[Th 4] Network security

TPC/IP SECURITY

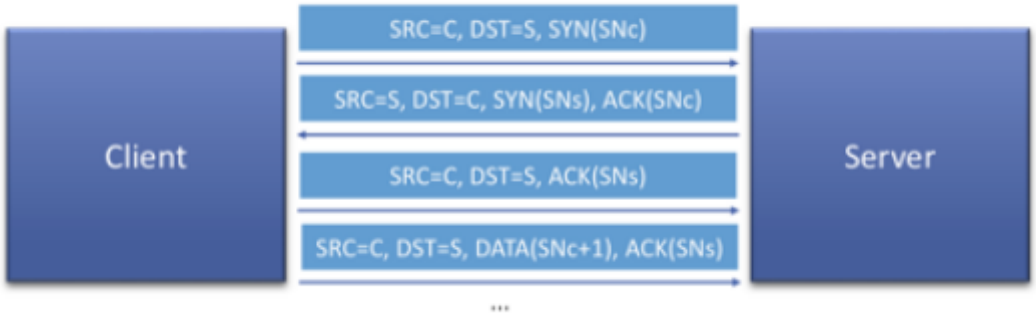
Sequence Prediction Attacks

One problem in TCP/IP is the **sequence prediction.**

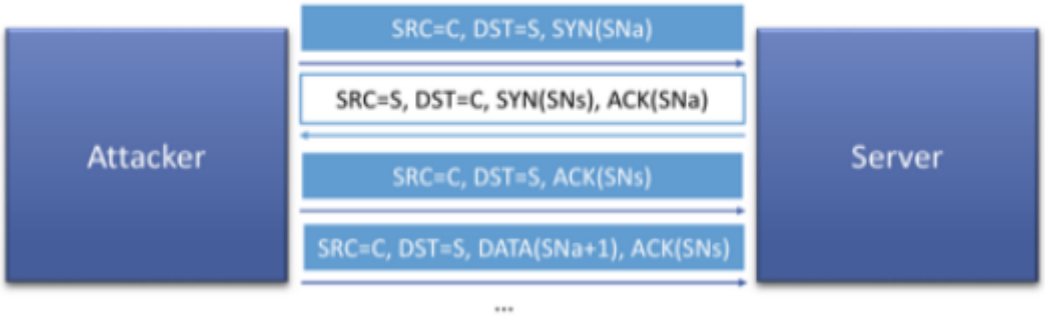
In TPC/IP each packet is enumerated with an integer that is a sequence of 32 bits in order to reconstruct the order.

The basic idea is that:

- **the client sends his sequence number to the server, with a SYN message, let's call it SNc**
- **the server sends back to the client his personal sequence number, with a SYN message, let's call it SNs, and an ACK value in order to explicit the fact that it obtained the client message**
- **the client sends to the server an ACK message**
- **the communication starts**



Now we talk about **spoofed connection** when an attacker knows the server sequence number (SNs) and a client C IP address and uses it to send messages to the server impersonating the legit client C.



How the attacker can spoof SNs?

TCP/IP was created in a way that this value was extracted by the time in which the connection was requested and for avoiding interferences with the other connection this value was increased by 250.000 per second.

So the attacker could predict it capturing one previous message.

Now we have to consider that if the client obtains the response of the server to the attacker's request (remember that it has the client IP address) it will send back an RST packet (reset) because there are no legitimate connections opened with the server.
To avoid this the attacker can easily send a large number of requests to the client to fill the buffer and force the client to don't process the server packet.

How the attacker can use the spoofed connections?

One possibility is to perform a reset attack:

- **the attacker predicts all server conversations SNc and impersonating the legitimate clients will send in a loop a reset packet**
- so this will generate a DoS attack

- this attack is based on the asymmetric between client and server resources (an attacker can send thousands of packets in a second)

A first line of defense is the introduction of the TTL. This is because each router was at 1 jump of distance so if the packet performs another path we are able to understand it.

DNS ATTACKS

DNS is based on the UDP protocol that doesn't use the sequence numbers.
UDP is based on the use of ports and for the DNS service it is fixed to port 53.

Now if an attacker knows that a client is going to perform a request then it can impersonate the DNS server and craft a malicious response (guessing the port).

One **defense to this attack was the use of DNSSEC, where we use a signature for DNS records.** But in this case, we have some troubles with authentication and key distribution.

SYN FLOODING ATTACKS (DoS)

In this attack, an attacker sends a lot of packets to the server with different source IPs.
So basically the server obtains a lot of SYN packets that are used to start a conversation and is forced to maintain these conversations semi-opened.

This is a DoS attack because when the memory of the server is saturated then it starts to reject new connections so legitimate clients are not able to connect to it.

A defense could be to make the Server stateless until it does obtain the third ACK packet from the client, but if we lose the third packet no one will resend it.

Another solution is to use SYN cookies:

- we use cryptography so the server doesn't need to maintain a state
- **Initial Sequence number = SNc + (timestamp || hash(src/dst IP + port, secret key, timestamp))**

BANDWIDTH AMPLIFICATION ATTACKS (DoS)

A bandwidth amplification attack is a DoS attack where the attacker sends a ping request (in a loop) in broadcast to the entire network (ICMP) with a source IP that is the victim IP.

So the attacker can send a lot of ping messages to the entire network and the victim will receive all the responses to this ping.

The defense now is performed on the routers that block "direct messages on broadcast".
The attacker could create a script to send them one by one the ping message but it is a high-cost strategy.

A modern variant is done on DNS where with a little query the DNS server can reply with a very huge message (with DNSSEC responses containing a lot of information).

Since DNS is on UDP source address is not verified and a defense is quite hard.

DHCP ATTACKS

DHCP protocol is based on the fact that a new host that wants to connect to the network sends in a broadcast a request for a valid IP. The server that gets the request sends back an IP offer and the host can accept or not it.

In this attack, the attacker can impersonate the DHCP server and can craft in a malicious way the DNS server address, the router IP address, and other information for the news hosts that connect to the network.

It is very hard to perform because the attacker needs to intercept the exact moment in which the victim connects to the network.

ARP SPOOFING

ARP spoofing is based on the Address Resolution Protocol.
This protocol is used to discover which is the physical MAC address of a host from his IP address.

The idea is that the host that needs to convert another host to a MAC address sends an "ARP request" in LAN broadcast and the owner of the IP replies with an "ARP reply" that contains the MAC address.

ARP cache poisoning (or poisoning routing)

The idea here is to force the victim to ARP table modification.

1. **the attacker A sends an ARP reply to another host B using his MAC address but the IP address of another host (R) (for example the gateway IP)**
    1. **so from now on, the victim will send the packets to the attacker A thinking he is the router R**

From now the attacker can perform:

- spying attack
- man in the middle
- DoS attack, making the victim lose packets

TCP/IP SECURITY

How can an attacker discover which software/protocols the victim is running?

Basically, the idea is to:

- verify that the system is listening on well-known ports,

- use DNS to understand the hostname of an IP address
- assume that the system is vulnerable.

In addition, an attacker can discover the victim system IP address using the traceroute, using Nmap, or just trying all 2^32 possible IPv4.

IMPROVING TCP/IP SECURITY

1. **The first approach is to apply protocol-compatible fixes to the TCP/IP implementation**
2. **The second approach is to deploy new protocols.**
3. **The third approach is to use cryptography, for example, IPsec**
4. **The fourth approach is to use:**
    1. **firewalls, VPN (virtual private networks), or intrusion prevention systems (IPS).**
5. **The last approach is to use and implement the security over the TCP/IP stack, so basically we could use Kerberos or TLS/SSL.**

The best thing to do is to mix these approaches to improve the security.

FIREWALLS

A firewall is a host that is responsible for monitoring network traffic (in and out traffic) following predetermined security rules.

They are classified by:

- placement
    - host-based firewalls
    - network-based firewalls
- kind of operations
    - packet filtering firewalls

- stateful inspection firewalls
- application-level gateways
- circuit-level gateways

Placement firewalls

Host-based firewall

It is a firewall installed on a single machine. It monitors the single host traffic and it is very flexible.
However, we have to consider that if the host is infected then it could be compromised.

Network-based firewall

This kind of firewall is used to split the private internal network (that is protected) and an untrusted network.

It is able to filter the traffic and draw the perimeter of the network at all. It is good at detecting malware-generated traffic but is not so flexible because the rules must be generic for each host.

Operational firewalls

Packet filtering firewall

The idea here is to understand which packet to forward and which one to discard.
This choice is based on rules that take into account the packet header information (source, destination, ports etc)

There are two default policies:

- **discard: discard everything until it is not explicitly permitted**
- **forward: forward everything until it is not explicitly prohibited**

Example

Policy	Regola
Negare l'accesso al Web	Scartare tuti i pacchetti in uscita verso ogni indirizzo IP, dalla porta 80
Negare le connessioni TCP in ingresso, ad eccezione dei Web Server pubblici	Scartare tutti i pacchetti TCP SYN in ingresso, ad eccezione di W.X.Y.Z, dalla porta 80
Prevenire il consumo di banda	Scartare tutti i pacchetti UDP in ingresso, eccetto quelli del DNS e i broadcast del router.

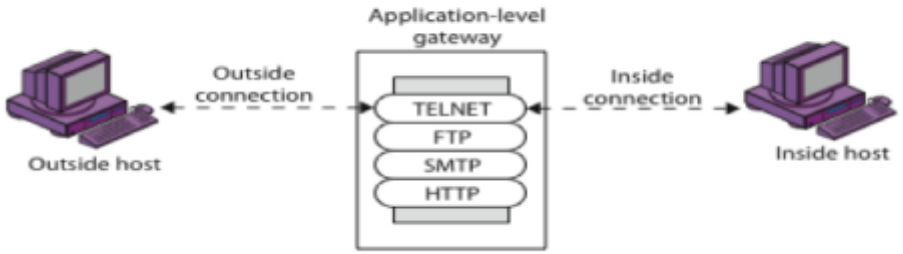
Stateful Inspection firewall

It monitors the packet headers and can take into account TCP connections and their sequence numbers and some UDP connections.

Application level gateways

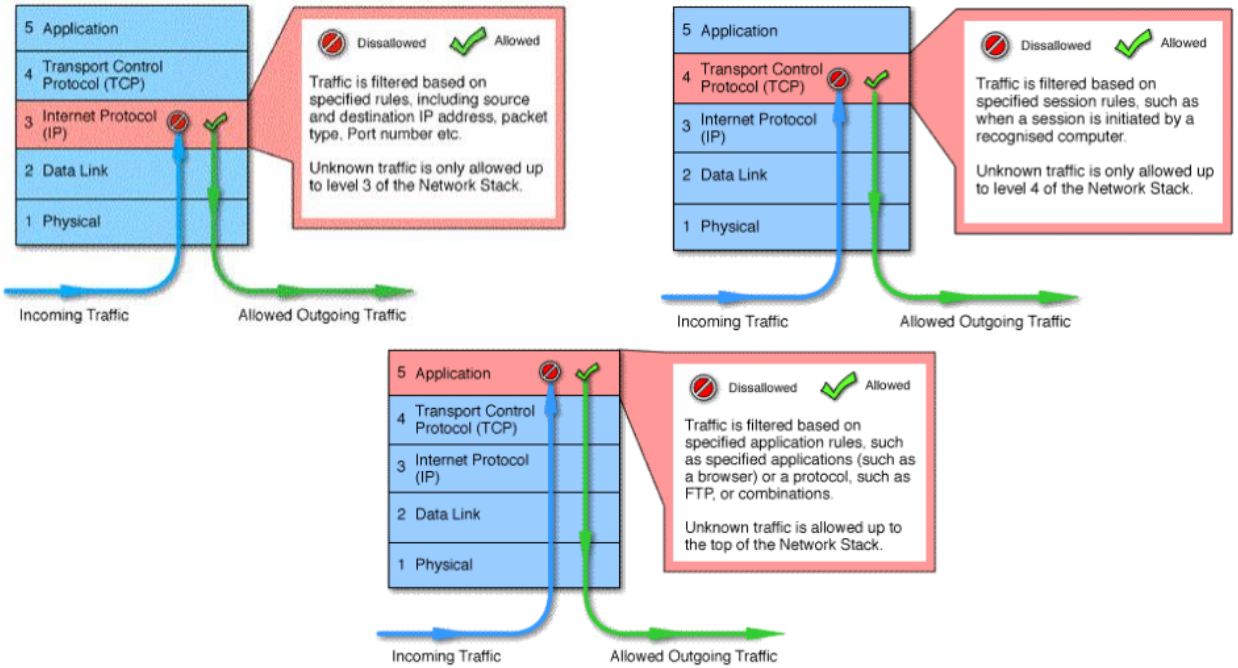
The idea here is to use an intermediary to perform the connection between two hosts at the application level.

The client contacts the gateway with the remote hostname and performs the authentication. The gateway contacts the remote hostname and the connection is enabled.



- The advantage of this approach is that is very secure but we introduce some overhead.

PACKET FILTERING VS. STATEFUL INSPECTION VS. APPLICATION-LEVEL



Circuit level gateways

The idea here is to avoid the proxying and instead create two different connections.

The first one is between the internal host and the firewall.

The second one is from the firewall and the external host.

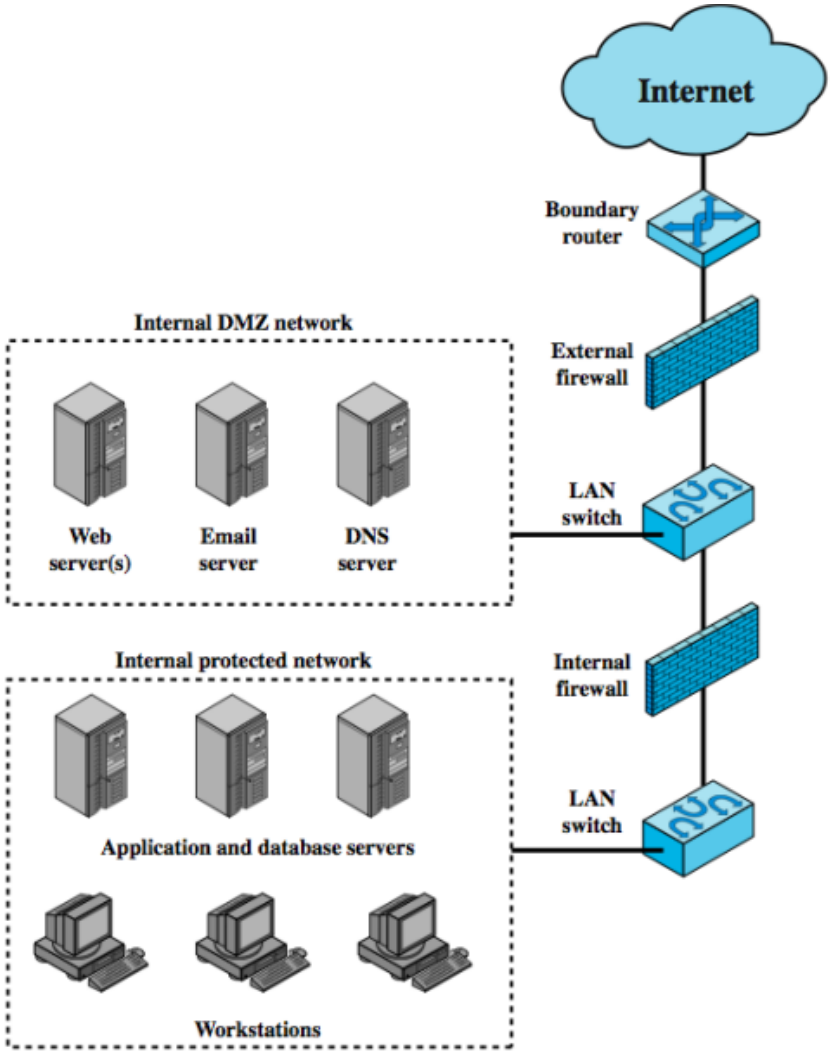
Once the connection is performed TCP segments are transmitted from the first connection to the second one without looking at them.

We assume that the internal hosts are "clean".

Firewall Placement

The most used option is to install two different firewalls:

- one external firewall that protects the DMZ network
  - demilitarized zone
- one more internal that protects the most internal network



In general, we can use multiple internal firewalls to protect internal networks from other internal networks.

Firewalls Limitations

The first problem is that we need to assume that internal hosts are "clean".

In addition, it is hard to identify untrusted packets and the TCP/IP implementation doesn't help. For example, we could not use firewalls when we fragment IP packets because in this case, we need TCP ports to be specified in some packets and the payload in another one.

VPN (virtual private networks)

It uses low-level cryptography and authentication in order to provide a secure connection on unsecured networks (also the Internet).

The most common protocol to secure IP packets is IPsec.
It is used on firewalls and for this reason the traffic in both directions is secure.

Intrusion Prevention System

They are added to the firewalls and are considered the new version of firewalls at all.
Also there we could have host-based or network-based IPS.

Host-based IPS

It is used to monitor a single host and can be configured for a specific platform or can be configured as an application sandbox.

Signature-based IPS

In this context, the idea is to look for malicious patterns in packets.
A signature is a pattern that is considered malicious.

- example telnet is trying to log in as root
- an email containing the subject "free .." and an exe file attached

They are very suitable for known threats and it is easy to create a signature DB.
But they can be inefficient with some evasion techniques and the knowledge about the attacks depends on the domain.

Anomaly-based techniques

The idea here is to generate models that are able to identify malware and anomalies.
So the approach is to generate a model based on the normal usage of the system, training it in a certain time period, and then using this model we can analyze the current behavior in order to identify strange behaviors.

This approach is very efficient with new threats and it is independent from the domain.
The main problem is the model construction and how we calculate the deviation from normal behavior.

There exists also the approach in which we create a model trained on malicious behavior in order to discover problems looking at the log files.

Network-based IPS

In NIPS the idea is to monitor the traffic discarding packets or closing TCP packets.
It is based on a hybrid approach of 4 techniques:

- pattern matching of specific byte sequences
- stateful matching, over a stream of packets and not on single ones
- protocol anomaly detection
- traffic anomaly detection

Intrusion Detection System

An intrusion detection system is a system that is able to identify an intrusion in the system.
It is based on the assumption that an intruder performs always these actions:

1. target acquisition and information gathering
2. initial access

3. privilege escalation
4. information gathering or system exploit
5. maintaining access
6. covering tracks

In this system, the main components are:

- **sensors that collect data**
- **analyzers that determine if an intrusion is occurred**
- **a user interface**

They must be 24/24 h active, be failure tolerant, have not a big overhead, etc.

Host-based detected events

These types of events are on the system level (USB insert), on the OS level (privilege escalation, buffer overflow), and on the application level (antivirus shut down etc).

Network-based detected events

In this case, we can find some attacks as:

- application layer attacks as buffer overflow, format string attacks
- transport layer attacks such as port scanning, SYN flooding
- network layer attacks as spoofed IPs
- unexpected application services such as tunneled protocols, backdoors, etc
- other policy violations

ALERTS LOGGING

Generally, the NIDS sensors, for a single detection, log in a file:

- timestamp
- connections and session IDs
- source and destination IP packets
- TCP or UDP used ports
- number of transmitted bytes

RULES EXAMPLE

In general, there are some common rules:

- a user cannot be logged in more than one session
- users cannot copy system files such as password files
- and so on

Current limitations

Currently, the biggest limitation is the number of false positive detections we can have:

		IDS output	
		Normal	**Attack**
Traffic	**Normal**	True negative	False positive
	Attack	False negative	True positive

HONEYPOTS

A honeypot is a host that is full of bad information for the attackers that are crafted to allow the attackers to exploit them, for example, fake DB.

So the idea is to attract the attackers into this machine and force them to avoid the real machines.
This strategy allows the defender to study the attacker's methodologies.

Low interaction honeypot

This is software that emulates a service or an entire system in a realistic way.

High interaction honeypot

This is a real machine with an OS and services and applications.