

SEGURIDAD DE LA INFORMACIÓN

TEMA 4 – PARTE 1

**SEGURIDAD Y PRIVACIDAD
EN APLICACIONES TELEMÁTICAS**

Indice del tema

- Otras herramientas de seguridad – *red team and blue team*
 - Herramientas de *red team*
 - Sistemas operativos
 - Herramientas
 - Herramientas de *blue team*
 - Seguridad en email: PGP y S/MIME
 - Conexión remota segura
 - Herramientas de cifrado
- Seguridad en pagos electrónicos
 - Conceptos generales
 - Protocolo SET
 - Protocolo Cybercash
 - Protocolo iKP
 - Protocolo Millicent

Indice del tema

- Privacidad de los usuarios en aplicaciones
 - Conceptos generales
 - Privacidad basada en esquemas avanzados de firma digital
 - Privacidad basada en protocolos criptográficos y de enrutamiento

OTRAS HERRAMIENTAS DE SEGURIDAD

- RED TEAM AND BLUE TEAM

Red Team

- *Red team* consiste en un equipo especializado en realizar actividades ofensivas
- Buscan vulnerabilidades y atacan a su propio sistema para:
 - Probar la eficacia de las soluciones y herramientas de seguridad y su robustez frente a ataques
 - Analizar las posibles fugas de información que puedan existir, y conexiones o redirecciones remotas
 - Determinar posibles comportamientos y técnicas de futuros atacantes contra su propio sistema
 - Etc.
- Metodología de trabajo:
 - Diseñan ataques específicos de acuerdo a la arquitectura de su sistema
 - Identifican herramientas de hacking para explotar vulnerabilidades
 - Ejecutan los ataques pre-diseñados
 - Extraen debilidades y vulnerabilidades para reportárselo al equipo Blue

Blue Team

- *Red team* consiste en un equipo especializado en realizar actividades defensivas
- Tratan, evitan o mitigan vulnerabilidades por:
 - Proporcionar herramientas, soluciones o estrategias defensivas
 - Analizan los informes detallados del equipo Red para identificar las vulnerabilidades y mitigarlas
 - Monitorizan las actividades del sistema y ofrecen planes de actuación
 - Etc.
- Metodología de trabajo:
 - Recopilan información del sistema y evalúan riesgos
 - Identifican soluciones de seguridad o diseñan nuevas soluciones para evitar riesgos
 - Monitorizan constantemente la seguridad del sistema por evaluar los eventos producidos
 - Trabajan en la continua mejora de la seguridad de un sistema
 - Etc.

RED TEAM

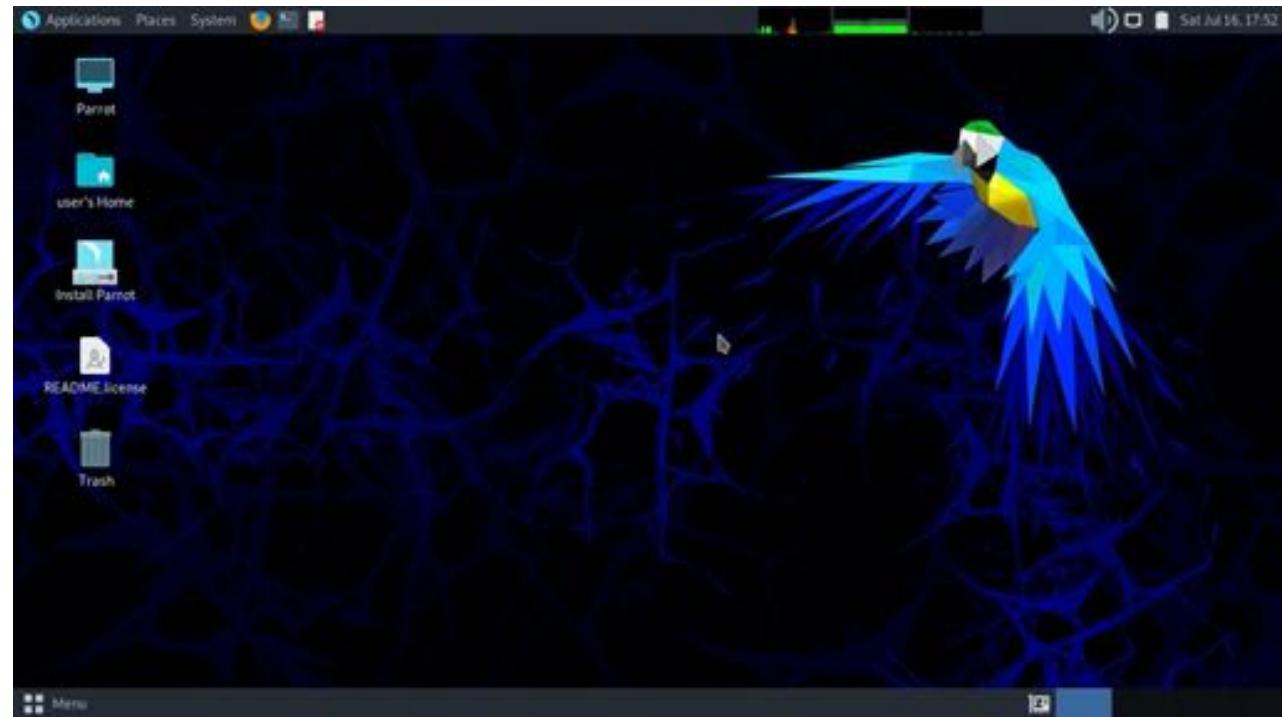
Kali linux

- Distribución Debian GNU/Linux diseñada principalmente para la auditoría y seguridad informática, ofrece un conjunto relevante de herramientas de seguridad



Parrot Security OS

- Distribución Debian GNU/Linux diseñada específicamente para realizar pruebas de penetración, evaluación y análisis de vulnerabilidades, análisis forense de sistemas, preservación del anonimato y análisis y prueba de criptografía

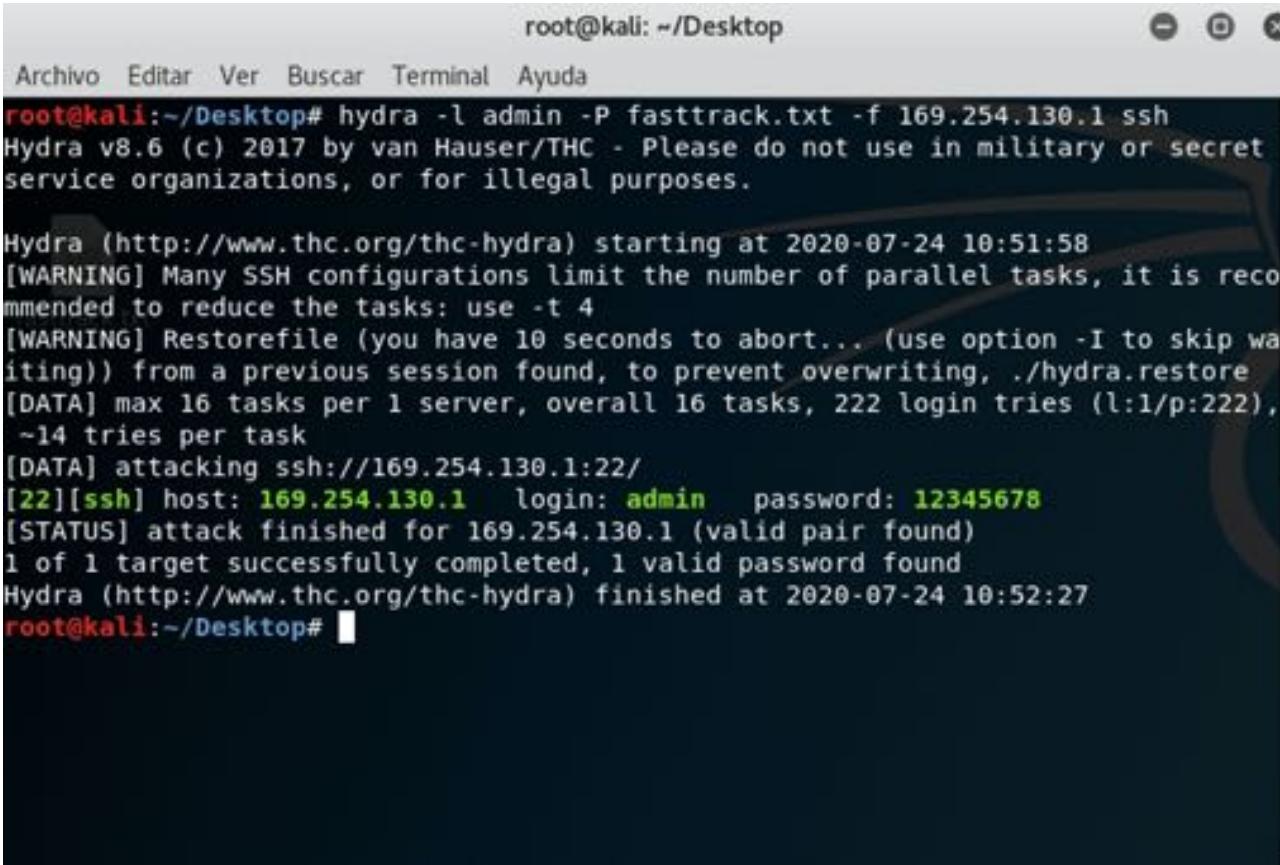


Algunas herramientas en Linux

- **Aircrack-ng** - herramienta para descifrar claves 802.11 TPA-PSA y WEP
- **Burpsuite** - herramienta integrada para probar aplicaciones web
- **Hydra** – herramienta para derivar credenciales de seguridad e iniciar sesión
- **John (the Ripper)** - herramienta derivar contraseñas
- **Maltego** – herramienta de inteligencia y forense
- **Metasploit framework** – un suite de herramientas para realizar pruebas de seguridad extremadamente flexible
- **Nmap** - herramienta de mapeo de redes
- **Owasp-ZAP** - herramienta de prueba de aplicaciones web
- **Wireshark** - La principal herramienta de análisis de protocolos de red
- **Lego** (la versión actualizada de Sparta) - herramienta de pruebas de penetración de la infraestructura de red
- **Scapy** – herramienta de manipulación de paquetes de red
- Etc.

Hydra - fuerza bruta

- Es un cracker de inicio de sesión que se basa en lanzar un ataque fuerza bruta por usar un diccionario de posibles contraseñas



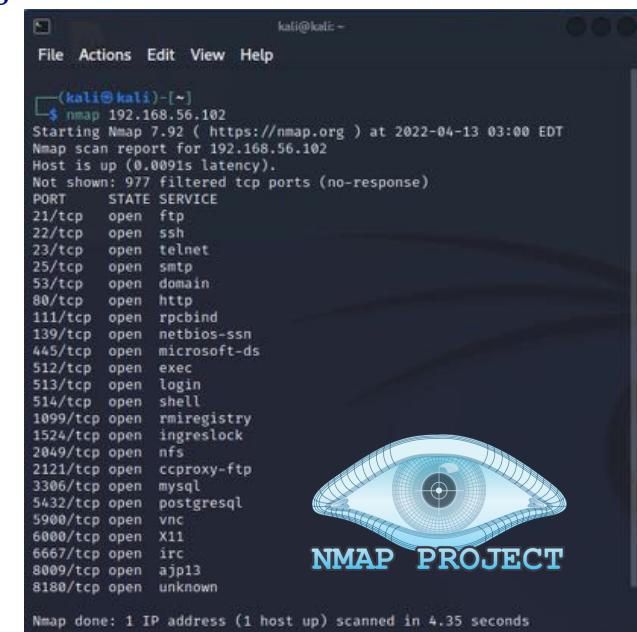
A terminal window titled "root@kali: ~/Desktop" showing the output of a Hydra attack. The terminal has a standard window title bar with minimize, maximize, and close buttons. The menu bar includes "Archivo", "Editar", "Ver", "Buscar", "Terminal", and "Ayuda". The main area displays the command and its execution:

```
root@kali:~/Desktop# hydra -l admin -P fasttrack.txt -f 169.254.130.1 ssh
Hydra v8.6 (c) 2017 by van Hauser/THC - Please do not use in military or secret
service organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2020-07-24 10:51:58
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recom
mended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip wa
iting)) from a previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 222 login tries (l:1/p:222),
~14 tries per task
[DATA] attacking ssh://169.254.130.1:22/
[22][ssh] host: 169.254.130.1    login: admin    password: 12345678
[STATUS] attack finished for 169.254.130.1 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2020-07-24 10:52:27
root@kali:~/Desktop#
```

Nmap y Zenmap - rastrear y recopilar información

- Nmap permite el *fingerprinting*
 - El fingerprinting consiste recolectar información de un sistema y conocer sus vulnerabilidades, y suele dejar rastro
- **Nmap:** herramienta de escaneo de puertos, rangos de IP, información del sistema operativo
 - Los análisis se realizan en línea de comando o mediante una GUI (Zenmap)
 - Compatible con Windows, Linux, MacOS y Solaris
 - Opciones de nmap:
 - **nmap IP/localhost:** puertos abiertos y servicios
 - **nmap -sP IP/localhost:** realizar un simple ping
 - **nmap -p T:1-65535 -T4 IP/localhost:** puertos (o un rango) en modo agresivo (-T4)
 - **nmap -p T:X-Y,Z -T4 IP/localhost:** puertos concretos
 - **nmap -sV -T4 IP/localhost:** puertos, sus servicios y versiones
 - ...

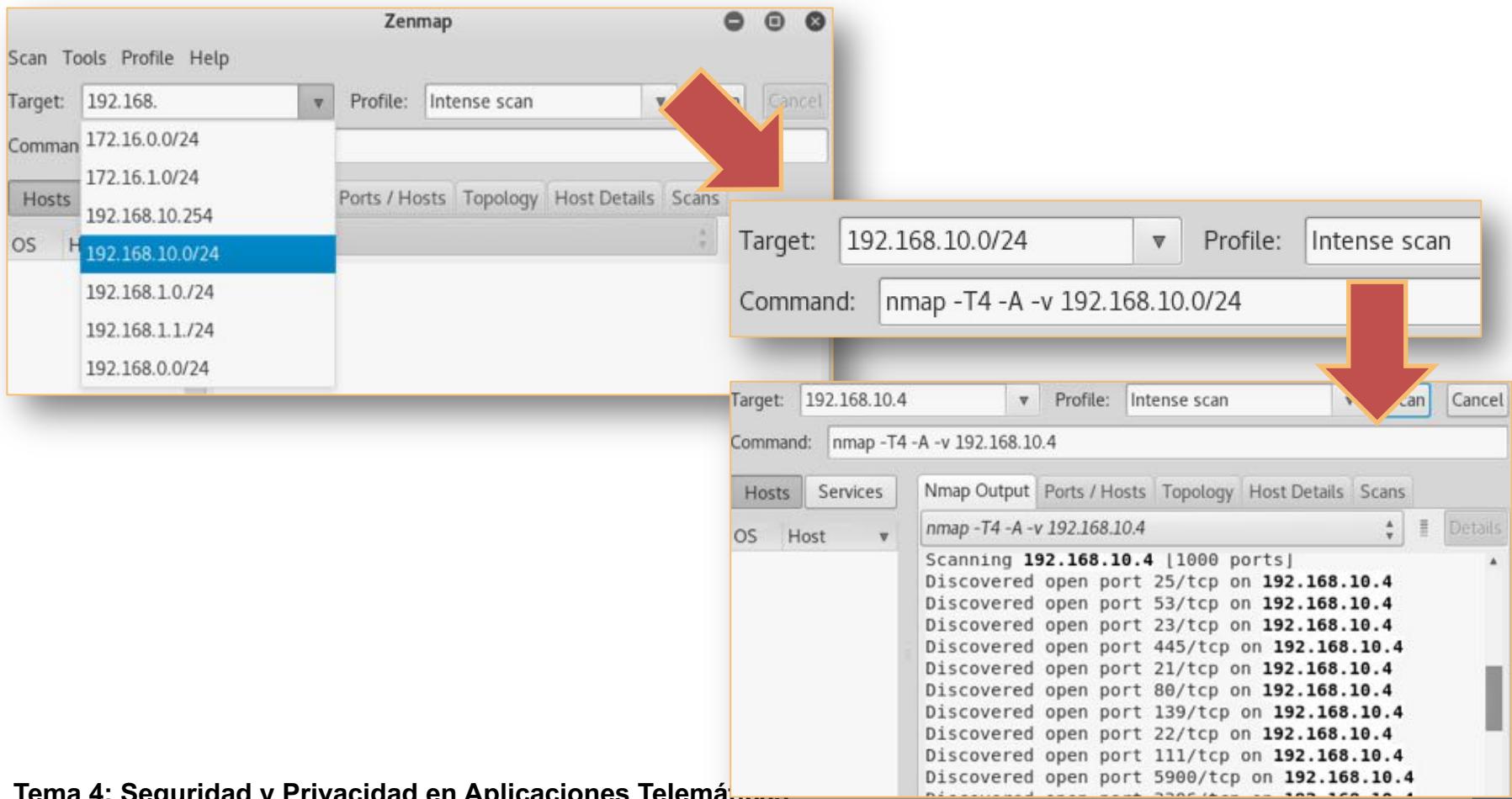


```
(kali㉿kali)-[~] $ nmap 192.168.56.102
Starting Nmap 7.92 ( https://nmap.org ) at 2022-04-13 03:00 EDT
Nmap scan report for 192.168.56.102
Host is up (0.0091s latency).
Not shown: 977 filtered tcp ports (no-response)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  cproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown

Nmap done: 1 IP address (1 host up) scanned in 4.35 seconds
```

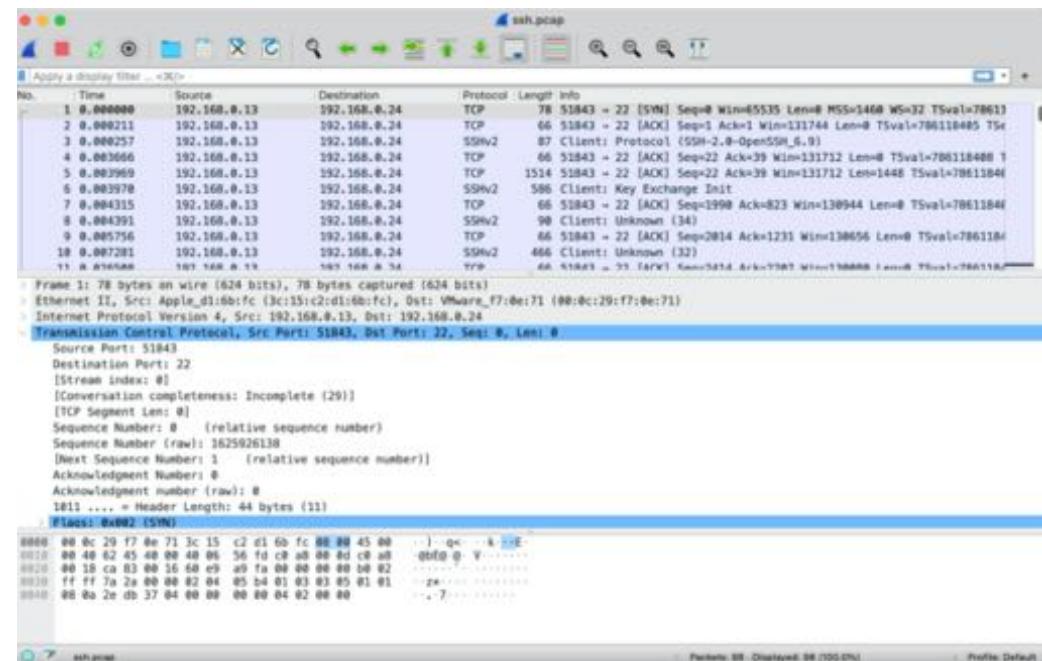
Nmap y Zenmap - rastrear y recopilar información

- Zenmap: open-source cuyo objetivo es ejecutar instrucciones nmap mediante una interfaz de usuario



Wireshark - captura de tráfico de red o sniffing

- Disector de red a cargo de analizar protocolos de red
 - Multiplataforma, y funciona en Windows, Linux, macOS, Solaris, FreeBSD, NetBSD y muchos otros
 - Captura en vivo y análisis fuera de línea
 - Soporte de descifrado para muchos protocolos, incluyendo IPsec, ISAKMP, Kerberos, SNMPv3, SSL/TLS, WEP y WPA/WPA2
 - Capacidad para exportar en XML, PostScript®, CSV o texto sin formato



Scapy - manipulación de paquetes

- Scapy es una librería desarrollada en Python, que tiene su propio interprete de línea de comandos, con la capacidad de crear, modificar, enviar y capturar paquetes de red

The screenshot shows the Scapy v2.4.3 terminal interface. It starts with a welcome message from Lao-Tze: "Craft packets like it is your last day on earth." Below this, a large ASCII art representation of the Chinese character for 'Scapy' is displayed. The terminal then shows a command being entered: `>>> send(IP(src="145.167.7.9",dst="169.254.130.252"))`. The response indicates that one packet was sent.

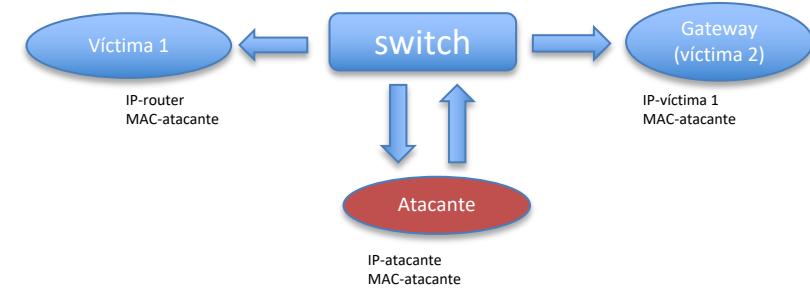
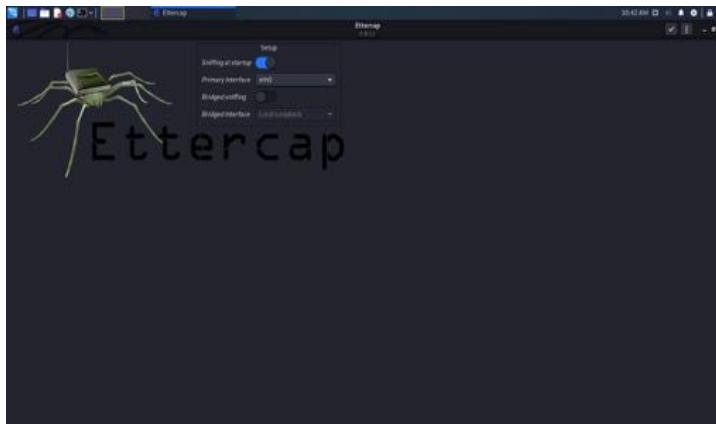
```
Scapy v2.4.3
File Actions Edit View Help
kaliKali:$ sudo scapy
INFO: Can't import PyX. Won't be able to use psdump() or pdfdump().
WARNING: No route found for IPv6 destination :: (no default route?)

          aSPY//YASa
          apyyyyCV/////////YCa
          sY/////////YSpCs  scpCY//Pp
          ayp ayyyyySCP//Pp      syY//C
          AYAsAYYYYYYYY///Ps      cY//S
          pCCCCY//p      cSSp y//Y
          SPPP//a      pP///AC//Y
          A//A      cyP///C
          p///Ac      sC///a
          P///YCpc      A//A
          sccccp///pSP//p      p//Y
          sY/////////y caa      S//P
          cayCayP//Ya      pV/Ya
          sY/PsY///YCc      aC//Yp
          sc sccaCY//PCyapaapYCP//YSs
          spCPY////VSpS
          ccaacs
                                         using IPython 7.14.0
>>> send(IP(src="145.167.7.9",dst="169.254.130.252"))
.
Sent 1 packets.
>>> 
```

- Compatible con múltiple plataforma: Linux, Mac OS x y Windows

Ettercap-graphical

- Ettercap es una herramienta que intercepta las comunicaciones mediante ARP spoofing (envenenamiento de la red por ARP) creando ataques de tipo “Man-in-the-Middle”
 - Es una herramienta capaz de analizar todo tráfico atrayendo todo el tráfico de red a la máquina del atacante en lugar del router
 - Básicamente, lo que hace la máquina del atacante es anunciar que su “IP” es la del router real pero la dirección MAC es justo de la máquina del atacante



MITRE ATT&CK

- MITRE es un repositorio (muy conocido) que ofrece un conjunto de técnicas de ataque y tácticas en diferentes ámbitos, a nivel de empresa, para sistemas móviles y sistemas de control industrial

<https://attack.mitre.org>

BLUE TEAM

Seguridad en email: PGP y S/MIME

- El correo electrónico es la aplicación más ampliamente utilizada en la gran mayoría de los entornos distribuidos
- El crecimiento en su uso ha conllevado una mayor necesidad de seguridad, y, más concretamente, de la integración de servicios de **autenticación y confidencialidad**
- Entre las soluciones de seguridad en e-mail disponibles en la actualidad, hay dos que destacan por su amplio uso:
 - PGP
 - S/MIME



PGP (Pretty Good Privacy)

- PGP es una solución diseñada por Phil Zimmerman en 1991
 - Básicamente, PGP consiste en seleccionar algoritmos criptográficos ya existentes en integrarlas en una única aplicación SW independiente del S.O.
 - Integra los algoritmos **RSA**, **DSS**, **Diffie-Helmann (ElGamal)**, **CAST-128**, **IDEA**, **3DES**, **SHA-1**
 - Proporciona servicios de **autenticidad y confidencialidad** que se pueden usar para:
 - **aplicaciones de e-mail**
 - **almacenamiento de ficheros**
 - Existen versiones libres para Windows, UNIX y Mac, además de versiones comerciales
- Incluso IETF ha realizado avances con PGP:
 - *RFC 3156: MIME Security with OpenPGP*



PGP (Pretty Good Privacy)

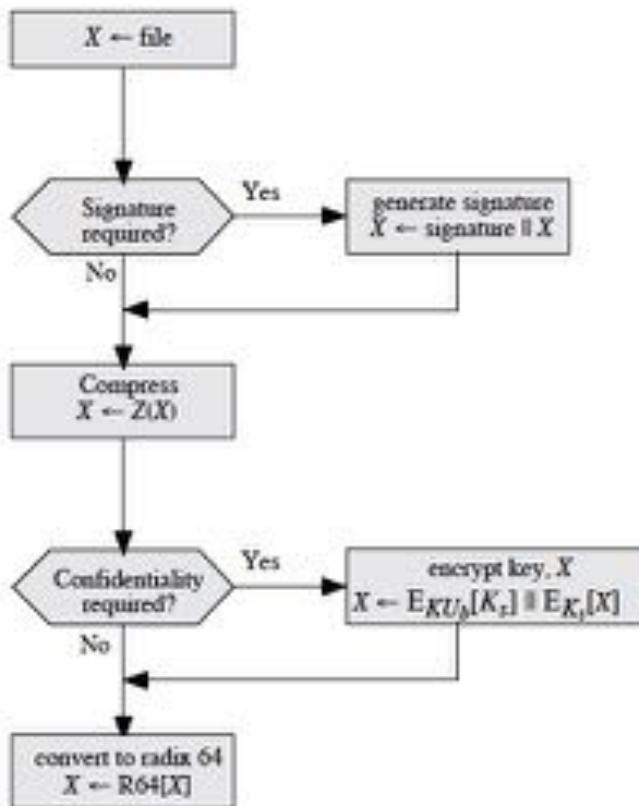
- Las operaciones de PGP incluyen, además de autenticación y confidencialidad, las operaciones de compresión y de compatibilidad de e-mail

Function	Algorithms Used	Description
Digital signature	DSS/SHA or RSA/SHA	A hash code of a message is created using SHA-1. This message digest is encrypted using DSS or RSA with the sender's private key and included with the message.
Message encryption	CAST or IDEA or Three-key Triple DES with Diffie-Hellman or RSA	A message is encrypted using CAST-128 or IDEA or 3DES with a one-time session key generated by the sender. The session key is encrypted using Diffie-Hellman or RSA with the recipient's public key and included with the message.
Compression	ZIP	A message may be compressed, for storage or transmission, using ZIP.
Email compatibility	Radix 64 conversion	To provide transparency for email applications, an encrypted message may be converted to an <u>ASCII string using radix 64 conversion</u> .

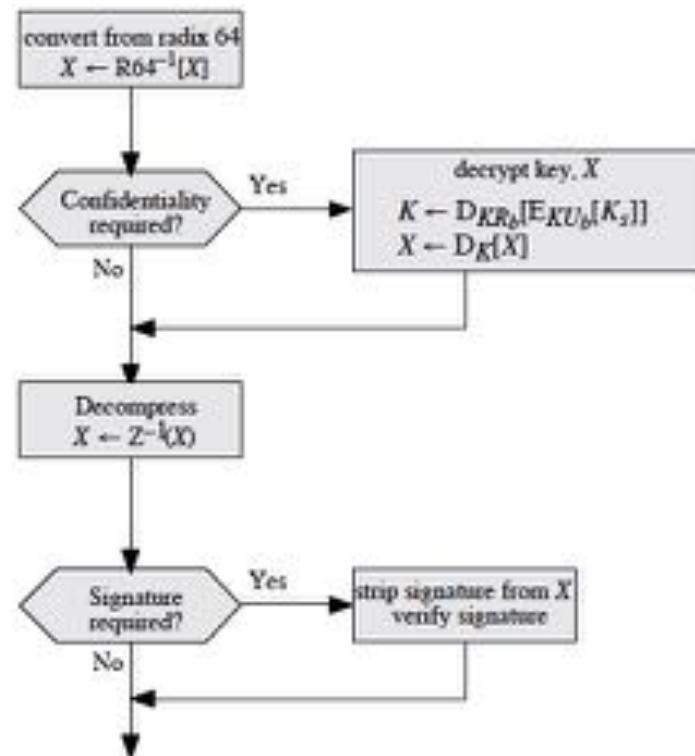


PGP (Pretty Good Privacy)

- Esquema de transmisión y recepción de mensajes PGP:

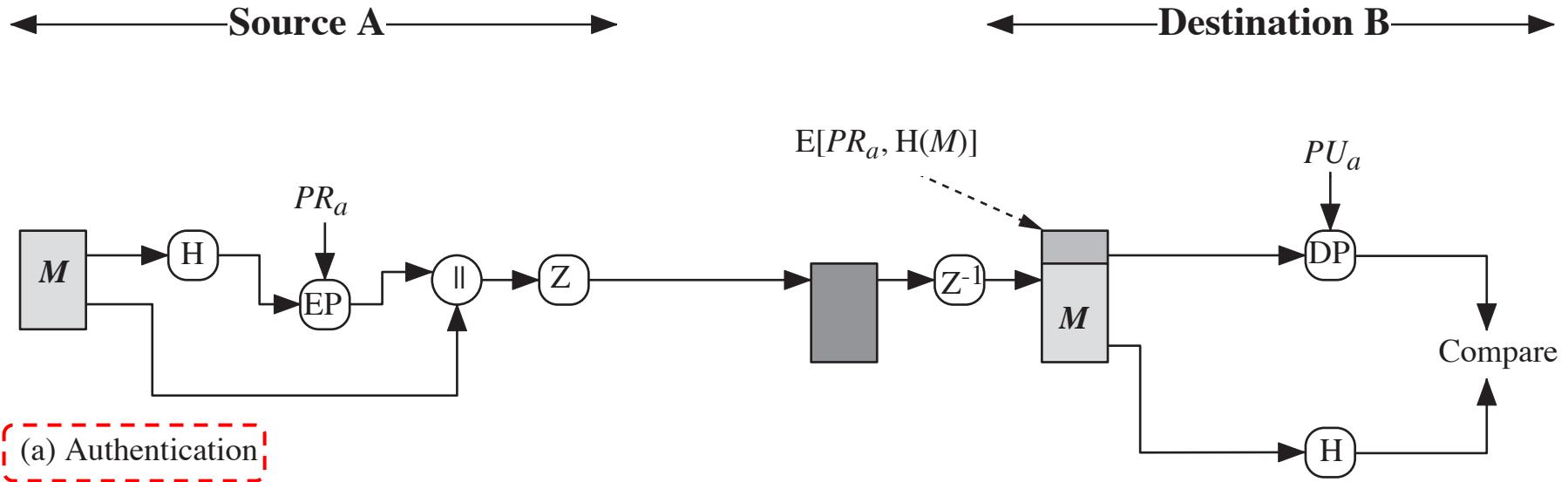


(a) Generic Transmission Diagram (from A)



(b) Generic Reception Diagram (to B)

PGP (Pretty Good Privacy)



K_s = session key used in symmetric encryption scheme

PR_a = private key of user A, used in public-key encryption scheme

PU_a = public key of user A, used in public-key encryption scheme

EP = public-key encryption

DP = public-key decryption

EC = symmetric encryption

DC = symmetric decryption

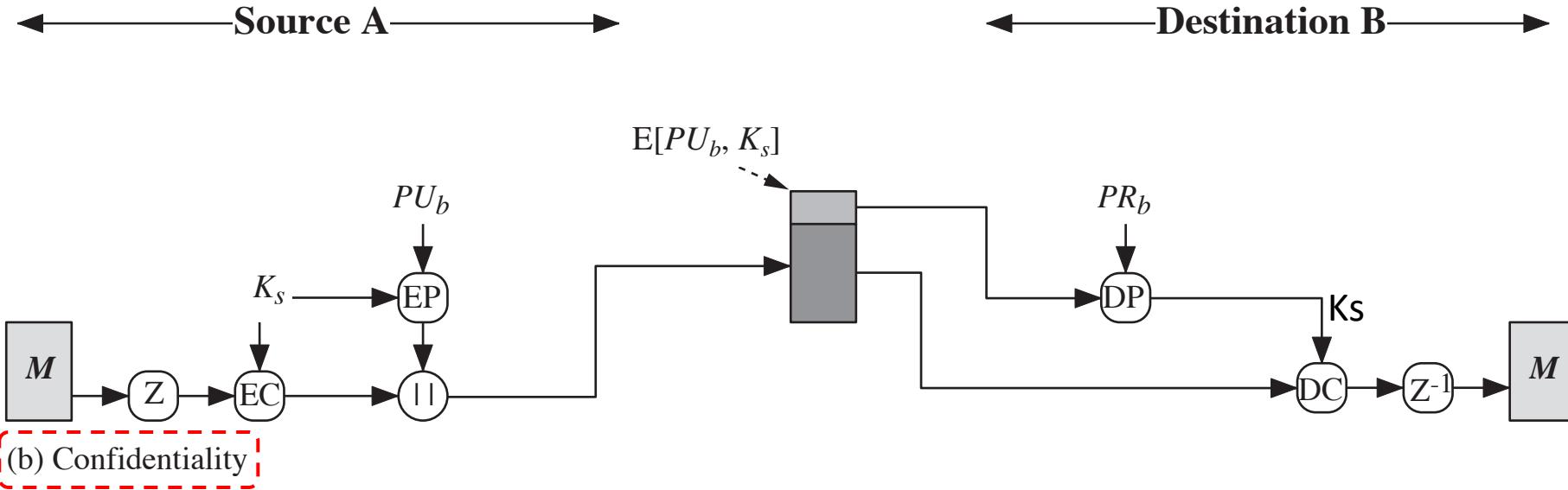
H = hash function

\parallel = concatenation

Z = compression using ZIP algorithm

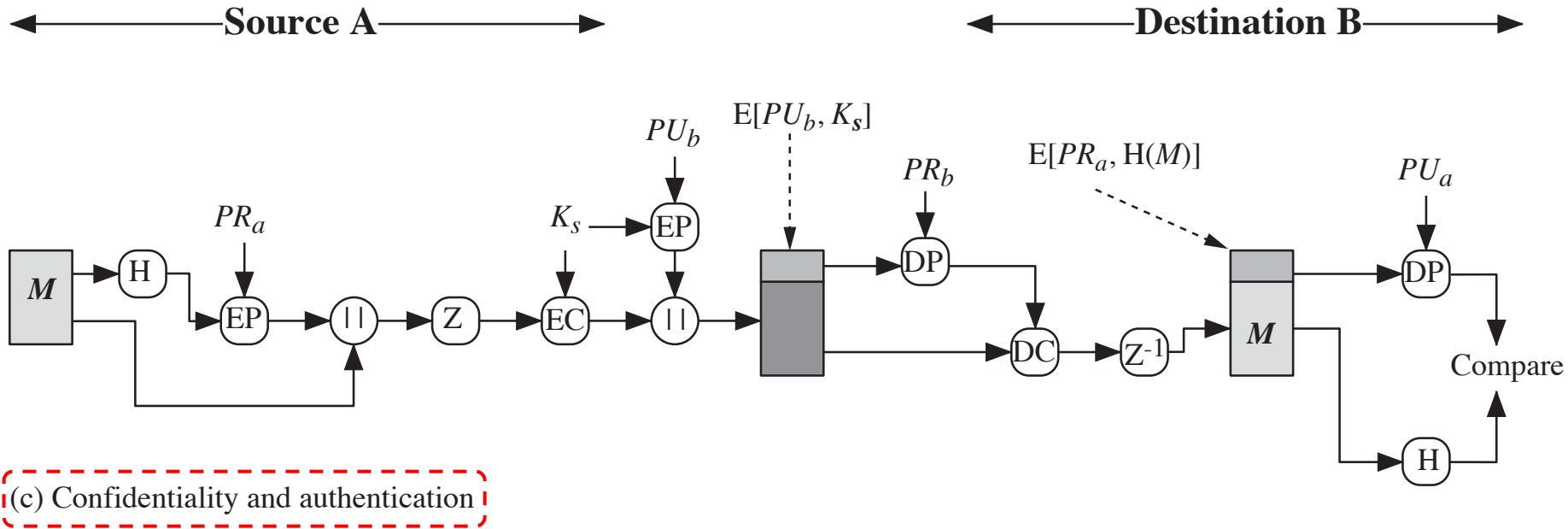
R64 = conversion to radix 64 ASCII format

PGP (Pretty Good Privacy)



K_s = session key used in symmetric encryption scheme
 PR_a = private key of user A, used in public-key encryption scheme
 PU_a = public key of user A, used in public-key encryption scheme
EP = public-key encryption
DP = public-key decryption
EC = symmetric encryption
DC = symmetric decryption
H = hash function
|| = concatenation
Z = compression using ZIP algorithm
R64 = conversion to radix 64 ASCII format

PGP (Pretty Good Privacy)



K_s = session key used in symmetric encryption scheme

PR_a = private key of user A, used in public-key encryption scheme

PU_a = public key of user A, used in public-key encryption scheme

EP = public-key encryption

DP = public-key decryption

EC = symmetric encryption

DC = symmetric decryption

H = hash function

\parallel = concatenation

Z = compression using ZIP algorithm

R64 = conversion to radix 64 ASCII format

PGP (Pretty Good Privacy)

- PGP proporciona, para cada usuario U , dos estructuras de datos:
 - **private-key ring**: para almacenar los pares <clave pública, clave privada> del propio usuario U
 - **public-key ring**: para almacenar las claves públicas de los otros usuarios con los que U se comunica

Private-Key Ring

Timestamp	Key ID*	Public Key	Encrypted Private Key	User ID*
•	•	•	•	•
•	•	•	•	•
•	•	•	•	•
T_i	$PU_i \bmod 2^{64}$	PU_i	$E(H(P_i), PR_i)$	User i
•	•	•	•	•
•	•	•	•	•
•	•	•	•	•

Public-Key Ring

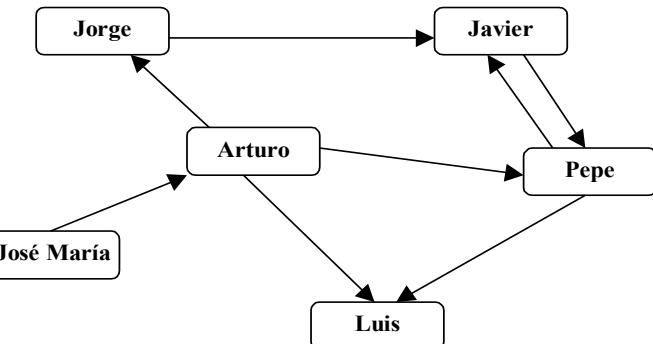
PGP (Pretty Good Privacy)

- Cada línea de la tabla del public-key ring puede considerarse en sí misma como un tipo de **certificado digital** (certificado de clave pública) **no estándar**
 - PGP no usa el estándar X.509, sino un formato propio



Public-Key Ring								
Timestamp	Key ID*	Public Key	Owner Trust	User ID*	Key Legitimacy	Signature(s)	Signature Trust(s)	
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	
T _i	$PU_i \bmod 2^{64}$	PU_i	trust_flag _i	User i	trust_flag _i			
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	

- Tampoco basa su funcionamiento en la existencia de una PKI jerárquica de Autoridades de Certificación, sino en un **modelo de PKI en malla**
 - o sea, no existen Autoridades de Certificación al uso, pero **cada usuario del sistema puede emitir certificados** al respecto de las claves públicas de los demás usuarios
 - de ahí los valores de confianza (**trust**) incluidos en el public-key ring



PGP con OpenPGP

- El OpenPGP Working Group se creó en 1997 y gracias en parte al Internet Engineering Task Force para definir el estándar
 - OpenPGP es una aplicación estándar y libre que permite cifrar emails usando criptografía de clave pública y un específico formato
- Está basado en el PGP original



[Docs] [txt|pdf] [draft-ietf-openpg...] [Diff1] [Diff2] [Errata]

Updated by: 5581 PROPOSED STANDARD
Network Working Group Errata Exist
Request for Comments: 4880 J. Callas
Obsoletes: 1991, 2440 PGP Corporation
Category: Standards Track L. Donnerhacke
IKS GmbH
H. Finney
PGP Corporation D. Shaw
R. Thayer
November 2007

OpenPGP Message Format

Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Abstract

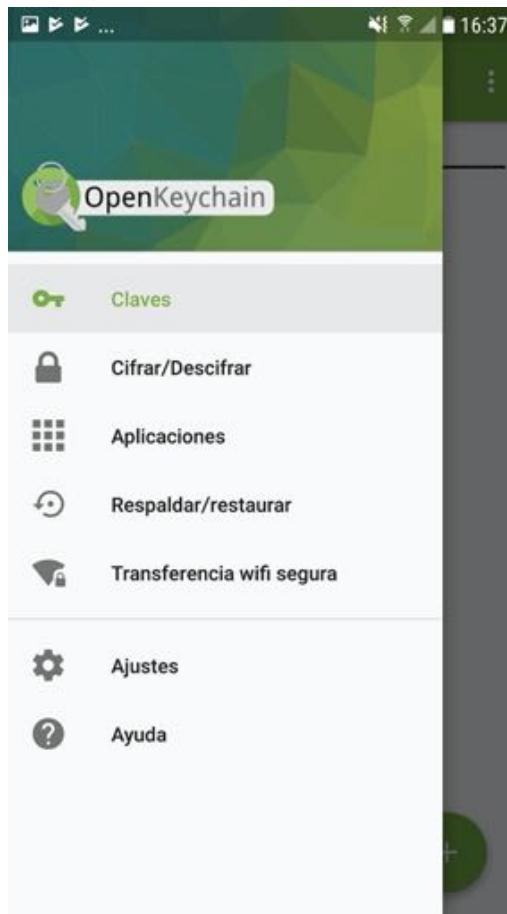
This document is maintained in order to publish all necessary information needed to develop interoperable applications based on the OpenPGP format. It is not a step-by-step cookbook for writing an application. It describes only the format and methods needed to read, check, generate, and write conforming packets crossing any network. It does not deal with storage and implementation questions. It does, however, discuss implementation issues necessary to avoid security flaws.

OpenPGP software uses a combination of strong public-key and symmetric cryptography to provide security services for electronic communications and data storage. These services include confidentiality, key management, authentication, and digital signatures. This document specifies the message formats used in OpenPGP.

PGP con OpenPGP

- Existen varias aplicaciones que soportan OpenPGP:
 - Windows
 - Outlook: gpg4o1, Gpg4win, p=p
 - Thunderbird: enigmail
 - Mac
 - Apple mail: GPGTools
 - Mutt
 - Thunderbird: enigmail
 - Android
 - K-9 mail: Openkeychain
 - P=p
 - R2Mail2
 - iOS
 - iPGMail
 - Linux
 - Evolution: Seahorse
 - Kmail:Kleopatra
 - Mutt
 - Thunderbird: enigmail
 - Solaris
 - Mutt
 - Thunderbird: enigmail
 -

OpenKeychain



The screenshot shows the "Configuración" (Configuration) screen of the OpenKeychain app. It lists several settings with their current status (indicated by blue toggle switches):

- Servidores de claves OpenPGP: Busca claves en los servidores de claves OpenPGP seleccionados (protocolo HKP). Enabled.
- Administrar servidores de claves OpenPGP: 3 servidores de claves; preferido: [hkps://keyserver.ubuntu.com](https://keyserver.ubuntu.com).
- keybase.io: Busca claves en keybase.io. Enabled.
- Web Key Directory: Buscar clave usando Web Key Directory. Enabled.
- Habilitar Tor: Requiere que Orbot esté instalado. Enabled.
- Habilitar otro proxy: Enabled.
- Servidor proxy: 127.0.0.1
- Puerto de proxy: 8118
- Tipo de proxy: HTTP

Texto y ficheros

iPGMail

Carrier 1:39 PM

Done **Create Private Key** **Create**

Passphrase
Confirm Passphrase

Key Size:

Expiration:

Real Name: John Q. Smith

Email Addr: user@domain.com

Carrier 1:39 PM

Edit **Public** **Private** **+**

Public Keys

0	1	2	3	4	5	6	7	8	9	C	F	I	K	N	Q	T	W	Z
adele-en <adele-de@gnupg.de> 4D486CC8 RSA(2048) 2013-07-08 2017-07-08																		
boo <boo@hoo.com> D4C8EB20 RSA(2048) 2013-07-23 (no exp)																		
foobar <foo@bar.com> CC33E7CC RSA(2048) 2013-07-29 (no exp)																		
FooFoo <foo@bar.com> 818F5EE0 RSA(1024) 2012-01-03 (no exp)																		
Wyllys <wyllys@me.com> 47E3234C RSA(2048) 2011-06-11 (no exp)																		

Keys **Compose** **Decode** **Files** **Settings**

Carrier 1:40 PM

Clear **Sign** **Encrypt** **Both** **Upload**

From: foobar <foo@bar.com>

To: info@ipgmail.com

Attach: Select Attachments

Great job!

Keys **Compose** **Decode** **Files** **Settings**

Carrier 11:05 AM

Done **Public Key Details**

Public Key Info

Key ID: 47E3234C
User ID: Wyllys <wyllys@me.com>
Fingerprint: A5D0 EFBC 4A52 B587 E68A 2451 7FEA 1CCB 47E3 234C
Created: 2011-06-11
Expiration:
Algorithm: RSA (enc/sign)
Image: N/A
Key Len: 2048

UID Info

Wyllys Ingersoll <wyllys@gmail.com>
47E3234C 2011-06-11 -----

UID Info

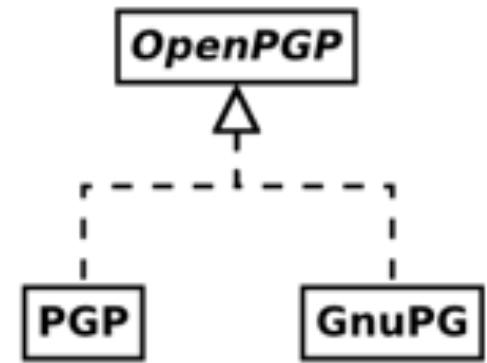
iPGMail Support <info@ipgmail.com>
47E3234C 2011-09-21 -----

UID Info

Wyllys <wyllys@me.com>

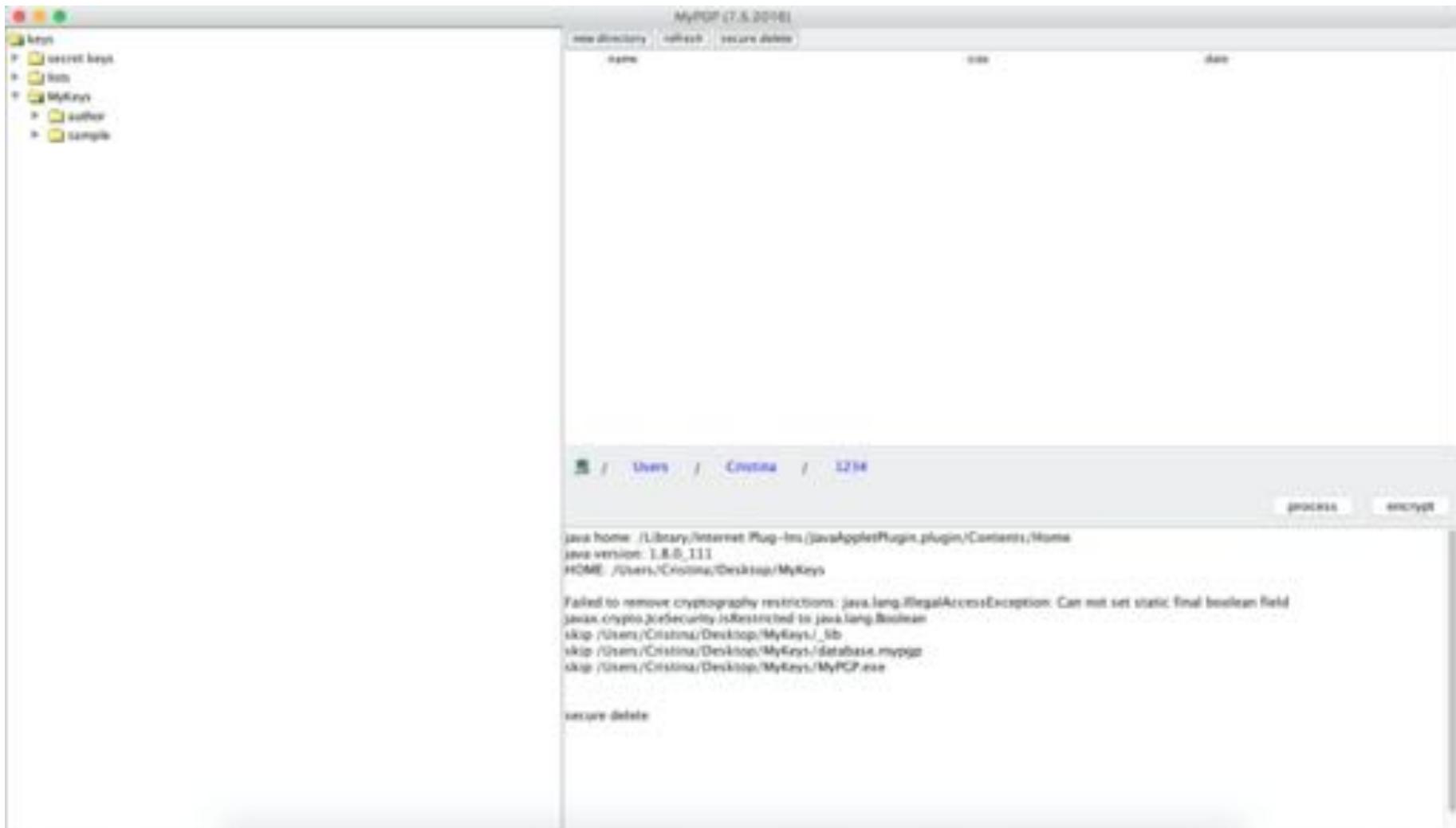
GnuPG

- GnuPG es una implementación del estándar OpenPGP que deriva del software criptográfico PGP desarrollado por Philip Zimmermann
- Con GnuPG se puede realizar las siguientes acciones:
 - gestionar y generar claves públicas y privadas
 - visualizar y distribuir las claves públicas, exportándolas e importándolas
 - cifrar y descifrar documentos
 - firmar y validar documentos



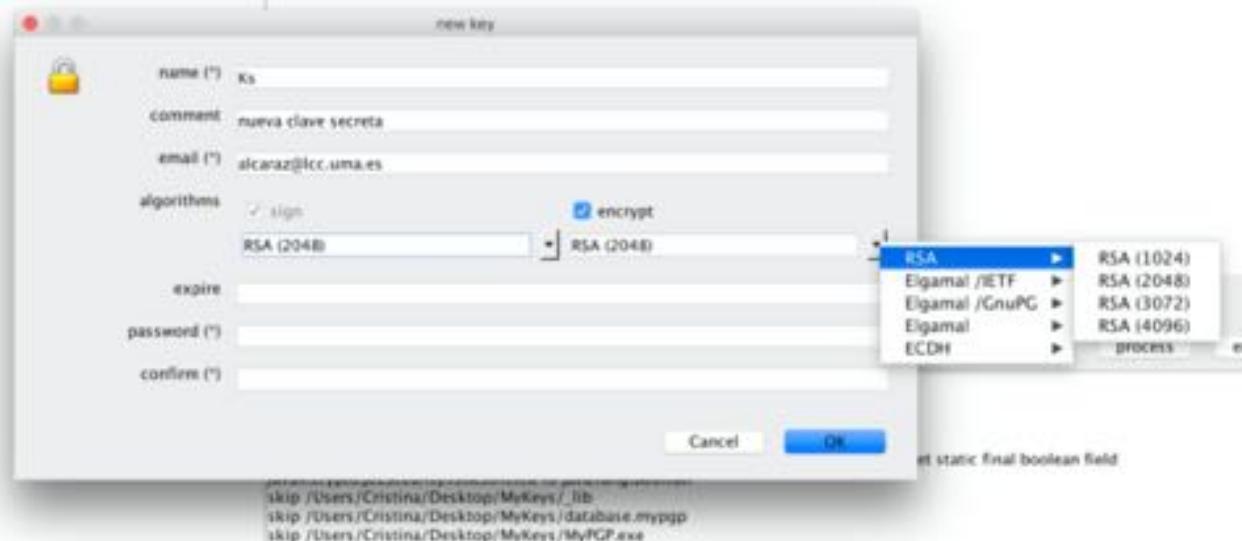
<https://www.gnupg.org>

MyPGP



MyPGP

Creando una pareja de claves (privada y pública):



The main interface shows a tree view of keys and lists. The 'keys' section contains:

- secret keys**:
 - Ks <calcaraz@lcc.uma.es> (nueva clave secreta)
 - Publius Cornelius Scipio Africanus <publius@roma.imperium> (password: password)
- lists**: (empty)
- sample**: (empty)
- MyKeys**:
 - Ks <calcaraz@lcc.uma.es> (nueva clave secreta)
 - author
 - sample

The 'secret keys' node is expanded, showing two entries:

- Ks <calcaraz@lcc.uma.es> (nueva clave secreta)**:
 - [28.11.2016] Ks <calcaraz@lcc.uma.es> (nueva clave secreta)
 - fingerprint: d539 9c93 8305 846f 8ffe ff43 e208 be32 8a8d c86e
 - RSA (2048) / RSA (2048)
 - encrypt: AES 256, AES 192, AES 128, CAST5 (128), 3DES (112/168)
- Publius Cornelius Scipio Africanus <publius@roma.imperium> (password: password)**:
 - [12.12.2014] Publius Cornelius Scipio Africanus <publius@roma.imperium> (password: password)
 - fingerprint: ace0 b6a1 38a3 1cff 91e4 0f87 cd60 b3e5 f241 1ec2
 - RSA (2048) / RSA (2048)
 - encrypt: AES 256, AES 192, AES 128, CAST5 (128), 3DES (112/168)

Below the tree view, there is a file browser window titled 'keys'. It shows a folder structure under 'secret keys':

- secret keys**:
 - Ks <calcaraz@lcc.uma.es> (nueva clave secreta)**:
 - [28.11.2016] Ks <calcaraz@lcc.uma.es> (nueva clave secreta)
 - fingerprint: d539 9c93 8305 846f 8ffe ff43 e208 be32 8a8d c86e
 - RSA (2048) / RSA (2048)
 - encrypt: AES 256, AES 192, AES 128, CAST5 (128), 3DES (112/168)
 - Publius Cornelius Scipio Africanus <publius@roma.imperium> (password: password)**:
 - [12.12.2014] Publius Cornelius Scipio Africanus <publius@roma.imperium> (password: password)
 - fingerprint: ace0 b6a1 38a3 1cff 91e4 0f87 cd60 b3e5 f241 1ec2
 - RSA (2048) / RSA (2048)
 - encrypt: AES 256, AES 192, AES 128, CAST5 (128), 3DES (112/168)
- lists**: (empty)
- MyKeys**: (empty)

On the right side of the interface, there is a vertical panel with tabs for 'new directory' and 'recent files'.

MyPGP

Si se visualiza la clave privada:

```
-----BEGIN PGP PRIVATE KEY BLOCK-----
Version: BCPG v1.55
Comment: MyPGP (7.5.2016)

10PGBF0c8cCABCAClL2mdceC4KvyPR0W2Y9Kbm08u+g/+0C+rFy90g28P7a6JIE
L0xhERq0FFFeeCiHix/h70mKLH+n5Ga@mH0Hx2xyevvvvT/HwKKqGFbyPt98i6boW
5p5a/k1mmgx1Pw5CaNckoE8+TnAyNDu/6E/cPNWbqWn7nEBJ6dcG3ogZGweG/6hz
14ksNakc2rcCRIf0ZLwntB8IuiFLjhKUxOwawNx+GjY8pLhgiiYtDRr5Zizj1AYH
dKbnwhLRzR0XgpKJ5i1HwG1N9BTfVUykgagQqthzDnHJctzcR0J0105n32vD6gko
0x+fqUOXi1vvxfPv796UL49E76P5aEKUueLAEBAAH+COMiq7i2TuGUwsbaU26m
6X7hjUj6tskb8Bjh+wpa9k8bPLVfZz7Glv30D7mt2icVKYhnyHn3VfR4054XTNrN
q0wArhhgnXnFog9e6aMw8ie8eEUJ+qqaOn3qgtVw9TPNJRqqF8oypi1uBYDrr2D
20/iv2ub32jTysdzLYKbc/TL/LSA+GP5czQ3TMpNIYd5Pq1oMwjiode/Pxd4NKu5PY
TsIC63qIIqVJcWtVnspICN0j5eVbQRVWESvb480tVxkcW7MsLWaKLp0Ke653s5w
5/LVi2ATedLaR8E8KNivCSg4orzi+bwsL/TDJHcxRGZ1Hs+XkVLukU2as0Plbizz
MTrh4M1K6pzEp/P4+FGLGre5wxluFFBQV0B8jW7Bzu3U6PvuvJn@x7TAFOjnCzOfs
1Z3vbwN29y9IIxlosTWW82dG8q8I3ae+LD2k09+uknWcalRgkcuQ10wFBemGiSg
/1jRv40XvQUYtQaqg46PhoeU0XP7uGz67jY1n@tP+HmIV0d00Lttqz4dw148B3jjq
trdf67zYKV541Xeqzb0T4xwIwaG4xkZA6JeoT92NbuhSRUVW4Ts3jCmWd/bDjcx
CSMVc5jhcqodulBG6MqmDnvE4HccMqYmSFg2d1+K7G/arg8tNRhkqg1RIP0dIeV
moXWE2LJ2lBoOCNPw=2651ph1TcJ+se/+AO/crtVmql9awdY1o4K0jIQze+nR8E/5
tkST0THqJTI4271hK0m10SzuifQX450lN41zH150P98AHGFdFckJXiHJ2monHS
t@l/E5Y1YscfY3USe3YkVvSBxjaip10Md4RC69yR7MpieoLGSNe4Kj7rTyRGYNM
7w6eXv1UUmqJFK42JU8V1/PYKfG9W/5ca1Cqwc9GK79QJ0QJ1RNqjT2qCbCwf3xf
+Lh9k8uShoYgtC1lcyaBYTwk1YXjhekB8y2Muwd1hLmVzPiAobnVld=EqY2xhdMq
c2VjcmV8Y5m3ATUEfwEIAB83FAlg8cCEFGwMGcwkIBwMCBRYCAwEGF0qJCgsCAh4B
AAoJE011vJKKJchufTah/RHuverbJ+GJRK9MSID848Rb0XhfQyL1KsAgsCaAsk/7
B8XvrZnE/B89G1CIG29ghn3PB48HQNYZT8qwMn13rDAduoNh5rCt0Sfc0fTL8Gs
a40RGEmYw57h/sFVCBH/VFRbyjdZJUaBe7vvnx/05vcqEa5pd4NBj0WKt7HF8FAr
0vynzJ1+IdCvthQ9vq01k1+49T94Xmz7Kmk/r/q3j3VjJU5trh5ukAvYkmkDBm/G
HDM2qyFt9EowTtpbraWhjN8xarVzcKvX15/rHa3Im7b16w005pCoAY680j9xy1
1PAmZQW2zehuA4mfqIR3xRZjQEKhzqIavUbeonpNjjidABUEwDxwIAEIAKm8duWt
7dhS1E+UpEHxF7/eY0OM1eu/M4GF0oVxZPRT7V1fd/isshbFMy85B+1GN1sk93R+
vFFfw91XT+SSh913sEQ1Rbb+ZqsA0Z2zdsDQovTFdc0vPZhng9E1lkNln0MU060p
tQk+d7gtJG/aoBE1kfxxh+UMgs6z2fXL0MBkScP+GMVHxb0LIKam2bTn0Lb8TLQ
5apZ9eN0C+1dB=6enBvxthKxw4d2zIHBTYKqKjrvjB89pmvc9ux7FZJYEJWENu9
6HF+nGw5s12c1w1Ax2BXwmtdAd8yT1Tgb/Gycd5tsT0Po12p7261Jnayf93YihmgNu
5QMnqB3512rMXUAEOEAAf4JAwiruXN04ZTCxsAGB82Kns+Vl6NhxUsh7Bpru6ut
w1u6s7RNWbMpug2znRKu8k1Vkb7XRkt1Z0dkZ0j10PX9cta1H4mMuYNG2mLoTJ
RCp3bT07EMPhAbmzH+071YE+INYczAHuV0sLw2db08T5MKKT91DrCX0w61FuYIA
jE5nfnbog3gm7mxz5hk0VJ1npl0e4inziGK1d1j0Z+B/RVQJV90n2d2Lx81hgmL8
60rnpmTzjgN1JW54uf3aV81tvIC9lu7Hw70PoP1zSEM9j9DEP01MLDq6h3v/7N1i
qyRtrPszaHzZ8+M9uUX9+r1YsBq3BvWCRowiu5Ngqkyz0/76w2Y2saiVlqp3DEifI
4/lAw/fsdhVt49sMiE0d1pxExksZxcfVMtA2NbibiET6n556okfEgMo9aiXcG0H/Y
1Nos910F48+1TILT10mIxzBaY1zx+qRM0j52KLk1ITHE6+xNxXutufxM1PYD92Y
KDqWhOnDjN2AKGqeWx6JY0hniG8kSxZ7BAP1qww1MErPmbRIwZg+3eDzf4MwUq
UKPuzfoQajL9pHy0eCtMKwJ5q1qiaenzulqtj18bUCl6pXhmFRYYEYL413eg1vByg
bCcDCLA7loqJ+7x1m0/dbW5TV65nC19P58Q12B1Atln1ur3RVmy@DC8k/mhKo3k
Exr65A05ycGEQo2I1dma/6ew3DkwkJC98XHQ+M7xzM0cvNC7YSWES2wfE2310xz/
1H7MaMT45Hg565YfMU9ULiwDgh80kTVsw/vx88A6RBZ61j4bK938d11zL+lwNXUP
tsuwjBD+zY12C1BxdZHWpmlCSt/0VqjNSXh726mr4G8qf2v+u1p2wHU3MFoE8j
Ee9NwjhbFmkwbRBvwB0o3Zw59hdwuUjZUuR+tyFHEb4UnUR74o/WBt4kBHwQYAQgA
CQUCWdxwI0ibDAAKCR01CL4y1o31biwuB/9GbTTfxjWvzrFicAa90GuRSvvFs/7F
r40MuKXXcKoPnw65Sgll5c4YkKPRgzmxxk1C6uLo8e1t2mxwNWwKTNoY4vNsY/LD
HNgs0/1FN5UKpmlbcRmkkPTX0gruk6MT2q5/b5gnAAxNwWItpU1UDDEfxh3LP/H
QEJM0D9dL7+bo1nrqHtu1b1Mwlv2ZDR0M+qLx5780G1bfu+8+bBKDAfrFwNIjohx
idJBHfup@v19FSXEmGPexJW1Iuz55ccYS60uXH1yuw3Xqg5jyLHTMwGZ51DU1+D6
IOgzSolr17R6Wlm0cLo38++RyIdnLSF+4DfZB6+78UW5INg16CcqahXw
==3c+
-----END PGP PRIVATE KEY BLOCK-----
```

MyPGP



S/MIME (Secure/Multipurpose Internet Mail Extension)

- S/MIME es una mejora en el ámbito de seguridad del formato MIME para correo electrónico
 - el cual a su vez es una mejora de SMTP
- Algunos de los documentos que describen S/MIME son:
 - RFC 5652: Cryptographic Message Syntax (CMS)
 - RFC 5750: Secure/Multipurpose Internet Mail - Version 3.2 -Certificate Handling
 - RFC 5751: Secure/Multipurpose Internet Mail Extensions -Version 3.2 - Message Specification



S/MIME (Secure/Multipurpose Internet Mail Extension)

- Aunque tanto PGP como S/MIME están en vías de llegar a estándar, todo apunta a que S/MIME se va a consolidar como estándar para uso comercial
 - mientras que PGP quedará para uso personal
- En términos de funcionalidad, S/MIME es similar a PGP en el sentido de que ambos ofrecen la posibilidad de **firmar y/o cifrar mensajes**
- S/MIME usa **certificados de clave pública** con formato **X.509v3**
 - Sigue también el **modelo de PKI híbrido** entre la jerarquía estricta de Autoridades de Certificación y el modelo en malla

S/MIME (Secure/Multipurpose Internet Mail Extension)

- Utiliza los algoritmos criptográficos de la siguiente tabla:

Function	Requirement
Create a message digest to be used in forming a <u>digital signature</u> .	MUST support SHA-1. Receiver SHOULD support MD5 for backward compatibility.
Encrypt message digest to form a <u>digital signature</u> .	Sending and receiving agents MUST support DSS. Sending agents SHOULD support RSA encryption. Receiving agents SHOULD support verification of RSA signatures with key sizes 512 bits to 1024 bits.
Encrypt <u>session key</u> for transmission with a message.	Sending and receiving agents SHOULD support Diffie-Hellman. Sending and receiving agents MUST support RSA encryption with key sizes 512 bits to 1024 bits.
Encrypt message for transmission with a one-time session key.	Sending and receiving agents MUST support encryption with tripleDES. Sending agents SHOULD support encryption with AES. Sending agents SHOULD support encryption with RC2/40.
Create a message <u>authentication code</u> .	Receiving agents MUST support HMAC with SHA-1. Sending agents SHOULD support HMAC with SHA-1.

Thunderbird

Seguridad

Para enviar y recibir mensajes firmados o cifrados, debe especificar tanto un certificado para firma digital como uno para cifrado.

Firmado digital
Usar este certificado para firmar los mensajes que envíe:

 Firmar mensajes digitalmente

Cifrado
Usar este certificado para cifrar/descifrar mensajes enviados a Vd.:

Cifrado elegido para enviar mensajes:
 Nunca (no usar cifrado)
 Siempre (no podrá enviar si algún receptor carece de certificado)

Certificados

Redacción: Prueba S/MIME

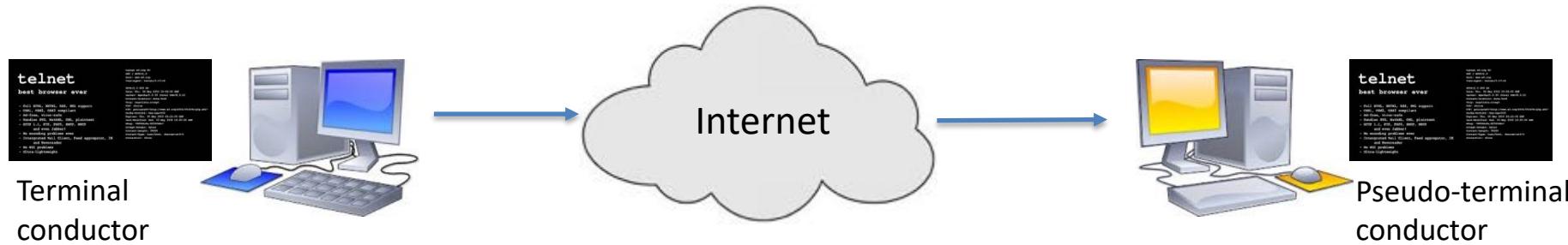
Archivo Editar Ver Insertar Formato Opciones Herramientas Ayuda
Enviar | Ortografía Adjuntar Seguridad Guardar
De:
Para:
 Cifrar este mensaje
 Firmar digitalmente este mensaje

smime.p7m

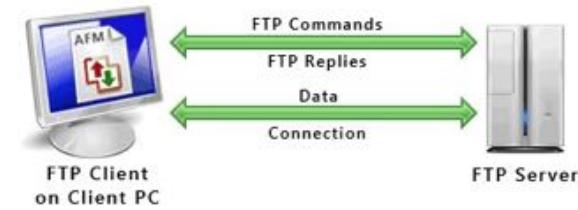
```
1 0€ACK *tHt+
2 SOHBELEOTX €0€STXSOHNUL1, SOH$0, SOH STXISOHNULO^0,1VTO ACKETXUEOTI
3 ACKETXUEOTB$DC3ACKMalaga1$T0
4 ACKETXUEOTBE1.DC3ACKMalaga1FF0
5 ACKETXUEOT
6 DC3ETXUMA1FF0
7 ACKETXUEOTVTO DC3ETXUMA1DC20DEACKETXUEOTETXDC3 Cristina 1!0USE
8 SOH SOHSYNDC2alcaraz@lcc.uma.esSTXSOHSTX0
9 ACK *tHt+
10 SOH$0SOH$0ENONULUO'e', 'NAK0'fi-c·Ó*, -jÓj (»..d£ò"DC1 [2v, SUB00ùSOI
11 \Íg..iÑADC4iè80CANiÀs<GDLÉ=ÛX: FF/Sé€S]56NAKIDOLEF6Jg\4Sf]Óauai}ETBK
12 SOHBELSOH$0DC4ACKBS*tHt+
13 ETXBELEOTBSIEP*t'aG €EOT,
14 @`FS3ñ-*piSUB&2,D/[0,483G^Z't^!1@GShfq23í]GSLjETX«"w=iBELVACKXç_
15 =iDC4aÀ@e)1=NscP2^ÓVÜB<"Vúcf0&"US«"dÀi-iðüEvY6ë€rDC1ZcË\3[írt
```

Telnet (TELetype NETwork) y FTP (File Transfer Protocol)

- Características funcionales:
 - **Telnet** (puerto 23): facilita el acceso remoto a otros sistemas y sobre TCP, de forma que el terminal local aparenta ser el terminal del sistema remoto

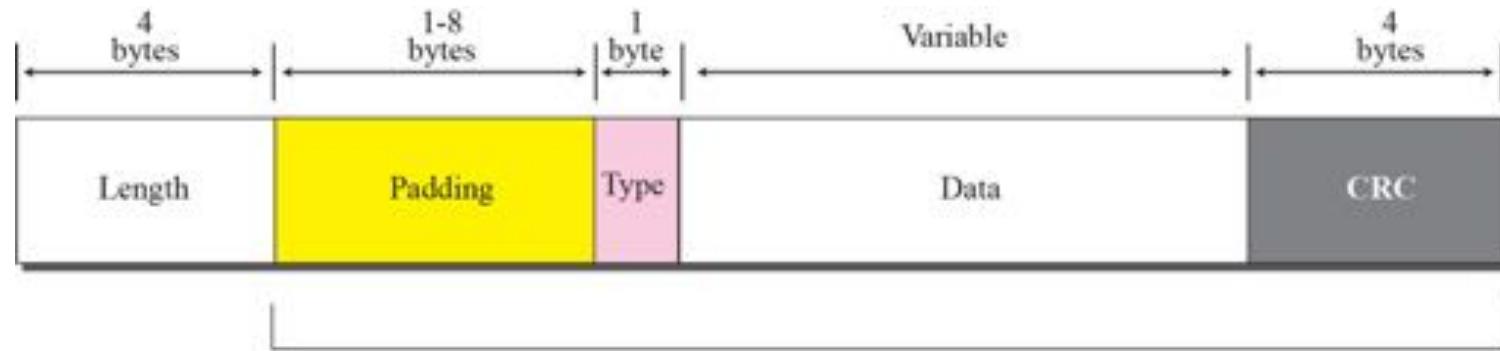
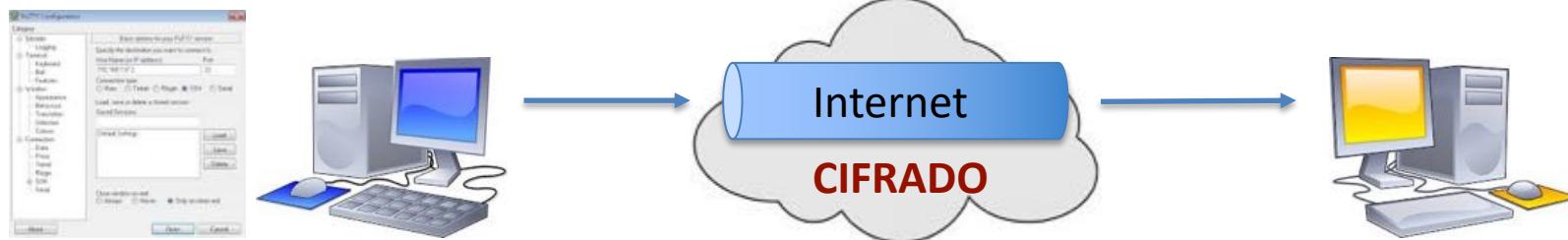


- **FTP** (puerto 20/21): permite la transferencia de ficheros entre diferentes recursos remotos
- Problemas:
 - La información transferida y las acciones remotas se realizan en claro, por lo que no se recomienda su uso



SSH: Secure Shell (v2)

- SSH es una herramienta similar al telnet funcionando en el puerto **22**
- Permite el acceso remoto sobre TCP a otros sistemas usando el concepto de cliente/servidor pero cifrando las transacciones usando **criptografía pública**



SSH: Secure Shell (v2)

- Funcionamiento general:
 - **Capa de aplicación:**
 - Gestiona la autenticación del cliente haciendo uso de un usuario/contraseña o aplicando criptografía de clave pública
 - **Capa de transporte:**
 - Gestiona e intercambia las claves iniciales
 - Establece los modos de cifrado y de comprensión
 - **Capa de red:**
 - Establece una “conexión directa” entre el cliente-servidor y redirige el tráfico entre estos puntos de conexión
 - Modo túnel (en base a cifrado simétrico negociado en la capa de transporte)
- Mitiga o evita ataques específicos:
 - **spoofing de IP o suplantación de identidad**
 - nodos remotos intentan suplantar la identidad de otro nodo de la red
 - **spoofing de DNS** en donde el atacante trata de suplantar el nombre del servidor

SSH: Secure Shell (v2)



PuTTY

OpenSSH

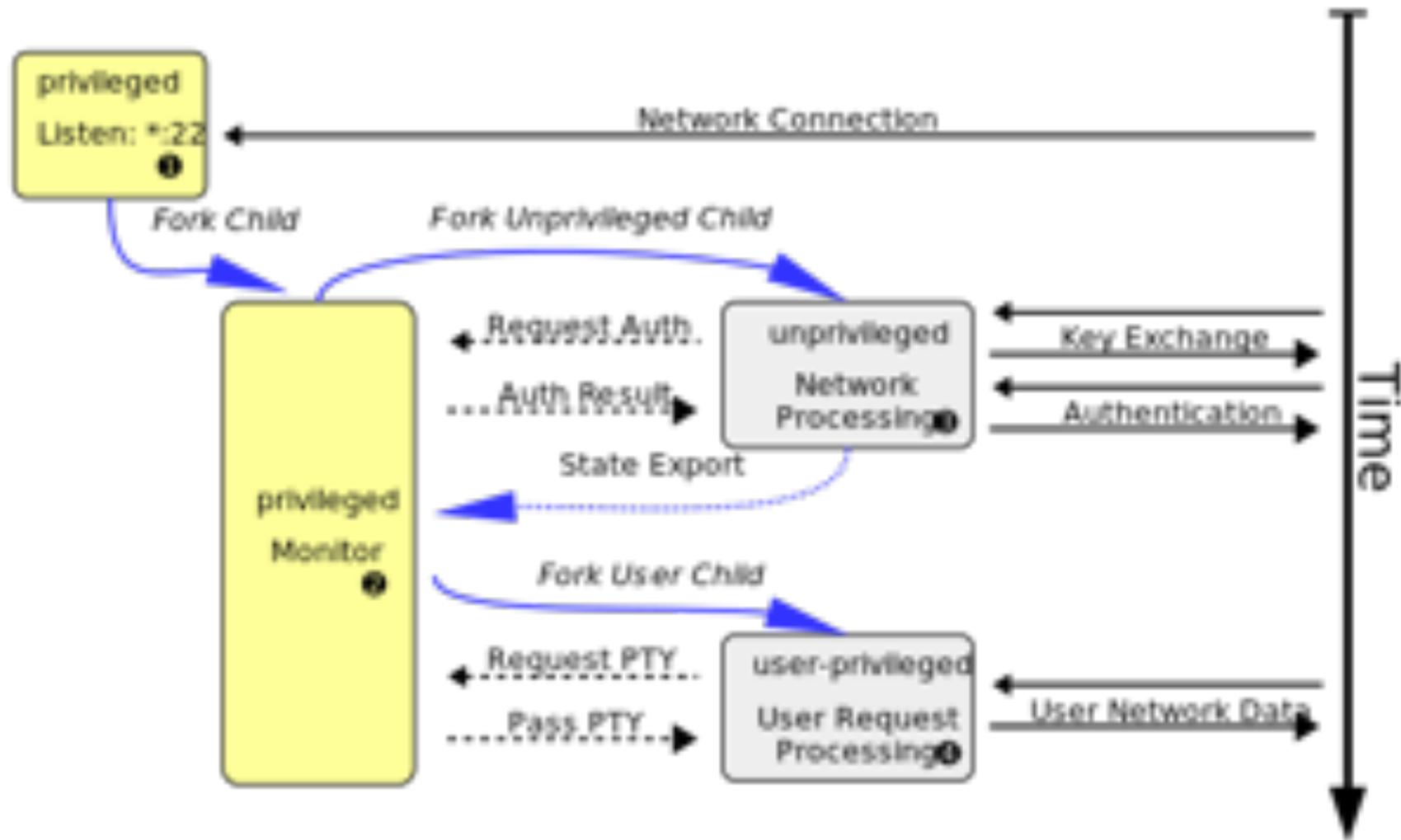
```
vdi-6E28:~ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/
[Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/vdi-6E28/.ssh/id_rsa.
Your public key has been saved in /home/vdi-6E28/.ssh/id_rsa.pub.
The key fingerprint is:
[SHA256]GX02RJH2boy@we6Pl481emRo0zTxY2deghkt1uK3vA
The key's randomart image is:
----[RSA 2048]----
|   .+0oo o |
|   ++0oo,++ |
|   ...B .+0o |
|   o++o,+ |
|   S+o+o+ |
|   . .o. . |
|   o . . . |
|   . + . . |
|   . E . . |
----[SHA256]----
```

SSH: Secure Shell (v2)

- SSH puede ser usado también para transferir ficheros como una alternativa a FTP, conocido como SFTP (SSH File Transfer Protocol) y SCP (Secure Copy)
- **SFTP (FTP sobre SSH) # FTPS (FTP sobre TSL)**
 - Funcionamiento de SFTP:
 - Se puede basar de diferentes modos de autentificación para conectar con el servidor SFTP:
 - **Modo básico:** usuario y contraseña
 - **Modo avanzado:** usando las claves públicas de SSH, previamente generadas, y compartiendo dichas claves públicas con el servidor SFTP
 - » De esta forma, cuando el cliente quiere establecer conectividad con el sistema remoto, el proceso software del cliente tendrá que transmitir su clave pública al servidor para su autenticación
 - Todas las conexiones SFTP están cifradas a nivel de red

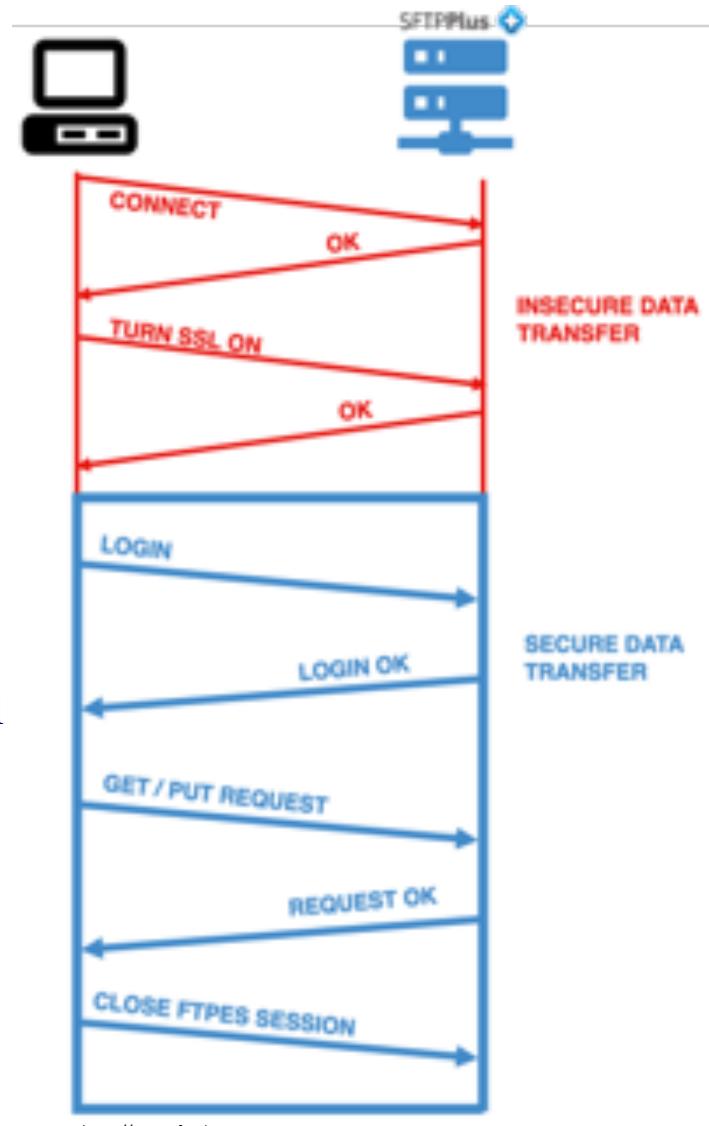


SSH: Secure Shell (v2)



FTPS (FTP Secure)

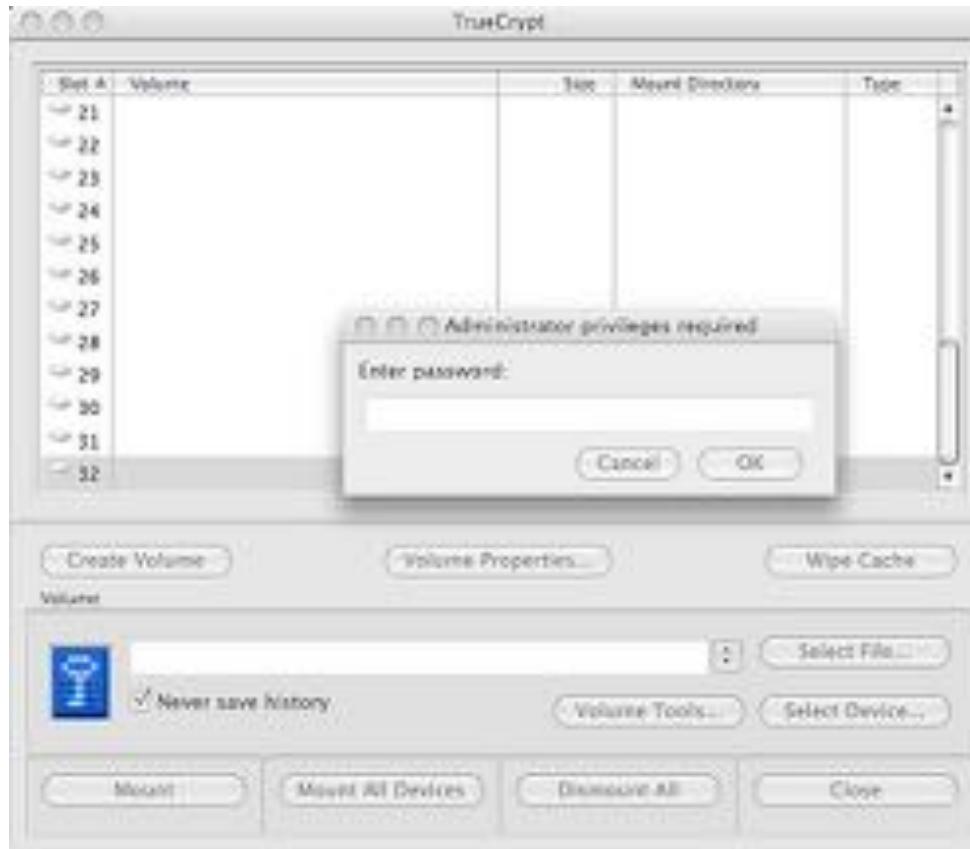
- FTPS proporciona un soporte adecuado para TLS (Transport Layer Secure)
- Funcionamiento:
 - Se basa de usuario, contraseña y certificados, de forma que:
 - Las credenciales de seguridad (usuario y contraseña) son cifrados a lo largo de la **conexión FTPS**
 - Para ello, el cliente primero verifica que el **certificado del servidor** es correcto y de confianza, y posteriormente se negocia una clave de session (ver Tema 5)



Herramientas de cifrado open-source

- **TrueCrypt:**

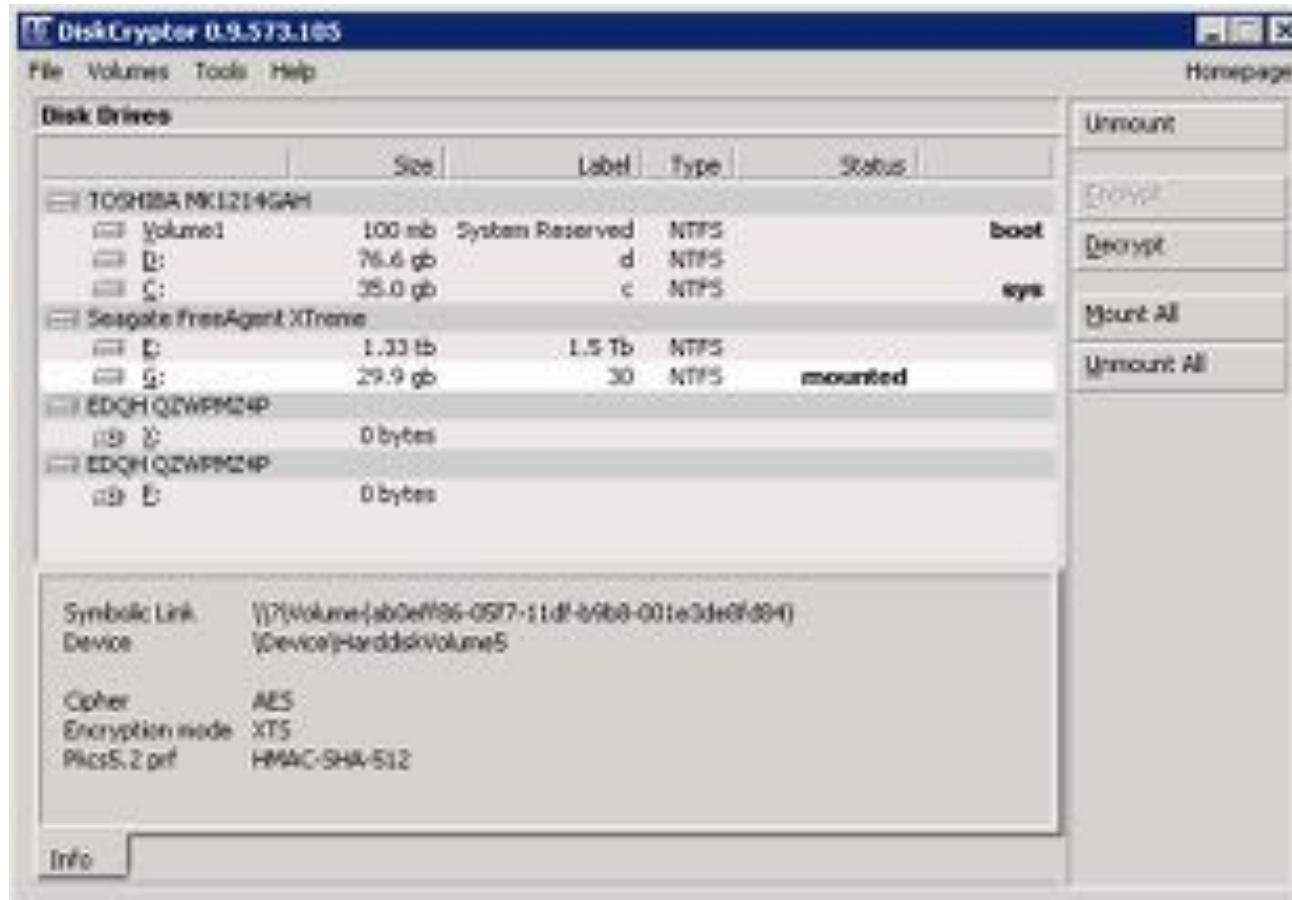
- herramienta de cifrado de discos duro disponible para Windows (XP/2000/2003) y Linux, haciendo uso de AES-256, Blowfish, CAST5, Serpent, Triple DES y Twofish
- También permite ocultación de particiones haciendo uso de cifrado y aleatoriedad de la información



Herramientas de cifrado open-source

- **DiskCryptor:**

- similar a TrueCrypt pero con capacidad de cifrar dispositivos de almacenamiento externo USB
- También incluye algoritmos de cifrado como AES, Twofish y Serpent

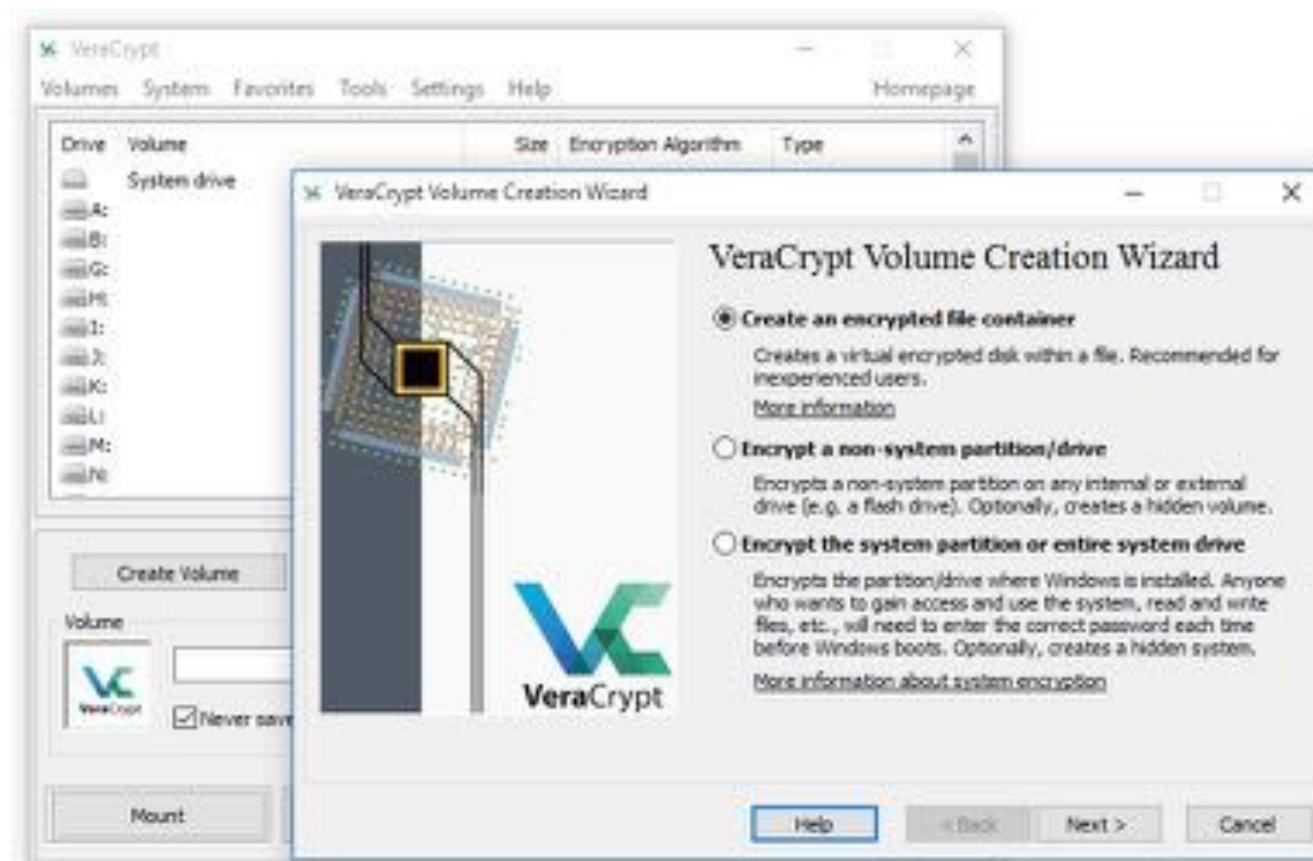


Herramientas de cifrado open-source

-

VeraCrypt:

- similar a TrueCrypt pero con la diferencia que incluye un número específico de iteraciones para el cifrado, incrementando la lentitud del sistema durante los procesos de lectura y escritura en el disco
- Como TrueCrypt, aplica AES, Twofish y Serpent



Herramientas de cifrado open-source

- **OpenStego**: aplica técnicas de **Esteganografía** (del griego "cubierto" u "oculto", y "escritura") para ocultar información haciendo uso de imágenes o cualquier archivo multimedia
- **OpenPuff**: es similar a OpenStego para Windows con soporte para BMP, JPG, MP3, WAV y MPG4



C0200' 80-
21B2C809 CB3EE8EF DF038D
04143B75 4F571C
B57C659E C820E
F743D 9A361
9A51

SEGURIDAD EN PAGOS ELECTRÓNICOS

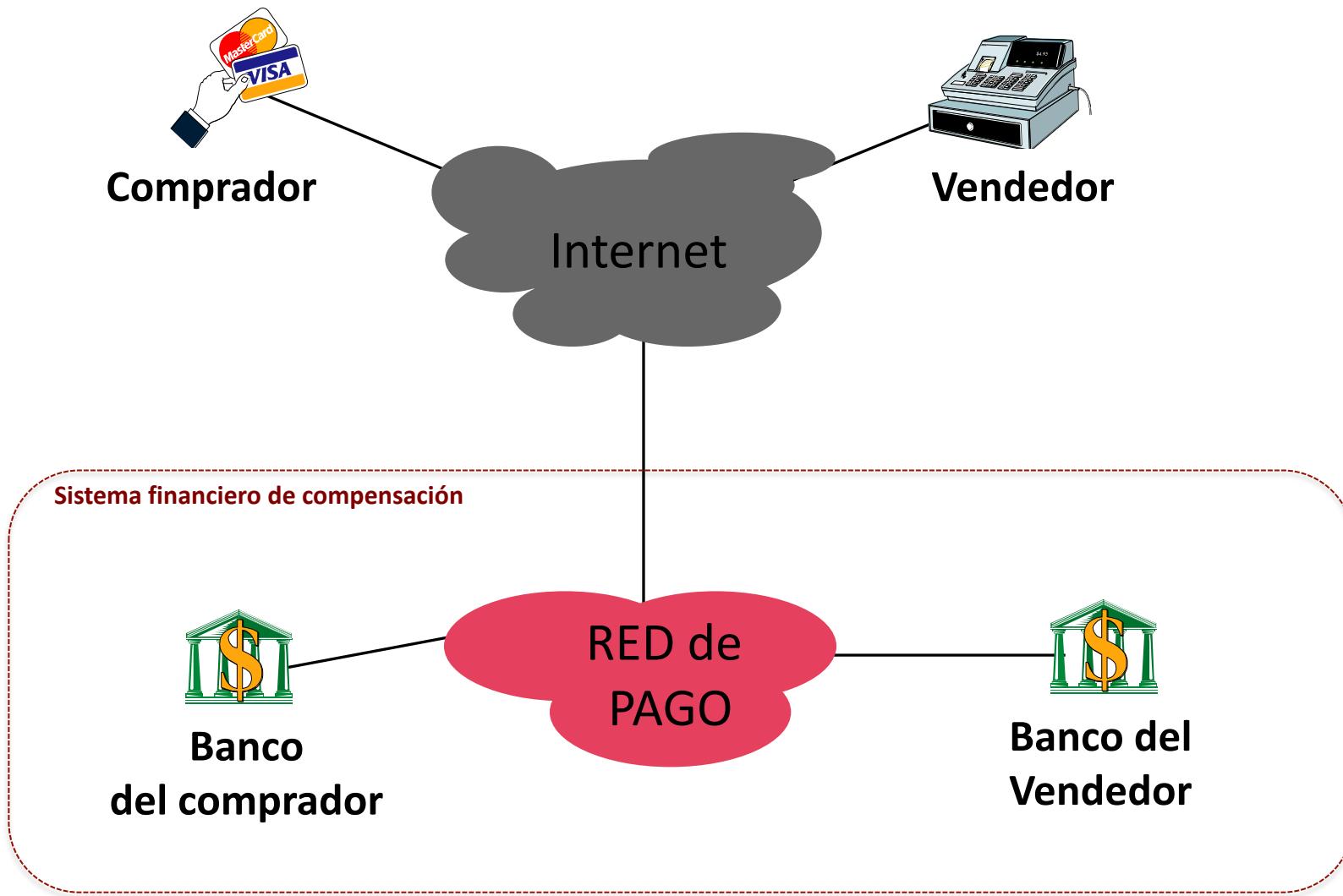


Introducción

- En el ámbito del e-commerce se han desarrollado esquemas de pago electrónico que proporcionan en el mundo digital la misma heterogeneidad y funcionalidad que los sistemas de pago tradicionales
- En la mayoría de los sistemas de pagos electrónicos disponibles, los pagos se realizan a través de **redes abiertas como el Internet**
 - pero la correspondencia entre los pagos electrónicos y la transferencia del valor real es realizada y garantizada por los bancos, a través de los **sistemas financieros de compensación**
 - estos sistemas utilizan para su funcionamiento las redes cerradas de las instituciones bancarias, las cuales son consideradas comparativamente más seguras



Introducción



Introducción

- En general, los pagos electrónicos involucran a un **comprador y a un vendedor**
 - pero pueden existir sistemas y protocolos que involucren a TTPs
 - y también puede existir algún tipo de entidad que ejerza de mediador para la **resolución de disputas**
 - por lo general, las disputas se resuelven fuera del sistema de pago, y en muchos casos el protocolo ni siquiera especifica cómo gestionarlas



Introducción

- Existen varias formas de clasificar los sistemas de pagos electrónicos, y dependen de:
 - **cuando el vendedor contacta con el banco** para verificar el proceso de pago (ej. si está autorizado)
 - **cuando el comprador procede con la transacción y carga de dinero** en la cuenta del vendedor
 - **la cantidad de dinero** implicada en cada transacción



Introducción - Cuando el vendedor contacta con el banco

- Los sistemas de pagos electrónicos se pueden clasificar según **cuando el vendedor contacta con el banco**:
 - **On-line**: antes de enviar el producto, el vendedor contacta con la entidad financiera para verificar la validez del pago del comprador
 - **Off-line**: cierto tiempo después de que el vendedor haya aceptado el pago y enviado el producto, realiza el depósito del dinero que le ha dado el comprador
 - para que la entidad financiera lo verifique y lo ingrese en su cuenta
 - es decir, el vendedor no contacta con el banco durante el proceso de compra-venta



Introducción -

Cuando el comprador procede con la transacción

- Existe otra forma de clasificar los pagos electrónicos, atendiendo al momento en que se retira el dinero de la cuenta del **comprador**:
 - **Sistemas de pre-pago**: el comprador ve decrementada su cuenta bancaria antes de realizar la compra
 - este método se correspondería con los sistemas de **monedero electrónico y tarjetas telefónicas**
 - éste sería el sistema más análogo al papel moneda tradicional
 - **Sistemas de pago instantáneo**: cuando al comprador se le realiza el cargo en cuenta justo en el momento de realizar la compra
 - se correspondería con los sistemas actuales de pagos con **tarjeta de débito**
 - **Sistemas de post-pago**: cuando el comprador realiza la compra, el banco asegura al vendedor que se le hará efectiva la cantidad acordada
 - pero el comprador sólo verá decrementada su cuenta cierto tiempo después de haberse realizado la compra



Introducción - La cantidad de dinero

- Otro criterio de clasificación es **según la cantidad implicada en la transacción**. De esta forma se clasifican los pagos electrónicos como:
 - **Macropagos**: cualquier pago superior a 10 euros
 - **Pagos**: la cantidad está comprendida entre 1 y 10 euros
 - **Micropagos**: cualquier pago inferior a 1 euro
- Normalmente, los **pagos inferiores a 10 euros** presentan el problema del coste de implementación
 - no tendría sentido utilizar un sistema de pago cuyo coste económico sea de orden de magnitud o superior al importe de la transacción



Introducción

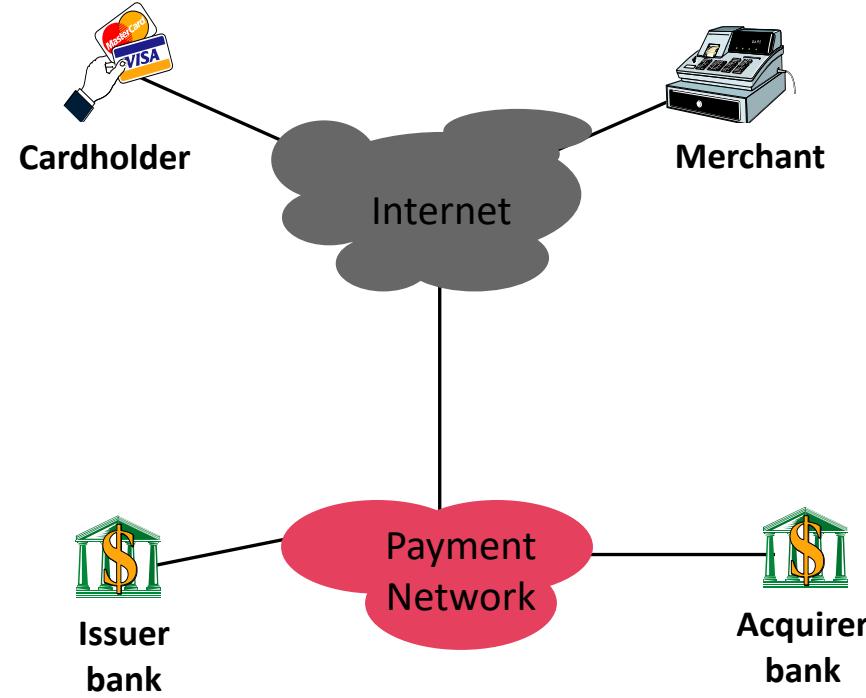
- Hay muchos protocolos que gestionan el proceso de pago online, tales como:

- **On-line y trazables:**

- First Virtual
 - CyberCash
 - iKP
 - SET
 - ...

- **Micropagos:**

- PayPal
 - Google Checkout
 - Amazon Payments
 - iTunes Store
 - ...

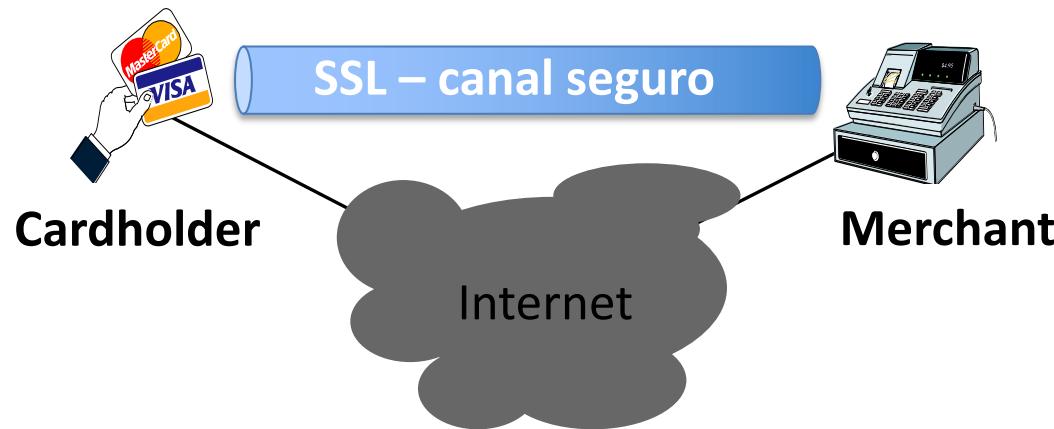


Introducción

- Sin embargo, existen numerosos problemas de seguridad en estos medios de pagos:
 - Ataques de escucha
 - Suplantación de identidad (cliente o comerciante)
 - Generación de dato falso
 - Modificación del dato
 - Etc.
- Para evitar estos problemas de seguridad, los protocolos suelen implementar:
 - Primitivas criptográficas
 - Mecanismos de autenticación y autorización de usuarios
 - Firma digitales
 - Certificados digitales
 - Etc.

Introducción - un poco de historia ...

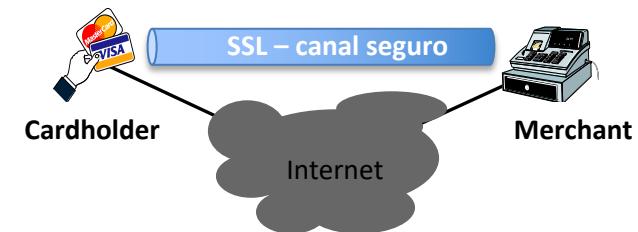
- Inicialmente se aplicaba el **protocolo Secure Sockets Layer (SSL)** como medida de protección de los canales de comunicación



- SSL es un protocolo de propósito general (como TLS) para establecer conexiones seguras
 - **No es un protocolo de pago**, pero se usaba por seguridad
 - SSL lo creó originalmente Netscape (1994).
 - La última versión: SSLv3 – ¡¡ESTÁ YA OBSOLETA!!

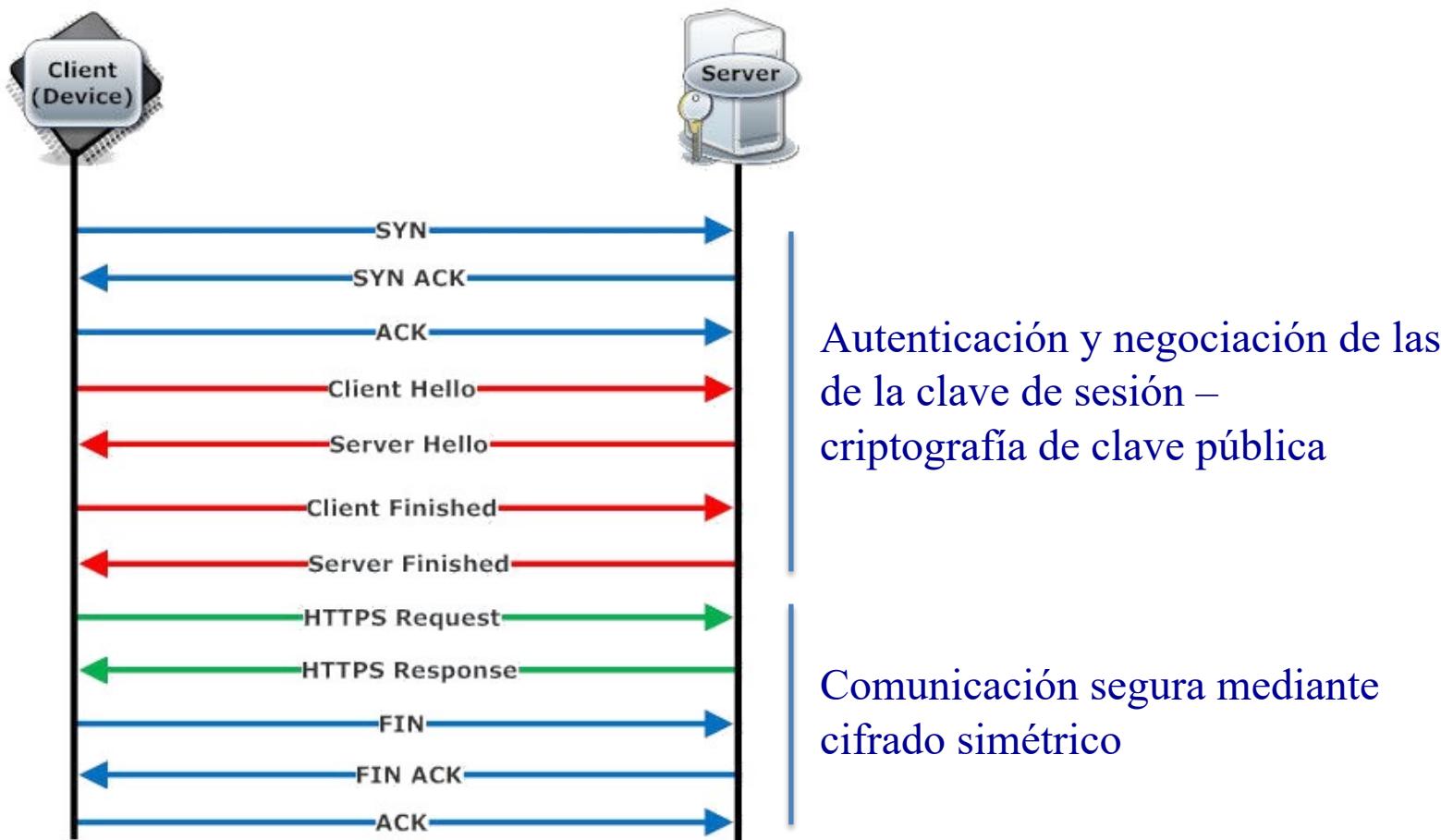
Introducción - un poco de historia ...

- SSL inicialmente se aplicaba porque proporcionaba el uso de la cripto. híbrida:
 - criptografía asimétrica para:
 - la autenticación usando certificados digitales
 - la negociación de las claves de sesión con RSA o Diffie-Hellman
 - criptografía simétrica para:
 - el cifrado de punto a punto usando DES, 3DES, RC2, RC4 o IDEA
- SSL autentifica al servidor y al cliente
 - utilizando certificados digitales X.509 v3
 - la autenticación con certificados es completamente opcional para ambas partes, el servidor y el cliente
- SSL también asegura la integridad de los datos
 - mediante MAC y una clave secreta para dicha MAC
 - pero uso MD5 o SHA-1 para la MAC ☹



Introducción - un poco de historia ...

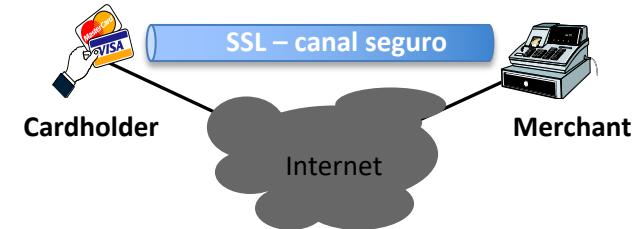
- Originalmente, SSL proveía **confidencialidad en el pago electrónico**
 - Garantizaba la creación de un canal seguro entre cliente y servidor



Introducción - un poco de historia ...

- Sin embargo, SSL presentaba algunos **problemas**:
 - Sólo protege transacciones entre dos puntos
 - mientras que una transacción electrónica basada en una tarjeta de crédito involucra al menos a un banco
 - SSL no protege al comprador frente al vendedor
 - el vendedor puede obtener información de la tarjeta que podría utilizar en un futuro de forma ilícita
 - No hay mecanismos de autentificación de tarjetas
 - no está la entidad bancaria quien autentica la entidad interesada
 - No hay mecanismos de facturación o de gestión de recibos
 - cualquier reclamación queda a la buena voluntad del vendedor

Por tanto, se requerían de otros tipos de protocolos más específicos ...



Protocolo SET (Secure Electronic Transactions)

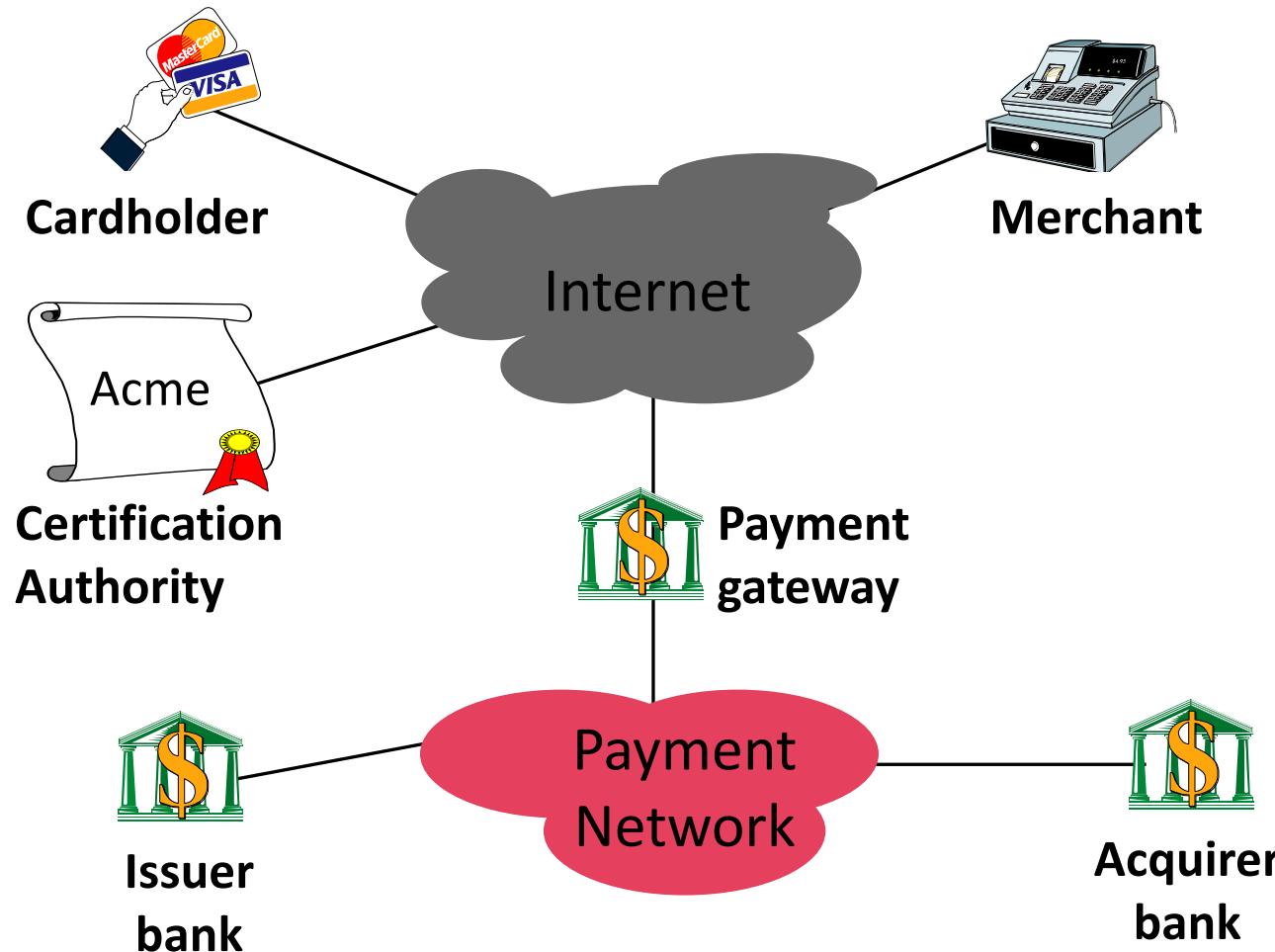


- Protocolo desarrollado en 1996 por VISA y Mastercard (+ American Express), en colaboración con:
 - IBM
 - Microsoft
 - Verisign
 - RSA
 - Netscape
 - GTE
- El objetivo era proporcionar seguridad a las **transacciones electrónicas basadas en tarjetas de crédito**
 - para poder reducir el fraude mercantil
 - y garantizar el pago a través de esas mismas redes



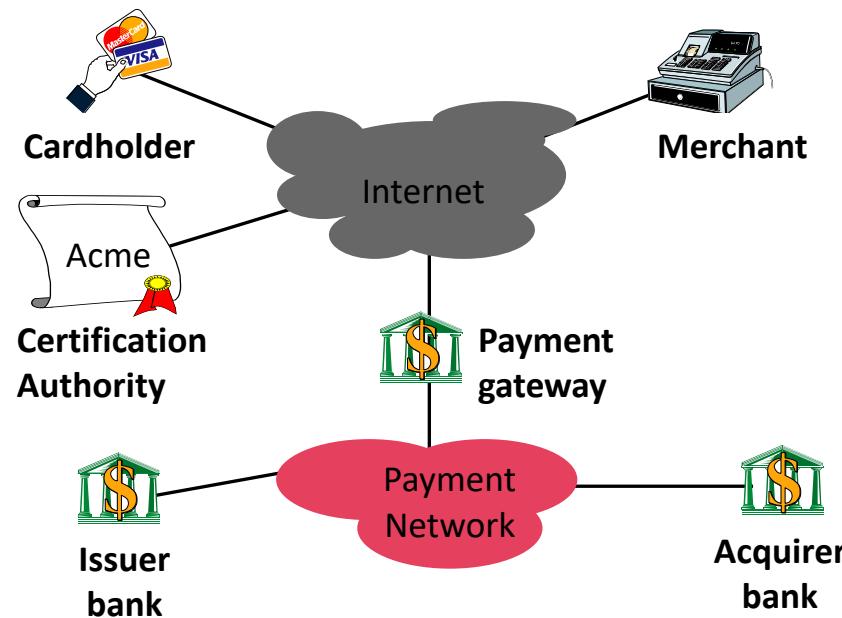
Protocolo SET (Secure Electronic Transactions)

- Respeta la infraestructura de pago existente:



Protocolo SET (Secure Electronic Transactions)

- A diferencia de SSL, fue diseñado para el comercio electrónico
- Sin embargo, no es un sistema de pago en sí mismo, sino un **conjunto de protocolos de seguridad y de formatos estándar**
 - que permiten a los usuarios usar de una forma segura a través de Internet, la infraestructura ya existente de tarjetas de crédito



Protocolo SET (Secure Electronic Transactions)

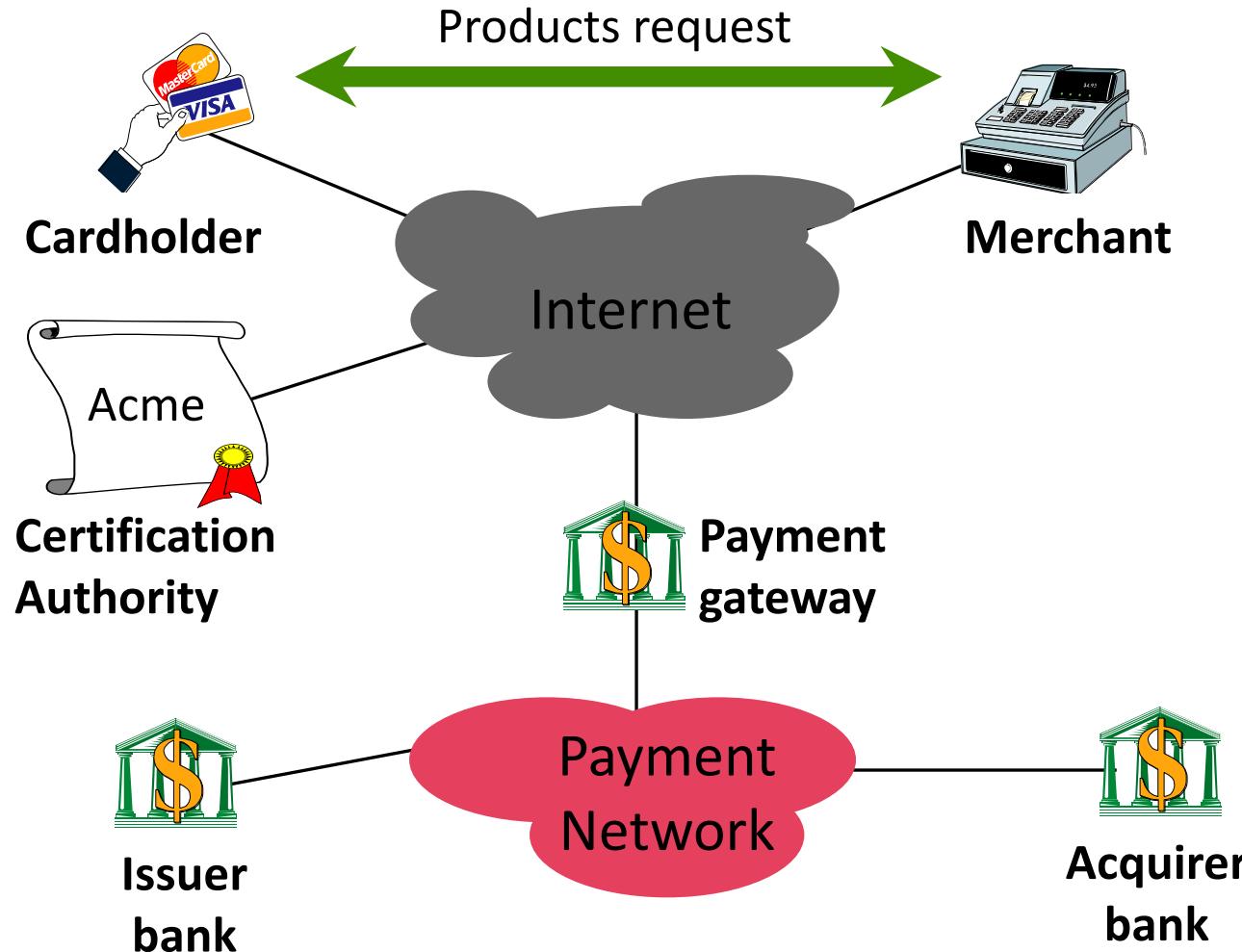
- SET proporciona un conjunto de servicios de seguridad:
 - **Confidencialidad** en las comunicaciones entre las entidades que intervienen en la transacción
 - **Autenticación**, a través del uso de certificados digitales X.509
 - Todas las entidades, incluyendo el cliente, el vendedor y la pasarela de pago, han de tener certificados X.509 [... OBLIGATORIO]
 - Es necesario el servicio de una o más Autoridades de Certificación
 - **Privacidad**, porque la información sólo está disponible para las diferentes entidades cuando y donde es necesario
 - **Integridad** por las firmas digitales
 - **Reduce las disputas** debido al no repudio
 - **Autorización** de pago y, por tanto, confirmación de la transacción y garantía de pago al vendedor

Protocolo SET (Secure Electronic Transactions)

- Pasos del protocolo SET:
 1. **Petición de producto**
 2. **Inicialización:**
 - envío de certificados
 3. **Información del pedido e instrucciones de pago:**
 - descripción de la compra
 4. **Petición de autorización:**
 - vendedor-pasarela y pasarela-banco emisor
 5. **Aprobación de autorización:**
 - el banco emisor autoriza el pago
 6. **Finalización:** el vendedor reclama la cantidad a la pasarela
 - Petición de compensación hacia el banco del vendedor

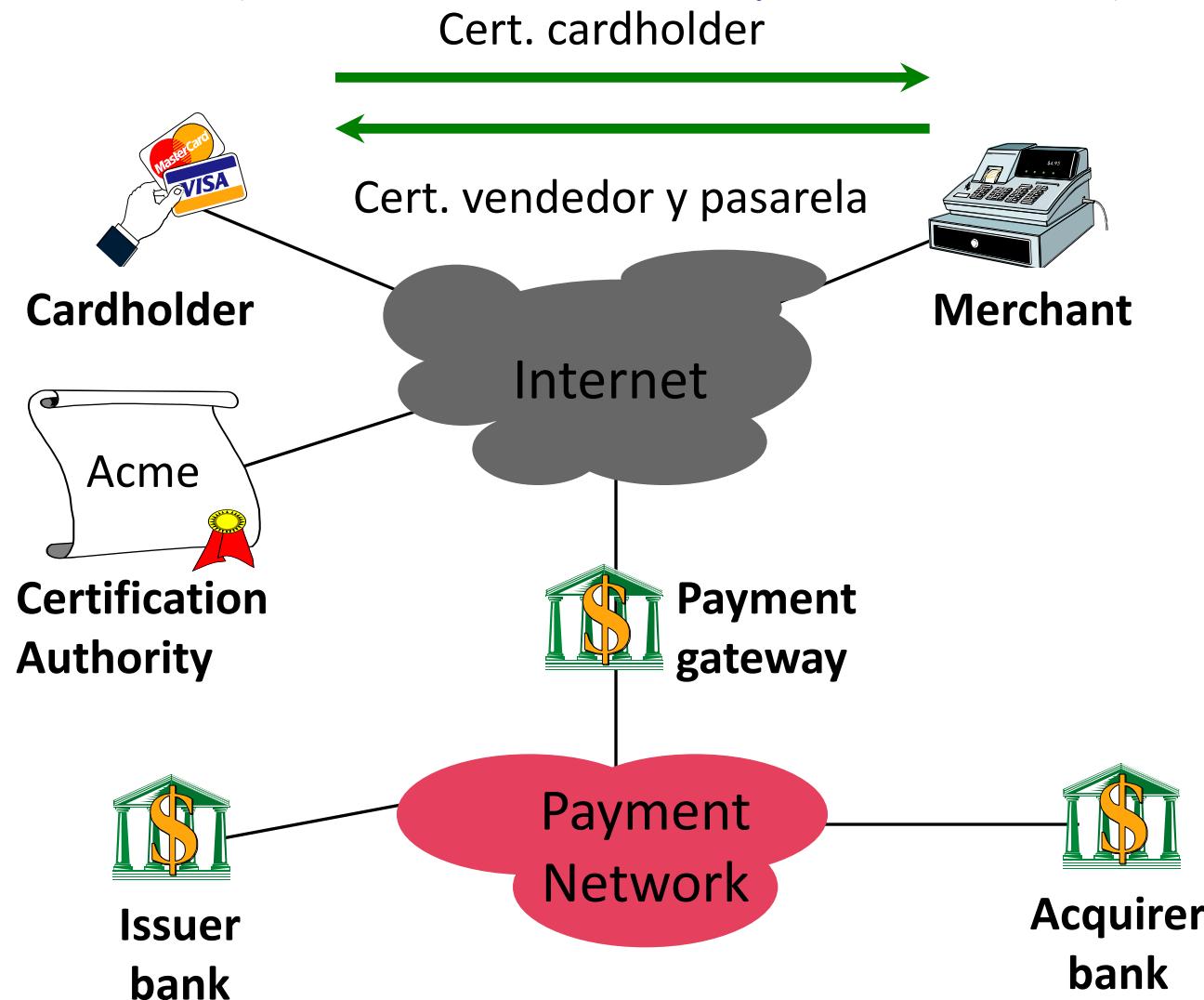
Protocolo SET (Secure Electronic Transactions)

1. Petición del producto



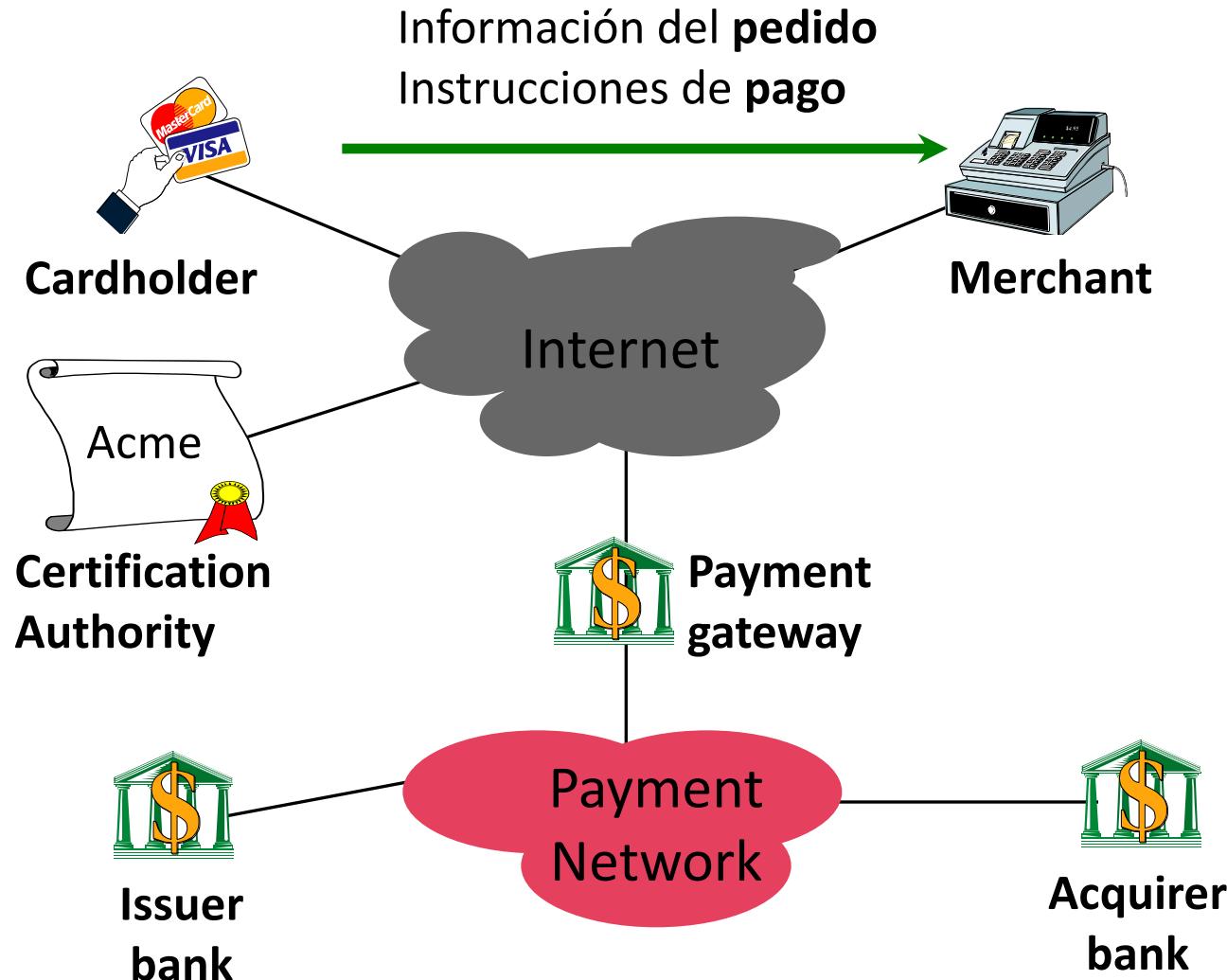
Protocolo SET (Secure Electronic Transactions)

2. Inicialización (envío de certificados y autenticación)



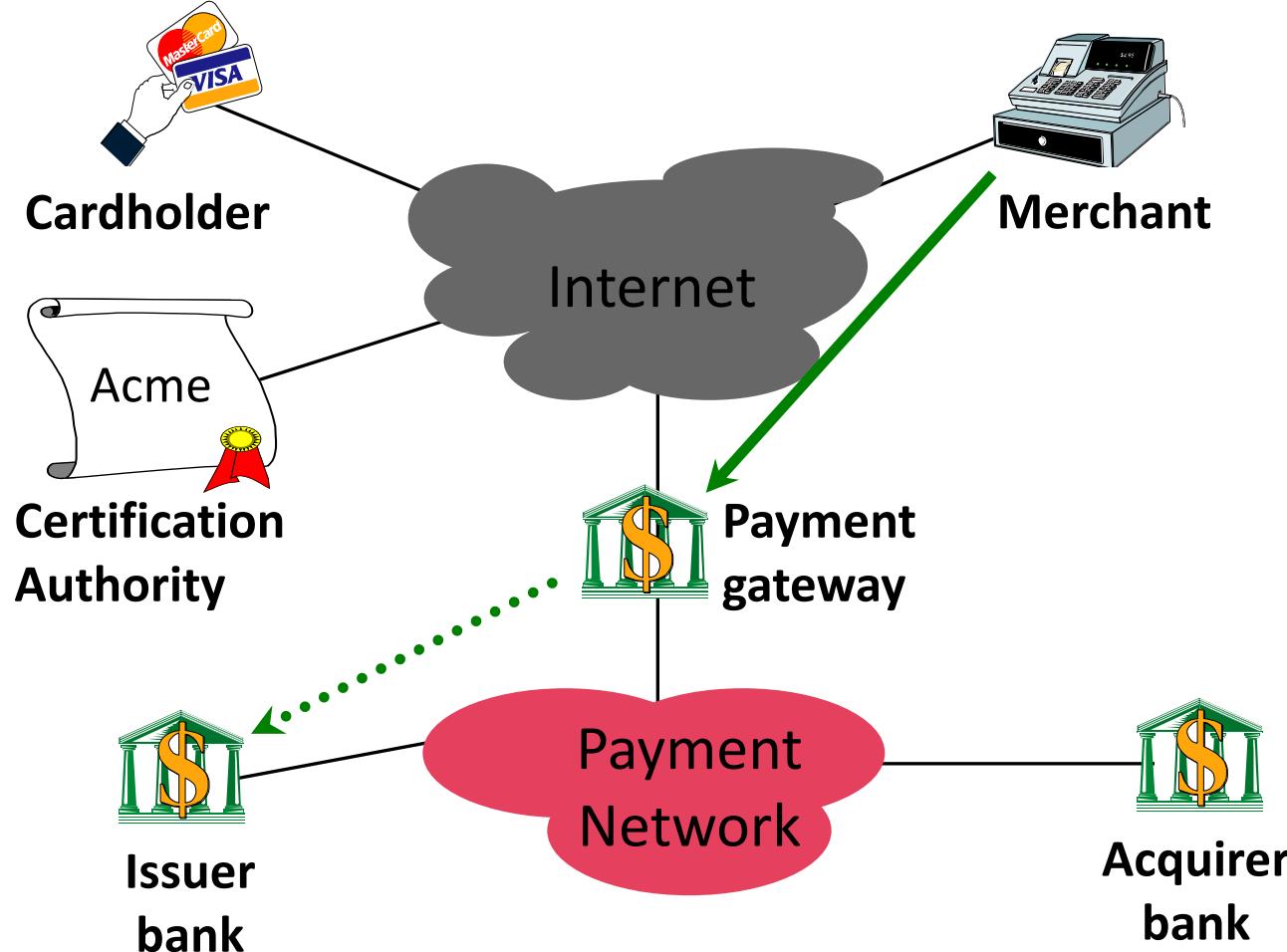
Protocolo SET (Secure Electronic Transactions)

3. Información del pedido e instrucciones de pago



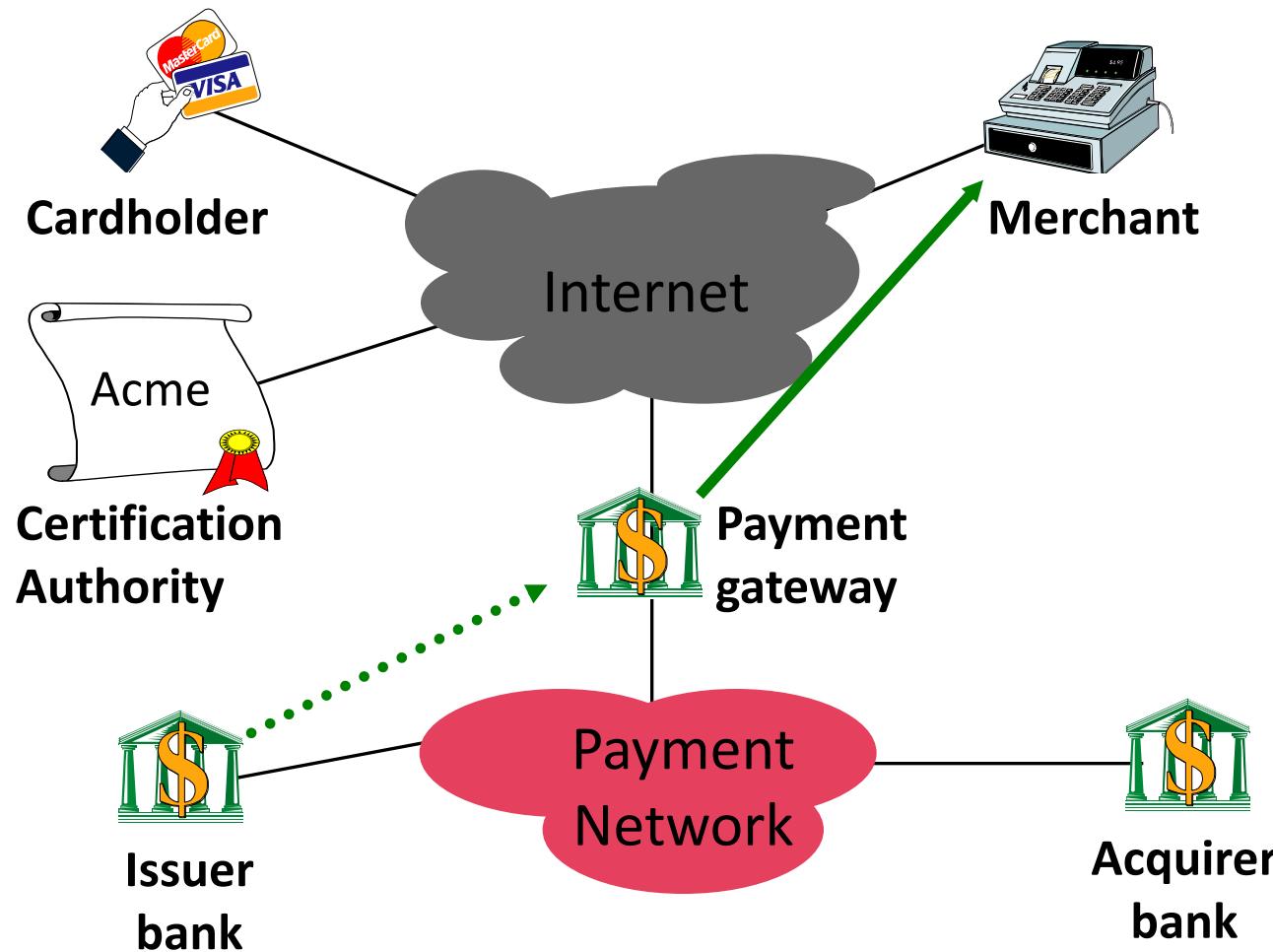
Protocolo SET (Secure Electronic Transactions)

4. Petición de autorización



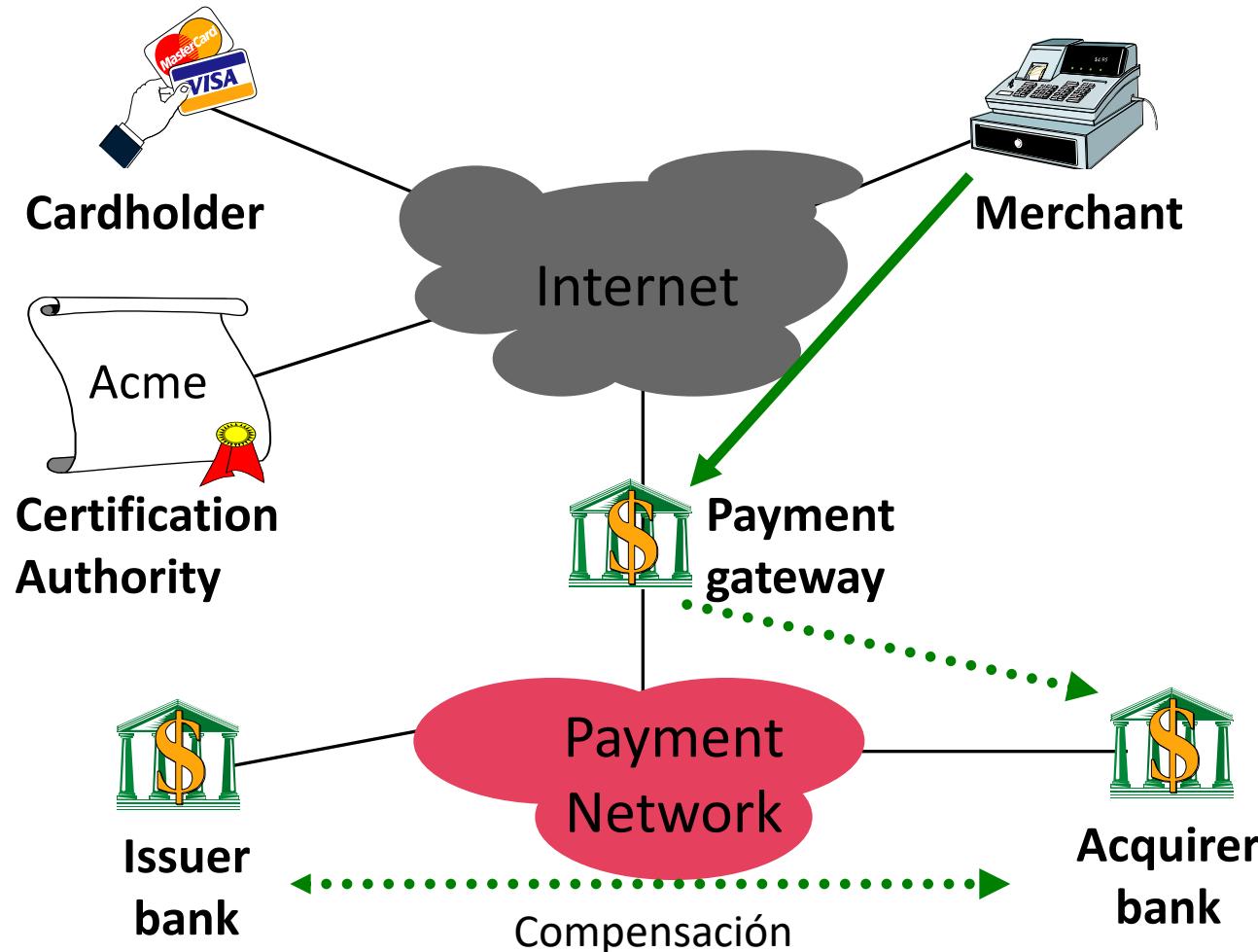
Protocolo SET (Secure Electronic Transactions)

5. Aprobación de autorización



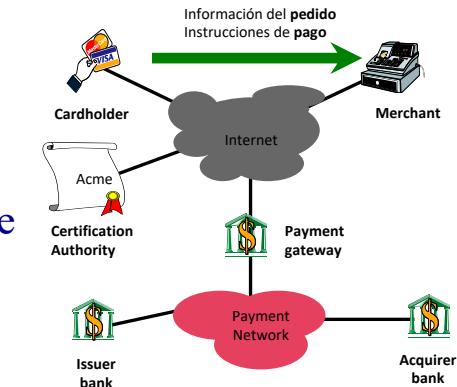
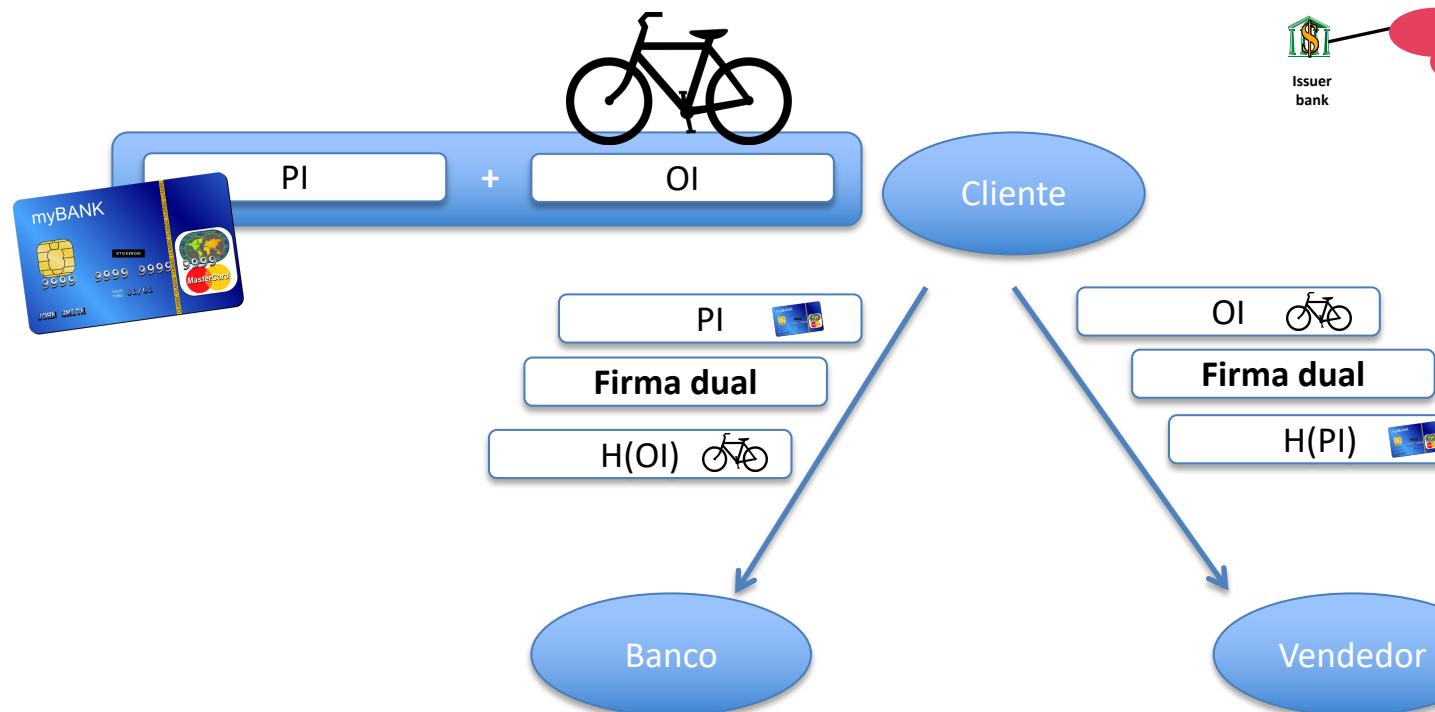
Protocolo SET (Secure Electronic Transactions)

6. Finalización



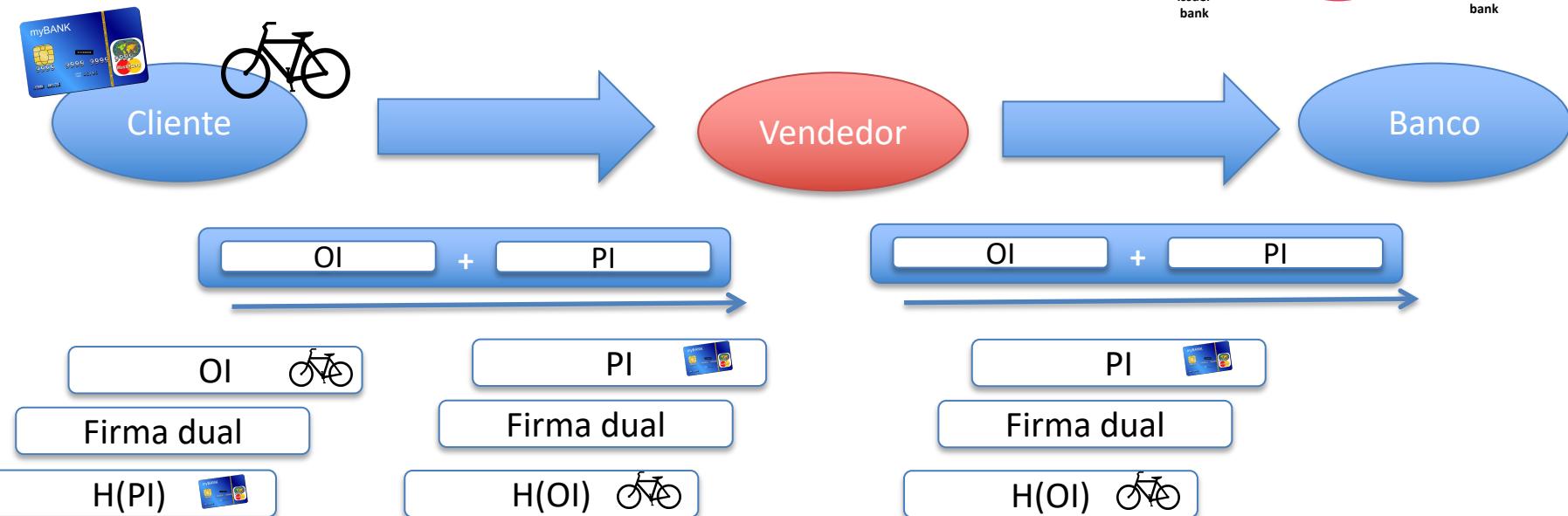
FIRMA DUAL (en SET)

- SET introduce una importante innovación técnica: la **firma dual**
 - El propósito de este tipo de firma es **enlazar** dos mensajes que han de ir a receptores distintos:
 - la información del **pago** (Payment Information) al banco
 - la información del **pedido** (Order Information) al comerciante



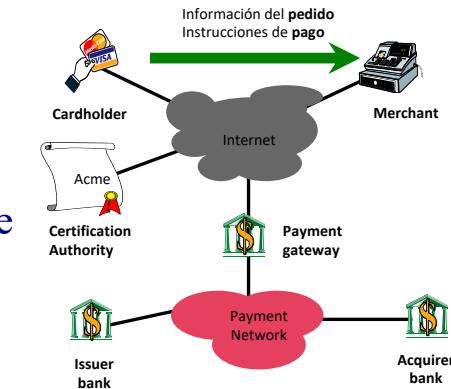
FIRMA DUAL (en SET)

- SET introduce una importante innovación técnica: la **firma dual**
 - El propósito de este tipo de firma es **enlazar** dos mensajes que han de ir a receptores distintos:
 - la información del **pago** (Payment Information) al banco
 - la información del **pedido** (Order Information) al comerciante



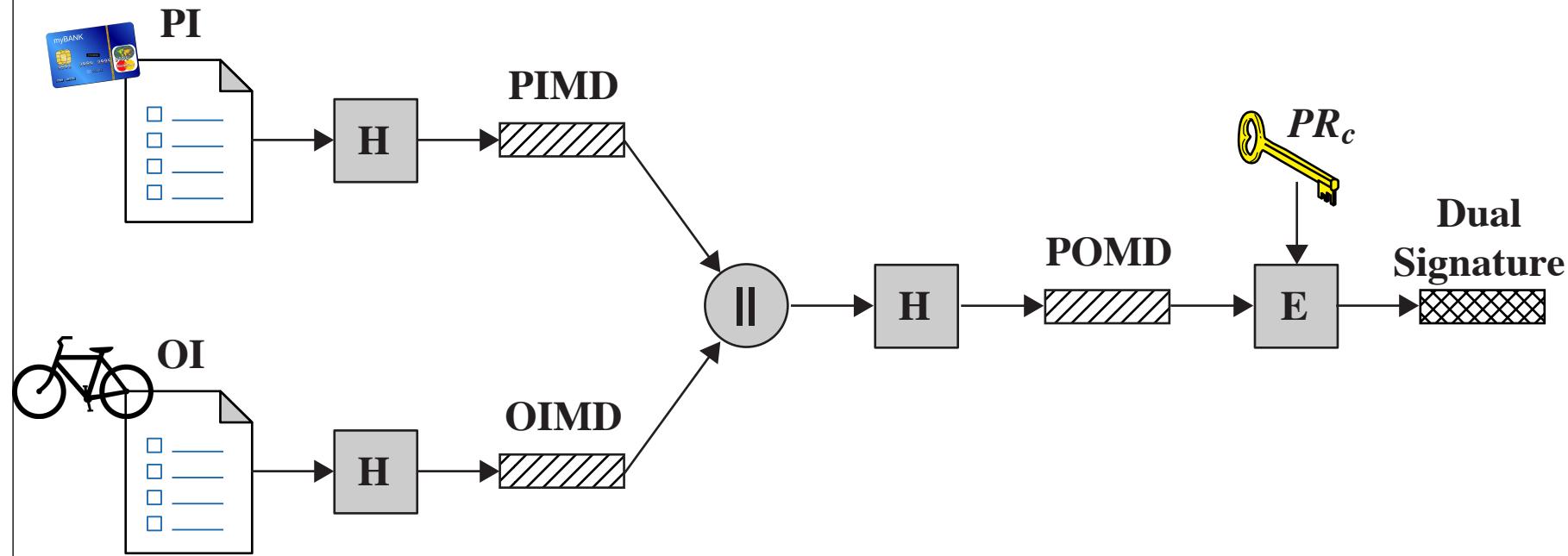
FIRMA DUAL (en SET)

- SET introduce una importante innovación técnica: la **firma dual**
 - El propósito de este tipo de firma es **enlazar** dos mensajes que han de ir a receptores distintos:
 - la información del **pago** (Payment Information) al banco
 - la información del **pedido** (Order Information) al comerciante
- Con la firma dual se ofrece mayor **privacidad** al cliente si ambos ítems (PI y OI) se mantienen por separado:
 - ni el comerciante necesita conocer el número de tarjeta del cliente
 - ni el banco necesita conocer los detalles del pedido del cliente
 - pero también es necesario que ambos ítems queden enlazados de alguna forma, para una posible **resolución de disputas** posterior



¡¡HAY NO-REPUDIO!!

FIRMA DUAL (en SET)



PI = Payment Information

OI = Order Information

H = Hash function (SHA-1)

|| = Concatenation

PIMD = PI message digest

OIMD = OI message digest

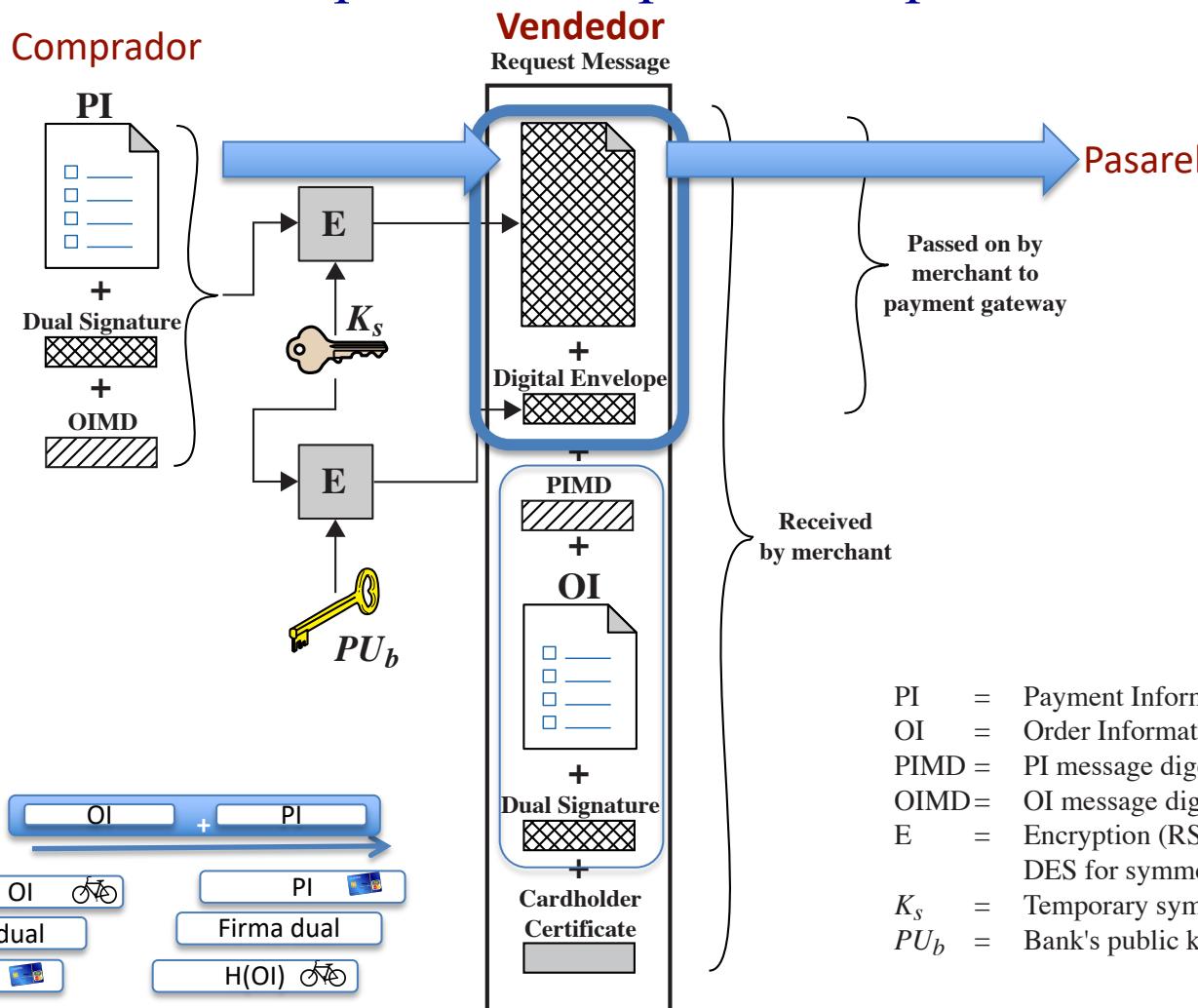
POMD = Payment Order message digest

E = Encryption (RSA)

PR_c = Customer's private signature key

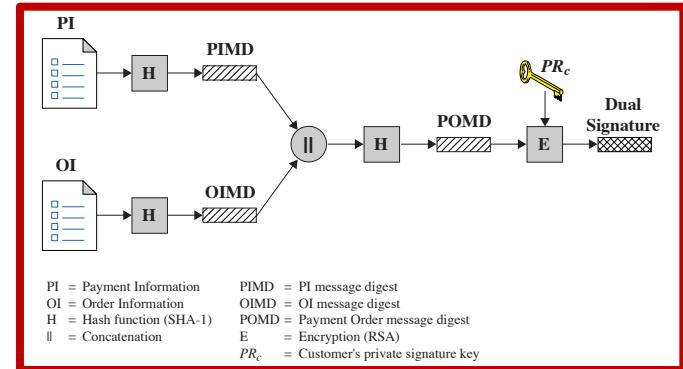
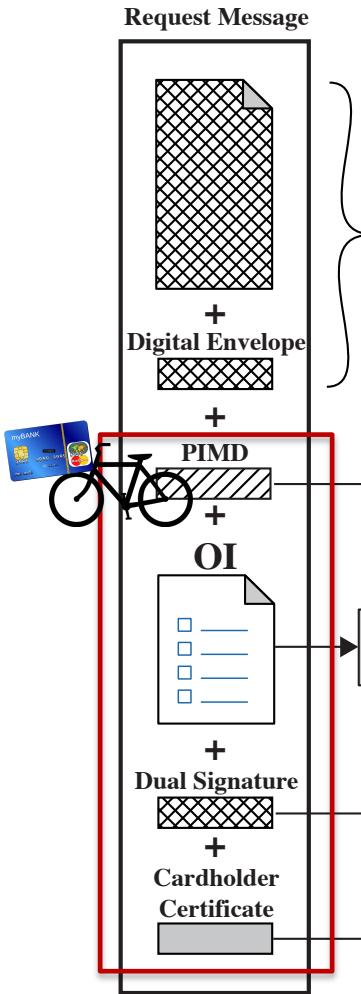
FIRMA DUAL (en SET)

- Petición de compra enviada por el comprador al vendedor:



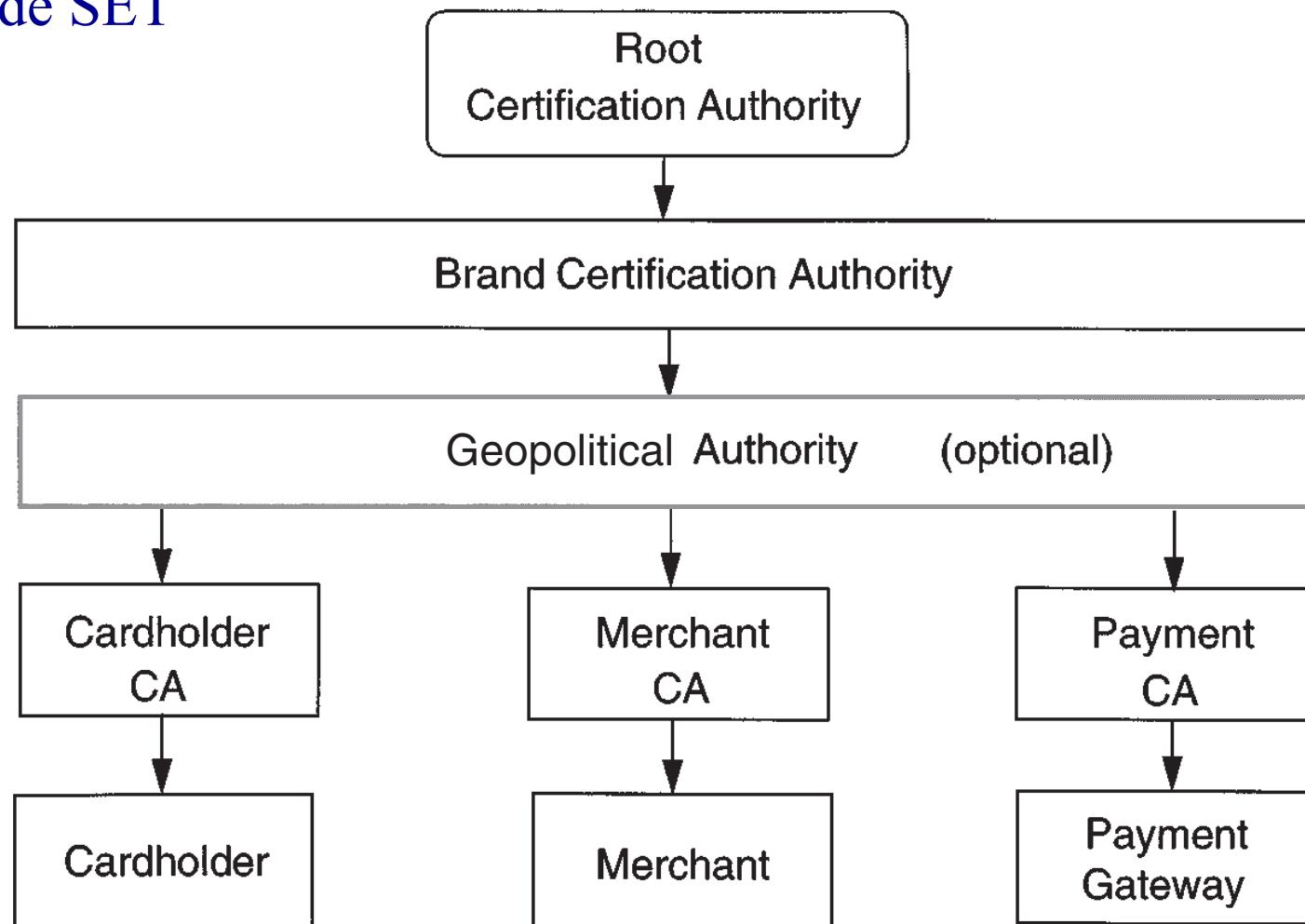
FIRMA DUAL (en SET)

- Verificación del vendedor:



Protocolo SET (Secure Electronic Transactions)

- PKI de SET



Protocolo SET (Secure Electronic Transactions)

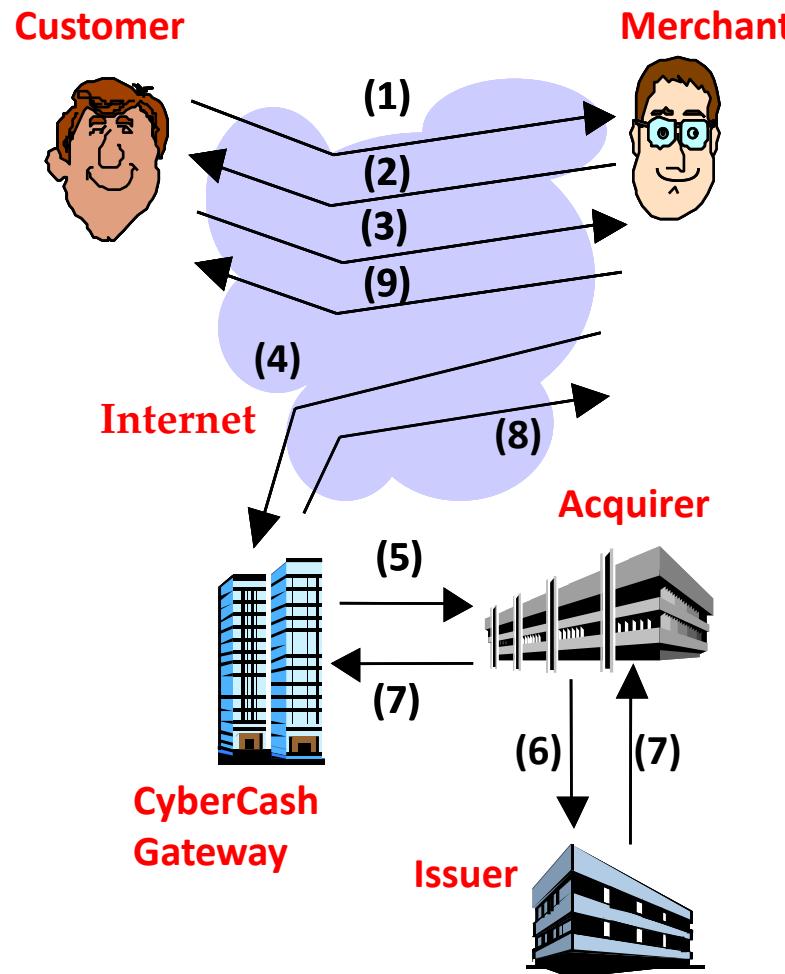
- Ventajas de uso del SET:
 - Muy seguro y bien diseñado
 - Garantiza:
 - autenticación, confidencialidad, integridad y no-repudio
 - Garantiza privacidad
 - evita que el vendedor acceda a los datos de la tarjeta
 - evita que el banco acceda a la información de los productos comprados
- Desventajas de uso del SET:
 - Es dependiente de algoritmos específicos (RSA, DES, SHA1)
 - Gestión compleja de certificados digitales
 - fuerte esfuerzo para la implantación (especialmente para el vendedor)
 - No adaptado a micropagos

Protocolo Cybercash

- Se basa en el uso de una **pasarela propia** que gestiona los pagos electrónicos
 - permite el uso de cualquier tipo de tarjeta
 - integra el software de cliente (**cyberwallet**) con la red financiera del banco del comprador
- Se realiza la **autenticación** de todas las entidades y el **cifrado** de los datos relativos al pago



Protocolo Cybercash



1. Purchase order (description).
2. Payment request (price).
3. Payment order (signed by the CyberWallet).
4. Redirection of the payment order.
5. Verification of the order. Authorization request.
6. Request for authorization of the issuer bank.
7. Authorization reply (acquirer and gateway).
8. Sending the **encrypted bills** (customer and merchant)
9. Redirection of the **customer's bill**.

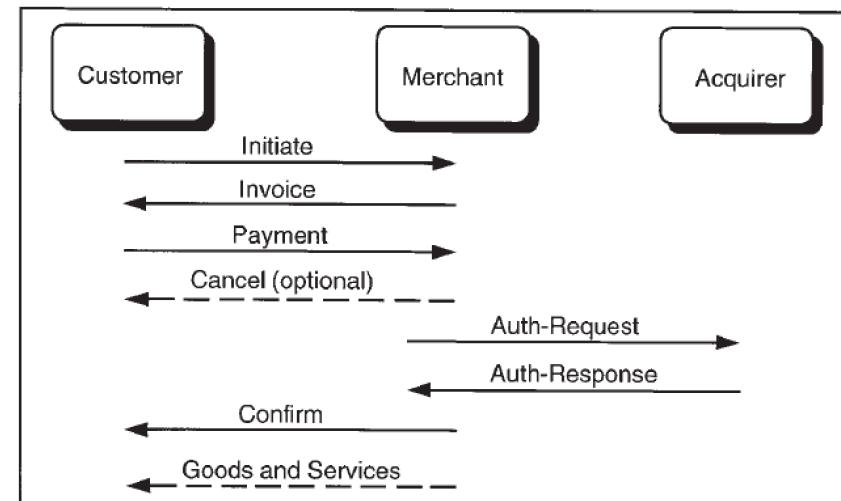
Protocolo Cyberscash

- Sus principales problemas son:
 - Parte de la información del cliente es conocida por la pasarela “privada”, por lo que se pueden analizar los hábitos del cliente
 - **problema de privacidad** que no existía en SET
 - Hace uso de **DES y RSA (1024 bits)**
- Tras caer en bancarrota, Verisign adquirió los derechos sobre la marca y el protocolo de pago
- Posteriormente, Paypal compró la solución a Verisign



Protocolos i-Key Protocol (iKP)

- iKP ($i = 1, 2, 3$) es una familia de protocolos de pago, basado en criptografía de clave pública, y desarrollado por IBM
- La implantación de los protocolos se realiza de forma gradual para conseguir **un pago seguro (multiparte – si se aplica 2/3KP)** completo, requiriendo una **infraestructura de certificación avanzada**
 - Estos protocolos (1KP, 2KP, 3KP) se diferencian entre sí dependiendo del nº de entidades que poseen el certificado digital
- El funcionamiento del pago es equivalente a otros protocolos



Protocolos i-Key Protocol (iKP)

- Los elementos intercambiados en una transacción iKP, y los campos formados por la combinación de tales elementos, son:

Item	Description
CAN	Customer's account number (e.g., credit card number)
ID _M	Merchant ID; identifies merchant to acquirer
TID _M	Transaction ID; uniquely identifies the transaction
DESC	Description of the goods; includes payment information such as credit card holder's name and bank identification number
SALT _C	Random number generated by C; used to randomize DESC and thus ensure privacy of DESC on the M to A link
NONCE _M	Random number generated by a merchant to protect against replay
DATE	Merchant's current date/time
PIN	Customer's PIN which, if present, can be optionally used in 1KP to enhance security
Y/N	Response from card issuer; Yes/No or authorization code
R _C	Random number chosen by C to form CID
CID	A customer pseudo-ID which uniquely identifies C; computed as CID = H(R _C , CAN)
V	Random number generated in 2KP and 3KP by merchant; used to bind the Confirm and Invoice message flows

Item	Description
Common	Information held in common by all parties: PRICE, ID _M , TID _M , DATE, NONCE _M , CID, H(DESC, SALT _C), [H(V)]
Clear	Information transmitted in the clear: ID _M , TID _M , DATE, NONCE _M , H(Common), [H(V)]
SLIP	Payment instructions: PRICE, H(Common), CAN, R _C , [PIN]
EncSlip	Payment instruction encrypted with the public key of the acquirer: PK _A (SLIP)
CERT _X	Public-key certificate of X, issued by a CA
Sig _A	Acquirer's signature: SK _A [H(Y/N, H(Common))]
Sig _M	Merchant's signature in Auth-Request: SK _M [H(H(Common), [H(V)])]
Sig _C	Cardholder's signature: SK _C [H(EncSlip, H(Common))]

Importante: no hay que estudiar ni memorizar estos datos

Protocolos i-Key Protocol (iKP)

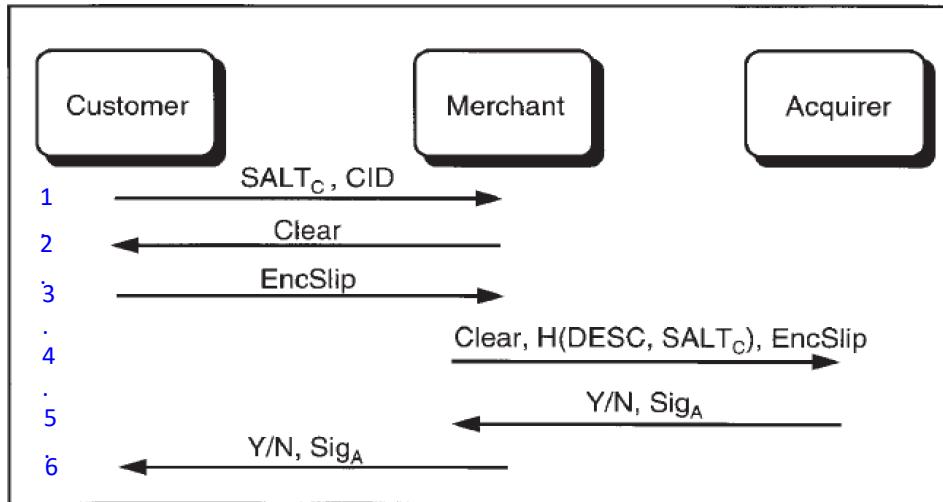
- El **protocolo 1KP** es el más básico
 - Sólo el banco necesita poseer (y distribuir) su certificado digital, CERT_A .
- La información de partida de cada una de las entidades es:

Item	Description
CAN	Customer's account number (e.g., credit card number)
ID_M	Merchant ID; identifies merchant to acquirer
TID_M	Transaction ID; uniquely identifies the transaction
DESC	Description of the goods; includes payment information such as credit card holder's name and bank identification number
SALT_C	Random number generated by C; used to randomize DESC and thus ensure privacy of DESC on the M to A link
NONCE_M	Random number generated by a merchant to protect against replay
DATE	Merchant's current date/time
PIN	Customer's PIN which, if present, can be optionally used in 1KP to enhance security
Y/N	Response from card issuer; Yes/No or authorization code
R_C	Random number chosen by C to form CID
CID	A customer pseudo-ID which uniquely identifies C; computed as $\text{CID} = H(R_C, \text{CAN})$
V	Random number generated in 2KP and 3KP by merchant; used to bind the Confirm and Invoice message flows

Actor	Information Items
Customer	$\text{DESC}, \text{CAN}, \text{PK}_{CA}, [\text{PIN}], \text{CERT}_A$
Merchant	$\text{DESC}, \text{PK}_{CA}, \text{CERT}_A$
Acquirer	$\text{SK}_A, \text{CERT}_A$

Protocolos i-Key Protocol (iKP)

- Los pasos del protocolo 1KP son:



Item	Description
Common	Information held in common by all parties: PRICE, ID_M , TID_M , DATE, $NONCE_M$, CID, $H(DESC, SALT_C)$, $[H(V)]$
Clear	Information transmitted in the clear: ID_M , TID_M , DATE, $NONCE_M$, $H(\text{Common})$, $[H(V)]$
SLIP	Payment instructions: PRICE, $H(\text{Common})$, CAN, R_C , [PIN]
EncSlip	Payment instruction encrypted with the public key of the acquirer: PK_A (SLIP)
CERT_X	Public-key certificate of X, issued by a CA
Sig_A	Acquirer's signature: $SK_A[H(Y/N, H(\text{Common}))]$
Sig_M	Merchant's signature in Auth-Request: $SK_M[H(H(\text{Common}), [H(V]))]$
Sig_C	Cardholder's signature: $SK_C[H(\text{EncSlip}, H(\text{Common}))]$

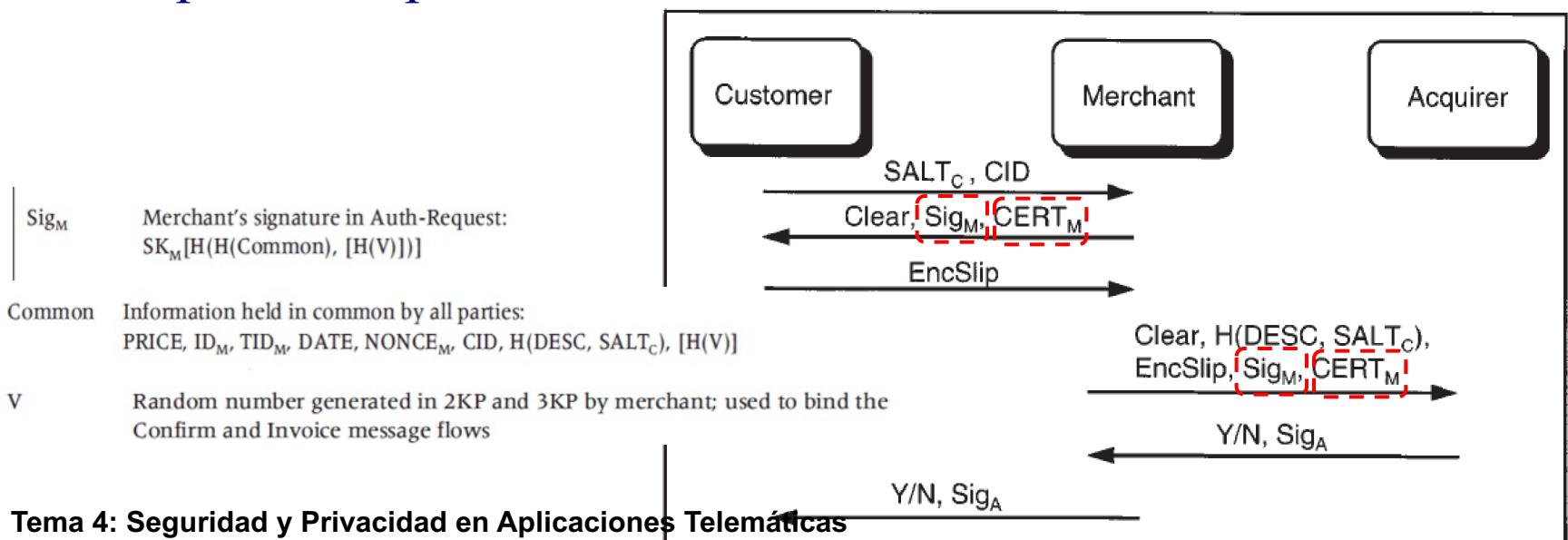
- Las desventajas de uso de 1KP son:

- El cliente se autentica utilizando sólo un número de tarjeta de crédito y, opcionalmente, un PIN, en lugar de firmas digitales
- El vendedor no se autentica ni ante el cliente ni ante al banco
- Ni el vendedor ni el cliente proporcionan evidencias de intervención en la transacción

Protocolos i-Key Protocol (iKP)

- En el **protocolo 2KP**, además del banco, cada vendedor necesita tener un par *<clave pública, clave privada>*, y está obligado a distribuir su certificado $CERT_M$ al cliente y al banco
- La información de partida de cada una de las entidades es:
- Los pasos del protocolo son:

Actor	Information Items
Customer	DESC, CAN, PK_{CA} , $CERT_A$
Merchant	DESC, PK_{CA} , $CERT_A$, SK_M , $CERT_M$
Acquirer	PK_{CA} , SK_A , $CERT_A$



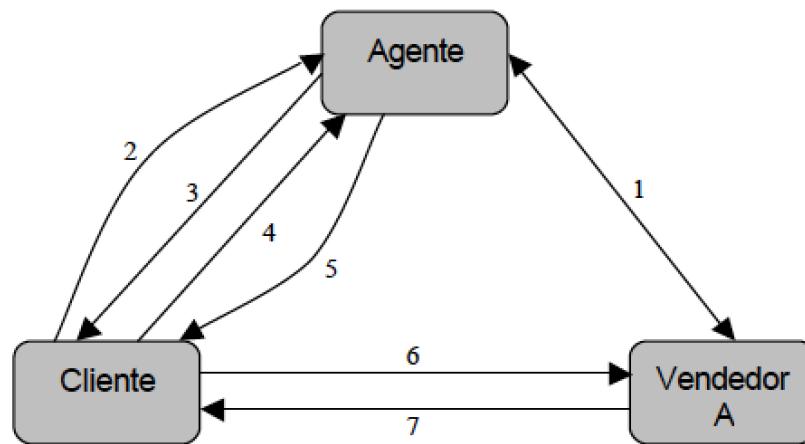
Micropagos

- En algunos escenarios hay que transferir una cantidad muy pequeña (**micropago**), y, por ello, hay que buscar la forma más eficiente y económica posible de hacerlo
 - minimizando el tráfico y los recursos utilizados, para que los costes de realizar el pago sean mínimos en comparación el pago en sí mismo
- Para reducir los costes e pueden utilizar varias soluciones:
 - servicios de prepago
 - autorizaciones off-line (ej. pagos en efectivos)
 - agrupación de facturas para el micropago por lotes
 - soluciones “online” un poco más complejas usando criptografía
 - en este caso, la criptografía de clave pública no es la más adecuada puesto que resulta costosa, e incluso, algunos criptosistemas simétricos pueden ser cuestionables
 - cada vez más se aplica el empleo de funciones hash
 - » sin embargo, conlleva también a la imposibilidad de garantizar el e no-repudio

Protocolo Millicent

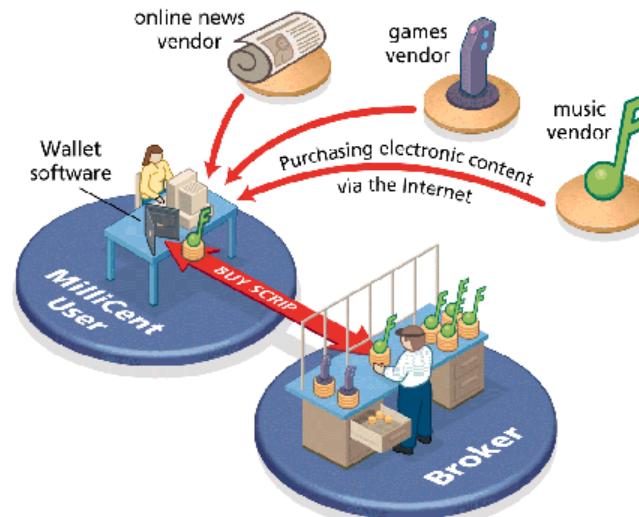
- Un ejemplo de protocolo para gestionar el micropago es **Millicent**:
 - basado en criptografía simétrica y NO utiliza procesamiento online
- Además de clientes y comerciantes, en Millicent existe la figura del **agente de negocios** (posiblemente una institución financiera)
- El sistema utiliza una forma de moneda electrónica, el **scrip**
 - los scrips vienen a ser “cupones electrónicos” que representan dinero, con los que el comprador obtiene la mercancía del vendedor
 - Para un cliente no sería eficiente comprar lotes de scrips para cada vendedor sino más bien para un conjunto de vendedores
 - se puede suponer que, durante un periodo de tiempo las compras de un cliente a varios comerciantes alcanzarán un importe equivalente a un macropago
- La función principal del **agente de negocios / broker** es la de vender a cada cliente, y dentro de un mismo lote mixto, scrips de distintos vendedores

Protocolo Millicent



1. Compra-Venta de scrips de A
2. Envío de scrips de agente
3. Envío de scrips de A
4. Envío de producto

2. Compra de scrips de agente (macropago)
4. Compra de scrips de A (micropago mediante scrips de agente)
6. Petición producto + micropago mediante scrips de A



Protocolo Millicent

- El modelo multiparte (cliente, vendedor y bróker) ayuda a tener cierto grado de **anonimato** por parte del comprador:
 - el agente conoce la identidad del comprador y su número de tarjeta de crédito, pero nunca llega a conocer qué producto compra
 - el vendedor sabe lo que el cliente compra, pero desconoce su identidad
- Hay otros muchos sistemas de micropago, como:
 - Subscrip
 - Kleline
 - Flattr
 - M-coin
 - etc.

PRIVACIDAD DE LOS USUARIOS EN APLICACIONES



Conceptos generales

- La **privacidad** puede definirse como:
 - *el derecho de los individuos y entidades de proteger, salvaguardar y controlar el acceso, almacenamiento, distribución y uso de información sobre su “propia persona”*
 - ¿Qué información es necesario proteger?
 - Dependerá de lo que el usuario considere información privada
 - Identidad, localización, preferencias, rutinas, ...
- **Confidencialidad NO es equivalente a privacidad**
 - Confidencialidad es relativa a los datos mientras que la privacidad es relativa a las personas
 - Por tanto: **Privacidad ≠ Confidencialidad**
 - Confidencialidad := mantener datos en secreto
 - Privacidad := mantener la integridad de la persona
- Hay varias formas de violar la privacidad, como:
 - Rastreo de actividad en la red
 - Análisis de tráfico

Conceptos generales - rastreo de act. en la red

- **Todo lo que se sube a la red es público** y se pierde el control sobre esa información 😞 ... ¡ así que cuidado !
 - Cualquiera puede encontrar los datos y utilizarlos para múltiples fines
 - Despidos, robos, discriminación, incriminación, ...
 - Las **redes sociales** ha facilitado la adquisición de información personal

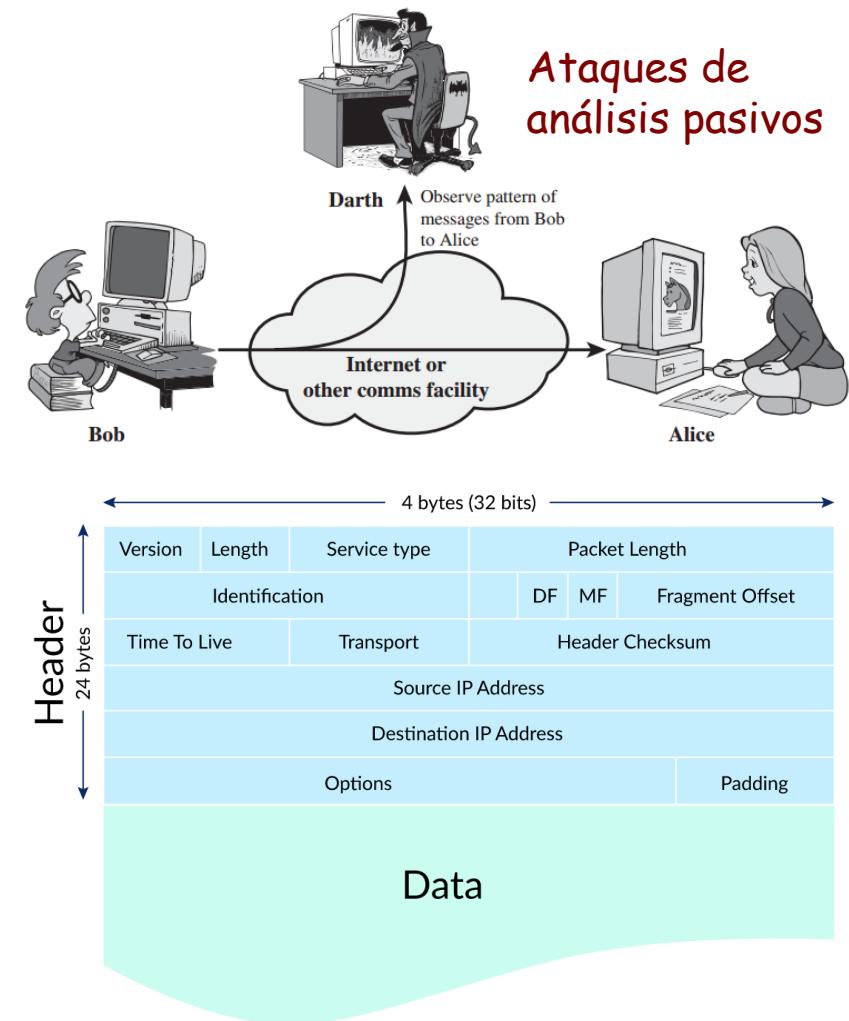
A screenshot of a website for "experto laboral online". The logo consists of a stylized 'el' icon followed by the text "experto laboral online". To the right is a menu icon (three horizontal lines). Below the logo is a photograph of a person with long blonde hair, seen from behind, wearing a red jacket, sitting at a desk and using a laptop. A camera is visible on the desk to the right.

Despidos, Derechos Laborales • 11 de febrero de 2014 •
por Experto Laboral Online

Me han despedido estando de baja por unas fotos en Facebook ¿Es legal?

Conceptos generales - análisis de tráfico

- El análisis de tráfico permite a observadores externos obtener información a través de las comunicaciones de un usuario
 - Si la comunicación está en claro se puede acceder a la carga útil
 - El cifrado no elimina el problema porque el cifrado sólo protege los datos, y las cabeceras no se pueden cifrar (ej. dirección IP origen y destino), y además se puede estimar aspectos que no se pueden controlar con el cifrado como el tamaño, número de paquetes, frecuencia y tiempos de envío, ...



Conceptos generales - análisis de tráfico

Low Disk Space: Your storage disk is almost full. 3.0G is available.

Guest upload is turned off Log In

tcp1337-netcat-chat.pcapng 1.4 kb · 12 packets · more info

Start typing a Display Filter ✓ Apply Clear Filters ▾

No.	No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	0.000000	10.185.220.101	10.185.220.139	TCP	62	2764 → 1337 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 SACK_PERM=1
2	0.000242	0.000242	10.185.220.139	10.185.220.101	TCP	62	1337 → 2764 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 SACK_PERM=1
3	0.000313	0.000313	10.185.220.101	10.185.220.139	TCP	54	2764 → 1337 [ACK] Seq=1 Ack=1 Win=64240 Len=0
4	1.831059	1.831059	10.185.220.101	10.185.220.139	TCP	59	2764 → 1337 [PSH, ACK] Seq=1 Ack=1 Win=64240 Len=5
5	1.832836	1.832836	10.185.220.139	10.185.220.101	TCP	60	1337 → 2764 [ACK] Seq=1 Ack=6 Win=65535 Len=0
6	9.291432	9.291432	10.185.220.139	10.185.220.101	TCP	66	1337 → 2764 [PSH, ACK] Seq=1 Ack=6 Win=65535 Len=12
7	9.491448	9.491448	10.185.220.101	10.185.220.139	TCP	54	2764 → 1337 [ACK] Seq=6 Ack=13 Win=64228 Len=0
8	15.405658	15.405658	10.185.220.139	10.185.220.101	TCP	60	1337 → 2764 [FIN, ACK] Seq=13 Ack=6 Win=65535 Len=0
9	15.405740	15.405740	10.185.220.101	10.185.220.139	TCP	54	2764 → 1337 [ACK] Seq=6 Ack=14 Win=64228 Len=0

Internet Protocol Version 4, Src: 10.185.220.139, Dst: 10.185.220.101
Transmission Control Protocol, Src Port: 1337, Dst Port: 2764, Seq: 1, Ack: 6, Len: 0
Source Port: 1337
Destination Port: 2764
[Stream index: 0]
[Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 0]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 3979006224
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 6 (relative ack number)
Acknowledgment number (raw): 2042183646
0101 = Header Length: 20 bytes (5)
Flags: 0x010 (ACK)
Window: 65535
[Calculated window size: 65535]
[Window size scaling factor: -2 (no window scaling used)]
Checksum: 0x5999 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
[Timestamps]
[SEQ/ACK analysis]

Low Disk Space: Your storage disk is almost full. 3.0G is available.

Guest upload is turned off Log In

tcp1337-netcat-chat.pcapng 1.4 kb · 12 packets · more info

Start typing a Display Filter ✓ Apply Clear Filters ▾

No.	No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	0.000000	10.185.220.101	10.185.220.139	TCP	62	2764 → 1337 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 SACK_PERM=1
2	0.000242	0.000242	10.185.220.139	10.185.220.101	TCP	62	1337 → 2764 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 SACK_PERM=1
3	0.000313	0.000313	10.185.220.101	10.185.220.139	TCP	54	2764 → 1337 [ACK] Seq=1 Ack=1 Win=64240 Len=0
4	1.831059	1.831059	10.185.220.101	10.185.220.139	TCP	59	2764 → 1337 [PSH, ACK] Seq=1 Ack=1 Win=64240 Len=5
5	1.832836	1.832836	10.185.220.139	10.185.220.101	TCP	60	1337 → 2764 [ACK] Seq=1 Ack=6 Win=65535 Len=0
6	9.291432	9.291432	10.185.220.139	10.185.220.101	TCP	66	1337 → 2764 [PSH, ACK] Seq=1 Ack=6 Win=65535 Len=12
7	9.491448	9.491448	10.185.220.101	10.185.220.139	TCP	54	2764 → 1337 [ACK] Seq=6 Ack=13 Win=64228 Len=0
8	15.405658	15.405658	10.185.220.139	10.185.220.101	TCP	60	1337 → 2764 [FIN, ACK] Seq=13 Ack=6 Win=65535 Len=0
9	15.405740	15.405740	10.185.220.101	10.185.220.139	TCP	54	2764 → 1337 [ACK] Seq=6 Ack=14 Win=64228 Len=0

Internet Protocol Version 4, Src: 10.185.220.139, Dst: 10.185.220.101
Transmission Control Protocol, Src Port: 1337, Dst Port: 2764, Seq: 1, Ack: 6, Len: 0
Source Port: 1337
Destination Port: 2764
[Stream index: 0]
[Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 0]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 3979006224
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 6 (relative ack number)
Acknowledgment number (raw): 2042183646
0101 = Header Length: 20 bytes (5)
Flags: 0x010 (ACK)
Window: 65535
[Calculated window size: 65535]
[Window size scaling factor: -2 (no window scaling used)]
Checksum: 0x5999 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
[Timestamps]
[SEQ/ACK analysis]

Follow TCP: tcp.stream eq 0 in tcp1337-netcat-chat.pcapng
Show only this stream | Filter out this stream

test
return test

Low Disk Space: Your storage disk is almost full. 3.0G is available.

Guest upload is turned off Log In

tcp1337-netcat-chat.pcapng 1.4 kb · 12 packets · more info

Start typing a Display Filter ✓ Apply Clear Filters ▾

No.	No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	0.000000	10.185.220.101	10.185.220.139	TCP	62	2764 → 1337 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 SACK_PERM=1
2	0.000242	0.000242	10.185.220.139	10.185.220.101	TCP	62	1337 → 2764 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 SACK_PERM=1
3	0.000313	0.000313	10.185.220.101	10.185.220.139	TCP	54	2764 → 1337 [ACK] Seq=1 Ack=1 Win=64240 Len=0
4	1.831059	1.831059	10.185.220.101	10.185.220.139	TCP	59	2764 → 1337 [PSH, ACK] Seq=1 Ack=1 Win=64240 Len=5
5	1.832836	1.832836	10.185.220.139	10.185.220.101	TCP	60	1337 → 2764 [ACK] Seq=1 Ack=6 Win=65535 Len=0
6	9.291432	9.291432	10.185.220.139	10.185.220.101	TCP	66	1337 → 2764 [PSH, ACK] Seq=1 Ack=6 Win=65535 Len=12
7	9.491448	9.491448	10.185.220.101	10.185.220.139	TCP	54	2764 → 1337 [ACK] Seq=6 Ack=13 Win=64228 Len=0
8	15.405658	15.405658	10.185.220.139	10.185.220.101	TCP	60	1337 → 2764 [FIN, ACK] Seq=13 Ack=6 Win=65535 Len=0
9	15.405740	15.405740	10.185.220.101	10.185.220.139	TCP	54	2764 → 1337 [ACK] Seq=6 Ack=14 Win=64228 Len=0

Internet Protocol Version 4, Src: 10.185.220.139, Dst: 10.185.220.101
Transmission Control Protocol, Src Port: 1337, Dst Port: 2764, Seq: 1, Ack: 6, Len: 0
Source Port: 1337
Destination Port: 2764
[Stream index: 0]
[Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 0]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 3979006224
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 6 (relative ack number)
Acknowledgment number (raw): 2042183646
0101 = Header Length: 20 bytes (5)
Flags: 0x010 (ACK)
Window: 65535
[Calculated window size: 65535]
[Window size scaling factor: -2 (no window scaling used)]
Checksum: 0x5999 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
[Timestamps]
[SEQ/ACK analysis]

Entire Conversation ASCII Hex Dump Wrap long lines

Ladder Diagram Open in new window Done

Fuente: cloudshark.org

Conceptos generales

- Existen dos **enfoques complementarios** para proteger la privacidad

- **Enfoque legislativo:**

- Creación de leyes que impongan **sanciones** para limitar prácticas abusivas de las empresas
 - En Europa tenemos el Reglamento General de Protección de Datos (**GDPR** - General Data Protection Regulation)

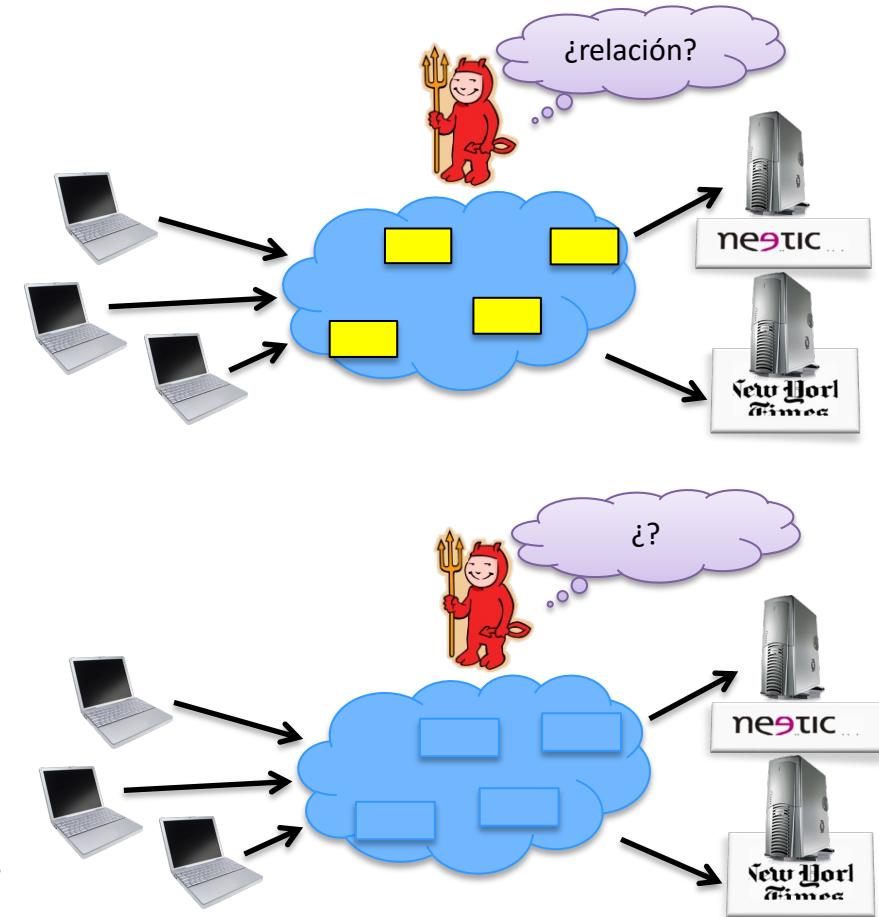


- **Enfoque tecnológico:**

- Existen mecanismos de preservación de la privacidad, principalmente basados en algoritmos criptográficos y probabilísticos, y también en conceptos basados en aleatorización

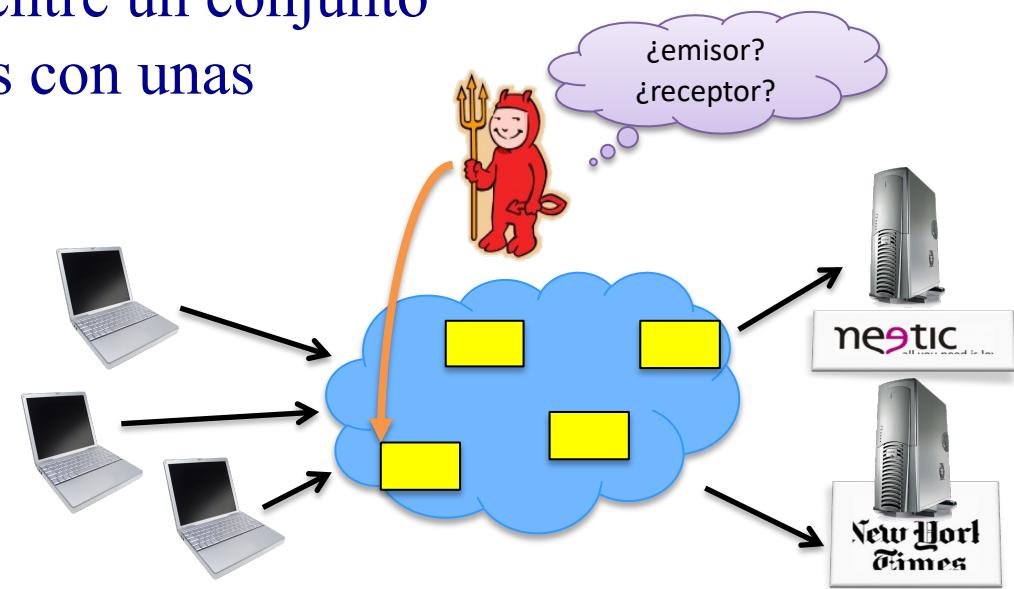
Conceptos generales

- Propiedades de la privacidad:
 - **No-vinculación / no enlazabilidad** (unlinkability): se refiere a la incapacidad de un atacante para relacionar dos mensajes o entidades observadas
 - **No-observabilidad** (unobservability): se refiere a la imposibilidad de distinguir la presencia de mensajes, ni la identidad de la entidades



Conceptos generales

- Relacionado con el concepto de privacidad, está el concepto de **anonimato**:
 - Anonimato se puede definir como “*el estado de no ser identificable entre un grupo de sujetos*”
 - Las técnicas de anonimato tratan de hacer indistinguible (u ocultar) a un individuo entre un conjunto suficiente de identidades con unas características similares



Conceptos generales

- El anonimato depende de la técnica aplicada y se distinguen 4 clases de técnicas:
 - **Pseudónimos:**
 - Se basa de técnicas para ocultar la identidad de un usuario a través de pseudónimos
 - Sin embargo, el uso continuado de pseudónimos pueden ser vinculantes, es decir, que se puede derivar la identidad del usuario, y todas las operaciones previas realizadas
 - **Anonimato rastreable:**
 - Ofrecer anonimato, pero en caso de necesidad se puede revelar la identidad del usuario
 - Ej. el vendedor y el banco: dependiendo de la honestidad de estas partes se puede o no revelar la identidad del usuario principal
 - **Anonimato no rastreable:**
 - Trata de resolver el problema del anonimato pero garantiza que la identidad de los usuarios no se va a revelar
 - **Anonimato no rastreable y no vinculante:**
 - Además de garantizar que la identidad de los usuarios no se puede revelar, la condición de no vinculante asegura que el conjunto de las operaciones realizadas por un usuario no se puedan vincular

Conceptos generales

- Para conseguir una o varias de las propiedades de privacidad mencionadas anteriormente, las soluciones suelen basarse en el uso de algunas de las siguientes técnicas:
 - **Esquemas avanzados de firma digital**
 - **Protocolos criptográficos y de enrutado**
 - Evitan que la dirección de red o el camino que siguen los paquetes puedan identificar a las partes comunicantes
 - **Técnicas de ofuscación**
 - Son mecanismos basados principalmente en la generalización o supresión de información para limitar la precisión de la información que se revela

Privacidad basada en
esquemas avanzados de firma digital

Esquemas avanzados de firma digital

- A partir del concepto básico de firma digital surgen esquemas más avanzados de firma digital con objetivos más ambiciosos:
 - **Firma ciega**
 - El firmante firma mensajes para otros usuarios, pero desconoce el contenido de los mensajes que firma
 - La firma se puede verificar posteriormente – **anonimato rastreable**
 - **Firma de grupo**
 - Cualquier miembro del grupo firma mensajes de forma anónima en nombre del grupo
 - En caso de disputa, una entidad determinada puede revelar la identidad del firmante – **anonimato rastreable**
 - **Firma de anillo**
 - Similar al anterior, pero el anonimato es total; es decir, no es posible saber la identidad del firmante bajo ninguna circunstancia – **anonimato no rastreable ni vinculante**

Firma ciega

- Existen aplicaciones (p. ej. el voto electrónico), donde una de las propiedades a preservar es el anonimato del usuario
- Concretamente, con la firma ciega, lo que se consigue es que el mensaje M generado por *Bob* sea firmado por *Alice* sin que esta conozca el contenido del mensaje
- La firma ciega resultante puede ser verificada públicamente con posterioridad, como una firma digital normal



Firma ciega

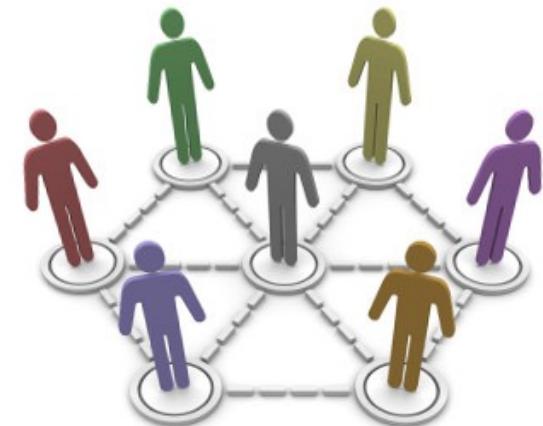
- El esquema de firma ciega se puede implementar con diferentes algoritmos de clave pública, entre ellos, RSA
 - Sean e y d , las claves pública y privada de *Alice*
 - Sea n el módulo RSA
 - Y sea r un número aleatorio $\text{mod } n$
(r es el *blinding factor*)



1. Bob: r
2. Bob: $M' = M \cdot r^e \pmod{n}$
3. Bob \rightarrow Alice: M'
4. Alice \rightarrow Bob: $(M')^d \pmod{n} = [M^d \cdot (r^e)^d \pmod{n}] = [M^d \cdot r \pmod{n}]$
5. Bob: $M^d \cdot r \cdot r^{-1} \pmod{n} = [M^d \pmod{n}]$

Firma de grupo

- Al contrario que con los esquemas tradicionales de firma, en los que sólo hay un firmante:
 - los esquemas de firma en grupo permiten que cualquier miembro de un grupo firme en nombre del grupo
 - Ej. un empleado de un banco firma en nombre del banco
- La figura del **administrador del grupo** controla quién pertenece al mismo y también emite la clave de firma del grupo
 - o sea, la clave con la que cualquier miembro firma en nombre del grupo



Firma de grupo

- De lo anterior, se puede deducir que en la utilización de estos esquemas de firma hay tres tipos de **participantes**:
 - Administrador del grupo
 - Miembro del grupo
 - Verificador (receptor del documento firmado)



Firma de grupo

- Un esquema de firma de grupo debe satisfacer las siguientes **propiedades iniciales** para cumplir la condición de “anonimato rastreable”:
 - Sólo los miembros del grupo pueden firmar mensajes de forma correcta (infalsifiable)
 - A excepción del administrador del grupo nadie puede descubrir:
 - qué miembro del grupo ha firmado el mensaje (anonimato)
 - si dos firmas han sido emitidas por el mismo miembro del grupo (no-vinculación)
 - Los miembros no pueden evitar la **apertura de la firma** por parte del administrador, ni firmar por otro

Firma de grupo

- Un esquema de firma de grupo consta de cuatro **procedimientos** distintos:
 - ① **Establecimiento** → es un protocolo entre el administrador y los miembros del grupo. Su ejecución origina:
 - la clave pública Y del grupo
 - las claves privadas individuales X_i de cada miembro del grupo
 - una clave secreta Z de administración para el administrador
 - ② **Firma** → es un algoritmo que: $S = E_{X_i}(M)$
 - tiene como entrada un mensaje M , y la clave privada de uno de los miembros del grupo
 - devuelve la firma S sobre el mensaje M

Firma de grupo

③ Verificación → es un algoritmo que: $E_Y(S) == M$

- recibe como entrada un mensaje M , una firma S , y la clave pública Y del grupo
- devuelve información de M si la firma S es correcta o no

④ Apertura → es un algoritmo que:

- recibe como entrada una firma S y la clave secreta de administración
- devuelve la identidad del miembro del grupo que realizó la firma S además de una prueba de este hecho
- Se asume que todas las comunicaciones entre el administrador y los miembros se llevan a cabo de forma segura

Firma de grupo

- **Ejemplo 1 - ejemplo básico de diseño:**
 - El administrador proporciona a cada miembro del grupo una lista de claves privadas
 - $L_1 = \{X_{1,1}, \dots, X_{1,n}\}$, $L_2 = \{X_{2,1}, \dots, X_{2,n}\}$, ..., $L_n = \{X_{n,1}, \dots, X_{n,n}\}$,
 - Esas listas son disjuntas, tal que: $L_1 \neq L_2$
 - El administrador divulga en un directorio público y en orden aleatorio, la lista completa de claves públicas correspondientes
 - $Y = L_1 = \{Y_{1,1}, \dots, Y_{1,n}\}$, $L_2 = \{Y_{2,1}, \dots, Y_{2,n}\}$, ..., $L_n = \{Y_{n,1}, \dots, Y_{n,n}\}$,
 - Esa lista completa hace las veces de clave pública del grupo
 - Cada miembro puede firmar un mensaje con una de las claves privadas de su lista
 - $L_1 = \{X_{1,1}, X_{2,1}, X_{3,1}, \dots, X_{1,n}\}$
 - Sólo utilizará la clave privada una vez para evitar la vinculación
 - El receptor puede verificar esa firma con la correspondiente clave pública (obtenida del directorio)
 - $X_{1,1} \rightarrow Y_{1,1}$
 - El administrador conoce todas las claves privadas, por lo que, en caso de ser necesario, puede saber fácilmente qué miembro del grupo realizó la firma



Firma de grupo

- **Ejemplo 2:**
 - Sea e el exponente público del grupo (es decir, Y)
 - Sea C_p el conjunto (p_1, p_2, \dots, p_L) de valores primos relativos a e
 - Sea n el módulo compuesto
- $$n = p_1 \cdot p_2 \cdot \dots \cdot p_L$$
- Se generan módulos individuales n_i para cada miembro
 - $$n_i = p_{i1} \cdot p_{i2}$$
- Se calculan los exponentes privados d_i (es decir, X_i) para cada miembro en base al exponente e y a los n_i

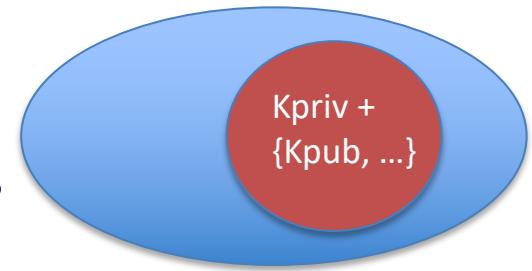
$$d_i = e^{-1} \bmod \theta(n_i)$$

Firma de grupo

- La **firma** por parte del miembro k se realiza así:
 - $S = M^{dk} \text{ mod } n_k$
- La **verificación** por parte de cualquier usuario se realiza así:
 - $M' = S^e \text{ mod } n$
- La **apertura de la firma**, en caso de que sea necesario, la realiza el administrador
 - Se realiza probando uno a uno la firma donde el administrador aplica la clave privada de cada miembro
 - $S_1 = M^{d1} \text{ (mod } n_1)$
 - $S_2 = M^{d2} \text{ (mod } n_2)$
 - ...
 - $S_N = M^{dn} \text{ (mod } n_n)$

Firma de anillo

- Al contrario que las firmas de grupo, las firmas de anillo:
 - no tienen ni administradores de grupo, ni procedimiento de establecimiento, ni procedimiento de revocación del anonimato del firmante, ni coordinación, ni procedimiento para distribuir claves
- Cualquier usuario puede elegir un conjunto de posibles firmantes incluyéndose él/ella mism@, sin la aprobación ni ayuda de esos otros miembros del conjunto
 - computa **la firma** por sí mism@, **usando sólo su clave privada y las claves públicas de los demás**
- Es decir, los otros miembros del conjunto pueden tener desconocimiento total de que sus claves públicas se están usando para ese proceso de firma
 - con el que quizás ni siquiera estén de acuerdo



Privacidad basada en
protocolos criptográficos y de enruteado

Protocolos criptográficos y de enrutado

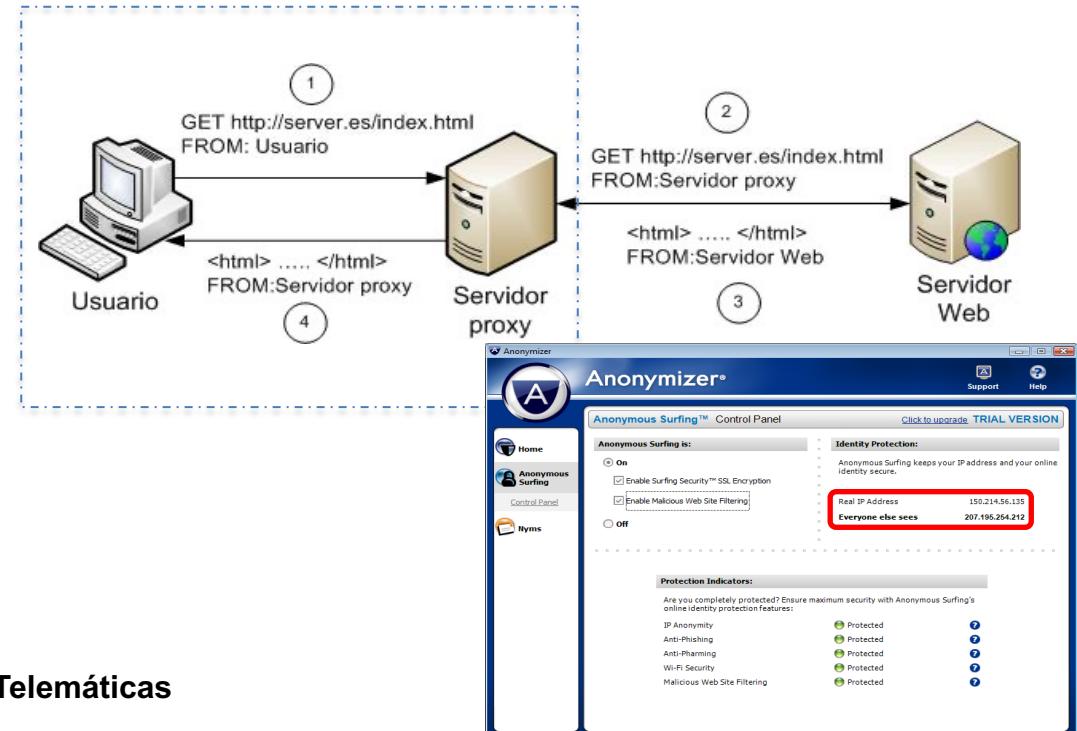
- Tratan de **proteger el tráfico** frente a entidades que se dedican a observar las comunicaciones
- Existen soluciones basadas en:
 - a) uso de proxy
 - b) uso de mezcladores
 - c) conocimiento parcial de la ruta
 - d) creación de gruposademás de algunas **soluciones híbridas**



Protocolos criptográficos y de enrutado

- a) **Basadas en el uso de proxy**: un servidor proxy hace de intermediario en la comunicación, aceptando conexiones de los clientes y reenviándolas
- ocultando así la dirección IP del verdadero origen, y proporcionando **anonimato respecto al destino** porque cree que el origen de la comunicación es el proxy

- Limitaciones:
 - No protegen frente a otros atacantes externos
 - El proxy puede ser honesto pero curioso



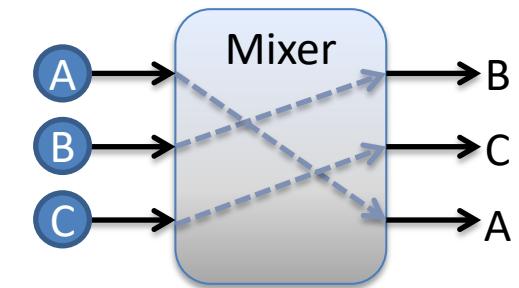
Protocolos criptográficos y de enrutado

b) **Basadas en el uso de mezcladores (mixers)**: es un tipo de proxy capaz de **ocultar la relación** entre los mensajes que recibe y los que envía, y son dispositivos de almacenamiento y envío

- el usuario envía el mensaje a través del mixer
- este lo almacenan durante cierto tiempo, con el fin de que pueda ser mezclado con otros mensajes recibidos, saliendo del mixer en un orden diferente
- el esquema funciona sólo si el número de mensajes almacenados temporalmente por el mixer es lo suficientemente grande

- **Limitaciones:**

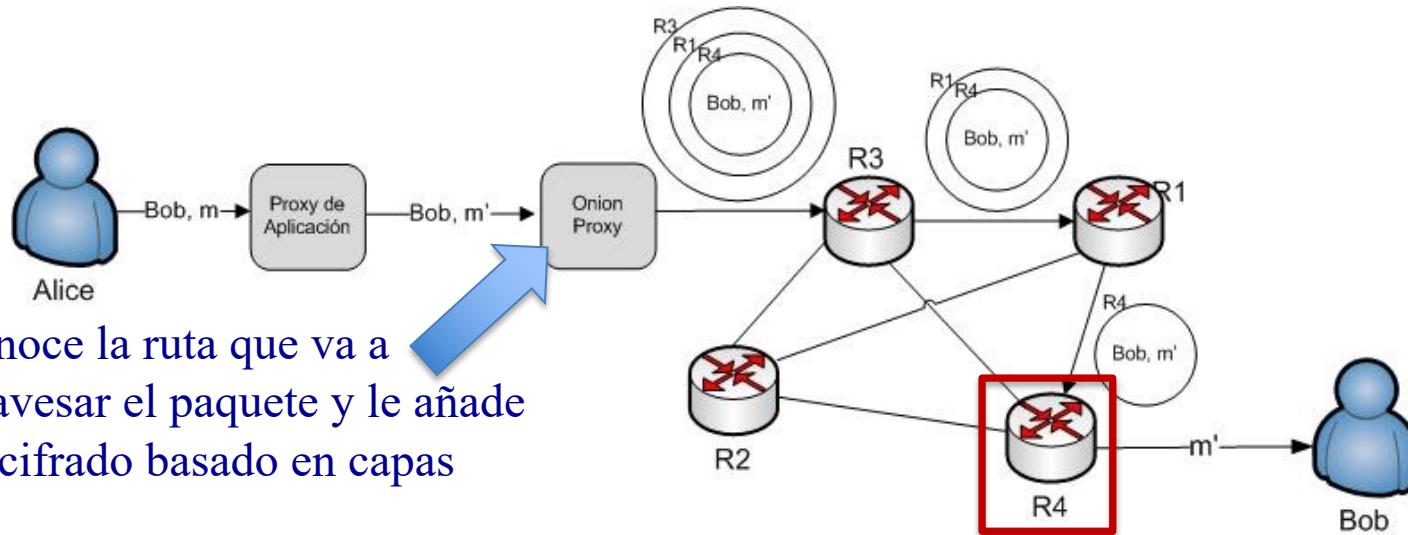
- Los retrasos introducidos pueden llegar a ser grandes
- El mezclador puede ser deshonesto u honesto pero curioso



Protocolos criptográficos y de enrutado

c) Basadas en el conocimiento parcial de la ruta

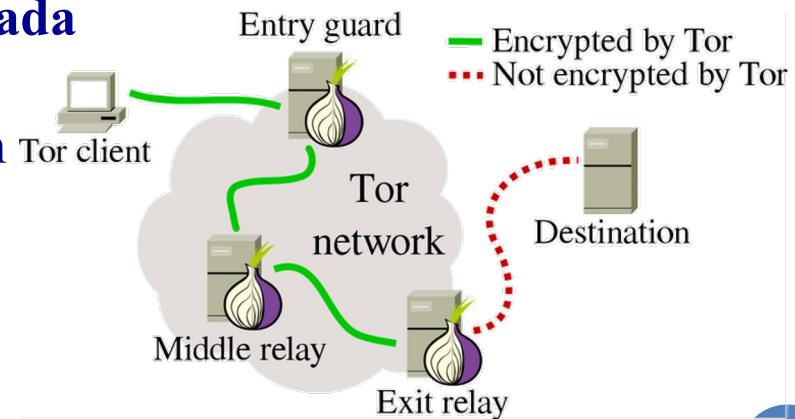
- **Onion Routing:** el onion proxy crea un paquete cifrado en capas, que se irán “pelando” a medida que atraviese el camino de onion routers



- Ataques en un extremo ☹
 - El atacante necesita controlar un extremo de la red

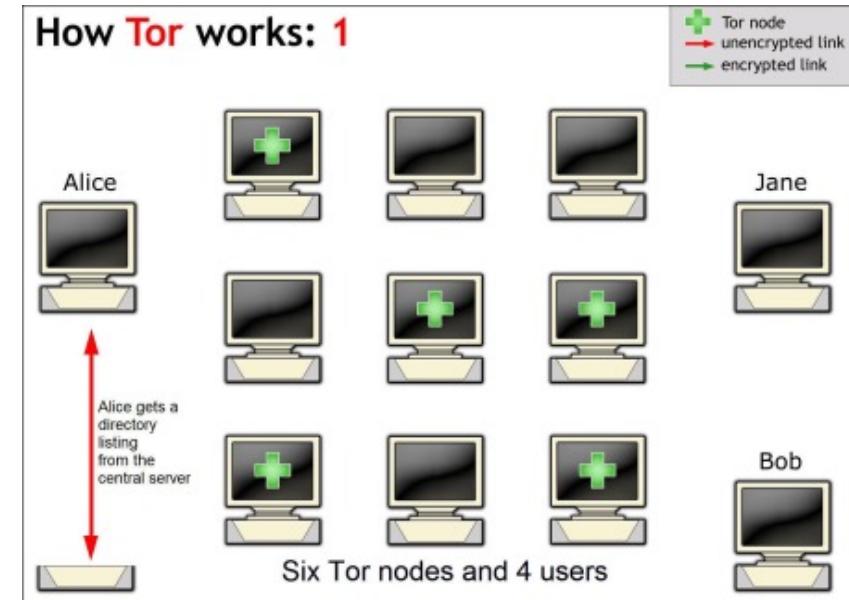
Protocolos criptográficos y de enrutado

- **TOR**: es la segunda generación de **Onion Routing**
 - evita algunas deficiencias del diseño original, añadiendo control de congestión, comprobación de integridad, etc.
 - permite navegar a través de una **ruta privada** de la red TOR, creada **de manera aleatoria** entre los distintos repetidores (onion routers) de la red
 - se negocian **nuevos caminos** a menudo y, tras su uso, las claves se borran para proporcionar *forward secrecy*
 - se aplican **guardianes de entrada** (**el onion proxy**) para reducir posibles ataques de correlación
 - **servidores ocultos** que sólo se pueden acceder desde dentro de la red Tor



Protocolos criptográficos y de enrutado

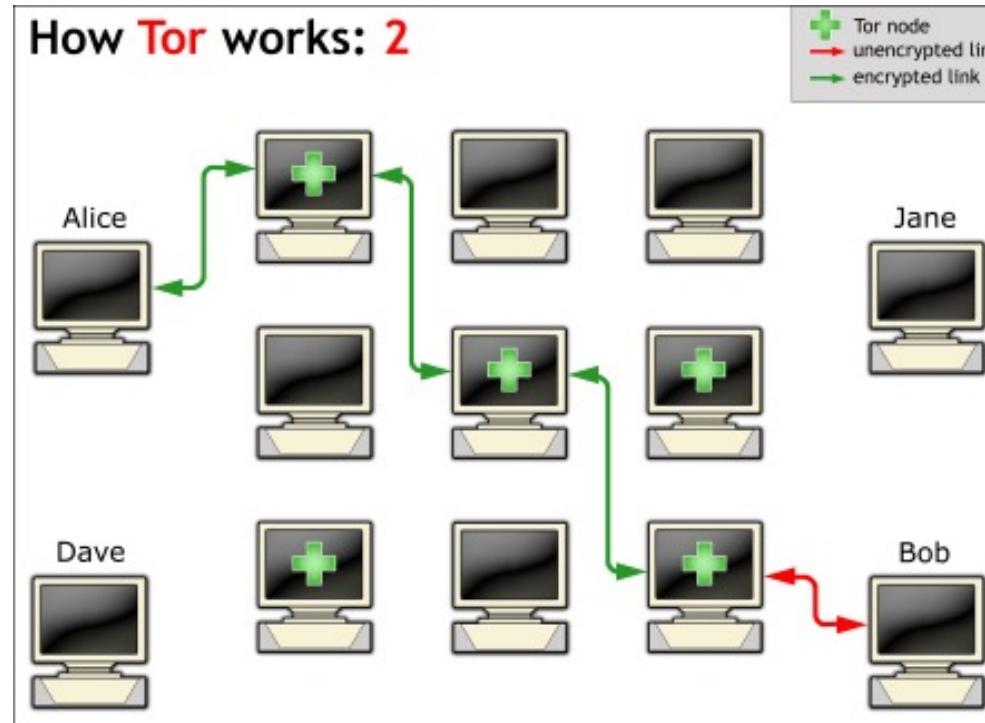
- Funcionamiento:
 - Enrutamiento → los paquetes se envían a través de varios *routers onions*, los cuales son elegidos de forma “aleatoria” y previamente por un “servidor central”
 - Todos los nodos Tor son elegidos al azar y ningún nodo puede ser utilizado dos veces



<https://geekytheory.com/que-es-y-como-funciona-la-red-tor>

Protocolos criptográficos y de enrutado

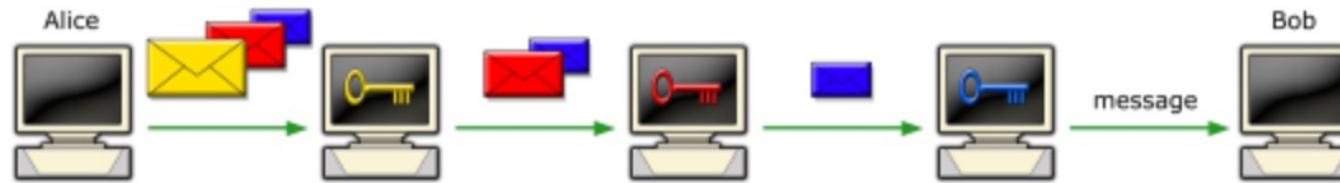
- Se conecta a un nodo aleatorio a través de una conexión cifrada
 - Una vez que el camino ya es conocido desde el origen, cada conexión y salto en la red deberá ser cifrada, excepto con el penúltimo nodo de la comunicación, el cual hará una conexión NO cifrada



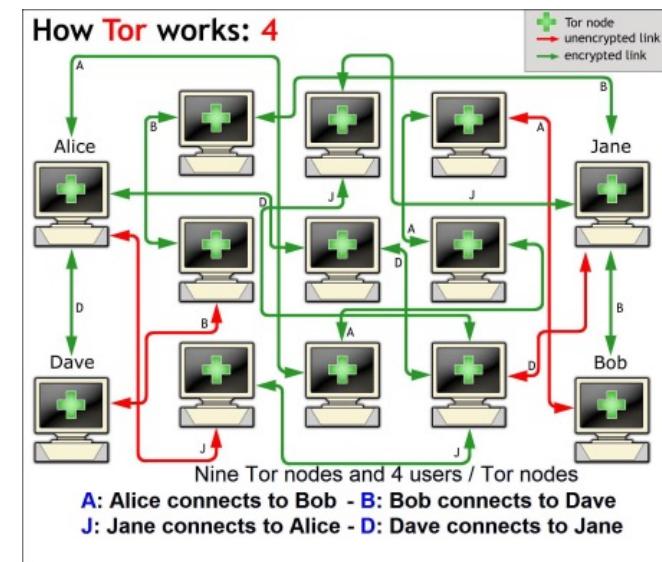
<https://geekytheory.com/que-es-y-como-funciona-la-red-tor>

Protocolos criptográficos y de enrutado

- El enrutado y el cifrado es “asimétrico” (*Onion Routing*):
 - Ej.: Alice lo primero que hace es cifrar el mensaje con la clave pública del último router onion de la lista, para que éste último lo pueda descifrar; y así con todos los demás



- Para evitar el **análisis de tráfico pasivo**, cada 10 minutos se cambian los nodos de la conexión Tor, escogiendo nuevos nodos



Protocolos criptográficos y de enrutado

– Limitaciones

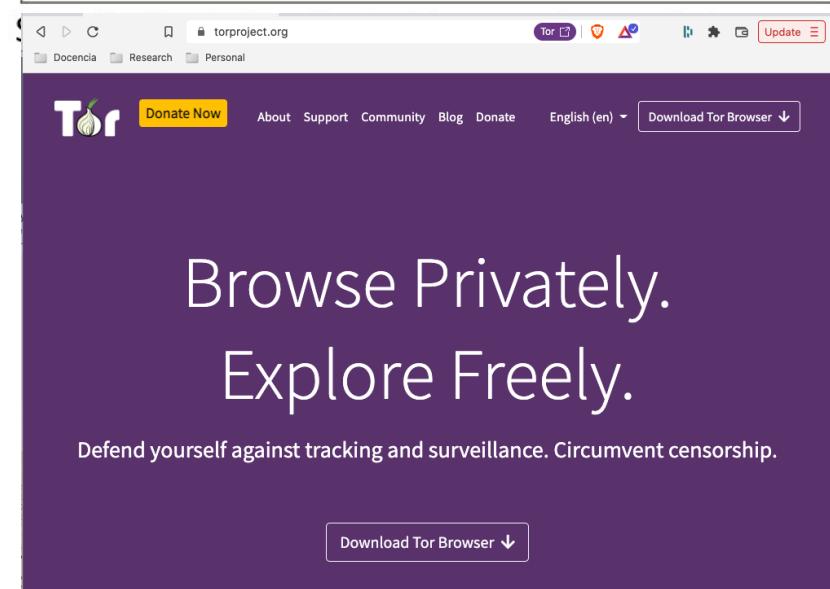
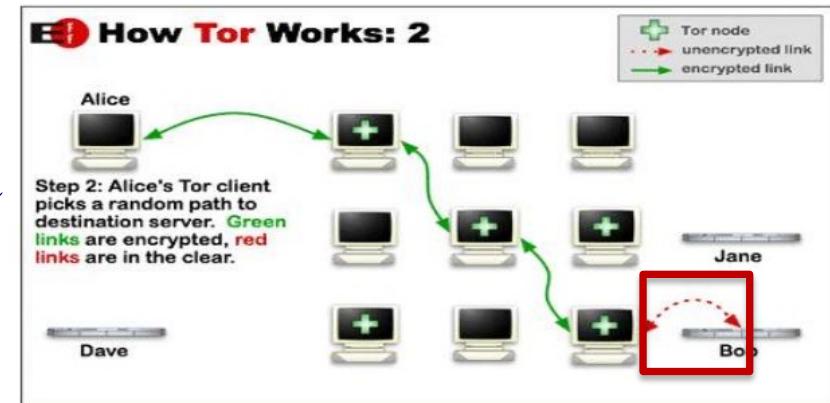
- Lento por su arquitectura:
 - conectividad basada en routers y uso de la criptografía de clave pública
- Garantiza el anonimato, pero no garantiza la “privacidad de los datos”

– Existen varias formas de acceder a la red Tor:

- Utilizar el navegador compatible con Tor
- También para móviles
- Utilizar una extensión del navegador

<https://geekytheory.com/que-es-y-como-funciona-la-red-tor>

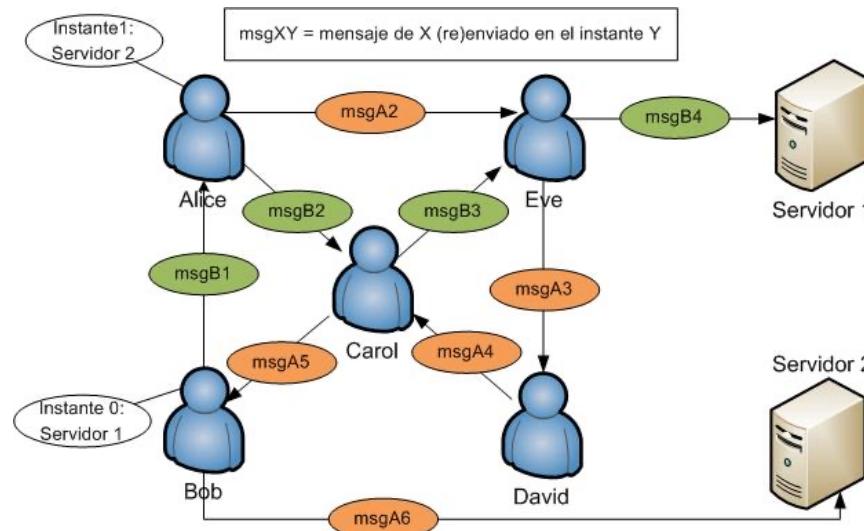
Tor in brief – 2/3



Protocolos criptográficos y de enrutado

d) Basadas en la creación de grupos

- **Crowds:** los emisores se agrupan creando una “multitud” en la que todos enrutan de manera aleatoria los paquetes recibidos de sus compañeros
 - cada nodo sólo conoce el destino y el nodo del que recibe el paquete
 - los nodos comparten claves con sus vecinos para cifrar los mensajes
 - el camino seguido por el mensaje es almacenado temporalmente en cada nodo en el que se transita para saber enrutar de vuelta (con la respuesta)



Protocolos criptográficos y de enrutado

- **Hordes**: es un esquema muy similar al de Crowds con algunas diferencias, principalmente en la forma de enrutar las respuestas de los mensajes
 - en este caso se hace un broadcast de la respuesta desde el router a todos los miembros del grupo donde se encuentra el originador del mensaje
 - esta diferencia supone una mejora significativa en términos de rendimiento (en término de tiempo) frente a Crowds
 - sin embargo, es menos robusto que Crowds ya que el mensaje de broadcast da ciertos indicios sobre la localización del usuario, y sobrecarga al sistema

Protocolos criptográficos y de enrutado

	Arquitectura	Latencia
Proxy		Baja
Mezcladores		Alta
Red de mezcladores	Centralizada	Muy alta
Enrutado de cebolla		Media-baja
Tor		
Crowds	Distribuida	Media-baja