

# Tema 3 → Mecanismos Control Acceso (Política)

## Objetivos

→ Prevenir Acceso terceros no autorizados  
→ " usuarios legítimos tocar cosas de admin  
→ Permitir " " acceder sus recursos

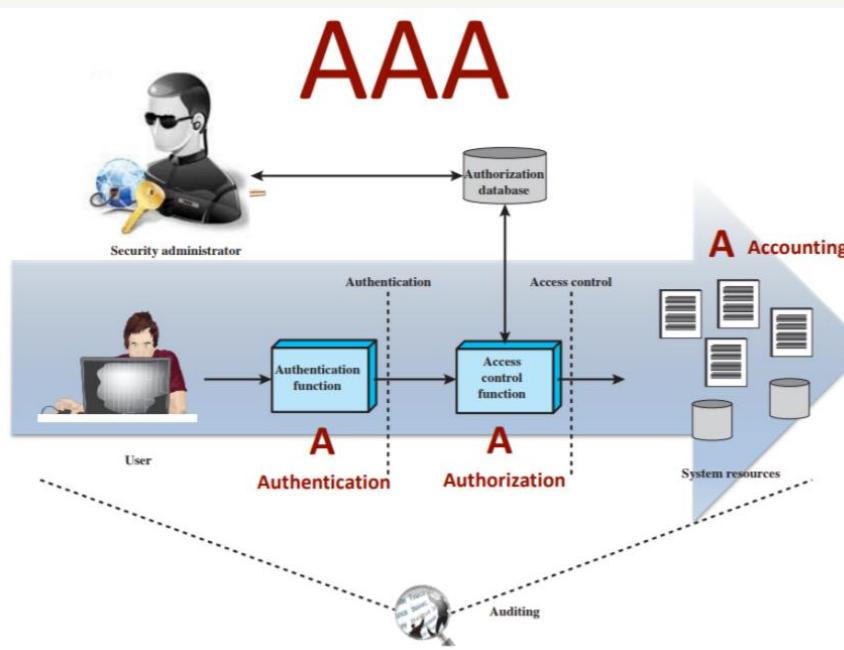
## Autenticación

## Autorización

Concesión permiso a acceder a un recurso

## Registro (Accounting)

tener registro de las actividades (detectar problemas seguridad, cambios política, ...)



## Elementos Básicos:

Sojeto: Accede a los objetos por un proceso.

Objeto: Recurso.

Derecho: Como el Sojeto Acceso accede al objeto.

## Mecanismos de Control (Cómo accede el sojeto)

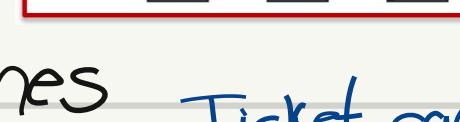
## DAC

### Matriz de Acceso

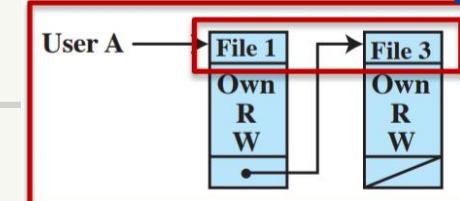
	File 1	File 2	File 3	File 4
User A	Own Read Write		Own Read Write	
User B	Read	Own Read Write	Write	Read
User C	Read Write	Read		Own Read Write

## ACL (columnas)

2 descomposiciones



Ticket capacidades (filas)



## MAC

clasifica objetos por etiquetas (MLS)

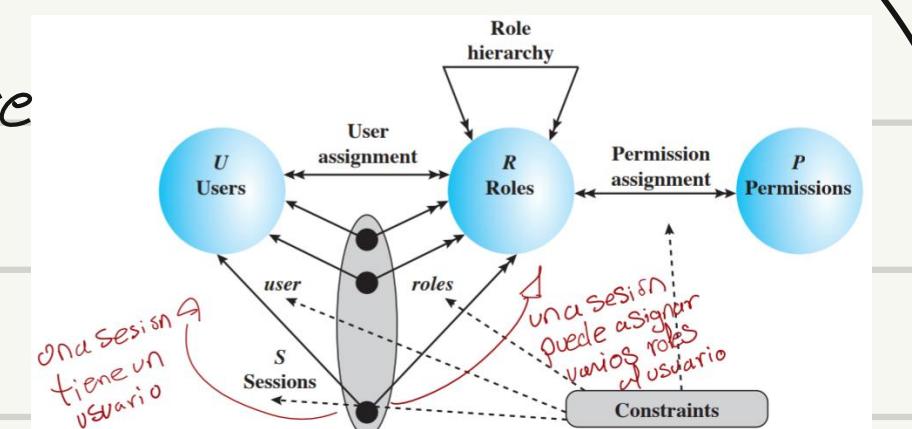
EJ: Bell-La Padula (BLP)

↳ Lee objetos de mismo nivel y abajo  
Escribe " " " " y 1 Superior

## RBAC

asignación roles

SSD: "cambiar entre varios tipos de usuarios y acceder a cosas diferentes."



DSO: Igual, pero cerrando sesión (volviendo a autenticarme).

## ABAC

Acceso basado en atributos sujeto (edad, carac. piel, bajo o alto, ...)

## ■ Protocolos Criptográficos Avanzados

Algoritmo con un objetivo

### ◦ División de Secretos

Mensaje  $M$  se divide en trozos

- Crea el nonce  $R$  del tamaño de  $M$  (creado por Trent).
- Aplico "XOR" a  $R$  con  $M \rightarrow$  Genero  $S$
- Trent distribuye  $R$  a Alice y a Bob.

"one time pad"  
→  $\rightarrow$  PDA otra persona  
texto cifrado  
a 1 persona

### PROBLEMA

Si una parte se pierde,  $M$  no se recupera.

↓ ↓ Solución??

### ◦ Compartición de Secretos

Mensaje  $M$  dividido en " $n$ " trozos.

- Se elige los grados de un polinomio de grado  $K-1$

EJ:  $(3, 5) \rightarrow$  (Sombras, n.º de usuarios) /  $D(\text{Mensaje secreto}) = \underline{\underline{11}}$   
 $\quad\quad\quad$  (Trazos de  $M$ )  $\quad\quad\quad$   $a_0$

$$q(x) = a_0 + a_1 x + a_2 x^2$$

$$\rightarrow EJ: a_1 = 2 / a_2 = 1$$

$$\rightarrow q(x) = 11 + 2x + x^2$$

- Asignamos trozos a cada usuario, reemplazando " $x$ "

$$q(1) = 14 \rightarrow Alice$$

$$q(2) = 19 \rightarrow Bob$$

$$q(3) = 26 \rightarrow Carol$$

$$q(4) = 35 \rightarrow Dave$$

$$q(5) = 46 \rightarrow Eve$$

- Con que 3 de ellos juntan sus ecuaciones y sustituyan por sus " $x$ ", quedarán un sistema de ecuaciones.

→ Deducimos  $\underline{\underline{a_0 = D}}$

Varios Usuarios!!! 😊

Si tengo  $n$  usuarios,  
genero  $n-1$  cadenas

# ◦ Bit Commitment (predicciones - apuestas)

2 maneras:

## ◦ Crip. Simétrica

- ① Bob genera aleatoriamente una cadena  $R$  de bits y se la envía a Alice  
 $Bob \rightarrow Alice: R$
- ② Alice crea un mensaje con el bit  $b$  (la apuesta), y la cadena aleatoria de Bob. Lo cifra con una clave aleatoria  $K$ , y envía el resultado a Bob  
 $Alice \rightarrow Bob: E_K(R, b)$
- ③ Alice envía a Bob la clave  $K$   
 $Alice \rightarrow Bob: K$
- ④ Bob descifra el mensaje para obtener el bit, y chequea su cadena aleatoria para verificar la validez del bit  
 $Bob: D_K(E_K(R, b))$

## ◦ Funciones Hash

- ① Alice genera aleatoriamente dos cadenas de bits  $R_1$  y  $R_2$  y crea un mensaje que consta de las dos cadenas aleatorias y el bit de compromiso  
 $Alice: (R_1, R_2, b)$
- ② A continuación, computa la función hash de ese mensaje y envía el resultado a Bob junto a una de las cadenas aleatorias  
 $Alice \rightarrow Bob: H(R_1, R_2, b), R_1$
- ③ Alice envía a Bob el mensaje original  
 $Alice \rightarrow Bob: R_1, R_2, b$
- ④ Bob computa la función hash del mensaje y compara ésta y  $R_1$  con el valor y la cadena aleatoria recibida en el paso 3. Si coinciden el bit es válido  
 $Bob: H(R_1, R_2, b)$

## ◦ Lanzamiento de Monedas (cara o cruz)

2 soluciones

### ◦ Bit Commitment

### ◦ Crip. Clave Pública

- ① Alice y Bob generan, cada uno, un par <clave pública, clave privada>. Las claves públicas NO se comparten.  
 $Alice: K_{Apu}, K_{Apr}$   
 $Bob: K_{Bpu}, K_{Bpr}$
- ② Alice genera dos mensajes, uno indicando la “cara” y otro la “cruz”. Estos mensajes están además basados de alguna cadena aleatoria, para verificar posteriormente su autenticidad (ej.  $M1=“1011110”$  y  $M2=“1110111”$ ).  
 $Alice$  cifra ambos mensajes con su clave pública y los envía a  $Bob$  en un orden aleatorio  
 $Alice \rightarrow Bob: E_{Apu}(M_1), E_{Apu}(M_2)$
- ③ Bob, que no puede leer los mensajes, elige uno, lo cifra con su clave pública y se lo devuelve a Alice  
 $Bob \rightarrow Alice: E_{Bpu}(E_{Apu}(M))$  donde  $M$  es o bien  $M_1$  o bien  $M_2$
- ④ Alice, que no puede leer el mensaje que Bob le ha devuelto, lo descifra con su clave privada y se lo devuelve a Bob  
 $Alice: D_{Apu}(E_{Bpu}(E_{Apu}(M))) \leftarrow D_{K1}(E_{K2}(E_{K1}(M))) = E_{K2}(M)$   
 $Alice \rightarrow Bob: E_{Bpu}(M)$
- ⑤ Bob descifra el mensaje con su clave privada para averiguar el lanzamiento de la moneda, y envía el mensaje descifrado a Alice  
 $Bob: D_{Bpr}(E_{Bpu}(M))$   
 $Bob \rightarrow Alice: M$
- ⑥ Alice lee el resultado del lanzamiento y verifica que la cadena aleatoria es correcta tal como ella lo había producido:  $M1=“1011110”/M2=“1110111”$

## ◦ Poker Mental

- ① Alice y Bob generan, cada uno, un par clave pública/clave privada  
 $Alice: K_{Apu}, K_{Apr}$   
 $Bob: K_{Bpu}, K_{Bpr}$
- ② Alice genera 52 mensajes, uno para cada carta de la baraja. Estos mensajes están además basados de alguna cadena aleatoria, para verificar posteriormente su autenticidad.  
 $Alice$  cifra todos los mensajes con su clave pública y los envía a  $Bob$  en un orden aleatorio  
 $Alice \rightarrow Bob: E_{Apu}(M_i)$  donde  $i = 1 .. 52$

- ③ Bob, que no puede leer ninguno de los mensajes, elige aleatoriamente cinco de ellos; los cifra con su clave pública y se los devuelve a Alice  
 $Bob: E_{Bpu}(E_{Apu}(M_j^B))$   
donde  $M_j^B (j = 1..5)$  son las cinco cartas que Bob ha elegido
- ④ Alice, que no puede leer los mensajes que Bob le ha devuelto, los descifra con su clave privada y se los devuelve a Bob  
 $Alice: D_{Apu}(E_{Bpu}(E_{Apu}(M_j^B)))$   
 $Alice \rightarrow Bob: E_{Bpu}(M_j^B)$
- ⑤ Bob descifra los mensajes con su clave privada para averiguar su mano  
 $Bob: D_{Bpr}(E_{Bpu}(M_j^B)) = M_j^B$
- ⑥ De los 47 mensajes  $E_{Apu}(M_i)$  que restan de los 52 que recibió en el paso 2, Bob elige aleatoriamente cinco de ellos,  $E_{Apu}(M_k^A)$ , y se los envía a Alice  
 $Bob \rightarrow Alice: E_{Apu}(M_k^A) (k = 1..5)$   
donde  $M_k^A (j = 1..5)$  son las cinco cartas de la mano de Alice
- ⑦ Alice descifra esos cinco mensajes y ve cuáles son las cartas que le han tocado  
 $Alice: D_{Apu}(E_{Apu}(M_k^A)) = M_k^A$

# Tema 4 → Seguridad y Privacidad en App. Web

## Red Team

Actividades Ofensivas (buscar vulnerabilidades)

Kali Linux

CERT/CSIRT

Analizan ciberincidentes de seguridad y solucionan

Criticó "APT" Stuxnet (central nuclear)

Muy Alto Malware, robo

Medio Ataque Dos

Bajo Spam

↓  
Ataque grandes equipos

Aircrack-ng

Wireshark

John (the Ripper)

Nmap

Mitre

Repositorio con distintos ataques

NVD(Nist)

API para ataque a fu código

## Blue Team

Actividades Defensivas (defender S.O.)

Seguridad E-Mail

Objetivo: Autenticación y Confidencialidad (capa Aplicación)

PGP (uso personal)

(uso comercial) S-MIME

Integrar varios algoritmos

Igual que "PGP" pero firma

RSA] Autenticidad

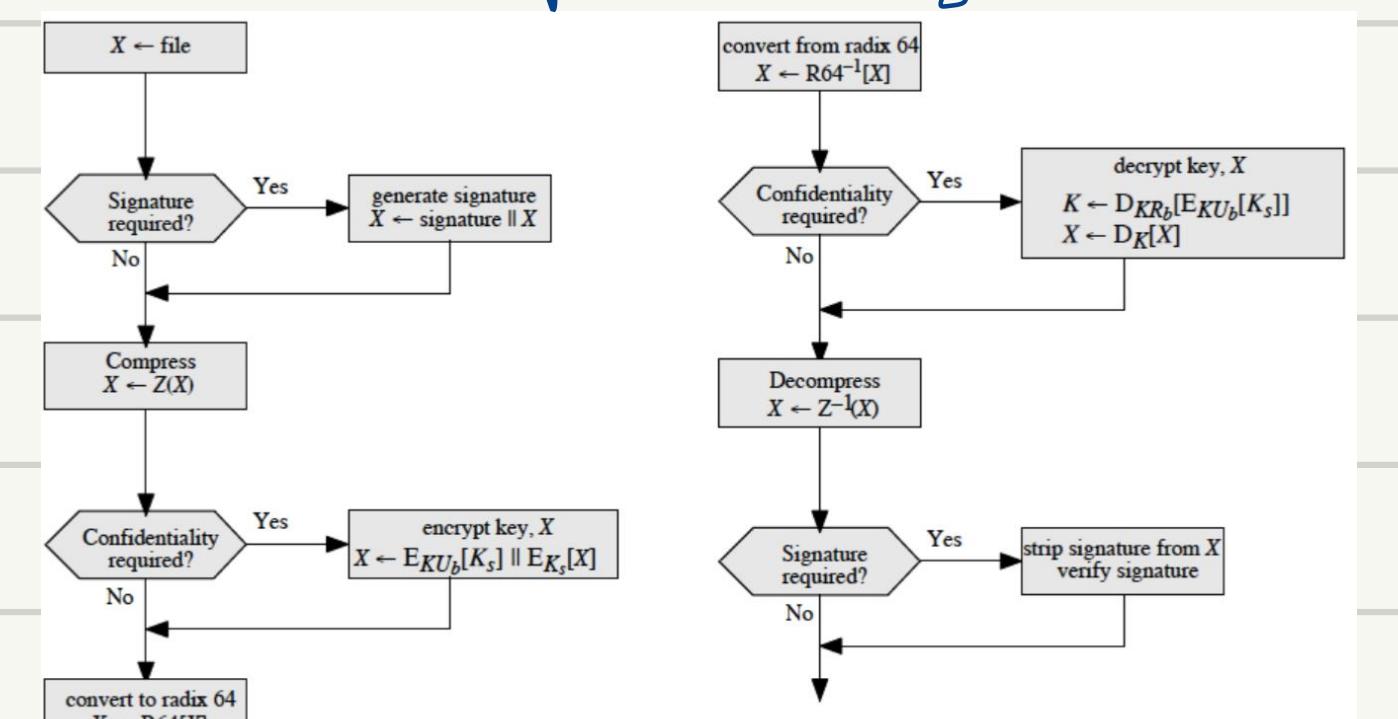
DH } Confidencialidad

DSS } Integridad

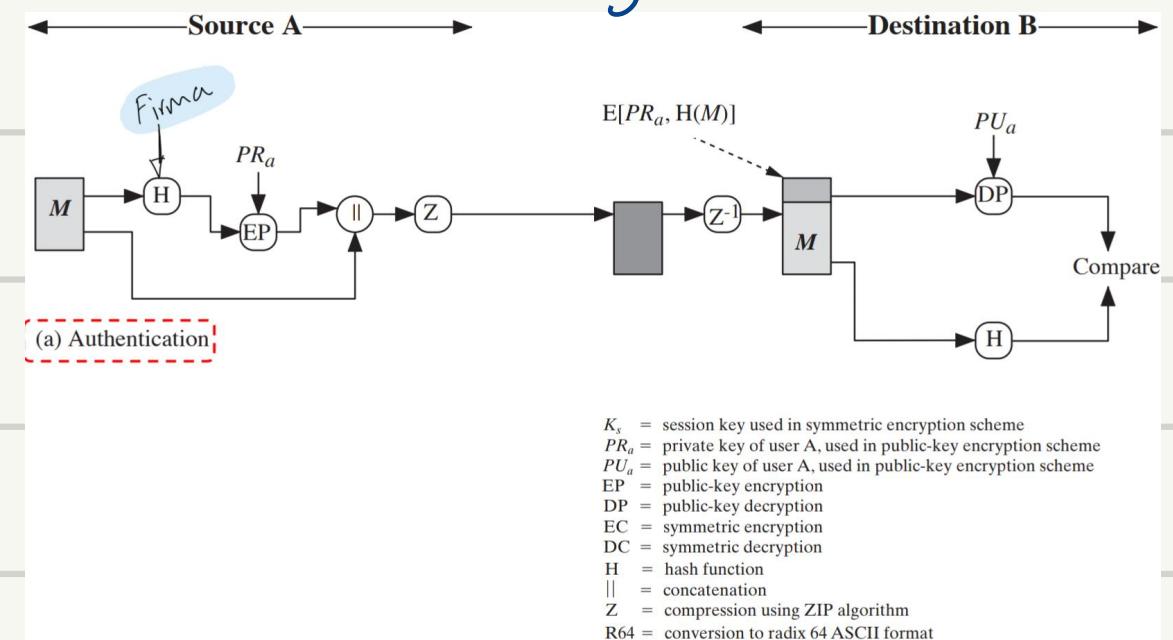
Uso en APP. E-Mail (incluye compresión)

Almacenamiento Ficheros

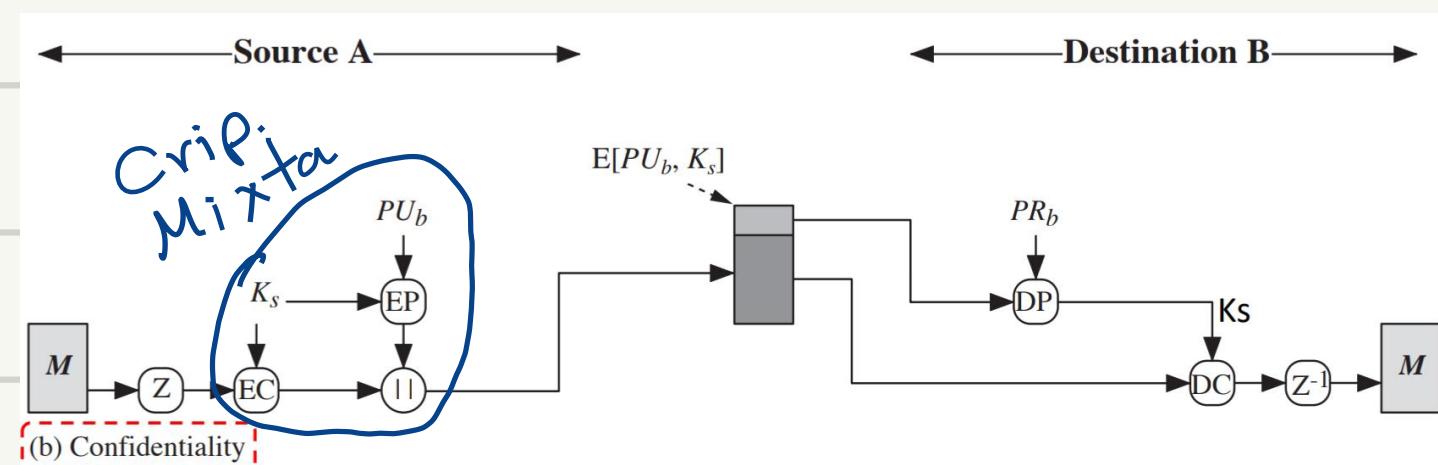
Trans. Recip. mensajes "PGP"



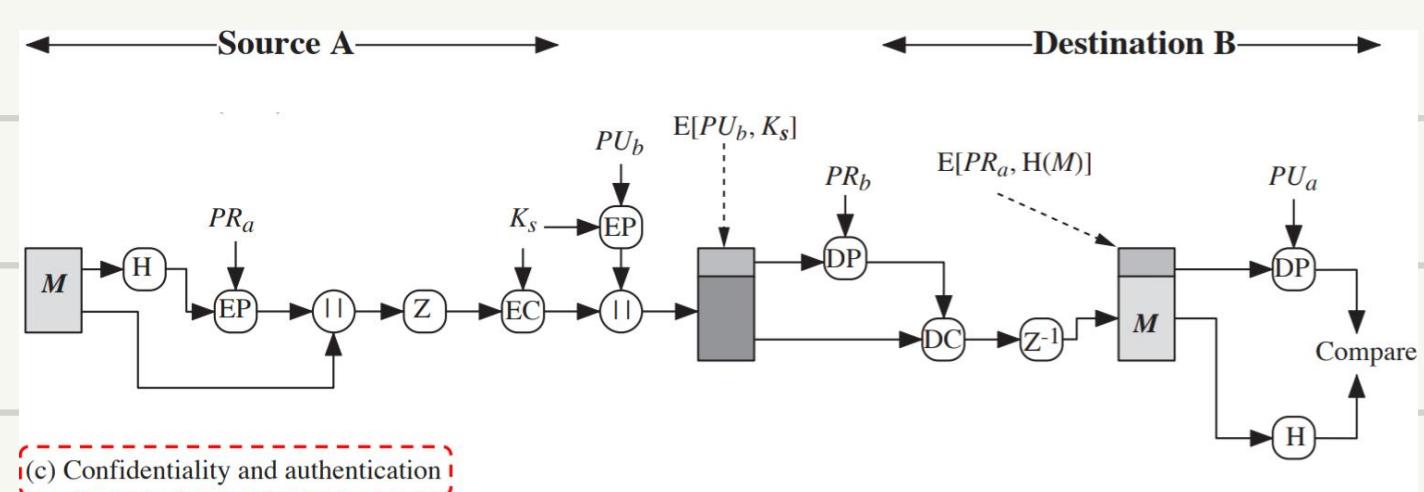
## Autenticación



## Confidencialidad



This is the Remix Brrr..



## Conexión Remota Segura

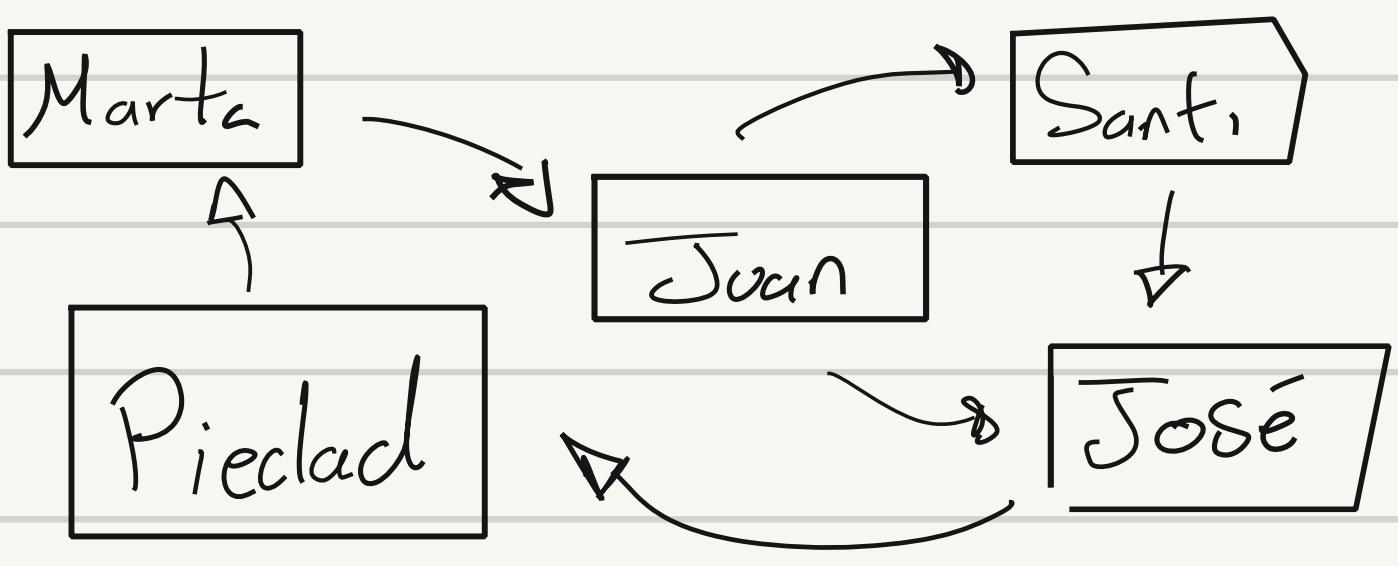
### Peligrosos: Siempre Cerrados

- Telnet 23: Control remoto por TCP
- FTP 20-21: Transferencia de ficheros

Información trans. en plano

## Modelo "PKI" en Malla

- Cada usuario emite certificados de claves públicas a otros usuarios



Flags (nivel de confianza): 1-10  
 menor → mayor confianza

### Seguro: SSH - Secure Shell (v.2)

- SSH 22: Se cifra con criptografía pública.
- Algoritmos: Cifrado: AES

Integridad: MAC

Autenticación: RSA, DSA

Inter. Claves: DH

{ PTTY: Windows  
 OpenSSH: Unix (Linux, Mac)

Aut Cliente  
 Int. claves / Negociación Modo cifrado  
 Conexión cliente-Servidor

Aplicación

Transporte

Red → Modo túnel (P2P) cifrado

[Esteganografía: Ocultar info haciendo uso de archivos multimedia]

Se puede usar para transferir ficheros Seguros

◦ SFTP (FTP sobre SSH) → todo cifrado a nivel de red (tunel)

Modo Básico: Usuario/contraseña

Modo Avanzado: Claves Públicas SSH (transmitidas previamente)

▫ Cuando se quiere enviar datos, se vuelven a intercambiar las claves → Autenticación

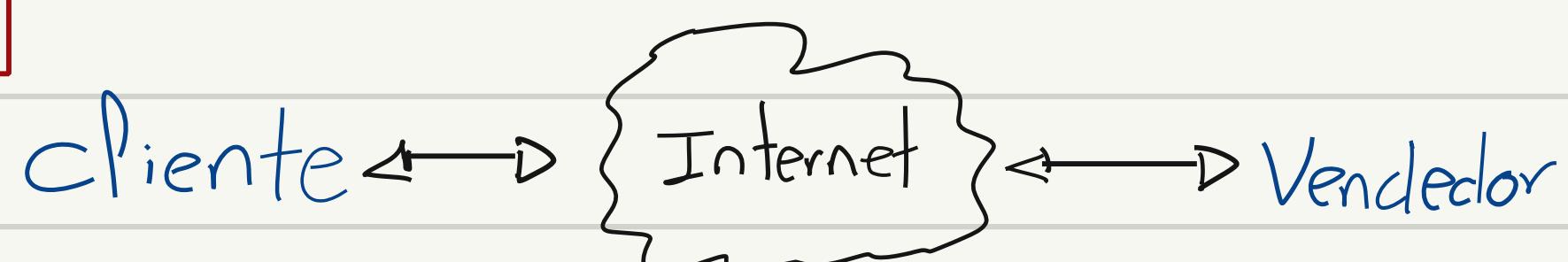
◦ FTPS (FTP sobre TLS) → Credenciales cifradas durante el proceso (tema 5)

EJ: BitLocker, Sealed

## • Seguridad en Pagos Electrónicos

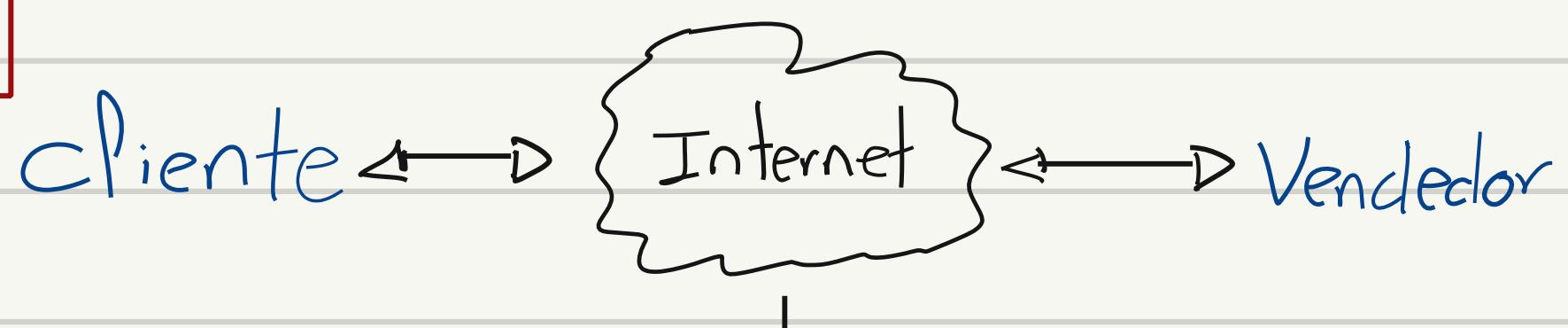
Objetivo: Misma funcionalidades que comprando en física

Antes



No intermediario → No Repudio info en plano

Ahora



Se controlan las transacciones

TTY → No-Repudio

Se cobra una comisión de uso

Gateway (con proxy)

Protocolo "Set"

Usado actual pero sobre TLS

## ◦ Formas de Pago

◦ Vendedor contacta con banco.

Online: Pago en el teléfono con tarjeta al contacto

Offline: Recargo tarjeta después de comprar (Amazon)

↳ Se deja un tiempo que la entidad financiera corrobore.

◦ Comprador procede con transacción

Sistema Pre-Pago: tarjetas prepago

Sistema Pago Instantáneo: tarjetas débito

Sistema Post-Pago: tarjetas crédito

↳ cierto tiempo después se decrementa la cantidad

◦ Cantidad de Dinero

Micropagos (- - 6 euros, España)

Coste elevado de implementación

Vendedor paga demasiada comisión por uso del gateway

Solución: Broker

Cupones "Script"

Comprar bonos (ej: 10 euros) y canjearlo en una cuenta para micropagos en distintas tiendas web.

Pagos: (6-100 euros)

Macropagos: (100- - euros)

◦ Primeros Protocolos → SSL

Uso de Cript. híbrida (RSA, DH, DES, RC4)

Autent. e integridad (MAC)

Problemas

- Vendedor recibe info del cliente en plano (tarjetas)
- No gateway → No bancos → No verificación de pagos



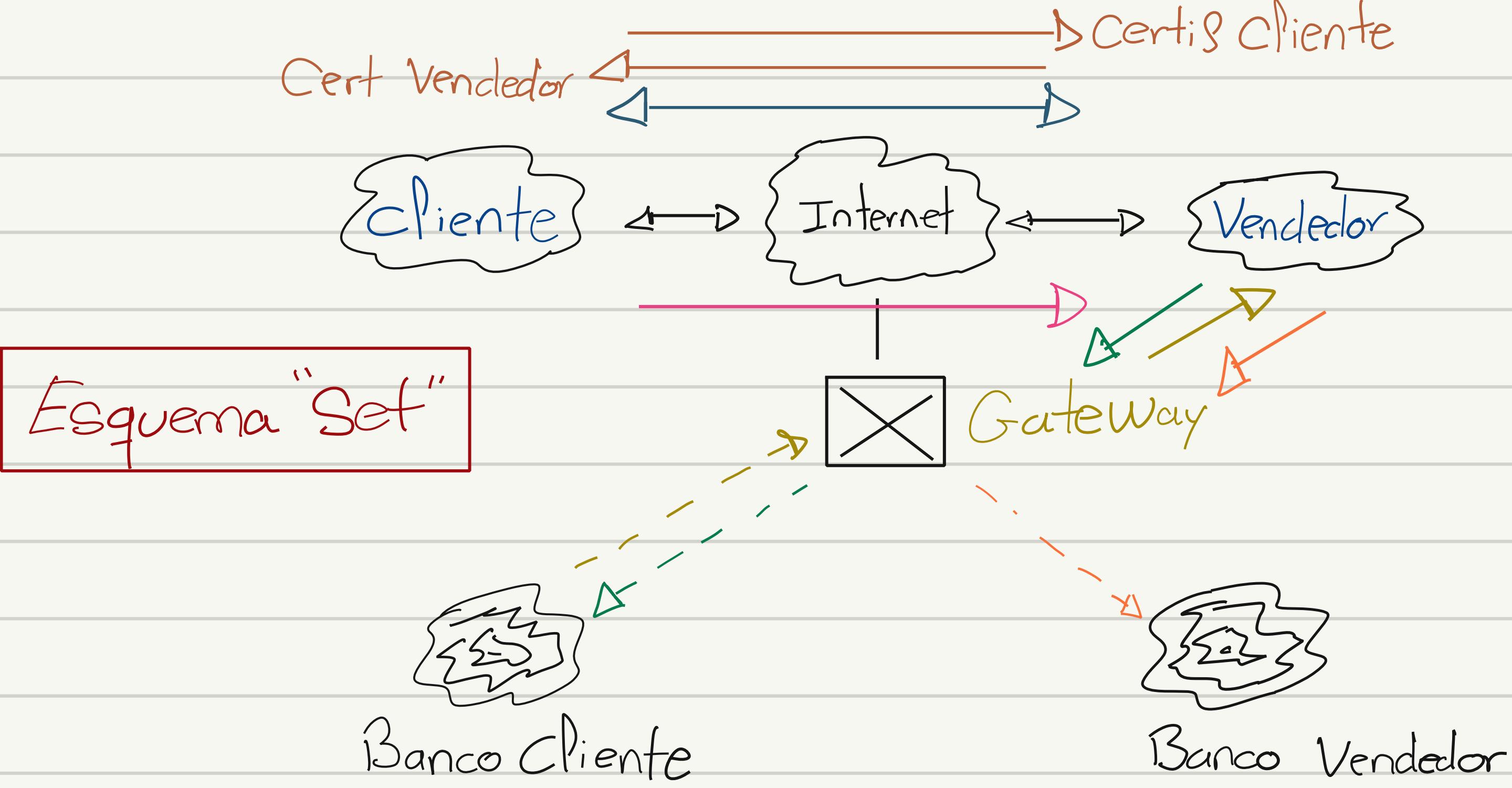
No recibos → No No-Reputio



Estafas

Surge Protocolo Set (conjunto protocolos seguridad)

Confidencialidad, Autenticación (cert. X.509), Privacidad (Firma dual),  
Integridad y no-reputio (Firma digitales), autorización.



1º Petición Producto

2º Envío de certificados (autentificación)

3º Pedido y Pago

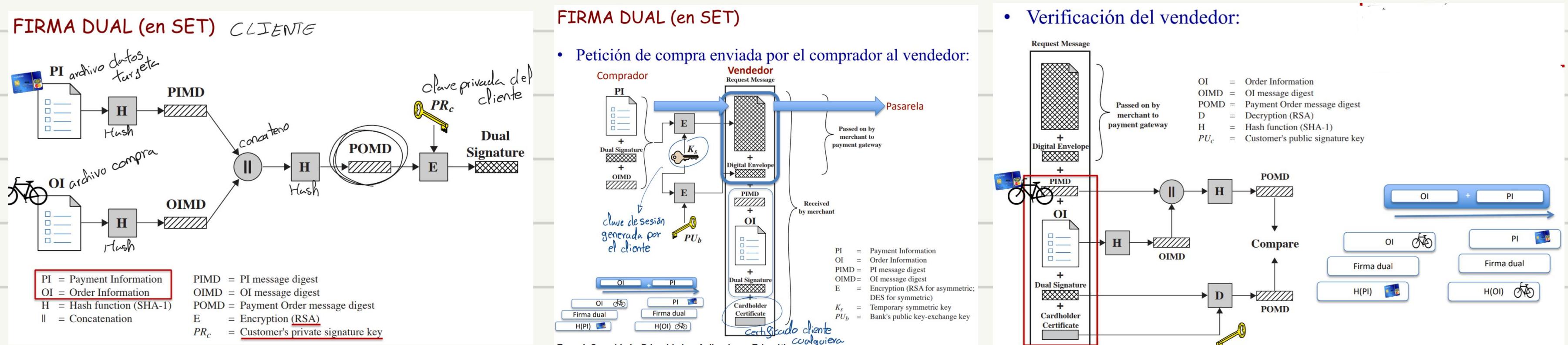
4º Autorización del vendedor a sacarle dinero al banco cliente

5º Aprobación

6º Vendedor reclama cantidad a Gateway

• Firma Dual → Privacidad

- Ni el comerciante ni vendedor necesitan conocer los datos de la tarjeta PI
- Ni el banco necesita saber lo que compró OI



Confidencialidad, autenticación, integridad, no-repudio, privacidad

Desventajas: Depende del protocolo → Si se descubre falla, inseguro

Dificultad con certificados

No para micropagos → Costosos

## • Privacidad de los usuarios en aplicaciones

Privacidad: Mantener la integridad de la persona  
 Confidencialidad: Mantener datos en secreto

Proteger la Privacidad =

- Leyes con Sanciones GDPR
- Algoritmos Criptográficos
- Atacante no puede relacionar 2 mensajes

Propiedades Privacidad =

- No puede rastrear el mensaje

Anonimato No Rastreable

Pseudónimos  
"mote"

Privacidad  
= Anonimato

Anonimato Rastreable  
2 personas se conocen pero uno puede desvelar al otro

Anonimato No Rastreable y No Vinculante

Firma Digital

## • Firma Digital

→ Firma verificada más tarde

▫ Firma a Ciega: Anonimato Rastreable

EJ: Voto Electrónico

"Mensaje  $M$  de Bob sea firmado por Alice sin saber que pone en  $M'$ "

$e$  = clave pública  
 $d$  = "privada"  
 $r$  =  $n \leq$  Aleatorio

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

▫ Firma de Grupo: Anonimato Rastreable

EJ: Bancos

"Admin. del grupo emite clave de firma de grupo → todos firman con ella, en nombre de todos."

▫ Admin. puede verificar quién firmó (nadie más sabe quién firmó → Anonimato)

$z$  = clave secreta admin.

Firma:  $S = E_{x_i}(M)$

↳ Claves privadas de cada miembro

→ Si la firma es correcta o no

Verificación:  $E_y(S) = M$

Clave pública grupo

▫ Firma Anillo: No vinculable ni Rastreable

EJ: Envío de un cheque firmado por cliente y trabajadores de la entidad bancaria.

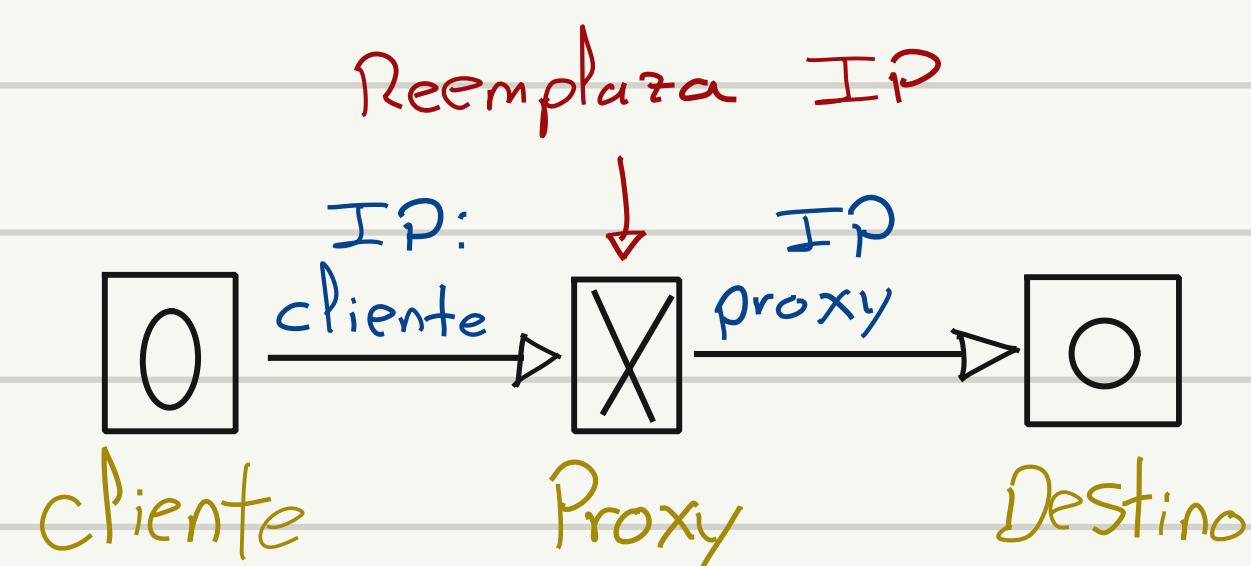
"Computa una firma con su clave privada y el resto de claves públicas".

## ◦ Protocolos Criptográficos y de Enrutado

▫ Basado en el uso de Proxy

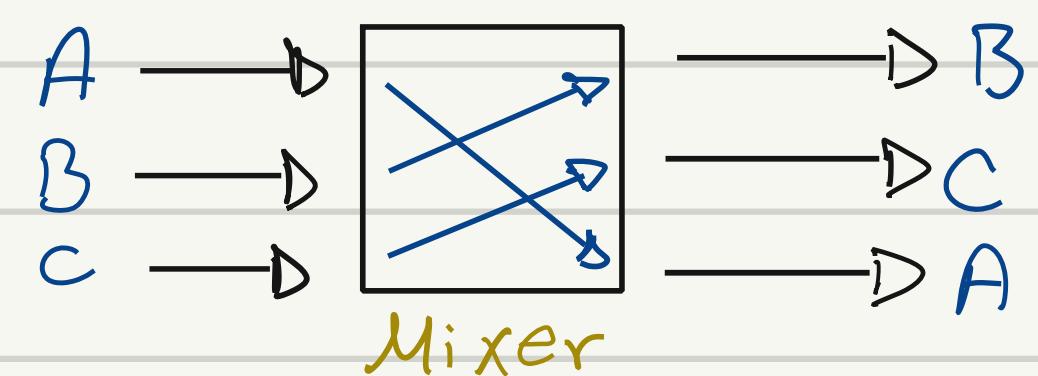
Servidor proxy hace de intermediario

- de llega un paquete con una IP y reemplaza por su propia IP antes de reenviar.



▫ Basado en uso de Mixers

Tipo de proxy → de llega un mensaje y lo almacena un tiempo. Al tiempo lo envía por un puerto diferente

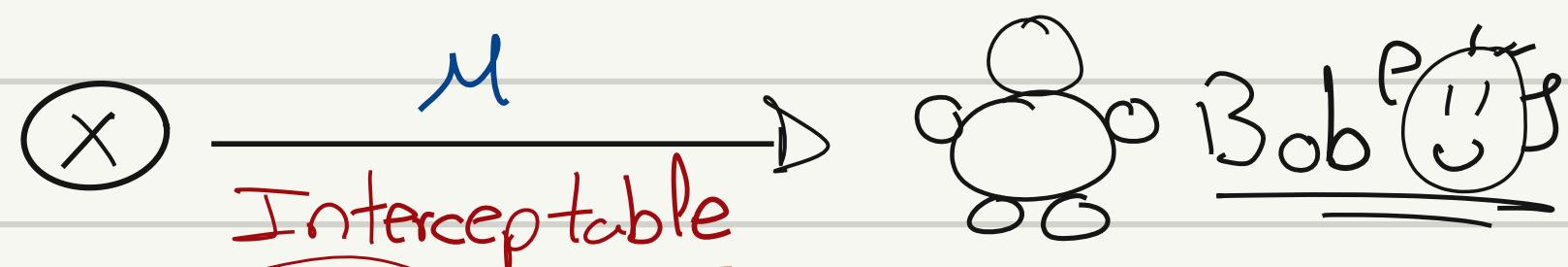


▫ Basado en el conocimiento parcial de la ruta

Paquete cifrado en varias capas → Se "pelean" rutas llegar al mensaje cada vez que pasa por un router.

Vulnerabilidad → El último router deja el mensaje en

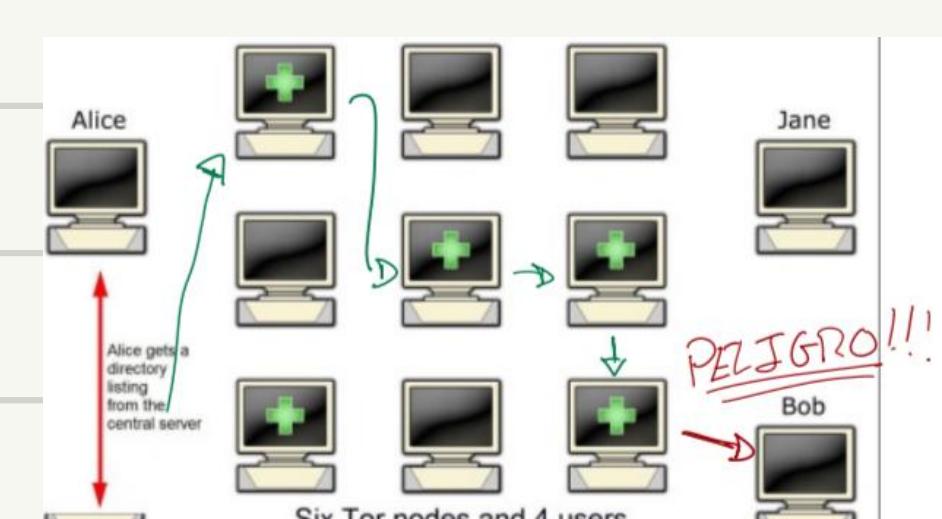
Surge TOR (v2) plano.



Red Privada: Alice quiere enviar a Bob

- Alice habla con Proxy → establece ruta de encaminamiento
  - ↳ Cada 10 minutos cambia

↳ Paquete dentro de la red: Seguro (encriptado)  
Fuera " " " : Peligro (texto plano)



▫ Basada en la creación de grupos

Crowds: Muchos emisores → Cada uno solo conoce el nodo destino y de donde viene el paquete.

Hordes: Igual al anterior, pero el reenvío hace broadcast a los vecinos para enviar.

Más rápido, muy costoso (Sobrecarga sistema)

## Tema 5

### Seguridad en la Capa de Transporte

Surge WTS (Web Transaction Security)

HTTP → Capa Aplicación

SSL → Capa Transporte

↳ Trama HTTP pasa por

un filtro de SSL antes de salir y enviarlo por un puerto TCP

Proporciona: Autenticación, Confidencialidad, Integridad.

HTTPS: 443 "HTTP sobre SSL/TLS"



Existen 2 fases en SSL: Sesión SSL y Conexión SSL

Establecer una conexión segura

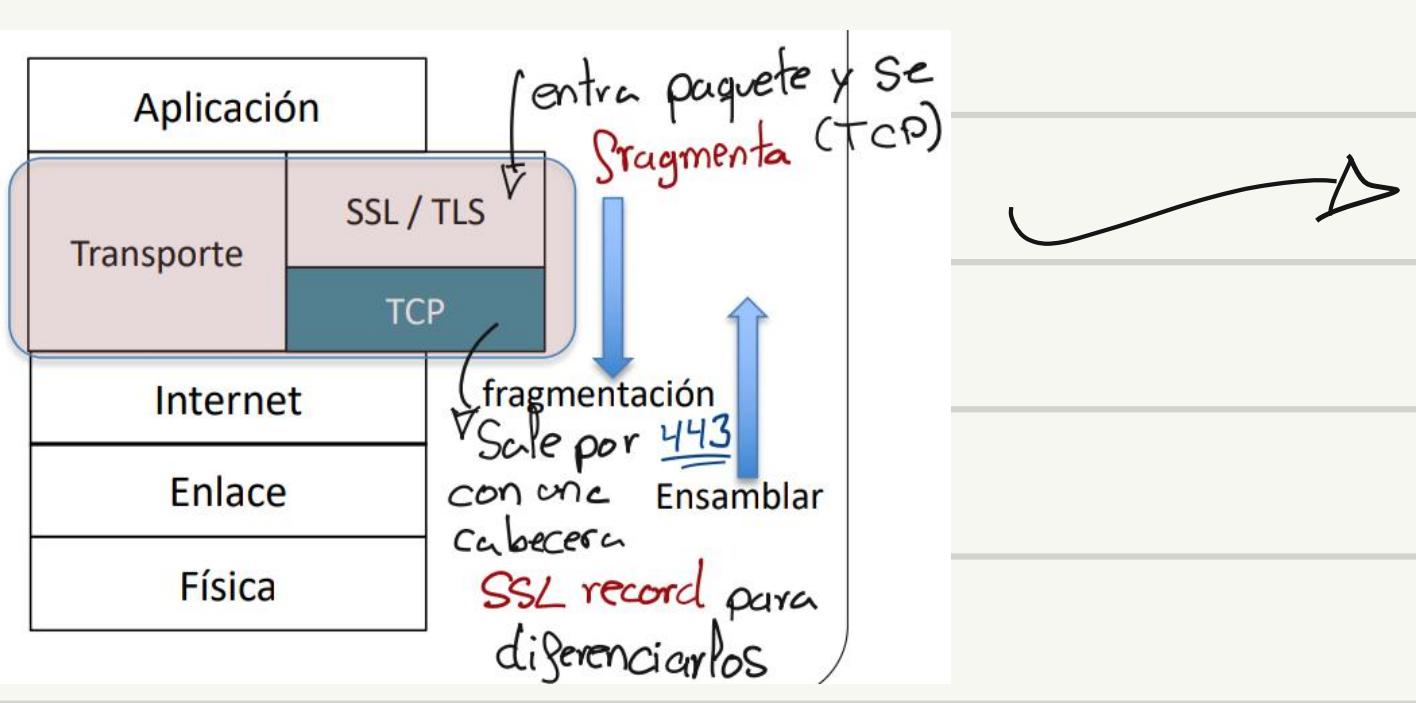
Transmitir por esa conexión

todos esos paquetes se fragmentan

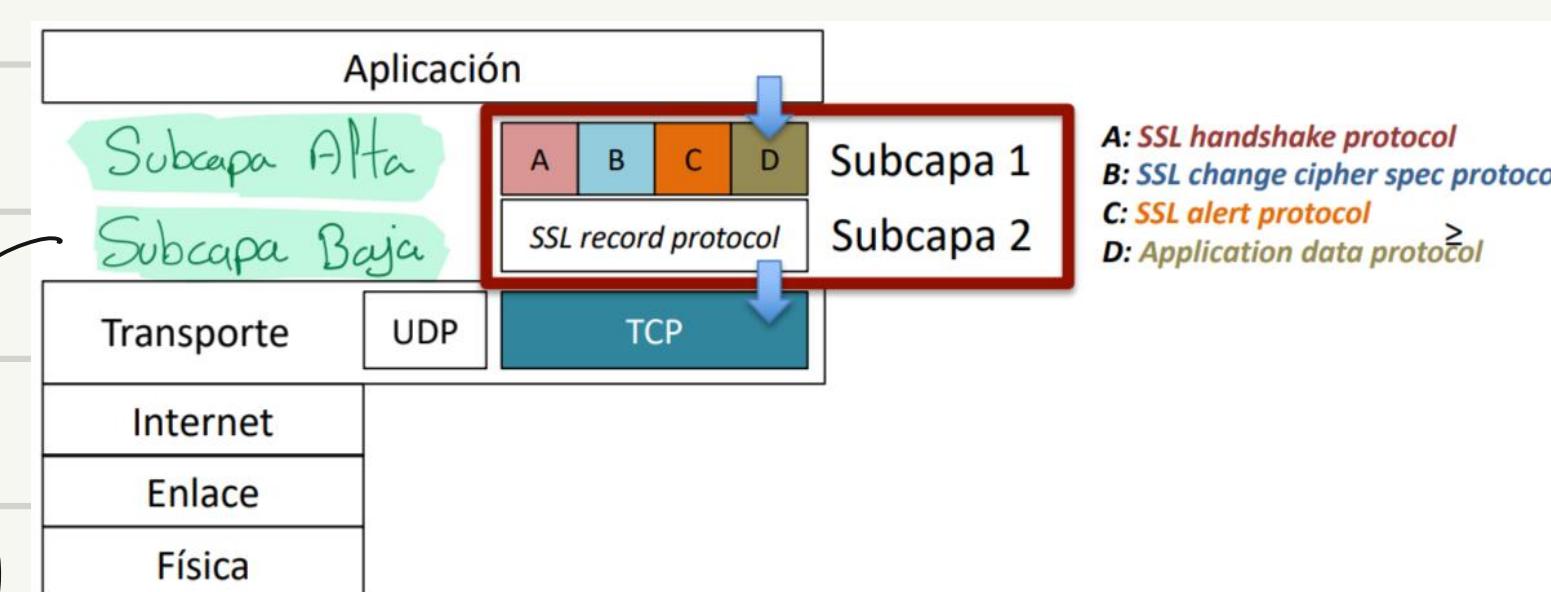
Dado que luego se envían por TCP

Fragmentación

Ensamblado



Para llevar a cabo las 2 fases, se divide en 2 Subcapas



Fragmenta los datos de la capa de aplicación y los procesa de forma individual

La subcapa alta contiene:

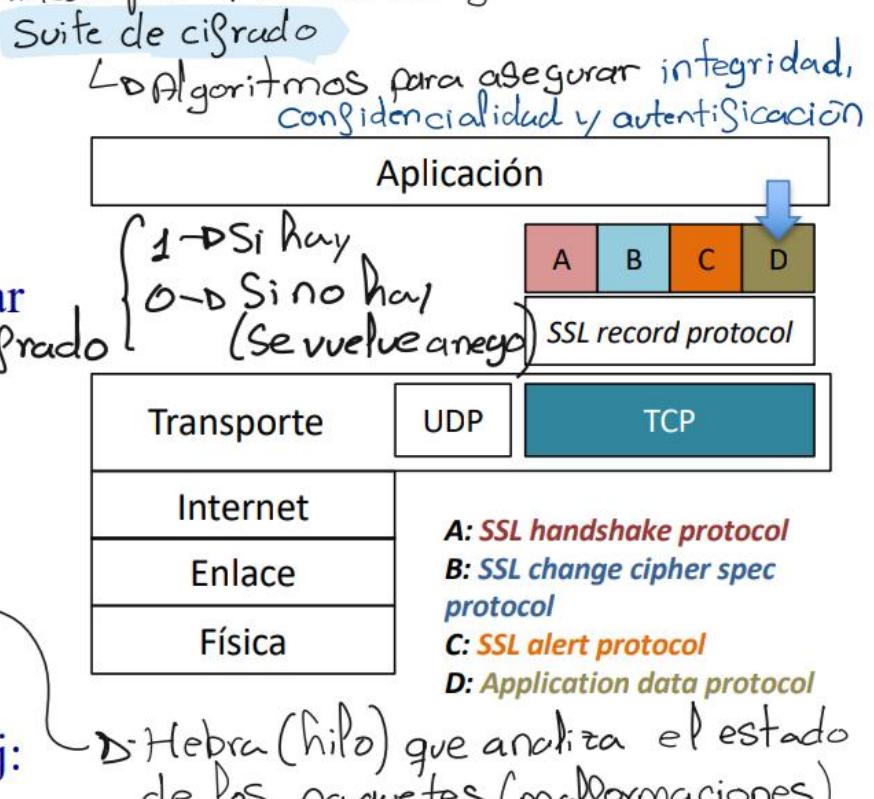
- (22) **SSL Handshake Protocol**: permite que los puntos de comunicación:  
- Se autentican ambas partes, sin ser obligatorio.  
- Se negocia la suite de cifrado.

- (20) **SSL Change Cipher Spec Protocol**: permite a los puntos de comunicación activar el cipher suite Negociar algoritmos de cifrado.

- (21) **SSL Alert Protocol**: permite a los puntos de comunicación indicar posibles problemas potenciales e intercambiar los correspondientes mensajes de alerta.

- (23) **SSL Application Data Protocol**: es el propio protocolo de la capa de aplicación (ej: HTTP) y alimenta al SSL Record Protocol

- Como un gateway, lo pasa todo de la capa de aplicación a la capa de red.



Si ServerKeyExchange

[ $K_s = C$ lave Sesión]

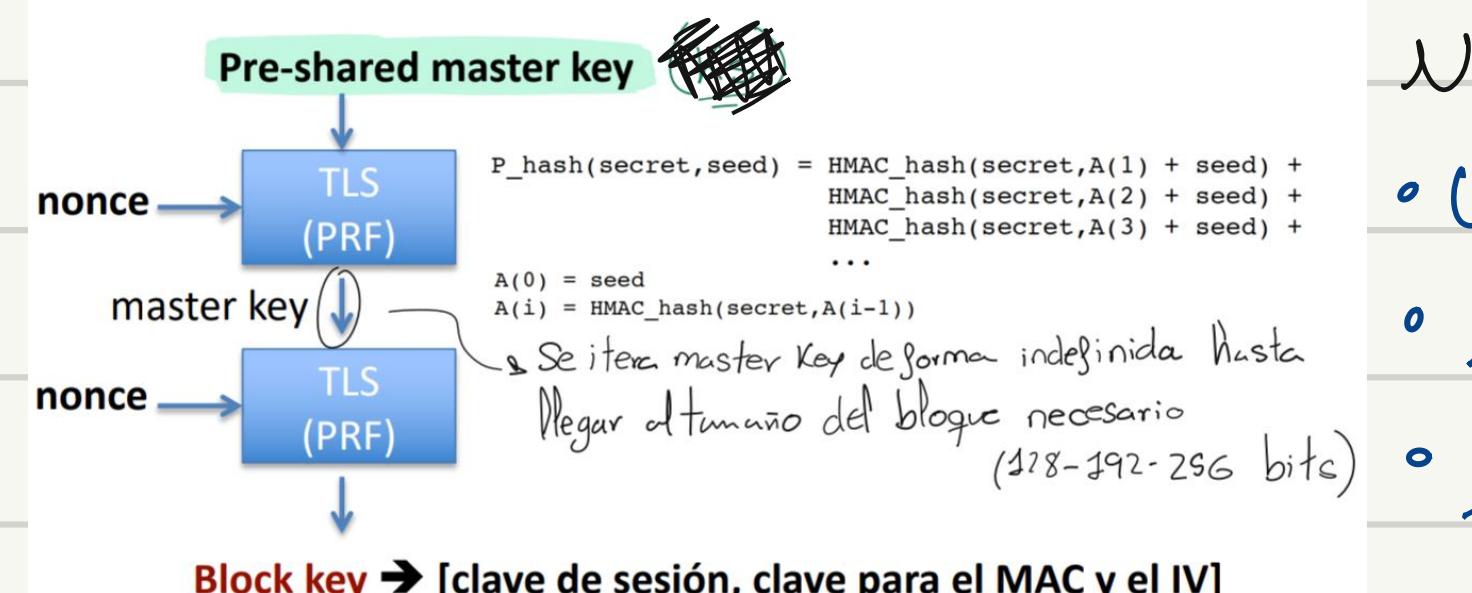
Hay → Se produce con los nonces y las claves públicas y se envía con las claves públicas cifradas.

No Hay → Se genera  $K_s$  con los nonces del cliente y de servidor.

Tema 5: Seguridad en Redes TCP/IP Primera trama que se cifra → Ya están todos de acuerdo con el Suite de cifrado.

## • Intercambio de Claves

Necesario: Clave Sesión  $K_s$  / Clave MAC / Vector IV



Necesitamos: Se generan con una Salt

- Una Semilla "pre-shared master Key"
- Nonce (valor aleatorio)
- Función Pseudorandom "PRF"

▷ También se pueden generar las claves por "DHE" (Diffie-Hellman esímero)

▫ "x" y "q"

no son constantes

Igual que DH pero...

↳ Evitamos Man in the Middle

Garantizamos "PFS" (Perfect Forward Secrecy)  
claves no comprometidas incluso si ya las hay

## • TLS 1.2 y TLS 1.3

TLS 1.2: ▷ Master Key con SHA-256 (no "PRF")

▷ Uso de extensiones (opcional → obligatorio en TLS 1.3)

▷ Elimina DES y IDEA → Añade AES

<https://www.>

SSL - Secure Sockets Layer

Servidor se autentica siempre, " " está autenticando?

Buscar trama con un certificado

Siempre en

[ ] → Opcional

Fase 1: se establece los parámetros de seguridad, incluyendo la versión del protocolo, la sesión de ID, el suite de cifrado, un número aleatorio y el método de compresión (opcional, e incluso, en SSLv3.0 no se especifica, pero sí en TLSv1.2, pero ya en la TLSv1.3 vuelve a ser opcional)

Fase 2: el servidor puede enviar un certificado (opcional), intercambia los parámetros necesarios para gestionar la clave secreta y puede solicitar un certificado al cliente

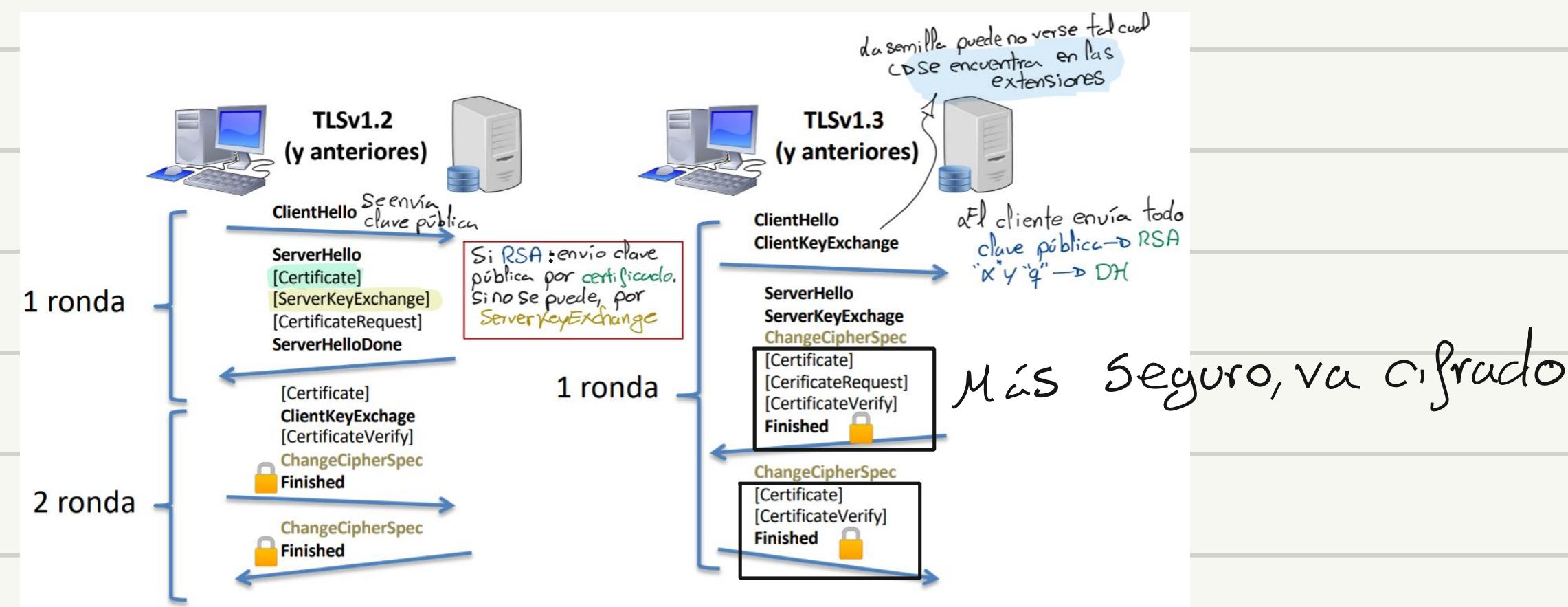
Fase 3: el cliente envía el certificado si es solicitado y los parámetros de seguridad necesarios para computar la clave de sesión

Si el cliente envía el certificado, entonces necesita enviar un certificado firmado para la verificación de la entidad de origen

Fase 4: Se establece el suite de cifrado y se termina el proceso de handshake

## TLS 1.3

Más tardío (+ milisegundos) → Eficiencia ahorrando tramas



### Nota

Existe una versión que funciona igual que TLS, pero para paquetes que usen el protocolo de transporte UDP

↳ Se le conoce como DTLS

## Seguridad en la Capa de Internet

IPSec: Especificaciones y Funciones Pidades de la capa de red para el modelo TCP/IP.

EJ: Acceso Remoto vía Internet / Apps. Comercio Electrónico

Se pretende: Autenticidad, Integridad, Confidencialidad

No proporciona Servicio de no-Reputación (ni de ataques DoS)

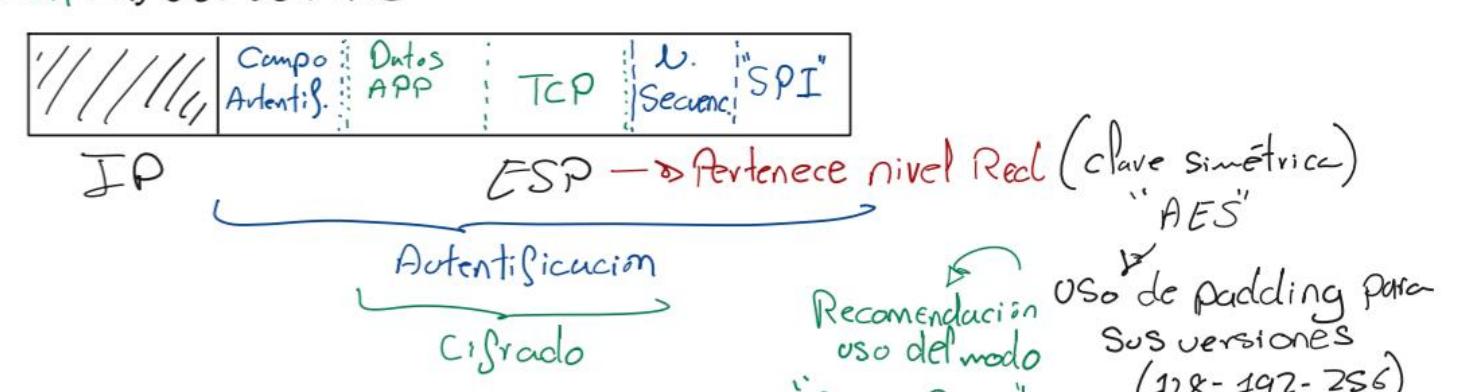
Si de repetición

## Protocolos de IPSec

"IANA": los valores asignados a IPSec ayudan a mantener la coherencia y compatibilidad entre diferentes implementaciones de IPSec en distintos sistemas y dispositivos.

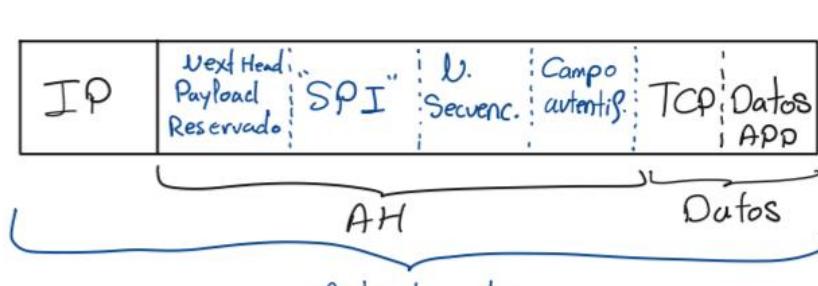
Carga de Seguridad encapsulada "ESP"

- Cifra todo el campo de datos → Confidencialidad  
↳ Uso de "Diffie-Hellman" (intercambio de claves públicas).
- Integridad y autenticación → Uso de MAC



Cabecera de autenticación "AH"

- Integridad y autenticación → Uso huella digital "HMAC"  
↳ Se calcula una función Hash al contenido del paquete IP
- NO Confidencialidad → No cifra los datos del paquete IP → info vista por terceros  
↳ Forma de evitarlo → Uso de HTTPS o FTPES con TLS



"IKE"

↳ Automatiza la generación y administración de claves

↳ Negociación de claves

• Se acuerda los tipos de cifrado y algoritmos de

Tema 5: Seguridad en Redes TCP/IP autenticación.

Pasos del "AH"

1º Emisor calcula la función Hash a partir del mensaje a transmitir.

2º Se transmiten los datos por Internet

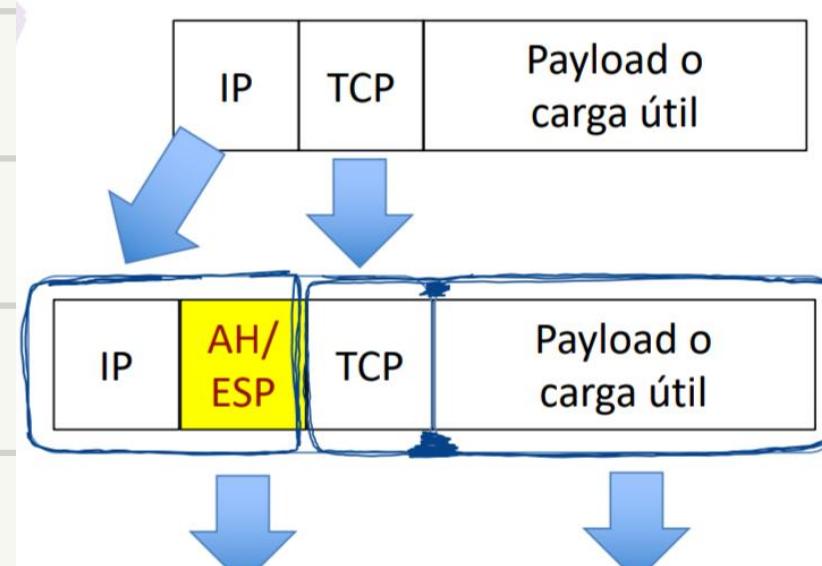
3º Paquete llega al receptor, aplica la función Hash y compara con la clave secreta que ya tenía

## Modos de IPsec

### Modo Transporte

entorno cerrado (intranet) · 2 dispositivos que no sean túneles

SOLO protege carga datagrama



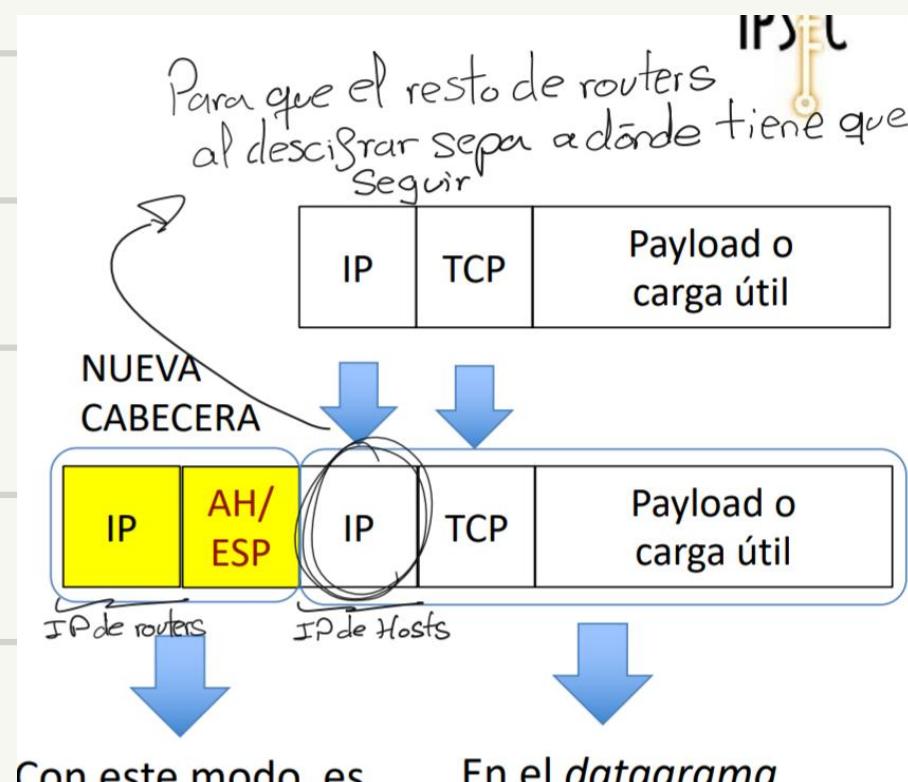
Con este modo, es visible la  
• IP origen - host  
• IP destino - host  
*Importante para saber a dónde voy*

En el payload se puede aplicar:  
• Cifrado  
• Autenticación  
• Integridad

### Modo Túnel

entre routers: Salgo de la LAN

encapsula todo datagrama IP



Con este modo, es solo visible la  
• IP origen - router  
• IP destino - router

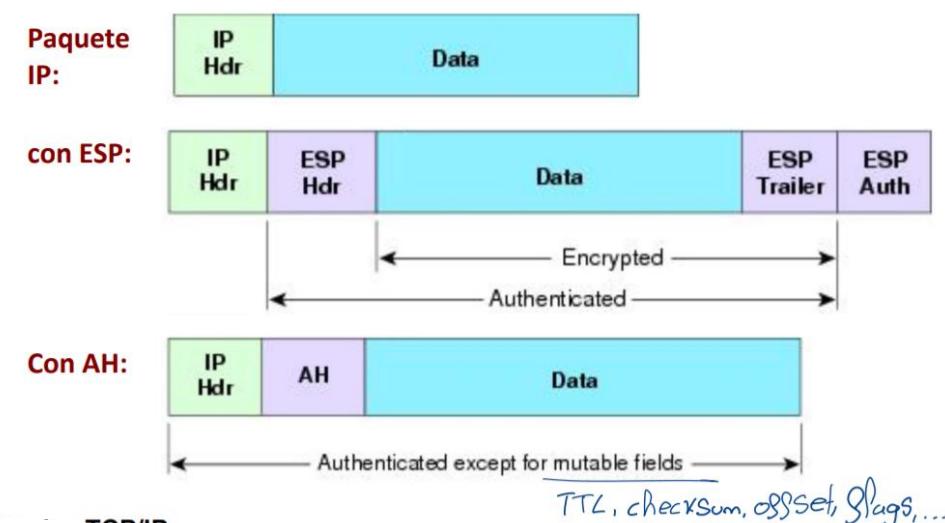
En el datagrama se puede aplicar:  
• Cifrado  
• Autenticación  
• Integridad

#### - ESP:

- se cifra el payload y
- opcionalmente lo autentica, pero no la cabecera IP

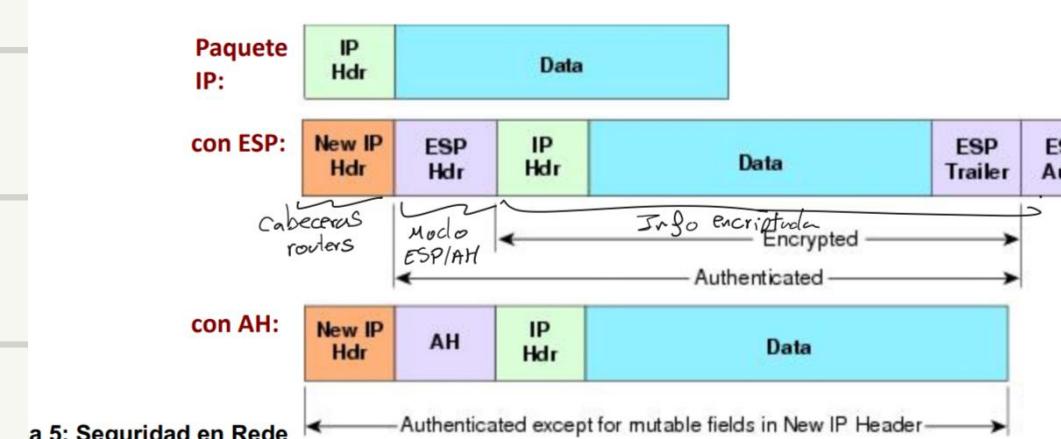
#### - AH:

- se autentica el payload y algunas porciones de la cabecera IP



Se puede aplicar:

- **ESP:**
  - se cifra y
  - opcionalmente autentica todo el paquete IP original (paquete interno), incluyendo la cabecera de ese paquete original
- **AH:**
  - se autentica todo el paquete original y algunas partes de la nueva cabecera externa



## Protocolo "IKE"

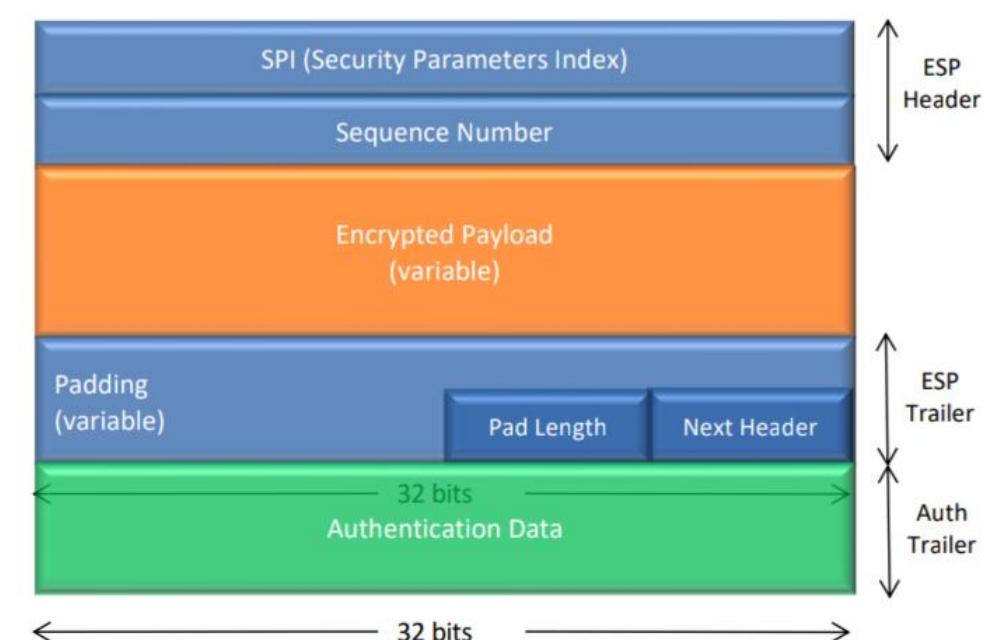
▷ Autenticación de las partes.

↳ Certificados X.509

▷ Establecimiento clave secreta  
↳ "DH"

### Encapsulating Security Payload – ESP

- cifrado + autenticación de dato+integridad:



## Uso de ISAKMP

2 fases

Autentica Cada Parte.

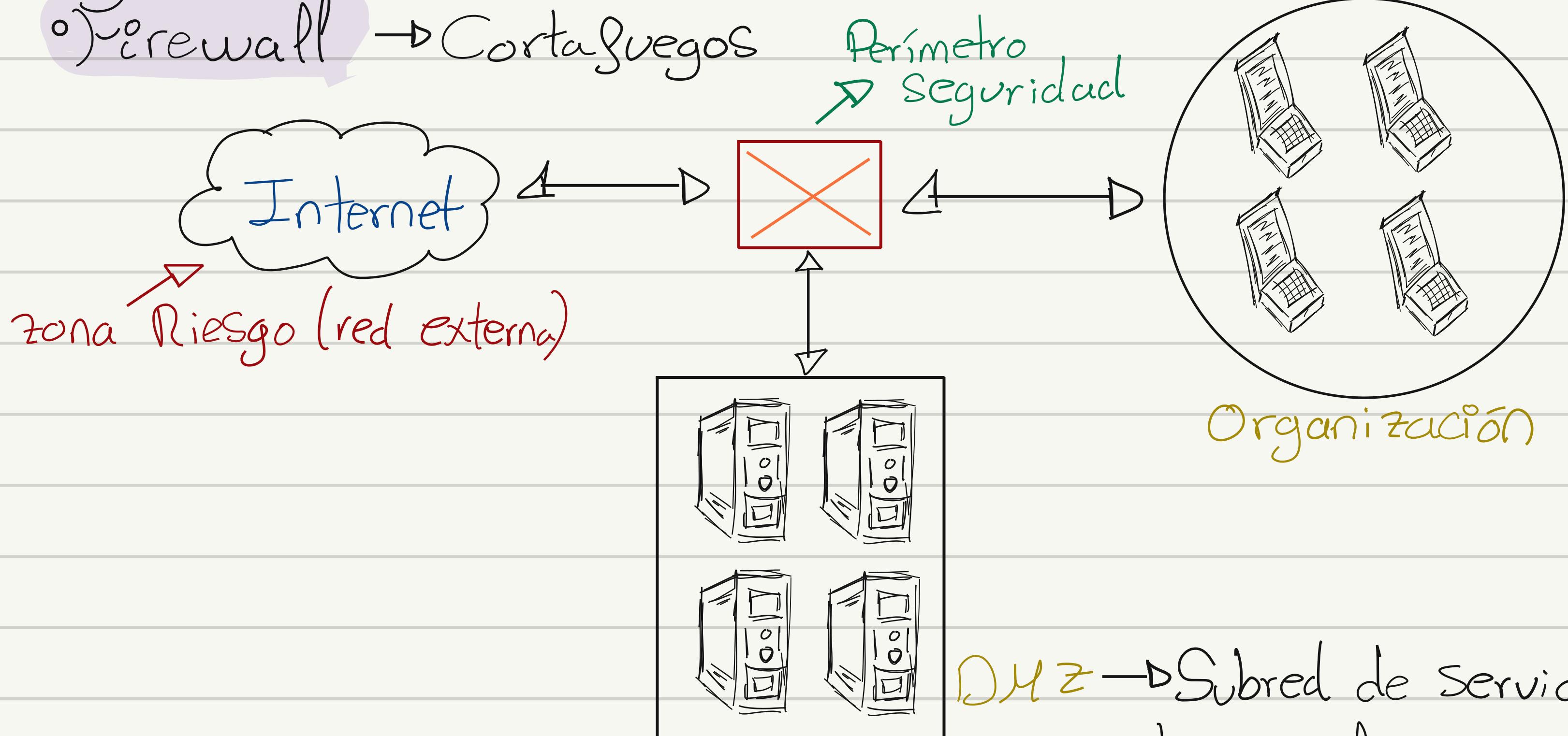
Modo Agresivo: Usa mitad de los mensajes.

No confidencialidad: ID cliente en claro

Modo Principal: Todos (Punto a punto y seguro)

Negocia y establece SAs de IPsec:

## o) Firewall → Cortafuegos



DMZ → Subred de servidores  
protege cualquier acceso a  
los host del sistema.

### 1<sup>era</sup> Generación

Revisa: IP emisor/receptor  
Protocolo, puerto, DNS

### 2<sup>da</sup> Generación

Revisa: IP emisor/receptor  
Protocolo, puerto, DNS  
Flags: SYN, ACK, ...

### Generaciones

### 4<sup>ta</sup> Generación (todo lo anterior)

+

- Control Móvil
- Validación APPs y eliminación
- Monitorización usuario y APPs

### 3<sup>ra</sup> Generación

Revisa: IP emisor/receptor  
Protocolo, puerto, DNS  
Flags: SYN, ACK, ...  
DPI: Uso firewall con  
VPN, TLS, ...

## o) Comandos

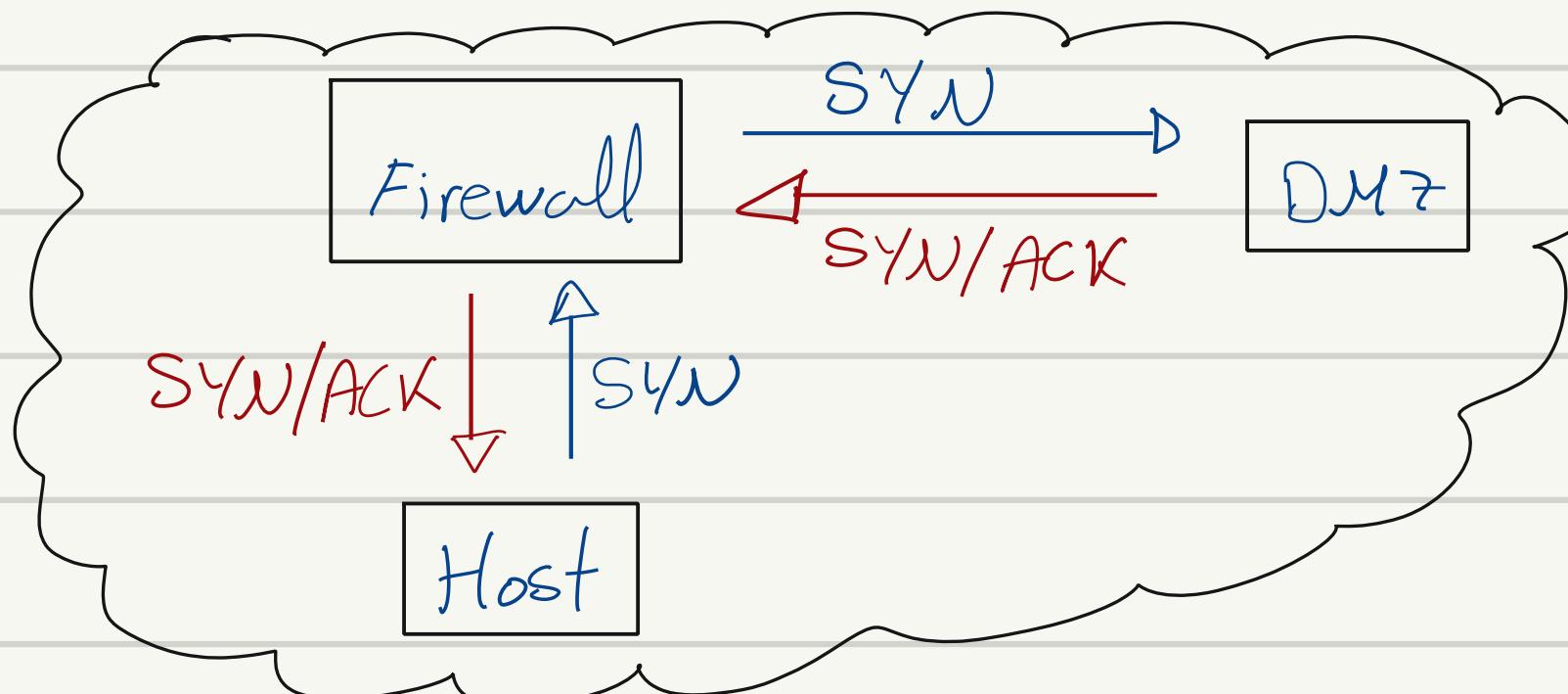
Input/Output: tráfico entrante / Saliente

Forward: Permitir (Accept) o denegar (drop) cierto tráfico por un puerto específico desde el exterior a una red interna o entre 2 LANs

Prerouting / Postrouting: Para tomar decisiones sobre un paquete antes de su enrutamiento EJ: redirigir un paquete de un puerto a otro antes de realizar enrutamiento.

El **postrouting** una vez tomada la decisión de enrutamiento, se usa para tomar acciones sobre los paquetes antes de salir del Sistema. EJ. **MASQUERADE** (traducir dirección NAT en conex. salientes).

Los **flags** (NEW, ESTABLISHED, RELATED) son importantes para comunicar la intención de establecer una conexión con otra máquina.



• **IDS** → Sistema detector de anomalías.

Pasivo "IDS": Detecta anomalía → Manda señal a una estación de un intruso. → **NO ACTUA**

Activo: "IPS": Detecta anomalía → Actua → **Elimina**

#### ► Tipos

**NIDS**: Buscan evidencias en paquetes

**DPI**: "Patrones" = Repeticiones (DoS, cambios variables, uso excesivo CPU, ...)

**Desventajas**: tráfico cifrado = nada que hacer  
Falsos positivos

**HIDS**: Supervisa Hosts (abuso de GPU, memoria, ...)

#### ► Técnicas de detección

- **Firma o abusos**: Patrones predefinidos (baja tasa falso positivos)
- **Anomalías**: "Machine learning" (alta tasa falso positivos)
- **Híbridas**: Ambas

## o Redes Inalámbricas

transm. Inalámbricas

Copia/modif mensajes

Inserción y **Dos (DoS)**

Ocultación / cifrado

Autenticación, nonce

Medidas Seguridad

Puntos Acceso(AP)

Acceso No Autorizado



Autenticación  
(Usuario/Contraseña)

Elem. Interconexión

Cifrado

desactivar broadcast

↳ Solo dejar ciertos equipos (por su MAC)

Cambiar password

## o Protocolos

Surge **WEP** (algoritmo RC4)

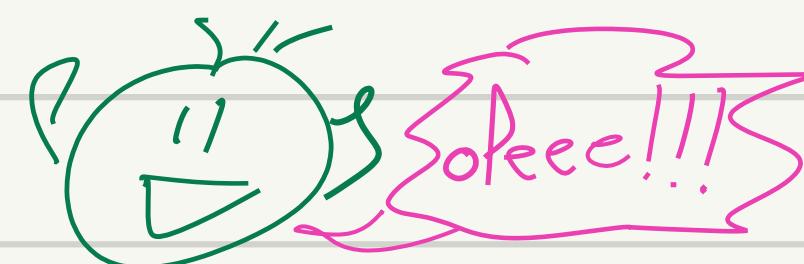
↳ Vulnerabilidad (24 bits vector inicialización IV)

↓ (con varios paquetes, se recupera clave Secreta)

Surge **WPA** (RC4)

◦ Igual que WEP, pero vector de 48 bits (vulnerable)

Con **WPA 2/3** → Usamos RSA



¡mínimo!!

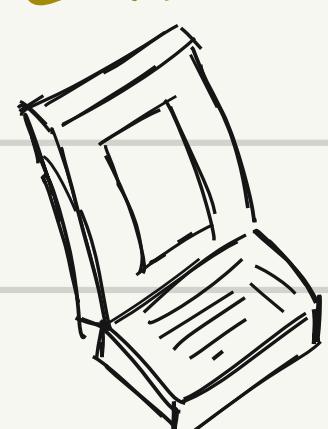
Autenticación: A través de un AP → Protocolo EAP

MAC/MIC usando HMAC (SHA-256)

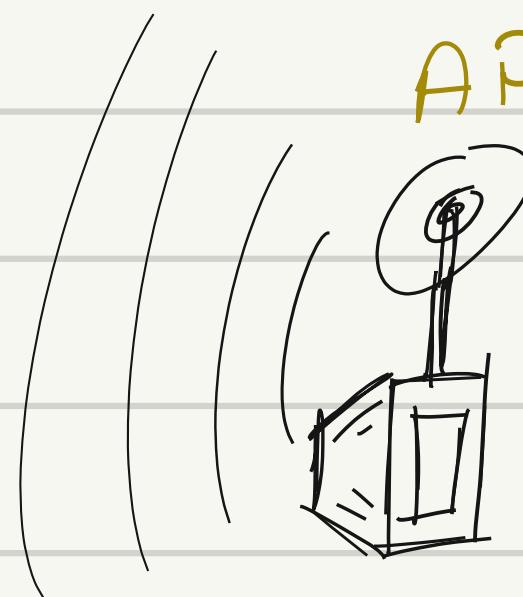
Ofrece Control Acceso: Basado en puertos (se bloquean hasta que se autentique).

Confidencialidad: Datos Se cifran

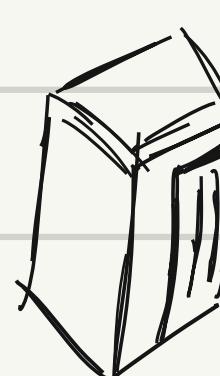
STA



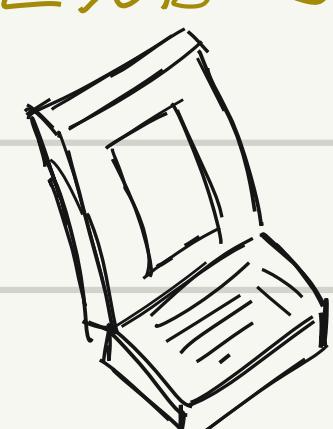
AP



AS



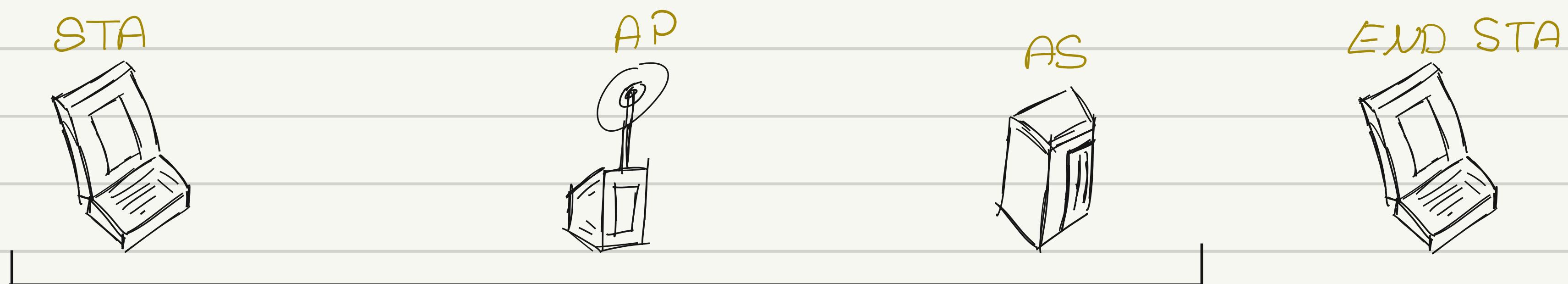
END STA



Fase 1: AP emite ondas beacon y probe responses para su política de seguridad.

↳ Se selecciona "Suite Cifrado" y mecanismo autenticación

Fase 2: STA y AS verifican la identidad AP bloquea tráfico hasta que la autenticación no acabe



Fase 3: AS y STA generan claves (Se guardan en AP y STA)

Comunicación Segura → Entre AP y STA

STA → AP → END STA

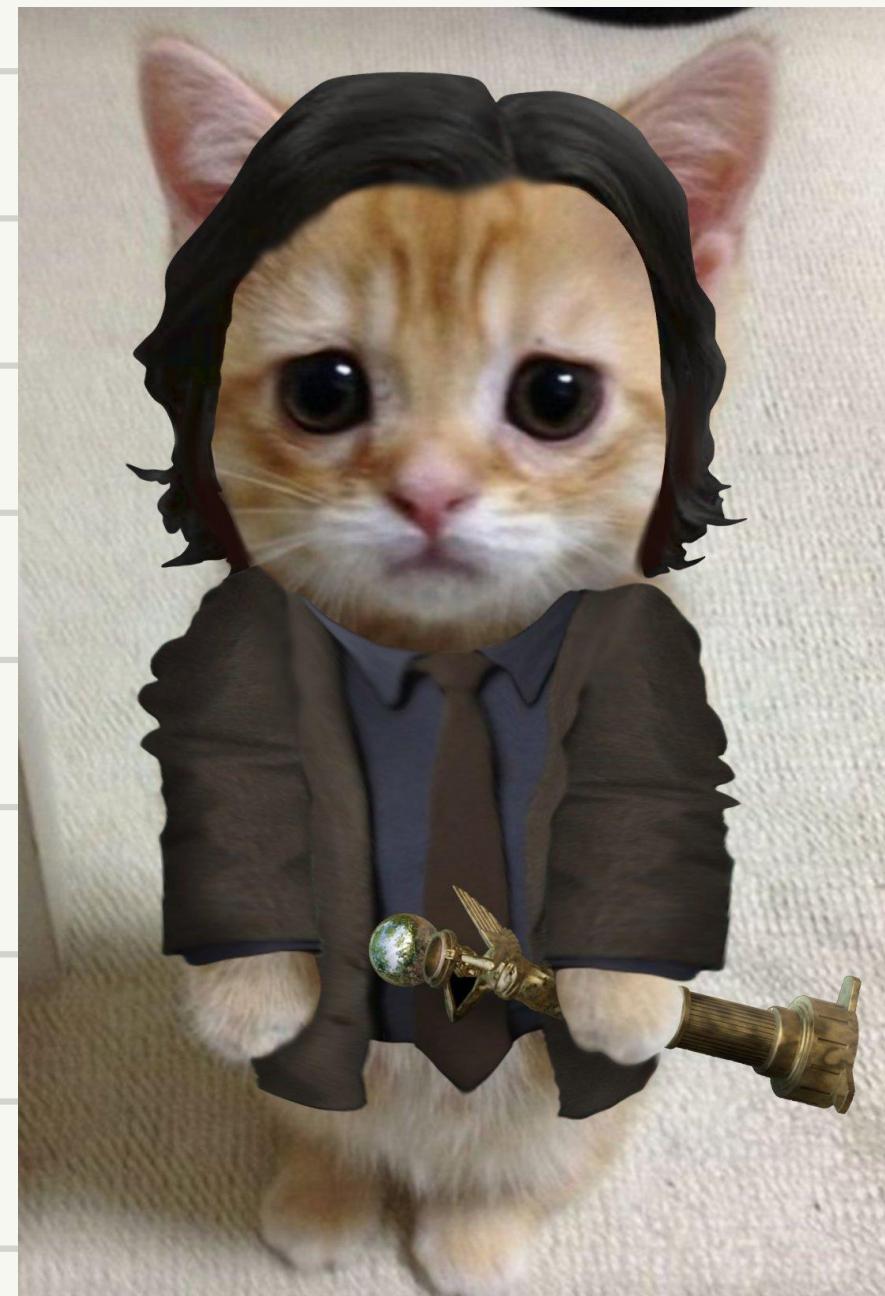
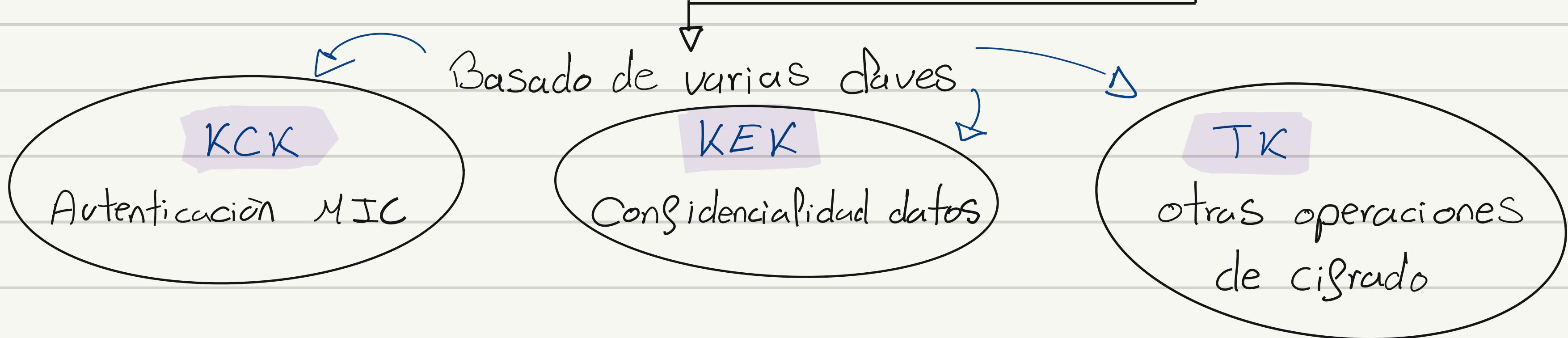
Fase 4: transferencia Segura SOLO pasando STA/END STA por AP

STA → AP → END STA

Fase 5: Finaliza Conexión

### Generación de Claves

PMK → Se genera una clave temporal → PTK



Alejandro WAKE