Attaques et sécurisation des couches OSI

Attaques et contres niveau 1-4

Carlos Aguilar

carlos.aguilar@enseeiht.fr

IRIT-IRT

Sources

Sources :

- Pierre-François Bonnefoi du Master CRYPTIS à Limoges
- Julien Cartigny du Master CRYPTIS à Limoges
- Dan Boneh de Stanford University
- Ron Rivest du M.I.T.
- Cédric Blancher (EADS)

Plan

Le Raté Historique Extraordinaire

Principe

Présentation de 5 minutes faite par un étudiant
Tout le monde en fait une (forme libre)
Une présentation en début de chaque créneau

Contenu

Objectif : mélanger drôle et culture générale
Une grande catastrophe stupide qui aurait dû être évitée
Une attaque dévastatrice (éviter quand même le morbide)
Un bug / une faille incroyable
... ou tout autre chose qui paraisse intéressante au présentateur

Contraintes

En rapport avec la sécurité
En rapport (potentiellement lointain) avec le réseau ou la crypto

Plan

Attaques sur le lien physique : Variantes

Déni de service

Très simple (attention aux canaux) mais avec un effet local
Émettre en continu (téléphone, Ethernet, wifi)
Utilise généralement un appareil dédié (brouilleur)
Interdit par la loi (comme la plupart des autres attaques)

Écoute

En cablé : dispositif enregistreur dédié
En aérien :

- Communications aériennes
- Par rayonnement parasite (paranoïa ?)

Attaques sur le lien physique : Sniffers logiciels

Comment faire ?

Carte réseau en mode promiscuous (filtre MAC matériel désactivé)
`ifconfig eth0 promisc`
Capture des paquets (libpcap, tcpdump, sniffit, dsniff, ettercap, wireshark)

Contre-mesures

Segmentation et Chiffrement
Détection (anti-sniff, sniffdet)

- Génération de requêtes DNS (nombreuses/pour fake IPs)
- Réponses ARP pour des paquets avec en-tête Ethernet mal-formée
- Création d'entrées ARP fausses
- Temps de réponse en cas de charge réseau

$\rightarrow$ Anti Anti-sniff

MAC spoofing (1/2)

Méthode What/When/Where/Why/How !

What ?

Donner une fausse adresse MAC, généralement usurpée

Why ?

Contournement des moyens d'authentification/autorisation

- accès à une adresse IP dans un VLAN donné
- accès à une connexion internet (portail captif)

Écoute

- MAC Flooding (et donc écoute)

MAC spoofing (2/2)

How ?

```
[root]# ifconfig eth0
eth0 Link encap:Ethernet HWaddr 00:A0:C9:29:3C:68
inet addr:192.168.0.1 Bcast:192.168.0.255 Mask:255.255.255.0
UP BROADCAST MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:4 dropped:0 overruns:0 carrier:4
collisions:0 txqueuelen:100
RX bytes:0 (0.0 b) TX bytes:168 (168.0 b)
Interrupt:11 Base address:0xdf00 Memory:df9ff000-df9ff038

[root]# ifconfig eth0 down
[root]# ifconfig eth0 hw ether 01:02:03:04:05:06
[root]# ifconfig eth0 up
[root]# ifconfig eth0
eth0 Link encap:Ethernet HWaddr 01:02:03:04:05:06
inet addr:192.168.0.1 Bcast:192.168.0.255 Mask:255.255.255.0
UP BROADCAST MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:4 dropped:0 overruns:0 carrier:4
collisions:0 txqueuelen:100
RX bytes:0 (0.0 b) TX bytes:168 (168.0 b)
Interrupt:11 Base address:0xdf00 Memory:df9ff000-df9ff038
```

Remarque : attaques manifestes

Modèles d'attaquant

Un attaquant peut être :

- Interne/Externe
- Contrôler une machine / plusieures
- Passif/Actif

Attaquants actifs

Peuvent dévier des protocoles mais tout en voulant rester indétectables !

Attaques physiques

Écoute : indétectables
DoS : Clairement manifestes mais difficiles de localiser (surtout en aérien)

MAC spoofing

Ça dépend de ce qu'on fait (inventer, remplacer, cloner)

ARP spoofing/poisoning (1/5)

What ?

Pervertir le cache ARP d'une machine

Why ?

Écoute et Contournement de l'authentification/autorisation
Par détournement du trafic : associer notre MAC à l'IP de qqun d'autre

Background sur ARP : généralités

- Traduction liaison/réseau (datagramme IP $\rightarrow$ trame Ethernet)
- Chaque requête est lue par tous les ordinateurs du sous-réseau
- Utilisation de caches de traduction
- Point d'entrée : m.à.j. du cache lors de la réception d'une requête ARP

ARP spoofing/poisoning (2/5)

Background sur ARP : paquets

Deux types de paquet : requête (who-has) et réponse (is at). Contiennent :

1. Adresse physique de l'émetteur
2. Adresse IP de l'émetteur
3. Adresse physique du destinataire
4. Adresse IP du destinataire

Paquet générique pouvant être utilisé entre différentes couches 2 et 3
Une entête du protocole utilisé en niveau 2 (Ethernet) est ajoutée

Background sur ARP : utilisation

Quand on doit contacter une adresse IP dont on a pas le MAC on envoie une requête who-has [1||2||FF :FF :...||4] (+diffusion sur l'en-tête Ethernet)

Quand on envoie un is at on remplit tout normalement, la réponse est en 1

Background sur ARP : actualisation du cache

Quand on reçoit un who-has (qu'on traite) on note le lien 1-2
Quand on reçoit un is at (qu'on a demandé) on note le lien 1-2

ARP spoofing/poisoning (3/5)

Et un MAC Spoofing ?

Envoyer un trame Ethernet avec pour source l'adresse de notre victime ⇒ Modification du *Content Adressable Memory* du commutateur/routeur.

```
# Avant

Port | Adresse MAC
-------------------------
1    | 52:54:05:F4:62:30                    # alice
2    | 52:54:05:FD:DE:E5                    # bob
3    | 00:10:A4:9B:6D:81                    # charly

#Après

Port | Adresse MAC
-------------------------
1    | 52:54:05:F4:62:30                    # alice
2    |
3    | 00:10:A4:9B:6D:81; 52:54:05:FD:DE:E5 # bob, charly
```

Pbs : Manifeste, pbs de comm avec Bob, trames contradictoires...
Peut avoir un intérêt pour faire passer le commutateur en mode pont

ARP spoofing/poisoning (4/5)

Contexte

Charly veut qu'Alice ajoute une en-tête Ethernet avec l'adresse MAC de Charly aux paquets IP destinés à Bob

How ? Création

Répondre à la requête who-has avant bob ? bof ...
Envoyer un is at non demandé ? marche pas généralement

Utiliser la mise en cache lors de la réception d'un who-has

Envoyer à Alice un who-has $[MAC_c || IP_b || FF:FF:... || IP_a]$
Rq : Il est possible de l'envoyer en unicast ! (pas de contrôle de cohérence ARP-Ethernet)

ARP spoofing/poisoning (5/5)

How ? Mise à jour

Envoyer des is at régulièrement avec l'adresse MAC de Charly et IP de Bob

How ? Pratique

Utilisation de Scapy

```
Welcome to Scapy (2.1.0)
>>> h=ARP()
>>> h.show()
###[ ARP ]###
  hwtype= 0x1
  ptype= 0x800
  hwlen= 6
  plen= 4
  op= who-has
  hwsrc= 00:21:5d:c5:9e:5e
  psrc= 192.168.1.65
  hwdst= 00:00:00:00:00:00
  pdst= 0.0.0.0
```

```
>>> h.psrc=("192.168.1.254")
>>> h.pdst=("192.168.1.73")
>>> h.show()
###[ ARP ]###
  hwtype= 0x1
  ptype= 0x800
  hwlen= 6
  plen= 4
  op= who-has
  hwsrc= 00:21:5d:c5:9e:5e
  psrc= 192.168.1.254
  hwdst= 00:00:00:00:00:00
  pdst= 192.168.1.73
```

Contres

IDS (ARPWatch)
Commutateurs de niveau 3 / routeurs : Cache ARP Statique

Application aux attaques

Écoute

1. Charly réalise l'attaque décrite :
   - Il reçoit les paquets qu'Alice veut envoyer à Bob
2. Il active le routage sur sa machine et le configure pour renvoyer les données reçues à Bob après interception

Man In The Middle

1. Charly réalise l'attaque décrite à la fois sur Bob et sur Alice
   - Il reçoit les paquets qu'Alice veut envoyer à Bob
   - Il reçoit les paquets qu'Bob veut envoyer à Alice
2. Il active le routage dans les deux sens

DoS

Il suffit de pas router. Un DoS discret ! Le cas des DNS est particulièrement intéressant (en cas de corruption du DNS secondaire).

Plan

Définition et objectifs

What ?

Utiliser la fragmentation IP à des fins autres que l'adaptation des MTU

Why ?

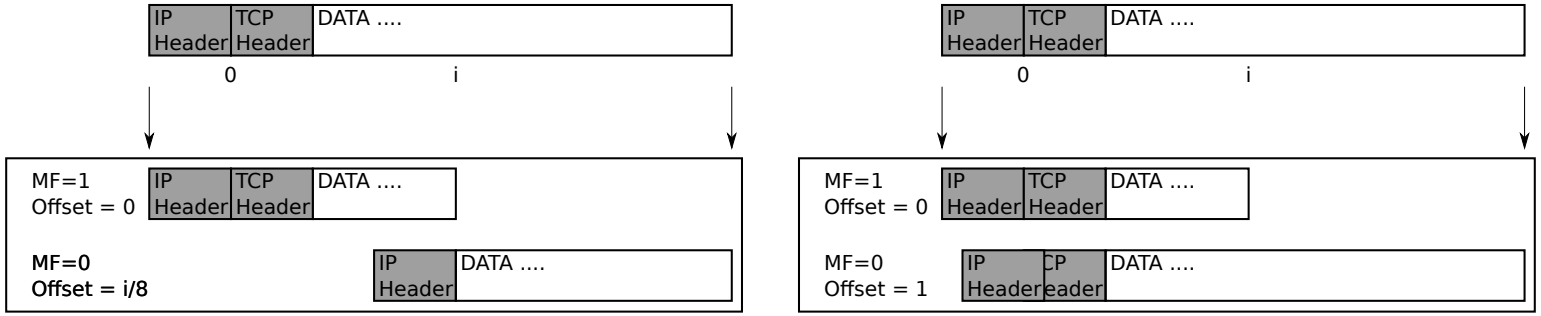
Contournement des moyens d'autorisation

- traverser les firewalls
- effectuer des actions interdites sans être détecté par un IDS

Déni de service

- saturer la mémoire d'une machine
- faire planter le driver réseau

Pourquoi ça marche ?



La fragmentation et les firewalls/IDS

Une application sans état analyse les paquets du réseau indépendemment.

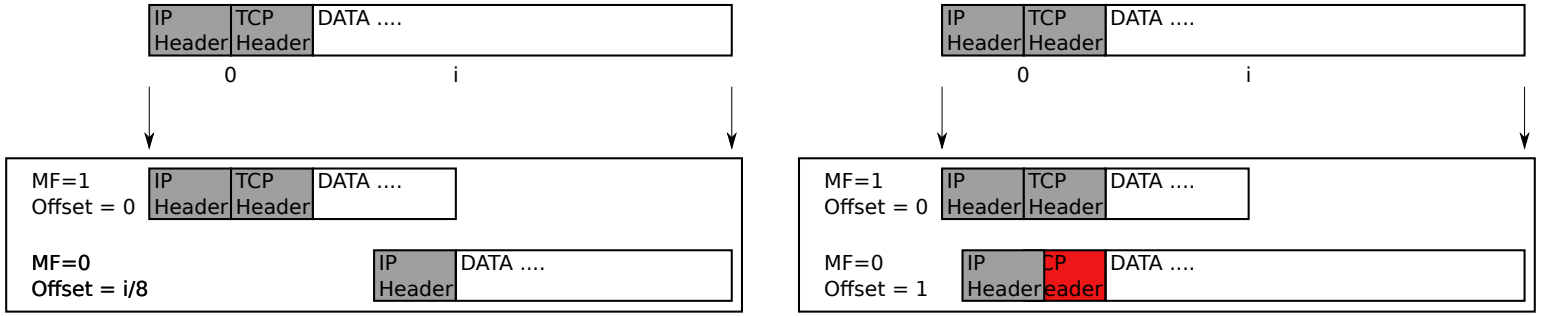
- Firewall : vérification de l'en-tête TCP/UDP du premier fragment
- IDS : recherche de signatures sur chaque fragment

Même une application stateful peut avoir une stratégie de reconstruction différente !

Deux voies d'attaque

Fragmentation à outrance
Messages superposés

Pourquoi ça marche ?



La fragmentation et les firewalls/IDS

Une application sans état analyse les paquets du réseau indépendemment.

- Firewall : vérification de l'en-tête TCP/UDP du premier fragment
- IDS : recherche de signatures sur chaque fragment

Même une application stateful peut avoir une stratégie de reconstruction différente !

Deux voies d'attaque

Fragmentation à outrance
Messages superposés

Première voie : la fragmentation à outrance

Idée

Envoyer dans chaque fragment huit octets (micro-fragmentation)
Firewall/IDS sans reconstruction verra même pas les en-têtes (sauf UDP !)

How ? Utilisation de Scapy

```
from scapy.all import *
destIP = ''123.123.123.123''
data = ''XXXXXXXXX''
ip = IP(dst=destIP, id=12345)/TCP(sport=65000,dport=25)/data
packetList = ip.fragment(francise=8)
for packet in packetList : send(packet)
```

Deuxième voie : superposition des fragments

Idée : plusieurs options de reconstruction

- First : Windows, MacOS, SUN
- Last : Cisco
- Linux : Linux (premier permettant de compléter)
- BSD : AIX, FreeBSD, Wireshark (dernier permettant de compléter)

Le firewall/IDS utilise une reconstruction et la victime une autre

How ? Utilisation de Scapy

```
from scapy.all import *
destIP = ''123.123.123.123''
data = ''XXXXXXXXX''
ip1 = IP(dst=destIP, id=12345, flags=1, frag=0)/TCP(sport=65000,dport=22)/data
ip2 = IP(dst=destIP, id=12345, flags=0, frag=0)/TCP(sport=65000,dport=25)/data

datas = ['' V'', ''AI'', ''BR'', ''CU'', ''DS'']; pL = []
pL[0] = IP(dst=destIP, id=12345, flags=1)/UDP(sport=65000,dport=25)/datas[0]
for i in range(1,4) : pL[i] = IP(dst=destIP, id=12345, flags=1,frag=i)/datas[i]
pL[4] = IP(dst=destIP, id=12345, flags=0,frag=4)/datas[4]
for packet in pL : send(packet)
```

Plan

L'usurpation d'adresse IP

What ?

Envoyer des paquets sur le réseau avec une adresse IP qui n'a pas été, ou n'est pas destinée à être, attribuée à l'émetteur

Why ?

Contournement des moyens d'authentification/autorisation
Éviter d'être tracé (quel que soit l'objectif)

Pourquoi ça marche ?

Aucun moyen d'authentification de l'émetteur mis en place (ni sur IP ni sur le routage)

Contre minimum

Filtrage `egress, ingress` au niveau des routeurs
Ne router que les paquets qui ont des IPs qui correspondent à leurs réseaux

Problèmes

ICMP, UDP, DNS

Aucun (pas d'ajout de sécurité par rapport à IP)

TCP

Circuit virtuel

- les paquets sont ordonnés
- les données sont acquittées

Difficile de spoofer en aveugle
Difficile d'insérer du trafic

TCP Connection Spoofing (1/2)

What ?

Établir une connexion TCP avec l'adresse IP d'autrui

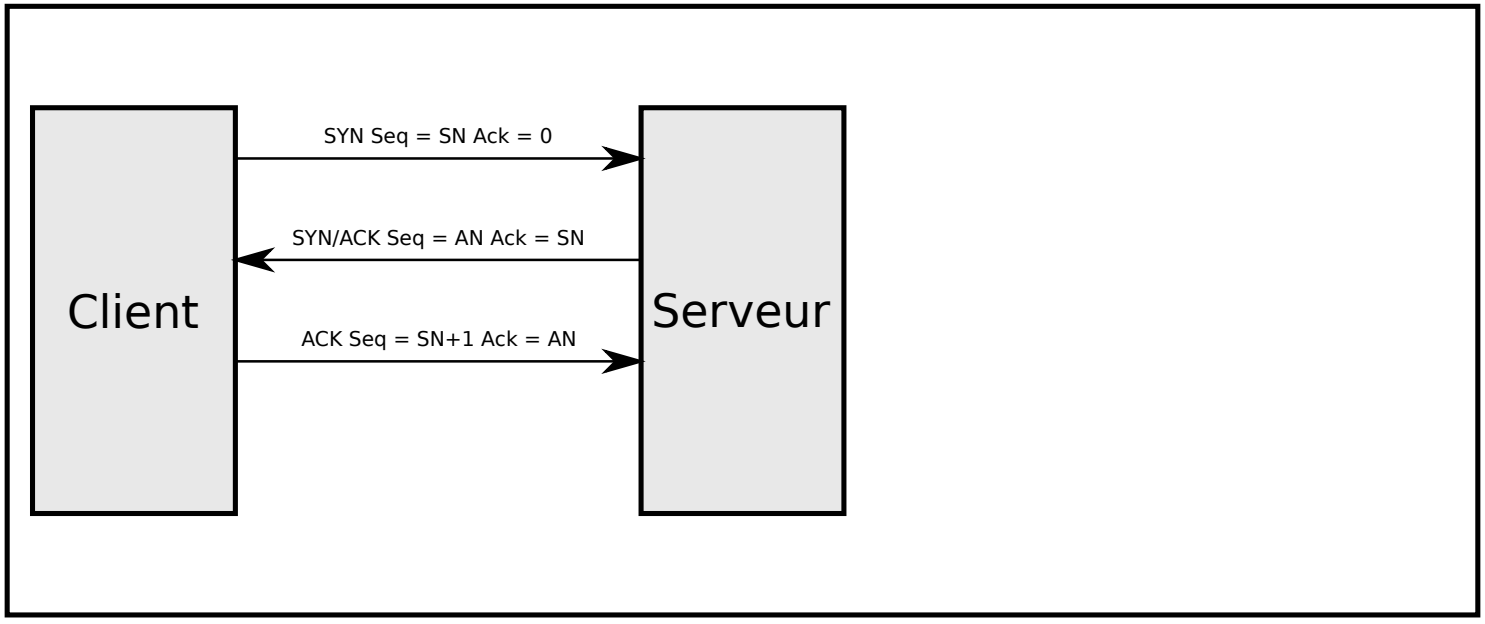
Why ?

Contourner les moyens d'authentification/autorisation

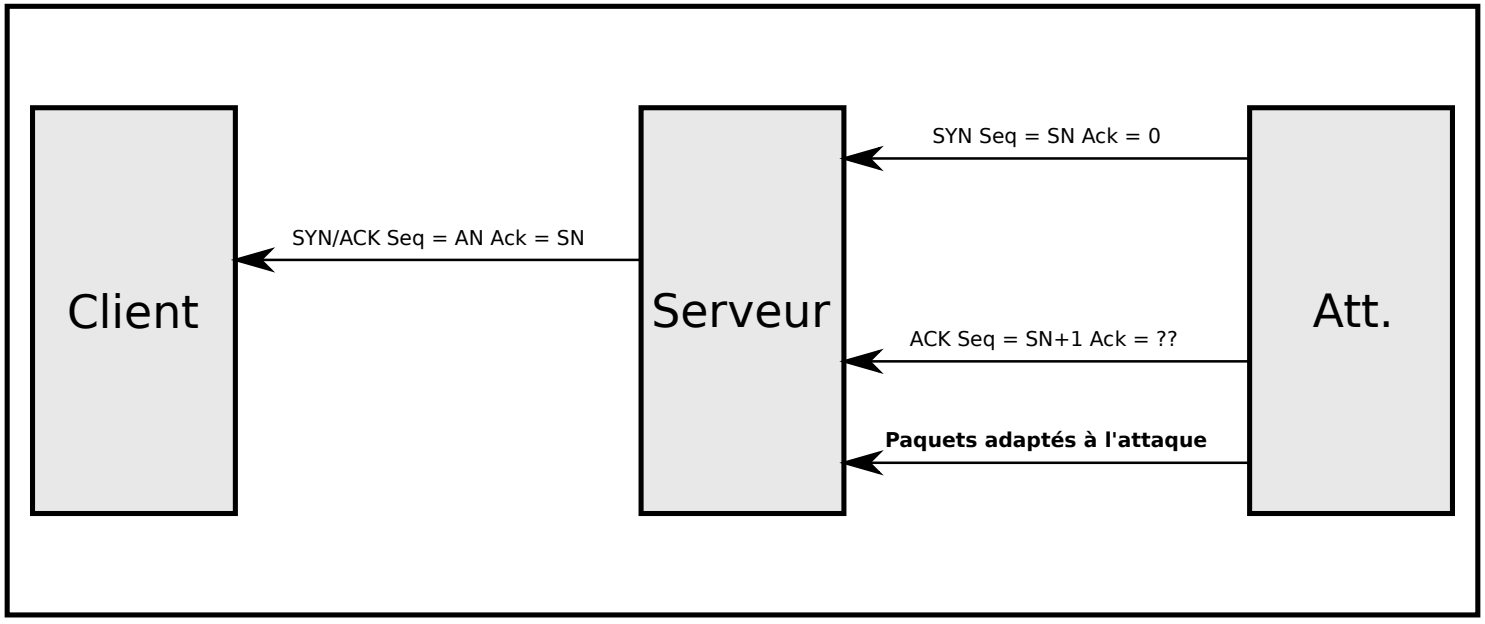
How ?

Pas évident ...

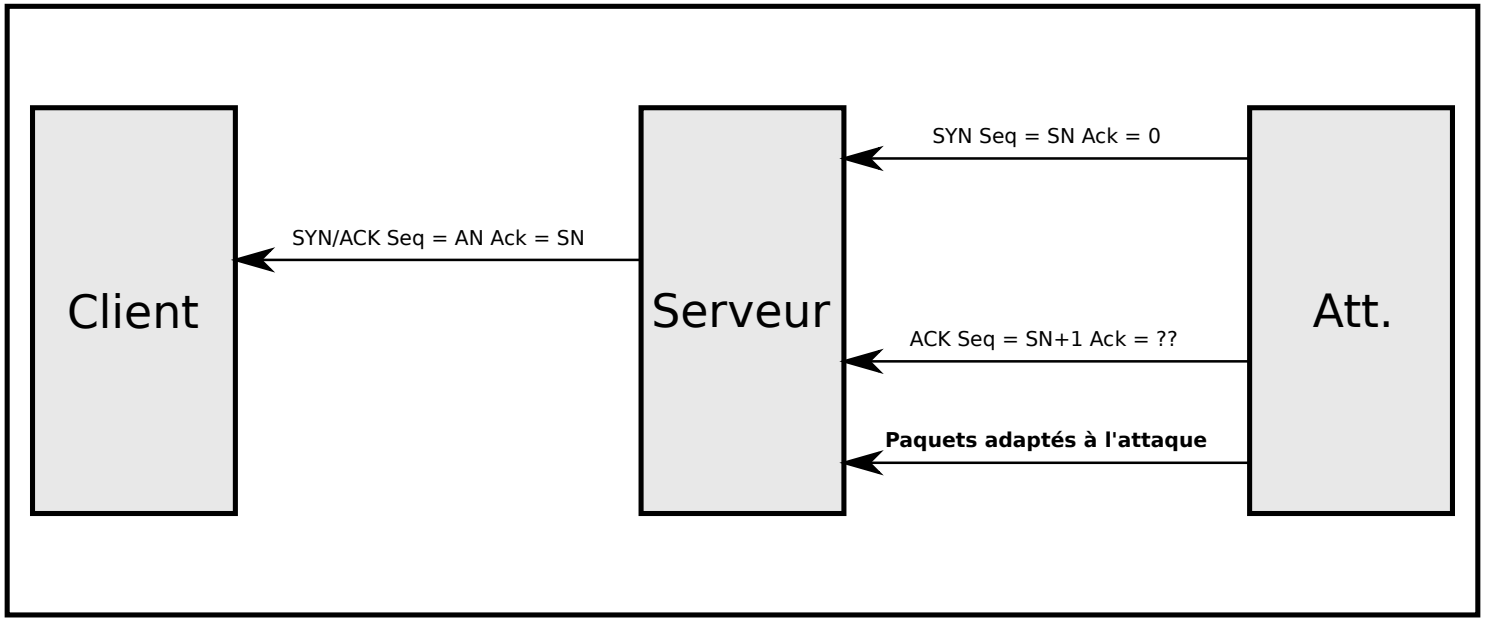
TCP Connection Spoofing (2/2)



TCP Connection Spoofing (2/2)



TCP Connection Spoofing (2/2)



Ça ne marche pas tel quel !

- Il faut connaître AN
  - En local on peut tenter l'écoute ou le détournement du trafic
  - En distant il faut trouver un moyen de le prévoir
- Le client va envoyer un RESET !

How ? L'attaque de Kevin Mitnick

Kevin Mitnick

Hacker de genie des années 90
Victimes : Pac. Bell, NorAD Command, Fujitsu, Motorola, etc.
Parmis les dix criminels les plus recherchés aux USA
Fait de la prison, maintenant co-fondateur de Defensive Thinking

Une époque dorée pour les hackers

On diffusait pas mal d'informations notamment par la commande `finger`
Il y avait la notion de noeuds de confiance /etc/rhosts (utilisé par `telnet`,`rsh`, `rcp`, `X`, etc.)

L'attaque de Tsutomu Shimomura (1995)

Attaque menée le jour de Noël 1994
Il apprend que l'utilisateur root de RIMMON est connecté sur OSIRIS sa cible
$\Rightarrow$ RIMMON est peut-être un noeud de confiance
Envoi de plusieurs SYN vers OSIRIS pour comprendre comment les AN sont générés
DoS sur RIMMON (SYN flood) puis TCP Connection Spoofing sur le port de `rsh`
Envoi un packet avec `echo + + > /etc/rhosts` qui s'exécute comme root ...

Et de nos jours ?

Découverte du réseau/services/connexions

Cartographie : cours RO
Plus compliqué mais faisable

Prédictions des numéros de séquence

Obtenus par des générateurs pseudo-aléatoires
À distance :

- Att. aux vieux systèmes comme NT !
- `nmap` → fingerprint OS puis recherche Internet :)

En local :

- Écoute (segment local, MAC flooding, ARP Spoofing) → AN

Et le payload ? `echo + + > /etc/rhosts` ?

Services “en r” fermés depuis longtemps ...
Mais en local on peut faire plus qu’envoyer un seul paquet !

Vol de session TCP

Limitations du connection spoofing

On peut envoyer des données mais pas recevoir les réponses
Le protocole au dessus (FTP, HTTP) peut demander une authentification

What ?

Émettre avec l'adresse IP d'autrui dans une connexion **déjà établie**

Pourquoi ça marche

On n'a pas besoin de deviner l'AN ou de s'authentifier
Pbs : SNs, ACKs, synchronisation
$\rightarrow$ Il faut être en local (au moins via un sniffer)

Cas le plus simple : MITM par ARP Spoofing déjà en place

Facile : une fois l'utilisateur authentifié on insère/modifie ce qu'on veut

Vol de session TCP : le Simple Active Hijack (1/2)

What ?

Désynchroniser les interlocuteurs au niveau des SNs

Why ?

Rendre la communication entre eux impossible et prendre la place

How ? Premier essai

Écouter pour obtenir le bon SN
Envoyer (au bon moment) un RST à A qui fermera la connexion
Envoyer des paquets à B avec l'adresse IP de A (vite)

Problèmes

B va générer des ACK et les envoyer à A ...
Qui va ... envoyer des RST (on est cuits)

Vol de session TCP : le Simple Active Hijack (2/2)

How ? L'attaque de Joncheray

Deux options :

- En début de connexion
  - Le client démarre une connexion avec le serveur
  - On coupe la connexion côté serveur (pas côté client)
  - On démarre la connexion côté serveur à nouveau

  $\Rightarrow$ Des nouveaux numéros de séquences sont générés

  Rq Il faut que le serveur accepte qu'une connexion plante
- À n'importe quel moment
  - On envoie des faux paquets au serveur avec les bons SN

  $\Rightarrow$ Le SN attendu par le serveur monte

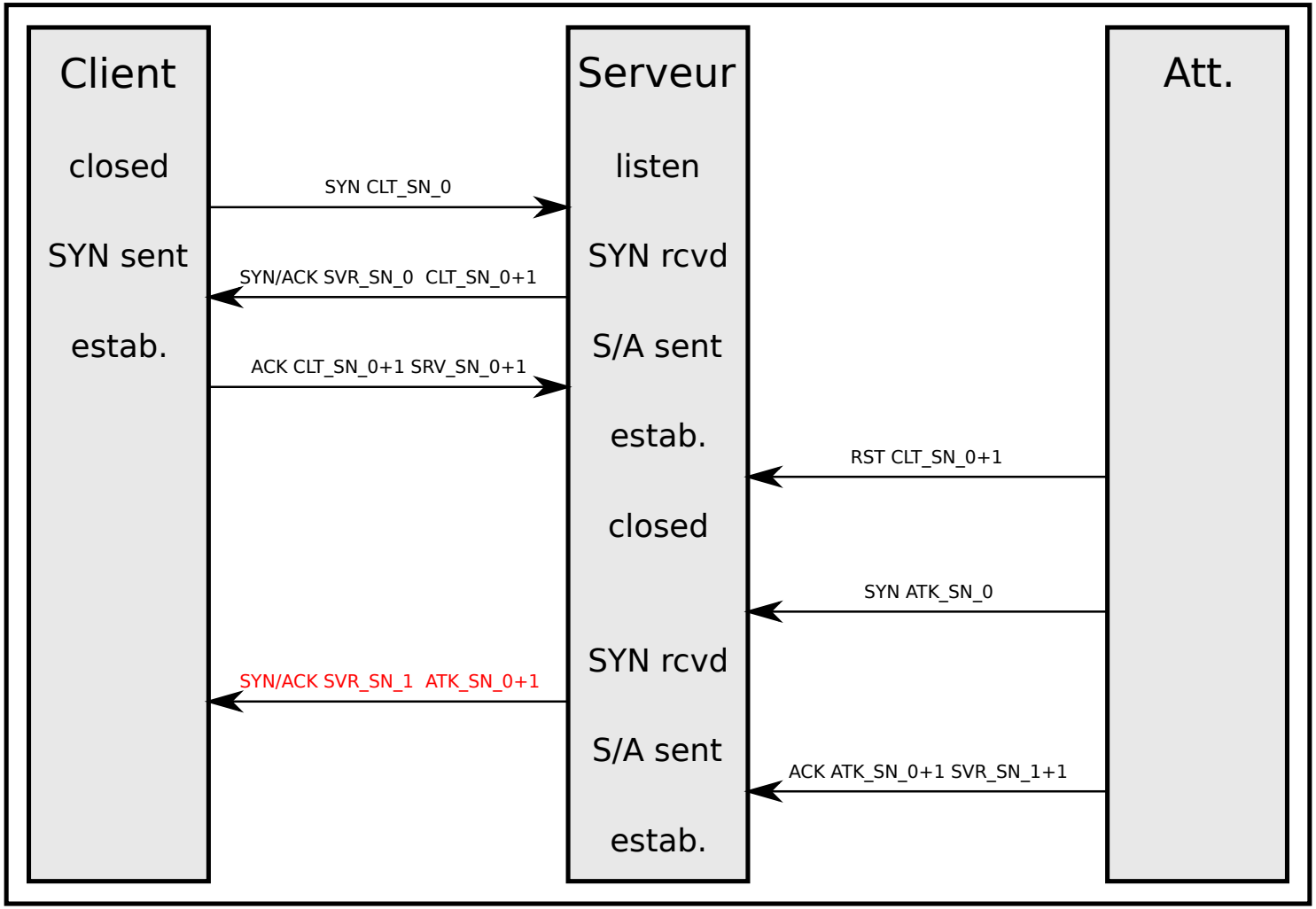
  Rq Il faut que ça ne fasse pas planter le serveur (plus dur)

Dans les deux cas on atteint une désynchronisation

$\Rightarrow$ Ils ne traitent pas leurs messages respectifs

$\Rightarrow$ L'attaquant traduit et remplace/introduit ce qu'il veut

Joncheray : en début de connexion



Joncheray : problèmes

ACK Storm

Paquet hors de la fenêtre TCP → ACK avec le SN et AN de récepteur
Même si c'est un ACK ! ! !

Si les interlocuteurs son désynchronisés chaque paquet de A est hors fenêtre pour B
Ce qui génère un ACK de B qui est hors fenêtre pour A
Ce qui génère un ACK de A qui est hors fenêtre pour B
...

ACKs pas renvoyés en cas de drop → processus fini (et auto-régulé)

Contres

Ingress filtering
Port Security

Fin !

Des questions ?