

SEGURIDAD EN PAGOS ELECTRÓNICOS



Conceptos generales

- En el ámbito del e-commerce se han desarrollado esquemas de pago electrónico que proporcionan en el mundo digital la misma heterogeneidad 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 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



- En general, los pagos electrónicos involucran a un **comprador** y a **un vendedor**
 - Adicionalmente, existen sistemas que involucran a TTPs
- Más aún, 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



- 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
 - Cuando el comprador procede con la transacción y carga de dinero en la cuenta del vendedor
 - La cantidad de dinero implicada en cada transacción



- 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
 - este mecanismo es el más usado hoy en día en sistemas basados en **tarjeta de crédito**
 - **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





- 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 Alice realiza la compra, el Banco asegura al vendedor que se le hará efectiva la cantidad acordada
 - pero Alice sólo verá decrementada su cuenta cierto tiempo después de haberse realizado la compra



- 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

- Ejemplos de protocolos:

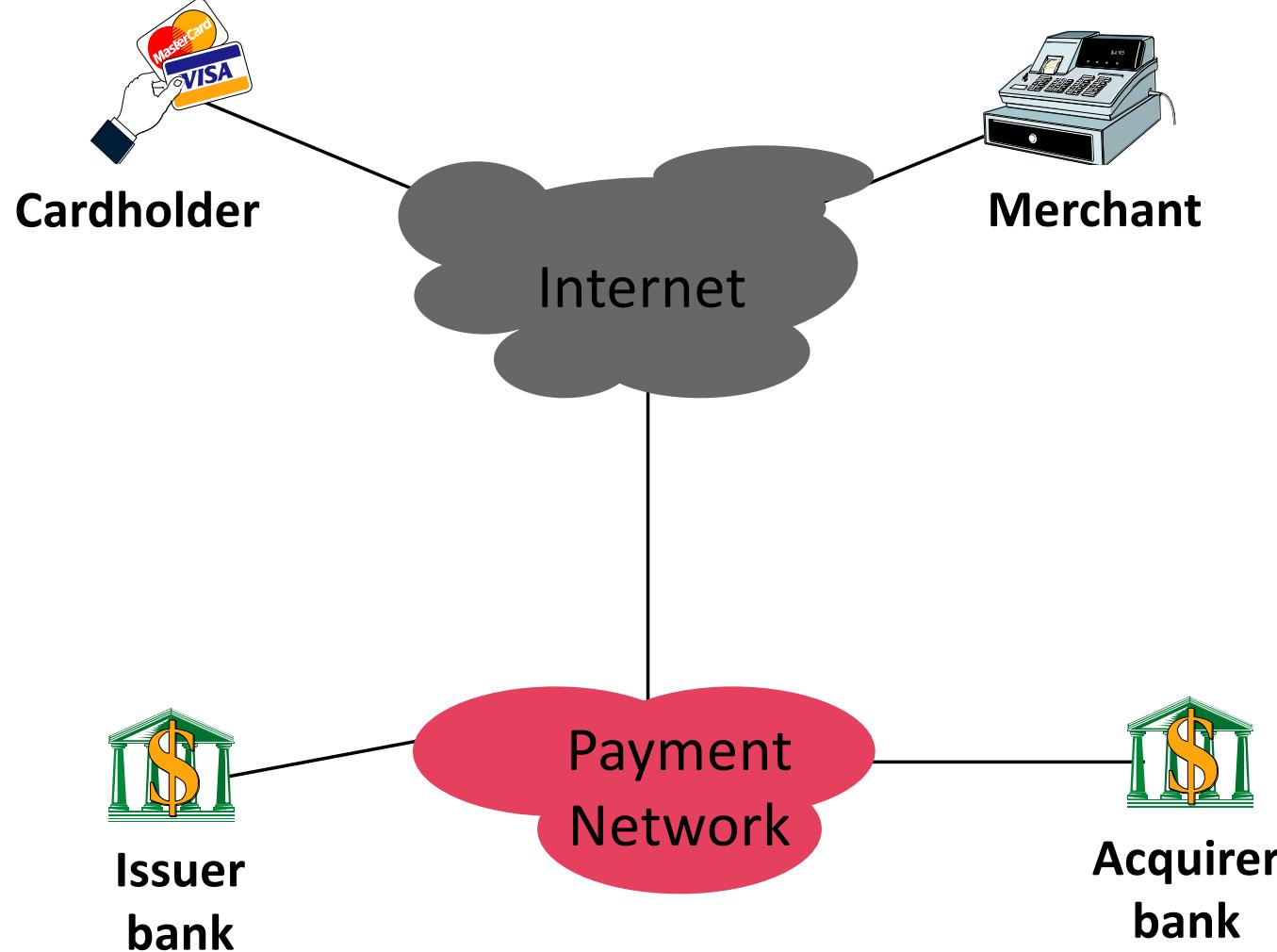
- **On-line y trazables:**

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

- **Micropagos:**

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

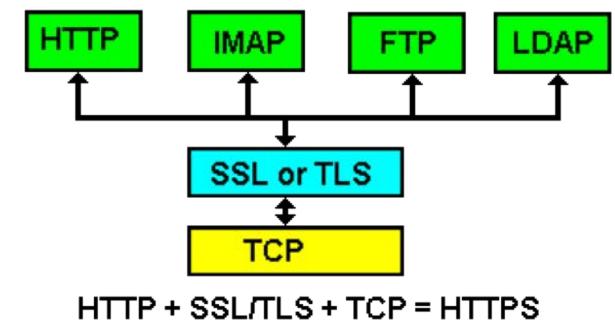
Tarjetas de crédito

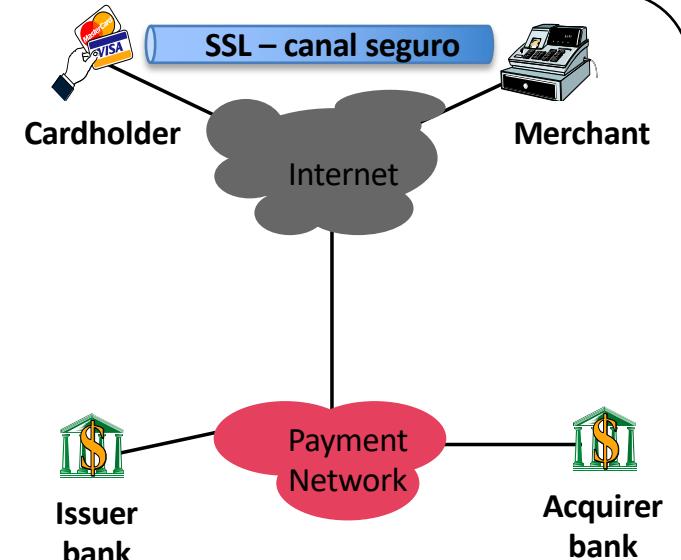


- Problemas en estos medios de pagos:
 - Ataques de escucha
 - Suplantación de identidad (cliente o comerciante)
 - Generación de dato
 - Modificación del dato
 - Etc.
- Soluciones para evitar estos problemas:
 - Mecanismos criptográficos
 - Mecanismos de autenticación de usuarios
 - Firma digitales
 - Certificados digitales

Protocolo SSL (protocolo original)

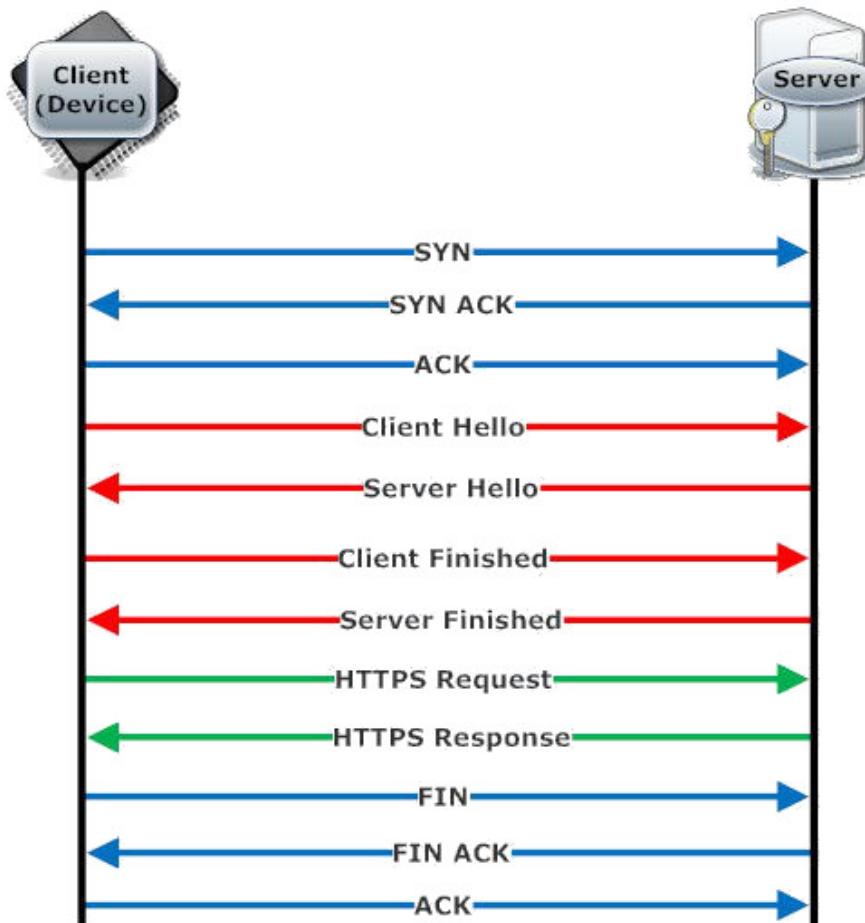
- SSL (Secure Sockets Layer) son protocolos de propósito general para establecer conexiones seguras
 - No son protocolos de pago, pero se usa por seguridad
 - SSL es original de Netscape (1994).
 - Última versión: SSLv3 – *No se utiliza en la actualidad*
- TLS (Transport Secure Layer) se creó dentro del IETF
 - Utiliza el mismo formato para la cabecera de los paquetes que SSL
 - La primera versión de TLS puede verse como SSLv3.1, pero difiere en:
 - Número de versión
 - En el código de autenticación del mensaje (MAC)
 - En la función pseudo-aleatoria
 - En los códigos de alerta
 - En la lista de algoritmos de cifrado
 - En los mensajes de verificación del certificado y de finalización
 - En algunas partes del algoritmo criptográfico





- SSL aplica:
 - Criptografía simétrica:
 - Cifrado de mensajes
 - DES, 3DES, RC2, RC4 o IDEA
 - Criptografía de asimétrica:
 - Generación y gestión de certificados digitales
 - Generación y transferencia segura de la clave de sesión
 - RSA o Diffie-Hellman
- SSL autentifica al servidor
 - Utilizando certificados digitales X.509 v3
 - Opcionalmente, también puede certificar al cliente
- SSL asegura la integridad de los datos
 - A través de códigos de autenticación de mensajes (MAC) que utilizan algoritmos de resumen digital (hash)
 - MD5 o SHA-1

- SSL provee **confidencialidad en el pago electrónico**
 - Garantiza la creación de un canal seguro entre cliente y servidor



- SSL provee **confidencialidad en el pago electrónico**
 - Garantiza la creación de un canal seguro entre cliente y servidor
- Sin embargo, presenta 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
 - cualquier persona con acceso al número de una tarjeta podría realizar una transacción
 - No hay mecanismos de facturación o de gestión de recibos
 - cualquier reclamación queda a la buena voluntad del vendedor

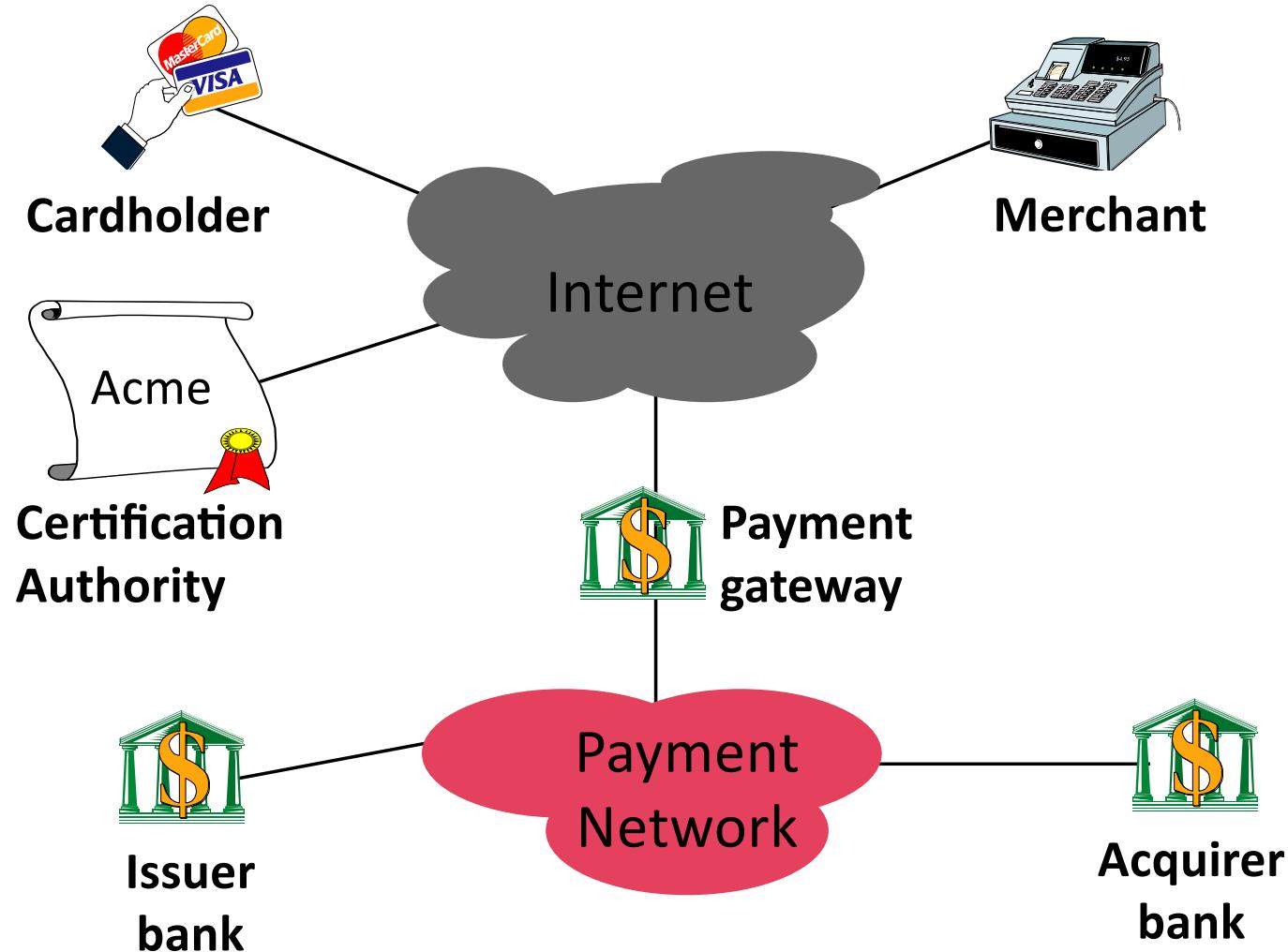
Se requiere otros tipos de protocolos más específicos ...

Protocolo SET (Secure Electronic Transactions)

- Protocolo desarrollado en 1996 por VISA y Mastercard (+ American Express), en colaboración con:
 - IBM
 - Microsoft
 - Verisign
 - RSA
 - Netscape
 - GTE
- Objetivo: 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



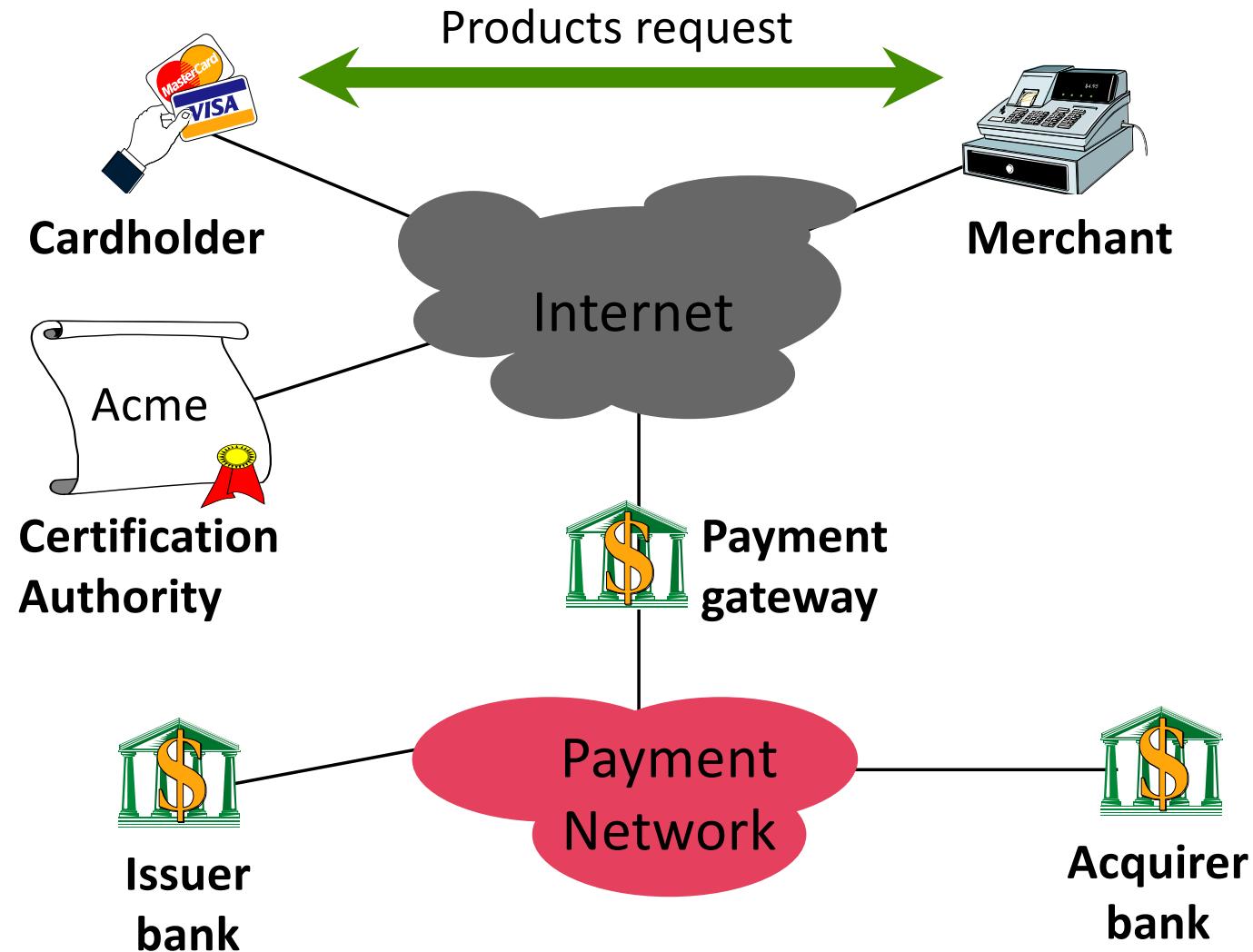
- Arquitectura SET:



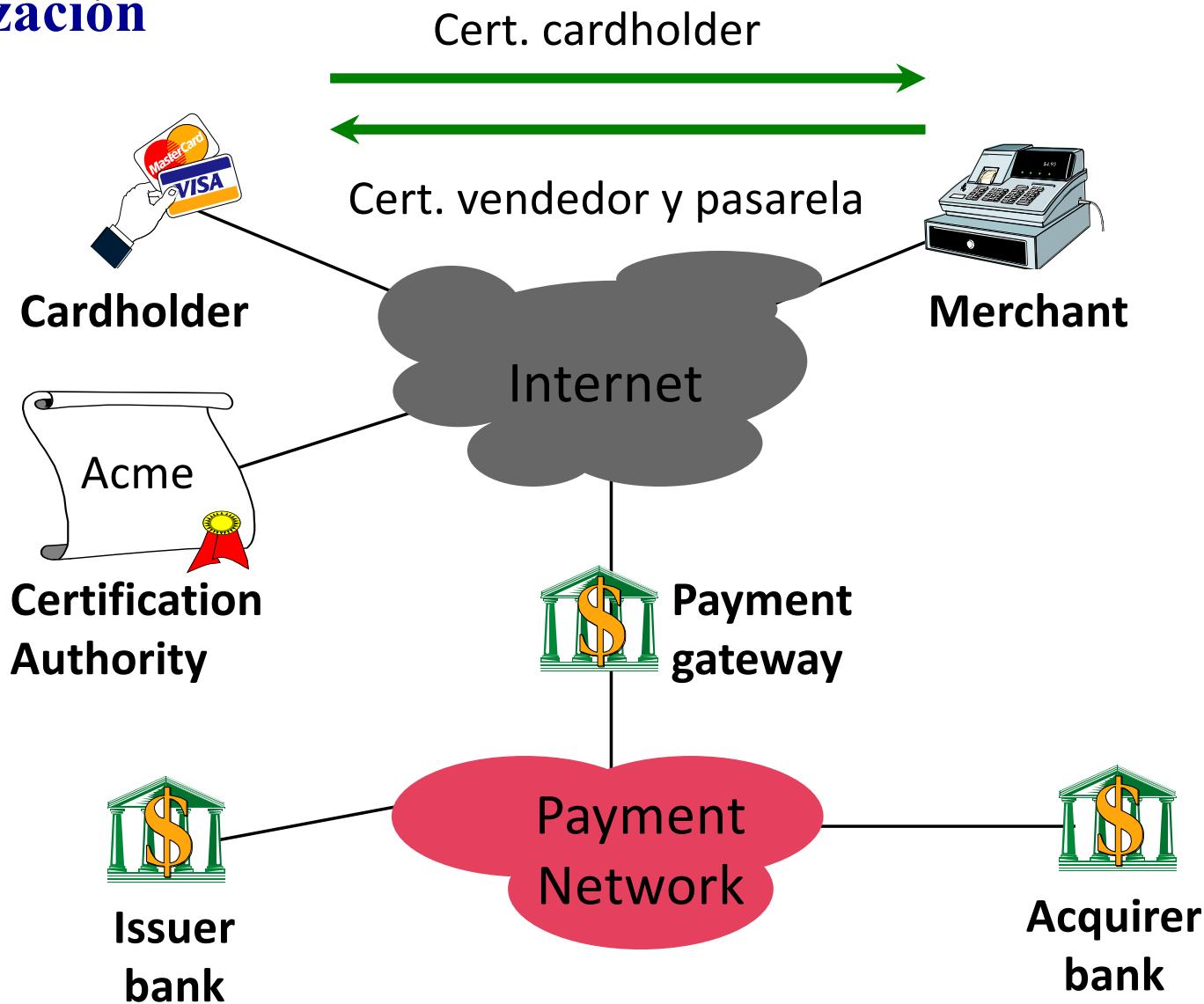
- SET
 - A diferencia de SSL, fue diseñado para el comercio electrónico
 - Sin embargo, no es un sistema de pago en sí mismo, sino un **conjunto de protocolos de seguridad y de formatos estándar**
 - que permiten a los usuarios usar de una forma segura a través de Internet la infraestructura ya existente de tarjetas de crédito
- SET proporciona:
 - **Canal confidencial** de comunicaciones entre las entidades que intervienen en la transacción
 - **Autenticidad**, 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
 - 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**, a través de firma digital
 - **Reduce las disputas y el no repudio**
 - Autorización de pago
 - Confirmación de la transacción
 - Garantía de pago al vendedor

- Pasos de una transacción:
 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

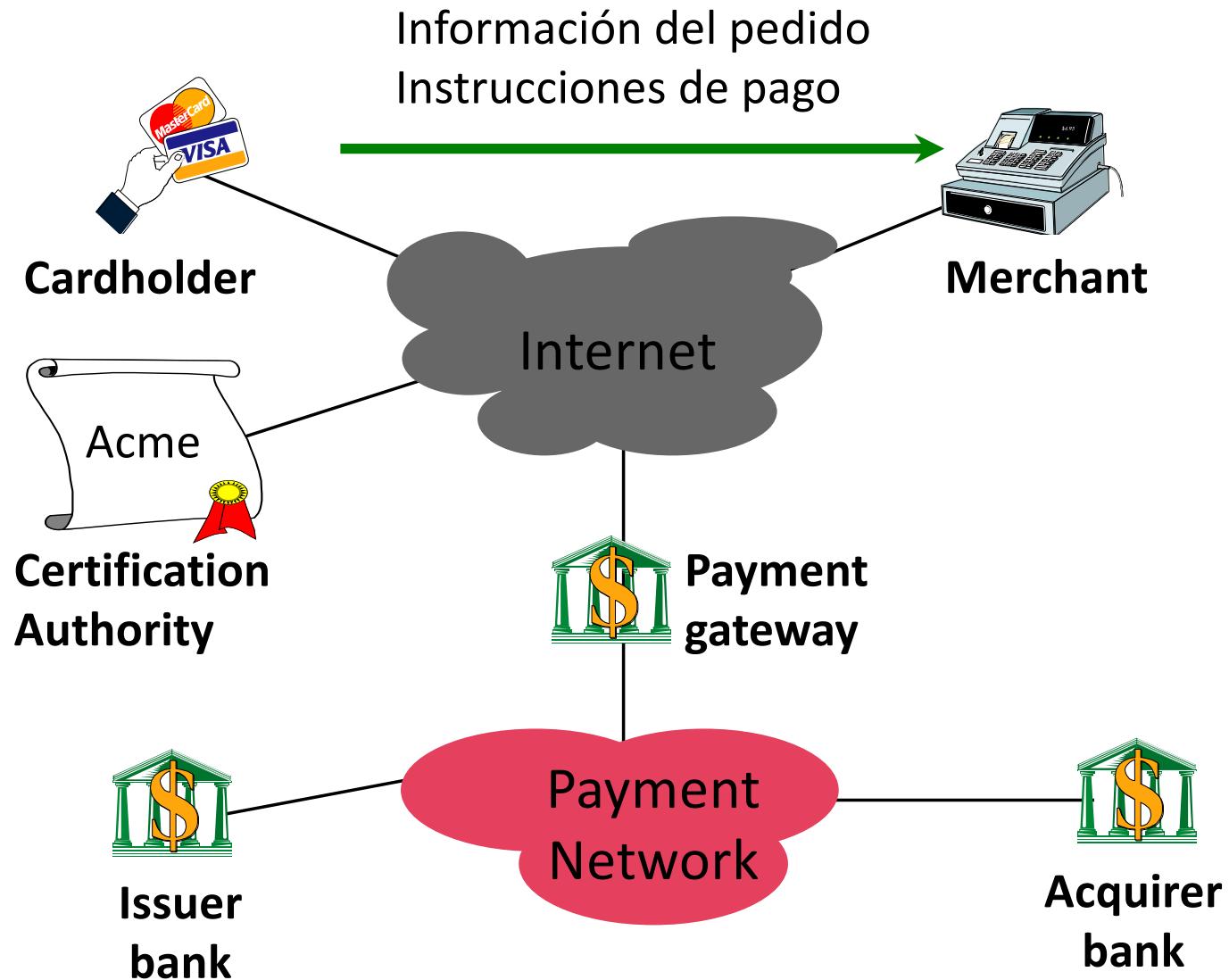
1. Petición



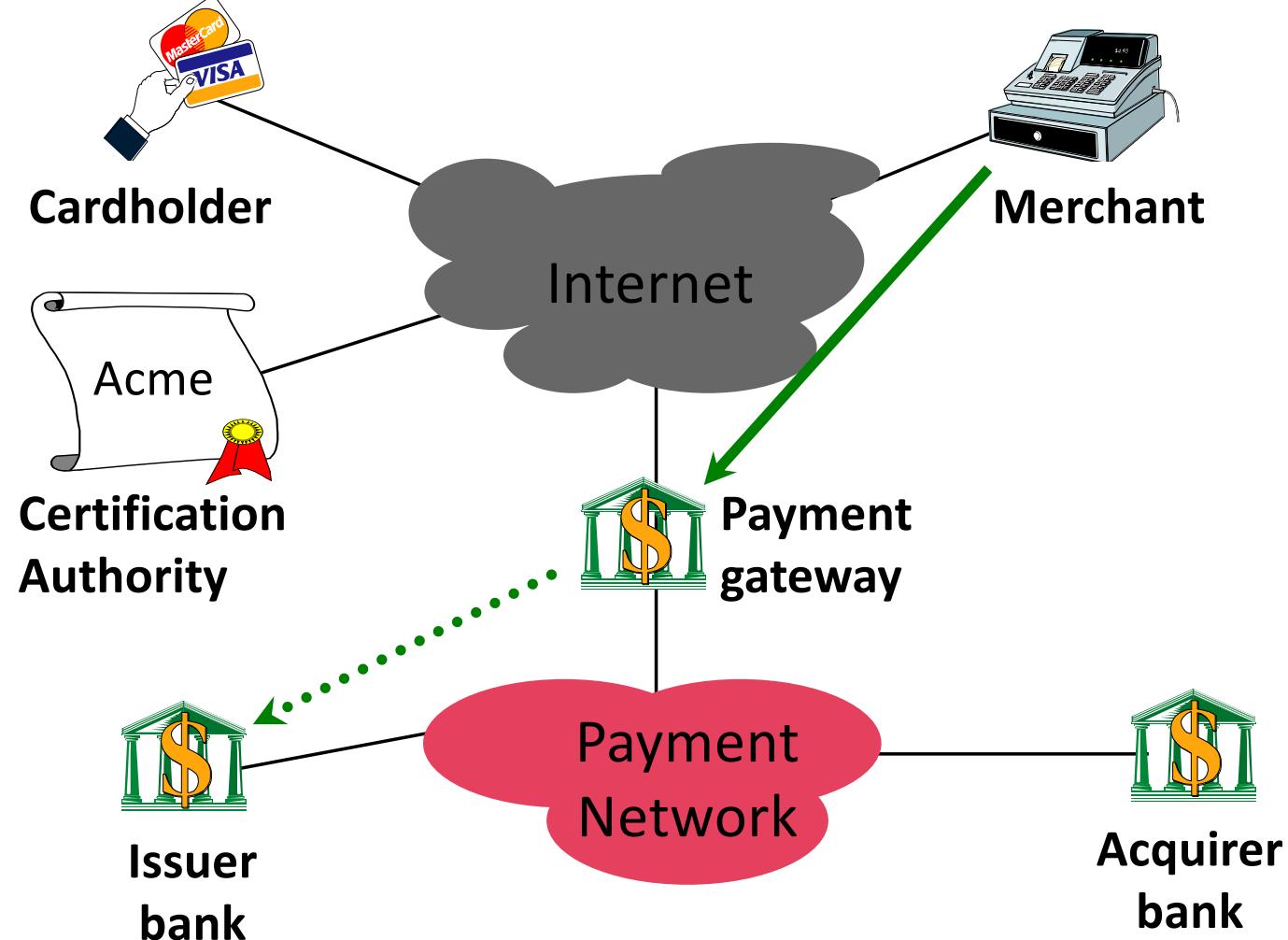
2. Inicialización



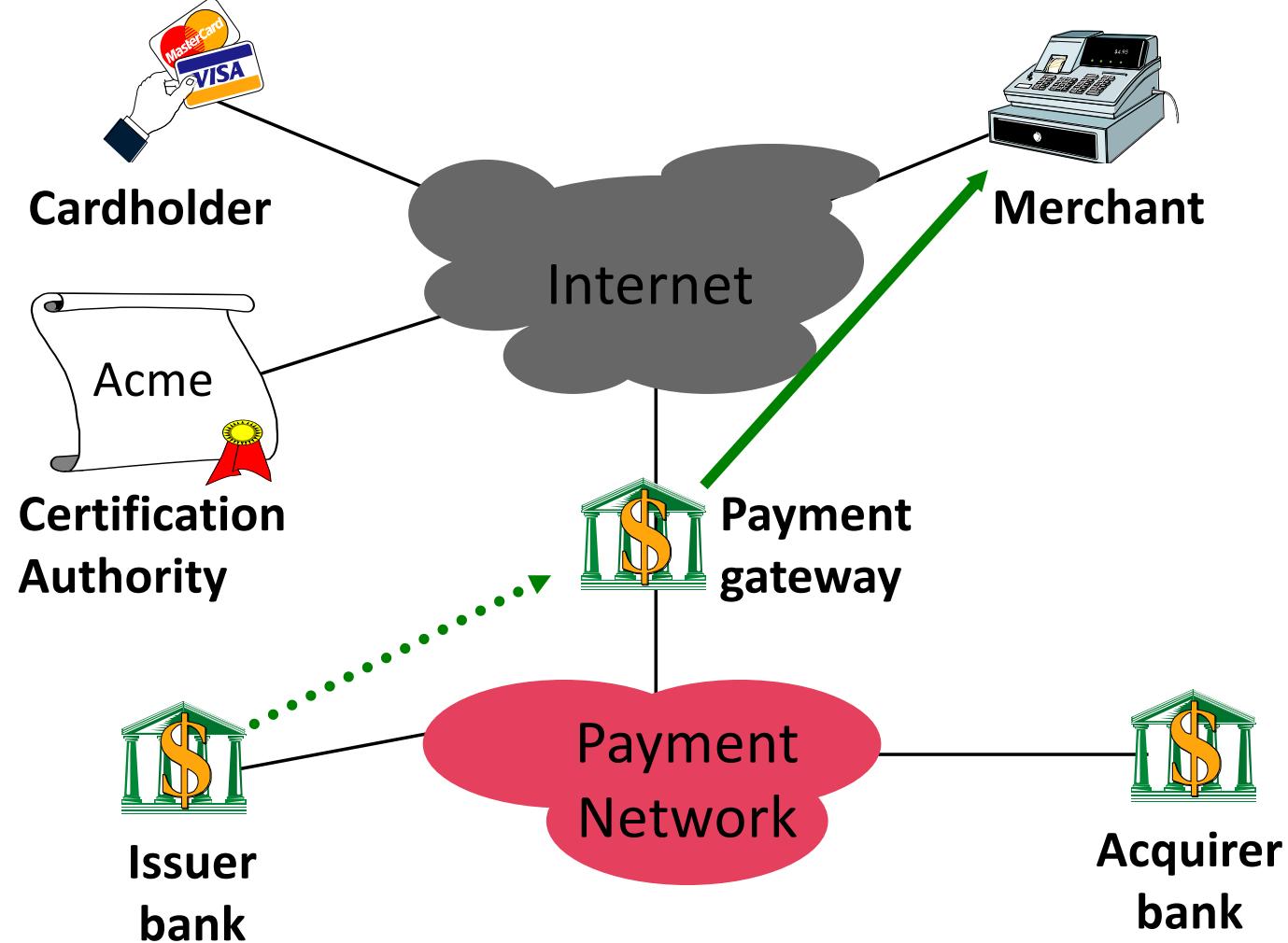
3. Información del pedido e instrucciones de pago



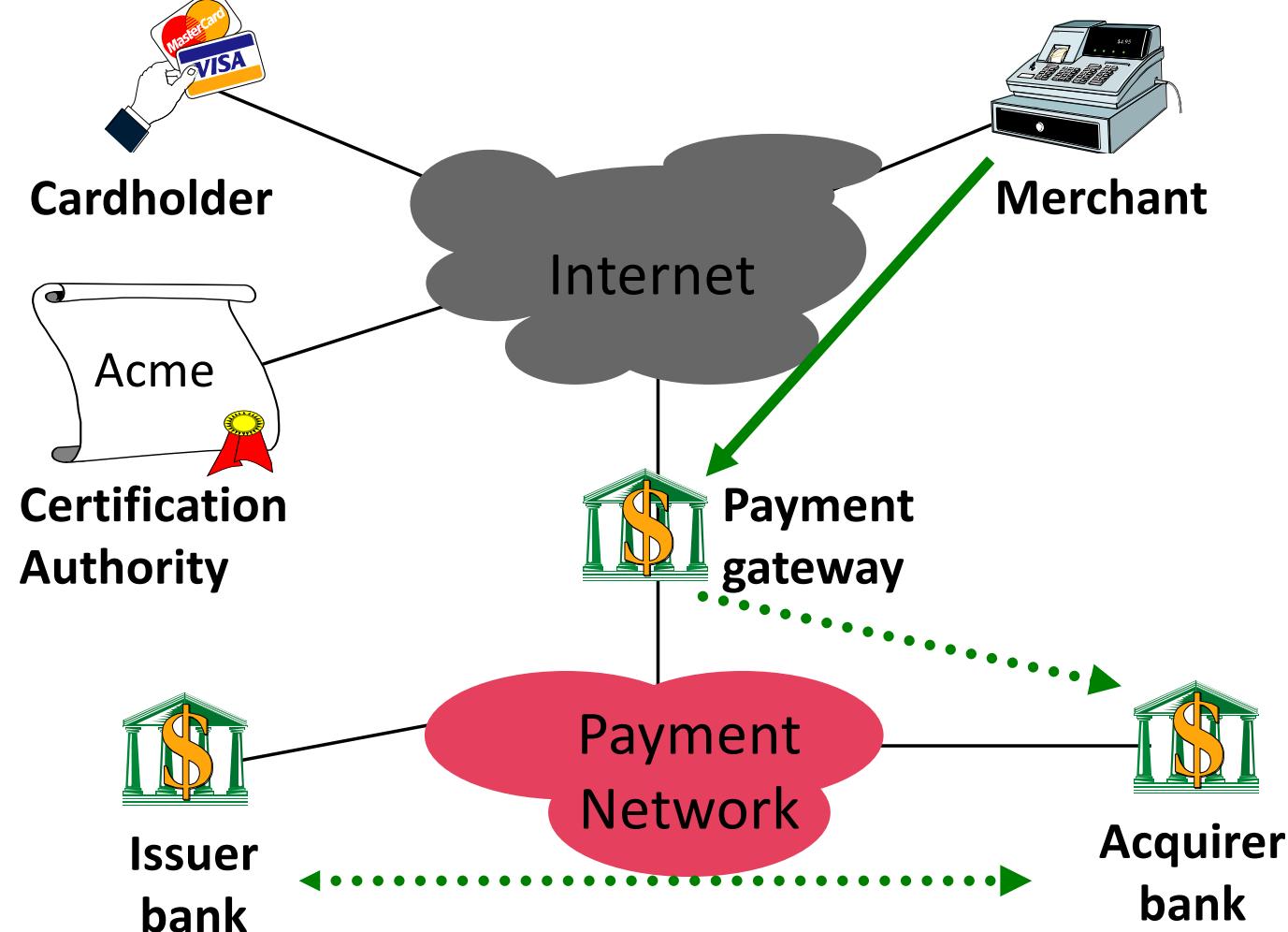
4. Petición de autorización



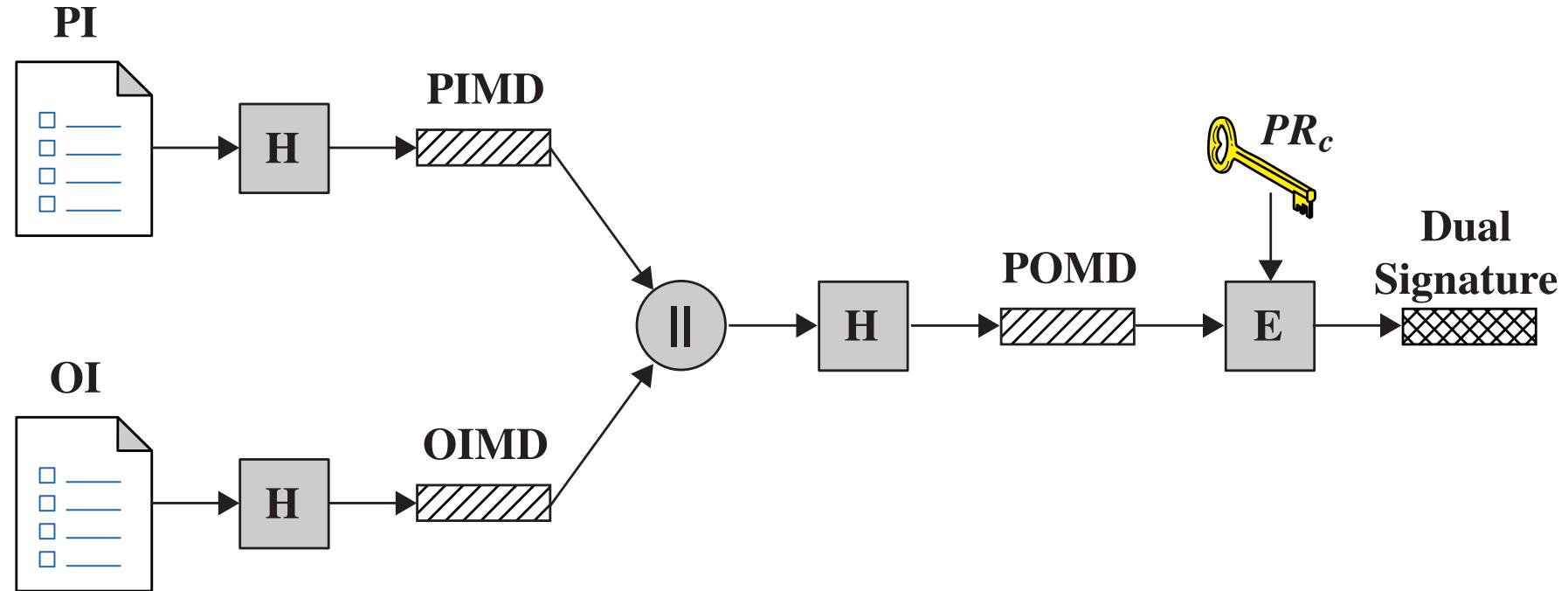
5. Aprobación de autorización



6. Finalización



- 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 diferentes
- En este caso, el cliente quiere enviar:
 - la información del pago (Payment Information) al banco
 - la información del pedido (Order Information) al comerciante
- Se ofrece mayor privacidad al cliente si ambos ítems se mantienen por separado dado que, de todos modos, para realizar sus funciones:
 - ni el comerciante necesita conocer el número de tarjeta del cliente
 - ni el banco necesita conocer los detalles del pedido del cliente
- No obstante, es necesario que ambos ítems queden enlazados de alguna forma, para una posible resolución de disputas posterior



PI = Payment Information

OI = Order Information

H = Hash function (SHA-1)

\parallel = Concatenation

PIMD = PI message digest

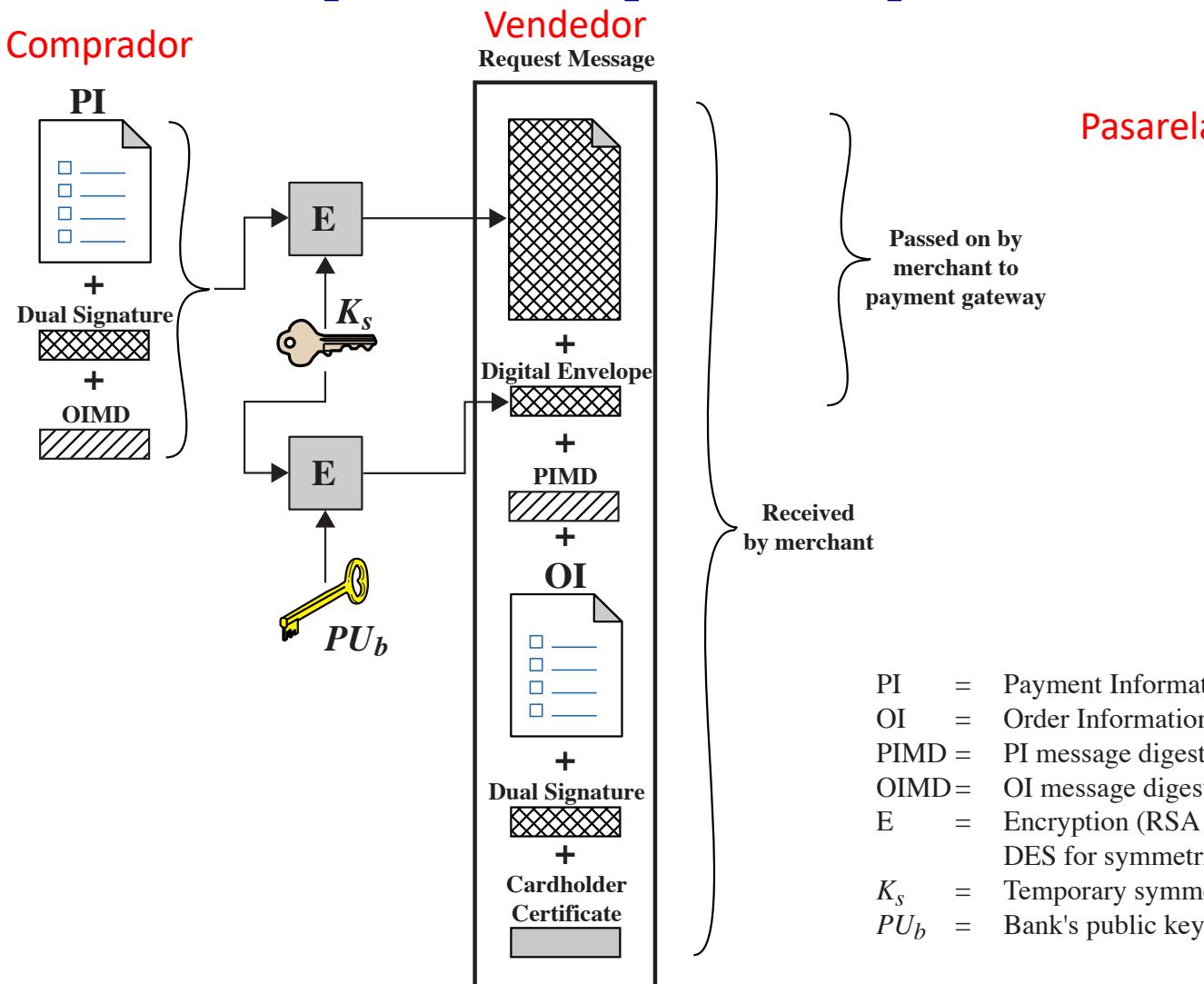
OIMD = OI message digest

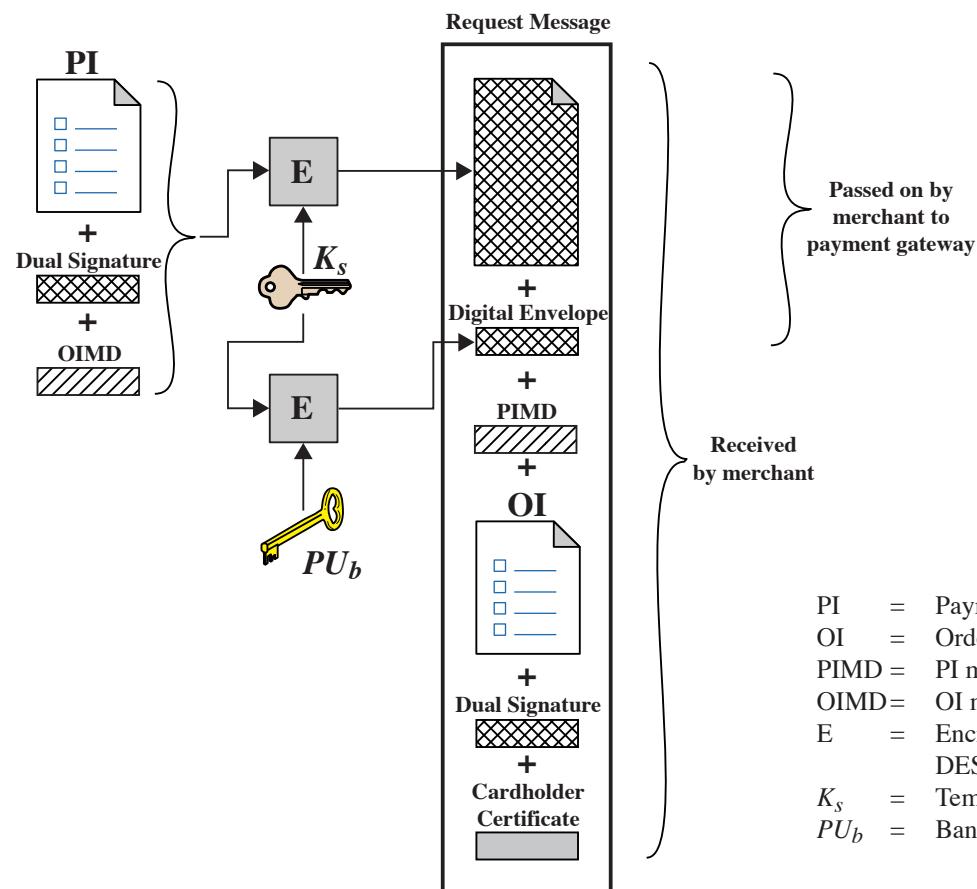
POMD = Payment Order message digest

E = Encryption (RSA)

PR_c = Customer's private signature key

- Petición de compra enviada por el comprador al vendedor:





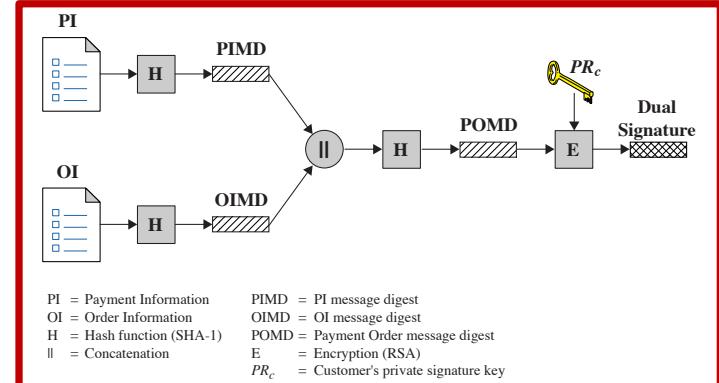
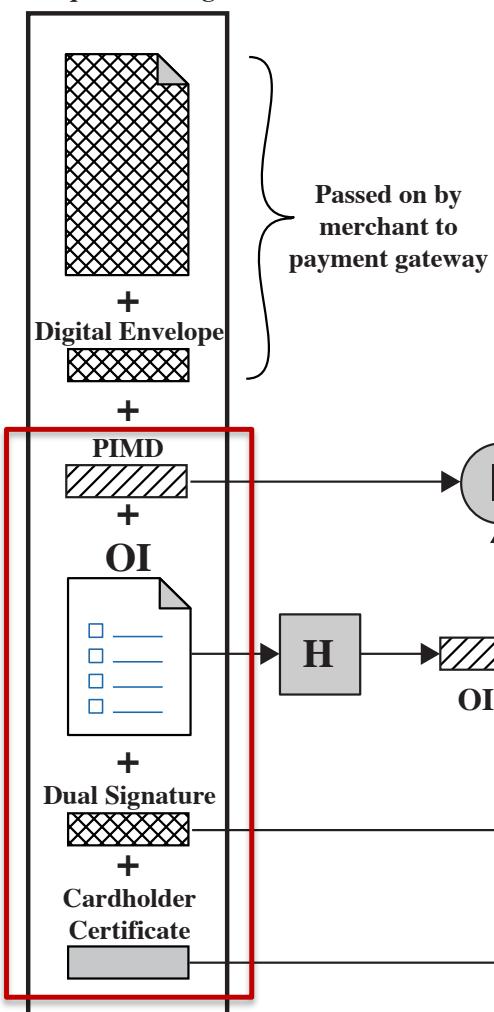
¿Qué hace el vendedor al recibir el mensaje?

¿Qué hace el banco al recibir el mensaje?

PI	= Payment Information
OI	= Order Information
PIMD	= PI message digest
OIMD	= OI message digest
E	= Encryption (RSA for asymmetric; DES for symmetric)
K_s	= Temporary symmetric key
PU_b	= Bank's public key-exchange key

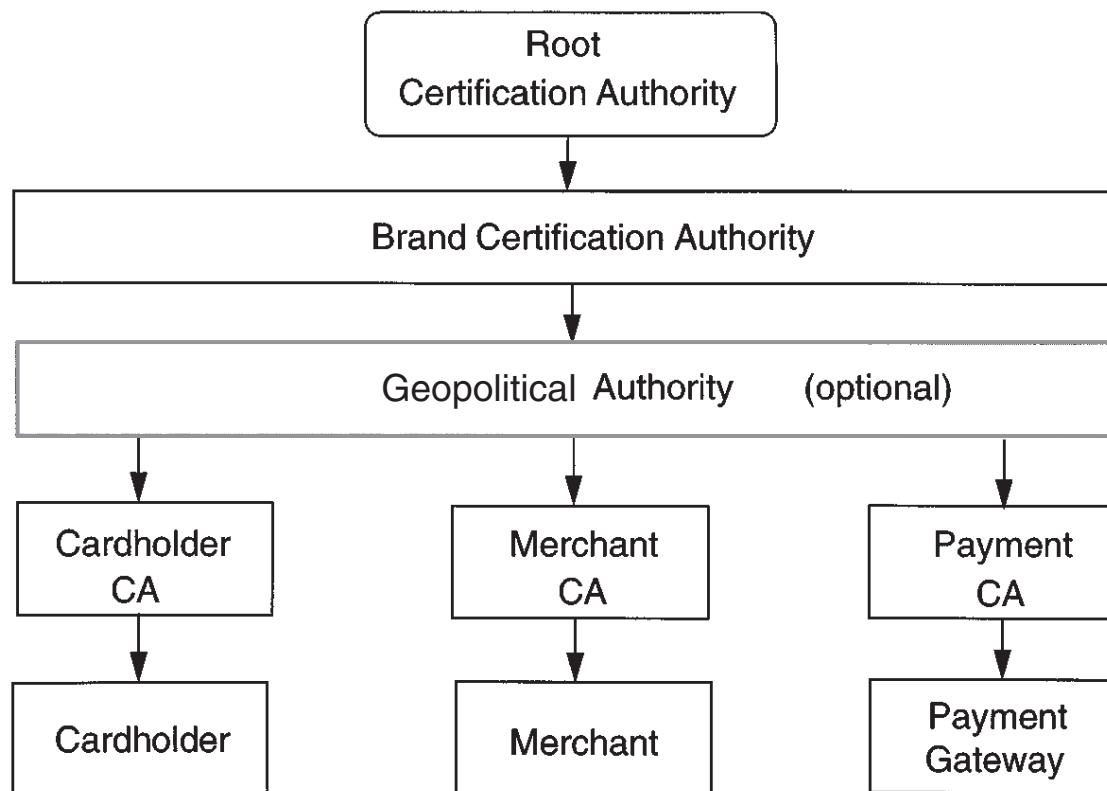
- Verificación del vendedor:

Request Message



OI = Order Information
 OIMD = OI message digest
 POMD = Payment Order message digest
 D = Decryption (RSA)
 H = Hash function (SHA-1)
 PU_c = Customer's public signature key

- PKI de SET

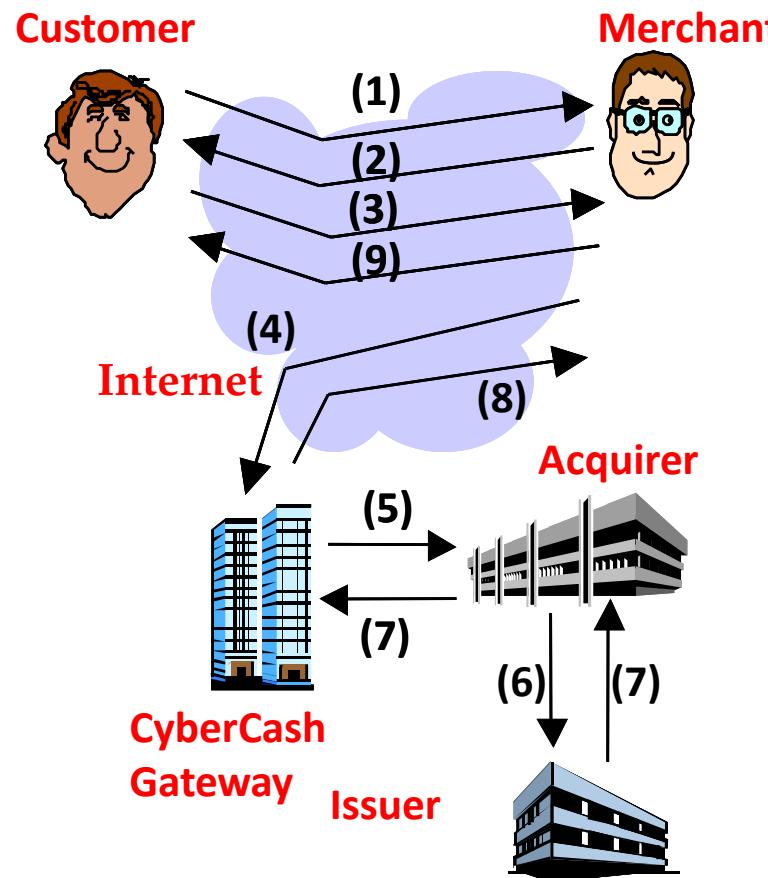


- Ventajas de uso:
 - Muy seguro y bien diseñado
 - Se garantiza autenticación, confidencialidad, integridad y no-repudio
 - Si el banco del comprador autoriza el pago, el vendedor tiene la garantía de ese pago
 - Evita que el vendedor acceda a los datos de la tarjeta
 - Evita que el banco acceda a la información de los productos comprados
- Desventajas de uso:
 - Dependiente de algoritmos específicos (RSA, DES, SHA1)
 - Gestión de certificados digitales
 - Fuerte esfuerzo para la implantación (especialmente para el vendedor)
 - No adaptado a micropagos

Protocolo Cyberscash

- 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





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**.

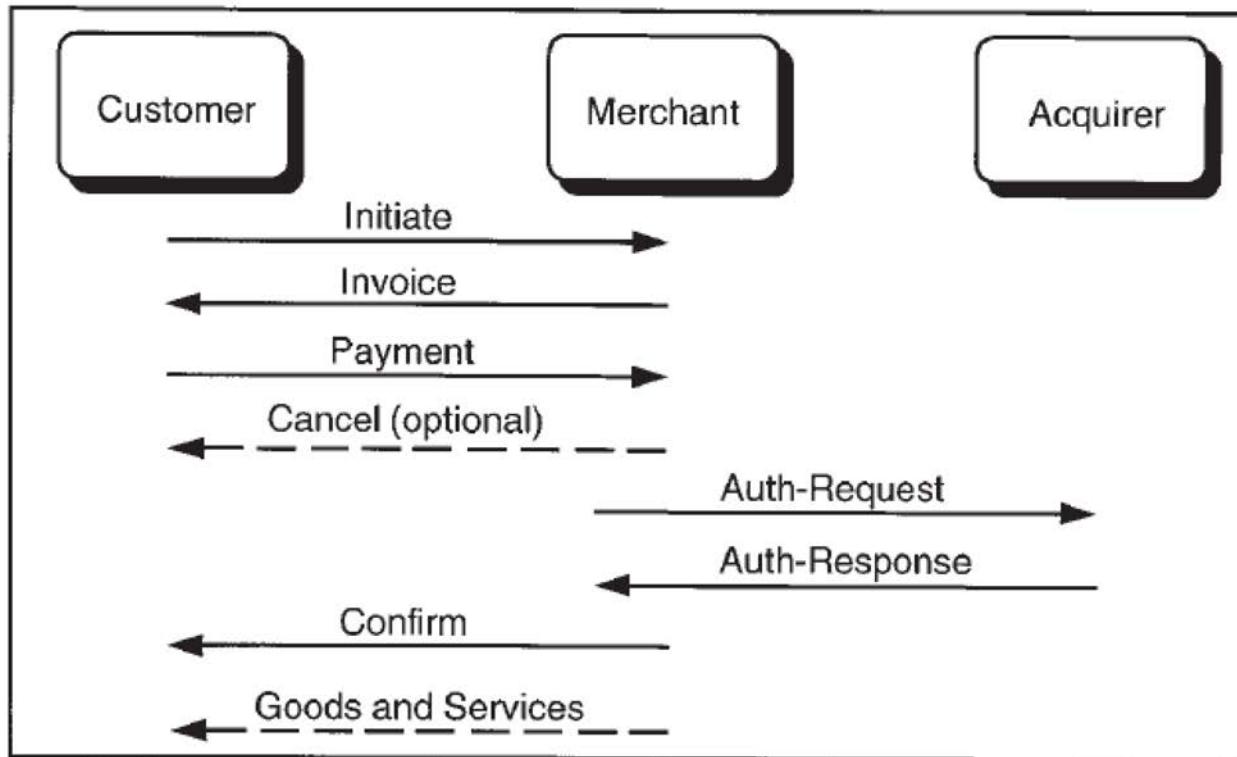
- Algunos problemas:
 - Parte de la información del cliente es conocida por la pasarela, por lo que se pueden analizar los hábitos del cliente
 - **Problema de privacidad** que no existía en SET
 - Uso de DES (56 bits) y RSA (1024 bits)
- Tras caer en bancarrota, Verisign adquirió los derechos sobre la marca y el protocolo de pago
- Posteriormente Paypal compró la solución a Verisign



Protocolos iKP

- iKP ($i = 1, 2, 3$) es una familia de protocolos de pago, **basado en criptografía de clave pública**, y desarrollado por IBM
- Estos protocolos (1KP, 2KP, 3KP) se diferencian entre sí en el número de entidades que poseen su propio par de claves públicas
 - Cuanto mayor sea el número de entidades que posean un par *<clave pública, clave privada>*, mayor es el nivel de seguridad proporcionado
- La implantación de los protocolos 2KP y 3KP se realiza de forma gradual para conseguir un pago seguro multiparte completo, requiriendo una infraestructura de certificación avanzada

- Cada uno de los protocolos consta de seis pasos:



- El contenido de estos pasos varía dependiendo de a qué protocolo nos estemos refiriendo

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

Item	Description	Item	Description
CAN	Customer's account number (e.g., credit card number)	Common	Information held in common by all parties: PRICE, ID _M , TID _M , DATE, NONCE _M , CID, H(DESC, SALT _C), [H(V)]
ID _M	Merchant ID; identifies merchant to acquirer	Clear	Information transmitted in the clear: ID _M , TID _M , DATE, NONCE _M , H(Common), [H(V)]
TID _M	Transaction ID; uniquely identifies the transaction	SLIP	Payment instructions: PRICE, H(Common), CAN, R _C , [PIN]
DESC	Description of the goods; includes payment information such as credit card holder's name and bank identification number	EncSlip	Payment instruction encrypted with the public key of the acquirer: PK _A (SLIP)
SALT _C	Random number generated by C; used to randomize DESC and thus ensure privacy of DESC on the M to A link	CERT _X	Public-key certificate of X, issued by a CA
NONCE _M	Random number generated by a merchant to protect against replay	Sig _A	Acquirer's signature: SK _A [H(Y/N, H(Common))]
DATE	Merchant's current date/time	Sig _M	Merchant's signature in Auth-Request: SK _M [H(H(Common), [H(V)])]
PIN	Customer's PIN which, if present, can be optionally used in 1KP to enhance security	Sig _C	Cardholder's signature: SK _C [H(EncSlip, H(Common))]
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		

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

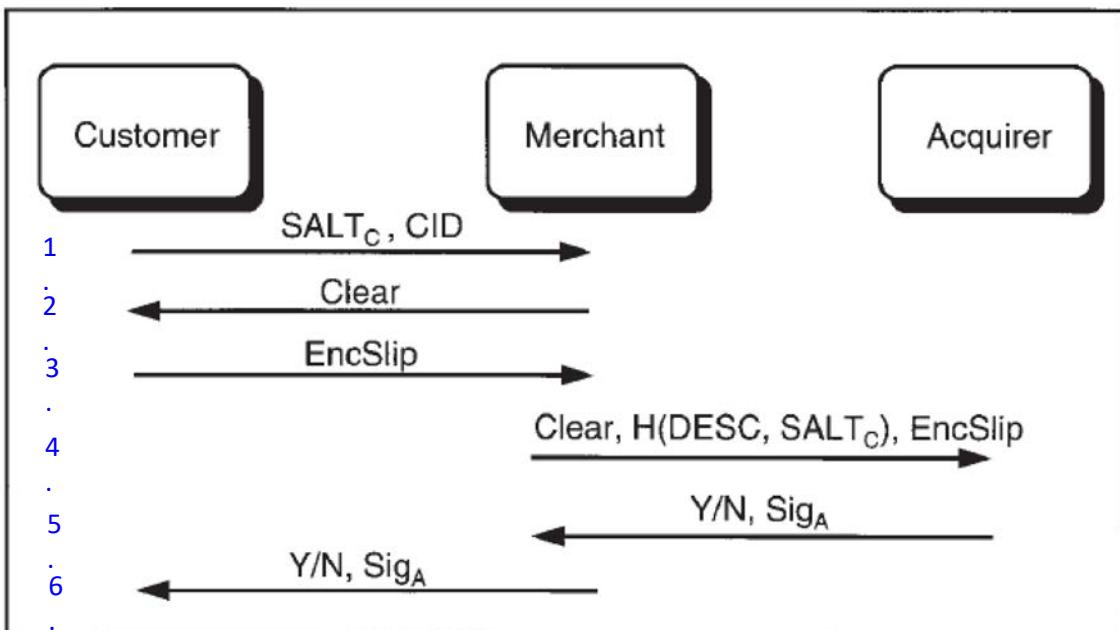
Item	Description
CAN	Customer's account number (e.g., credit card number)
ID_M	Merchant ID; identifies merchant to acquirer
TID_M	Transaction ID; uniquely identifies the transaction
DESC	Description of the goods; includes payment information such as credit card holder's name and bank identification number
$SALT_C$	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 $CID = H(R_C, CAN)$
V	Random number generated in 2KP and 3KP by merchant; used to bind the Confirm and Invoice message flows

Actor	Information Items
Customer	DESC, CAN, PK_{CA} , [PIN], CERT_A
Merchant	DESC, PK_{CA} , CERT_A
Acquirer	SK_A , CERT_A

- El **protocolo 1KP** es el más básico. Sólo el Acquirer necesita poseer (y distribuir) su certificado de clave pública $CERT_A$.
- La información de partida de cada una de las entidades es:
- Los pasos del protocolo son:

Actor	Information Items
Customer	DESC, CAN, PK_{CA} , [PIN], <u>$CERT_A$</u>
Merchant	DESC, PK_{CA} , <u>$CERT_A$</u>
Acquirer	SK_A , <u>$CERT_A$</u>

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: $PK_A (\text{SLIP})$
$CERT_X$	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: $SK_C[H(\text{EncSlip}, H(\text{Common}))]$

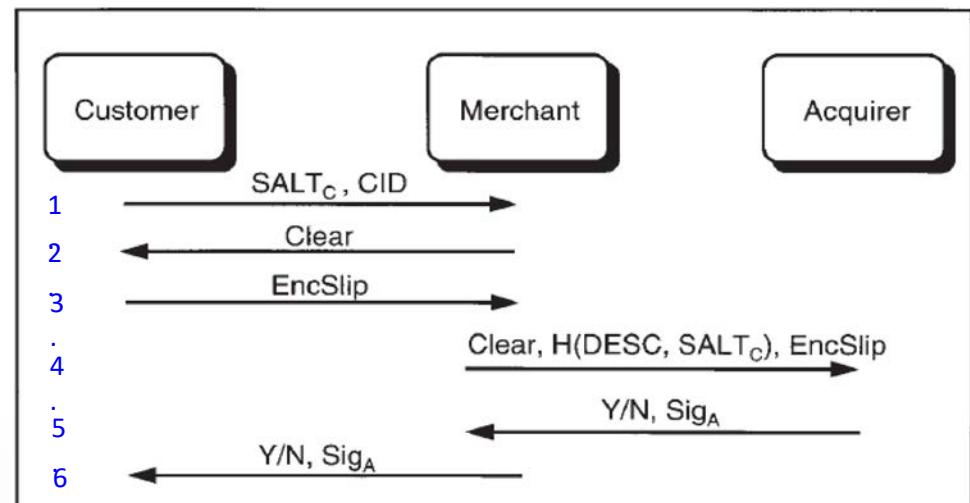


• Ejercicio:



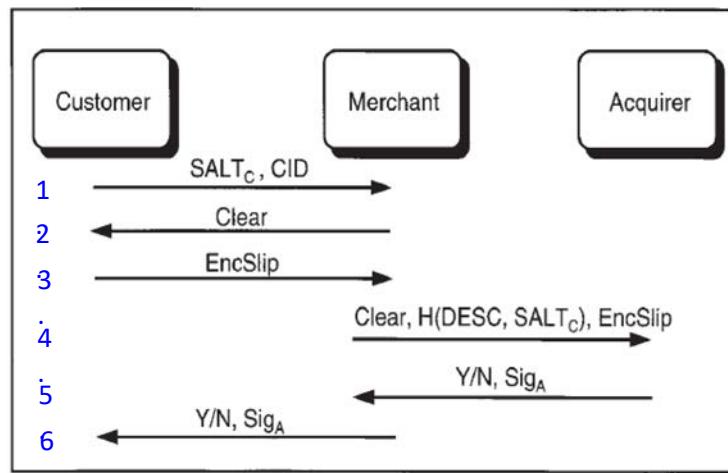
Item	Description
CAN	Customer's account number (e.g., credit card number)
ID _M	Merchant ID; identifies merchant to acquirer
TID _M	Transaction ID; uniquely identifies transaction
DESC	Description of the goods; includes cardholder's name and bank information
SALT _C	Random number generated by customer to ensure privacy of DESC on the message
NONCE _M	Random number generated by merchant
DATE	Merchant's current date/time
PIN	Customer's PIN which, if known, can enhance security
Y/N	Response from card issuer to merchant's request
R _C	Random number chosen by cardholder
CID	A customer pseudo-ID where $CID = H(R_C, CAN)$
V	Random number generated by merchant to confirm and invoice message

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))]



- Pasos en detalle:

1. $SALT_C, H(R_C, CAN)$
2. $ID_M, TID_M, DATE, NONCE_M, H(PRICE, ID_M, TID_M, DATE, NONCE_M, H(R_C, CAN), H(DESC, SALT_C))$
3. $P_{KA} (PRICE, H(PRICE, ID_M, TID_M, DATE, NONCE_M, H(R_C, CAN), H(DESC, SALT_C)), CAN, R_C, [PIN])$
4. $ID_M, TID_M, DATE, NONCE_M, H(PRICE, ID_M, TID_M, DATE, NONCE_M, H(R_C, CAN), H(DESC, SALT_C)), H(DESC, SALT_C), P_{KA} (PRICE, H(PRICE, ID_M, TID_M, DATE, NONCE_M, H(R_C, CAN), H(DESC, SALT_C)), CAN, R_C, [PIN])$
5. $Y/N, \text{Sig}_A$
6. $Y/N, \text{Sig}_A$



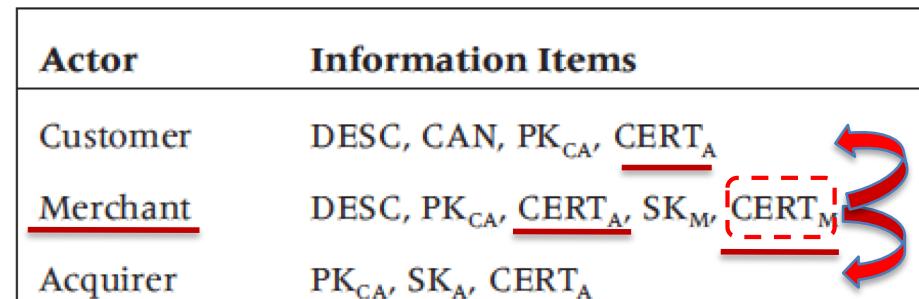
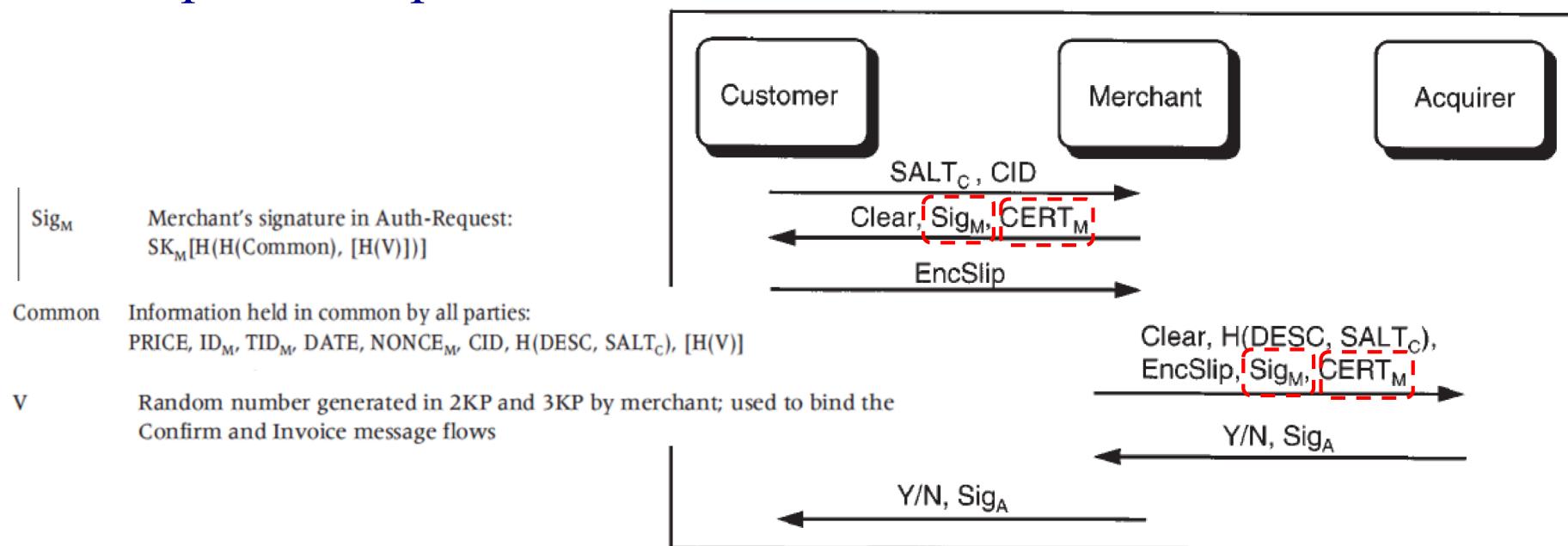
Item	Description
CAN	Customer's account number (e.g., credit card number)
ID_M	Merchant ID; identifies merchant to acquirer
TID_M	Transaction ID; uniquely identifies the transaction
DESC	Description of the goods; includes payment information such as credit card holder's name and bank identification number
$SALT_C$	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, $NONCE_M$, $H(\text{Common})$, $[H(V)]$
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 el Acquirer
 - Ni el vendedor ni el cliente proporcionan evidencias de intervención en la transacción

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

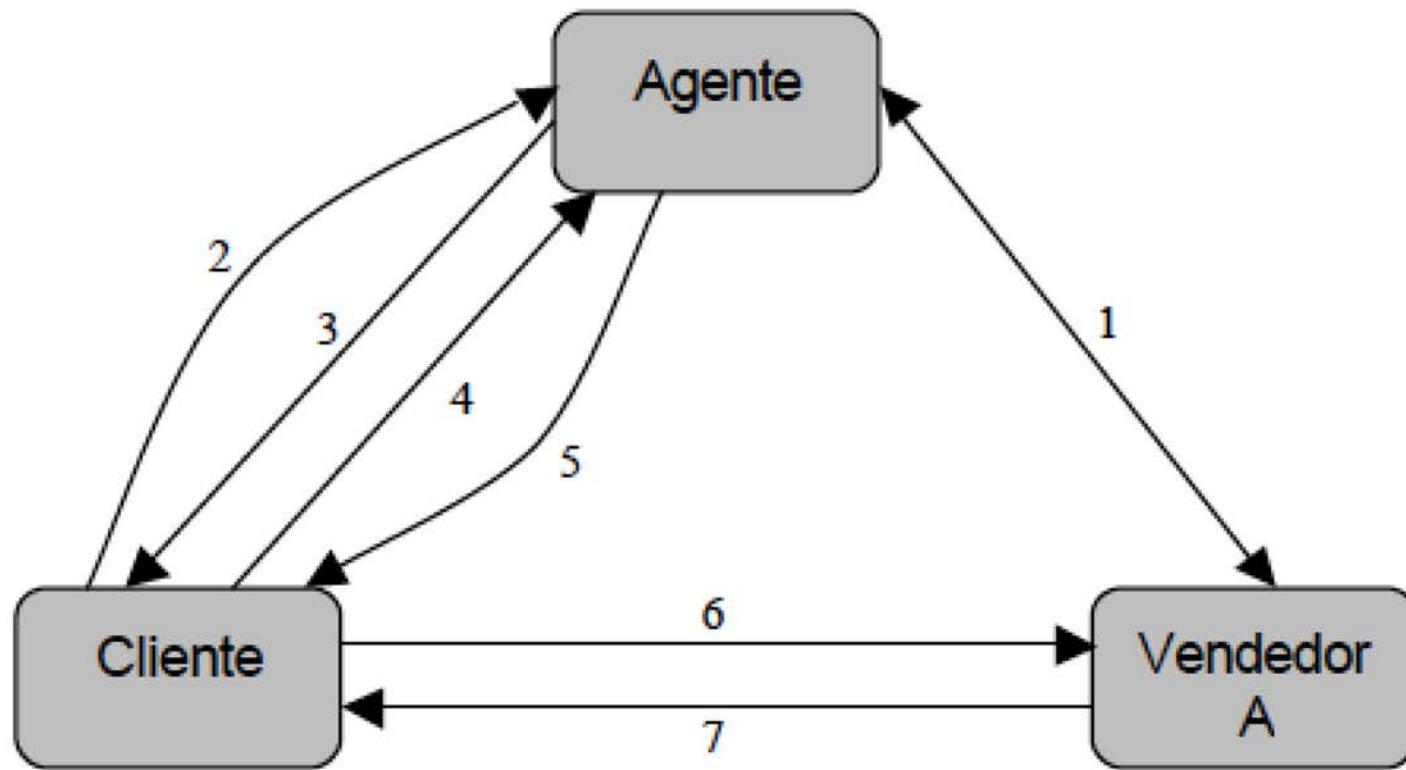
Actor	Information Items
Customer	DESC, CAN, PK_{CA} , $CERT_A$
Merchant	DESC, PK_{CA} , $CERT_A$, SK_M , $CERT_M$
Acquirer	PK_{CA} , SK_A , $CERT_A$

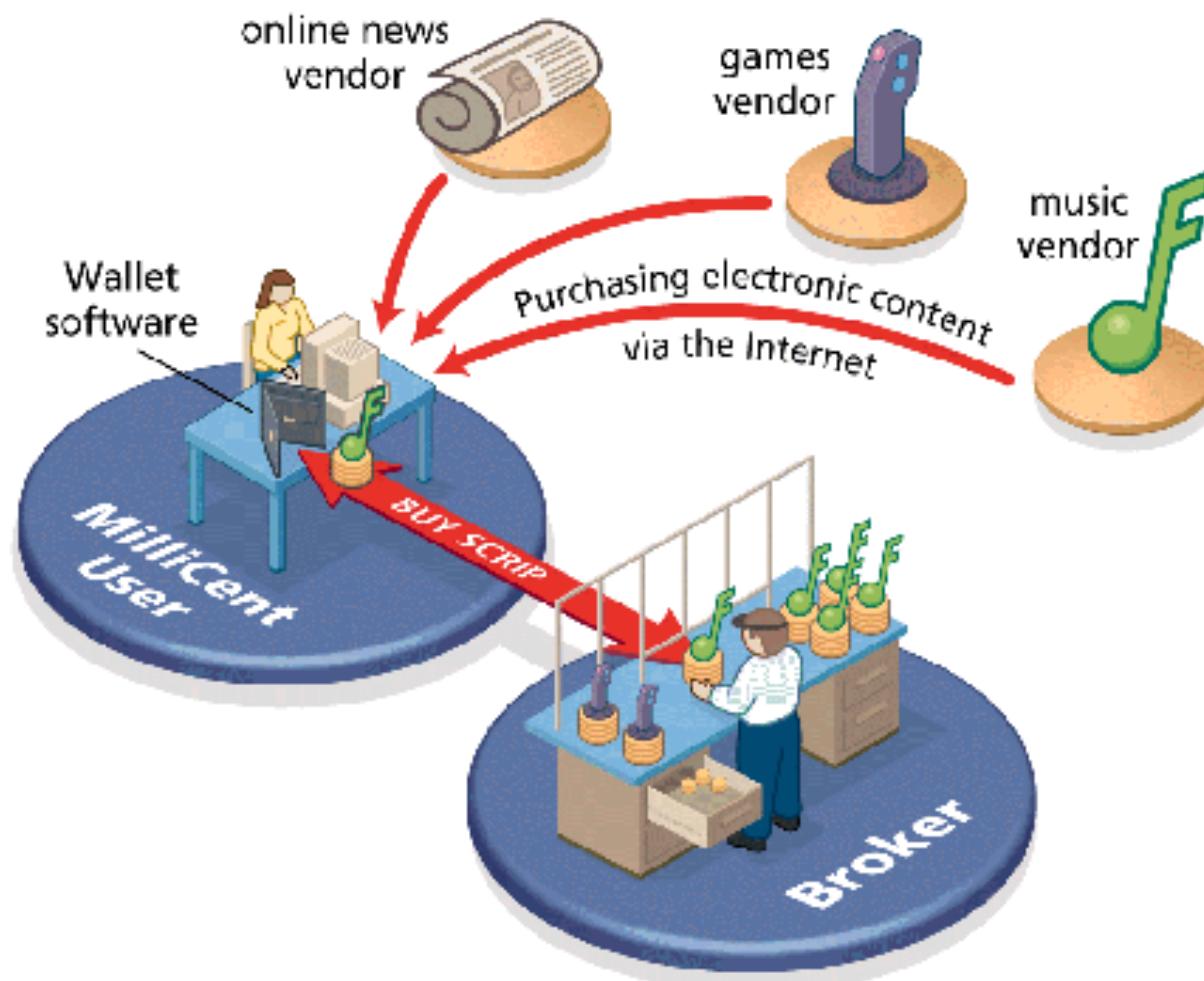
Protocolo Millicent

- 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 operacionales se utilizan varias soluciones:
 - servicios de prepago
 - autorizaciones off-line
 - agrupación de la facturación de los micropagos en lotes
 - reducción del coste computacional
 - la criptografía de clave pública no es la más adecuada pues resulta cara, e incluso, los criptosistemas simétricos pueden ser cuestionables
 - cada vez más se aplica el empleo de **funciones hash**
 - sin embargo, conlleva también la imposibilidad de proporcionar servicios de no-repudio

- Un ejemplo es Millicent que utiliza cifrado simétrico, y no utiliza procesamiento on-line
- 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 a cada uno de los potenciales vendedores del sistema
 - se puede suponer que, durante un periodo, las compras de un cliente a varios comerciantes alcanzarán un importe equivalente a un macropago
 - la función principal del **agente de negocios** es la de vender a cada cliente, y dentro de un mismo lote mixto, scrips de distintos vendedores



1. Compra-Venta de scripts de A
2. Compra de scripts de agente (macropago)
3. Envío de scripts de agente
4. Compra de scripts de A (micropago mediante scripts de agente)
5. Envío de scripts de A
6. Petición producto + micropago mediante scripts de A
7. Envío de producto



Fuente: <http://magsastre.eresmas.com/3-6comer.html>

- Millicent no es un sistema totalmente anónimo, aunque gracias a su modelo a tres bandas se consigue un cierto anonimato para el 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
- Otros sistemas de micropago son:
 - Subscrip
 - Kleline
 - Flattr
 - M-coin
 - etc.

PRIVACIDAD DE LOS USUARIOS EN APLICACIONES



Conceptos generales

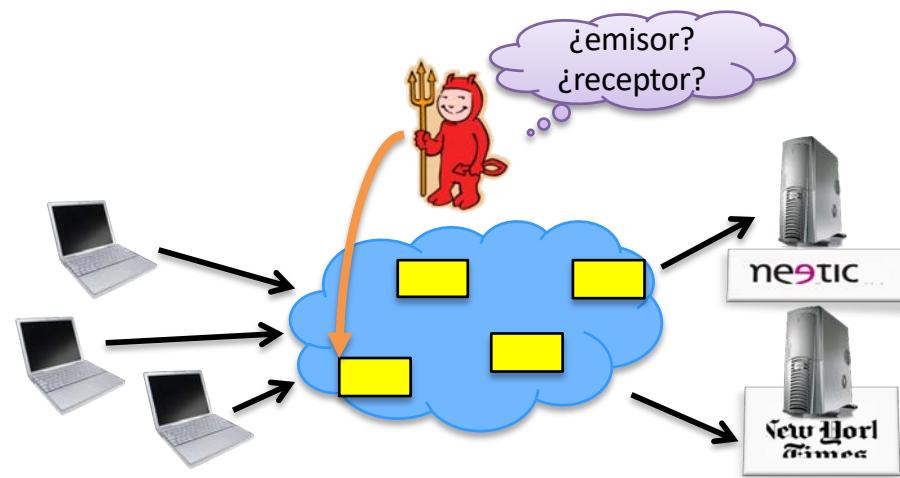
- La **privacidad** puede definirse como:
 - el derecho de los individuos y entidades de proteger, salvaguardar y controlar el acceso, almacenamiento, distribución y uso de información sobre su “**propia persona**”

¿Cómo se aseguraría la privacidad?

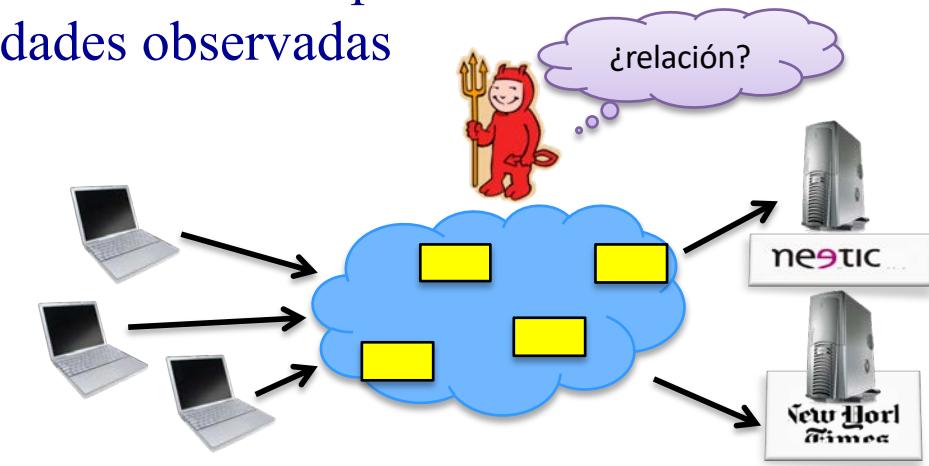
Conceptos generales

- 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**”
 - confidencialidad no es equivalente a privacidad
 - la confidencialidad es relativa a los datos mientras que la privacidad es relativa a las personas
- ¿Qué información es necesario proteger?
 - Dependerá de lo que el usuario considere información privada. Además, también es relevante de quién se protege la información
 - Por lo general, se protege: identidad, localización, preferencias, ...

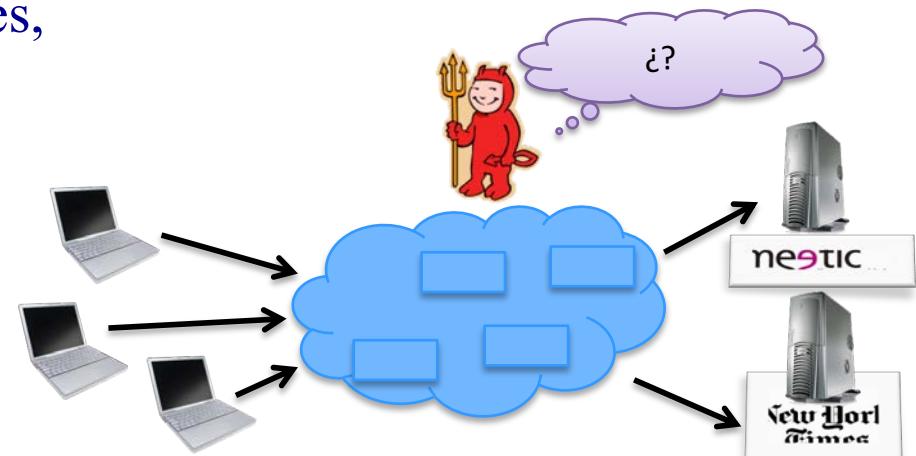
- Estrechamente relacionado con el concepto de privacidad, está el concepto de **anonimato**:
 - “Es el estado de no ser identifiable entre un grupo de sujetos, conocido como conjunto de anonimato”
 - Las **técnicas de anonimato** tratan de **hacer indistinguible a un individuo entre un conjunto** suficiente de **identidades con unas características similares**
 - **Pseudoanonimato**: se refiere al uso de pseudónimos en lugar de los identificadores normales con el fin de proporcionar anonimato



- Otros conceptos adicionales relacionados:
 - **No-vinculación** (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 las entidades



- Por tanto, el anonimato depende de la técnica aplicada y se distinguen 4 tipos de anonimatos:
 - **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:**
 - Cuando se “comparte” conocimiento entre dos o más partes implicadas (ej. el vendedor y el banco)
 - Dependiendo de la honestidad de estas partes se puede o no revelar la identidad del usuario principal (ej. el comprador)
 - **Anonimato no rastreable:**
 - Trata de resolver el problema del anonimato rastreable a través de técnicas mucho más estrictas desde el punto de vista a la privacidad del usuario
 - **Anonimato no rastreable y no vinculante:**
 - La condición de anonimato no rastreable garantiza que la identidad de los usuarios no se puede revelar, y la condición de vinculante asegura que el conjunto de las operaciones realizadas por un usuario anónimo no se puedan vincular

- Para conseguir una o varias de las propiedades de privacidad mencionadas, 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

- A partir del concepto básico de firma digital estudiado en temas anteriores, surgen esquemas más avanzados de firma, con objetivos técnicamente más ambiciosos:
 - **Firma ciega**
 - El firmante firma mensajes para otros usuarios, pero desconoce el contenido de los mensajes que firma
 - **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
 - **Firma de anillo**
 - Similar al anterior, pero el anonimato es total; es decir, no es posible saber la identidad del firmante bajo ninguna circunstancia
 - **Firma umbral**
 - La firma la producen conjuntamente un mínimo de t usuarios, en nombre del grupo de n usuarios ($t < n$) del que forman parte

Firma ciega

- Existen aplicaciones, como por ejemplo, las de dinero digital y 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 *Alice* sea firmado por *Bob* sin que este conozca el contenido del mensaje
- La firma ciega resultante puede ser verificada públicamente con posterioridad, como una firma digital normal



- 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 *Bob*, y sea n el módulo RSA
- Y sea r un número aleatorio $\text{mod } n$ (r es el *blinding factor*)

1. Alice: r

2. Alice: $M' = M \cdot r^e \pmod{n}$

3. Alice → Bob: M'

4. Bob → Alice: $(M')^d \pmod{n} = [M^d \cdot (r^e)^d \pmod{n}] = [M^d \cdot r \pmod{n}]$

5. Alice: $M^d \cdot r \cdot r^{-1} \pmod{n} = [M^d \pmod{n}]$

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 un documento (en nombre del grupo)
- 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



- 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)
- Un esquema de firma de grupo debe satisfacer las siguientes propiedades o condiciones iniciales:
 - Sólo los miembros del grupo pueden firmar mensajes de forma correcta (infalsificable)
 - 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



- 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 de cada miembro del grupo
- una clave secreta de administración para el administrador

② **Firma** → es un algoritmo que:

- 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

③ Verificación → es un algoritmo que:

- 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

- **Ejemplo 1** - ejemplo básico de diseño:
 - El administrador proporciona a cada miembro del grupo una lista de claves privadas
 - Esas listas son disjuntas
 - El administrador divulga en un directorio público, y en orden aleatorio, la lista completa de claves públicas correspondientes
 - 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
 - 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)
 - 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

- **Ejemplo 2:**
 - Sea e el exponente público del grupo
 - 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 para cada miembro, en base al exponente e y a los n_i

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

- La **firma** por parte del miembro k se realiza así:

$$S = M^{dk} \bmod n_k$$

- La **verificación** por parte de cualquier usuario se realiza así:

$$M' = S^e \bmod n$$

- La **apertura de la firma**, en caso de que sea necesario, la realiza el administrador.

- ¿Cómo se haría?



- La **firma** por parte del miembro k se realiza así:

$$S = M^{d_k} \text{ mod } n_k$$

- 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:

$$S_1 = M^{d_1} \text{ (mod } n_1), S_2 = M^{d_2} \text{ (mod } n_2), \dots, S_N = M^{d_N} \text{ (mod } n_N)$$

Firma de anillo

- Las firmas de grupo son útiles cuando los miembros quieren cooperar, mientras que las de anillo son útiles cuando no quieren o no pueden cooperar
- Al contrario que las firmas de grupo, las firmas de anillo:
 - no tienen administradores de grupo
 - ni procedimiento de establecimiento
 - ni procedimiento de revocación del anonimato del firmante
 - ni coordinación
 - ni procedimiento para distribuir claves

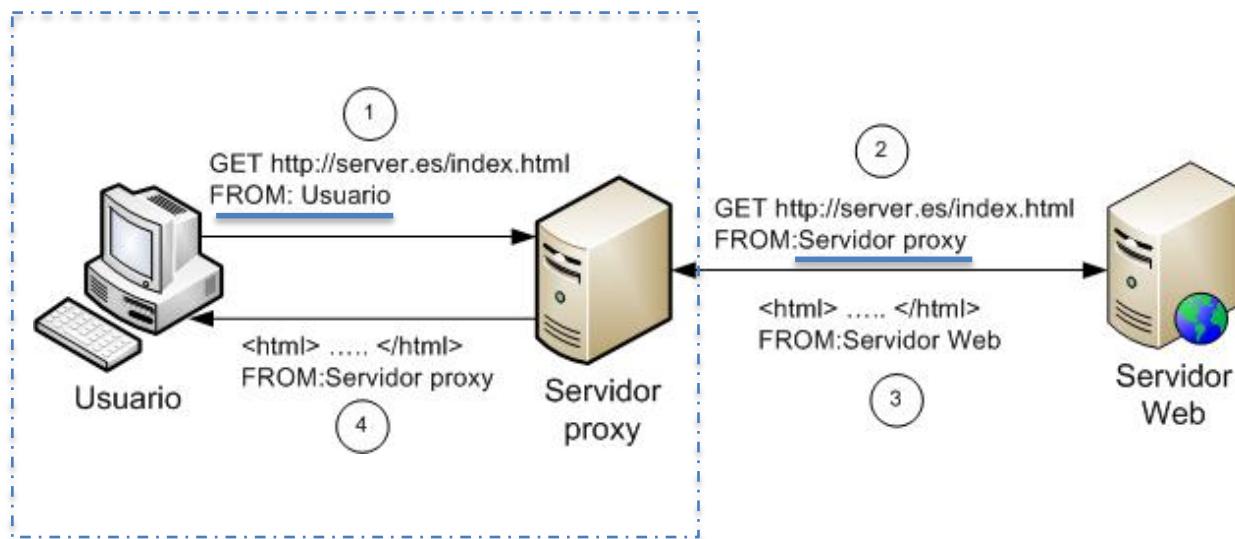
- Cualquier usuario puede elegir un conjunto de posibles firmantes incluyéndose él mismo, sin la aprobación ni ayuda de esos otros miembros del conjunto
 - computa la firma por sí mismo, usando sólo su clave privada y las claves públicas de los demás
- Es decir, los otros miembros del conjunto pueden 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
- El verificador es incapaz de determinar la identidad del firmante dentro de un anillo de tamaño r

Privacidad basada en
protocolos criptográficos y de enrutado

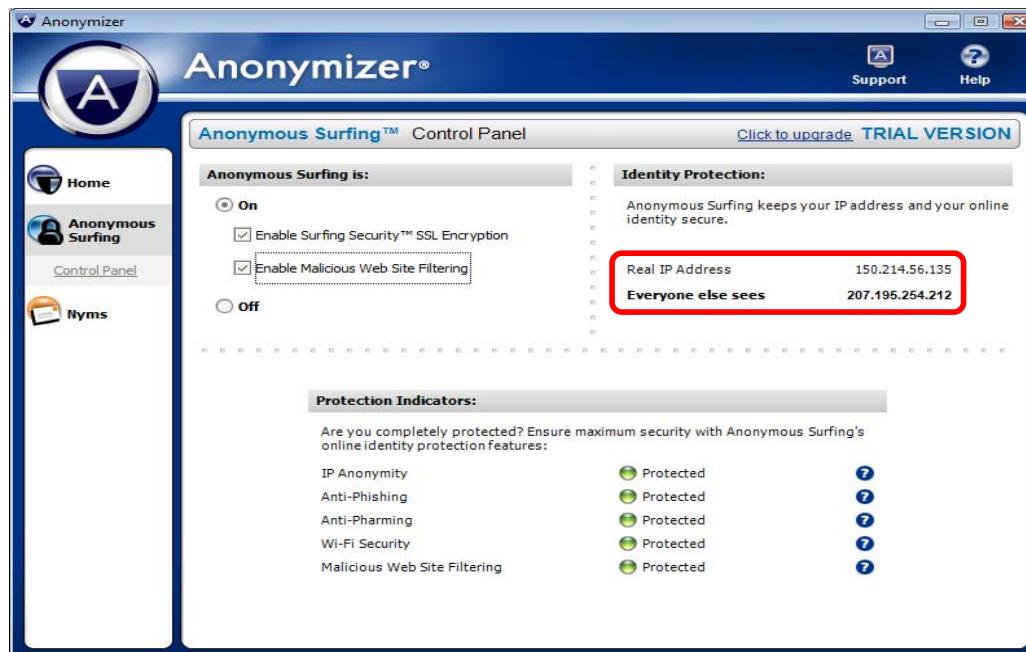
- Tratan de proteger el tráfico frente a entidades que se dedican a observar las comunicaciones
- Existen soluciones basadas en:
 - a) uso de proxy
 - b) uso de mezcladores
 - c) conocimiento parcial de la ruta
 - d) creación de gruposademás de algunas **soluciones híbridas**



a) **Soluciones basadas en el uso de proxy:** un **servidor proxy** hace de intermediario en la comunicación, aceptando conexiones de los clientes y reenviándolas



- Entre las soluciones basadas en proxy destacan:
 - **Anonymizer**: el cliente solicita una página web al proxy y éste origina una nueva petición, almacenando internamente el cliente y la petición que realizó para reenviarle la respuesta
 - Anonymizer® Anonymous Surfing™ es un software comercial que mientras está activado permite la navegación a través de un proxy que oculta la verdadera dirección IP del usuario
 - Actualmente es posible el cifrado de los datos entre el cliente y el proxy para proporcionar confidencialidad en ese enlace



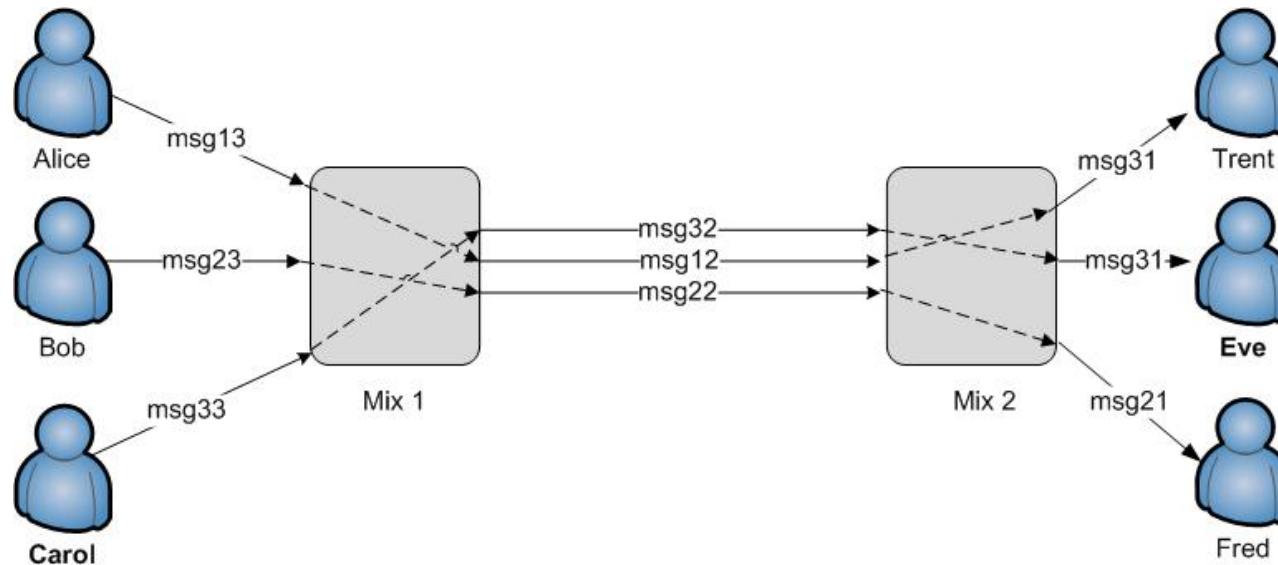
- LPWA (*Lucent Personalized Web Assistant*): elimina cualquier información de identidad de los datos del usuario y crea una conexión SSL entre el usuario y el servidor proxy
- Proxy caché: cuando un usuario solicita una página web el proxy la recupera de una caché donde guarda las peticiones de otros usuarios
 - esto mejora el tiempo de respuesta además de evitar ataques en los que se pueda relacionar el tráfico generado por un cliente con la recibida por un servidor

- Se pueden encontrar otras soluciones proxy:

- The Cloak <http://www.the-cloak.com>
- Proxy Web.net <http://www.proxyweb.net>
- SnoopBlocker <http://www.snoobblocker.com>
- Proxify <http://proxify.com>
- Anonymouse <http://anonymouse.org>
- Web Warper <http://webwarper.net>

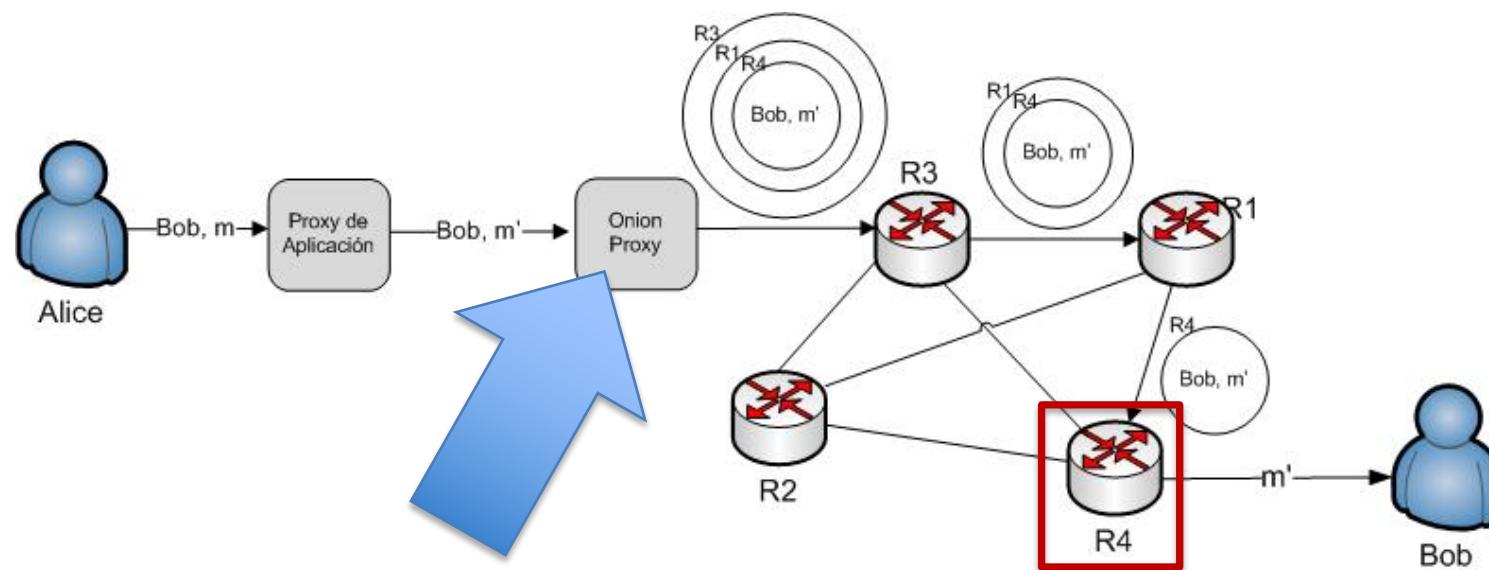
b) Soluciones basadas en el uso de mezcladores (mixers)

- Mixnets: se basan en la utilización de *mixers*, que son dispositivos de almacenamiento y envío
 - el usuario envía el mensaje a través de **mixers**
 - estos 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



c) Soluciones basadas en el conocimiento parcial de la ruta

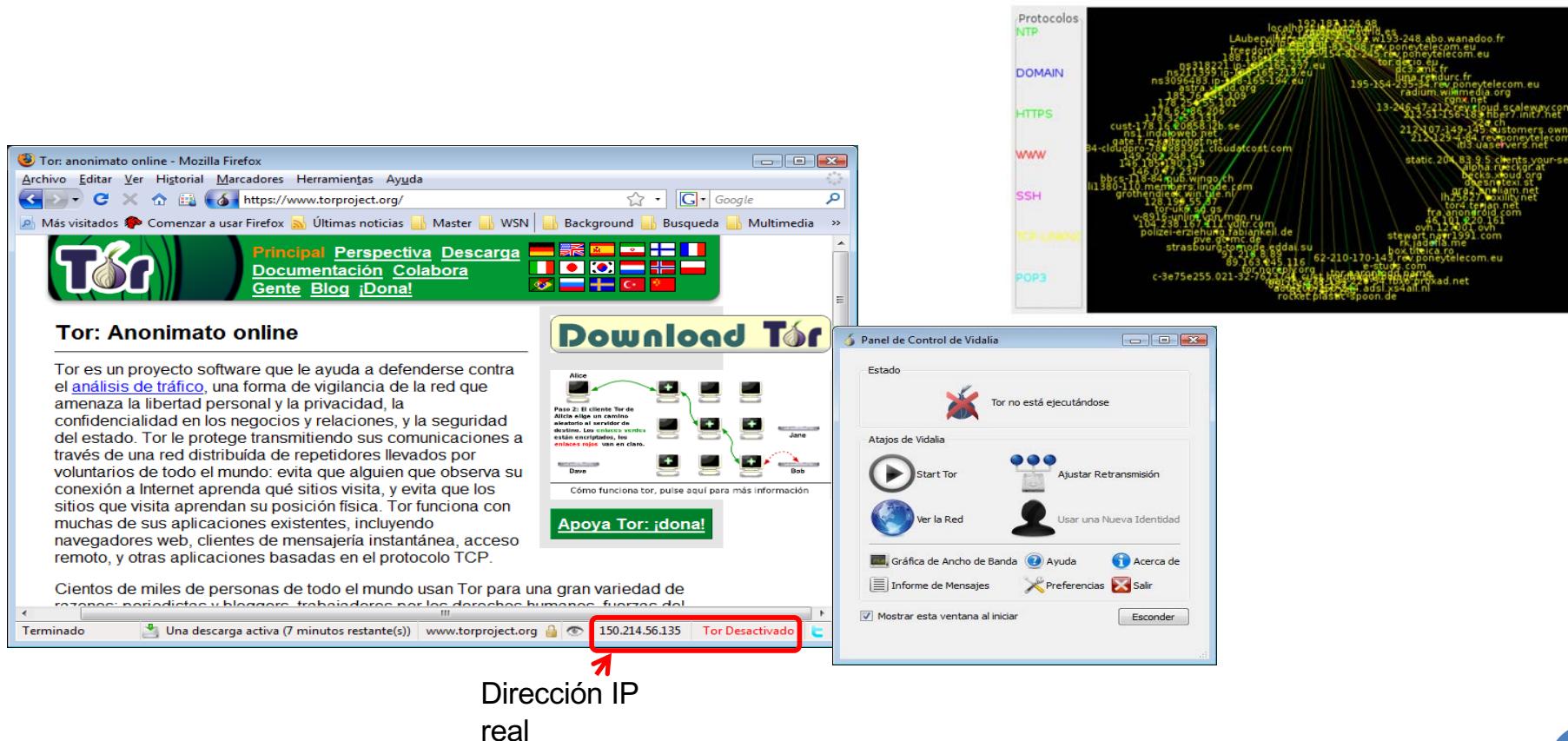
- **Onion Routing:** el onion proxy crea un paquete cifrado en capas, que se irán “pelando” a medida que atravesese el camino de onion routers



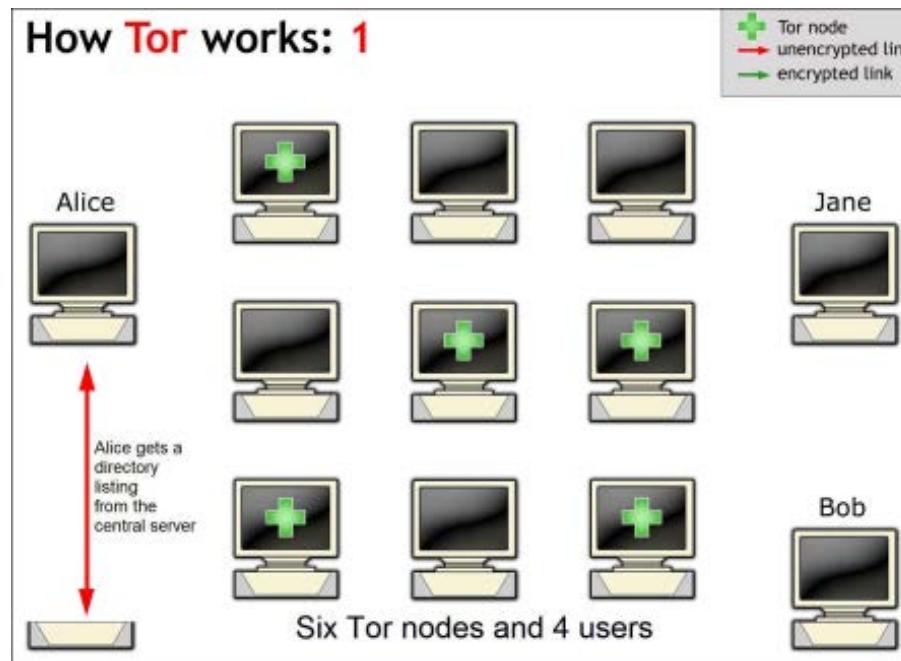
Conoce la ruta que va a
atravesar el paquete y le añade
un cifrado basado en capas

– TOR: es la segunda generación de Onion Routing

- suple 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

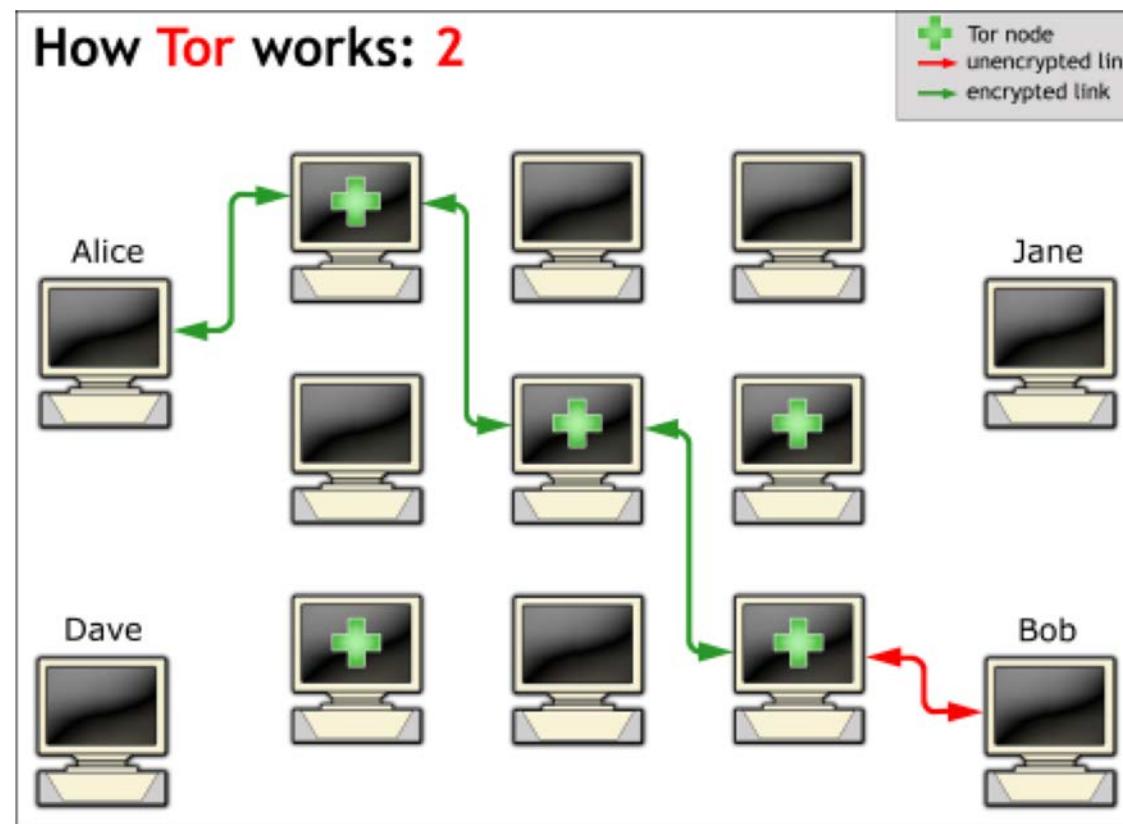


- Funcionamiento:
 - Enrutamiento → los paquetes se envían a través de varios routers onions, los cuales son elegidos de forma “aleatoria” y previamente por un “servidor central”
 - Todos los nodos Tor son elegidos al azar y ningún nodo puede ser utilizado dos veces

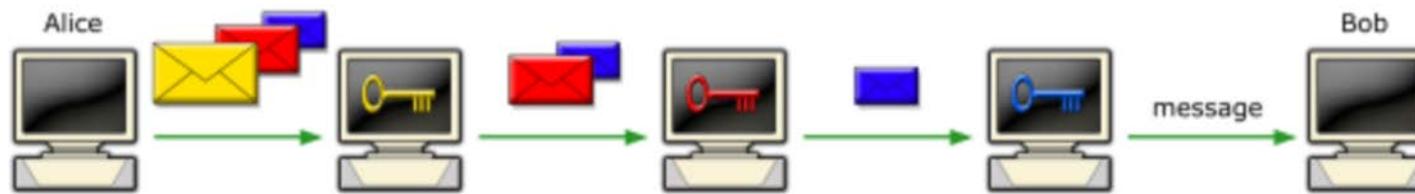


<https://geekytheory.com/que-es-y-como-funciona-la-red-tor>

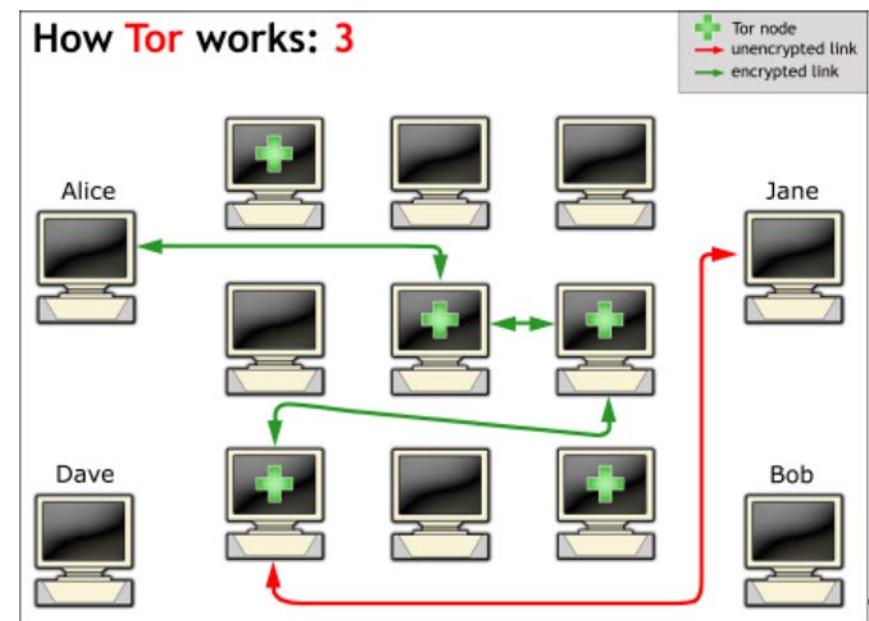
- Se conecta a un nodo aleatorio a través de una conexión cifrada
 - Una vez que el camino ya es conocido desde el origen, cada conexión y salto en la red deberá ser cifrada, excepto con el penúltimo nodo de la comunicación, el cual hará una conexión NO cifrada



- El enruteado 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

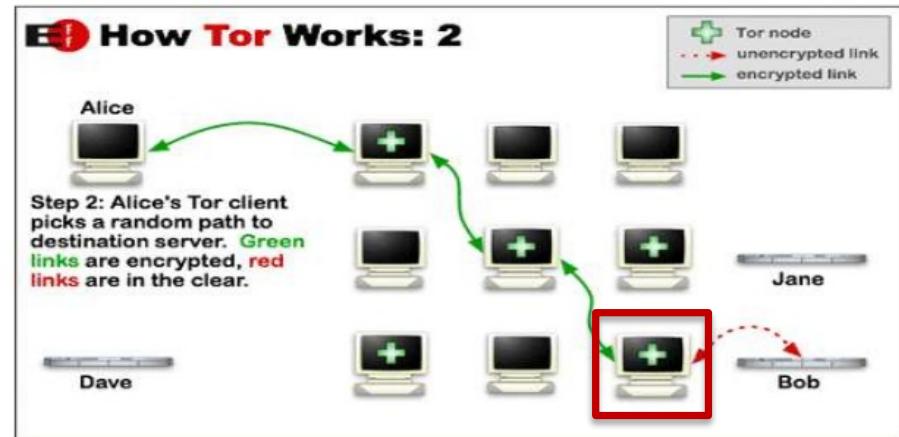


- Para evitar el **análisis pasivo**, cada 10 minutos se cambian los nodos de la conexión Tor, escogiendo nuevos nodos



- Tor:
 - Es lento:
 - Por su arquitectura:
 - conectividad basada en routers y salto de router en router usando criptografía
 - Garantiza el anonimato, pero no garantiza la "privacidad de los datos"... ¿por qué?

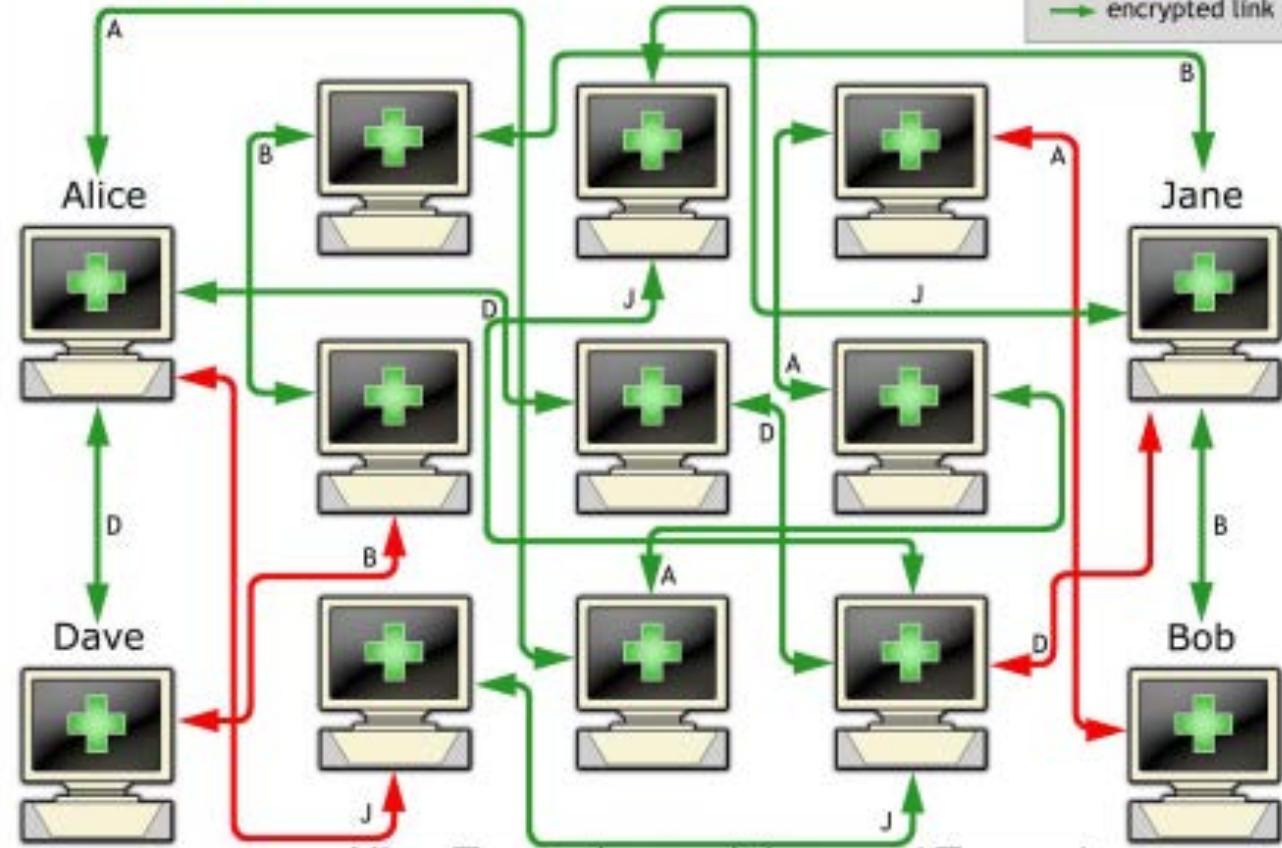
Tor in brief – 2/3



Step 2: Alice's Tor client picks a random path to destination Server. Green links are encrypted, red links are in the clear.

How Tor works: 4

 Tor node
— red link — unencrypted link
— green link — encrypted link

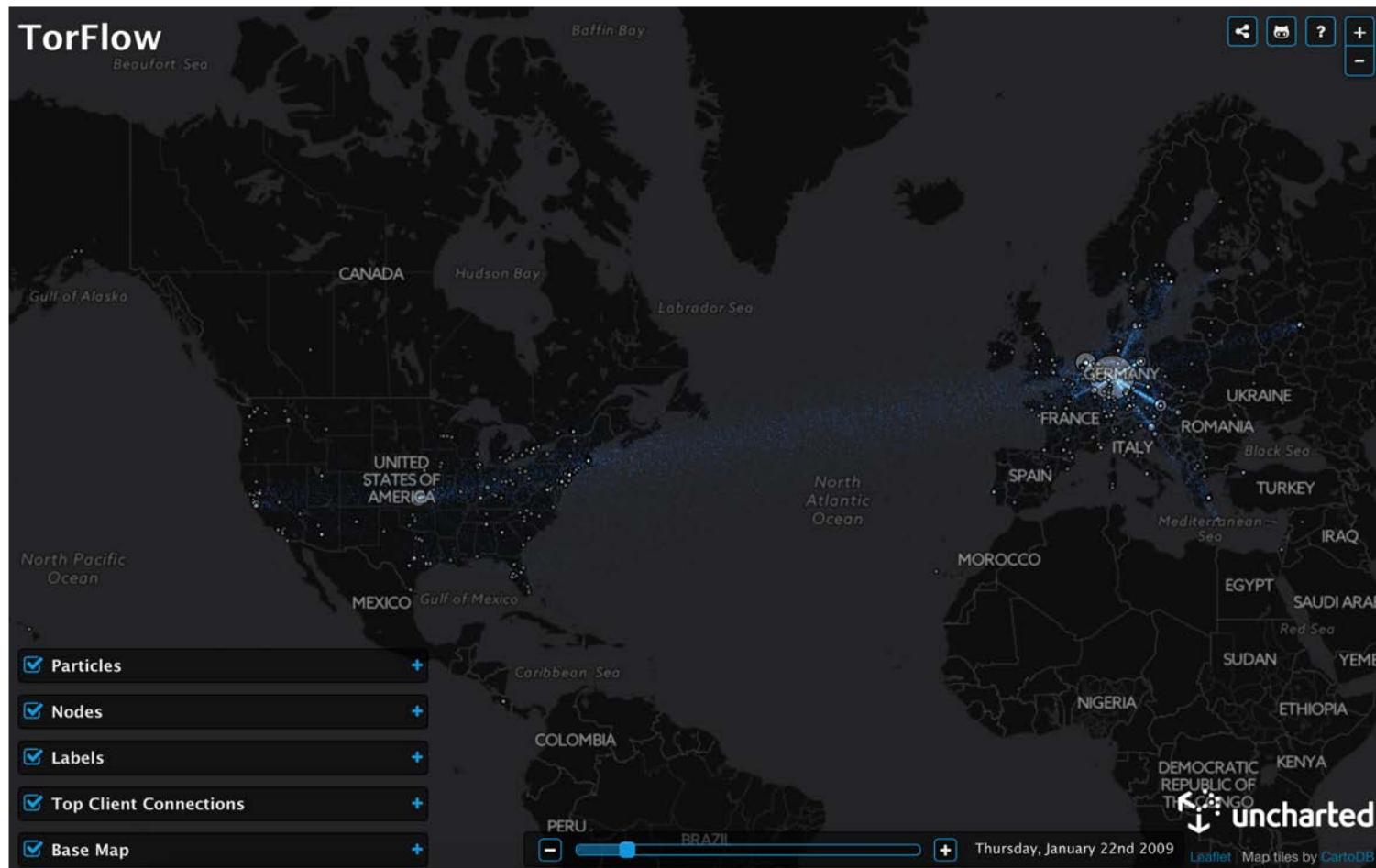


Nine Tor nodes and 4 users / Tor nodes

A: Alice connects to Bob - **B:** Bob connects to Dave

J: Jane connects to Alice - **D:** Dave connects to Jane

- Torflow: <https://torflow.uncharted.software/#/2009-6-11?ML=-15.8203125,34.813803317113155,3>



- Torflow en España:

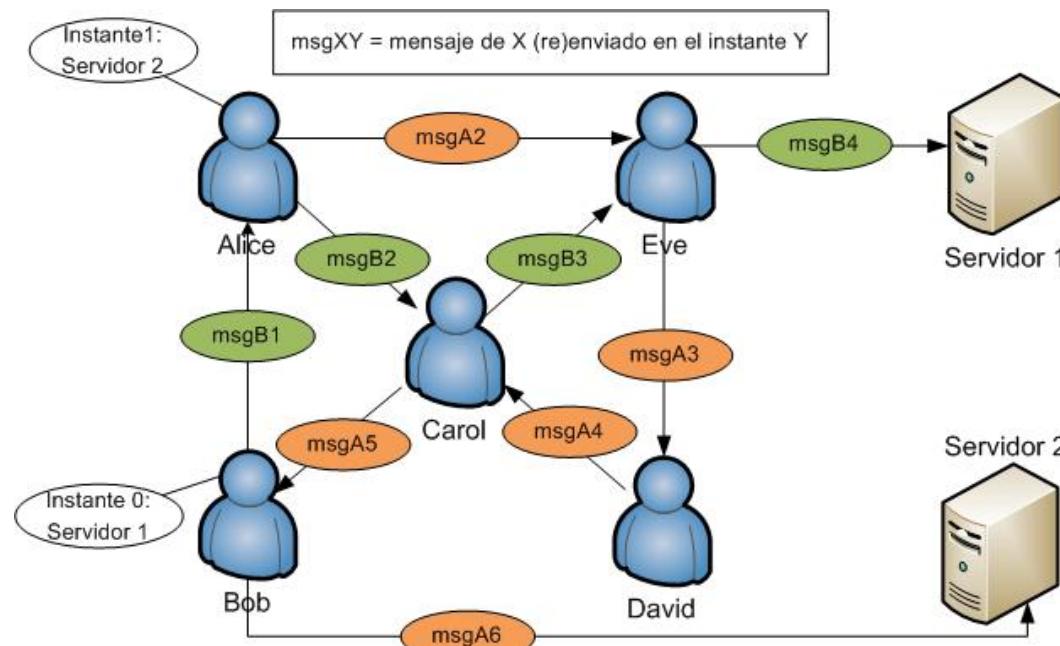


- Torflow en Estados Unidos:



d) Soluciones 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 por un tiempo para saber enrutar de vuelta posibles respuestas



- **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** frente a Crowds
 - sin embargo, es **menos robusto** que Crowds ya que el mensaje de broadcast da ciertos indicios sobre la localización del usuario

e) Soluciones híbridas

- **Tarzan:** es una solución que combina la creación de grupos con el uso de onion routing
 - Cuando un nodo quiere enviar un paquete elige un camino dentro del grupo y le aplica un cifrado sucesivo con las claves simétricas de cada nodo del camino
 - Estos al descifrar serán capaces de determinar el siguiente salto
 - Al final, un nodo especial se encarga de sustituir la dirección de cada nodo por un identificador del grupo y viceversa

	Main goal	Architecture	Techniques							
			SK	PK	LE	PD	PR	FT	MB	
Single-proxy	Sender Anonymity	Centralized	✓							
Mix-nets	Unlinkability			✓	✓		✓			
Onion routing			✓	✓	✓			✓		
Tor			✓	✓	✓					
Crowds	Sender Anonymity	Decentralized	✓							
Hordes	Unobservability		✓	✓					✓	
GAP			✓	✓			✓	✓		
DC-nets	Unobservability		✓						✓	
Herbivore			✓	✓					✓	

- SK/PK: Symmetric/Public-key crypto
- LE: Layered encryption
- PD/PR/FT: Packet delay/replay/injection
- MB: Multicast/Broadcast communications

REFERENCIAS BIBLIOGRÁFICAS

Bibliografía básica

- “*Cryptography and Network Security: Principles and Practice*”
William Stallings
Prentice Hall, 2010 (5^a edición)
- “*Electronic Payment Systems for E-commerce*”
Donal O’Mahony
Artech House, 2001 (2^a edición)
- “*Blind Signatures for Untraceable Payments*”
David Chaum
- “*Group Signatures*”
David Chaum, Eugtne van Heyst

Referencias

- “*The International PGP Home Page*” <<http://www.pgpi.org>>
- *RFC 3156: MIME Security with OpenPGP*
- *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*
- *SET Secure Electronic Transaction Specification (V1.0)*, Books 1, 2 and 3.