

Anonimato in Rete

Definizioni

- **Anonimato**

Lo stato di non identificabilità di uno specifico soggetto in un insieme di utenti.

Ha essenzialmente due finalità:

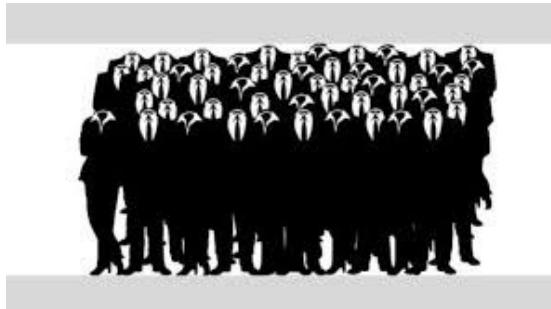
- **Nascondere la reale origine** di un attacco o di una specifica azione effettuata attraverso la rete
- **Garantire la Privacy**

Diritto di poter scegliere cosa della nostra vita privata può essere divulgato (tracciamento, e-voting etc.)



Finalità

- Persone che vogliono aggirare la **censura** statale
- Persone che vogliono partecipare in modo anonimo a discussioni dagli **argomenti sensibili** o che non vogliono subire l'opinione degli altri
- Aziende che non vogliono pubblicizzare alcune relazioni
- Persone che non vogliono essere oggetto di **profiling** commerciale
- Persone che vogliono offrire servizi senza poter esser localizzati
- Autorità giudiziare che non vogliono esser identificate come tali durante attività investigative



Ma anche..

- Persone che vogliono organizzare **truffe** online e rimanere impuniti
- Persone che visitano siti dai contenuti illegali, ad esempio pedopornografici
- Criminali che devono comunicare con altri criminali
- **Criminalità organizzata** che vede nuove frontiere di business
- **Tutte le tecnologie** possono esser usate per scopi criminali, ma questo non è un problema tecnico
- L'uso di strumenti di anonimizzazione non introduce nuovi tipi di abusi



Lo scopo è ...

- **La privacy degli individui sia rispettata**
 - Possa essere anonimo se lo decido
 - Essere tecnicamente certo che i miei dati siano visibili solo a chi li concedo con il mio consenso
- **I criminali siano catturati**
 - Gli organi di Pubblica Sicurezza (e solo loro!) abbiano strumenti per identificare i malfattori
 - Abbiano strumenti per rilevare e produrre prove di reati criminosi



Stato attuale...

Il **diritto alla privacy** e' un diritto fondamentale sancito dal **Testo Unico della tutela dei dati personali**. Ma su Internet:

Non esiste privacy

- L'origine e la destinazione di ogni comunicazione e spesso anche i contenuti sono identificabili con tecniche elementari

Non esiste anonimato

- Ogni persona che accede ad internet è identificata e registrata

Nessuno è libero di esprimersi

- La pubblicazione e fruizione di contenuti è facilmente censurabile (vedi Cina e <http://www.rsf.org/24h/map.php>)

Quindi internet è tutt'altro che anonima!

Anonimato in rete

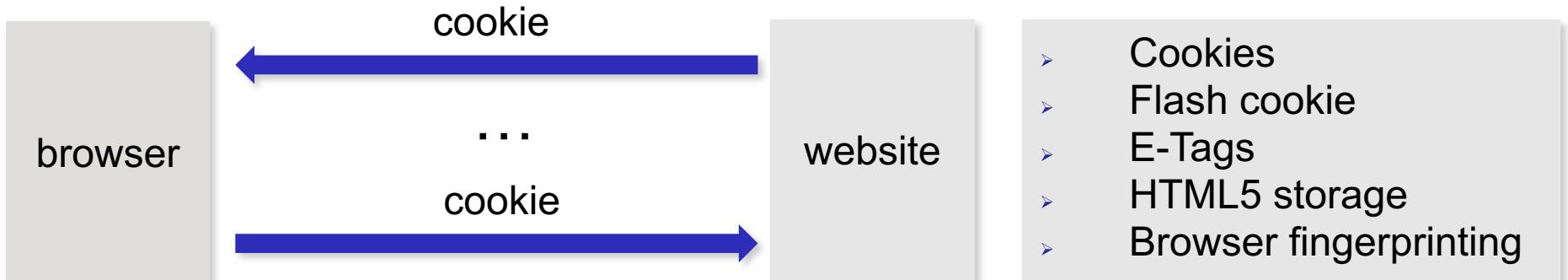
- Quindi, senza usare particolari accorgimenti la rete è in grado di garantire l'anonimato?

Assolutamente no!



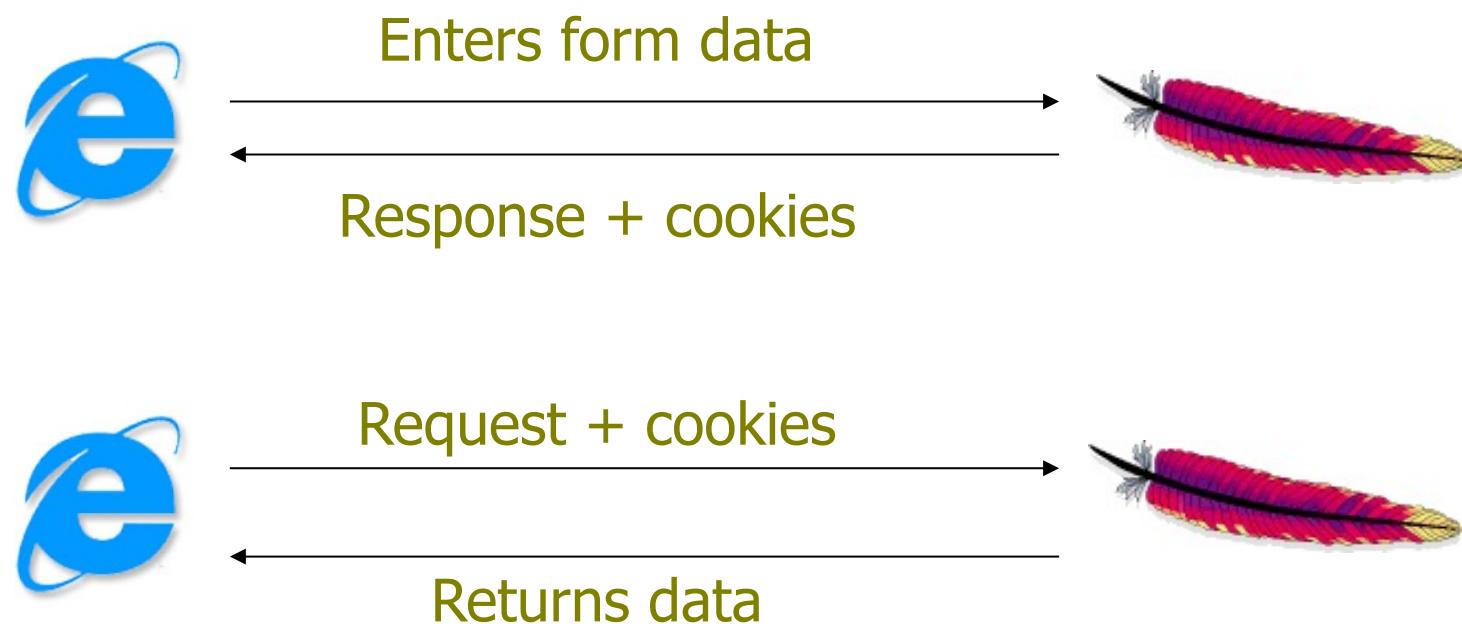
Evidenze digitali in rete

- Indirizzo IP, facilmente tracciabile a ritroso
 - La possibilità di spoofing rende concettualmente anonimo il singolo IP
 - Il routing della rete di appartenenza ci riconduce univocamente alla localizzazione
- Tracciabilità browser
- Profilazione utente



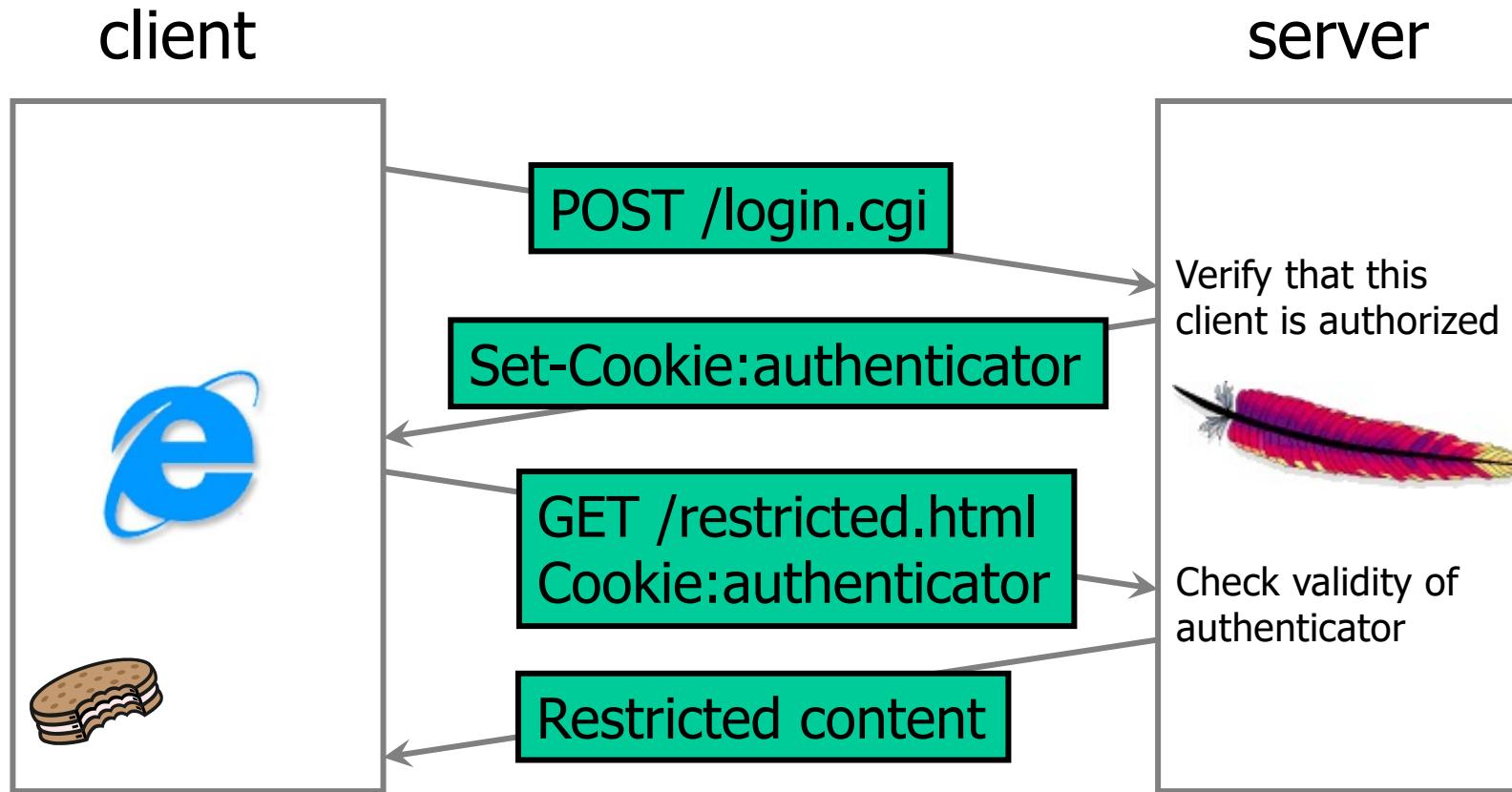
Cookies

- Un cookie è una coppia name/value creata da un web server per conservare informazioni di stato sul computer che ospita il browser
- Solo il server che ha creato il cookie può usarlo



HTTP è un protocollo stateless; I cookies introducono informazioni di stato

Una Sessione con i Cookies



Il cookie è implementato un header aggiuntivo presente in una richiesta (*Cookie:*) o risposta (*Set-cookie:*) HTTP:

- il server per assegnare un cookie, lo aggiunge tra gli header di risposta.
- Il client nota la presenza del cookie e lo memorizza in un'area apposita
- Il browser web client rimanda il cookie, senza alcuna modifica, allegandolo a tutte le richieste HTTP che soddisfano il pattern, entro la data di scadenza.
- Il server può scegliere di assegnare il cookie di nuovo, sovrascrivendo il vecchio.

Cookies

- Possono essere usati dai browsers per le applicazioni web-based a scopo di:
 - Autenticazione
 - Tracking
 - Conservazione di informazioni specifiche dell'utente
- Contengono dati sensibili
- Il cookie è composto da una stringa di testo arbitraria, una data di scadenza (oltre la quale non deve essere considerato valido) e un pattern per riconoscere i domini a cui rimandarlo.
 - Name session-token
 - Content "s7yZiOvFm4YymG...."
 - Domain .amazon.com
 - Path /
 - Expires Monday, September 08, 2031 7:19:41 PM

E-Tags

- L'ETag è un header della risposta HTTP che contiene tipicamente un identificativo o un hash.
 - Viene generato dal server e mantenuto in cache dal client.
 - Quando il client effettua nuovamente una richiesta, invia in allegato l'ETag all'interno dell'header **If-None-Match**.
 - Se il valore corrisponde alla versione corrente sul server, quest'ultimo ritorna una risposta di tipo **304 - Not Modified**, che non contiene alcun body (e pertanto è molto leggera e veloce) e che istruisce il client a mantenere i dati precedenti in cache per un ulteriore TTL.



Scopi: Pubblicità & Guadagno

- **Profiling:** ovvero tenere traccia delle abitudini e costruire un profilo dei consumatori:
 - politico
 - religioso
 - commerciale
 - sessuale
- Agenzie che registrano l'attività e le preferenze di un utente per produrre **pubblicità personalizzata**
- Esiste un mercato di **vendita di dati personali**
- Utilizzo della tecnologia per raggiungere questi scopi (vedi spider, cookies, malware, ...)

Le minacce fantasma



Ogni citazione è puramente casuale

Comunicazioni sicure: Cifratura di flusso

Secure Sockets Layer (SSL / TLS)

- Trasparente all'applicazione
- Autenticazione a chiave pubblica (di server e client)
- Cifratura della sessione
- Integrità dei dati e non ripudiabilità
- Protegge i dati solo nella comunicazione (ma se uno ha accesso all'host...)
- Tunneling https, pops, imaps, smtps, sftp, etc.

http://en.wikipedia.org/wiki/Transport_Layer_Security

<http://tools.ietf.org/html/rfc4346>

<http://www.openssl.org/>



Limiti SSL / TLS

- Protegge i dati solo nella comunicazione (privacy e integrità dei contenuti)
- Non garantisce l'anonimato!
- Vulnerabilità dei browser



L'esperienza di HeartBleed

Basato su una banale vulnerabilità di OpenSSL relativa a blocchi malformati con lunghezza



Historical

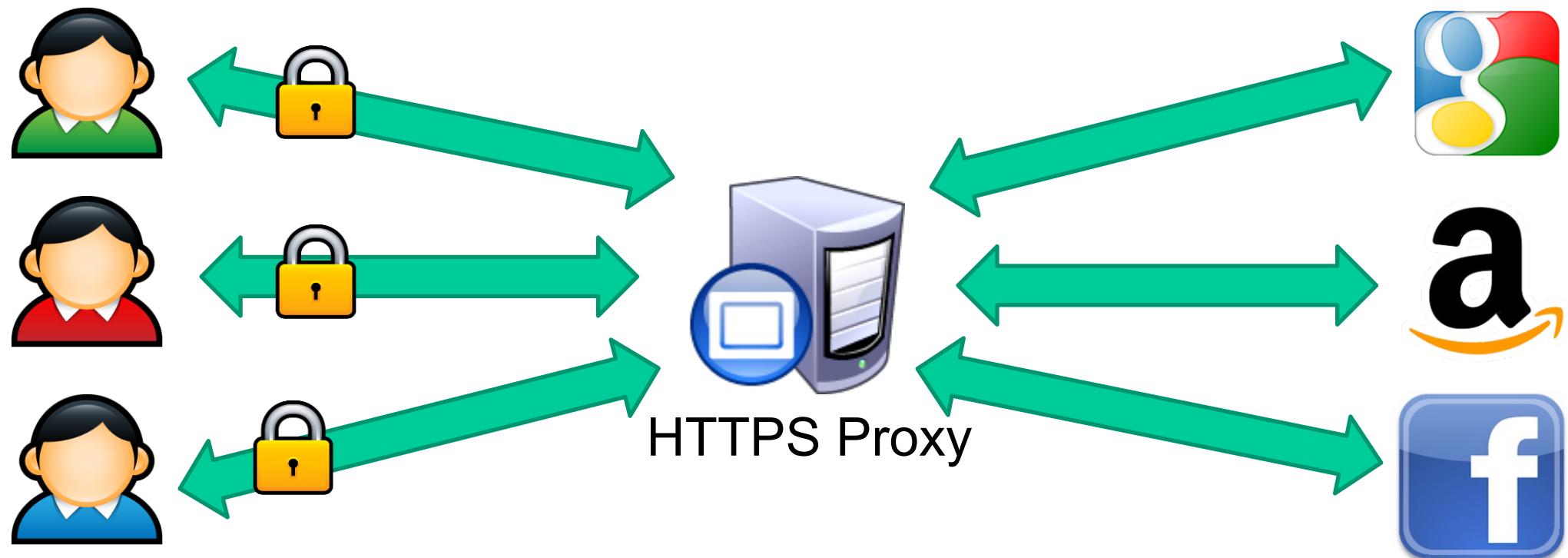
Websites



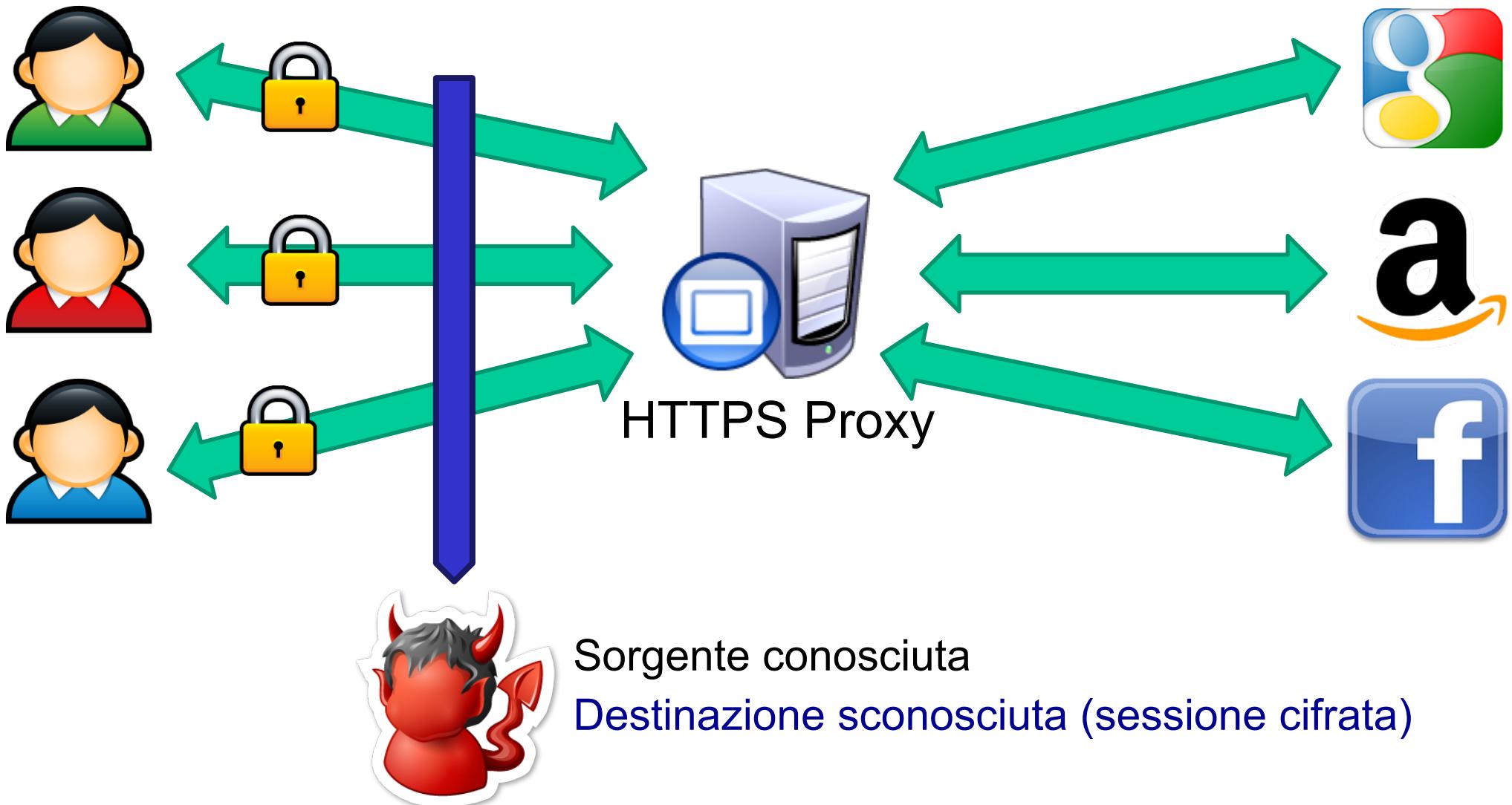
Proxy anonimizzatori



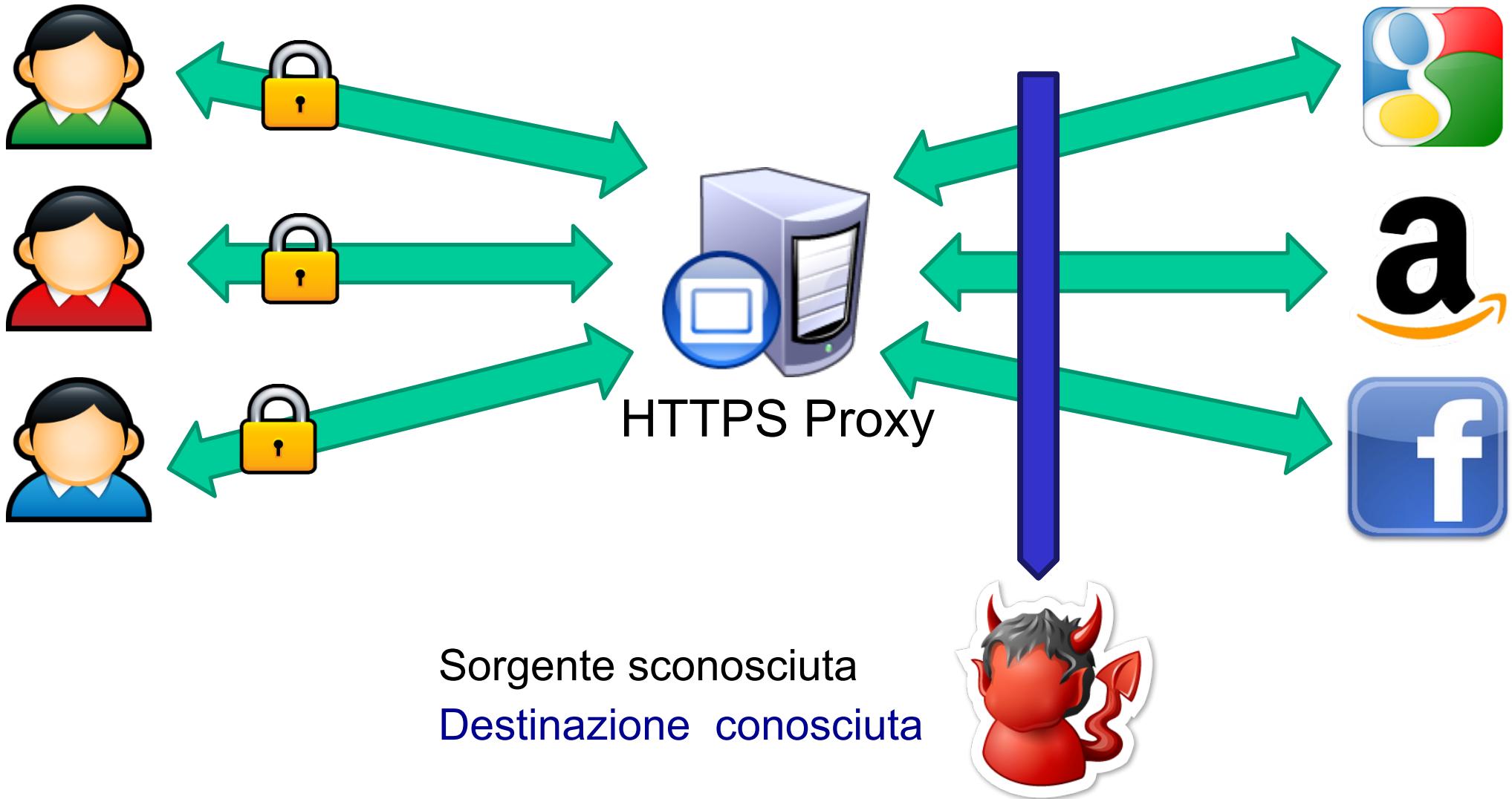
Usiamo un proxy HTTPS per anonimizzare!



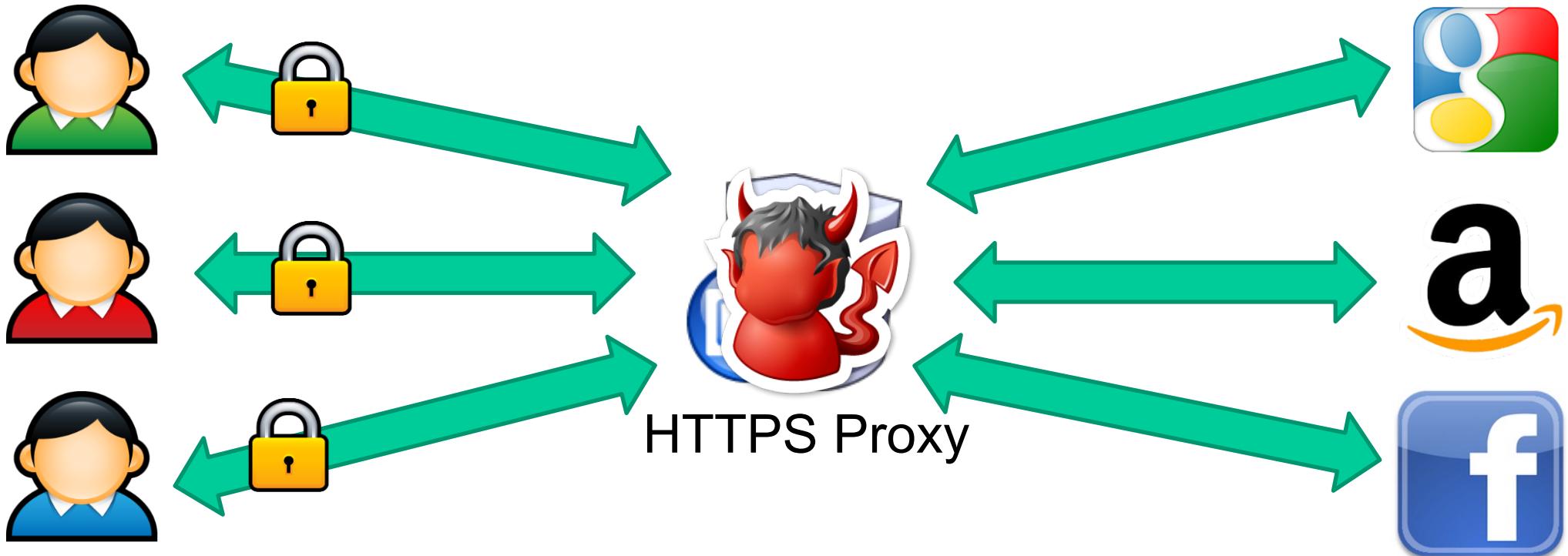
Proxy anonimizzatori



Proxy anonimizzatori

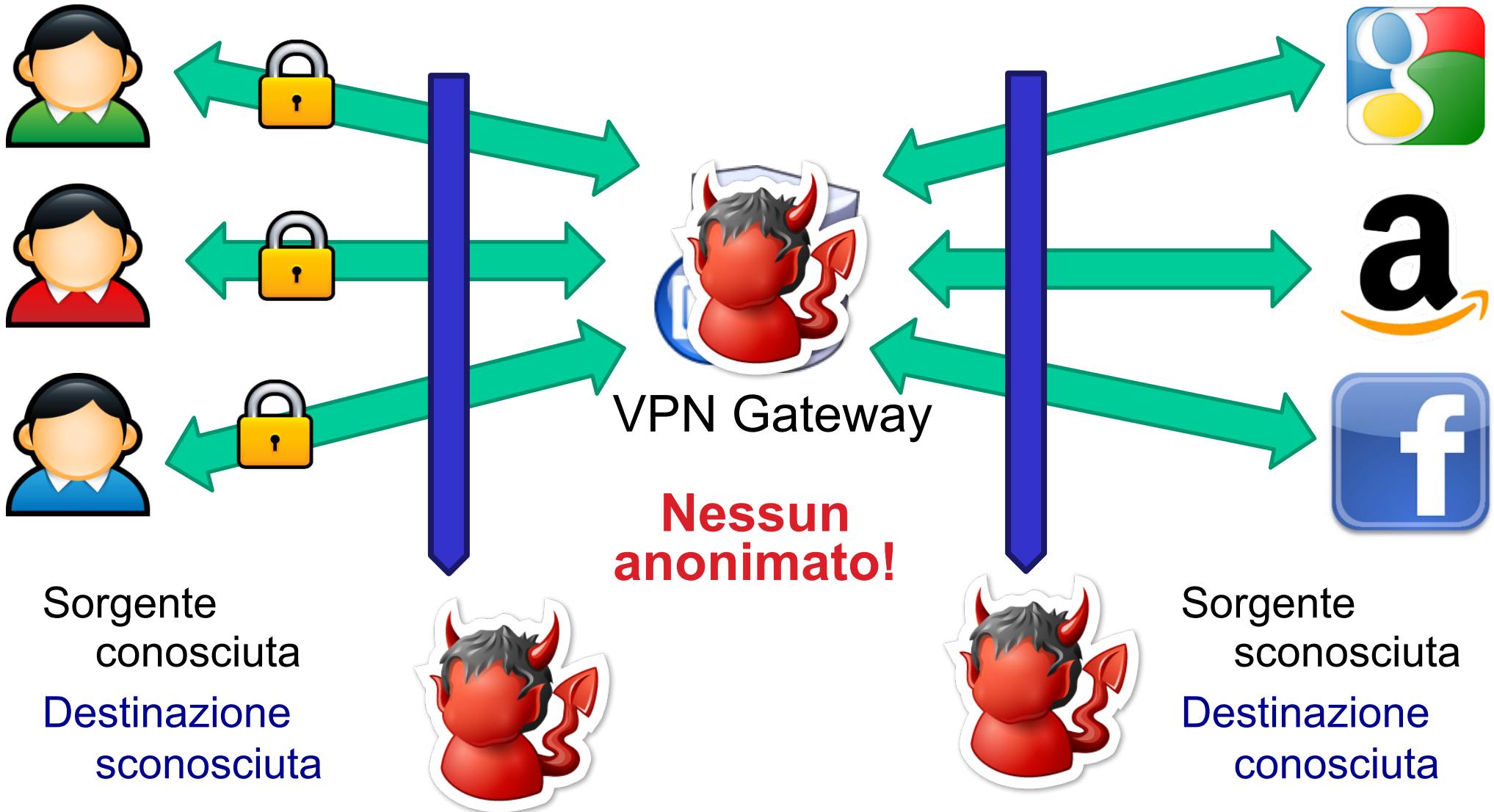


Proxy anonimizzatori

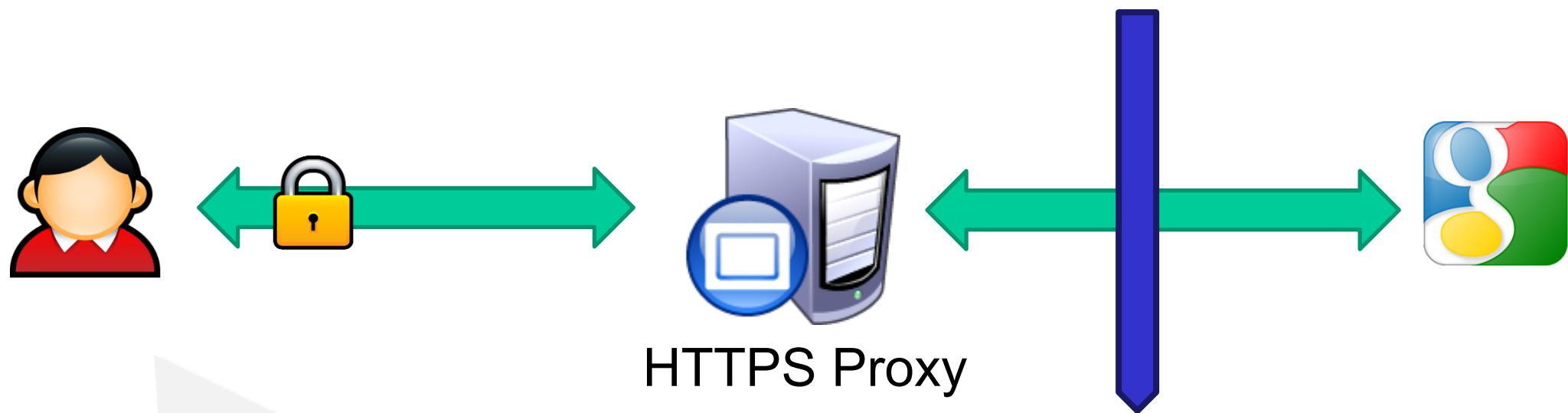


Nessun
anonimato!

Proxy anonimizzatori



Proxy anonimizzatori: Contenuto



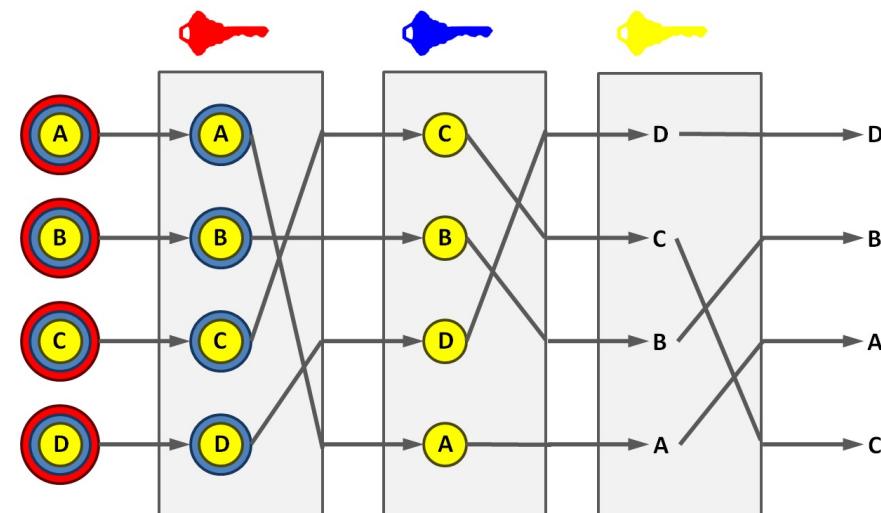
- Gmail (lettura/invio email)
- Google+ (aggiornamento profilo)
- Google Maps (navigazione)



Non è anonimo!

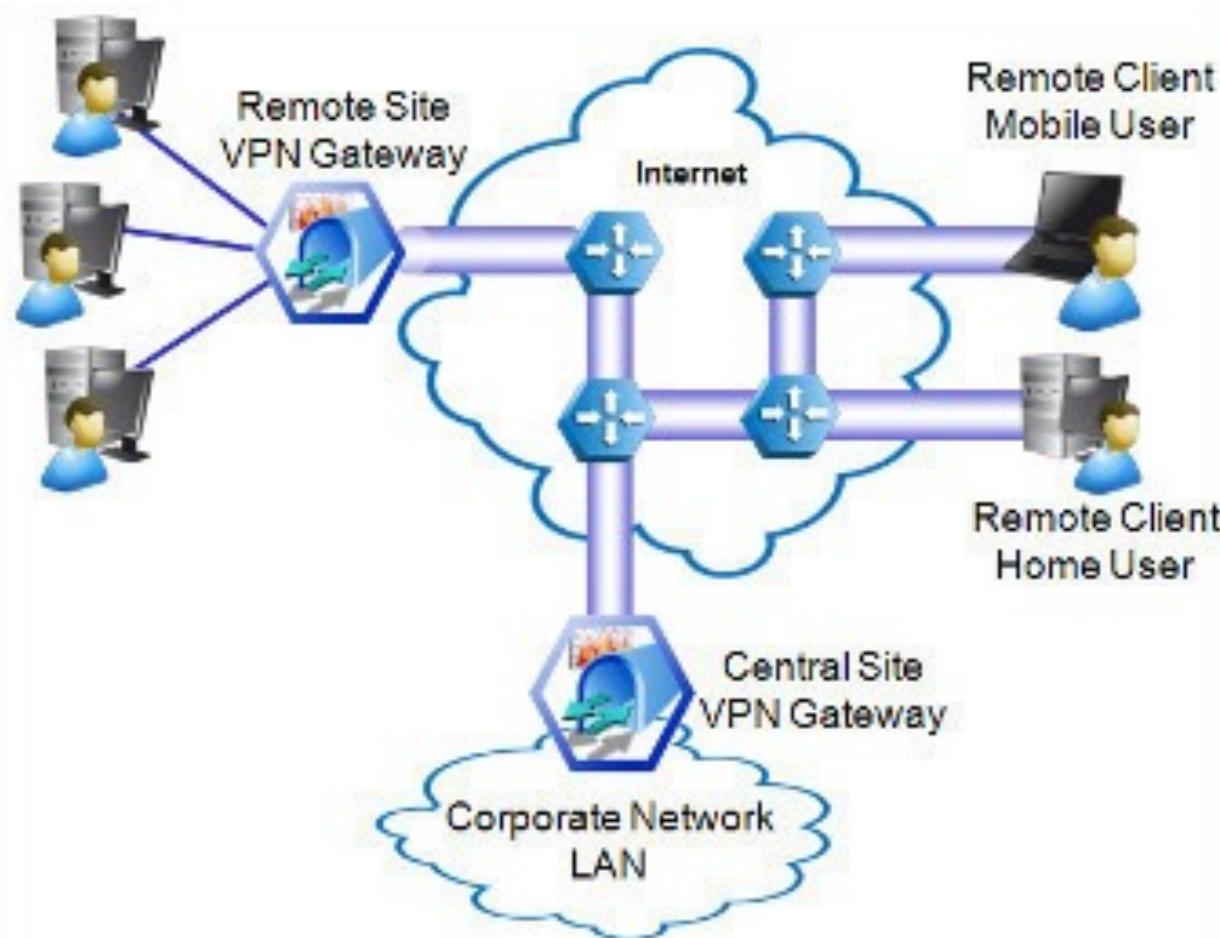
Mix Network

- David Chaum, 1981
- Catena di proxy server (mix)
 - Ricevono un messaggio e lo inviano ad altro mix scelto casualmente
 - Ogni mix sulla path conosce solo il nodo precedente e quello successivo



Proxy anonimizzatori

- Non solo HTTPS proxy
- Anche VPN Gateway



Proxy anonimizzatori: Esempi



NewIPNow.com

Premium, anonymous, multi-ip web browsing

Connect to www.google.com

With the IP

151.80.203.127	(Nord-Pas-de-Calais, France)	<100ms	free
167.114.100.72	(Quebec, Canada)	308ms	free
89.248.164.184	(Noord-Holland, Netherlands)	510ms	free
216.27.27.74	(North Carolina, United States)	529ms	free
85.159.237.152	(NOORD-HOLLAND, NETHERLANDS)	807ms	free

PROXY [Contact Us](#) [FAQ](#) [Advertise Here](#) [Web Proxies](#) [Proxy Forum](#)

THE FASTEST, MOST SECURE VPN AVAILABLE [READ MORE](#) [FREE 30 DAY TRIAL](#)

The proxy list is reordered randomly every 10 minutes to allow each proxy to get some exposure near the top. [Click here](#) to get a listing.

Proxy Lists >> Web Proxy List

Enter a URL to visit via proxy:
 [GO](#)

Choose one of 3,385 working proxies:
(Out of 50,352 total proxy servers)
Last updated: April 17, 2015 at 7:10 AM EDT

[random proxy](#)

newipnow.com (US, GProxy)
proxylist.com (US, Custom, SSL)
proxysite.com (US, GProxy, SSL)
proxy.net (US, Custom, SSL)
proxyvirus.us (US, GProxy, SSL)
proxystubborn.com (US, GProxy, SSL)
proxy4u.com (US, GProxy, SSL)
proxypoint.com (US, GProxy, SSL)
megaproxy.com (US, GProxy, SSL)
proxy.co.uk (GB, Custom, SSL)
fastproxyproxy.com (GProxy)
proxify.us (Custom, SSL)
proxypointproxy.com (Custom, SSL)
unblockyoutube.co (US, GProxy, SSL)
gumim.org (US, PHProxy)
unblock-me.co (US, GProxy)
torrentproxy.co (US, GProxy)
gogetproxies.org (US, GProxy, SSL)
proxify.us (Custom, SSL)
unblockyouku.com (US, GProxy)
securetunnel.com (US, GProxy, SSL)
proxies.us (US, Unknown)
whatismyip.info (US, Unknown)
proxysite.com (US, GProxy)
virtual-ip-test.com (GProxy)
unblockwebsites.ninja (US, GProxy)
suche99.com (US, PHProxy)
unknownproxy.com (US, GProxy)
proxy-youtube.com (US, GProxy)
proxies.biz (GProxy)
proxypointproxy.com (Custom, SSL)
proxify.de (GB, Custom, SSL)
proxify.eu (GB, Custom, SSL)
sslsecureproxy.com (US, GProxy)
hidethisime.com (US, GProxy)
proxify.org (US, Custom, SSL)
freeproxy.ca (US, Unknown)
freevideoclip.com (US, GProxy)

Top Ten Listings

newipnow.com
proxify.com
proxysite.com
proxy.virus.us
proxyturbo.com
4everproxy.com
proxify.co.uk
fastproxynetwork.com

Tier Two Listings

proxify.net
mod5.cc
proxify.co.uk
fastproxynetwork.com

Bold Listings

javascriptproxy.com
proxify.us
peopleproxy.com
unblockyoutube.co
gumim.org
unblock-me.org
torrentproxy.co
gogetproxies.org
proxify.cc
unblockyouku.com
securetunnel.us
proxies.us
whatismyip.info
proxyn.com
virtual-ip-test.com
unblockwebsites.ninja
suche99.com
unknownproxy.com
proxy-youtube.net
proxies.biz
anonymysurf.com
proxify.de
proxify.eu

For Your Website

- Small Proxy Form
- Large Proxy Form

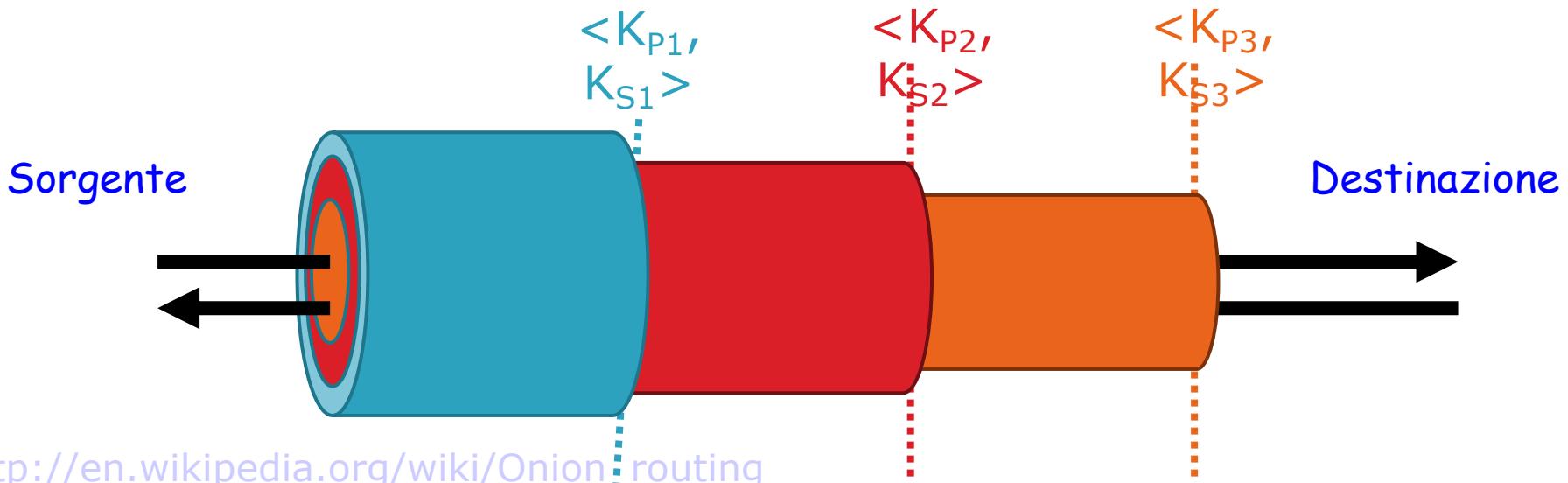
Most Popular:

(sorted by total # of unique visitors in the past 7 days)

- [proxify.com](#) (5,551)
- [megaproxy.com](#) (3,933)
- [newipnow.com](#) (3,694)
- [proxify.us](#) (3,514)
- [proxysite.com](#) (3,327)
- [proxystubborn.com](#) (1,986)
- [fastproxynetwork.com](#) (1,626)
- [mod5.cc](#) (1,596)
- [proxify.co.uk](#) (1,555)
- [proxify.eu](#) (1,498)

Onion routing

- Tecnica di tunneling del traffico su circuiti virtuali (U.S. Naval Research Laboratory, 1998)
 - **Incapsulamento telescopico** dei pacchetti in strutture dati cifrate
 - Instradamento attraverso vari nodi della rete con **multipli layer di cifratura** (una per ogni hop)
 - **Resistente** a tecniche di analisi del traffico
 - Bassa latenza



http://en.wikipedia.org/wiki/Onion_routing

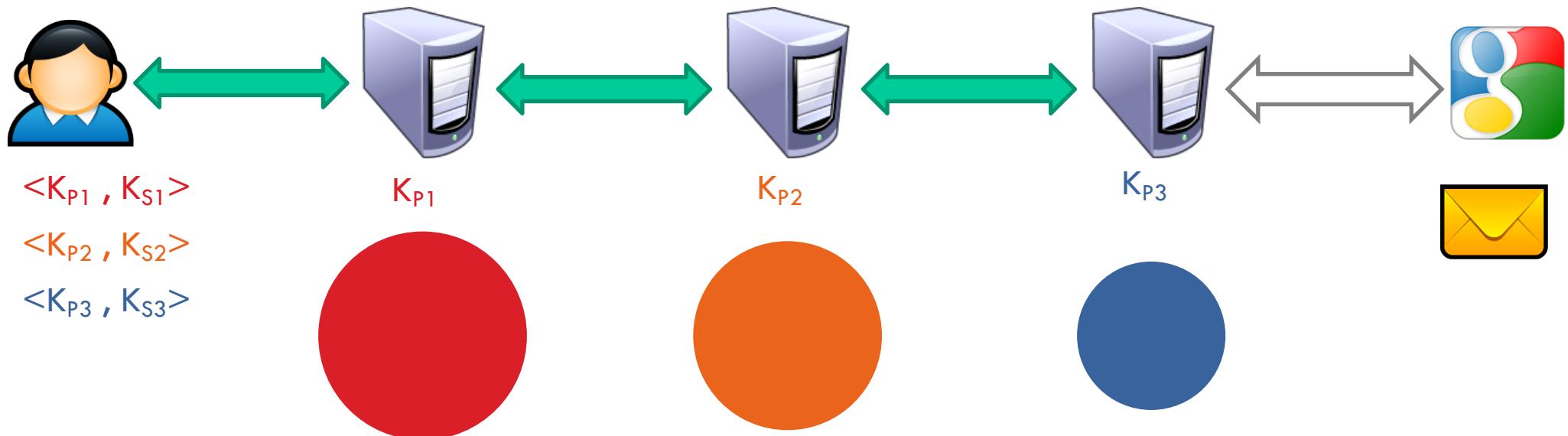
<http://www.onion-router.net/>

<http://www.onion-router.net/Publications.html>

Cammino dei dati

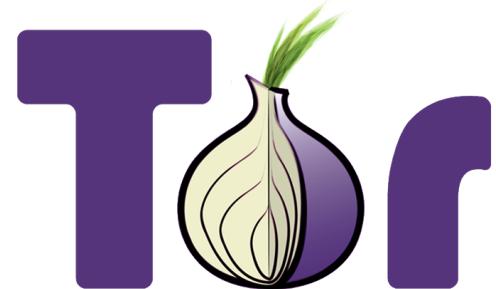
- Un **circuito virtuale** è costruito incrementalmente un hop per volta in accordo a uno schema di cifratura a cipolla
- L'origine sceglie un cammino (insieme di hop) da una lista fornita da un "**directory node**"
- L'origine **negozia una chiave** (o una **coppia di chiavi**) con ogni nodo intermedio
- I dati sono cifrati in origine e decifrati lungo il cammino
- Ogni nodo intermedio conosce **solo** il suo **predecessore** e **successore**
- Solo il nodo di **uscita** può vedere il messaggio, decifrando l'ultimo layer ma non sa da dove o stesso proviene

$$E(K_{P1}, E(K_{P2}, E(K_{P3}, M))) = C$$



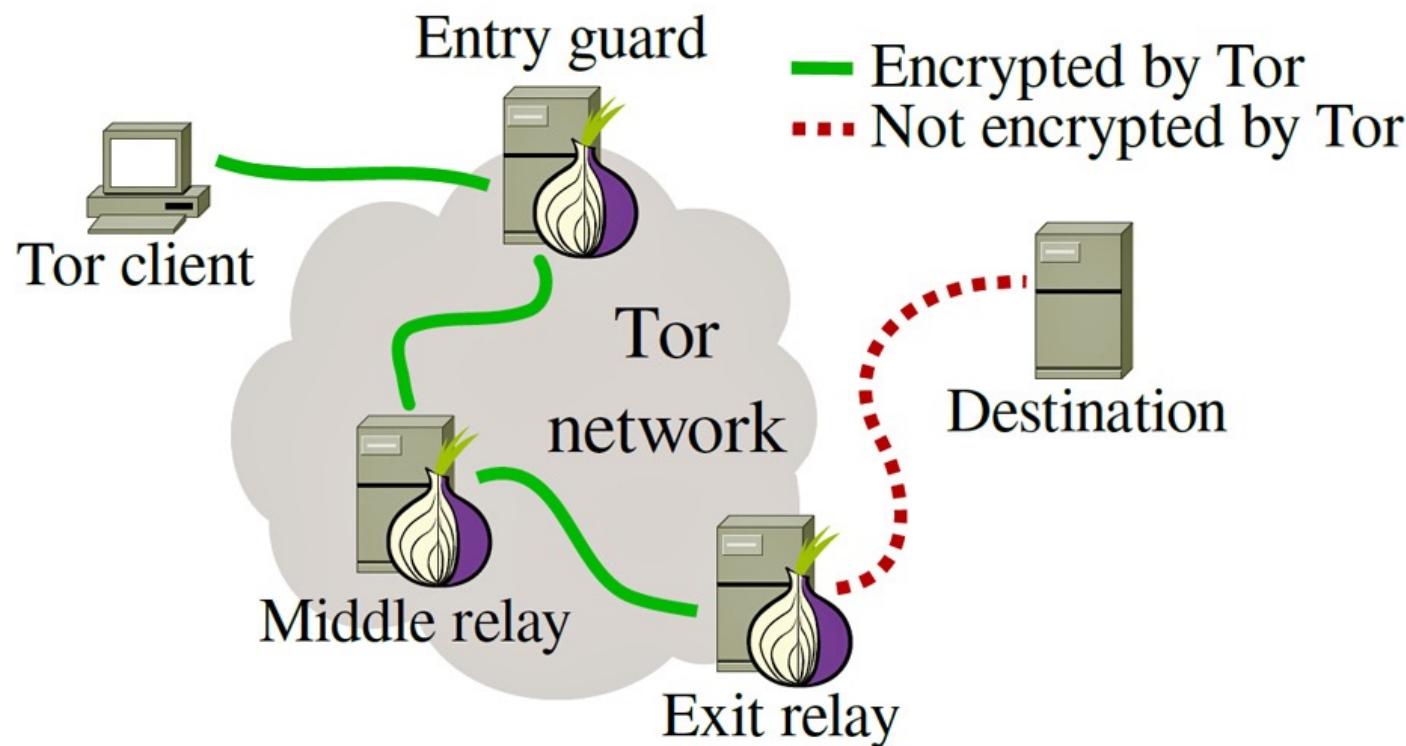
The Onion Router (TOR)

- Rete **Overlay** di Onion Routers col ruolo di **relay**
- Sviluppato per la **US Naval Research Laboratory** e poi da **DARPA**, per proteggere le comunicazioni dei servizi segreti.
- Fornisce essenzialmente due servizi:
 - **Connessioni anonime** in uscita
 - Fornitura di **servizi nascosti**
- Supporta:
 - **Perfect Forward Secrecy**
 - Utilizzo di TLS nelle comunicazioni tra nodi
 - Controllo di **integrità** dei dati end-to-end
 - Controllo di **congestione**
 - Interfacciamento tramite il protocollo **SOCKS**
 - Scelta dei relay basata sulla loro **larghezza di banda**



Architettura hop-by-hop relay

- Tor è un'infrastruttura overlay (rete sulla rete)
- I nodi sono connessi fra loro tramite collegamenti logici (circuiti virtuali), ciascuno dei quali corrisponde ad un percorso nella rete sottostante.
- Il nodo di ingresso (entry guard) garantisce l'accesso all'infrastruttura overlay
- Il nodo di uscita della rete overlay garantisce l'accesso alla global internet



Tipi di nodi

La rete **Tor** prevede quattro tipologie di nodi:

- **Client**

In questa configurazione normale di base, Tor gestisce unicamente le connessioni dell'utente permettendogli di collegarsi alla rete Tor.

- **Middleman router (o middle relay)**

È un nodo che gestisce traffico di terzi da e per la rete Tor, senza collegarsi direttamente all'esterno. Nel caso funga anche da client, esso gestisce anche le connessioni dell'utente, garantendo un maggiore anonimato. Tutti nodi sono pubblicamente noti, per scelta progettuale.

- **Exit router (o exit relay)**

È un nodo Tor che gestisce traffico di terzi da e per la rete Tor, e verso l'esterno. È possibile definire una exit policy sulle connessioni in uscita all'esterno della rete Tor. Come il caso precedente, offre una protezione maggiore all'utente che lo usa per le proprie connessioni. Come tutti i router Middleman Tor, essi sono pubblicamente noti.

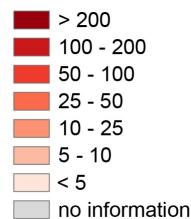
- **Bridge router**

I bridge router sono nodi semi-pubblici di tipo sperimentale, studiati per permettere di collegarsi anche in presenza di un filtraggio efficace contro Tor (come in Cina, Iran ecc.). Non compaiono nelle liste pubbliche dei nodi noti ma devono venire richiesti esplicitamente.

Estensione rete TOR

The anonymous Internet

Daily Tor users per 100,000 Internet users

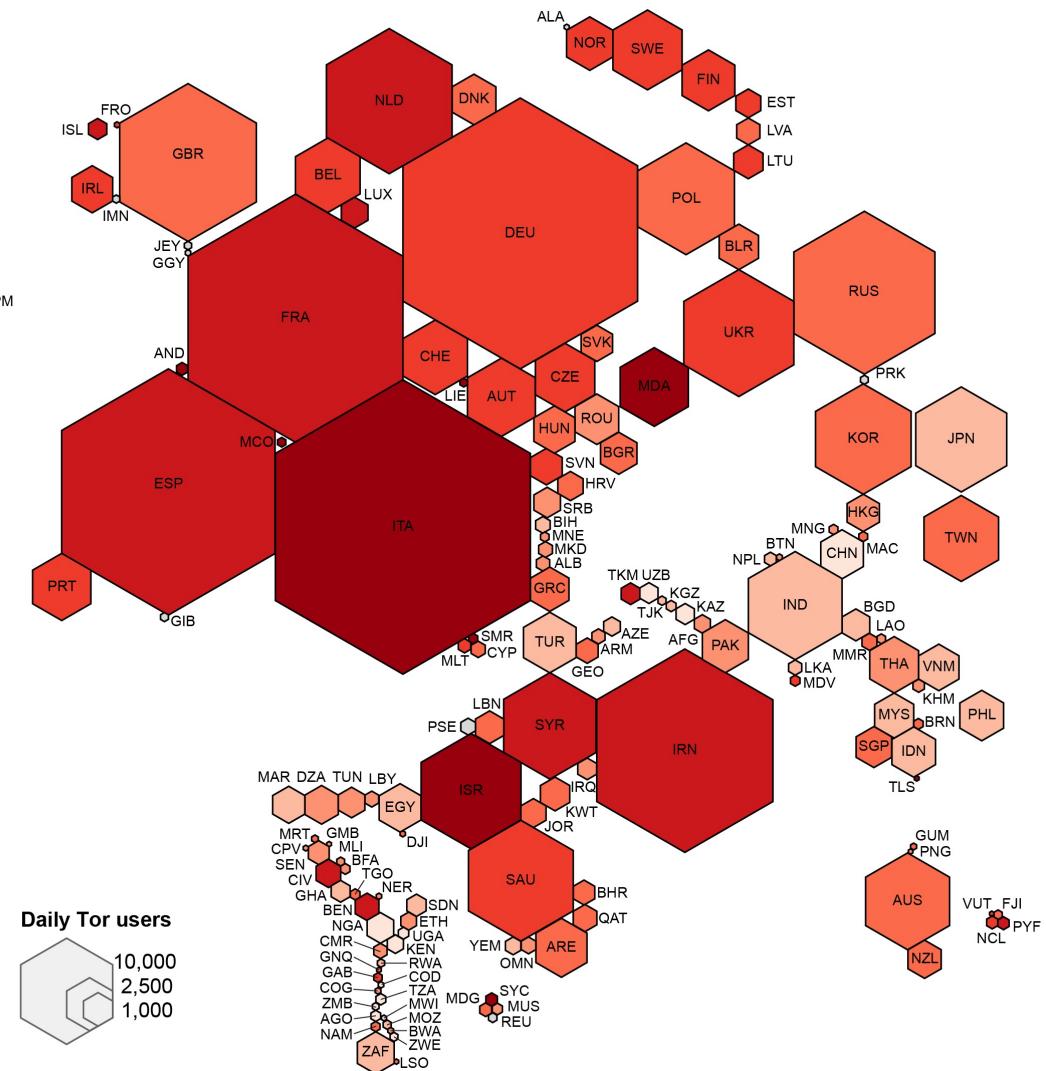
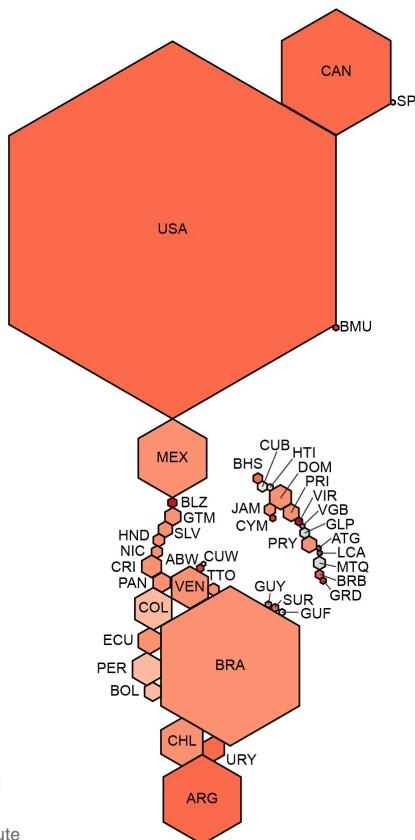


Average number of Tor users per day calculated between August 2012 and July 2013

data sources:
Tor Metrics Portal
metrics.torproject.org
World Bank
data.worldbank.org

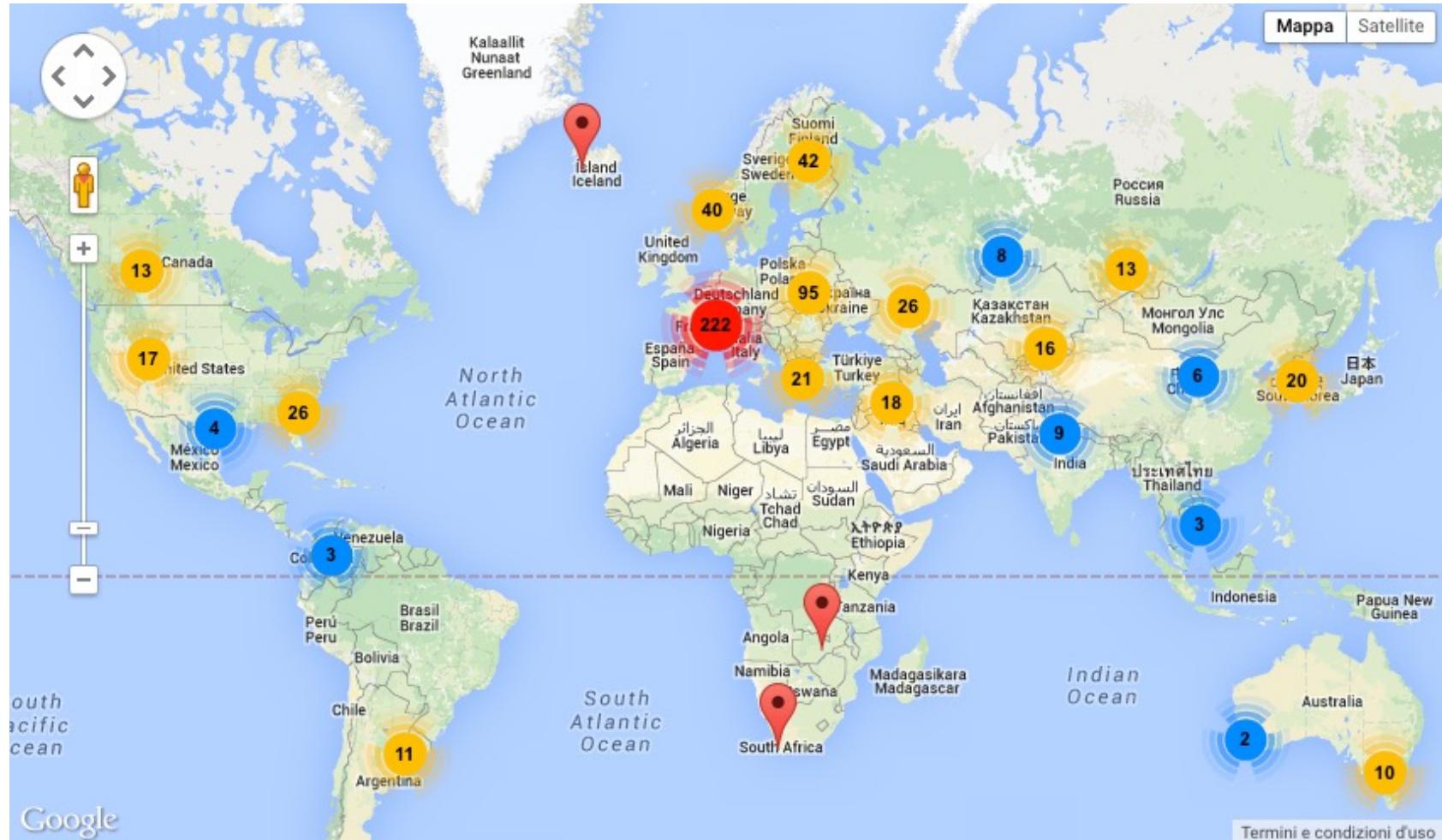
by Mark Graham (@geoplace) and Stefano De Sabbata (@maps4thought)
Internet Geographies at the Oxford Internet Institute
2014 • geography.ox.ac.uk

Oxford Internet Institute
University of Oxford



<https://metrics.torproject.org/oxford-anonymous-internet.html>

Nodi Exit

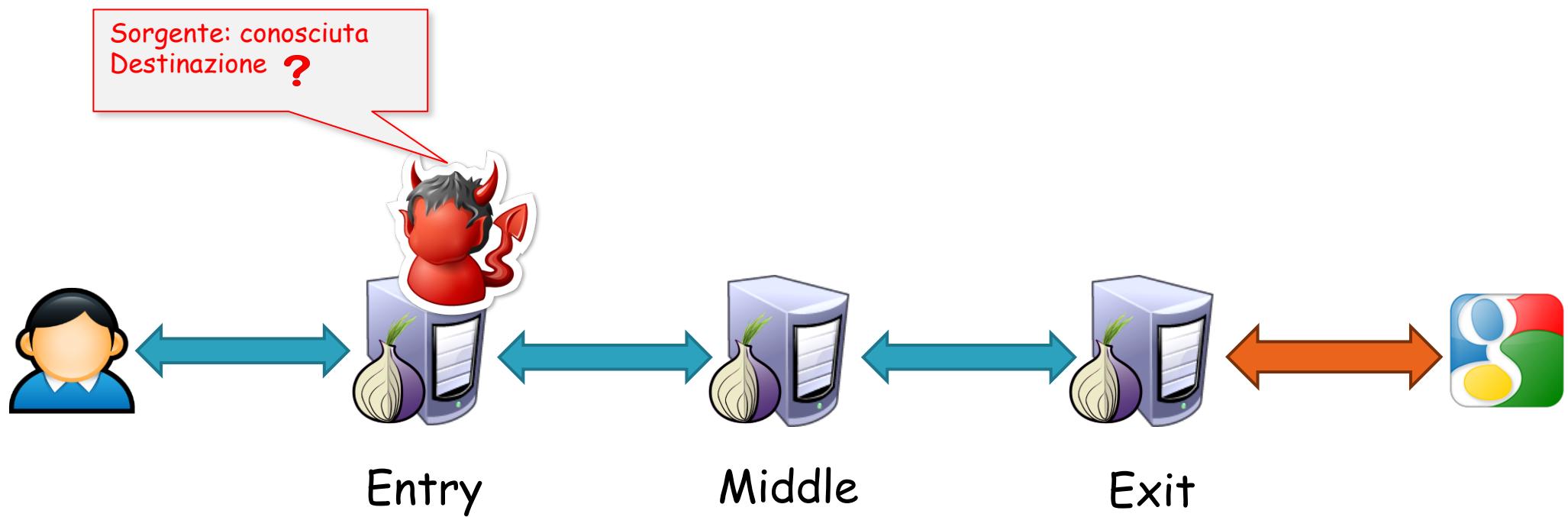


23 febbraio 2015

<http://hackertarget.com/tor-exit-node-visualization/>

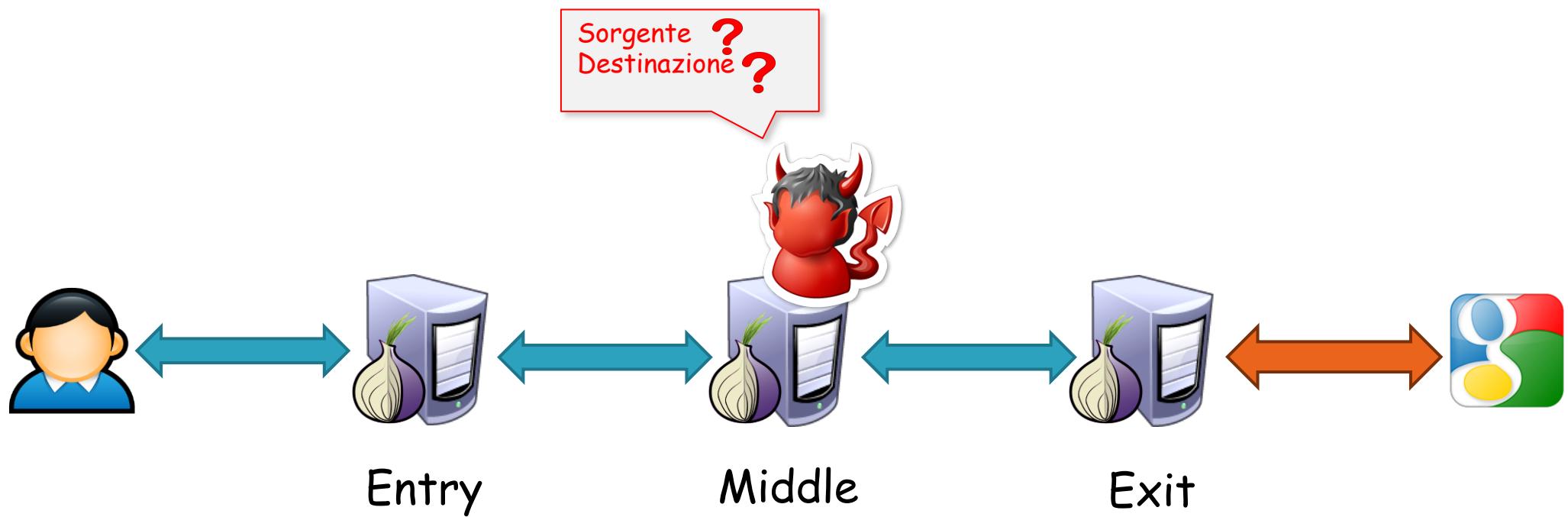
Numero di relay

- Possibile scegliere numero di relay
 - Configurazione di default: 3
 - Meglio di più o di meno?



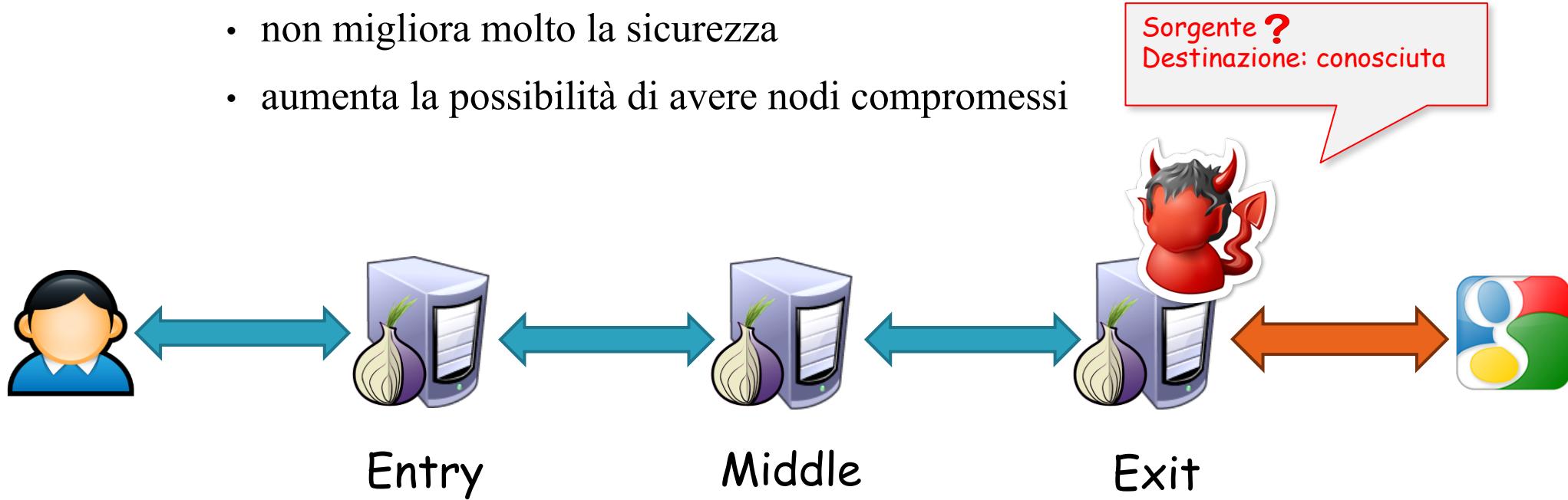
Numero di relay

- Possibile scegliere numero di relay
 - Configurazione di default: 3
 - Meglio di più o di meno?



Numero di relay

- Possibile scegliere numero di relay
 - Configurazione di default: 3
 - Meglio di più o di meno? Con numero maggiore:
 - prestazioni minori e latenza maggiore
 - non migliora molto la sicurezza
 - aumenta la possibilità di avere nodi compromessi



Bridges

- Gli Indirizzi IP dei Tor relay sono noti
- Alcuni paesi bloccano il traffico a questi IP
- Soluzione: Tor Bridges
 - Essenzialmente, Tor proxies che non sono pubblici
 - Usati per la connessione di client al resto della rete Tor
- Tor ha bridges in molti paesi. Come ottenere la lista:
 - <https://bridges.torproject.org/bridges>
 - Email a bridges@bridges.torproject.org
body "get bridges"
 - Esempio risposta:

```
5.20.130.121:9001
63dd98cd106a95f707efe538e98e7a6f92d28f94106.186.19.58:443
649027f9ea9a8e115787425430460386e14e0ffa69.125.172.116:443
43c3a8e5594d8e62799e96dc137d695ae4bd24b2
```

Celle

- Concetto simile alle celle in ATM
- Tutti i dati sono inviati in celle di taglia fissa (numero bytes)
- Celle di controllo:
 - Creazione (CREATE), riscontro (CREATED) distruzione di circuiti
- Celle di Relay:
 - Trasporto dati end-to-end
 - Apertura, chiusura stream, estensione circuiti (EXTEND)...

2	1	509 bytes			
CircID	CMD	DATA			

2	1	2	6	2	1	498
CircID	Relay	StreamID	Digest	Len	CMD	DATA

In caso di compromissione...

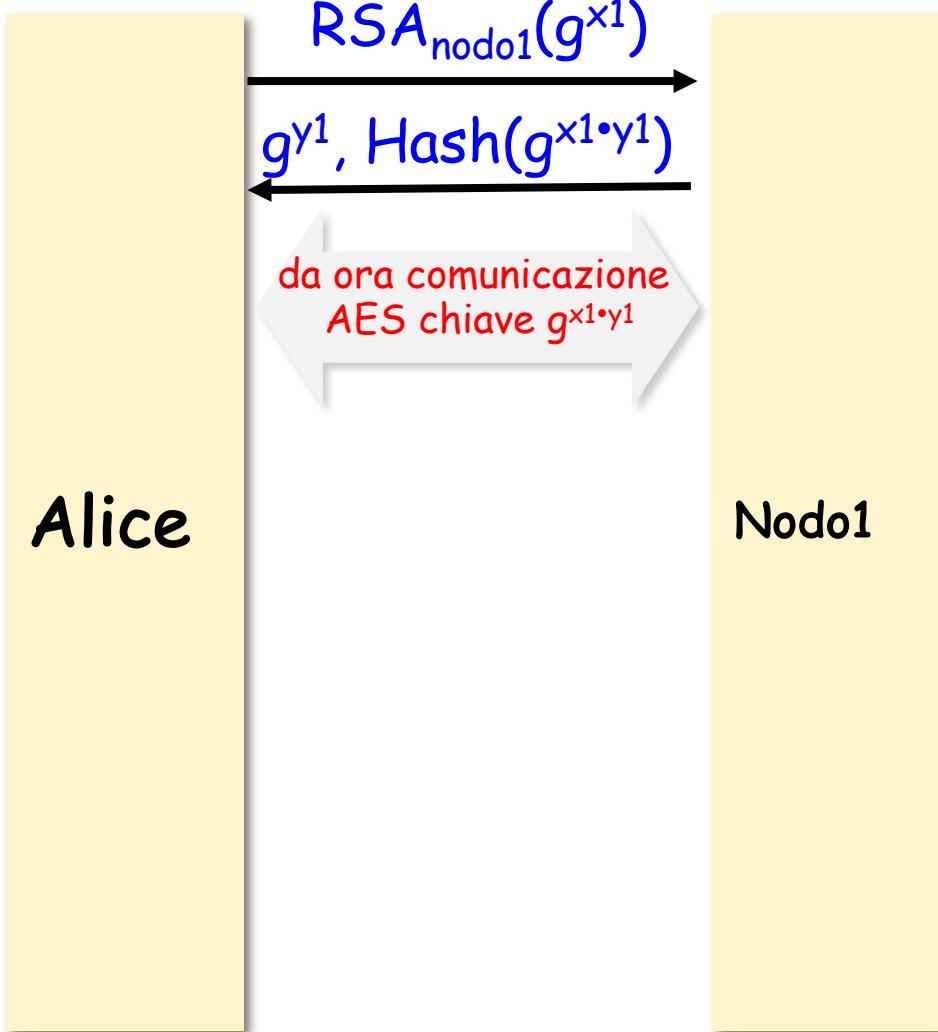
- Nella logica onion routing di base viene usata una coppia di chiavi pubblica/privata negoziate con ogni relay per cifrare il traffico...
- Che cosa succede se una chiave privata viene rubata?
 - Tutto il traffico **futuro** può essere decifrato
 - Se il traffico **passato** è disponibile, può essere decifrato

Perfect Forward Secrecy

- La perfect forward secrecy, è una proprietà dei protocolli di negoziazione delle chiavi che assicura che se una chiave di cifratura a lungo termine viene compromessa, le chiavi di sessione generate a partire da essa rimangono riservate....
- Tor usa **Perfect Forward Secrecy**
 - Il client negozia una nuova coppia di chiavi con ogni relay
 - Chiavi originali usate solo per firma (cioè per verificare l'autenticità dei messaggi)

- Se  ottiene una chiave privata può decifrare tutto il traffico futuro
- ... ma il traffico passato è cifrato con chiavi non più disponibili

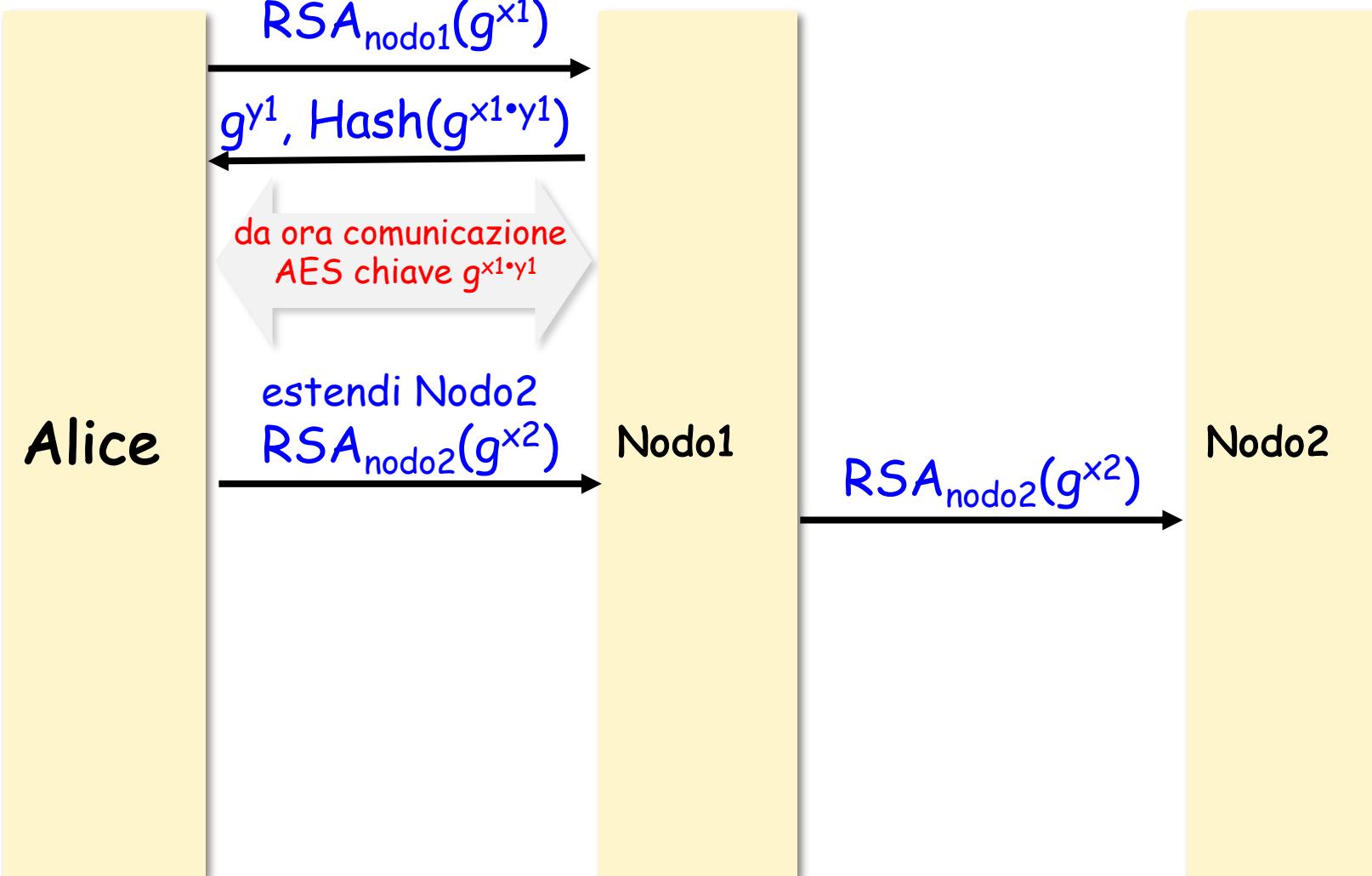
Perfect Forward Secrecy



Alice stabilisce un circuito con Nodo1

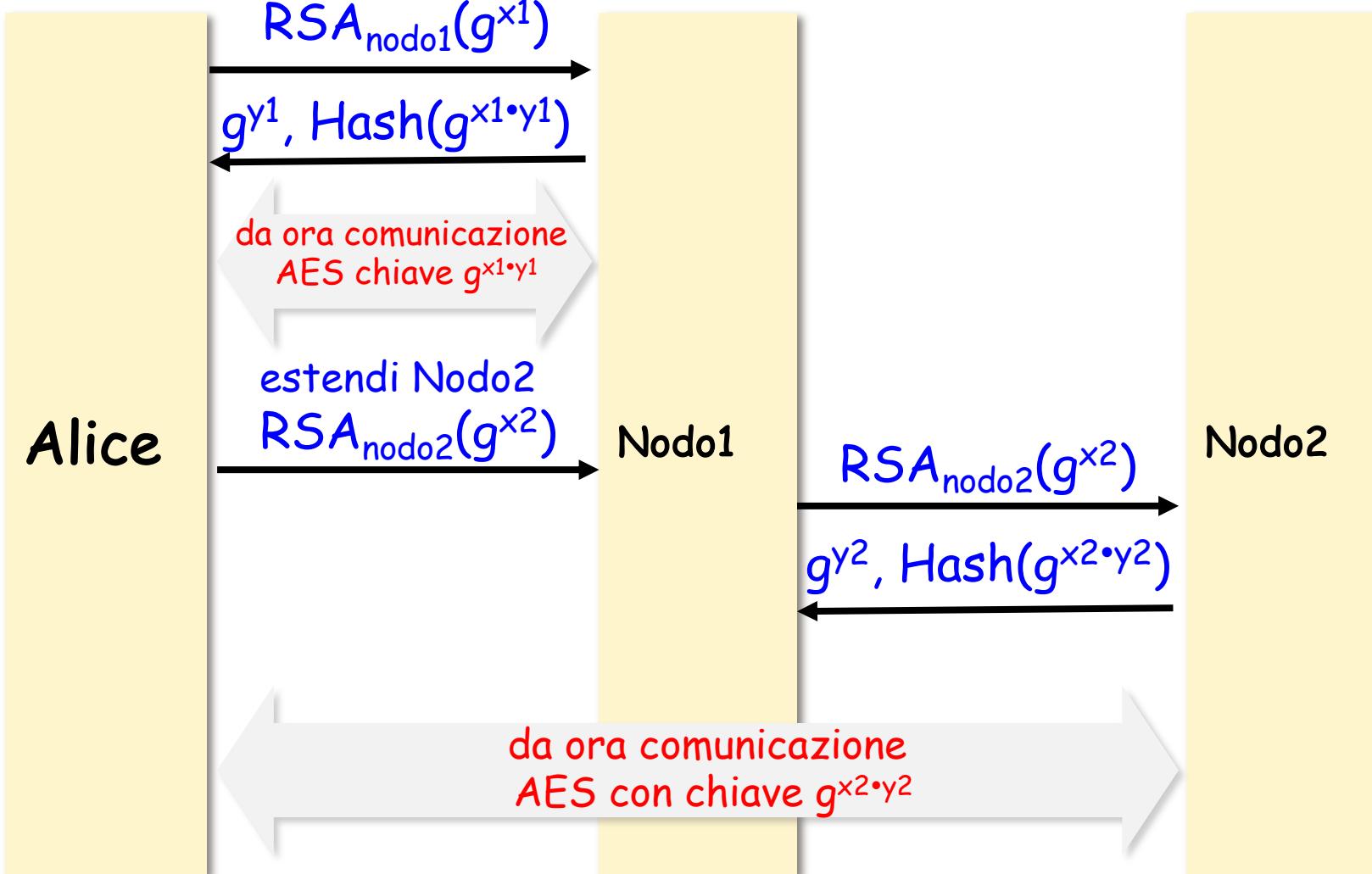
- Alice invia una cella CREATE a Nodo1 contenente g^{x1} cifrandola con la chiave pubblica di Nodo1
- Nodo1 decifra con la sua chiave privata la cella CREATE, calcola $g^{x1}y1$ (prima parte dell'handshake di Diffie-Hellman) e deriva K_1 (chiave di sessione tra Alice e Nodo1)
- Poi invia ad Alice la cella CREATED contenente g^{y1} e l'hash della chiave $g^{x1}y1$ $H(g^{x1}y1)$
- Alice calcola $g^{x1}y1$ (seconda parte dell'handshake di Diffie-Helmann) e deriva K_1 utilizzando $H(K_1)$.
- Il circuito tra Alice e Nodo1 è creato e K_1 sarà la chiave usata da AES

Perfect Forward Secrecy



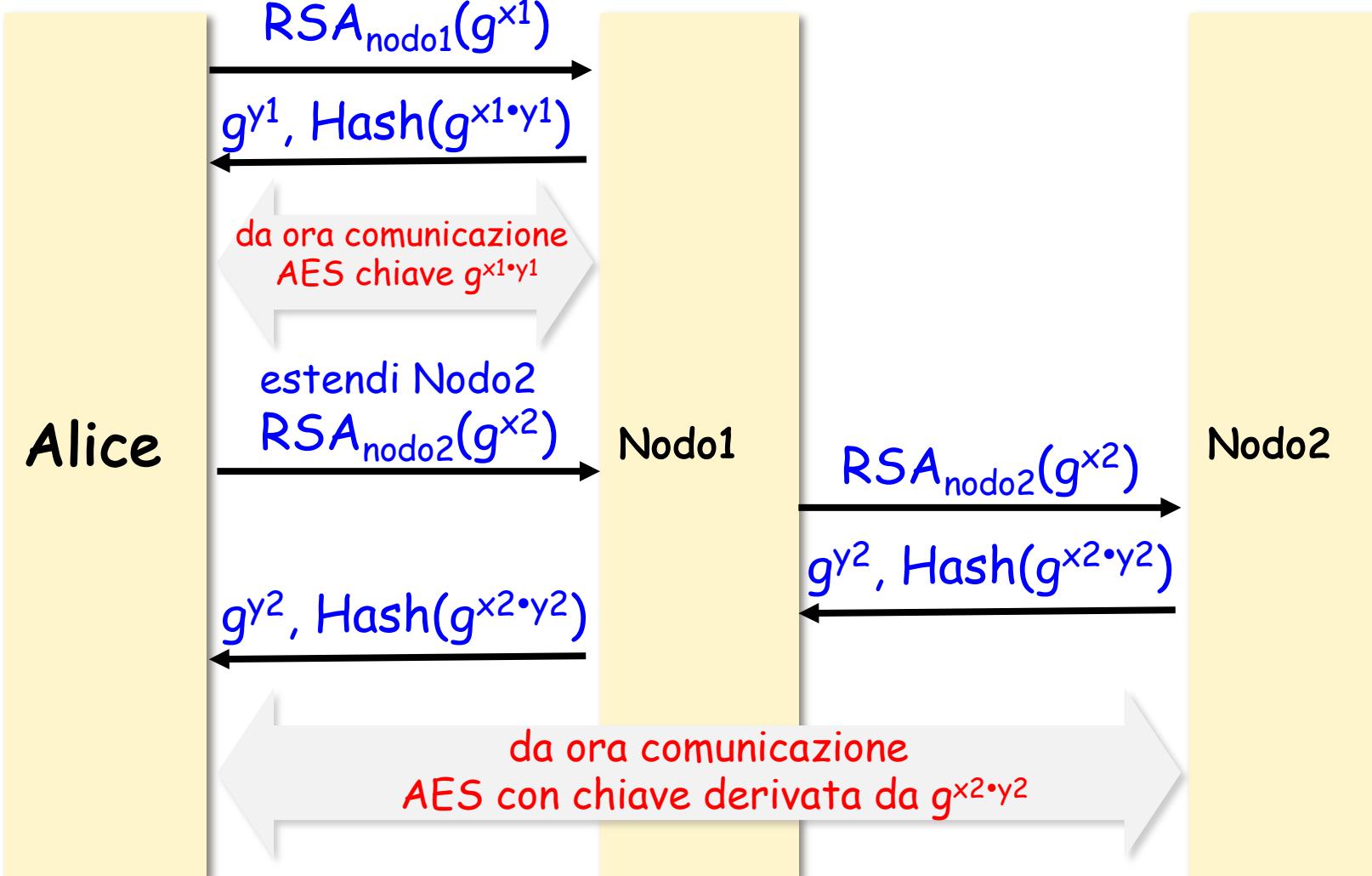
- Alice invia a Nodo1 una cella RELAY EXTEND cifrata con K_I , tale cella contiene g^{x2} cifrata con la chiave pubblica di Nodo2;
- Nodo1 decifra la cella con K_I , essendoun RELAY EXTEND, legge l'indirizzo del prossimo nodo
- Nodo1 estrae il payload della cella e lo inserisce in una nuova cella CREATE inviadola a Nodo2

Perfect Forward Secrecy



- Nodo2 decifra la cella, calcola g^{x2y2} e deriva K_2
- Nodo2 invia la cella CREATED contenente g^{y2} e l'hash della chiave $H(K_2)$ a Nodo1
- Nodo1 estrae il payload dalla cella, lo mette in una cella RELAY EXTENDED cifrandola con K_1

Perfect Forward Secrecy



- Nodo1 invia la cella appena creata a Alice
- Alice decifra con K_1 , calcola g^{x2y2} e deriva K_2 .
- Il circuito virtuale tra Alice e Nodo è stato creato e userà AES con chiave K_2 .

Controlli

Controllo di integrità

Effettuato su ogni end-point dei circuiti con un digest SHA-1 derivato dalla chiave negoziata ad ogni hop che rende impossibile la modifica dei dati in transito

Rate Limiting

Usa un approccio token per limitare I volumi di traffico in ingresso

Controllo di congestione

Realizzato via Circuit-level throttling basato su 2 finestre

- Packaging window: quante celle un relay può raggruppare e mandare indietro
- Delivery window: quante celle possano essere mandate avanti sulla rete esterna

Privoxy

Proxy HTTP con funzioni avanzate di **filtraggio**:

- Modifica dei contenuti del protocollo HTTP
- Blocco di pop-up
- Blocco di informazioni pubblicitarie
- Controllo dei cookie
- Rimuove informazioni superflue che il nostro browser comunica

L'uso di Privoxy è necessario perché i browser fanno passare le richieste DNS quando usano direttamente un proxy SOCKS, e ciò non è buono per mantenere l'anonimato.

<http://www.privoxy.org/>



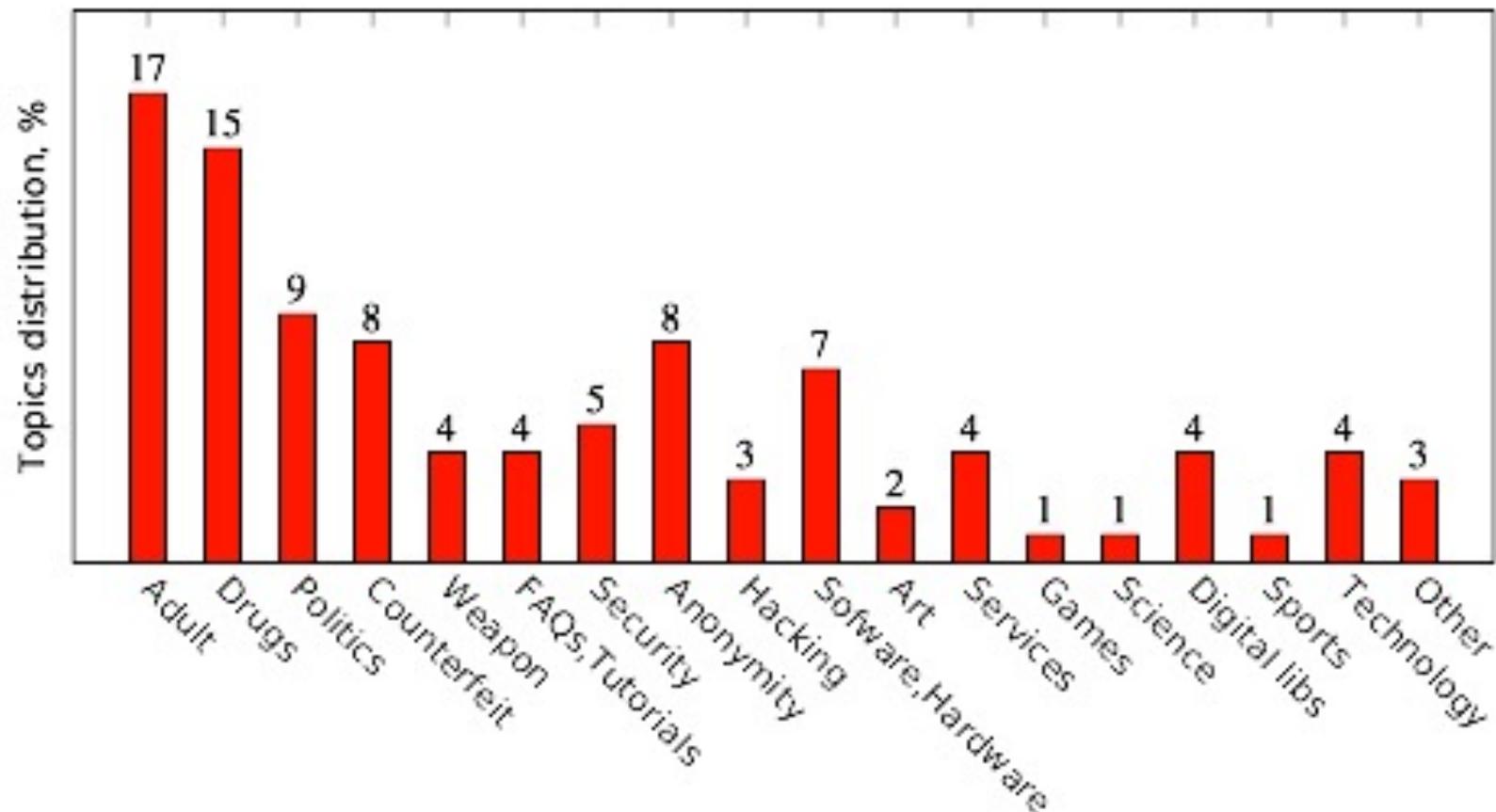
Hidden Services

- L'identità in rete dell' host ospitante il servizio non viene rivelata realizzando:
 - Servizi di rete di cui non si conosce la collocazione fisica
 - Servizi resistenti alla censura (filtraggio del traffico, filtraggio DNS)
 - Resistenza alla violazione fisica (se non si sa dove sia il server..)
 - Impossibilità di risalire a chi pubblica e a chi fruisce il servizio
- Ciò consente di avere un server in grado di erigere servizi senza che si conosca l'indirizzo IP oppure il nome DNS



<http://tor.eff.org/docs/tor-hidden-service.html>

Hidden Services



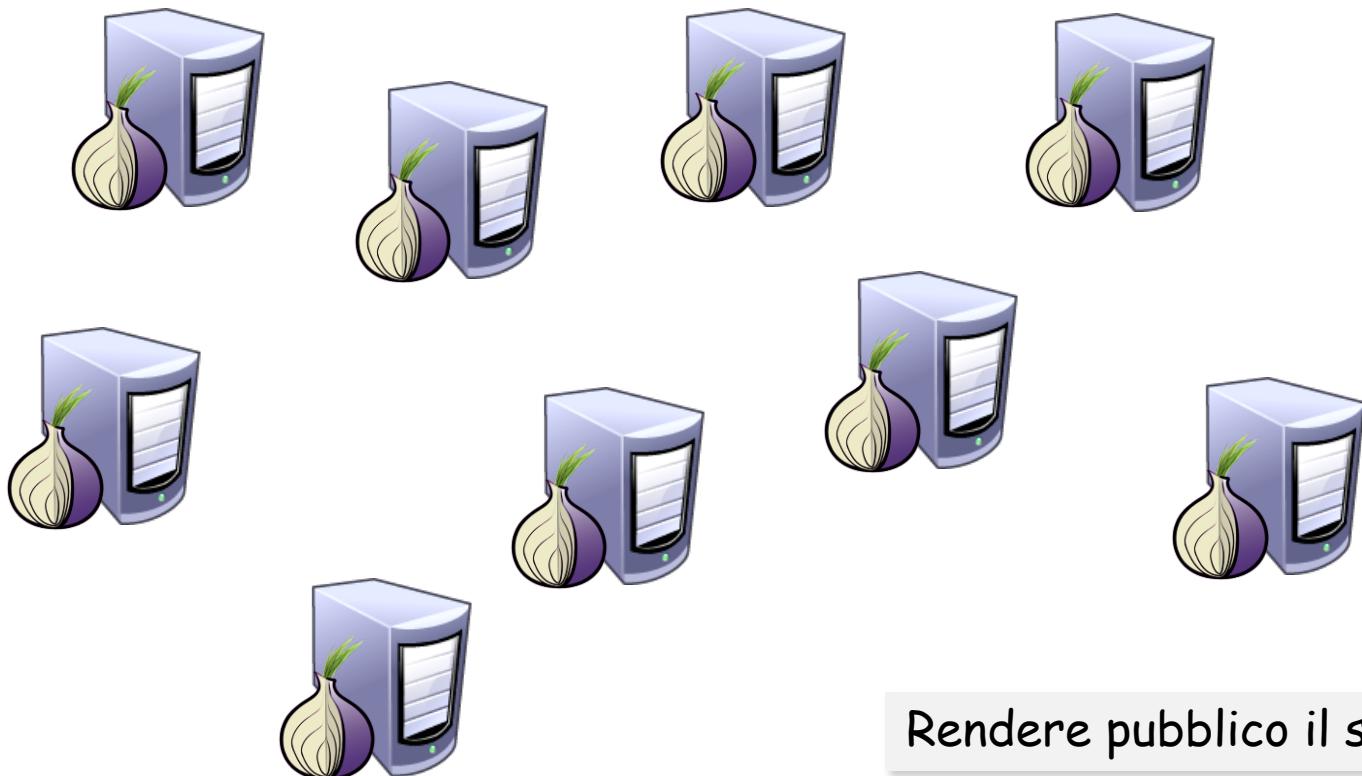
<http://arxiv.org/pdf/1308.6768v1.pdf>, 30 agosto 2013

Hidden Services

- Come funzionano?
- E' possibile individuare il server?

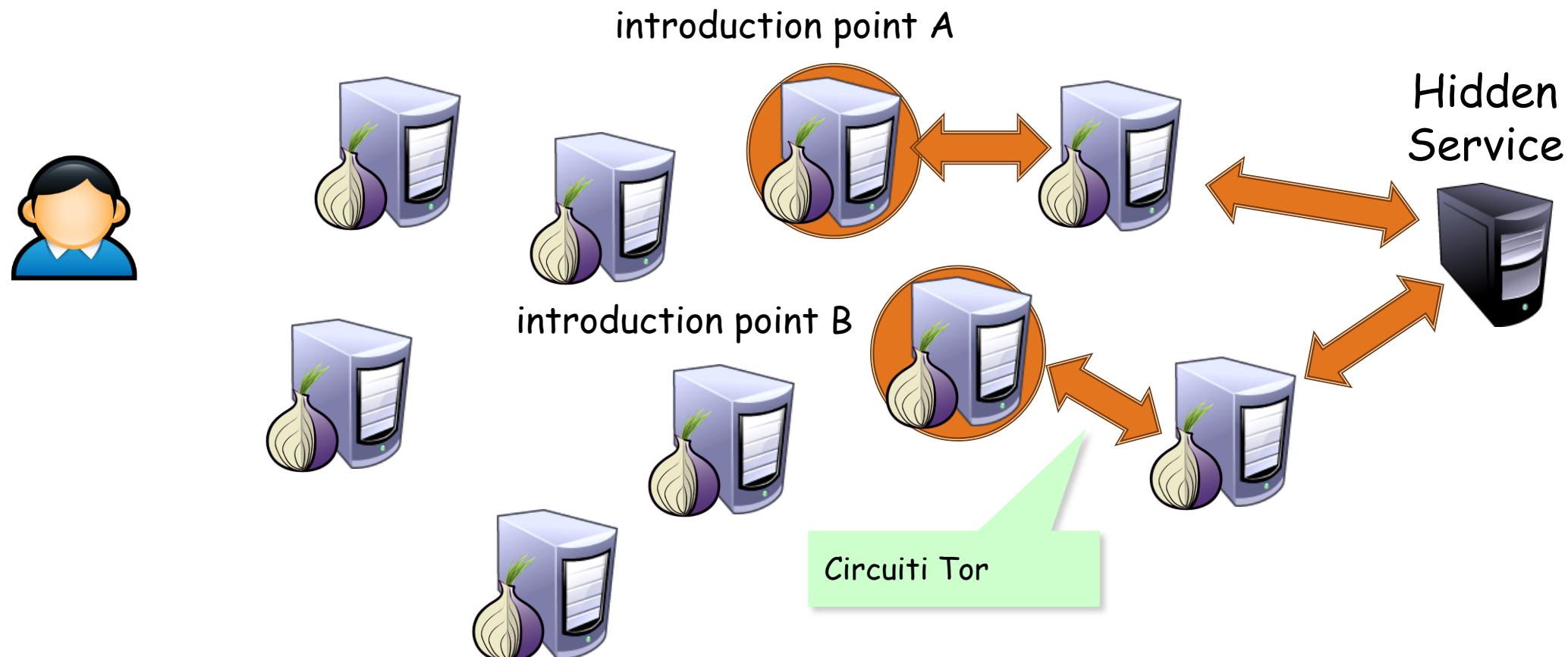


Hidden Services



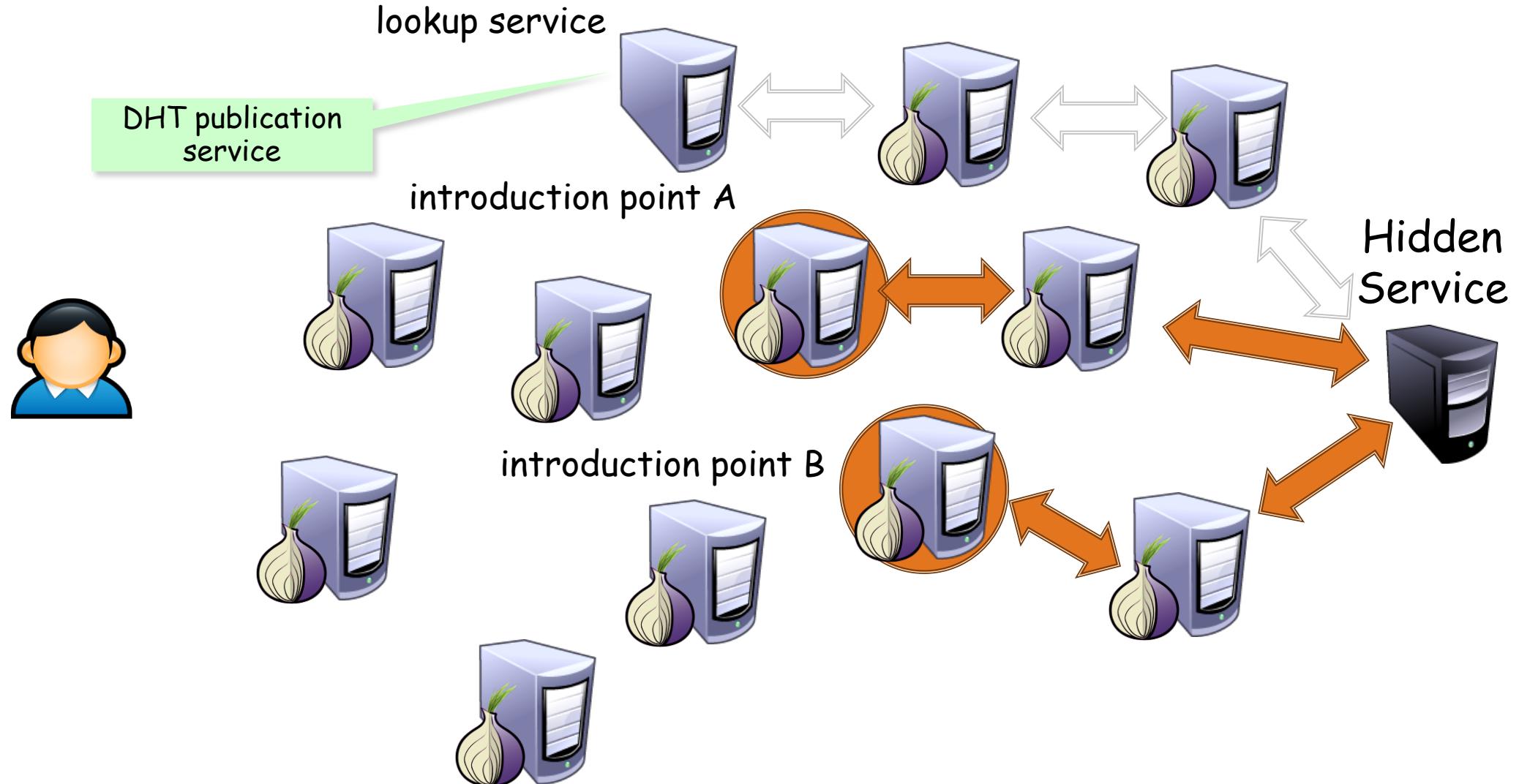
Rendere pubblico il servizio

Hidden Services



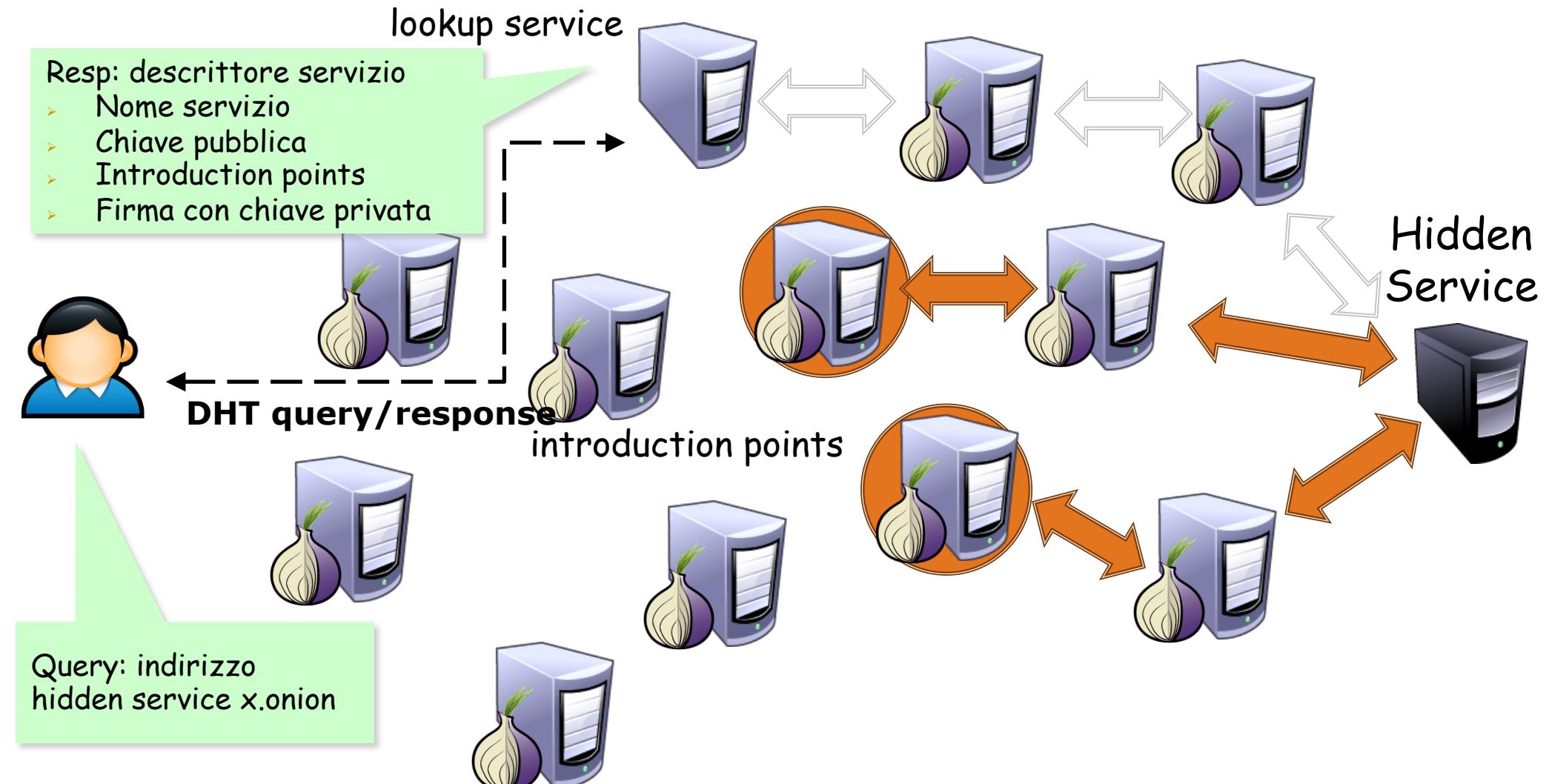
L'hidden service sceglie in modo casuale alcuni relay (A e B nell'esempio), costruisce dei circuiti TOR verso di essi e chiede loro di agire da **introduction point** fornendo loro la sua **chiave pubblica** che identifica il servizio. A e B **sanno come arrivare** all'hidden service attraverso circuiti TOR ma non conoscono la sua identità

Hidden Services



L'hidden service costruisce un "service descriptor", contenente la chiave pubblica e un summary di ogni introduction point, firma questo descrittore con la sua chiave privata.
Il descrittore viene inviato ad un lookup service (es. una DHT), usando sempre un circuito TOR in modo da proteggere sempre l'anonimato dell'hidden service

Hidden Services

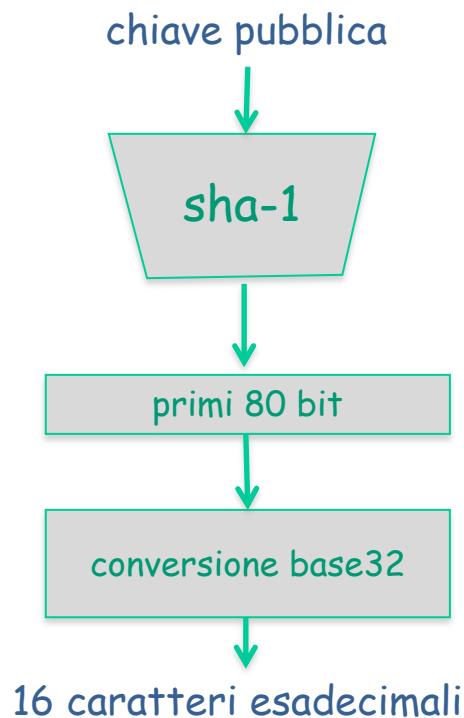


Un client per contattare un hidden service ha bisogno di conoscere il suo indirizzo x.onion. Dopo, il client può avviare la creazione della connessione scaricando il descrittore dal lookup service che ha il compito di pubblicizzare il servizio nascosto nella rete Tor.

URL x.onion

<http://go2ndkjdf7whfanf.onion>

- E' un hash della chiave pubblica
- Meglio evitare nomi non correlati alla chiave pubblica



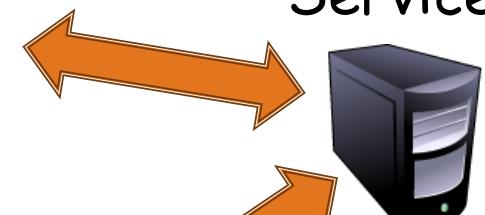
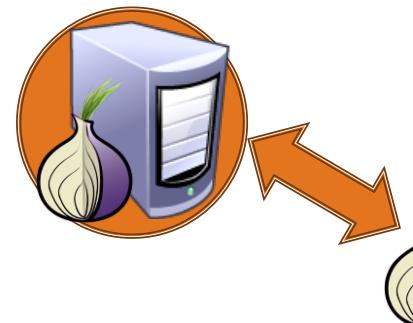
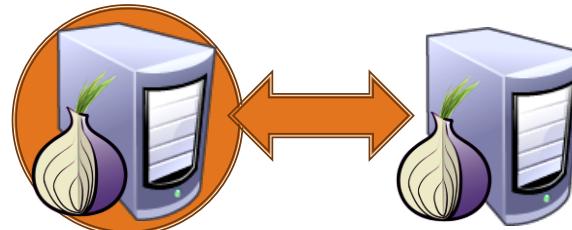
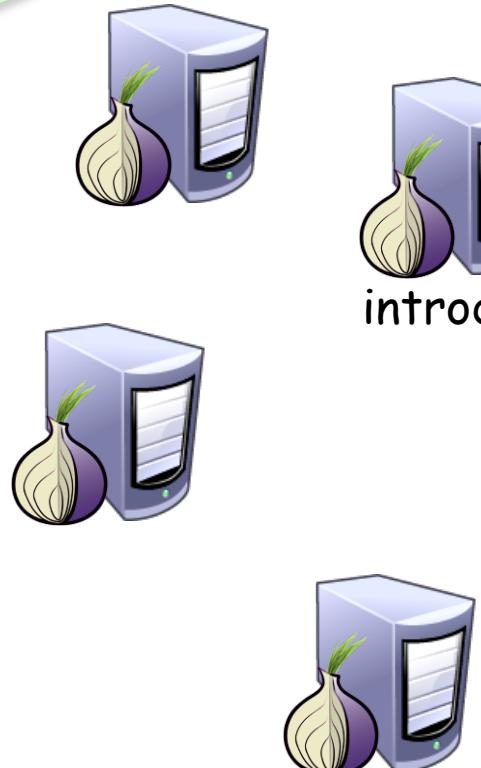
Hidden Services

Descrittore servizio

- Nome servizio
- Chiave pubblica
- Introduction points
- Firma



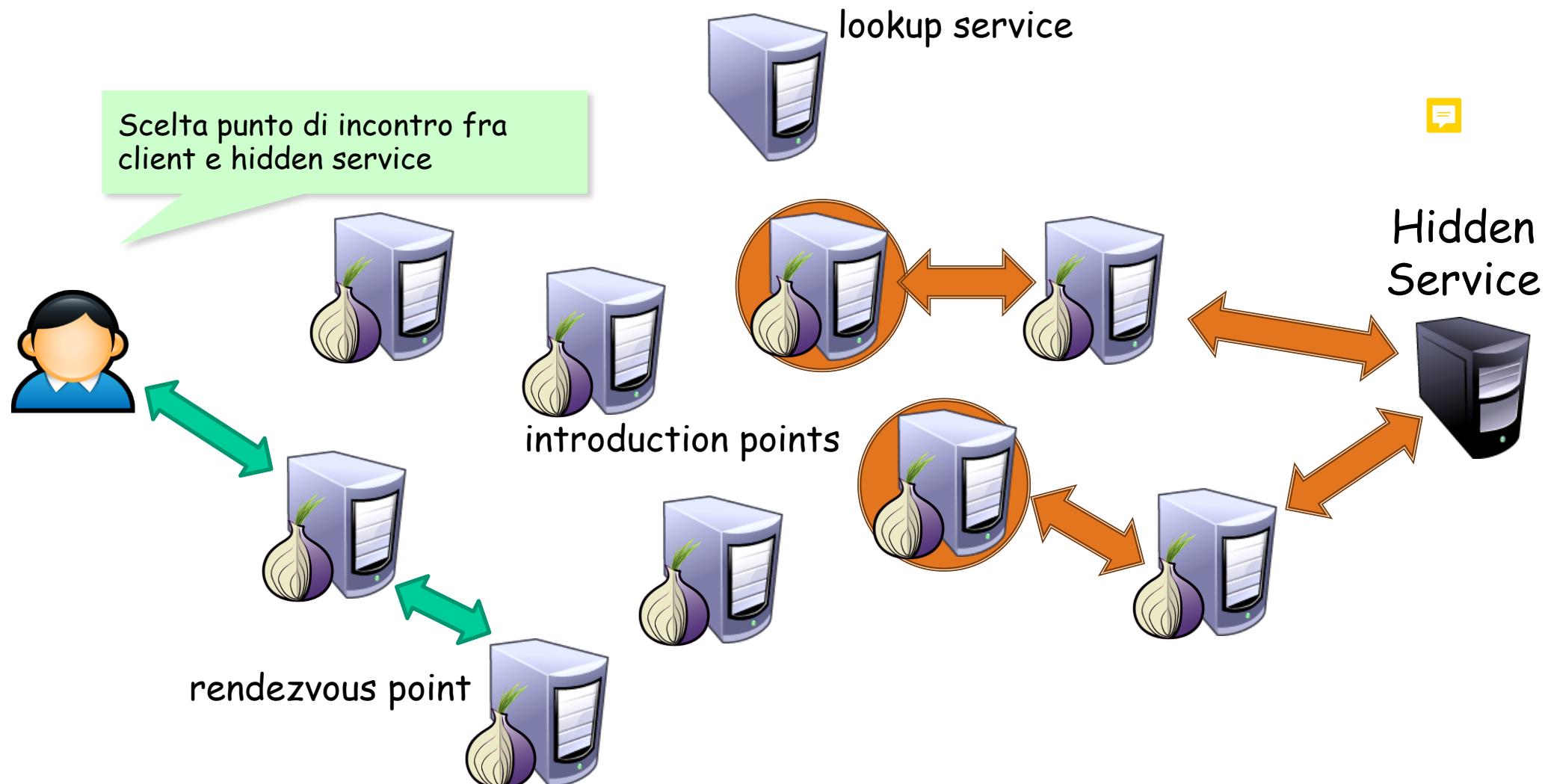
lookup service



Hidden
Service

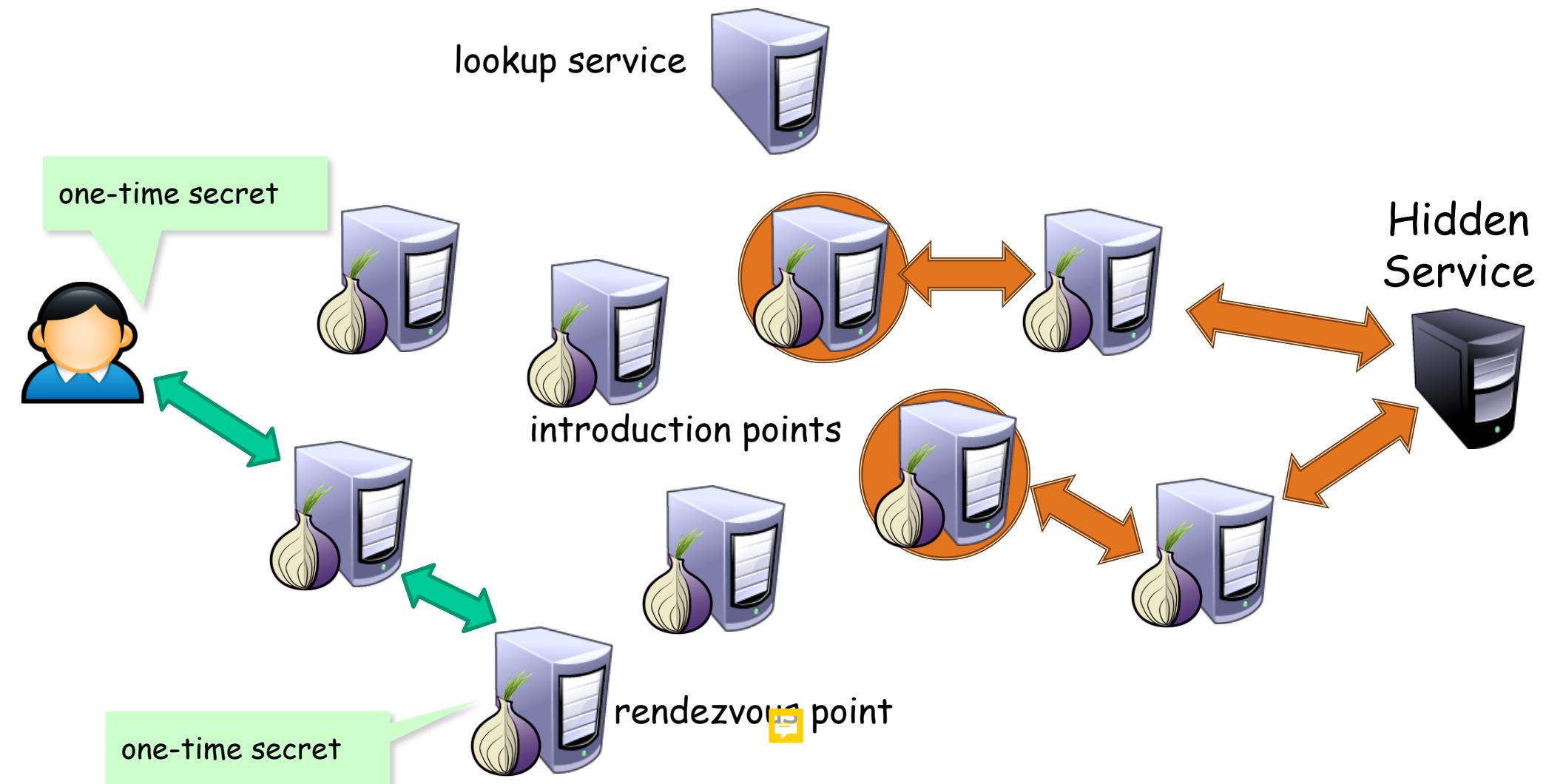
Il client conosce ora dal descrittore la lista degli introduction point da contattare e la chiave pubblica da usare per comunicare con loro.

Hidden Services



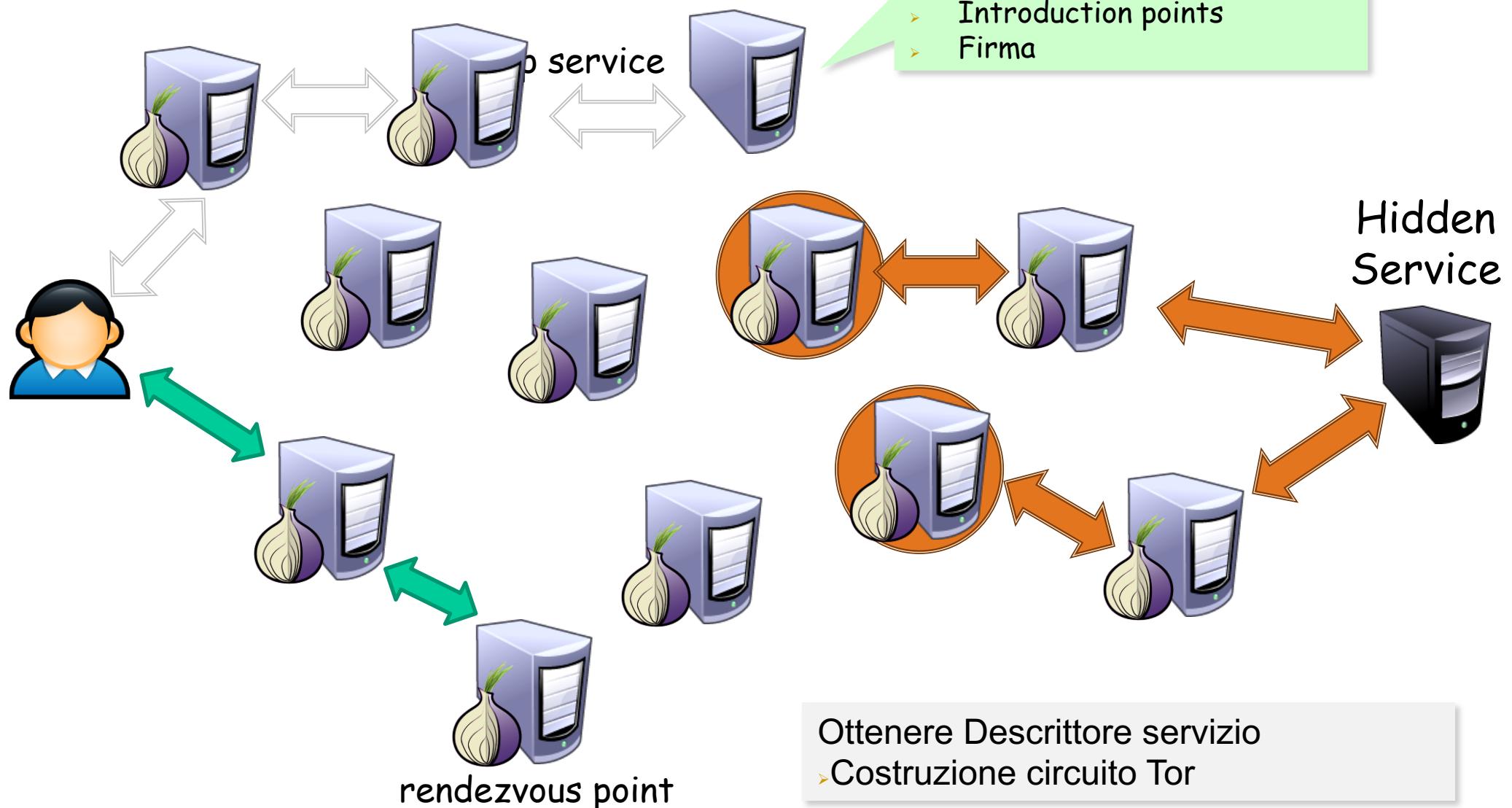
- Il client crea un circuito TOR verso un altro relay scelto a caso e gli chiede di fungere da rendezvous point (punto d'incontro tra client e server)

Hidden Services



Il client manda al rendezvous point un "One-time secret" (rendezvous cookie).

Hidden Service



Ottenerne Descrittore servizio
► Costruzione circuito Tor

Hidden Services

Introduction MSG:

- . Indirizzo RP
 - . One time secret
- Da mandare a HS

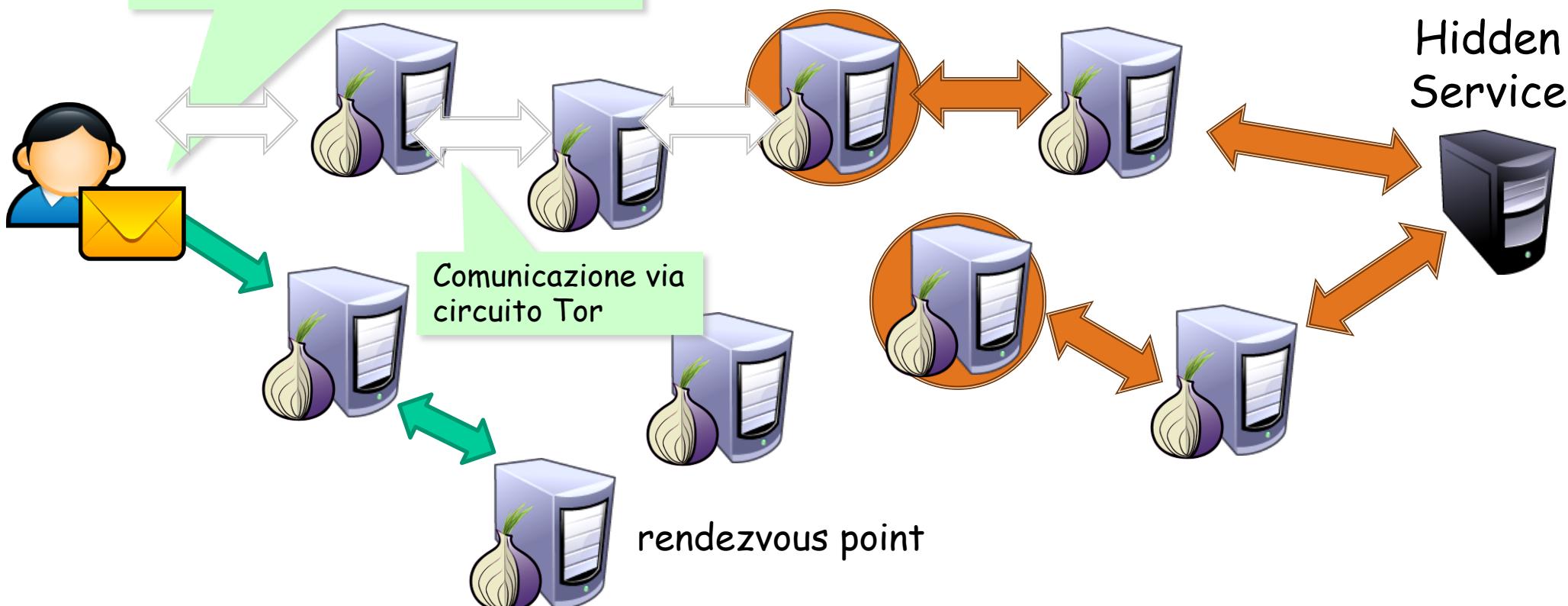


Hidden Service



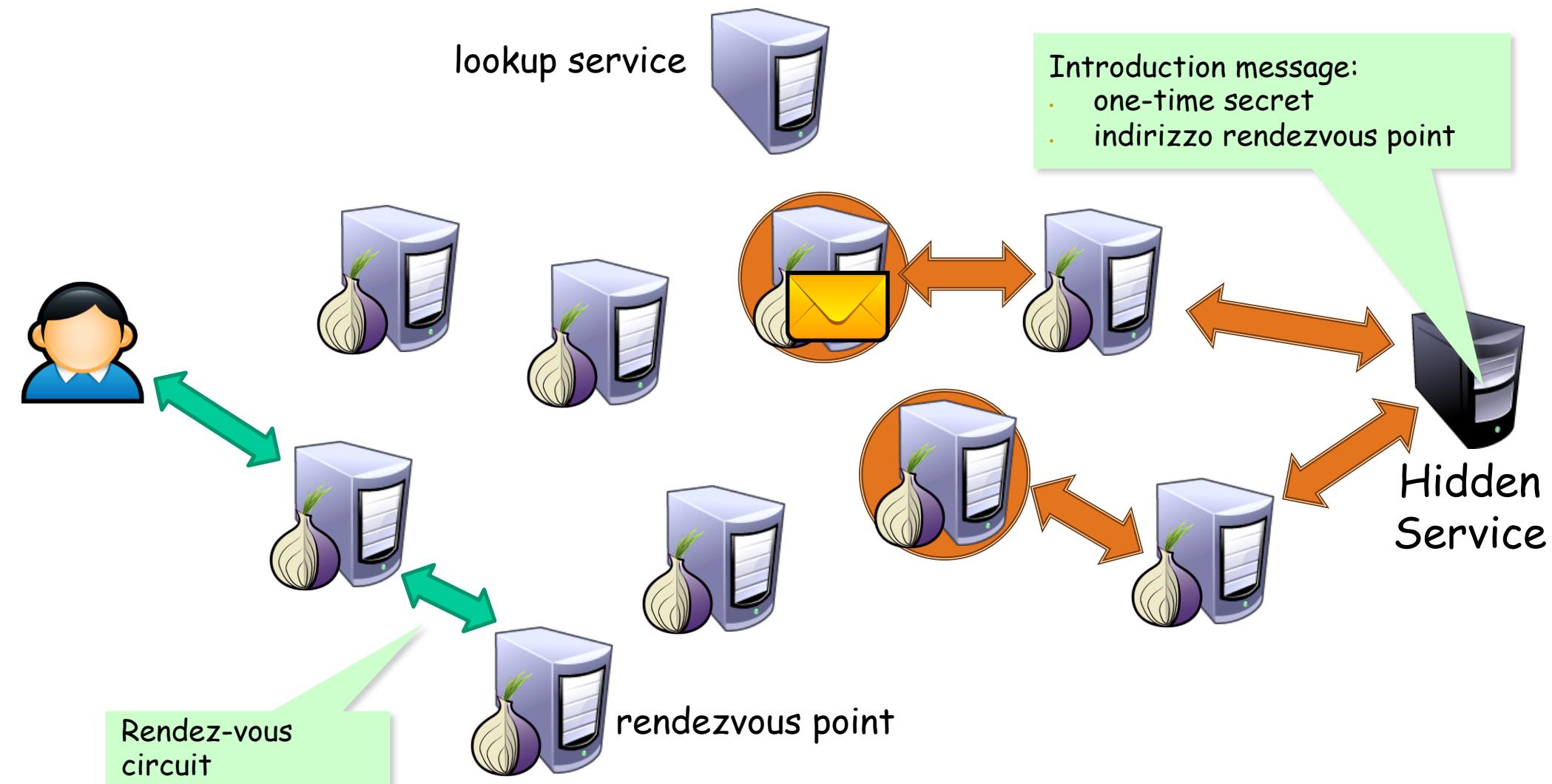
Comunicazione via
circuito Tor

rendezvous point



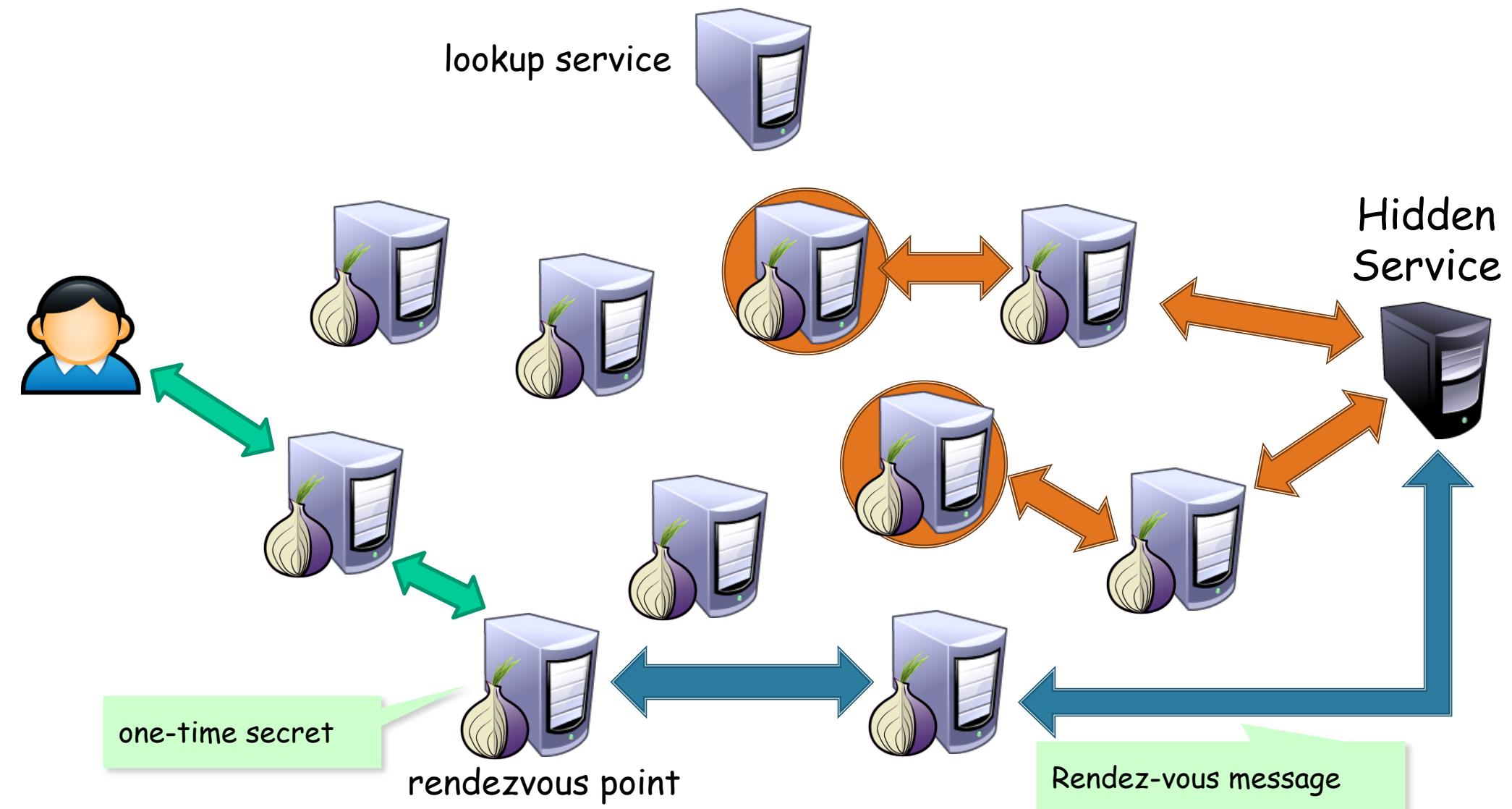
il client costruisce un "introduction message" (cifrato con la chiave pubblica dell'hidden service), contenente l'indirizzo del rendezvous point ed il "one-time secret" e lo invia a uno degli introduction point, chiedendo che venga consegnato all'hidden service.

Hidden Services



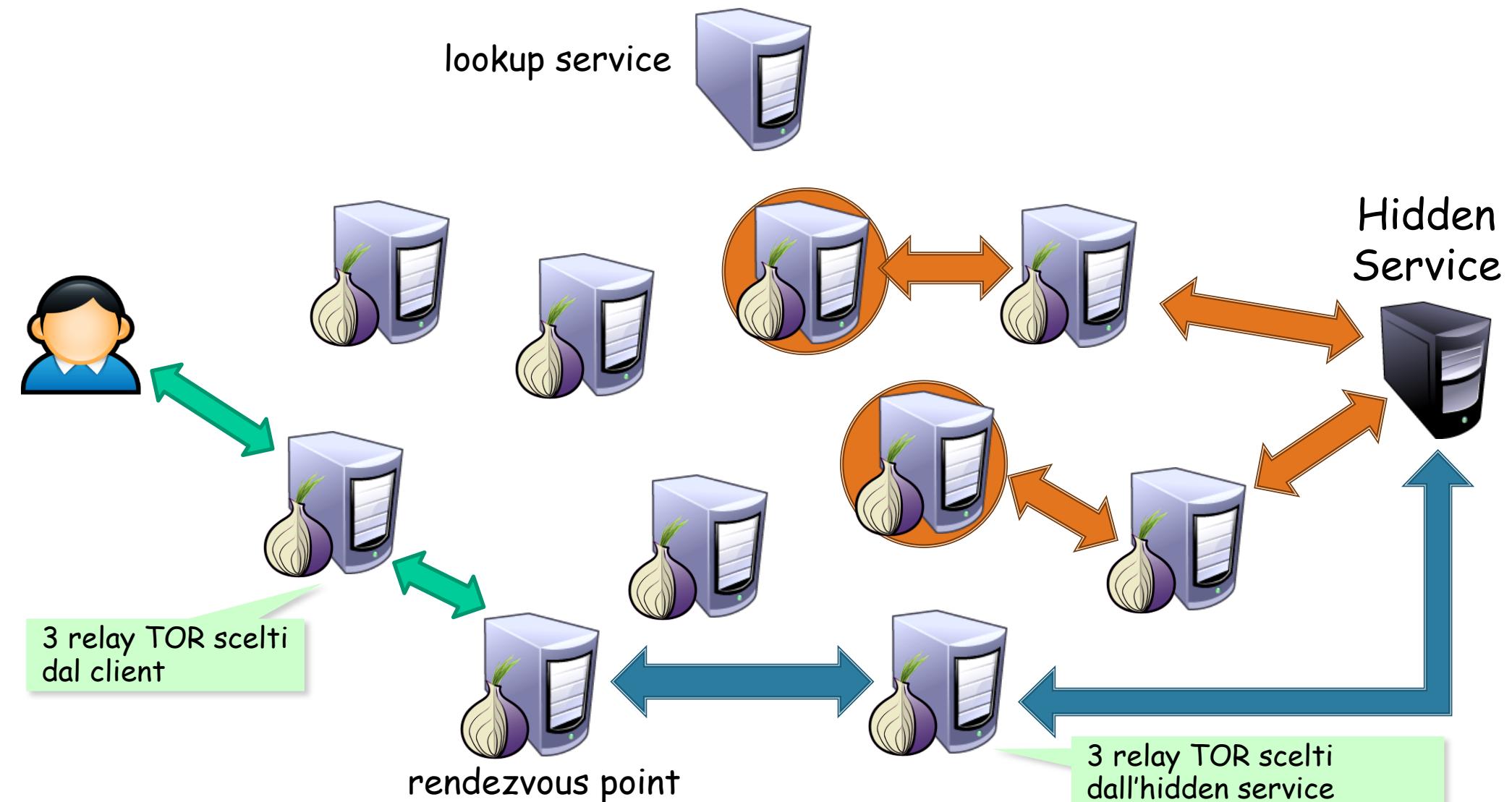
L'introduction point inoltra il messaggio di introduzione ricevuto dal client all'hidden service che lo decifra e scopre l'indirizzo del rendezvous point ed il "One-time secret" contenuto.

Hidden Services



L'hidden service crea un circuito verso il rendezvous point e gli invia il “One-time secret” in un rendezvous message.

Hidden Services

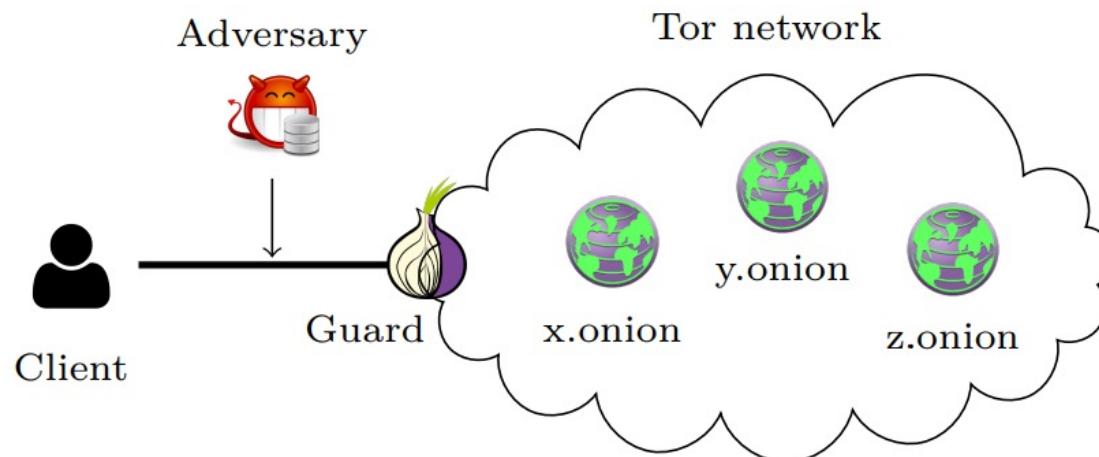


Il rendezvous point notifica al client che la connessione è stata stabilita con successo

Il client e l'hidden service possono usare i loro circuiti verso il rendezvous point per comunicare

Sicurezza Hidden Services

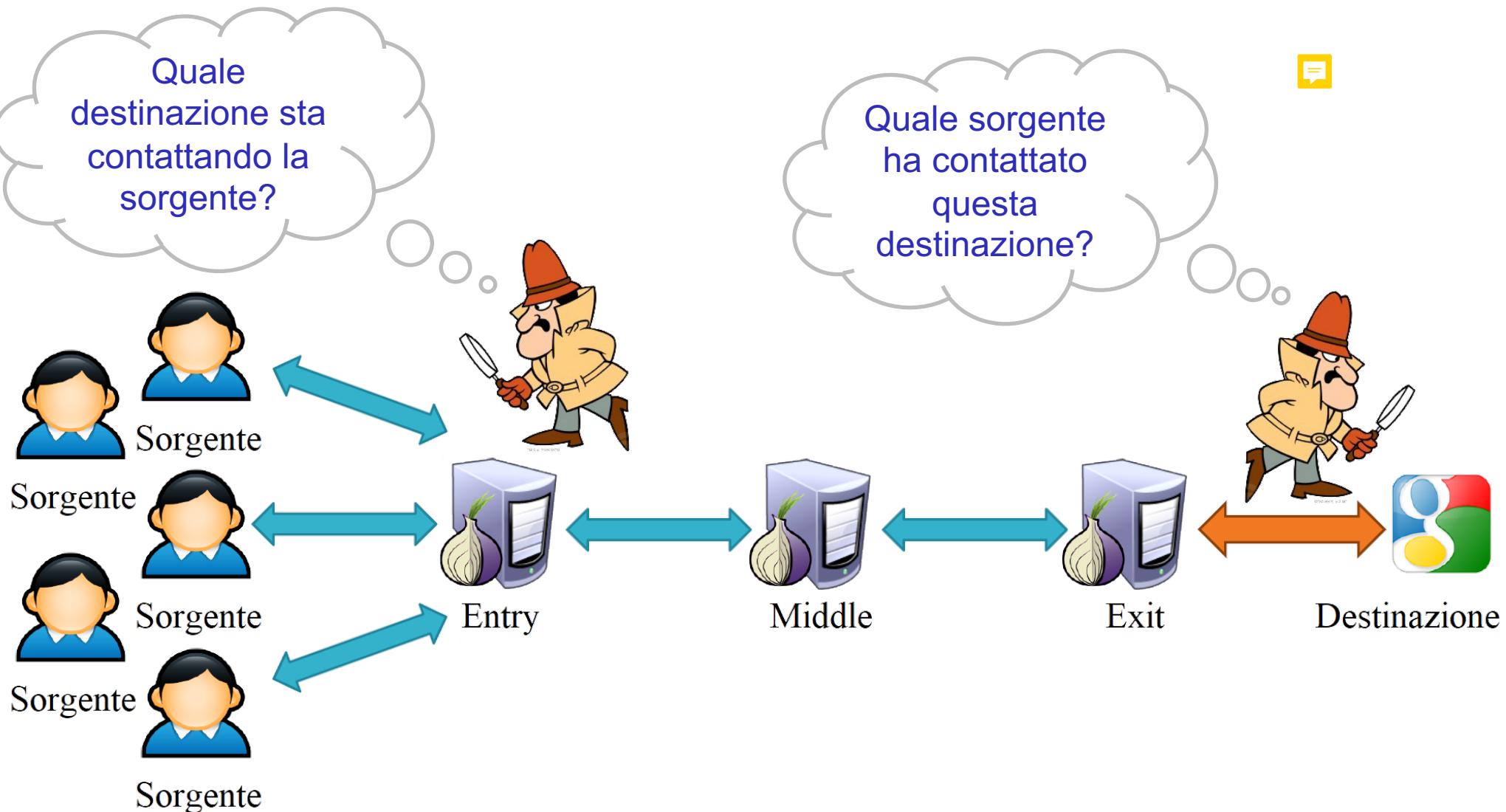
- Il lookup service e gli introduction point non conoscono l'IP dell'hidden service
- La comunicazione avviene sempre tramite circuiti Tor: nessuno può collegare l'invio dell'“introduce message” all' IP del client che rimane anonimo.
- Nel rendezvous point i dati decifrati provenienti dal client vengono nuovamente cifrati per essere consegnati al server e viceversa.
- La compromissione del rendezvous point permetterebbe di accedere ai dati “in chiaro” ma non comprometterebbe l'anonimato non potendo ottenere informazioni sul mittente e sul destinatario.



Attacchi a Tor

- Exit node maliziosi che monitorano il traffico
- Il contenuto fornito dall'exit node non è validato, può essere modificato dall'exit node
- Tor assicura anonimizzazione layer 3.. **e il resto?**
Application layer
 - javascript
 - ActiveX
 - plug-in vari (Shockwave)
 - vulnerabilità del browser

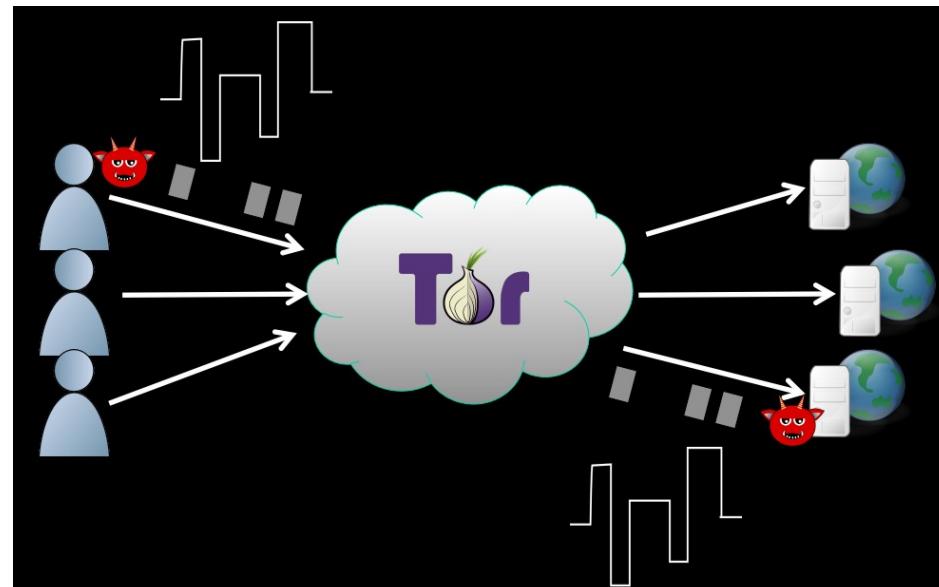
Attacchi a Tor



- **Obiettivo:** Associare sorgente e destinazione attraverso l'analisi del traffico ai due punti di osservazione

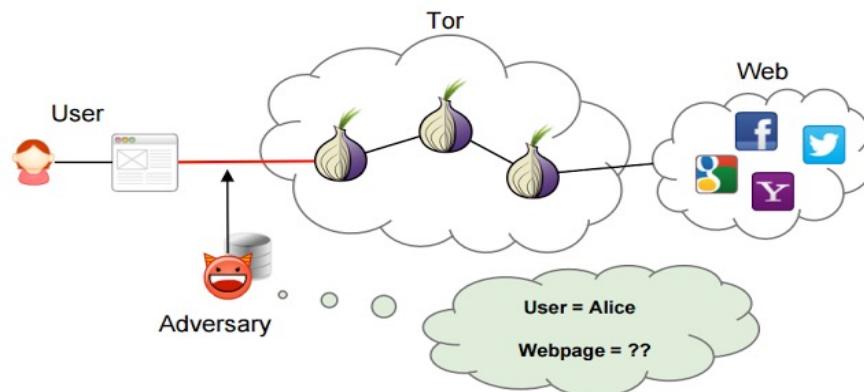
Correlazione

- **Correlazione temporale:** un attaccante che è in grado di osservare il traffico sull' Entry Guard e sull' Exit node, può prelevare informazioni sui pacchetti e sui tempi di invio/interarrivo e correlare il traffico osservato rompendo l'anonimato.
- **Correlazione sulla dimensione dei pacchetti:** tale attacco è simile al precedente ma, anzichè osservare i tempi di risposta in ingresso e in uscita, l'attaccante prende in considerazione i dati relativi al numero e alla dimensione di pacchetti scambiati. Se riesce a inferire pattern simili può correlare i due nodi.



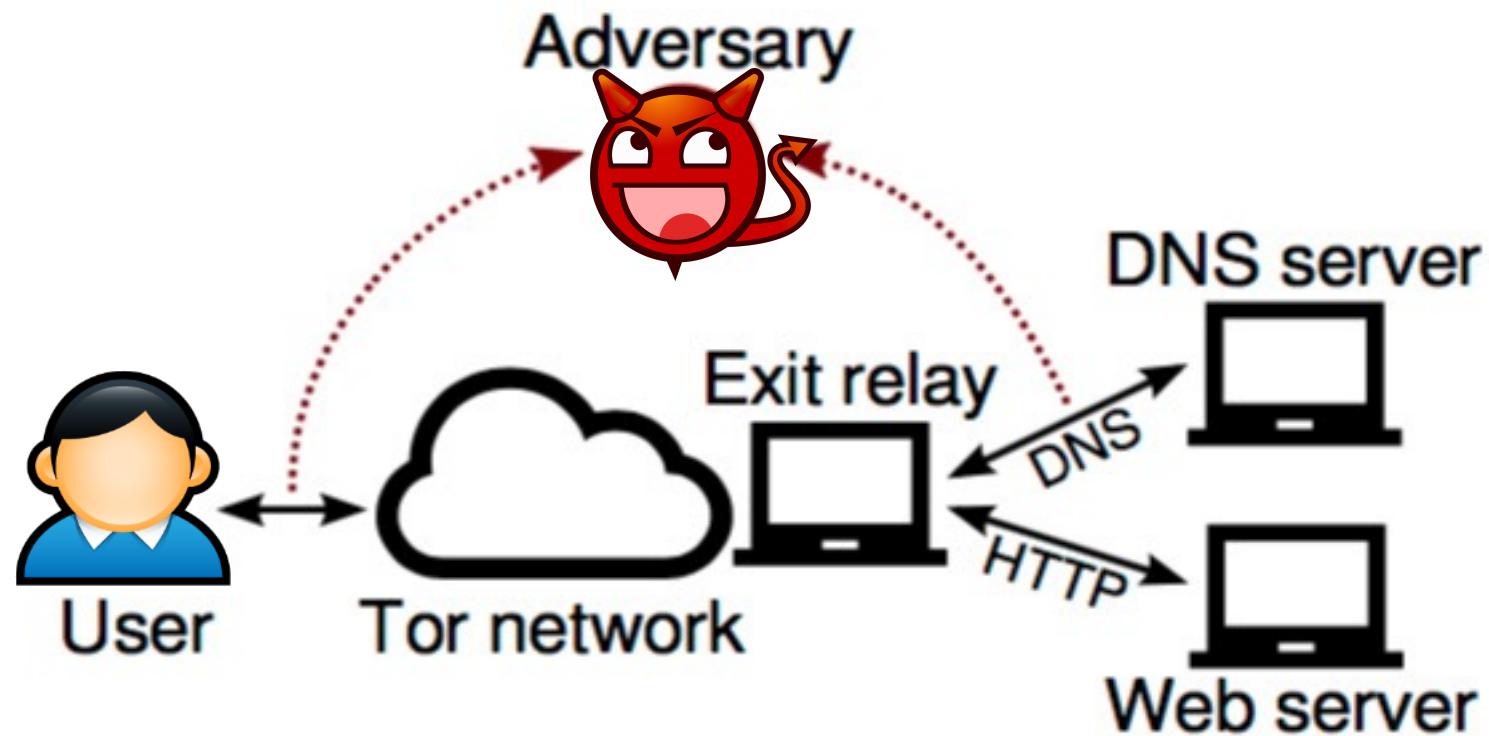
Osservazione pattern di traffico

- Osservare il traffico in uscita di un utente Tor anche se non rileva la sua destinazione o il contenuto dei messaggi trasmessi, può indicare il tipo di traffico:
 - Es. Il traffico HTTP è piuttosto intervallato e poco voluminoso. Non si tratta di un attacco facile da portare a termine poichè Tor accorpa più flussi di dati in un singolo circuito.
- Il traffico che attraversa la rete Tor è cifrato, ma non è detto che lo sia all'uscita dall'Exit node.
 - Es: Se il traffico in entrata è HTTP, quando l'Exit node toglie l'ultimo strato di cifratura, avremo di nuovo traffico HTTP in uscita e per l'attaccante risulterebbe molto facile catturare le richieste e prelevare informazioni importanti come: User Agent, COOKIE e parametri GET e POST.



Monitoraggio DNS

- Molti software continuano ad effettuare richieste DNS dirette, senza usare il proxy Tor.
- Ciò compromette l'anonimato perché rivela ad un osservatore le richieste DNS fatte, e quindi le destinazioni finali delle connessioni.

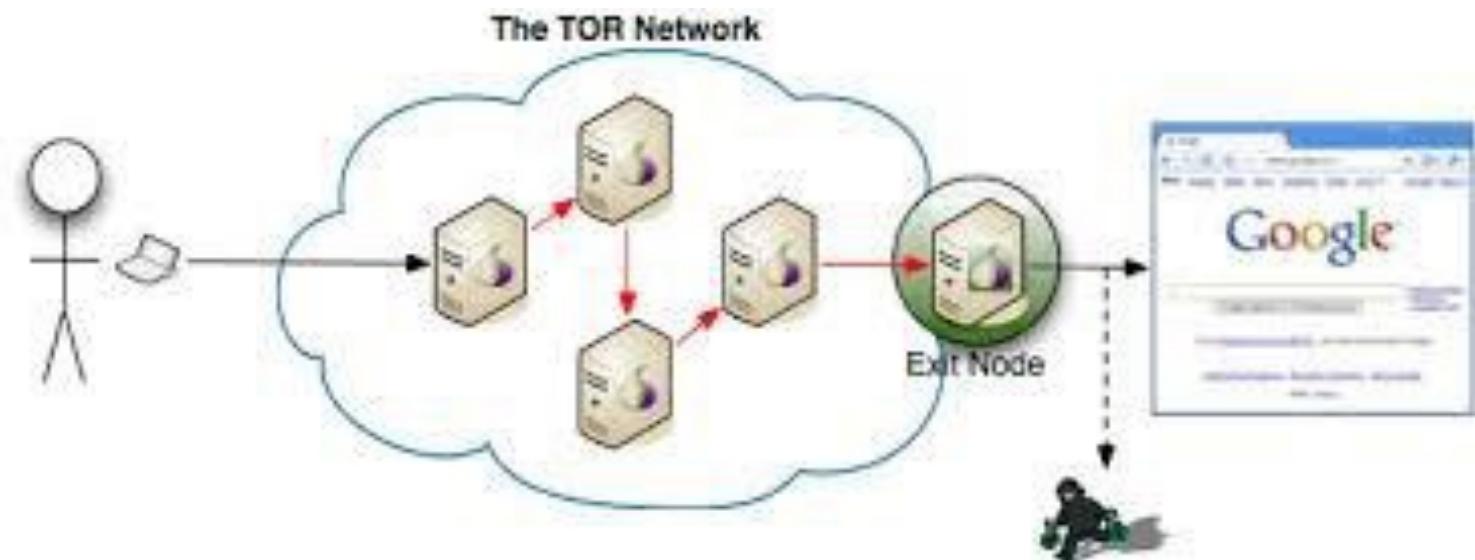


Estrazione comportamento

- Tor è altamente configurabile, è possibile configurare i tempi di rotazione dei circuiti, scegliere nodi in base alla larghezza di banda, ecc.
- Un attaccante può clusterizzare gruppi di utenti osservandone il comportamento.
- I gruppi con minor numero di individui sono facilmente identificabili.

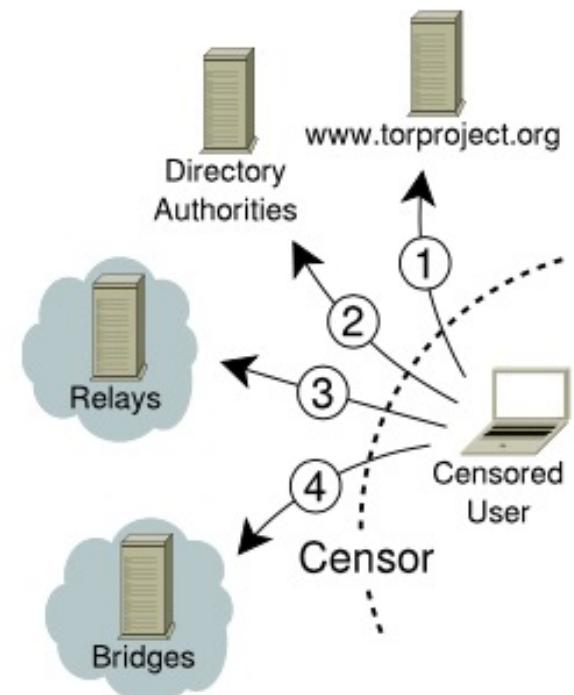
Fingerprinting

- L'attaccante conosce un set di dati riguardo i tipi di browser e le loro risposte, la taglia dei file e i pattern di accesso verso alcuni siti web, dunque, confronta queste informazioni con il traffico generato da un nodo client.
- L'attaccante è in grado di correlare l'ingresso con l'uscita se la generazione di traffico combacia con i pattern noti.



Blocco rete Tor

- Bridges potrebbero non bastare per evitare di essere bloccati
 - *Deep Packet Inspection classifica i flussi del traffico Internet a secondo dei protocolli per riconoscere e filtrare i frames Tor*
 - Possibile bloccare tutto il traffico cifrato
- Strategia governo Iran (febbraio 2012):
 - Deep Packet Inspection
 - Blocco degli indirizzi IP e porte
 - ad es., 86.59.30.36 (torproject.org), porta 443 (https)
 - Filtro con keyword
 - ad es., “tor”



Offuscamento traffico Tor

Tor usa **pluggable transport** per la comunicazione tra client e bridge

- Traffico Tor trasformato in altro tipo di traffico
- Un offuscatore trasforma il traffico Tor in modo che appare come altro protocollo
 - BitTorrent, HTTP, streaming audio, etc.
- Un deoffuscatore dalla parte del ricevente estrae il traffico Tor



Offuscatori:

- obfs2
- obfs3
- FTE
- scramblesuit
- meek
- Flashproxy

Mixing di traffico

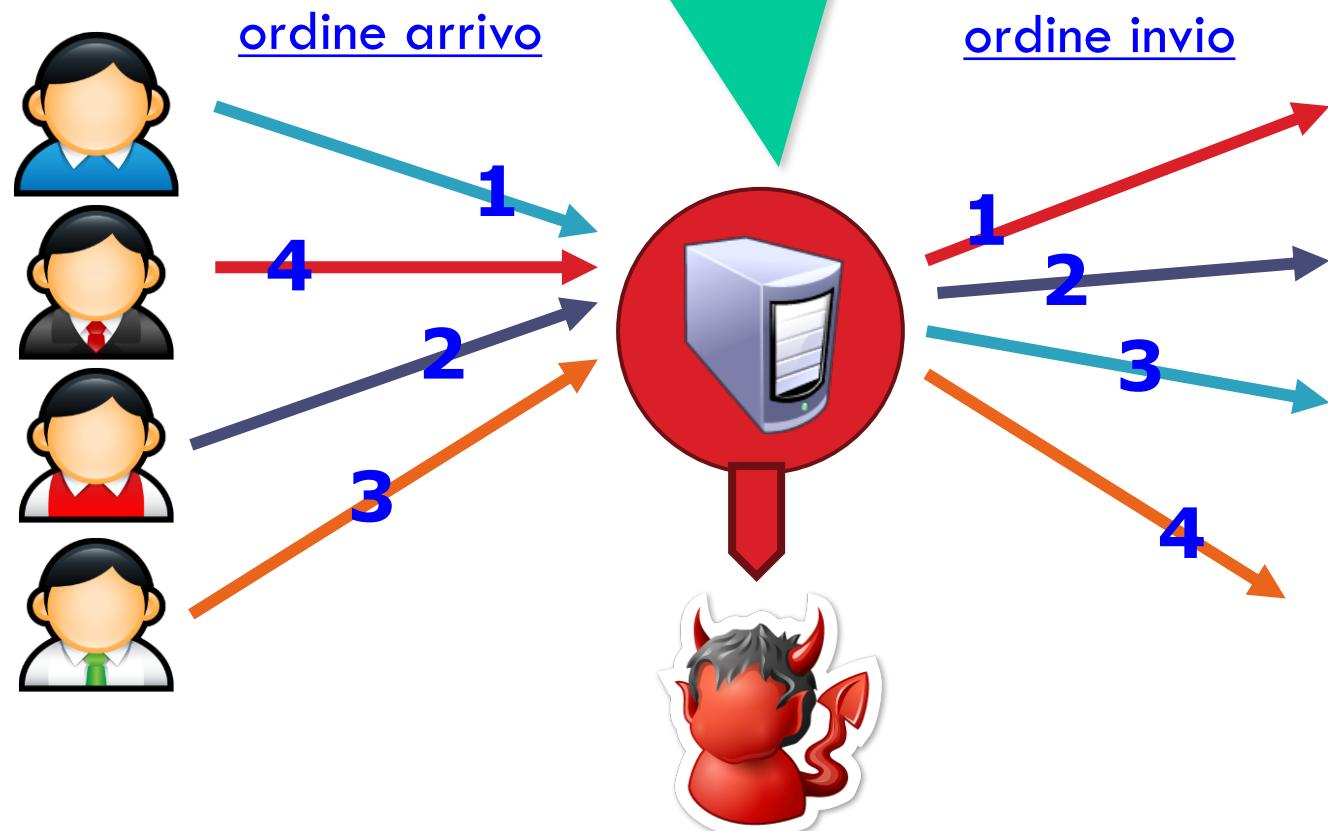
- Evitare attacchi basati sullo studio dell'andamento dei flussi di traffico per correlare sorgente e destinazione

- Messaggi artificialmente ritardati
- Si rende difficile la correlazione temporale

- Problemi

- Molto traffico
- Aggiunta latenza ai flussi

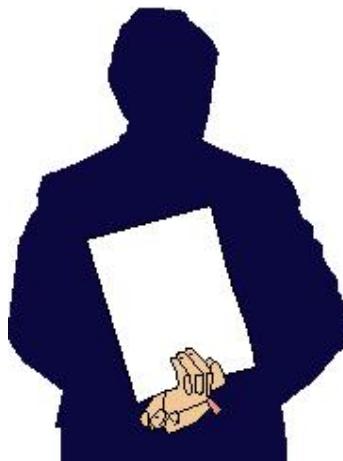
> Mix colleziona messaggi for t secondi
> Messaggi sono mischiati casualmente ed inviati in un ordine differente



Pubblicazione anonima

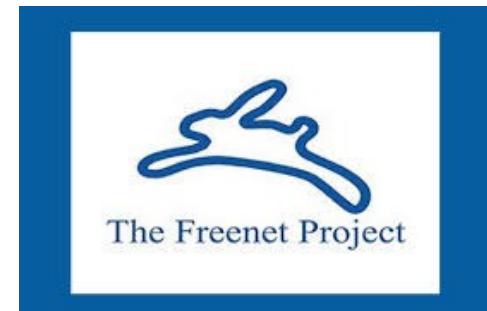
Un sistema di pubblicazione anonima deve garantire:

- Privacy per **chi pubblica** i contenuti
- Privacy per **chi fruisce** dei contenuti
- **Protezione** dei contenuti
- Resistenza ad attacchi volti alla **censura**



Freenet

- L'unico sistema di pubblicazione anonima sopravvissuto e disponibile al pubblico
- Non è possibile risalire a chi pubblica, memorizza e richiede i contenuti
- I gestori dei nodi non possono conoscere il contenuto dei dati da loro memorizzati (protezione legale in alcuni Stati)
- Nulla può essere cancellato (nemmeno dall'autore)
- Non ha funzionalità di ricerca, ma solo indici dei contenuti
- E' un protocollo, non un'applicazione



<http://freenetproject.org/>

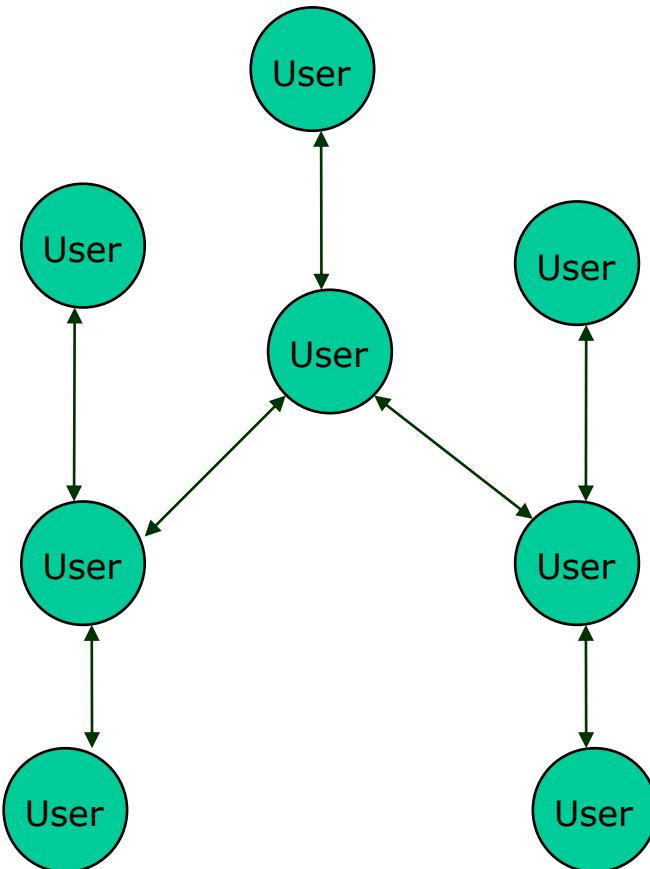
Freenet

- Rete adattativa di nodi peer-to-peer che si interrogano reciprocamente per immagazzinare e recuperare file di dati identificati da nomi (chiavi) indipendenti dalla locazione.
- Sistema per scrivere e leggere file da Internet senza che si possa risalire a chi li ha scritti, chi li conserva sul disco e chi li recupera.
- Formata da server (nodi) paritetici; i nodi normalmente includono un proxy che permette di accedere al server con un form, utilizzando il protocollo HTTP.

Freenet

- Freenet, spezzetta, crittografa, duplica, disperde i contenuti del file, e riesce ad eseguire l'operazione inversa per recuperarli.
- Più un'informazione è richiesta più si moltiplica sui nodi, informazioni non richieste a lungo si cancellano
- Adattività della rete - il grafo delle connessioni logiche tra i nodi evolve nel tempo verso una stabilità ed efficienza maggiore.
- Il sistema non è completamente deterministico, e non consente di provare che un certo file proviene da un nodo locale e non da un altro nodo della rete

Architettura FreeNet



- Struttura overlay pura (peer to peer)
- Ogni utente agisce in maniera indipendente
- Non esiste un directory server centrale
- I nodi si scambiano direttamente informazioni in una logica query/response
- Resilienza della rete - l'informazione non puo' essere rimossa da Freenet ma solo lasciata "morire" di morte naturale.
- Comportamento "ecologico" della rete - l'informazione che viene richiesta si moltiplica su più nodi e si "avvicina" ai nodi che la richiedono

Comunicazione e routing in Freenet

- Comunicazione autenticata e cifrata fra nodi
- I nodi comunicano tra loro con un semplice protocollo connection-oriented chiamato FNP (Freenet Network Protocol)
- I client applicativi che vogliono utilizzare i servizi Freenet di un nodo locale utilizzano un altro protocollo chiamato FCP (Freenet Client Protocol)
- I nodi comunicano tra loro sulla base di una conoscenza locale dinamica dei nodi limitrofi
- Instradamento key-based: ogni nodo richiede una chiave, nell'ordine, ai nodi limitrofi
- Ogni nodo costruisce dinamicamente una routing table che contiene gli indirizzi degli altri nodi e le chiavi che si suppona posseggano

Gestione contenuti in Freenet

- I contenuti (files) sono **suddivisi** in atomi chiamati “chiavi”
 - KSK (keyword signed key)
 - SSK (signed subspace key)
 - CHK (content hash key)
 - MSK (map space key)
- Nota : la funzione hash utilizzata è lo SHA-1 a 160 bit mentre l’algoritmo asimmetrico di cifratura è il DSA
- Le chiavi vengono crittografate e **distribuite casualmente** tra vari server
 - SSK@rBjVda8pCKq04jUurIAb8IzAGcPAgM/TFE//thelist.html
 - CHK@nor6G5qLSxiKsy1EUiKV~5lBLH4NAwI,OejC1NKzRVt1GkRpW0f4Q
(06 Amerika - America.mp3)
 - KSK@text/philosophy/sun-tzu/art-of-war

La chiave **CHK** (content hash key)

- Una chiave *CHK* è un hash del documento:
 - confrontando chiave e documento, il nodo può controllare se la trasmissione è avvenuta correttamente.
 - questo tipo di chiave viene usato per le trasmissioni di dati e per la loro lettura.
 - un nodo malevolo che volesse modificare un documento verrebbe immediatamente scoperto dal nodo successivo, grazie al controllo della chiave.
 - file che hanno la stessa chiave sono uguali; file anche solo leggermente diversi hanno chiavi diverse

La chiave CHK (content hash key)

- La chiave è specificamente derivata dall'hash SHA-1 del contenuto del file corrispondente.
- Tutti i file sono associati a chiavi CHK
- Il file viene inoltre criptato utilizzando una chiave generata in modalità random
- Vengono pubblicati sia l'hash che la chiave di decrittazione
 - Esempio -> freenet:CHK@foto.gif
 - Una volta inserito, il dato potrà essere richiesto fornendo la seguente stringa :
 - CHK@zdfaGT....,fpR12.....

La chiave SSK (signed subspace key)

- Le chiavi SSK sono basate sul concetto di crittografia asimmetrica, in particolare sul Digital Signature Algorithm (DSA).
 - I documenti inseriti con questo tipo di chiave vengono firmati dagli autori, in modo che tutti possano verificare l'integrità del documento che leggono.
- Le chiavi possono essere usate per creare uno pseudonimo all'interno di Freenet mantenendo l'anonimato
 - Consentono la creazione di una «namespace» personale
 - permettono l'aggiornamento dei documenti solo da parte di chi li ha inseriti.

La chiave SSK (signed subspace key)

- Per costruire un “namespace” personale su Freenet
 - Creiamo una coppia di chiavi pubblica/privata di tipo SSK
 - Utilizzeremo la chiave privata per inserire documenti “sotto” il nostro namespace
 - Pubblicheremo la nostra chiave pubblica per rendere accessibili i file pubblicati
 - Esempio:

SSK@public_key/musica/song1.mp3

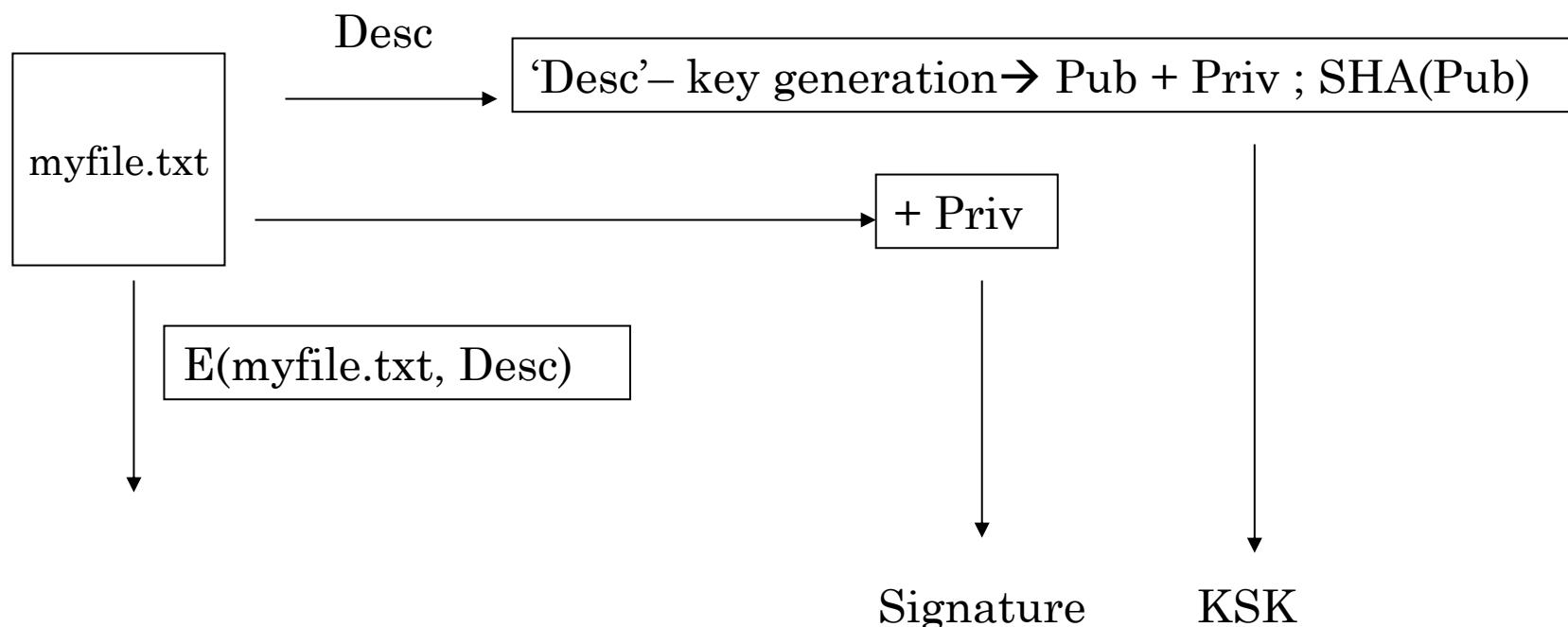
SSK@public_key/musica/song2.mp3

La chiave KSK (keyword signed key)

- E' la chiave più semplice e user-friendly
 - Esempio: freenet:KSK@myfile.txt
- Le chiavi KSK, sottocategoria delle SSK, vengono generate a partire da una frase chiave
 - i documenti segnati con questo tipo di chiave possono essere ottenuti e decriptati se e solo se si è a conoscenza della frase chiave.
- I file in Freenet sono associati solo a chiavi CHK.
- La chiave KSK è solo un puntatore che permette di recuperare piu' facilmente il contenuto di una chiave CHK

La chiave KSK (keyword signed key)

- La stringa descrittiva (mio_file) viene utilizzata per generare una coppia di chiavi pubblica/privata (DSA)
- La chiave pubblica viene utilizzata per produrre l'hash associato al file inserito (SHA-1)
- La chiave privata viene utilizzata per “firmare” il file inserito.



La chiave MSK (map space key)

- Risolve il problema di aggiornare contenuti che non possono essere cancellati (freesite)
- Viene utilizzata come home page di un freesite
- Puo' essere acceduta direttamente ...
`freenet:MSK@SSK@11...11/nomesito//`
- ... o indirettamente, e Freenet seleziona quella che si riferisce alla data corrente
`freenet:MSK@SSK@11...11/yyyyyyymmddhhmmss-nomesito//`

La chiave MSK (map space key)

- Le chiavi MSK vengono inserite in batch prima della data a cui si riferiscono
- Il meccanismo così creato si chiama DBR - **date** base redirect (**date**, non data !)
- Le chiavi MSK ormai vecchie e che non vengono più richieste “muoiono” di morte naturale
- La chiave MSK di un freesite contiene gli “indirizzi” delle chiavi CHK a cui i link di tutto il freesite si riferiscono

Boot di un nodo in Freenet

- Il nodo che effettua il boot deve conoscere almeno un nodo già' in rete tramite un metodo out-of-band.
- Il problema del boot di un nodo (conoscenza di un altro nodo affidabile a cui connettersi) non è risolto. Attualmente si utilizza una pagina web del Progetto Freenet o si fornisce un nodo manualmente (file "sendnode.ref")
- Poi il nodo genera un random seed per se stesso e lo invia a un nodo scelto a caso
- Il nodo ricevente crea un altro random seed facendo XOR con l'hash ricevuto e creando un nuovo hash
- Il nuovo nodo viene aggiunto nella routing table.
- I nodi "scoprono" altri nodi durante il funzionamento.

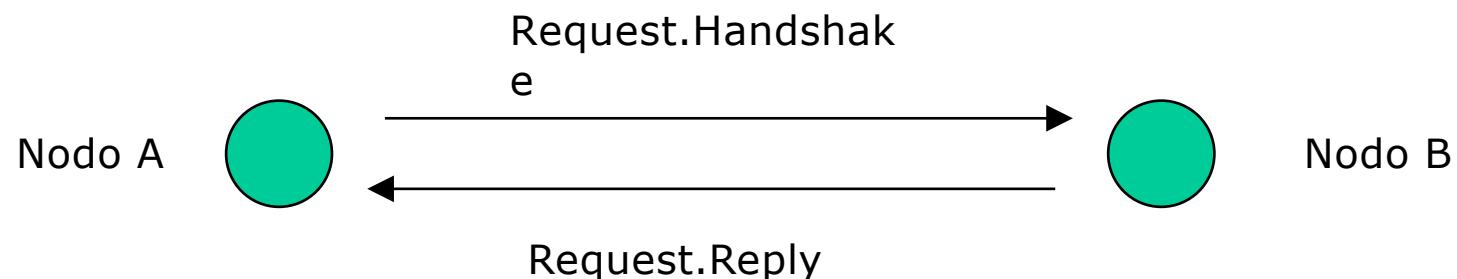
Accesso contenuti in FreeNet

- Una richiesta di contenuto (un file associato a una chiave ottenuta o calcolata) può passare per multipli nodi differenti.
- Se un nodo non ha il contenuto richiesto in cache rigira la query a uno dei suoi vicini che ha maggiori probabilità di averlo
 - I messaggi di richiesta formano via via una catena
- Esiste un meccanismo di timeout che impedisce la formazione di catene troppo lunghe
- La catena termina in presenza di un timeout o quando un nodo possiede e rimanda indietro il contenuto richiesto.
- I nodi in transito durante un reply possono fare caching dei contenuti

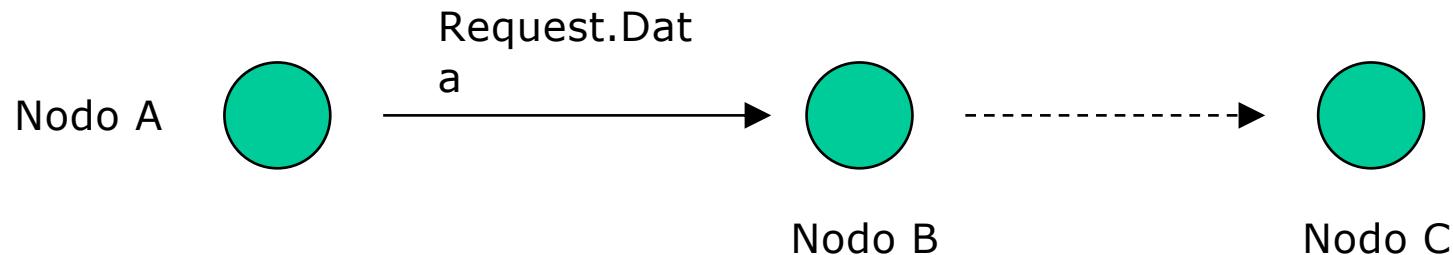
Accesso contenuti in FreeNet

- Le richieste di accesso a contenuto sono sempre relative a una chiave e constano di 2 fasi

Fase di Handshake

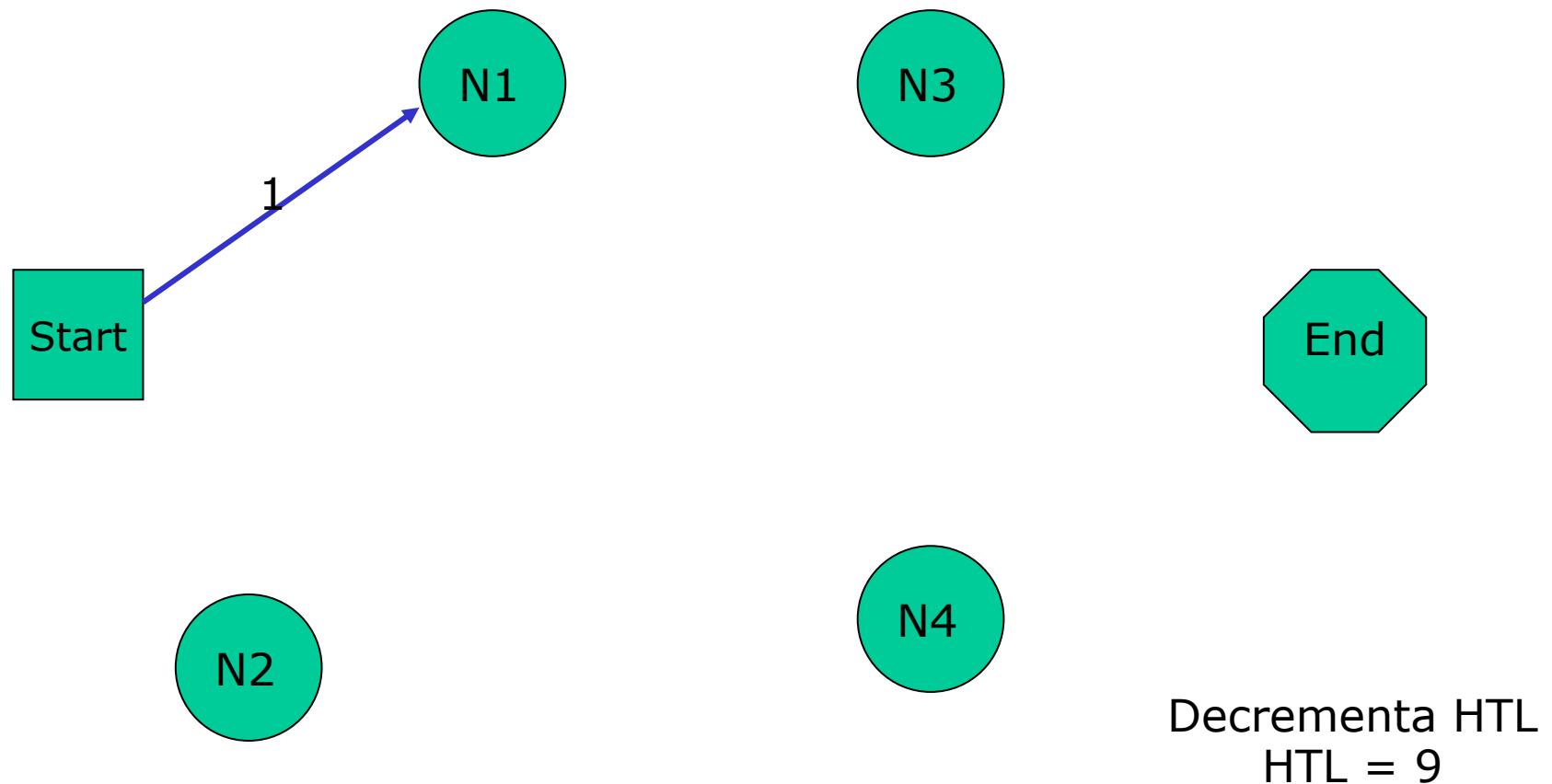


Fase di richiesta dati



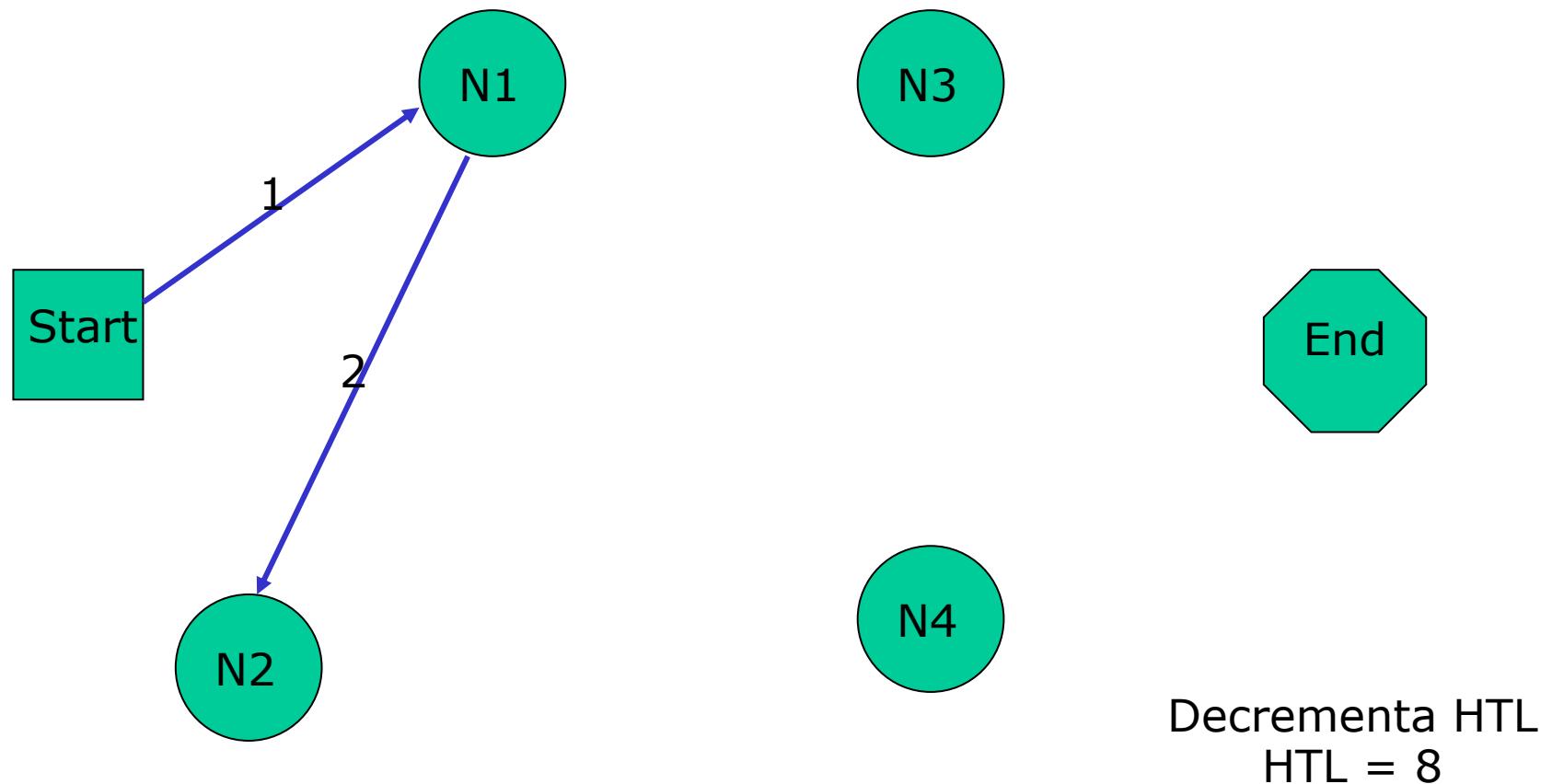
Accesso contenuti in FreeNet

- Il nodo Start invia la sua query al vicino N1



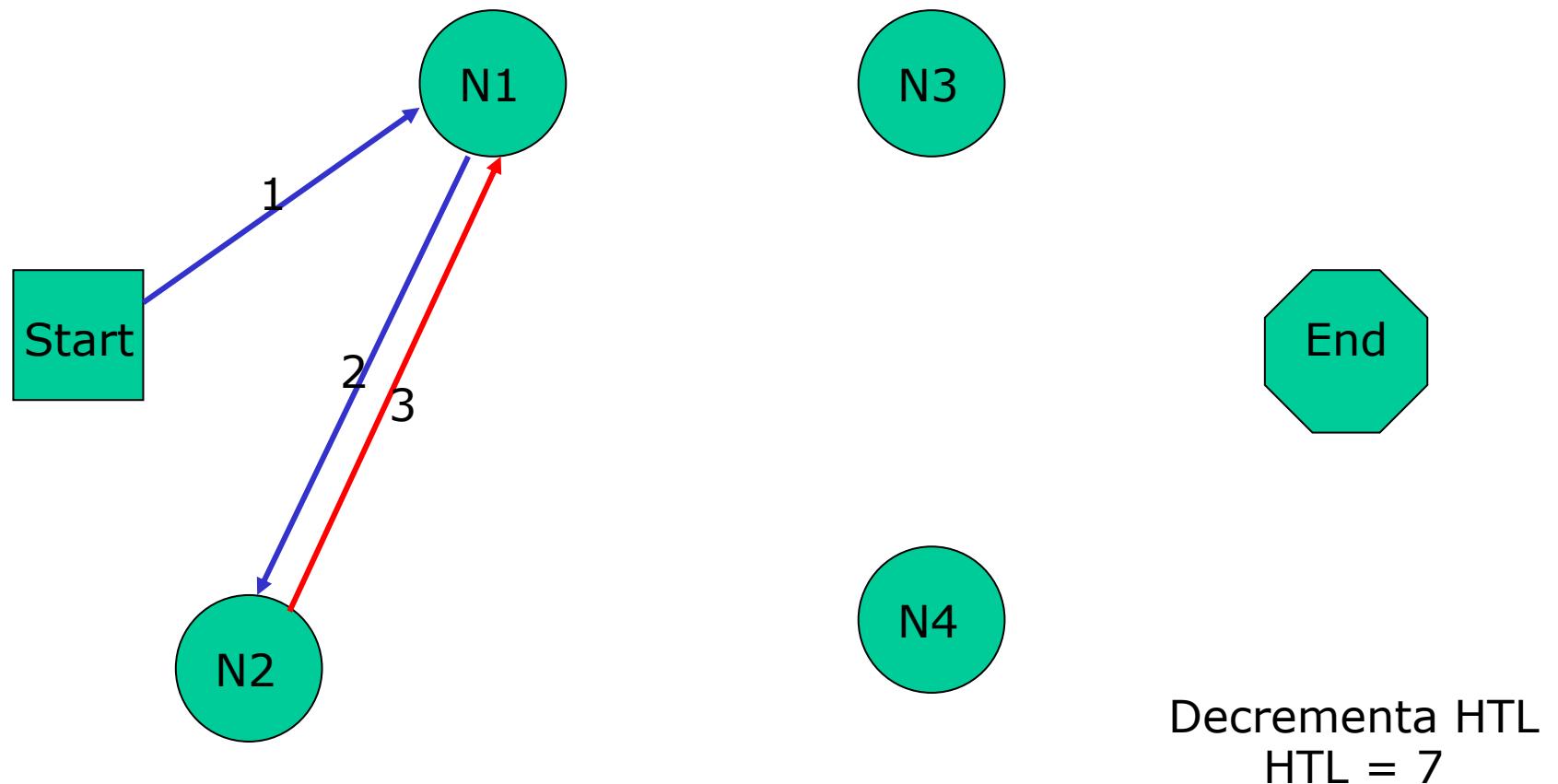
Accesso contenuti in FreeNet

- Il nodo N1 re-invia la query al vicino N2



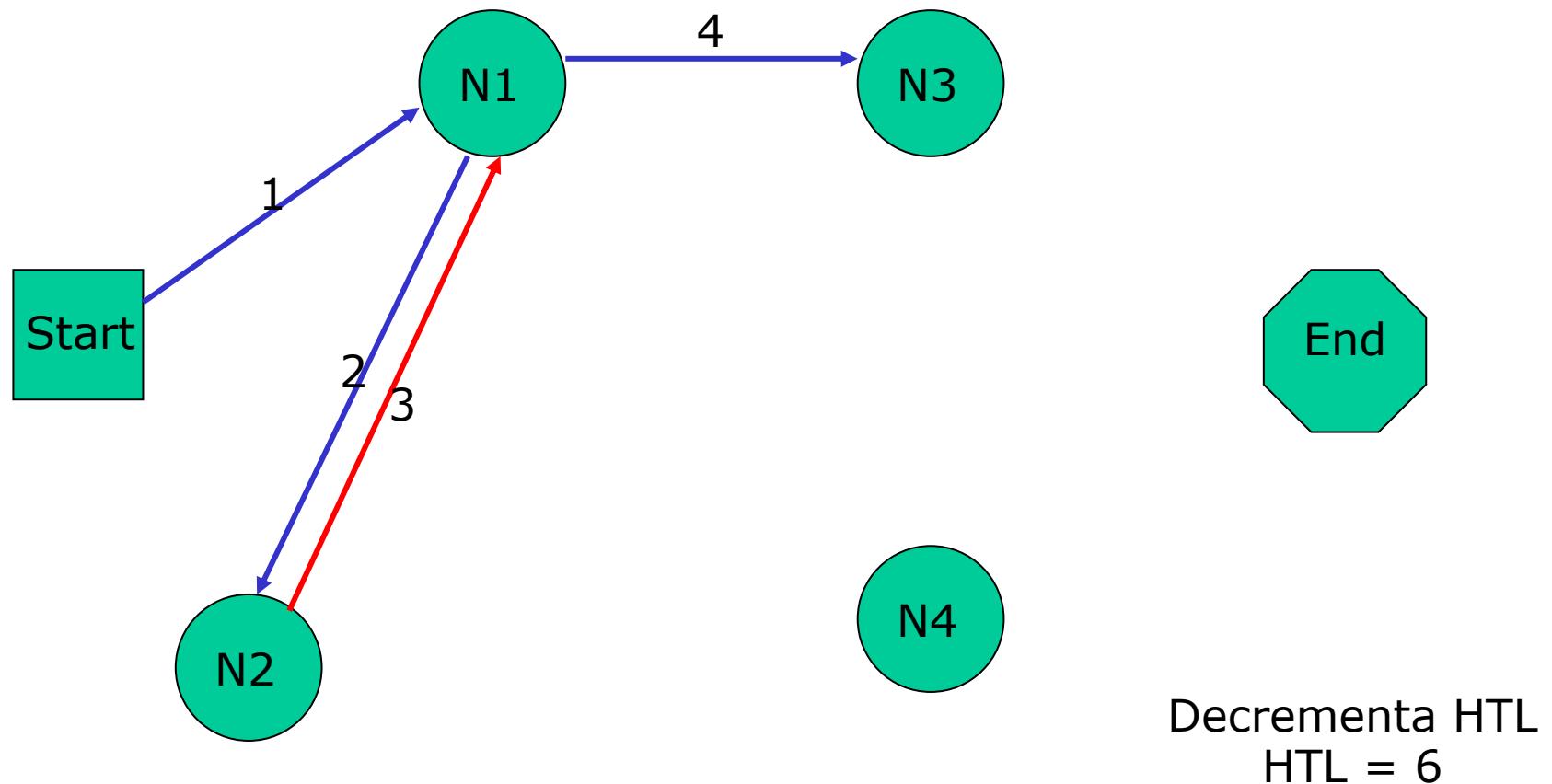
Accesso contenuti in FreeNet

- ... che a sua volta non avendo scelte la reinoltra al suo unico vicino N1



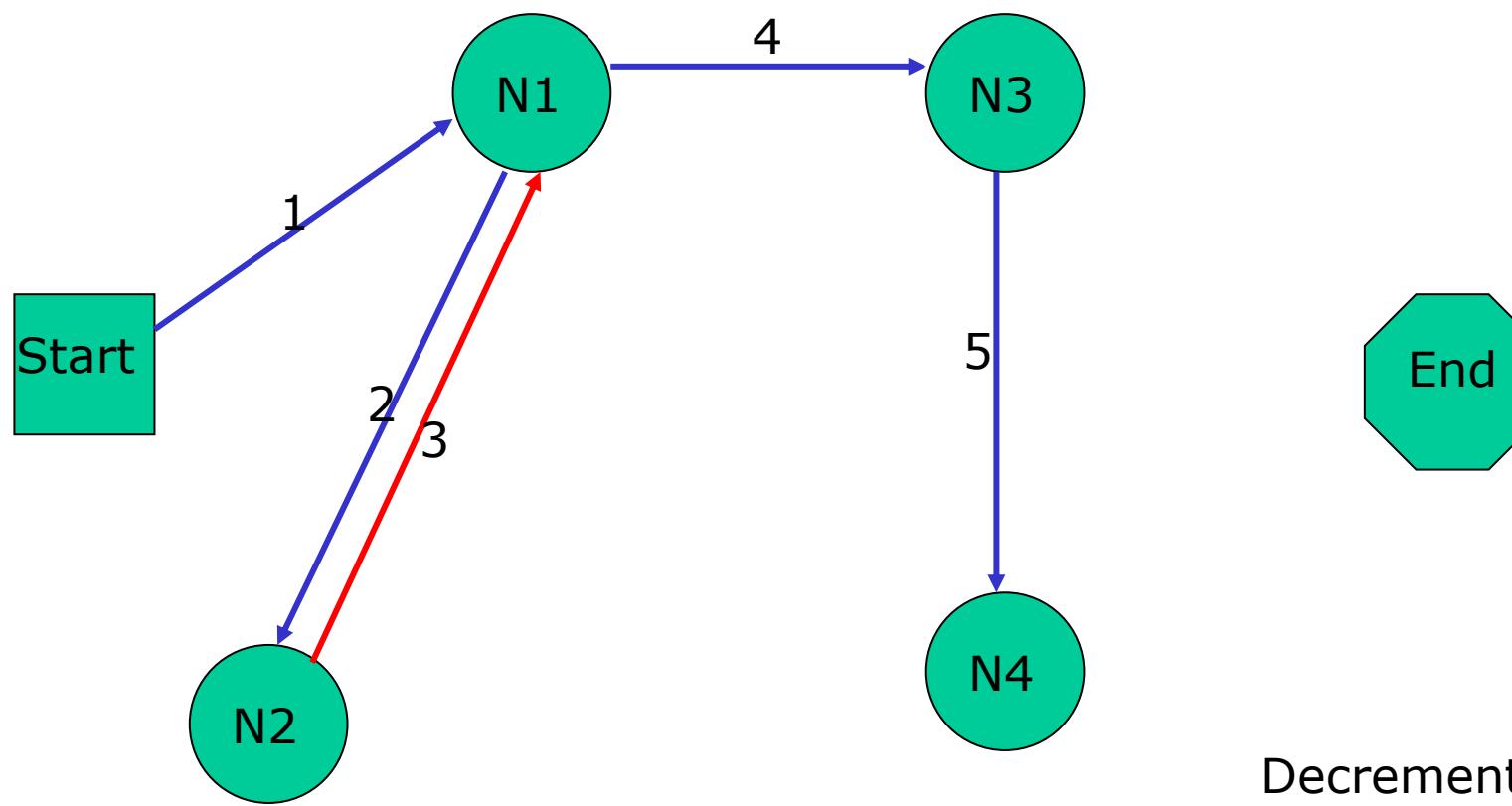
Accesso contenuti in FreeNet

- ... allora nodo N1 re-invia la query al vicino N3



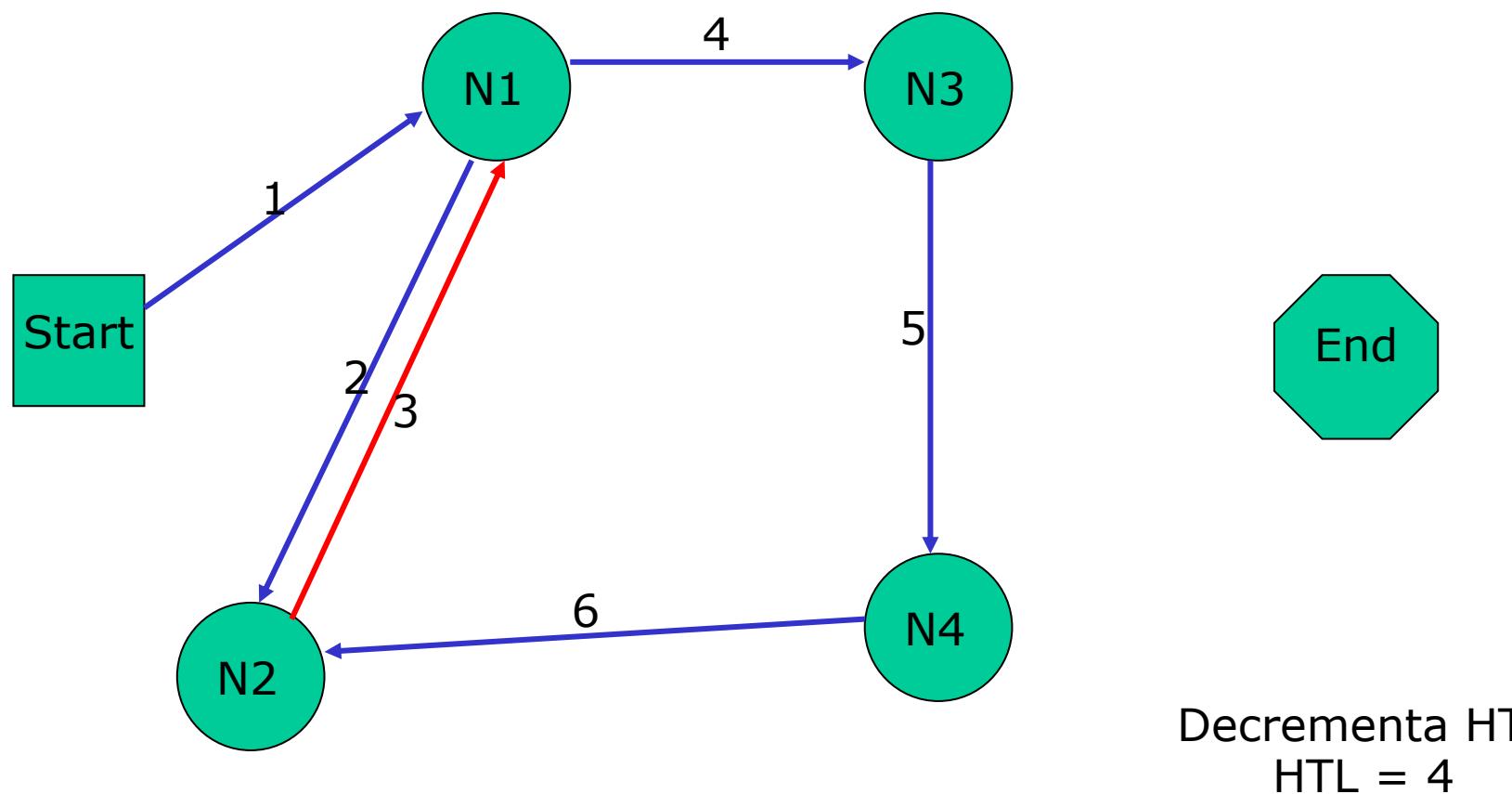
Accesso contenuti in FreeNet

- ... che a sua volta la reinoltra al vicino N4



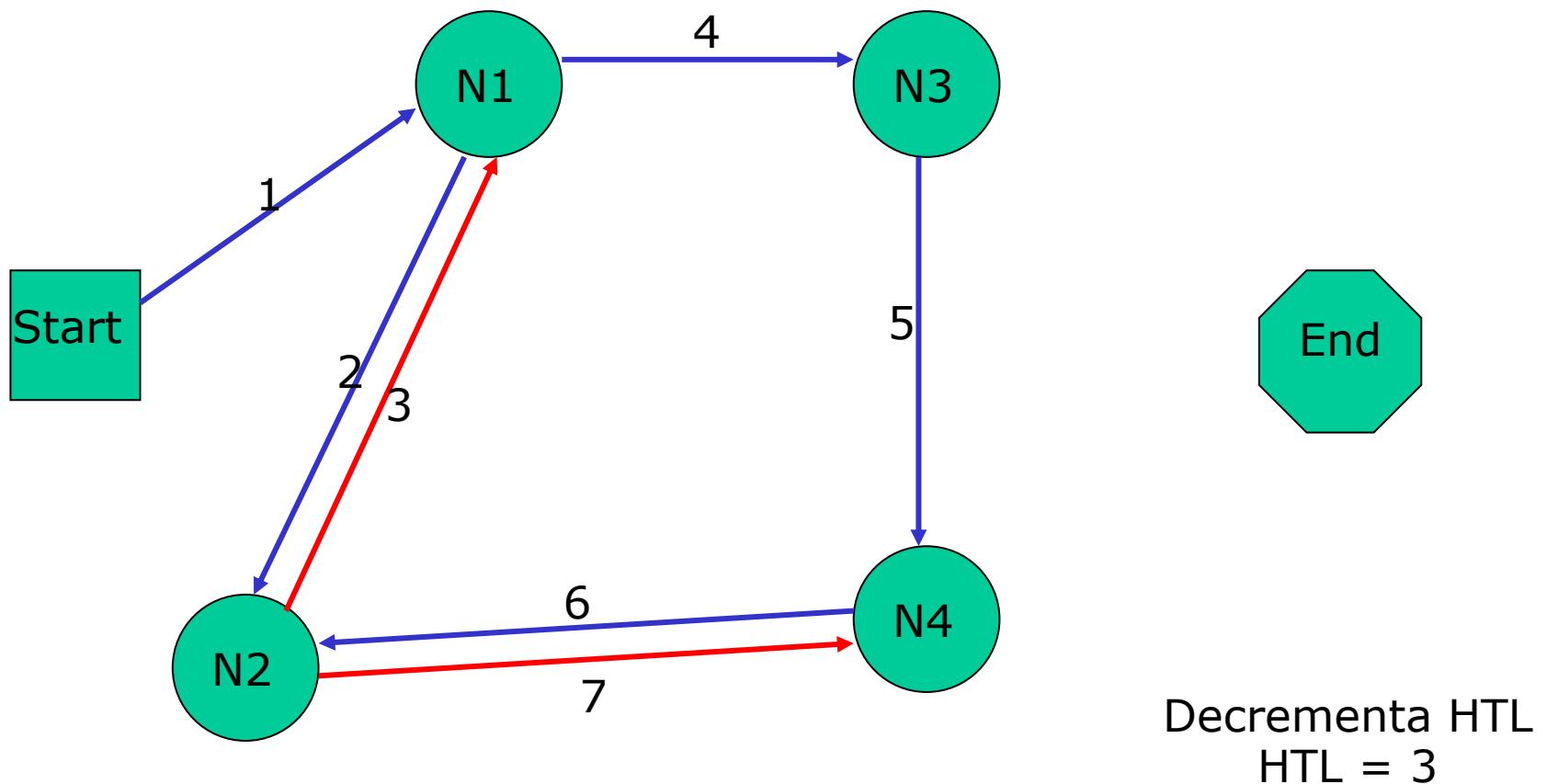
Accesso contenuti in FreeNet

- Il nodo N4 re-invia la query al vicino N2



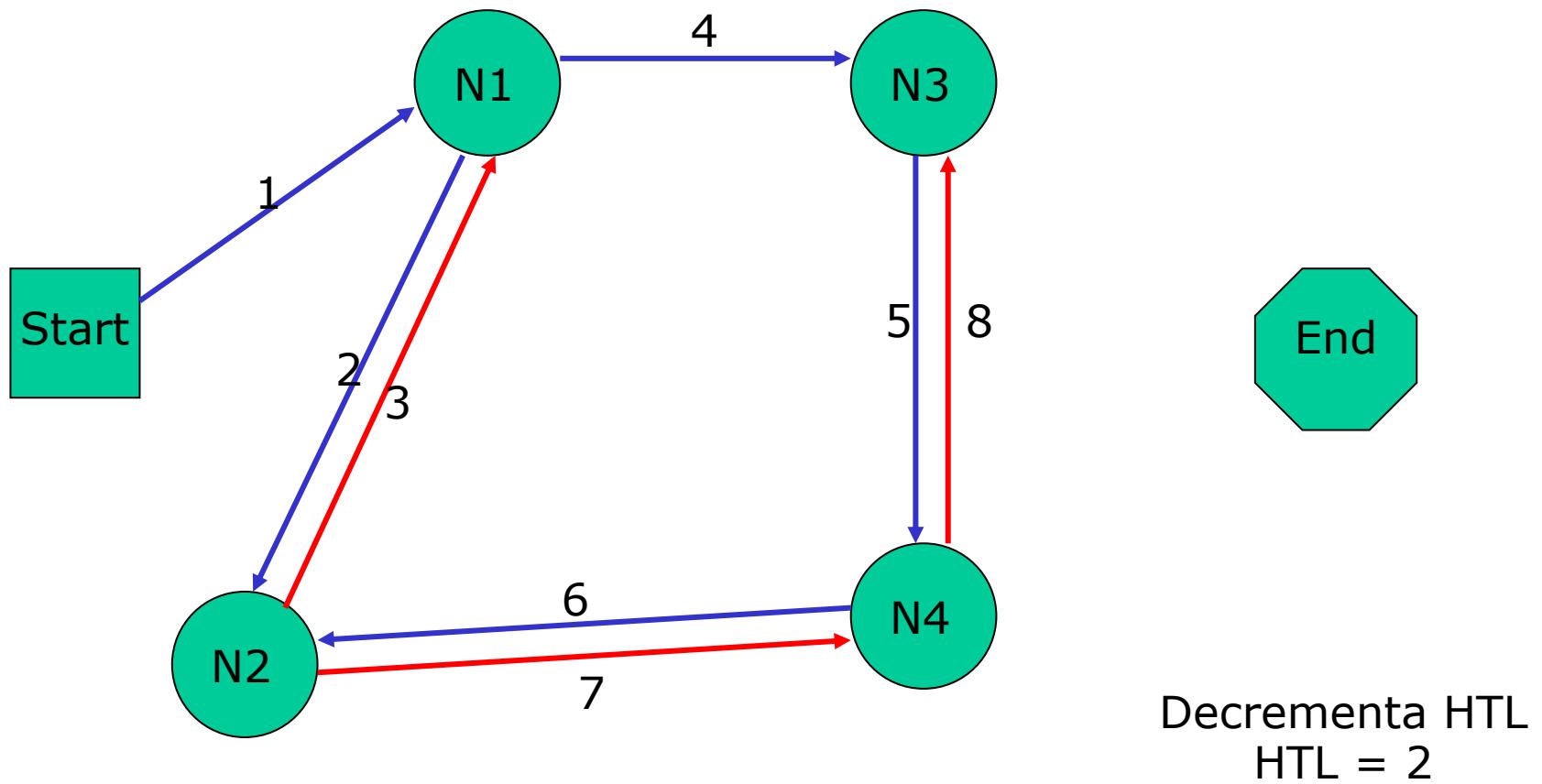
Accesso contenuti in FreeNet

- Il nodo N2 re-invia la query al vicino N4



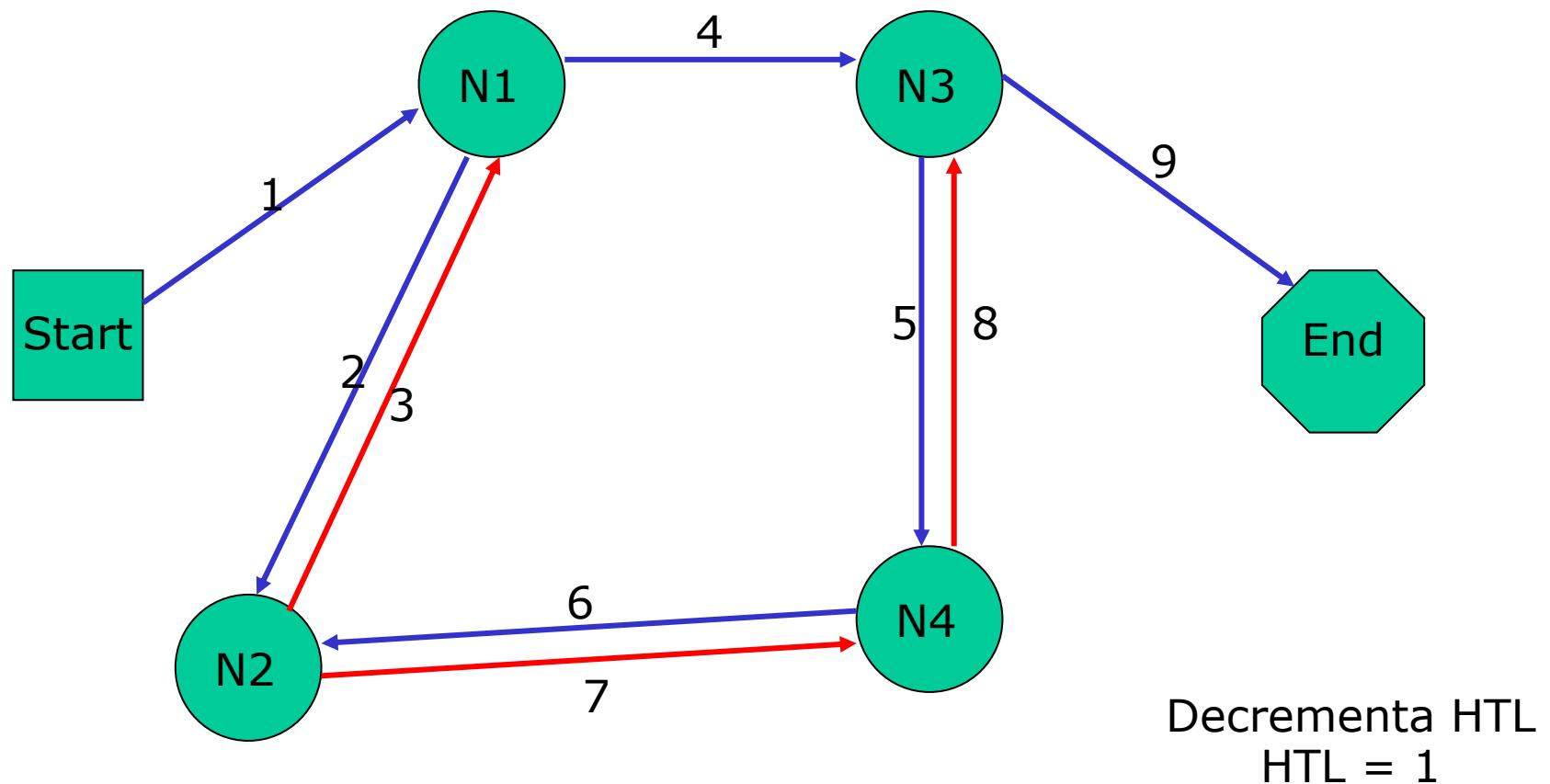
Accesso contenuti in FreeNet

- Il nodo N4 re-invia la query al vicino N3



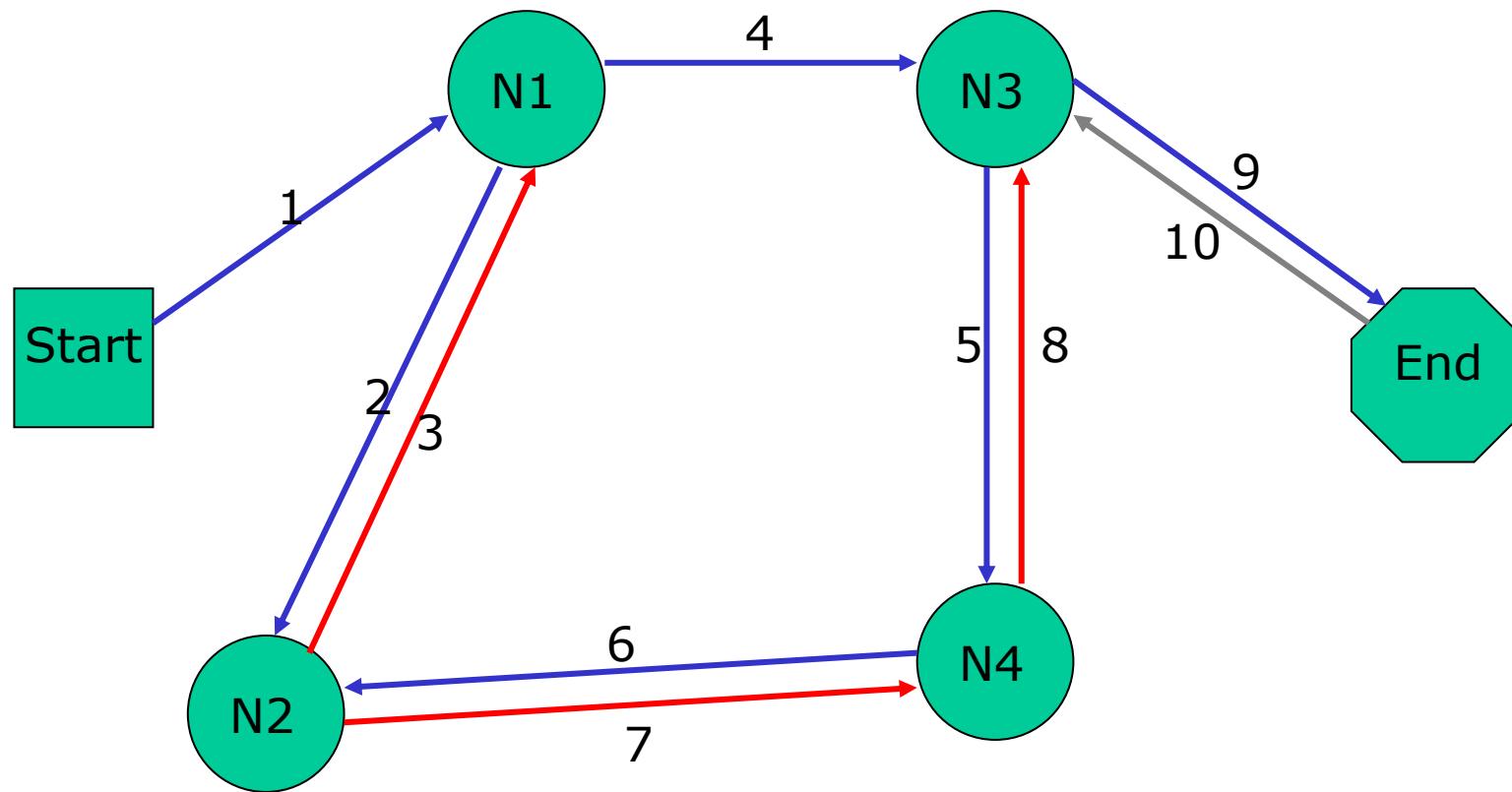
Accesso contenuti in FreeNet

- Il nodo N3 re-invia la query al vicino End dove trova il contenuto richiesto



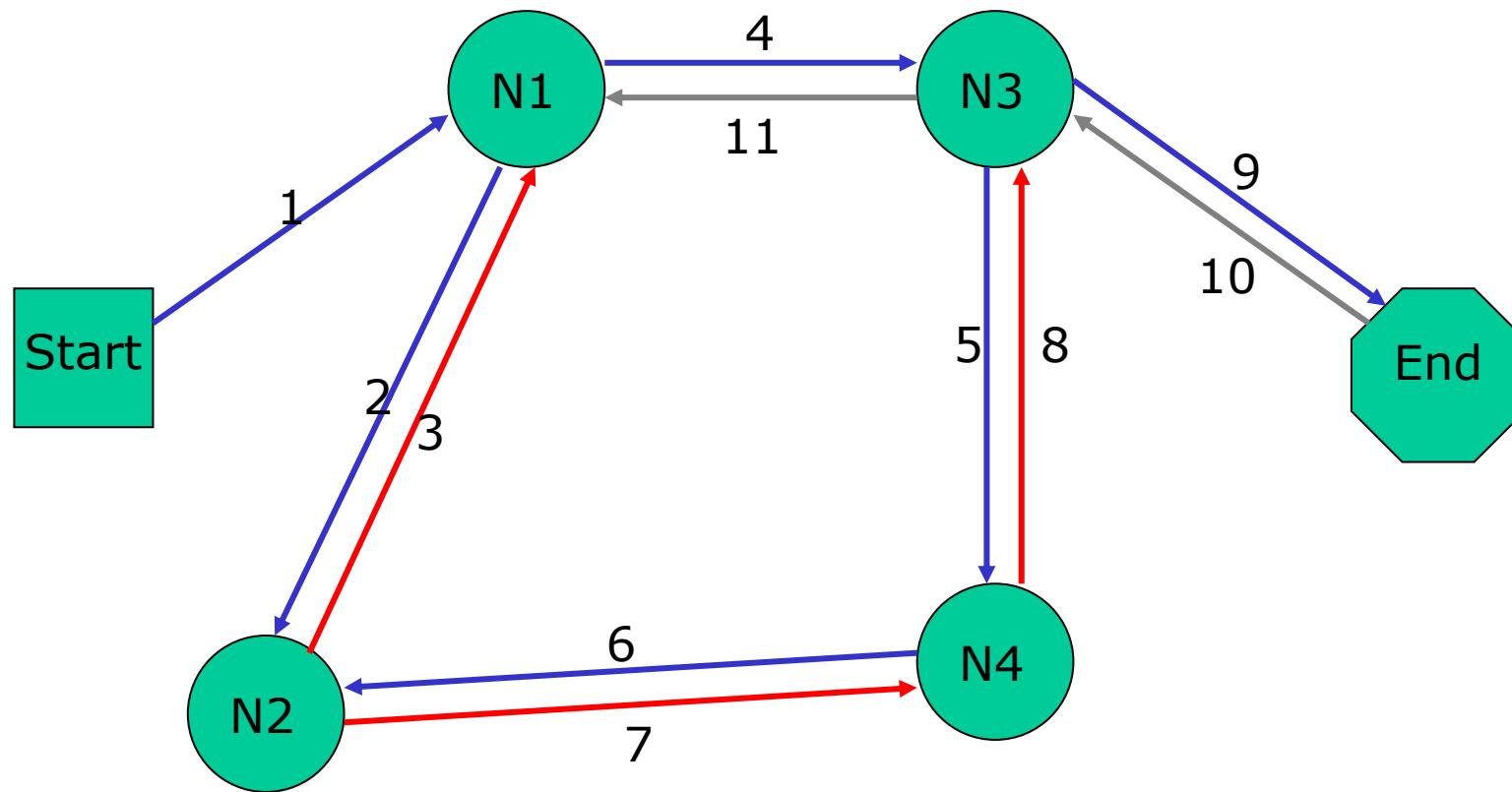
Accesso contenuti in FreeNet

- Il nodo End invia a ritroso la risposta verso N3



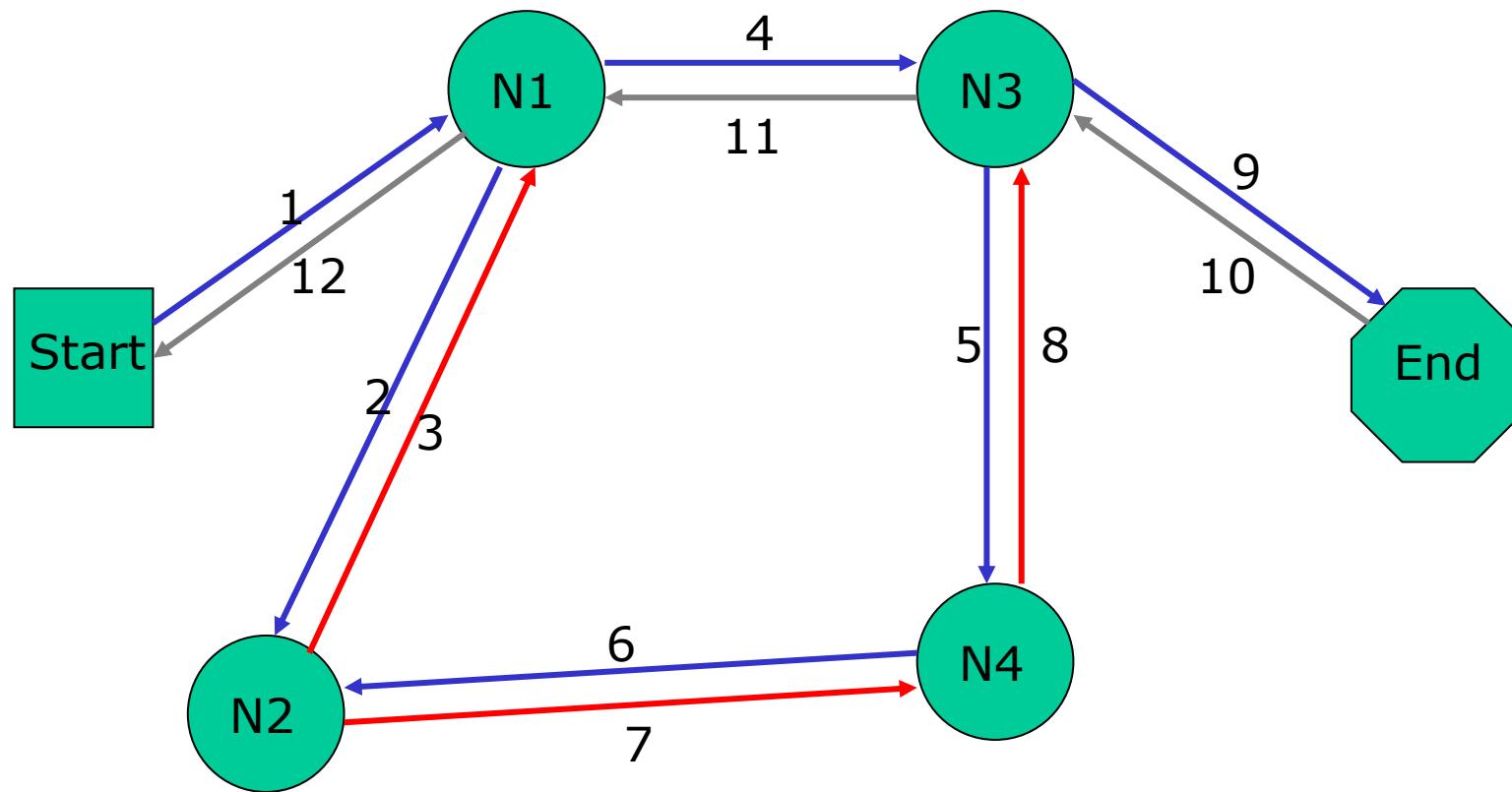
Accesso contenuti in FreeNet

- Il nodo N3 re inoltra la risposta direttamente verso N1 per portarla a destinazione



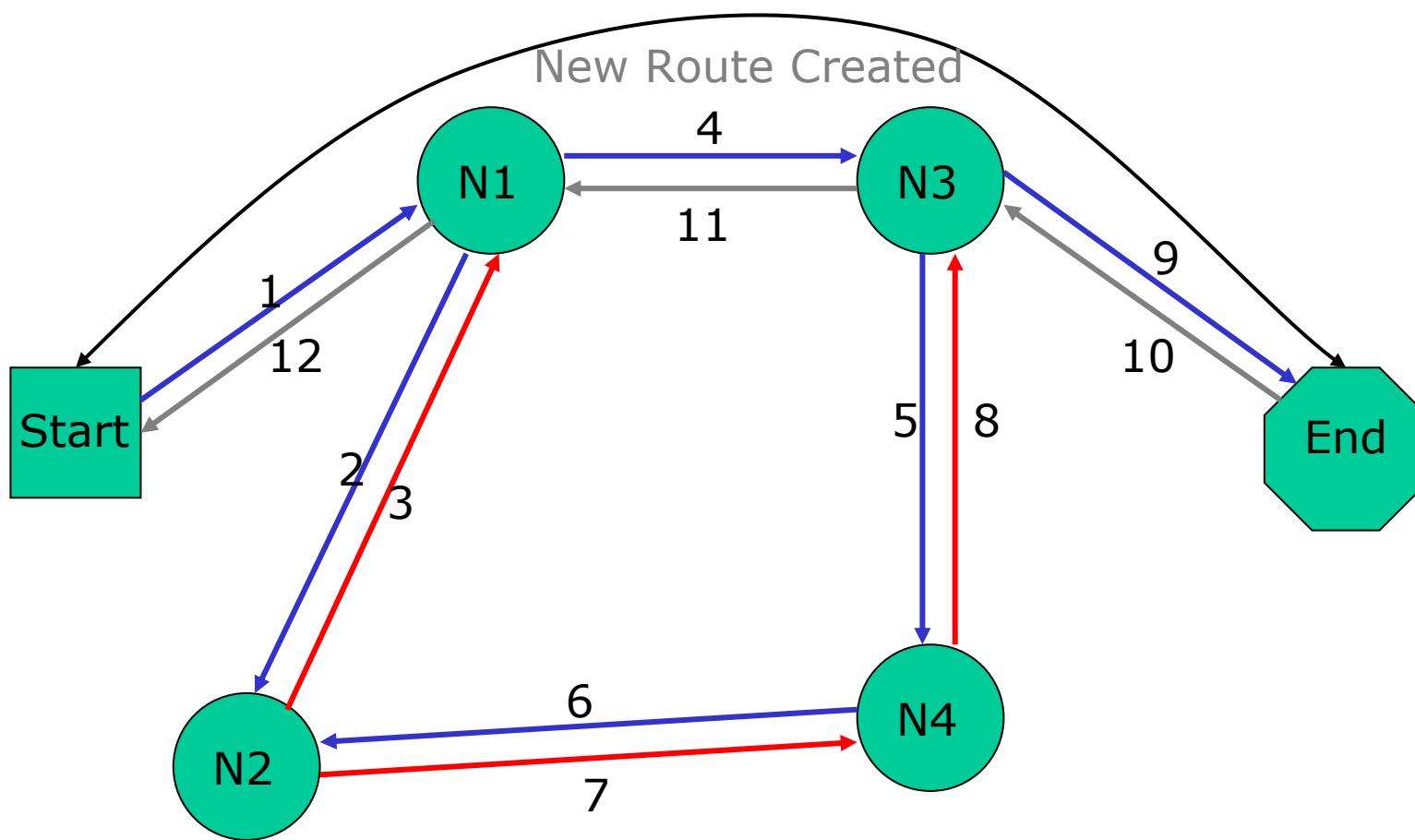
Accesso contenuti in FreeNet

- Il nodo N1 recapita la risposta all'origine della query Start



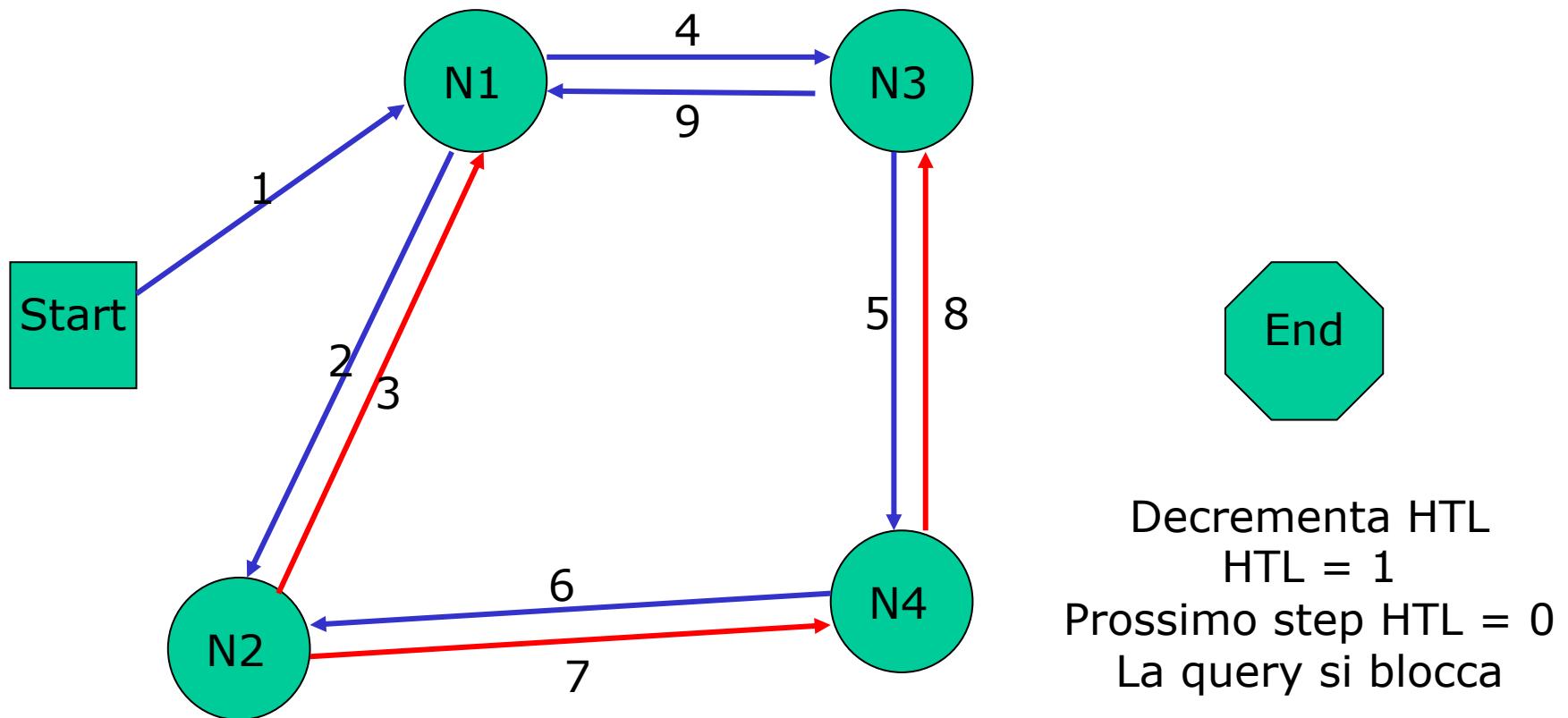
Accesso contenuti in FreeNet

- La risposta è tornata indietro lungo ogni nodo della catena che ha passato la richiesta, fino all'origine che ha generato la query creando un nuovo percorso



Caso Peggiose: Query Fallita

- Il nodo N3 re-invia la query al vicino N1 dove non trova il contenuto richiesto ... ma l'HTL arriva a 1
- Il nodo End ha il contenuto richiesto, ma la query da N3 non lo raggiunge in seguito alla decisione di routing sbagliata al nodo N1
- Al passo successivo query termina senza successo e va fatta ripartire ex novo passando attraverso una catena diversa



Accesso contenuti in FreeNet

- Un nodo che riceve da un confinante la richiesta di una chiave che ha precedentemente cercato e non trovato la rigetta immediatamente.
- Un nodo che deve inserire una chiave, prima la ricerca per evitare una collisione, e successivamente la inserisce
- La “profondità” della ricerca o dell’inserimento di una chiave è data dall’HTL (hops to live)
- Ogni nodo che deve passare una richiesta decrementa l’HTL di 1 arrivati a 0 la query si arresta
- Un “rumore di fondo probabilistico” viene inserito in tutte le decisioni di routing (variazione dell’HTL, etc.) per offuscare il traffico e evitare correlazioni

Caching contenuti in Freenet

- Ogni nodo memorizza le chiavi “alla rinfusa” in un database che viene denominato “datastore” gestito in logica LRU
- Quando arriva una nuova chiave/file che farebbe superare la dimensione designata del datastore, le chiavi meno recentemente usate vengono eliminate in ordine finché non ci sarà spazio
- Una chiave esiste in piu’ copie, dipendenti dalla profondita’ di inserimento della richiesta originale
- Ogni nodo che, dopo aver trasmesso una richiesta che ha avuto successo riceve la chiave da ripassare al nodo richiedente se ne fa una copia in locale
- I singoli datastore vengono gestiti con un watermark, sulla base della data e del numero degli accessi alle singole chiavi

Caching contenuti in Freenet

- Le chiavi “popolari” si moltiplicano e si spostano “vicino” a chi le richiede 
- Le chiavi “impopolari” scompaiono
- Si tratta di un comportamento “ecologico” che permette di realizzare un sistema in cui non esiste il comando “~~e~~lete”
- I singoli nodi si “specializzano” nel memorizzare alcune chiavi, basandosi su una “distanza lessicale” che viene calcolata utilizzando un hash del contenuto della  chiave, e specializzandosi in un segmento di essa
- Le decisioni di routing delle richieste vengono fatte in maniera intelligente, poiche’ i server pubblicizzano il segmento di spazio delle chiavi in cui sono “specializzati”

Memorizzazione nuovi contenuti

- L'utente genera un valore di chiave da associare al file e invia un messaggio al nodo a cui è connesso specificando la chiave proposta e un valore di HTL
- Il nodo verifica se la chiave è già in uso con una ricerca
- Se viene trovata la chiave associata a un altro file, il nodo restituisce lo stesso come risultato e l'utente seleziona una chiave diversa e ripete la procedura
- Se non viene trovata la chiave, il nodo esegue la ricerca della chiave più vicina e inoltra l'inserimento nel corrispondente nodo
- Se il limite di hop-to-live è stato raggiunto senza che si verifichi una collisione di chiave, un OK sarà propagato all'inseritore originale

Architettura FreeNet: pro e contro

- Vantaggi:
 - Convergenza veloce (successo nella ricerca) con traffico limitato nel caso medio
 - Si ferma solo quando il contenuto richiesto viene trovato
 - Scalabilità elevata
- Svantaggi:
 - Convergenza lenta nel caso peggiore

Hacking Freenet

- La rete Freenet è una **rete di sviluppo**, quindi si potrebbero trovare bug del protocollo
- E' facile individuare un nodo Freenet e causargli un DoS
- Analisi del traffico avanzata(issima)
- Il primo nodo a cui ci si collega conosce il mittente
- Freenet è un protocollo, quindi forse è più fruttuoso concentrarsi sui vari client specifici

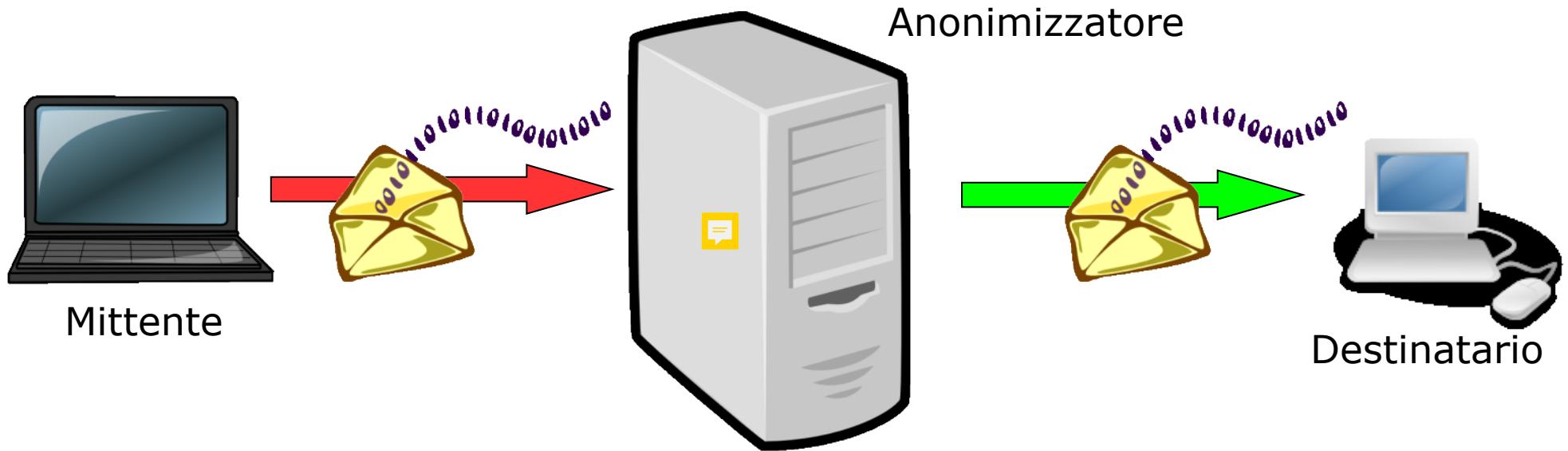
<http://freenetproject.org/index.php?page=faq#attack>

Email Anonime

- All'interno ci sono informazioni che ci **identificano** (headers)
- E' possibile capire **da dove** l'email è stata spedita e che percorso ha fatto
- Esistono tecnologie per rendere confidenziali le email
 - Crittografare il contenuto di una email
 - La firma elettronica garantisce che una email proviene effettivamente dal mittente legittimo
 - Esistono vari tipi di meccanismi per la firma digitale
 - Problema dell'associazione macchina-uomo: Web of trust o Certification Authority
- E per garantirne l'**anonimato**?...



Idea: remailer



Header della mail:

```
Received: from [192.168.0.100]
(host228-234-dynamic.14-87-r.retail.telecomitalia.it [87.14.234.228])
by giovanni.lonerunners.com (Postfix) with ESMTP id B771F4001
for <lol@lists.lonerunners.net>; Thu, 5 Jul 2007 12:45:53 +0200 (CEST)
Message-ID: <468CCBB9.7070501@lonerunners.net>
Date: Thu, 05 Jul 2007 12:45:13 +0200
From: Alessandro Tanasi <alessandro@lonerunners.net>
User-Agent: Thunderbird 1.5.0.12 (X11/20070604)
To: Lot of Fun Multimedia <lol@lists.lonerunners.net>
Subject: [Lol] Tartarughe wifi
```

Ciao, vista la tartaruga?

Header della mail:

```
Received: from [foobar]
[foobar [127.0.0.1]]
by giovanni.lonerunners.com (Postfix) with ESMTP id B771F4001
for <lol@lists.lonerunners.net>; Thu, 5 Jul 2007 12:45:53 +0200 (CEST)
Message-ID: <468CCBB9.7070501@lonerunners.net>
Date: Thu, 05 Jul 2007 12:45:13 +0200
From: Nessuno

To: Lot of Fun Multimedia <lol@lists.lonerunners.net>
Subject: [Lol] Tartarughe wifi
```

Ciao, vista la tartaruga?

Remailer

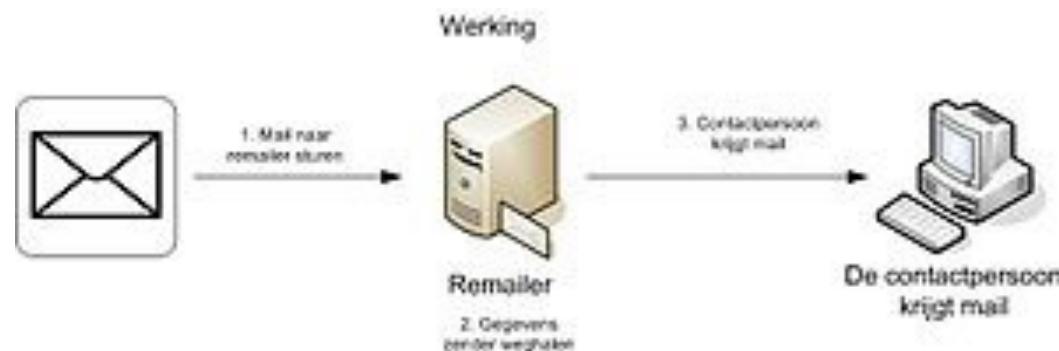
- Sistemi conosciuti che gestiscono l'anonimato della messaggistica email
- Tipologie:
 - Tipo 0: “Penet”
 - Tipo 1: “Cyberpunk”
 - Tipo 2: “Mixmaster”
 - Tipo 3: “Mixminion



Remailer di tipo 0



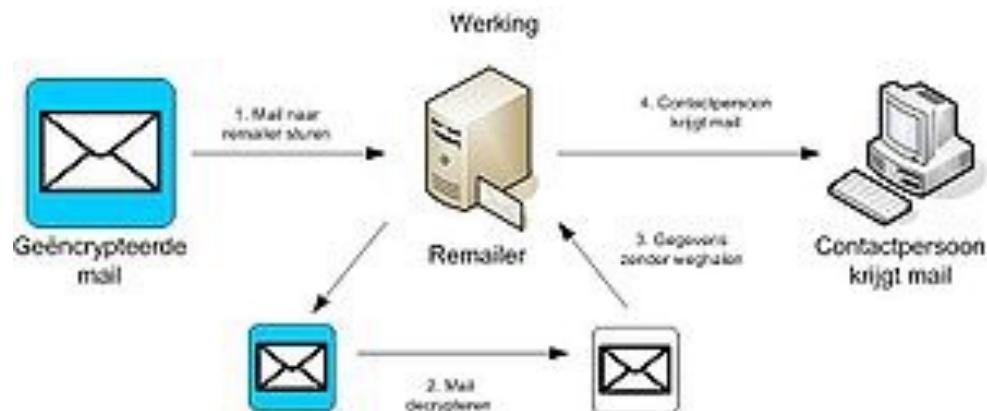
- Ricevono posta opportunamente formattata
- Inviano la posta che ricevono al destinatario togliendo le informazioni relative al mittente
- E' stata la prima implementazione
- Facilmente attaccabili:
 - violazione del server
 - tecniche di analisi temporale del traffico
 - azione legale che richiede i file di log all'amministratore



Remailer di tipo 1

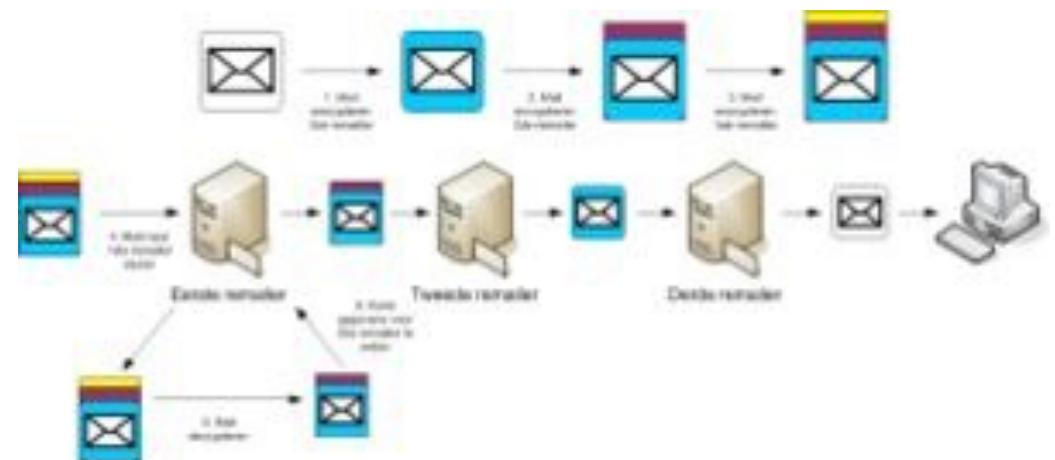


- Remailer usati in catene di almeno 3 elementi
- Utilizzo di metodi crittografici a chiave pubblica
- Crittografa il messaggio in successione per ogni remailer specificato
- La compromissione di $n-1$ elementi della catena non è sufficiente
- Comunque attaccabile con tecniche di analisi temporale del traffico



Remailer di tipo 2

- I messaggi vengono resi di dimensione uniforme: vengono spezzettati e frammenti sono di dimensione uguale
- I messaggi sono spediti con un ritardo casuale: sono mantenuti in una coda che viene svuotata casualmente
- Generazione di traffico casuale: vengono generati e scambiati falsi messaggi che l'ultimo remailer della catena poi scarterà
- Utilizzano ancora SMTP e una rete separata per la distribuzione delle chiavi crittografiche
- Suscettibili ad attacchi DoS



Remailer di tipo 3

- Utilizzano un proprio protocollo client-server
- Al posto dell' SMTP usa delle connessioni SSL tra server e per accettare i messaggi dagli utenti. Supporta anche la ricezione di risposte anonime usando dei reply-block a uso singolo, "Single Use Reply Blocks" o "SURBs". I messaggi inviati e quelli di risposta risultano indistinguibili.
- Sono flessibili e permettono l'interfacciamento con altre reti di tipo diverso (Mixmaster, MMS)
- Directory distribuita gestita automaticamente dai remailer per la gestione delle chiavi crittografiche
- Rete in fase di sviluppo ma comunque già utilizzabile

<http://www.mixminion.net/>

<http://en.wikipedia.org/wiki/Mixminion>

Vulnerabilità dei Remailer

- Gli attacchi per scoprire l'origine di un messaggio variano in base al tipo di remailer:
- Analisi statistica del traffico
- Analisi temporale del traffico in ingresso e uscita
- Violazione logica o fisica del server
- Generazione di Denial of Service
- ... violazione del client
- Non è detto che un utente usi reti di tipo 3 per spedire mail, le vecchie reti sono ancora in utilizzo!

Server di pseudonomi

- Problema: chi riceve un messaggio attraverso un remailer non può rispondere al mittente
- Pseudonimo: **indirizzo fittizio**, non riconducibile ad un indirizzo reale, tramite cui si possono far giungere messaggi al destinatario senza conoscerne l'identità
- Anonimato anche **all'indietro**, mittente e destinatario possono scambiarsi email in modo totalmente anonimo



http://en.wikipedia.org/wiki/Pseudonymous_remailer

Remailing Newnym

- Usando metodi di crittografia si permette ad un utente di creare un indirizzo fittizio (pseudonimo)
- I messaggi di quell'utente usciranno dal remailer con tale pseudonimo
- Si può caricare un reply block: sequenze di istruzioni crittografate su come far pervenire il messaggio all'indirizzo reale tramite una serie di remailer



```
=====
Config:
From: Smith
Reply-Block:
::
Anon-To: AAA@remailer.aaa.com
Latent-Time: +0:00
Encrypt-Key: password_a
::
Encrypted: PGP
-----BEGIN PGP MESSAGE-----
Version: 2.6.3i
```

```
/S3vZw+95ZuCZfqxKE0XrgZXzOEwfoyBcpVvf9Pb9D19TqEMTmmL/Jp11xcxmbJ2
vRoiG8ZhXs4r3E8liFsNtMMf6CUAsdv2ZoX1Hw==
=Bla3
-----END PGP MESSAGE-----
```

Pseudonym server

- Alice non sa come farsi rispondere da Bob senza rivelargli la sua reale identità.
- Infatti, quando Bob riceve il messaggio, lo vede arrivare da un remailer (in questo caso Freddy).
- Se Alice vuole ricevere una risposta, non può che mettere il suo indirizzo di posta dentro il corpo del messaggio; perdendo in questo modo l'anonimato.
- E' necessario ricorrere a uno **pseudonym server**

Pseudonym server

- . Uno pseudonym server consente all'utente di registrare un proprio alias (nym o pseudonimo), che viene associato ad una casella di posta elettronica.
- . L'aspetto innovativo sta nel fatto che né il nym né la casella postale sono direttamente riconducibili all'utente stesso. 
- . Ciò è possibile perché tutti i messaggi che transitano in entrata e in uscita da e per il nym server, passano prima attraverso una serie concatenata di anonymous remailer.

Pseudonym server

- Alice fa entrare in ballo un nuovo attore, Ziggy, che fa di mestiere lo **Pseudonym server**.
- Alice crea una catena di buste simile a quella precedente, ma che vada da Ziggy stesso, attraverso magari i soliti Danny, Eddy e Freddy, fino a lei.
- La invia poi a Ziggy, dicendogli di creare lo pseudonimo di AliBaba.
- Ha l' accortezza di fare questo invio utilizzando una catena di remailer, diversa dalla solita, supponiamo Harry, Indy e John

Pseudonym server

- Quando Ziggy riceve la busta, che contiene il contenuto precedentemente encapsulato preparato da Alice (definito come “**reply block**” e la richiesta di creare per lei uno pseudonimo, controlla sulla sua lista se questo pseudonimo esiste già, ed in caso negativo usa il reply block per farle avere l’ok alla creazione dello pseudonimo AliBaba.
- Il reply-block è paragonabile a un insieme concentrico di istruzioni, che possono essere lette solo una per volta e solo da uno specifico remailer alla volta perché crittate con la sua chiave pubblica.
 - Si può indicare il proprio reply-block una volta per tutte mentre si crea ilnym sulnym server, oppure si può cambiarlo di volta in volta per ogni messaggio:
 - la prima pratica è la più diffusa, ma bisognerà almeno controllare che i remailer attraverso cui si vuole far transitare il proprio messaggio siano sempre attivi.

Pseudonym server

- Si noti che Ziggy non ha la più pallida idea di chi Alice sia, perchè tutti i messaggi che riceve provengono da una catena di remailer, e quelli che manda usano il reply block

Percorso della richiesta di Alice:

**Alice -> Harry -> Indy -> John ->
Ziggy**

Percorso dell' OK di Ziggy:

**Ziggy -> Danny -> Eddie -> Freddy
-> Alice**

Pseudonym server

- A questo punto Alice possiede un indirizzo fittizio (uno pseudonimo, appunto) nel dominio di posta di Ziggy; se l' indirizzo di Ziggy fosse Config@Ziggy.org, lo pseudonimo di Alice sarebbe AliBaba@Ziggy.org.
- Per scrivere ad Alice, Bob mandera' un messaggio normale ad AliBaba@Ziggy.org.
- Ziggy ricevera' il messaggio, ed in base allo pseudonimo, selezionera' l' opportuno reply block da usare per inoltrare il messaggio attraverso la catena di remailer scelta da Alice.
- Né il server di pseudonimi né i singoli remailer usati saranno mai in grado di stabilire una connessione tra la casella reale e la casella *nym* (a meno che non si usi un solo remailer).

Pseudonym server

- Lo pseudonimo di Alice, AliBaba@Ziggy.org puo' essere usato anche per spedire messaggi.
- Se Alice vuole spedire un messaggio a Bob dal suo pseudonimo, usera' una catena di remailer qualsiasi per spedire a Ziggy il messaggio e l' indirizzo a cui spedirlo a nome del suo pseudonimo.
- A tutti gli effetti Bob può addirittura non sapere che l' indirizzo da cui riceve ed a cui spedisce posta sia diverso da uno "normale".

Percorso del messaggio da Alice a Bob:

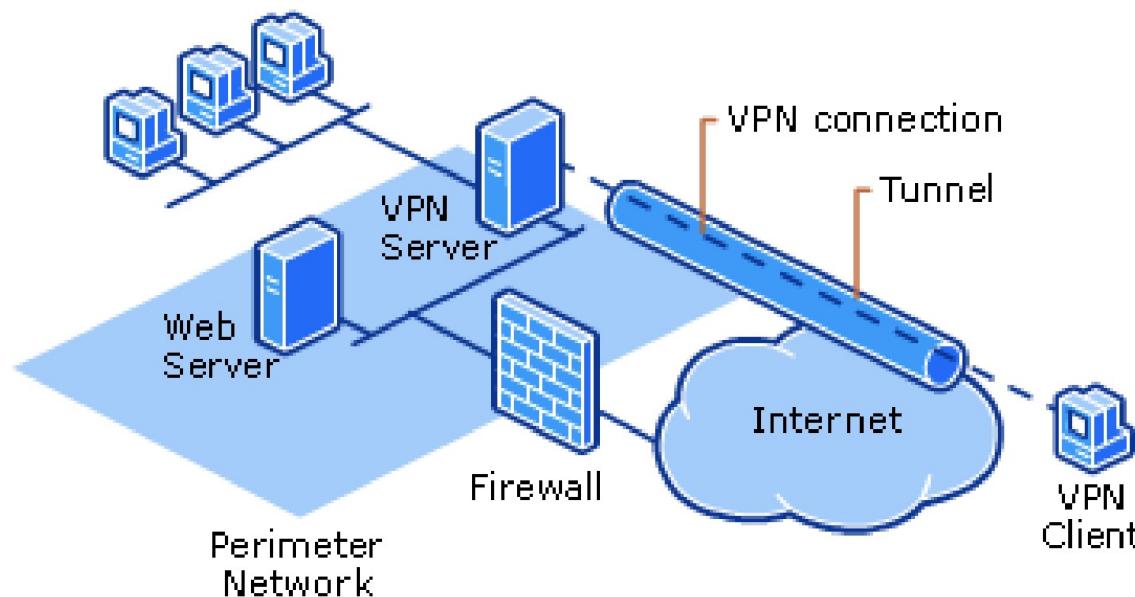
Alice -> Harry -> Indy -> John -> Ziggy -> Bob

Percorso della risposta di Bob ad Alice:

Bob -> Ziggy -> Danny -> Eddie -> Freddy -> Alice

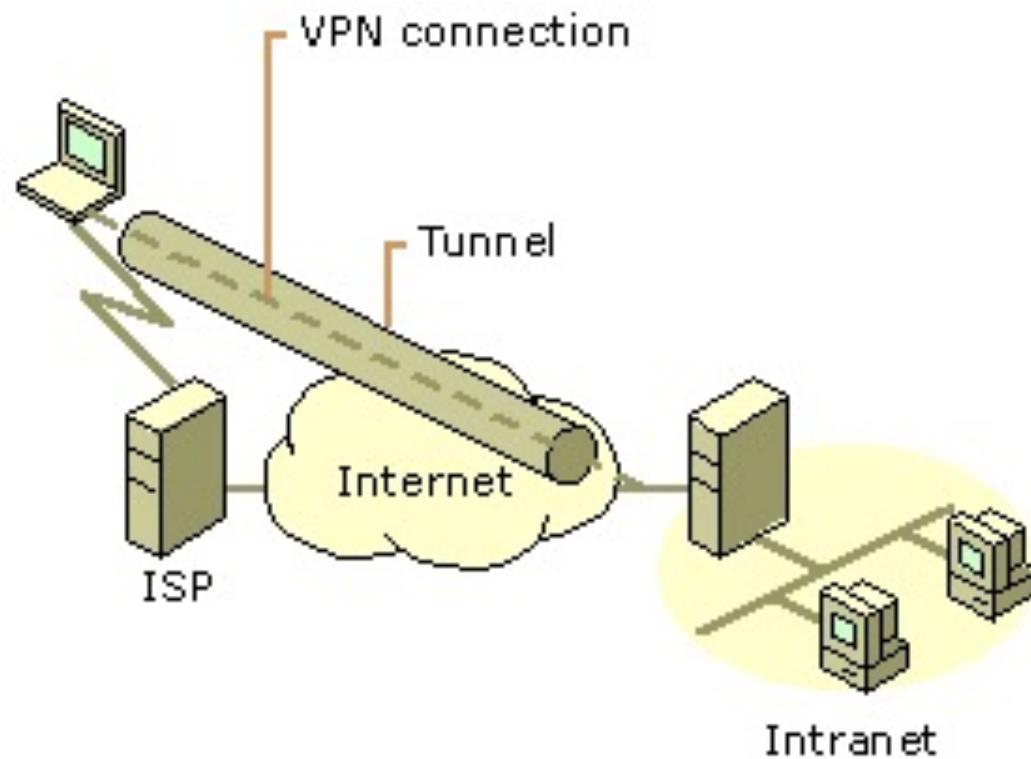
VPN: concetti generali

- Una VPN realizza un canale di comunicazione end-to-end su rete pubblica, (es. Internet) anziché usare una linea dedicata.
 - E' possibile connettere singoli utenti o intere reti
 - User-to-Site
 - Site-to-Site
 - Consente la connessione remota da una rete esterna, attraverso canali condivisi, a una LAN privata (**LAN Extension**) attraversandone il perimetro ed emulando un collegamento dedicato



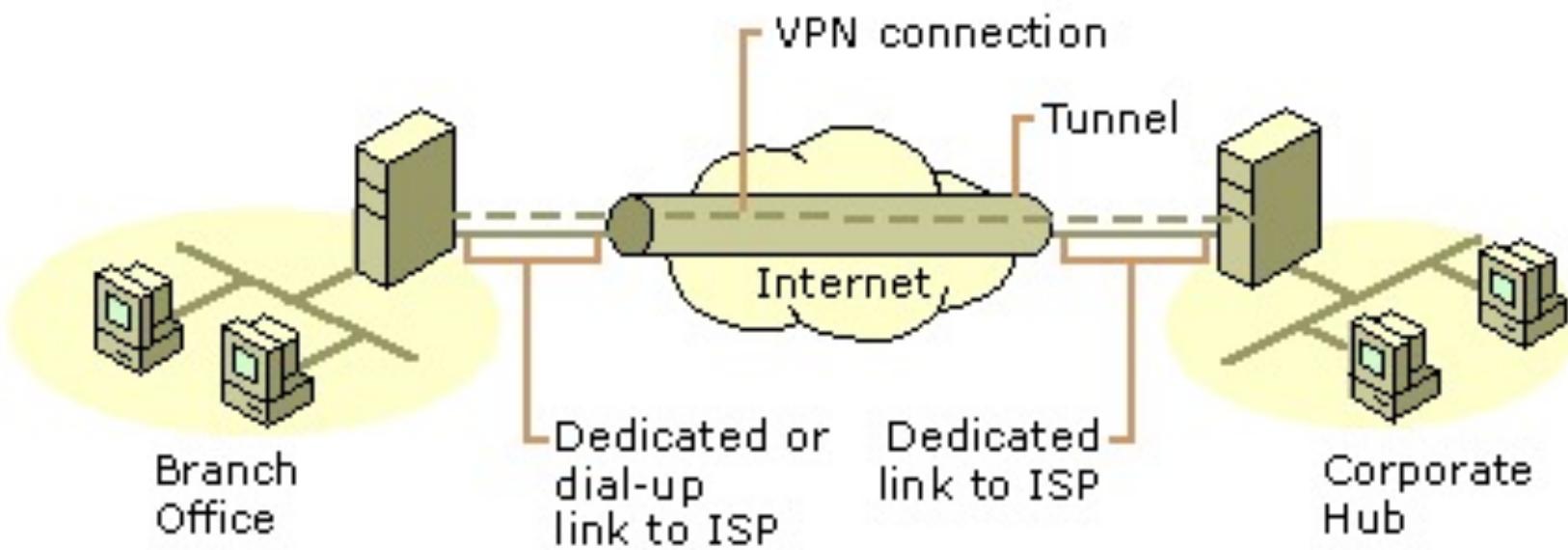
VPN: usi comuni (1/3)

1. Accesso remoto a una LAN private attraverso Internet



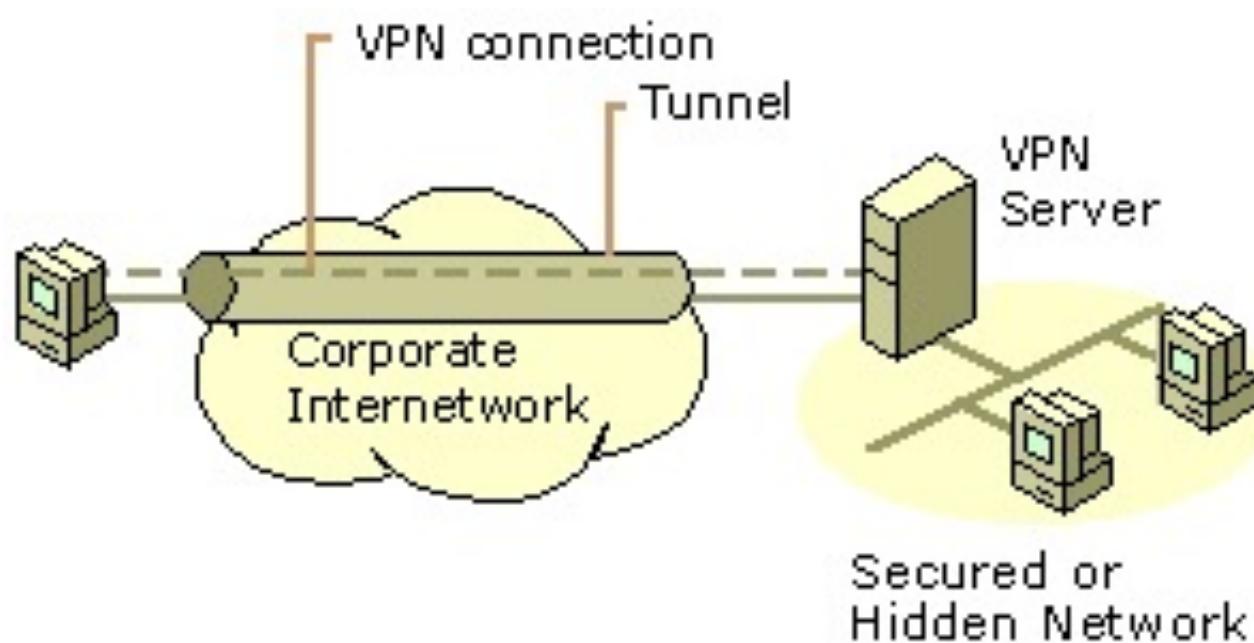
VPN: usi comuni (2/3)

2. Connessione fra reti private attraverso Internet (Site to Site VPN)



VPN: usi comuni (3/3)

3. Connessione fra Computers attraverso una Intranet (simile a 1.)

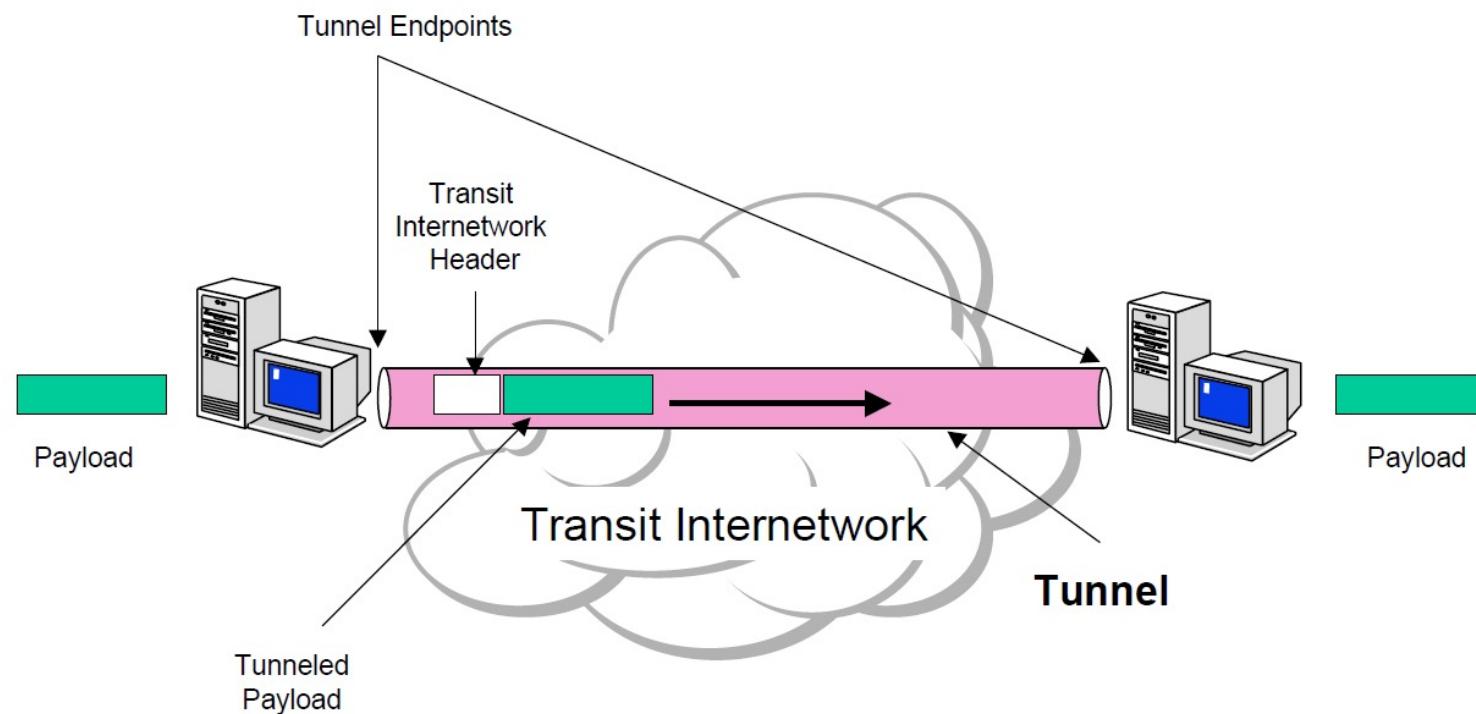


Perchè usare VPN?

- Economicità
 - Realizzare connettività remota tramite modem dial-up o tramite linee dedicate, è costoso.
- Scalabilità
 - Estendere una rete attraverso unai linea dedicata è complesso e non scala facilmente
 - Una VPN su rete pubblica è Facile da amministrare.
- Sicurezza
 - Una VPN garantisce confidenzialità e integrità end-to-end nonché mutua autenticazione fra gli estremi tramite tecniche crittografiche

Il concerto chiave: Tunneling

- La VPN è costituita da una serie di connessioni punto a punto trasportate su Internet.
- Per ottenere il tunneling, i pacchetti IP sono incapsulati come payload di altri pacchetti.



Requisiti di base

- Autenticazione e controllo accessi
 - Verifica l'identità dell'endpoint consentendo la connessione e l'accesso alla rete solo a entità autorizzate
 - Garantisce funzioni di audit e accounting per dimostrare in maniera non ripudiabile chi ha acceduto, quali informazioni ha scaricato etc..
 - Usa certificateiX.509, pre-shared key, etc.
- Riservatezza
 - E' garantita l'assoluta confidenzialità dei dati che transitano sulla rete pubblica
- Integrità dei dati
 - Nessuno dall'esterno della VPN può alterare nulla senza essere rilevato
- Key Management
 - Genera e aggiorna le chiavi di crittografia per il client e il server.
 - Gestione chiavi per IP effettuata via : ISAKMP/Oakley, etc.

Implementazioni di riferimento

- VPN basate su PPP
 - Point-to-Point Tunneling Protocol (PPTP) (PPP + encryption + GRE)
 - Layer Two Tunneling Protocol (L2TP) (PPTP + L2F)
- VPN basate su TCP/IP
 - L2TP/IPsec
 - IPsec Tunnel mode [[RFC 4301](#)]
 - BGP/MPLS IP VPN [[RFC 4364](#)]
- VPN basate su SSL/TLS
 - Secure Socket Tunneling Protocol ([SSTP](#)) (PPTP + SSL)
 - SSL VPN
 - OpenVPN

PPP - Point-to-Point Protocol

- PPP [[RFC 1661](#)] garantisce una modalità standard per il trasporto multi-protocollo dei pacchetti su collegamenti punto-punto (link diretti end-to-end).
 - Data link layer (layer 2) protocol
- Tre componenti fondamentali
 - Incapsulamento: a scopo di trasporto
 - Link Control Protocol: per garantire la connettività data-link
 - Network Control Protocols (NCP): Supporto e gestione L3
- Opzioni extra
 - Autenticazione: [PAP](#), [CHAP](#), [EAP](#), [MS-CHAP](#), [MS-CHAPv2](#), etc.
 - Qualità del link e rilevamento errori
 - Compressione
 - Cifratura: [MPPC](#) + [MPPE](#), etc.
 - Multilink ([MP](#), The PPP Multilink Protocol)

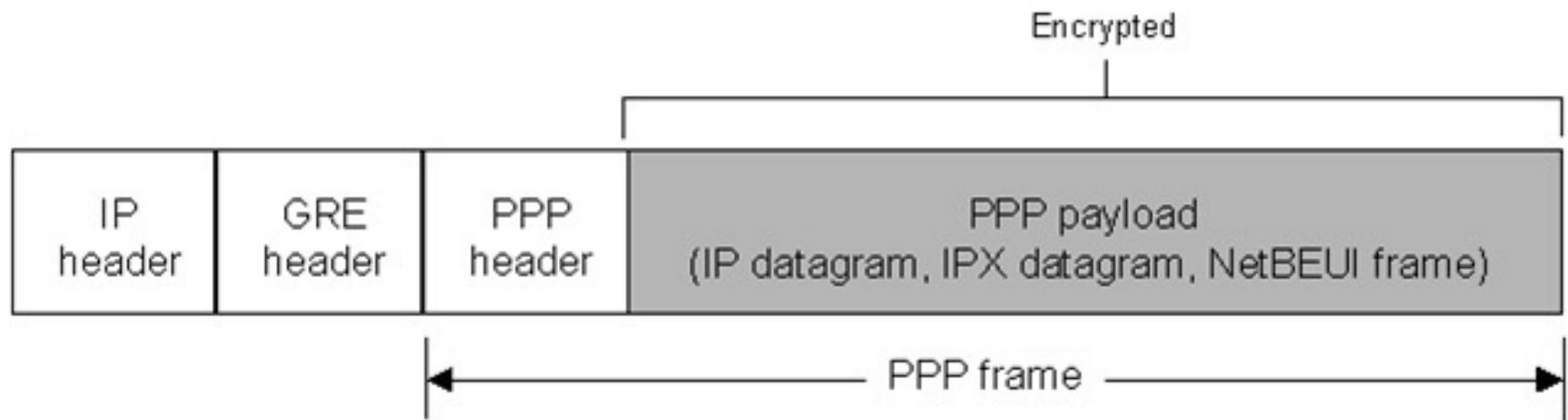
Protocolli di Tunneling

Consentono a un utente della rete di accedere o fornire un servizio di rete che la rete sottostante non supporta o non fornisce direttamente (es. IPV6 su IPv4)

- GRE (Generic Routing Encapsulation): Stabilisce una connessione virtuale point-to-point fra due reti.
 - IP è usato come protocollo per il delivery
 - Virtual Tunnel: (Tunnel) IP header + GRE header
 - Solo incapsulamento, *nessuna cifratura*
- PPTP / L2TP
- IPsec
- OpenVPN (with SSL/TLS)
- etc.

PPTP - Point-to-Point Tunneling Protocol

- PPTP [RFC 2637] usa un meccanismo avanzato GRE-based per garantire un servizio di trasporto incapsulato con controllo di flusso e congestione per pacchetti PPP.
- PPTP crea un tunnel GRE attraverso cui i pacchetti PPTP vengono inviati.

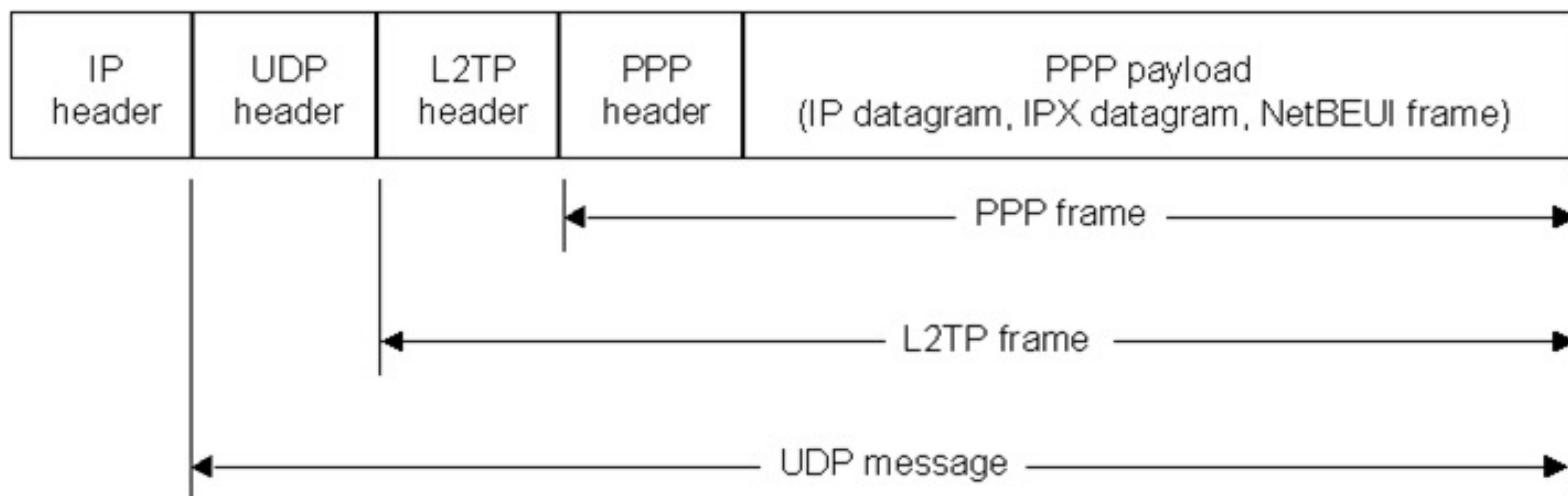


SicurezzaPPTP

- PPTP presenta note vulnerabilità in termini di sicurezza
 - MS-CHAP è fondamentalmente insicuro.
 - MS-CHAPv2 è vulnerabile ad attacchi forza-bruta basati su dizionari su pacchetti di challenge response catturati.
- Il payload PPP può essere cifrato usando Microsoft Point to Point Encryption (MPPE) in presenza di MS-CHAPv1/v2
- EAP-TLS (Extensible Authentication Protocol – TLS) è la scelta migliore di autenticazione per PPTP.

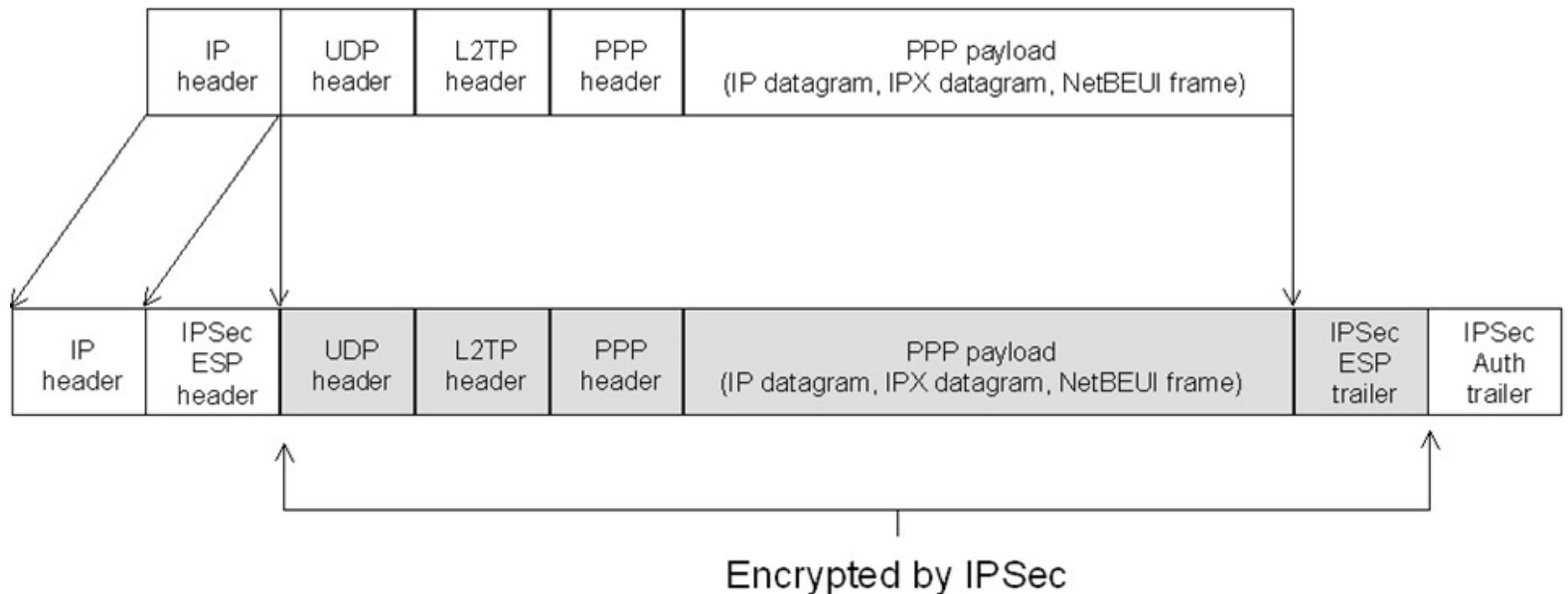
L2TP - Layer Two Tunneling Protocol

- L2TP [[RFC 2661](#)]: PPTP + L2F (Layer Two Forwarding)
 - Un protocollo di alto livello (e.g., PPP) stabilisce una sessione L2TP (“call”) all’interno del tunnel L2TP, isolando il traffico per ogni sessione.
 - Un tunnel può contenere simultaneamente multiple connessioni.
 - L2TP su reti IP si basa su UDP e una serie di messaggi L2TP per la gestione del tunneling
 - [L2TPv3](#) offre funzionalità avanzate di sicurezza, miglioramenti nell’incapsulamento, e possibilità di trasportare protocolli datalink diversi da PPP su IP



L2TP/IPsec

- L2TP non garantisce confidenzialità o autenticazione forte.
- E' possibile usare IPsec ESP (Encapsulating Security Payload) per cifrare pacchetti L2TP.
 - La cifratura si avvia prima di stabilire la connessione PPP negoziando un'associazione sicura IPsec
 - Richiede autenticazione end-to-end basata su certificati digitali

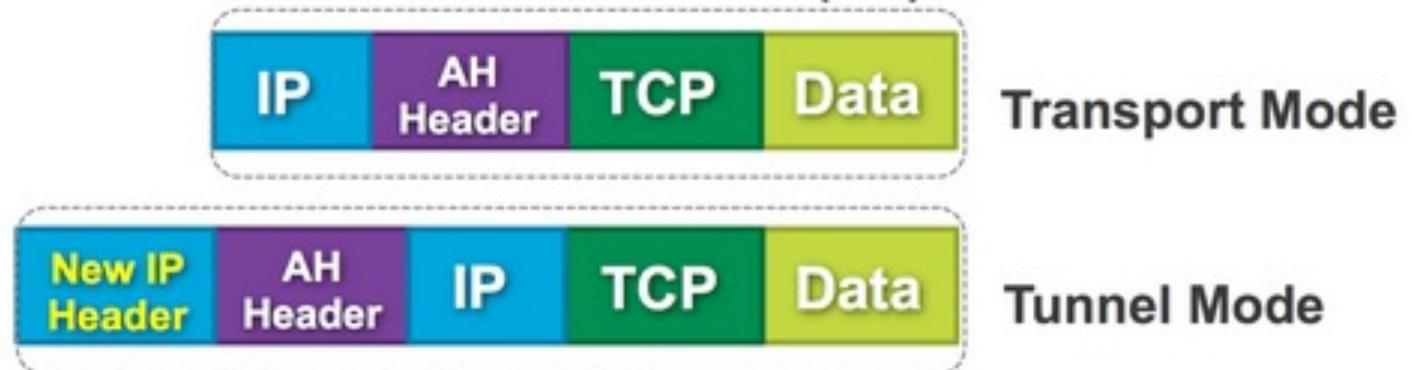


IPsec

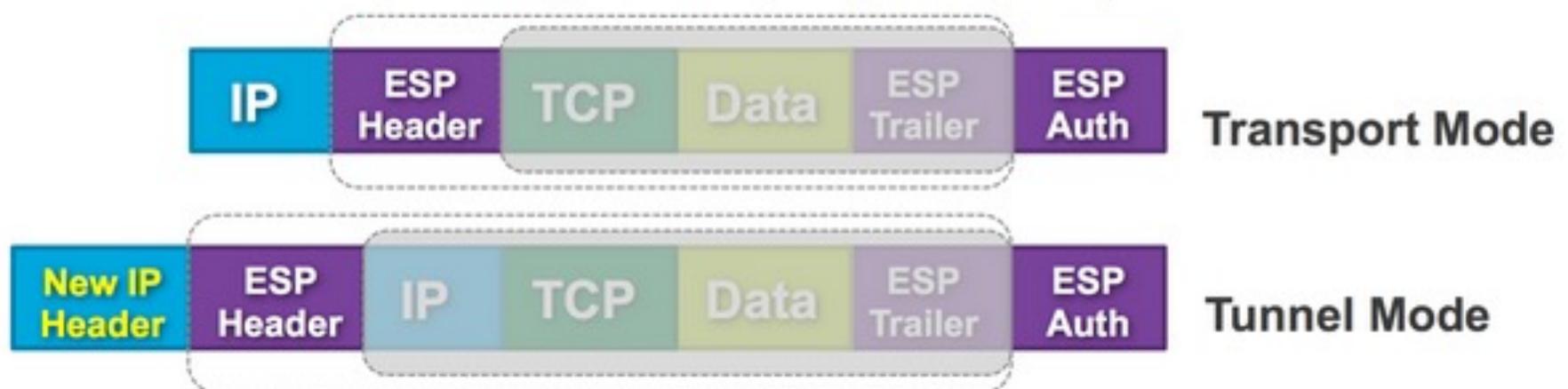
- IPsec [[RFC 4301](#)] è una suite di protocolli per la sicurezza end-to-end che garantisce autenticazione e cifratura su reti IPv4
- Due modalità in IPsec
 - **Transport mode:** Inserisce un header IPsec (AH/ESP) fra quelli IP e TCP, successivamente modifica l'header IP originale.
 - **Tunnel mode:** Incapsula i pacchetti all'origine, inserendo in testa nuovi header IP e IPsec.
- Due funzioni garantiscono la funzionalità:
 - Authentication Header (AH)
 - Garantisce autenticazione dell'origine e integrità senza cifratura.
 - Encapsulating Security Payload (ESP)
 - Garantisce autenticazione, integrità e cifratura.
- Le Security Associations (SA) forniscono i parametri necessari per le operazioni relative a AH e ESP.
 - IKE (Internet Key Exchange): garantisce autenticazione e scambio di chiavi. e.g., ISAKMP, OAKLEY

Ipsec: Modalità

Authentication Header (AH)

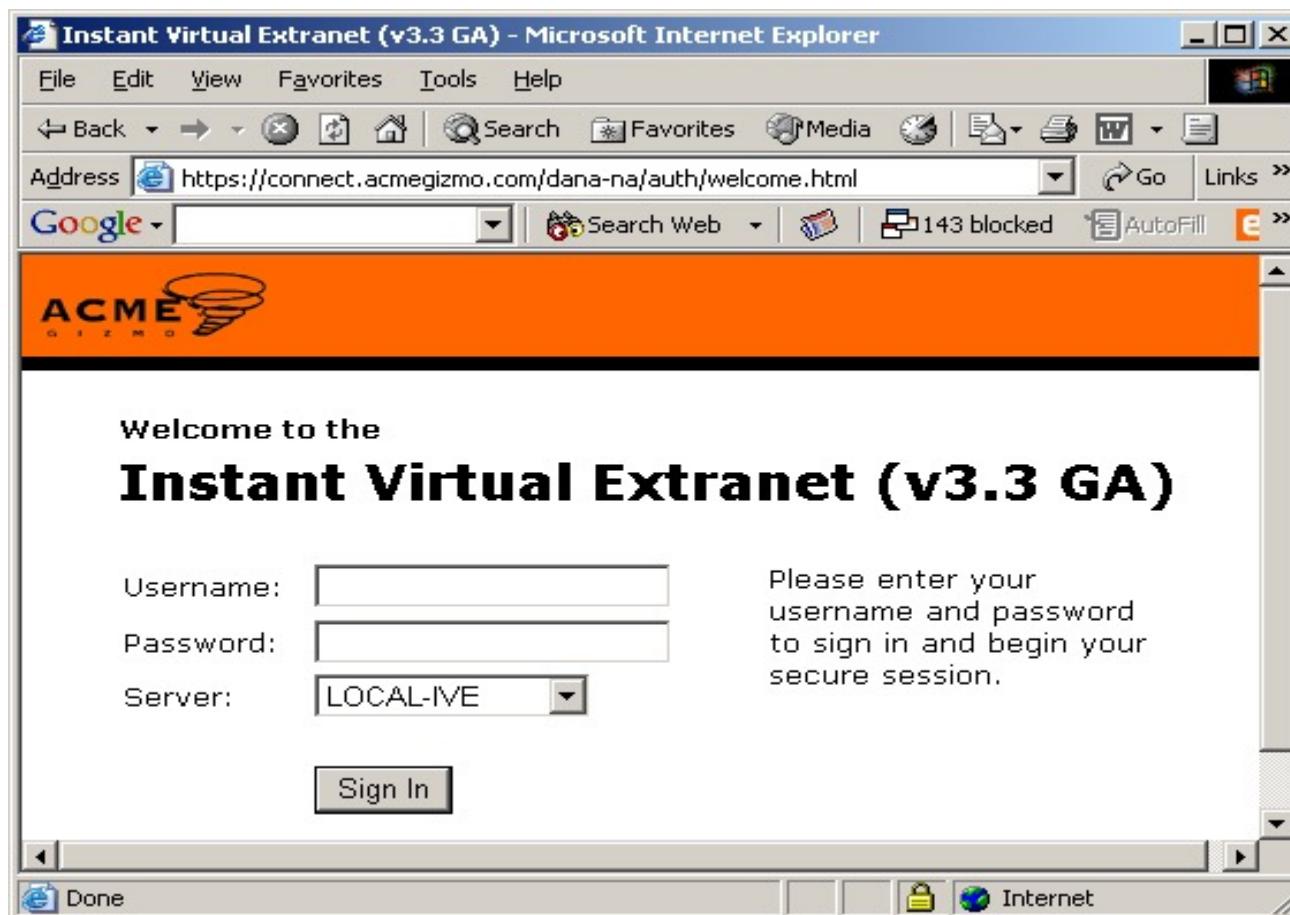


Encapsulating Security Payload (ESP)



SSL VPN

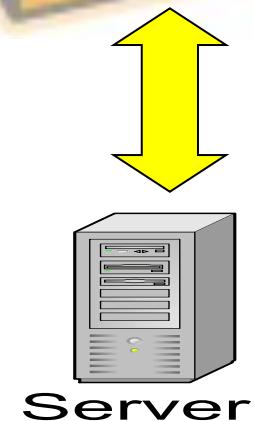
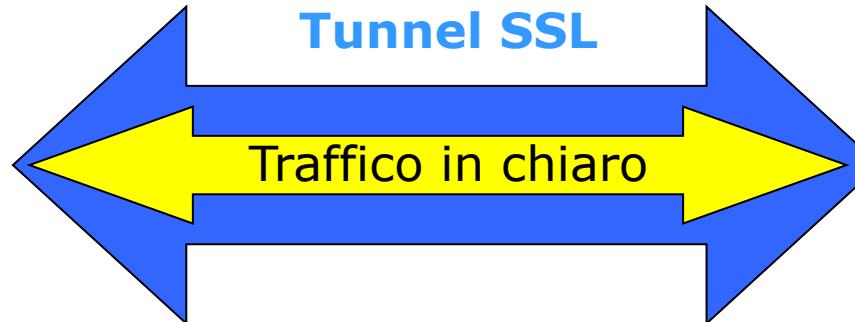
- Un tipo di VPN che può essere utilizzata attraverso un Web browser standard.
- Il traffico è cifrato tramite i protocolli SSL o TLS



Connessione VPN SSL



Remote
User



- Implementazione SSL dei browser
- Download di un applet java
- Incapsulamento del traffico

- L'idea alla base delle connessioni VPN su SSL è di sfruttare l'implementazione SSL/TLS dei più diffusi browser e incapsulare il traffico all'interno di una connessione SSL.
- Per fare questo si scarica un applet java o un controllo Active X in grado di incapsulare tutto il traffico che normalmente viaggerebbe in chiaro in una connessione SSL
- Il traffico una volta arrivato al dispositivo VPN SSL viene decapsulato e reinstradato verso server/applicazione destinataria.

Tipologie d'accesso

WEB translation/proxy

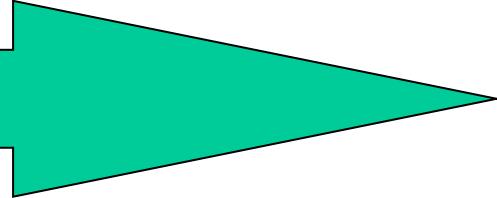
1

- Applicaizoni e portali WEB
- Supporto nativo di applicazioni: RDP, Telnet, File sharing
- CLIENTLESS

3

Network Connector

- Visibilità a livello IP
- Accesso completo alla rete
- Assegnazione IP interno
- Client scaricato (Diritti amministrativi)



SSL application wrapper

2

- Applicazioni client/server TCP
- Active X – JAVA applet
- Distinzione Windows – Unixlike
- Local listener – Wrapper syscall
- Client/Applet scaricata
- Al primo accesso si installa ActiveX

Conclusioni

- **Esistono sistemi anonimi** di comunicazione e per la difesa della privacy dei dati e sono pienamente utilizzabili
- Manca una loro diffusione e conoscenza
- Nel caso serva :) probabilmente alcuni sistemi possono essere manomessi o aggirati
- La **sensibilizzazione** sul problema privacy è fondamentale
 - ... *per far in modo che Internet rimanga libera per le generazioni future*