address of B as source and MAC address of A as destination, picked from the original ARP request. Then

host A can finally sends directly the IP packets to host B. The main problem with this protocol is that it's untrusted, i.e. we cannot really trust the information that you see in ARP request and ARP reply. Whenever we receive an ARP reply, we store this information in an **ARP table**, which is a temporary table that is filled with this IP-MAC pairings. Whether we want to send a packet to a host in the same network, we check if the ARP table contains the MAC address corresponding to the target IP address. In a positive case, we use exactly that MAC address, otherwise we proceed as before (ARP request - ARP reply). Another mechanism used by this table is the **address aging**, which remove unused MAC addresses after a specific timeout. Whether we receive an ARP reply from the network, we extract the IP-MAC pairing and update the corresponding entry in the ARP table. This activity is typically performed with packets that are called **Gratuitous ARP**. It's an ARP reply that is not really an answer to an ARP request, but it's something performed by ourselves, used to announce my association between IP and MAC address. In this way, we avoid duplicate IP addresses in the network. What if we misuse the Gratuitous ARP ? A situation in which a host frequently announce its pairing using not his IP address but another one, for example the address of the network default gateway. So, all the other network hosts will be fooled to think that the default gateway is no more the MAC of the real router, but now the new MAC is the one of the attacker. In this case, the attacker will receive all the packets that are intended to go outside the network. What we have is an hijacking in the local network, i.e. you can pretend to be anybody in your network. The very first type of attack is the **Denial of Service**. Another type is the **Man In The Middle** attack, in which the attacker intercept the traffic and reroute it to get the reply, then forward the reply back.

- **Wireless LAN** : if we used weak encryption or no encryption, the scenario becomes equivalent to LAN with HUBs. If we use more recent encryption tools like WPA then we are protected, i.e. all the transmissions that we can potentially capture, are encrypted.

IPv6 Neighbor Discovery threats ICMPv6 Redirect can be abused in some way to create a rogue router. Whenever we have a host that sends a

packet to the router because it has to be sent to a network X and the router reaches another router in the same network, then the first router forward the traffic to the second one, but sends also an ICMPv6 Redirect message to the host, informing he that he can reach network X with one hop less. In IPv6 a router can known if two hosts are not really in the same network, but if they are in the same link local. Some types of attacks based on IPv6 ND are :

- Neighbor Solicitation / Advertisement spoofing : same as ARP spoofing, i.e. pretends to be somebody else.
- DAD DoS attack : imagine you use SLAAC in order to get a Global Unicast address, and the attacker replies to you that the address is used. This scenario is repeated and repeated, generating a DoS attack. At some point the host stop to sends Neighbor Solicitation, deciding to not be part of the network.
- spoofed Redirect message, default router compromise, malicious last hop router (router threat).

Rogue Router Advertisement What if the RA comes from a malicious router ? One of the most common problem due to rogue RA is the **VPN bypass**. Whenever you connect to a VPN, you will be part of a new network. Usually, this network is an IPv4 network and you're supposed to use that network for all the communication that should go in the VPN. If you're able to inject a rogue RA, you can fool the host and make it to use your own network, the one injected in the RA, instead of the VPN network. If the attacker is able to inject a new DNS or a new default gateway, he can sniff all client traffic, attempt MITM attacks, impersonate server/systems and capture presented user credentials, gain access into the other networks of the systems. The countermeasures are at network level. It means that your switches/routers should protect you against these types of attacks. The mechanism is not called anymore port security but is called **RA guard**. We can write some policies that states from which port the RAs have to come, i.e. it's the port where default gateway is supposed to be. If we receive a RA from another port, we just discard it.

Router Advertisement flooding We can also have a DoS attack, using the RA flooding. A host can potentially generate an IPv6 address for each prefix that he has received from RAs. The same reasoning is applied for the routing information.

DHCP rogue server It's a situation in which an attacker is able to request all the available DHCP addresses. Once the addresses are gone, an attacker could use a rogue DHCP server to provide addresses to clients. Since DHCP responses include DNS servers and default gateway entries, the attacker can pretend to be anyone.