Seguridad de la Información

TEMA 4 – PARTE C

Seguridad y Privacidad
En Aplicaciones Telemáticas

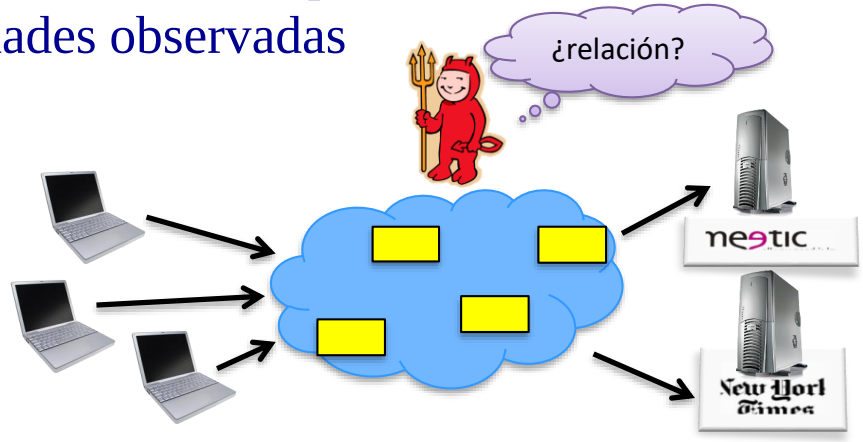
Privacidad de los Usuarios en Aplicaciones

Conceptos generales

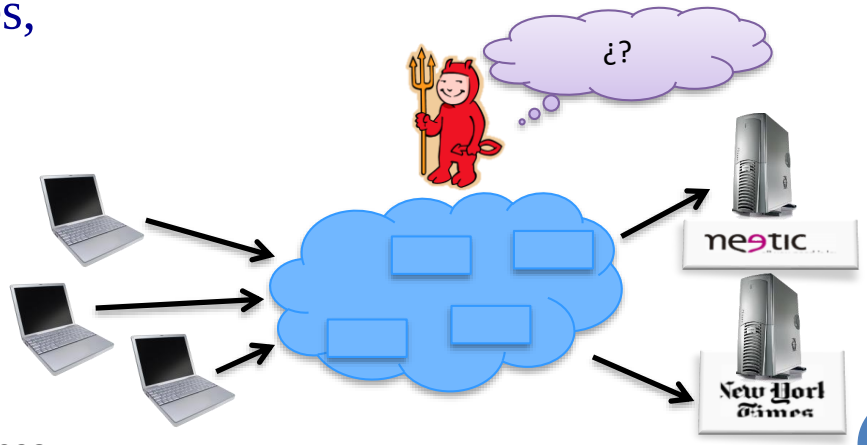
- La **privacidad** puede definirse como:
  - el derecho de los individuos y entidades de proteger, salvaguardar y controlar el acceso, almacenamiento, distribución y uso de información sobre su "**propia persona**"
  - confidencialidad no es equivalente a privacidad
    - la confidencialidad es relativa 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
    - Identidad,
    - Localización,
    - Preferencias,
    - Rutinas
    - …

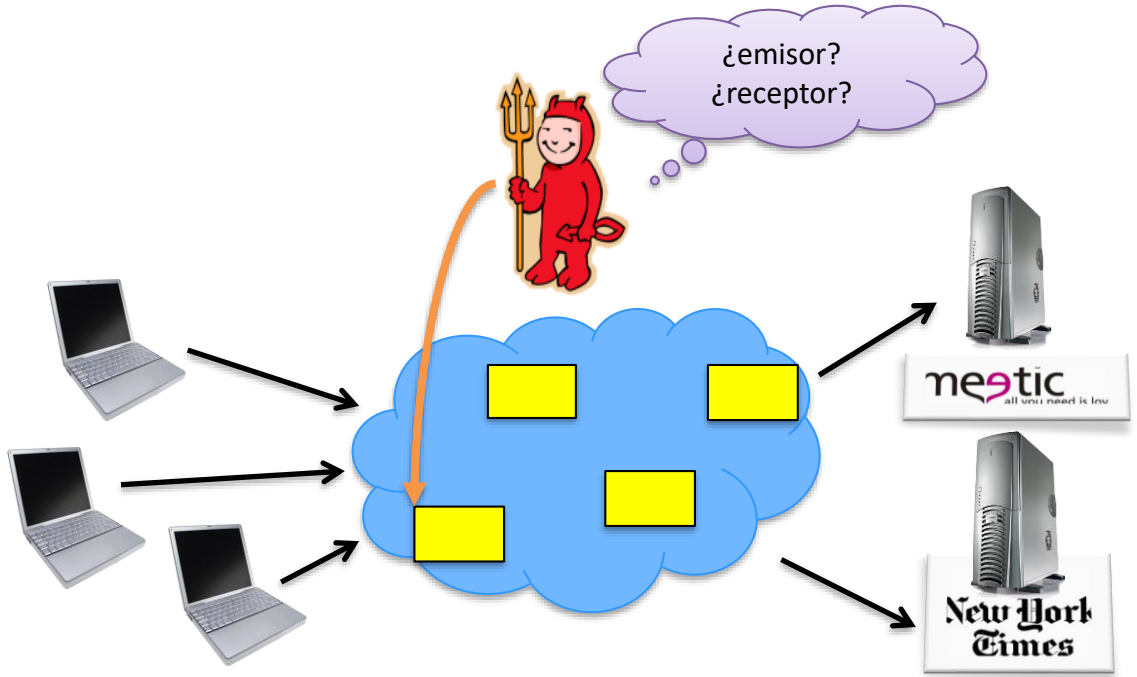
- **Propiedades** de la privacidad:
  - **No-vinculación** (unlinkability): se refiere a la incapacidad de un atacante para relacionar dos mensajes o entidades observadas



  - **No-observabilidad** (unobservability): se refiere a la imposibilidad de distinguir la presencia de mensajes,
    ni la identidad de las entidades



- Estrechamente relacionado con el concepto de privacidad, está el "concepto" de **anonimato**:
  - "*Es el estado de no ser identificable 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



- 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**:
    - Se centra en 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 (ej. el comprador)

  - **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 garantiza que la identidad de los usuarios no se puede revelar, 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 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

- A partir del concepto básico de firma digital estudiado en temas anteriores, surgen esquemas más avanzados de firma, con objetivos 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 – anonimato rastreable ni vinculante
  - **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 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 (ej. dinero digital o 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 *Alice* sea firmado por *Bob* sin que éste 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*
  - Sea *n* el módulo RSA
  - Y sea *r* un número aleatorio *mod n* (*r* es el *blinding factor*)

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

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 $(p_1, p_2, \ldots, p_L)$ de valores primos relativos a e
  - Sea n el módulo compuesto

    $$n = p_1 \cdot p_2 \cdot \ \ldots \ \cdot p_L$$

  - 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:

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

- 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. Se realiza probando:

  $$S_1 = M^{d1} (\bmod n_1), S_2 = M^{d2} (\bmod n_2), \ldots, S_N = M^{dN} (\bmod n_N)$$

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

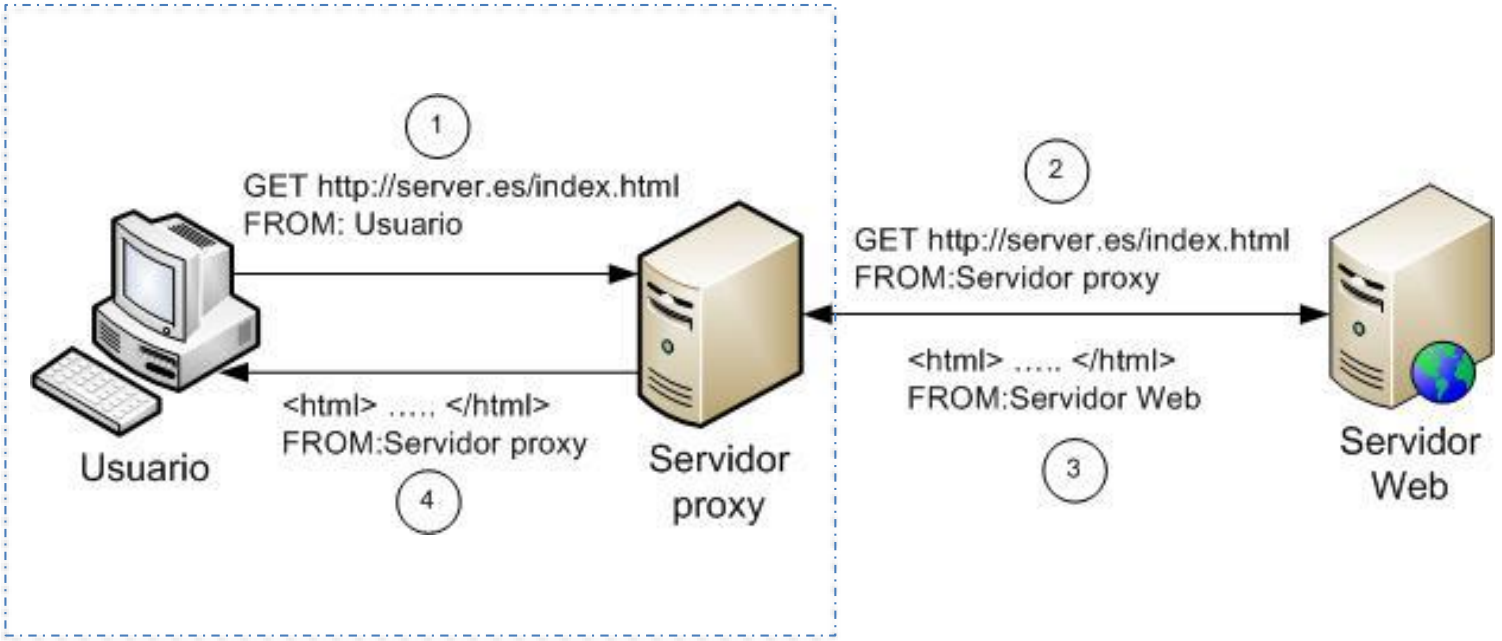
  b) uso de mezcladores

  c) conocimiento parcial de la ruta

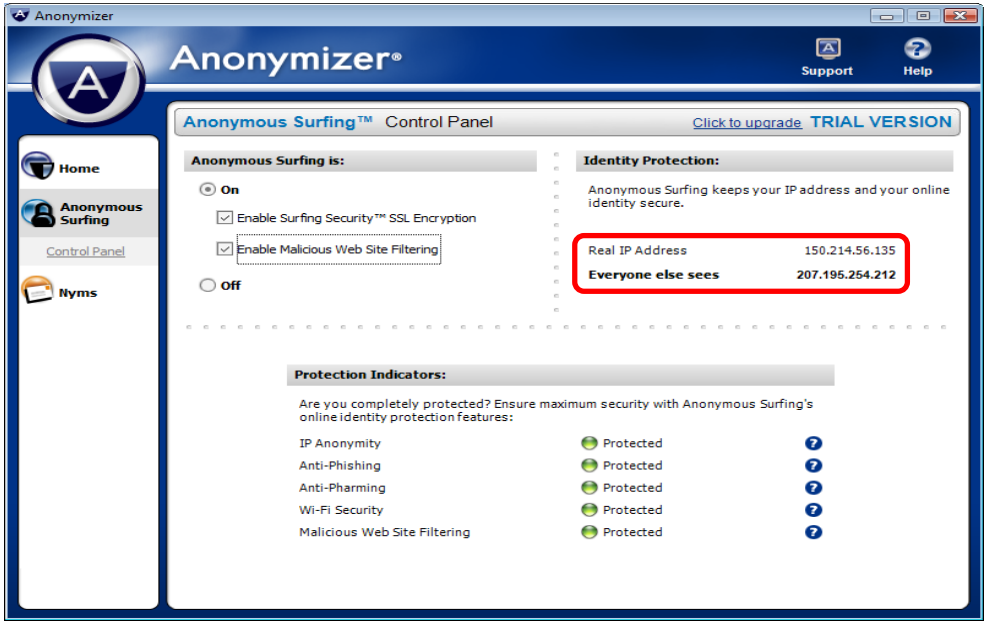
  d) creación de grupos

  además 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



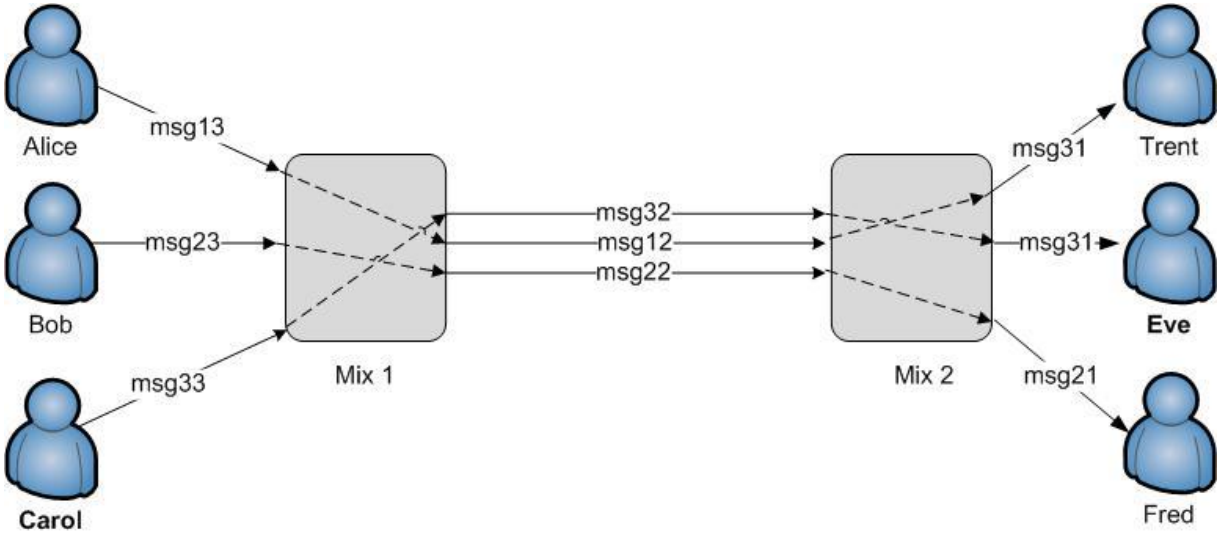
- LPWA *(Lucent Personalized Web Assistant):* elimina cualquier información de identidad de los datos del usuario y crea una conexión SSL entre el usuario y el servidor proxy

- 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.snoopblocker.com
  - Proxify http://proxify.com
  - Anonymouse http://anonymouse.org
  - Web Warper http://webwarper.net

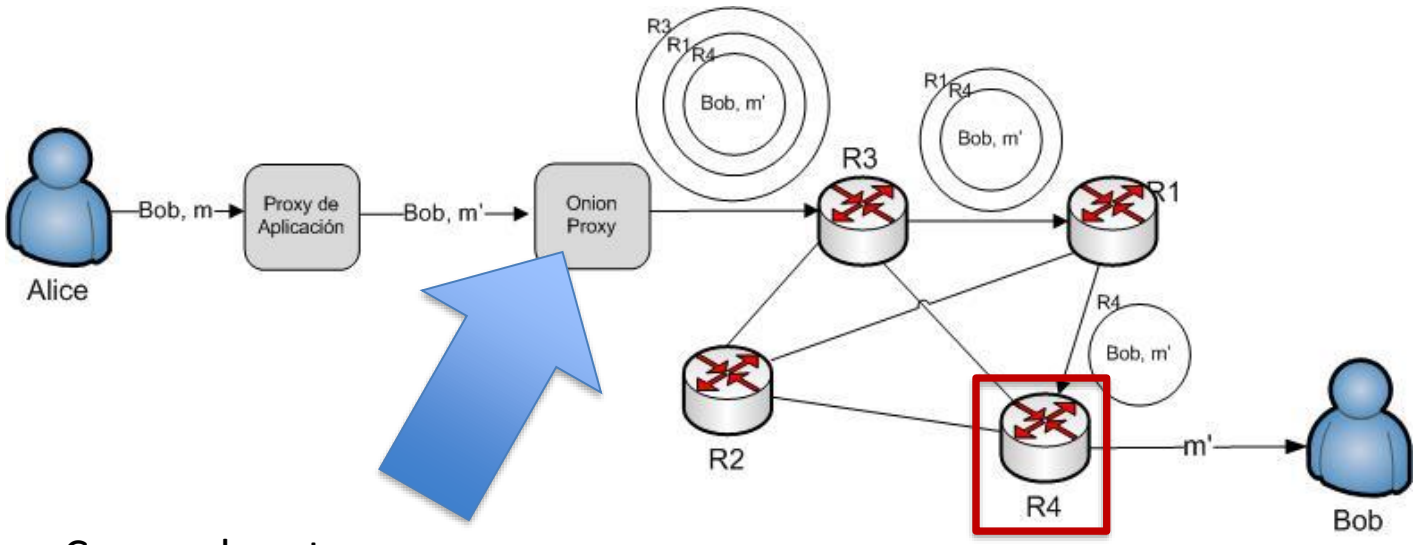
b) Soluciones basadas en el uso de mezcladores (mixers)

- 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



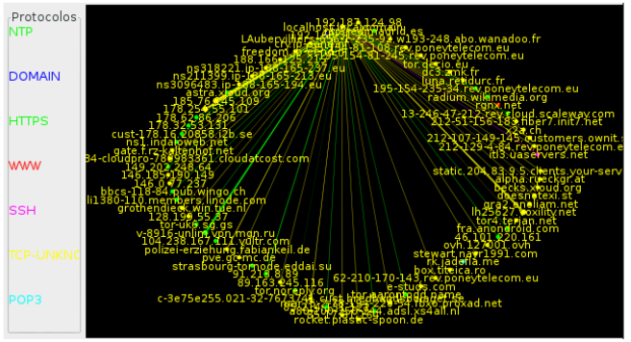
c) Soluciones basadas en el conocimiento parcial de la ruta

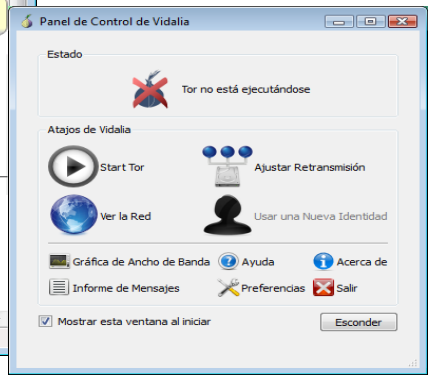
- **Onion Routing**: el onion proxy crea un paquete cifrado en capas, que se irán "pelando" a medida que atraviese el camino de onion routers



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

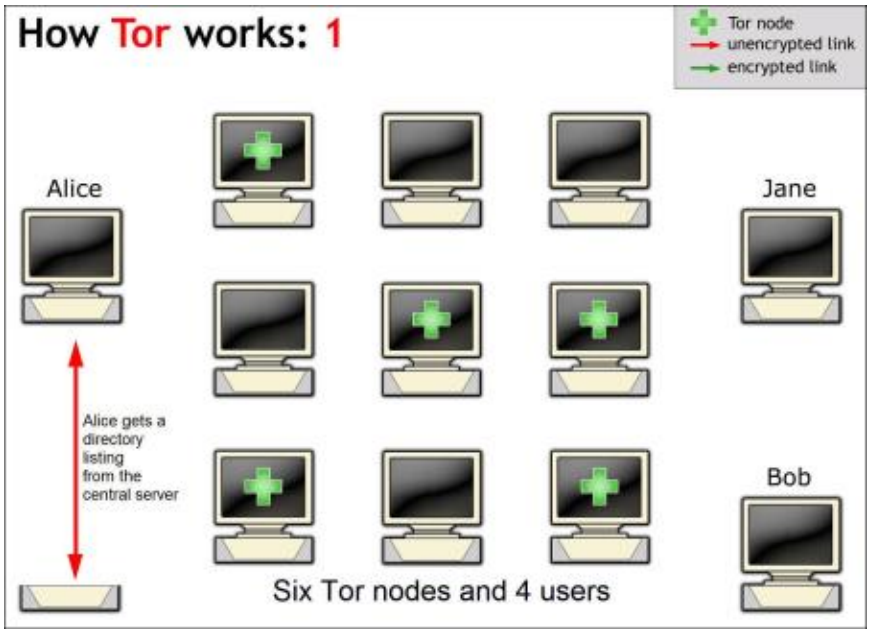
- **TOR**: es la segunda generación de **Onion Routing**
  - evita algunas deficiencias del diseño original, añadiendo **control de congestión**, **comprobación de integridad**, etc.
  - permite navegar a través de una ruta privada de la red TOR, creada de manera aleatoria entre los distintos repetidores (onion routers) de la red





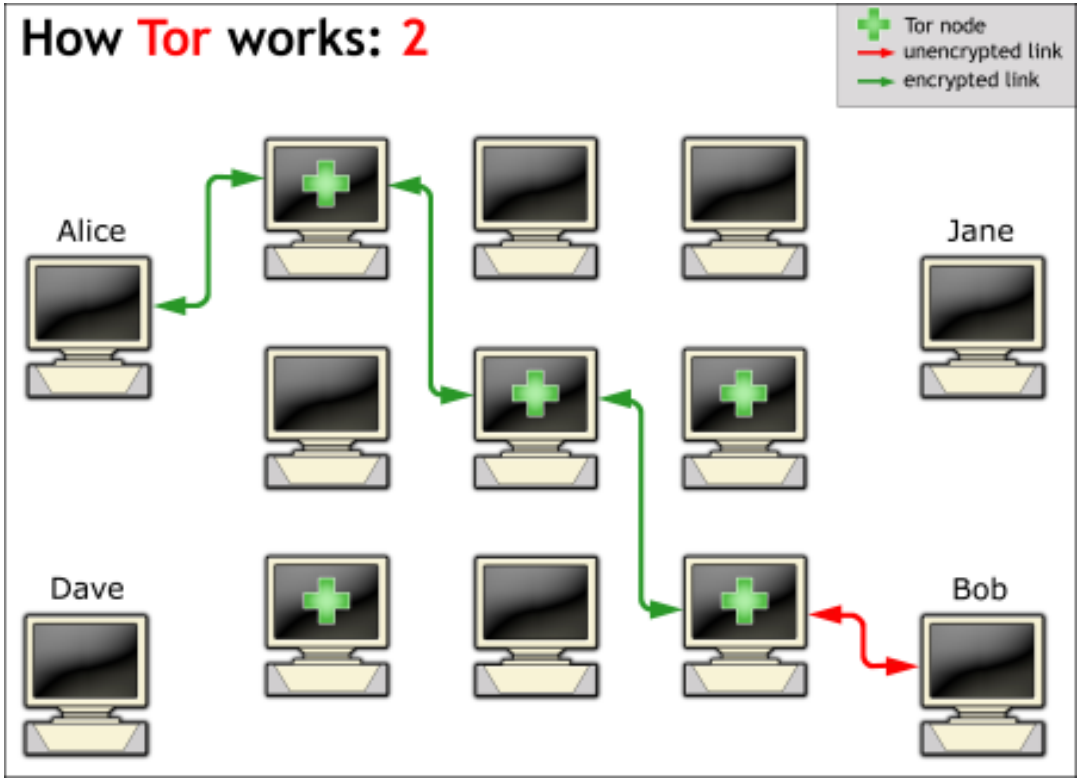
Dirección IP real

- 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



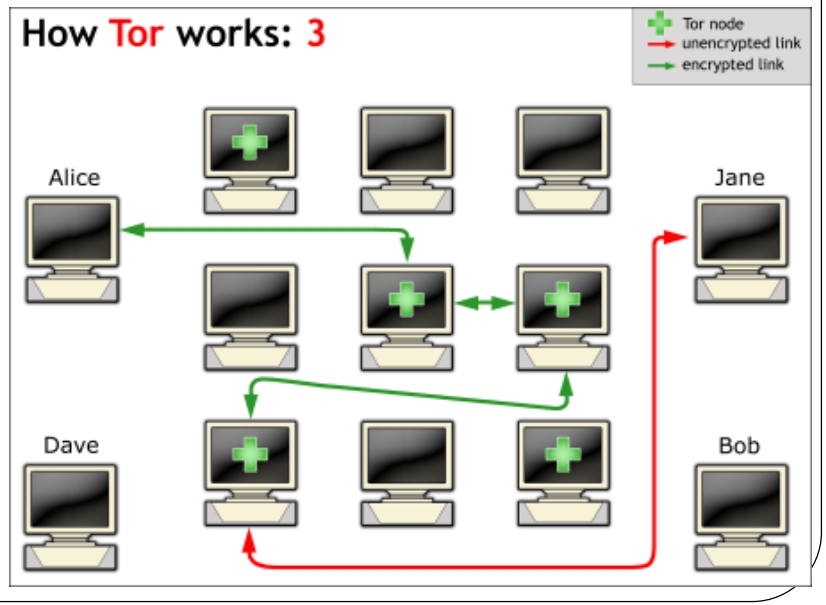
https://geekytheory.com/que-es-y-como-funciona-la-red-tor

- 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



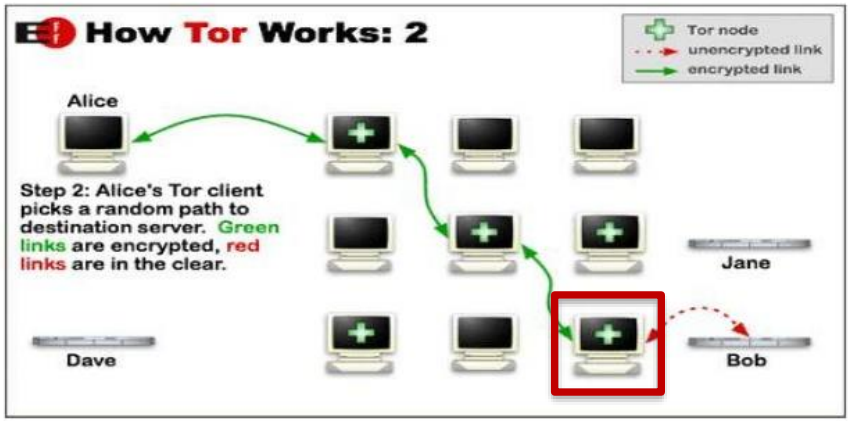
- El enrutado y el cifrado es “asimétrico” (Onion Routing):
  - Ej.: Alice lo primero que hace es cifrar el mensaje con la clave pública del último router onion de la lista, para que éste último lo pueda descifrar; y así con todos los demás

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

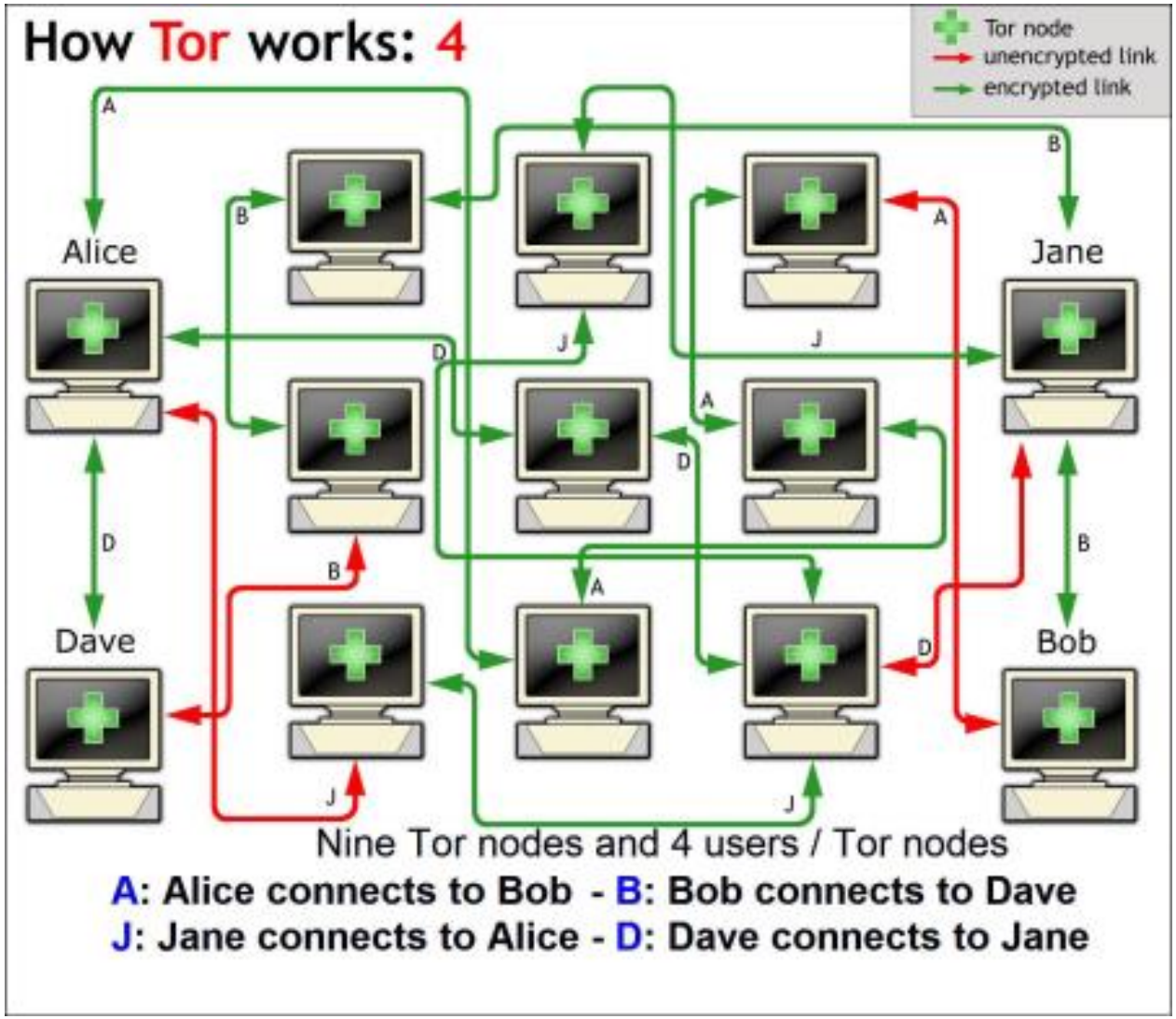


- Tor:
  - Es lento por su arquitectura:
    - conectividad basada en routers y salto de router en router usando criptografía
  - Garantiza el anonimato, pero no garantiza la ”privacidad de los datos”… ¿por qué?

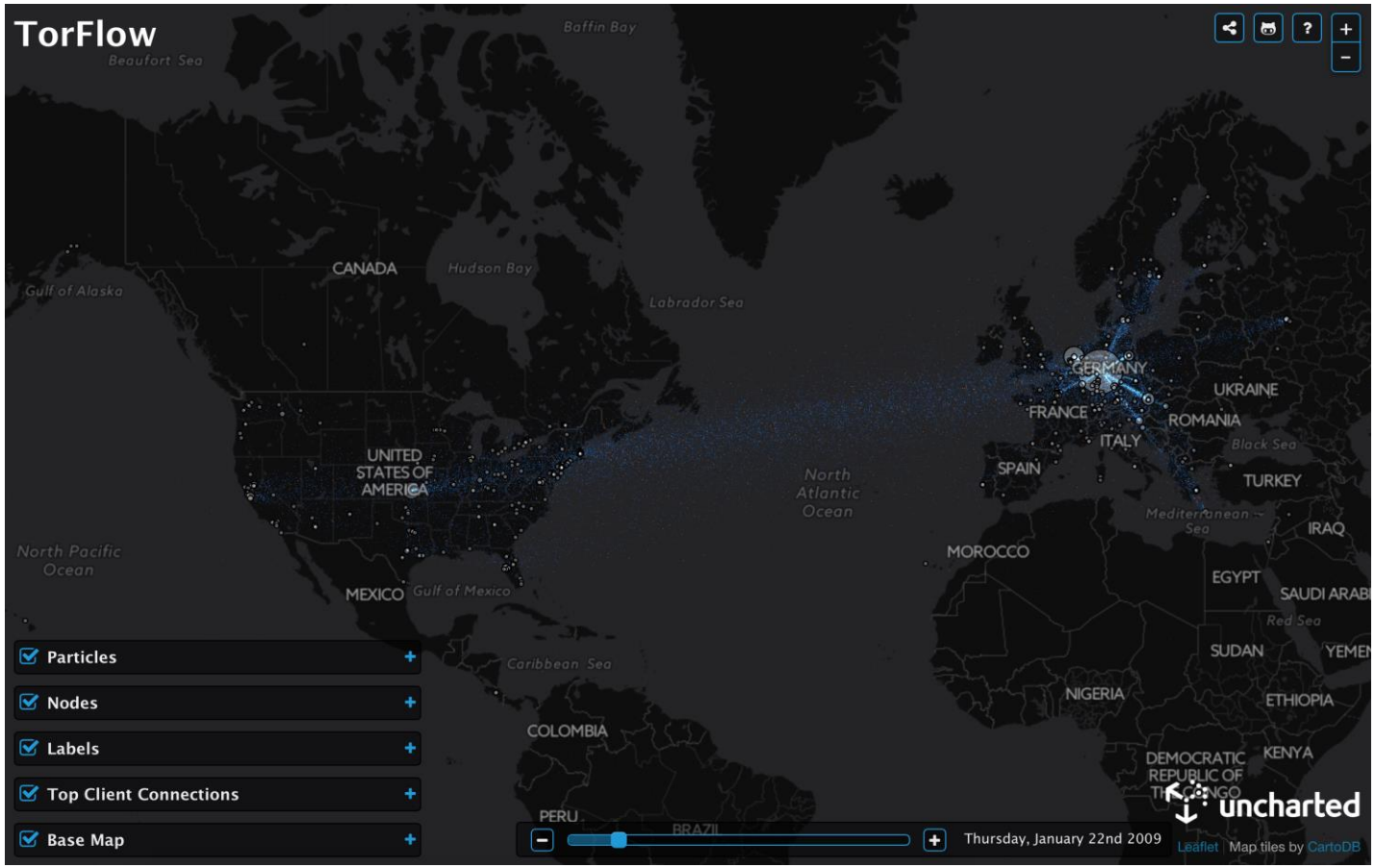
Tor in brief – 2/3



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
unencrypted link
encrypted link
Alice
Jane
Dave
Bob
A
B
D
J
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

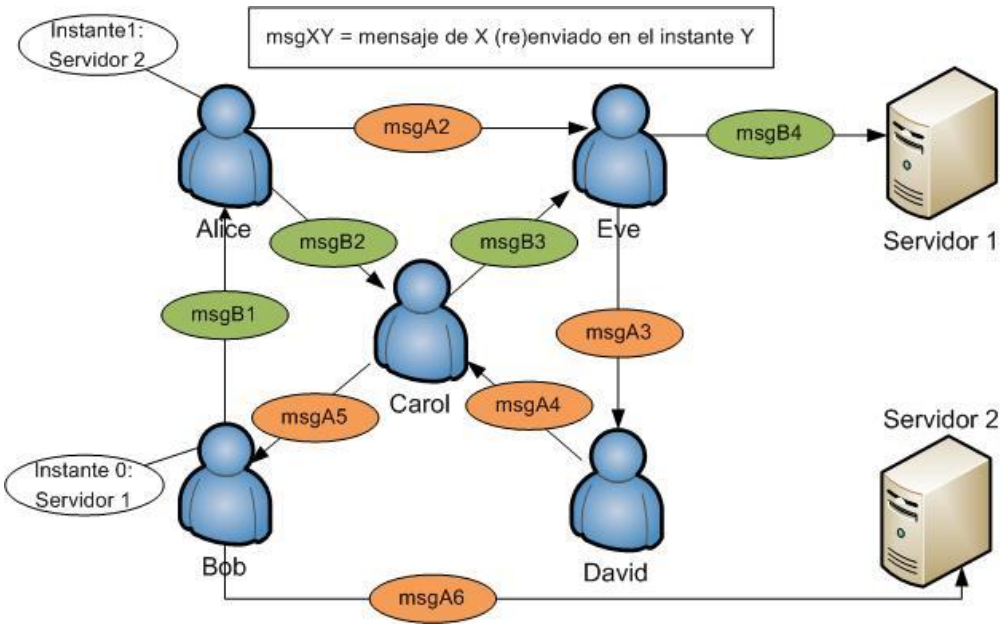


- Torflow en España:

- 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 temporalmente en cada nodo en el que se transita para saber enrutar de vuelta (con la respuesta)



- **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 (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

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			√	√					√
GAP	Unobservability		√	√		√	√	√	
DC-nets			√						√
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 (5ª edición)
- *"Electronic Payment Systems for E-commerce"*
  Donal O'Mahony
  Artech House, 2001 (2ª 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.