Corso di Sicurezza su reti II

Prof. Alfredo De Santis

The Onion Router

Falso Michele matr. 0521/436

Abstract

- Introduzione
  - Privacy e anonimato
  - Classificazione dei sistemi di comunicazione anonima
  - Cui prodest?
  - Un po’ di storia
- Tor: una panoramica
- Tor: una rete anonima distribuita
- Tor: La tecnologia
- I Servizi
  - Connessione anonima in uscita
  - Hidden Services
- Attacchi e Difese
- Tor in 4 semplici passi
- Etiquette e abuso
- Conclusioni

Introduzione: Privacy ed Anonimato

- Anonimato: *“Lo stato di non essere identificabile in un insieme”*
- Navigare su Internet
  - Indirizzo IP
  - Analisi del traffico:
    - l’analisi delle intestazioni dei *pacchetti di dati*
    - Tecniche statistiche
  - Archiviazione dati in file di log su Provider
- The Onion routing

Comunicazione anonima

- **In una comunicazione si definisce :**
  - **Accesso *anonimo in avanti***:

    l'accesso ad un servizio senza che nessuno possa risalire all'indirizzo IP da cui è stata generata la richiesta.
  - **Accesso *anonimo all'indietro*:**

    l'accesso ad un servizio senza che nessuno possa risalire all'indirizzo IP del server che ospita il servizio.
  - **Accesso *riservato*:**

    Quando nessuno, tranne mittente e ricevente, può conoscere il contenuto dei dati scambiati.

Tecnologie per la comunicazione anonima

- Sistemi anonimi ad alta latenza
  - Anonymous remailer
- Sistemi anonimi a bassa latenza
  - The Onion Routing
  - Anonymizer
  - Java Anon Proxy
  - FreeNet

Sistemi anonimi ad alta latenza

- **Anonymous remailer**

  Sono server che ricevono messaggi di posta elettronica e li rinviano seguendo apposite istruzioni incluse nei messaggi stessi, senza rivelare la loro provenienza originaria.

  - **Tipologie**
    - **Remailer pseudoanonimi**
    - **Cypherpunk**
    - **Mixmaster**

Sistemi anonimi ad alta latenza (2)

- **Remailer Pseudoanonimi – Tipo 0**
  - Hanno ormai un interesse più che altro storico.
  - Fornivano all'utente un account al quale veniva assegnato casualmente uno pseudonimo di identificazione.
  - Mantenevano database che memorizzavano le coppie *"pseudonimo-indirizzo di posta"*.
  - Nella posta in uscita gli *header* di identificazione del messaggio venivano sostituiti con lo pseudonimo.

Sistemi anonimi ad alta latenza (3)

- **Cypherpunk – Tipo 1**
  - Non esiste un database degli utenti e non vengono conservati log dell'utente.
  - Ogni anonymous remailer Cypherpunk conserva la propria chiave pubblica PGP.
  - Tali remailer sono in grado di accettare posta cifrata mediante la chiave pubblica PGP, decifrarla e inviarla all'indirizzo richiesto.
  - Il sistema offre la possibilità di:
    - Concatenare remailer in successione.
    - Riordinare in modo casuale i messaggi.
    - Introdurre del ritardo nella trasmissione dei messaggi.

Sistemi anonimi ad alta latenza (4)

- **Mixmaster – Tipo 2**
  - Rappresentano il tipo più recente di remailer operativo su Internet e costituisce lo stato dell'arte nell'ambito degli anonymous remailer.
  - Predispongono il concatenamento e la cifratura dei messaggi in modo automatico al momento della preparazione del messaggio mediante un apposito client.
  - Usano il pacchetto **RSAREF** e alcuni formati proprietari per cifrare i messaggi.
  - Scompongono i messaggi, li sottopongono a cifratura multipla e li incapsulano in uno o più pacchetti di dati di uguale dimensione.

Sistemi anonimi a bassa latenza

- **The Onion Routing**

  Tali sistemi mirano:

  - a rendere non tracciabile la fruizione di servizi di rete generici (WWW, IM, ecc.) in modo trasparente all'utente.
  - a garantire latenze tollerabili e predicibili.

- **Questo tipo di tecnologia anonima è resistente alle comuni tecniche di analisi di traffico ed effettua:**

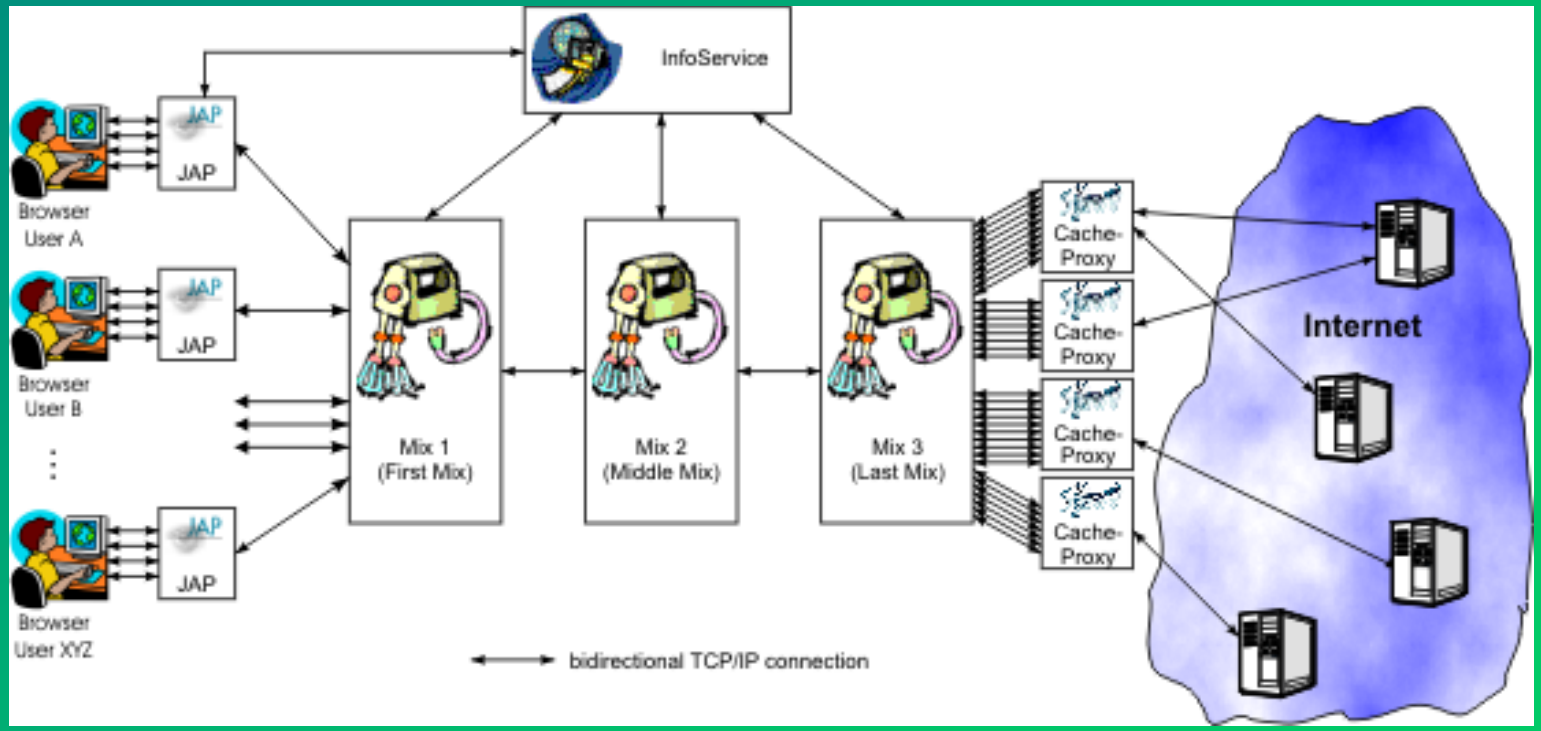
  - Tunneling di stream TCP attraverso circuiti virtuali.
  - l'incapsulamento dei pacchetti TCP in strutture dati *(onion)* ripetutamente cifrate.
  - l'instradamento delle onion attraverso nodi successivi.

Sistemi anonimi a bassa latenza (2)

- **Anonymizer**
  - Sistema a single-hop proxy: un proxy elimina le informazioni sul mittente della richiesta prima di inviarle sulla rete.
- **Java Anon Proxy**
  - Permette agli utenti di spedire i propri dati sulla rete Internet, utilizzando percorsi alternativi che attraversano una serie di intermediari detti mix.
  - Il sistema fa uso di :
    - **JAP Client:** software da installare sulla macchina dell'utente.
    - **Info service:** fornisce informazioni sulle mix-cascade disponibili, chiavi pubbliche di ciascun mix etc.
    - **Cache proxy:** ricevono dati dai mix e li inoltrano sulla rete Internet.
    - **Cifratura ibrida** (RSA/AES-128) per lo scambio di messaggi tra i JAP client e le mix-cascade.

Sistemi anonimi a bassa latenza (3)

La figura mostra una possibile mix-cascade scelta da un utente mediante il programma client e condivisa da altri utenti.



Sistemi anonimi a bassa latenza (4)

- **Freenet**
  - Freenet è una rete adattativa peer-to-peer di nodi paritetici che si interrogano reciprocamente per immagazzinare e recuperare file di dati cifrati e identificati da nomi (chiavi) indipendenti dalla locazione.
  - I nodi includono un proxy che permette di accedere al server Freenet con un form, utilizzando il protocollo HTTP.

Sistemi anonimi a bassa latenza (5)

- **Le chiavi (GUID-Key) che identificano un file sono di tre tipi:**

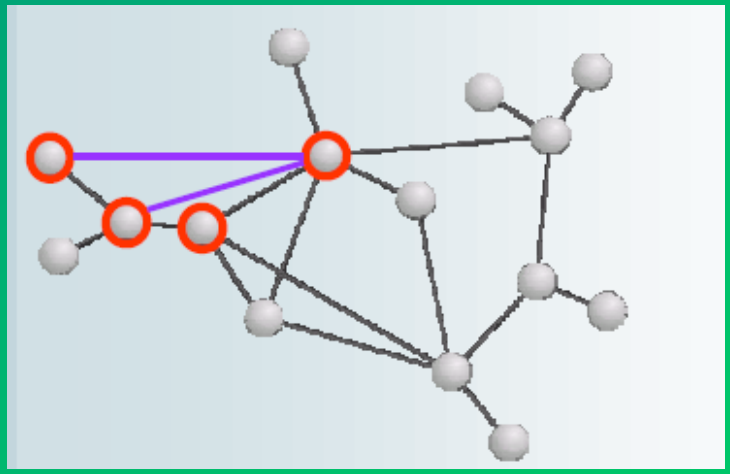
- **Keyword-Signed Key (KSK):** è la più semplice ed è creata sulla base di una descrizione del file data dall'autore. Per permettere il recupero del file basta conoscere la sua descrizione, semplice da ricordare e da comunicare.

- **Content-Hash Key (CHK):** è la chiave che viene usata per la memorizzazione dei dati a basso livello e viene generata calcolando un codice hash in base ai contenuti del file.

- **Signed-Subspace Key (SSK**): Un utente crea un proprio namespace personale generando in modo casuale una coppia di chiavi pubblica/privata che serviranno da identificativo e una breve descrizione del file. La chiave pubblica e la stringa descrittiva verranno utilizzate per generare la chiave SSK, mentre la chiave privata firmerà il documento.

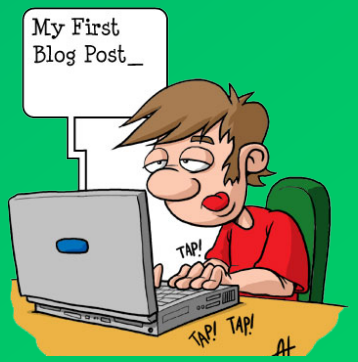
Sistemi anonimi a bassa latenza (6)

- Ogni nodo comunica con gli altri sulla base di una conoscenza locale dinamica dei nodi limitrofi.
- In FreeNet i messaggi, anziché viaggiare direttamente dal mittente al destinatario, attraversano una sequenza di nodi.
- Quando si richiede un file, i nodi per nascondere il reale possessore del file, alterano i messaggi di risposta fingendo di esserne i reali possessori.



Cui prodest ?

- **Persone normali**
  - Le persone normali proteggono la loro privacy da marketing senza scrupoli e furto di identità
- **I militari**
  - Agenti in missione
  - hidden services
  - Raccolta di intelligence
- **I giornalisti**
  - Reporter senza frontiere
  - Cittadini giornalisti in Cina
- **Polizia**
  - Sorveglianza on-line
  - Operazioni sotto copertura
  - Linee di denuncia davvero anonime
- **Dirigenti di impresa**
- **Blogger**



Tor: un po’ di storia

- TOR -”The second generation Onion Router”- è l'erede "OpenSource" di un progetto militare dal nome “Onion Routing”, avente come fine la creazione di una rete di proxy a bassa latenza in grado di garantire un certo livello di anonimato ai suoi utenti.

- Il progetto Onion Routing viene per la prima volta presentato nel Dicembre 1996 alla dodicesima Conferenza Annuale delle Applicazioni della Sicurezza Informatica di San Diego - CA, dal “Centro di Sistemi Informatici ad alta Sicurezza” (CHACS) del Laboratorio di Ricerca Navale (NRL) della Marina Militare Americana.

- Nel 1998 la tecnologia "Onion Routing" viene Brevettata da "The United States of America as represented by the Secretary of the Navy“.

Tor: un po’ di storia (2)

- Nel 2001 Syverson tiene un talk al decimo simposio sulla sicurezza chiamato "USENIX", presentando la tecnologia brevettata.

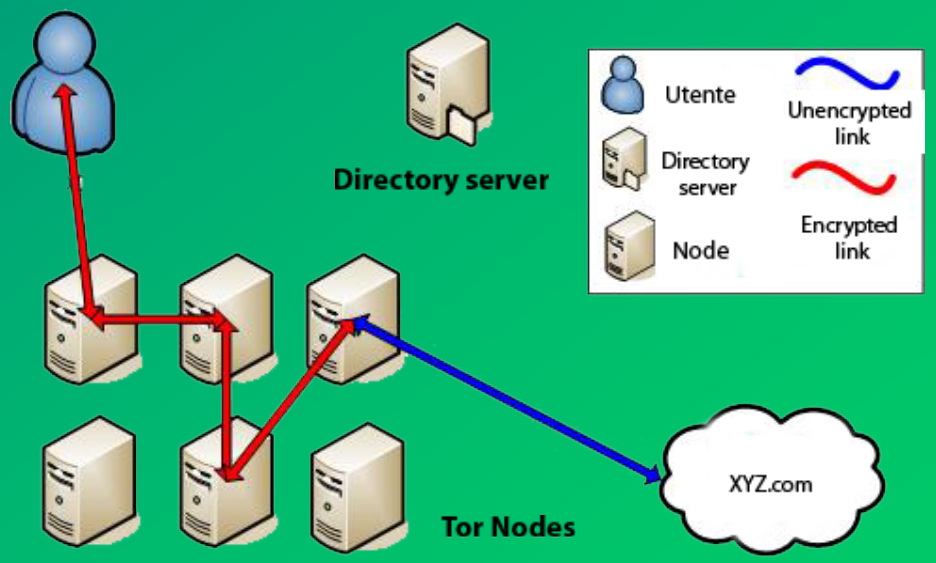
- Nel 2002 la Marina rilascia il codice a due programmatori di Boston: Roger Dingledine, ex dipendente dell'NSA e Nick Mathewson . Costoro ne continuano pubblicamente lo sviluppo fino al 2004 quando, insieme a Syverson, presentano ufficialmente TOR al tredicesimo simposio sulla sicurezza: “USENIX”.

- Nel Dicembre 2004 il progetto TOR viene adottato e sponsorizzato dall'EFF (Electronic Frontier Foundation) e da allora è possibile raggiungere tale progetto all'indirizzo tor.eff.org.

Tor: una rete Anonima distribuita

Tor

- Aiuta a ridurre i rischi derivati dall'analisi del traffico distribuendo le transazioni attraverso molti nodi della rete Internet.
- Attua l'idea di costruire un percorso tortuoso e difficile per depistare un inseguitore, cancellando periodicamente le proprie orme.
- I pacchetti di dati nella rete Tor prendono un percorso casuale attraverso molti relay che ne coprono le tracce.



Tipi di nodi

- **Client**

  Gestisce unicamente le connessioni dell'utente permettendogli di collegarsi alla rete Tor.

- **Middleman relay**

  È un nodo Tor 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.

- **Exit relay**

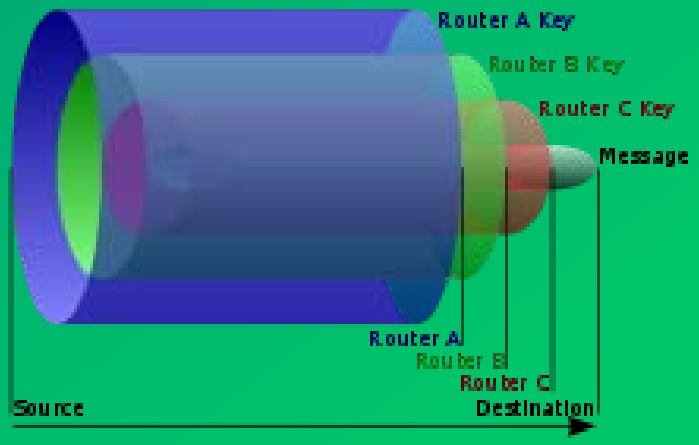
  È un nodo Tor che gestisce traffico di terzi *da* e *per* la rete Tor e *verso* l'esterno. È' possibile definire delle exit policy sulle connessioni in uscita dalla rete Tor.

- **Bridge relay**

  È un nodo di tipo sperimentale studiato per permettere di collegarsi alla rete Tor anche in presenza di un filtraggio efficace contro di essa.I bridge relay non appaiono nelle liste pubbliche dei nodi noti, ma devono essere richiesti seguendo le istruzioni all'indirizzo https://bridges.torproject.org/

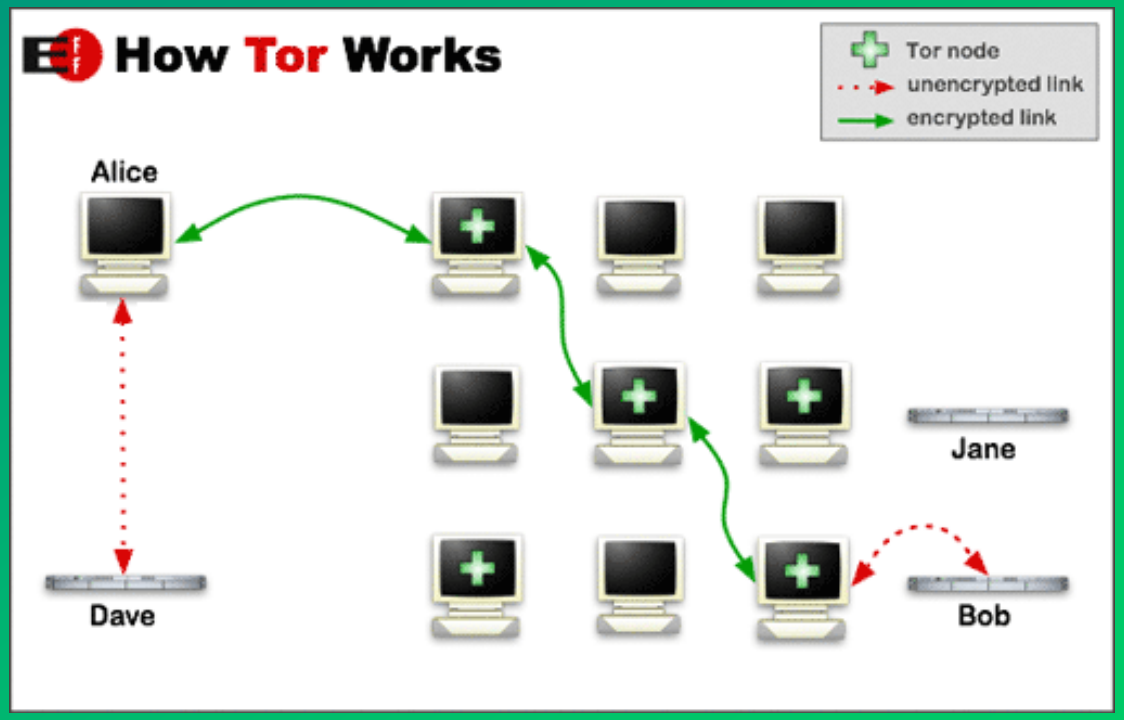
Connessione anonima in uscita

- Il software crea incrementalmente un circuito di connessioni cifrate attraverso i relay.
- Il circuito viene esteso un salto alla volta.
- Ogni relay conosce solo quale relay gli ha dato le informazioni e verso quale relay inoltrarle.
- Nessun relay conosce il percorso completo che il pacchetto ha intrapreso.
- Il software negozia un nuovo insieme di chiavi crittografiche per ogni salto lungo il circuito.



Circuito

- Un circuito Tor è il cammino attraverso la rete, scelto dai client, nel quale ogni nodo conosce soltanto il suo predecessore ed il suo successore.
- Il primo OR su un circuito è chiamato entrance router, il secondo OR è detto mix router e l'ultimo hop è l'exit router.



Tor: la tecnologia

- Gli Onion Router (OR) comunicano tra loro mediante una connessione **TLS (Transport Layer Security)**

- Ogni utente esegue un software locale chiamato Onion Proxy (OP) che:
  - Stabilisce i circuiti nella rete Tor
  - Gestisce le connessioni con l'applicazione utente
  - Accetta Stream TCP e li multiplexa lungo il circuito

Tor: la tecnologia (2)

- Ogni OR mantiene due chiavi:
  - Una identity key che principalmente firma i certificati TLS.
  - Una onion key
    - Usata per decifrare le richieste fatte dall'utente per la creazione di un circuito.
    - Usata per negoziare la chiave associata al circuito corrispondente.

Tor: la tecnologia (3)

- Il traffico Tor viene trasportato in ***celle*** di grandezza fissa (512 byte) costituite da:
- Header che include:
  - Identificativo di circuito (**CIRCID**)
  - Un comando che descrive cosa fare con il contenuto del payload.
- Payload

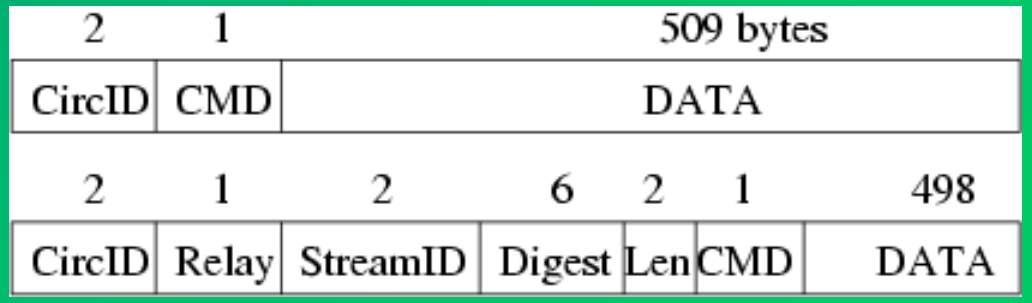
Tor: la tecnologia (4)

- Le celle, a seconda del comando, si distinguono in :
  - Celle di controllo, interpretate sempre dal nodo che le riceve
  - Relay cells che trasportano i dati
- Il campo “command” può contenere i seguenti valori:
  - 1 – CREATE (creazione di un circuito)
  - 2 – CREATED (ACK di create)
  - 3 – RELAY (End-To-End data)
  - 4 – DESTROY (stop del circuito)

Tor: la tecnologia (5)

- Le celle di tipo relay hanno un header addizionale detto relay header che contiene un campo STREAMID usato per multiplexare più stream in un circuito

- L'intero relay header e il payload vengono crifrati e decifrati usando AES – 128 bit



Tor: la tecnologia (6)

Per creare un nuovo circuito:

- l'OP manda una cell - CREATE al primo nodo da lei scelto (OR1) e sceglie un nuovo CIRCID (C1). Il payload della cella spedita, detto anche “onion skin”, contiene la prima parte dell'handshake di D.H. (g^{x1}), cifrato con l'onion key di OR1;
- OR1, dopo aver decifrato la cella con la propria chiave privata, risponde con una cell - CREATED contenente g^{y1}, con un hash della chiave negoziata $K1=g^{x1y1}$, indicato con H(K1);

Il CIRCID per una cell - CREATE è un intero arbitrario di 2 byte, scelto dal nodo (OP oppure OR) che manda la cella stessa.

Una volta che un circuito è stato stabilito, l’OP ed OR1 possono mandare una cell - RELAY cifrata con la chiave negoziata K1.

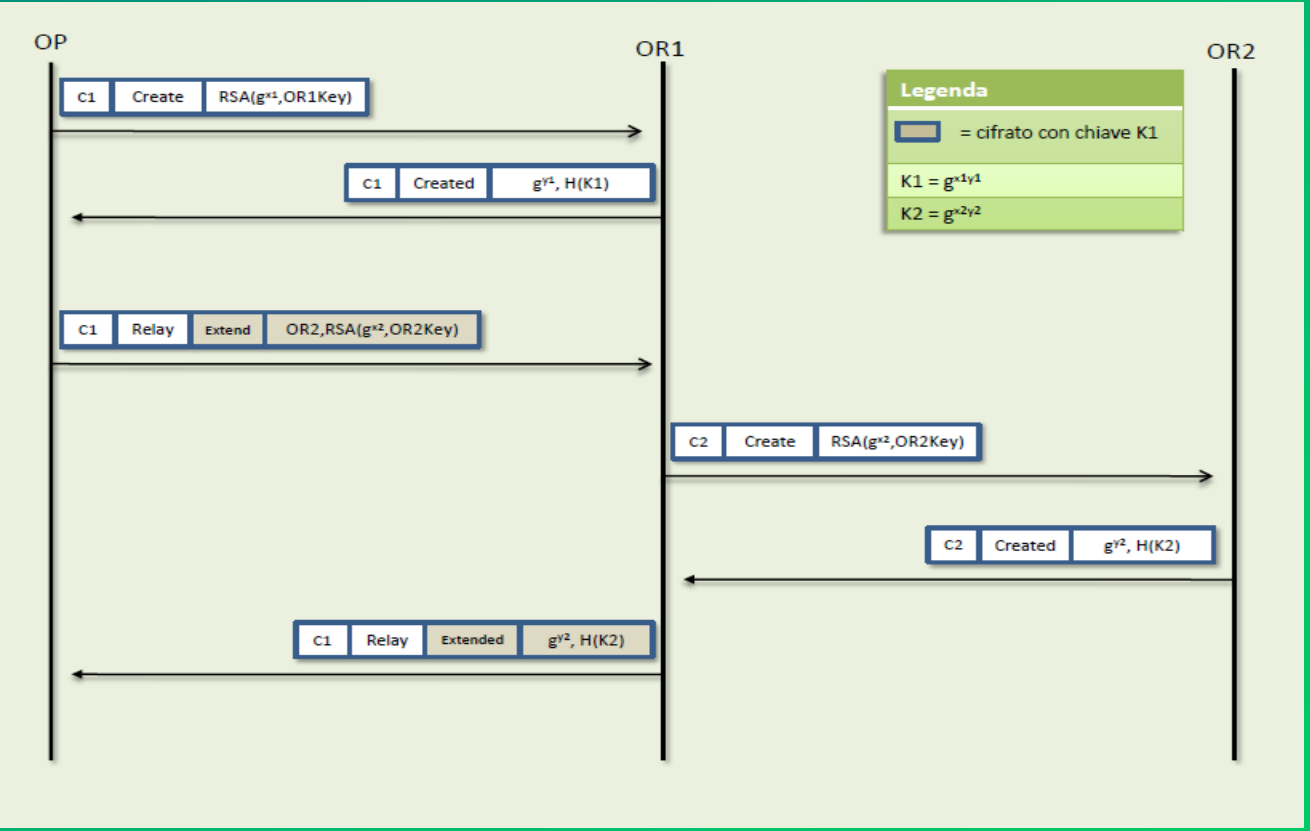
Tor: la tecnologia (7)

Per estendere il circuito

- l'OP manda una cell – RELAY EXTEND a OR1, il cui header e payload è cifrato con chiave K1 (come indicato nella figura sopra), specificando l'indirizzo del successivo hop (OR2) e la prima parte di una nuova chiave D.H. g^{x2} cifrata con la onion key di OR2 ;
- OR1 sceglie un nuovo CIRCID (C2) per la propria connessione con OR2, dunque copia la prima parte dell' handshake in una cell - CREATE e invia tutto a OR2, estendendo il circuito;
- OR2 risponde con una cell - CREATED contenente g^{y2}, con un hash della chiave negoziata $K2=g^{x2y2}$, indicato con H(K2);
- Dopo aver ricevuto la cella, OR1 immette il tutto in una cell - RELAY EXTENDED e la passa indietro ad OP. Adesso il circuito è esteso a OR2, dunque OP ed OR2 condividono una chiave comune, $K2=g^{x2y2}$.

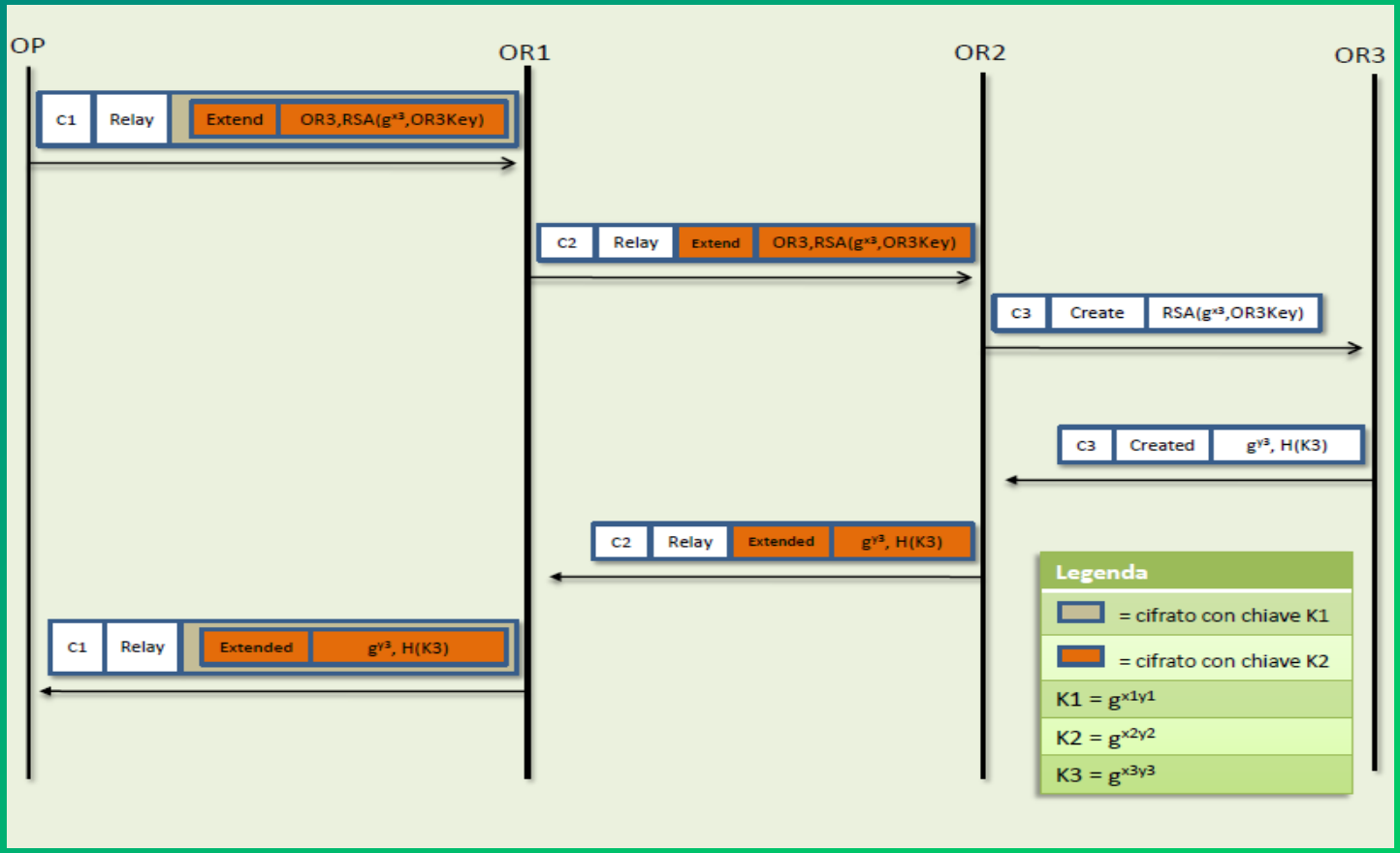
Tor: la tecnologia (8)

- La creazione ed estensione di un circuito



Tor: la tecnologia (9)

- Ulteriore estensione del circuito verso un nuovo nodo (OR3)



Tor: la tecnologia (10)

Una volta che **Alice** ha stabilito il circuito (quindi condivide una chiave con ogni OR del circuito), può mandare una cella di *RELAY.*

Per costruire una RELAY cell indirizzata ad un dato OR:

- **Alice** assegna un digest e iterativamente cifra il *RELAY* header ed il payload con la chiave simmetrica di ogni hop fino all'OR desiderato.

- OR risponde ad **Alice** con una *RELAY* cell, cifra l'header e il payload con la chiave che condivide con **Alice** e spedisce indietro il messaggio così formato.

Tor: la tecnologia (11)

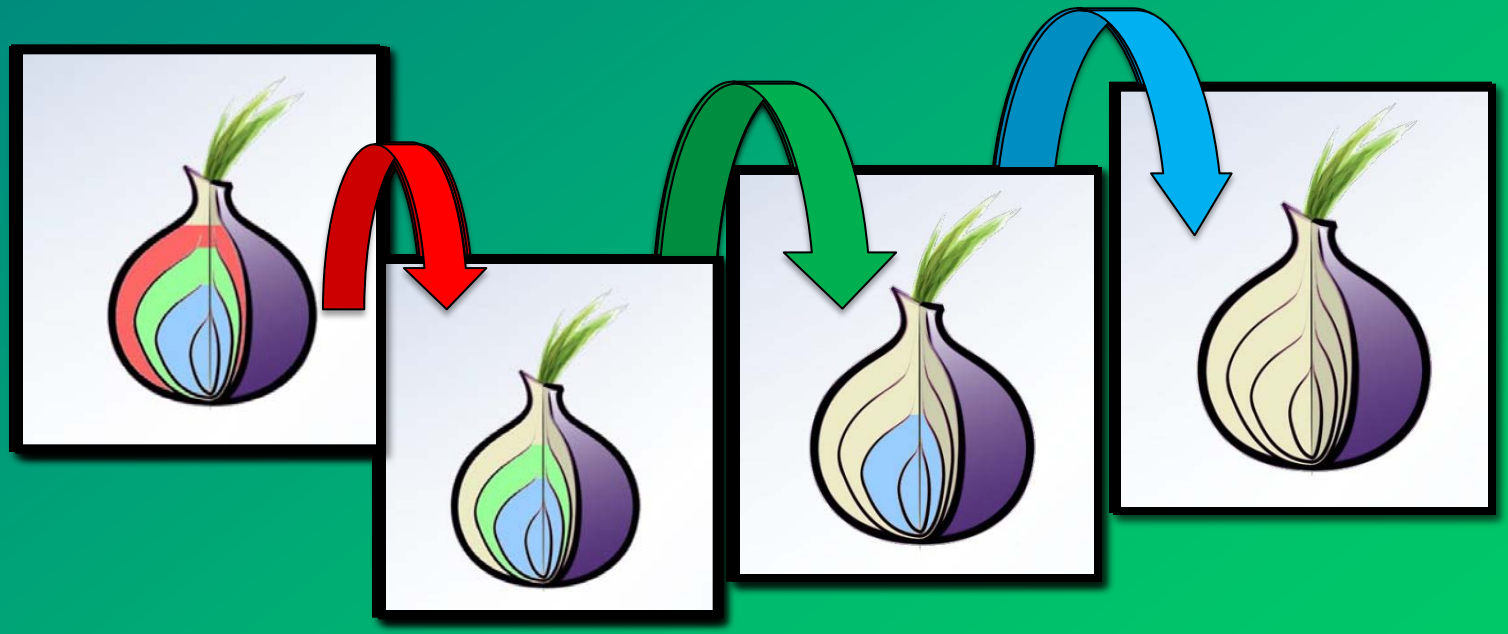
Quando un OR riceve una RELAY cell:

- controlla il ***circuito corrispondente*** e decifra il RELAY header ed il payload con la ***chiave simmetrica*** per quel circuito.
- controlla il digest: Se questo è valido, accetterà la RELAY cell e processerà le istruzioni.
- Controlla, in caso contrario, il ***CIRCID*** per stabilire il successivo hop nel circuito; sostituisce il CIRCID con quello appropriato, e manda la RELAY cell al prossimo OR.

L'OP tratta le RELAY cell in arrivo in maniera simile:

- iterativamente toglie uno strato dalla relay cell con la chiave di sessione che condivide con gli altri OR, dal più vicino al più lontano;
- OR sequenziali aggiungono ad ogni step uno strato di cifratura.

La cipolla



Hidden Services: motivazioni

Tor fornisce anonimità non solo ai clients ma anche ai servers e fa in modo che la loro localizzazione nella rete sia sconosciuta, perché:

- un server costituisce tipicamente un single point of failure: se esso viene attaccato, isolato, rimosso o distrutto, contenuti e servizi non sono più disponibili

- ogni host presente sulla rete, ma anche l'attività temporanea su una risorsa pubblica (ad es. un internet point), è riconducibile ad una persona o ragione sociale

- La conoscenza della collocazione di un server e dell'insieme dei servizi da esso offerti aumenta notevolmente la probabilità di successo di un attacco

Hidden Services: obiettivi

- **Resistenza alla censura**
  - **Il servizio deve rimanere accessibile in presenza di provvedimenti (filtraggio del traffico, redirezione DNS) atti ad impedirne la fruizione**
- **Resistenza alla violazione fisica e logica**
  - **Impossibilità di conoscere sia l'ubicazione del server sia l'insieme delle sue funzionalità**
- **Minima necessità di ridondanza**
  - **per ottenere un servizio resiliente in caso di attacchi denial of service distribuiti(DDoS)**
- **Impossibilità di risalire**
  - **a chi fornisce il servizio e/o pubblica le informazioni**
- **Impossibilità di risalire**
  - **a chi fruisce del servizio**

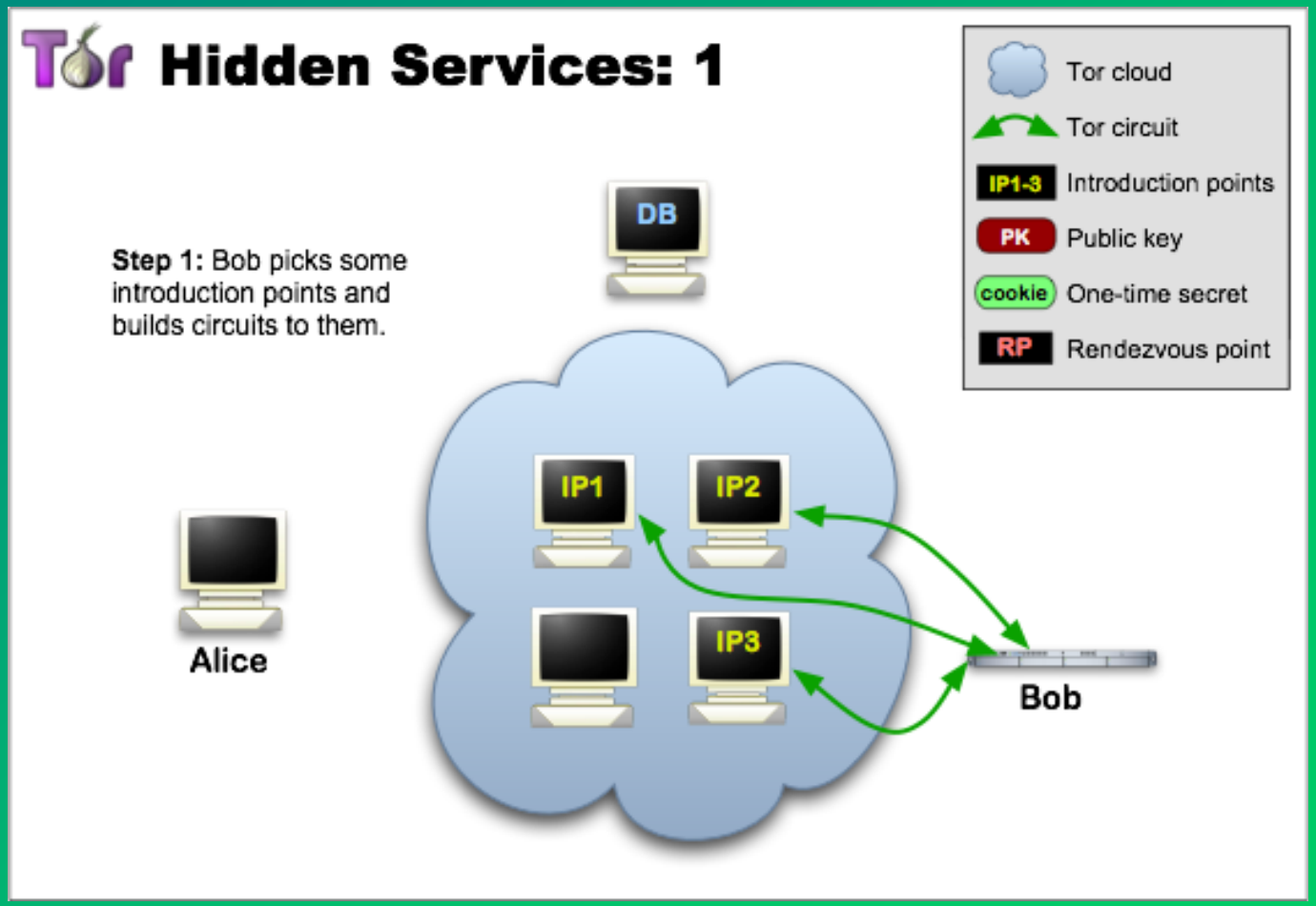
Hidden Services: Funzionamento

1. Affinché i client possano contattarlo, un hidden server deve anzitutto rendere nota la sua esistenza nella rete Tor. Dunque Bob effettua le seguenti operazioni:
   - sceglie alcuni relay a caso
   - stabilisce delle onion routes verso di essi
   - chiede loro di fungere da *introduction point* comunicandogli la propria chiave pubblica.

   Questa operazione rende impossibile associare gli Introduction point con l'indirizzo ip dell'hidden server.

Hidden Services: Funzionamento

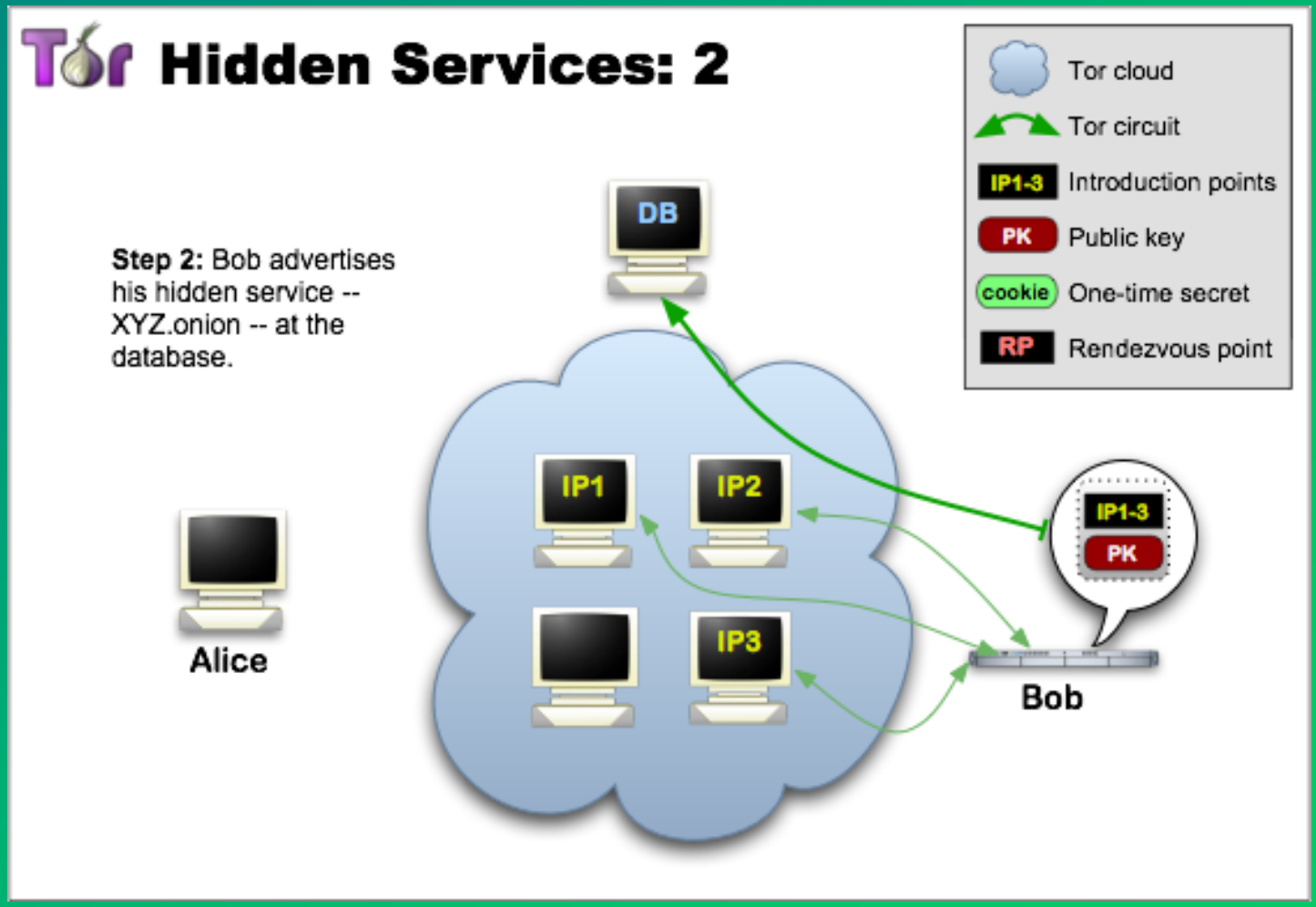
I link di colore verde sono circuiti e non percorsi diretti



Hidden Services: Funzionamento (2)

2. l'hidden server crea un *hidden service descriptor* contenente
   - la sua chiave pubblica.
   - un sommario degli introduction point .
   - firma questo descriptor con la sua chiave privata.
3. invia il descriptor ad un directory
4. Il descriptor verrà successivamente trovato dai client che richiederanno “XYZ.onion”, dove XYZ è un nome di 16 caratteri derivato in modo unico dalla chiave pubblica dell'hidden server.
5. Dopo ciò l'hidden service è attivo.

Hidden Services: Funzionamento (2)



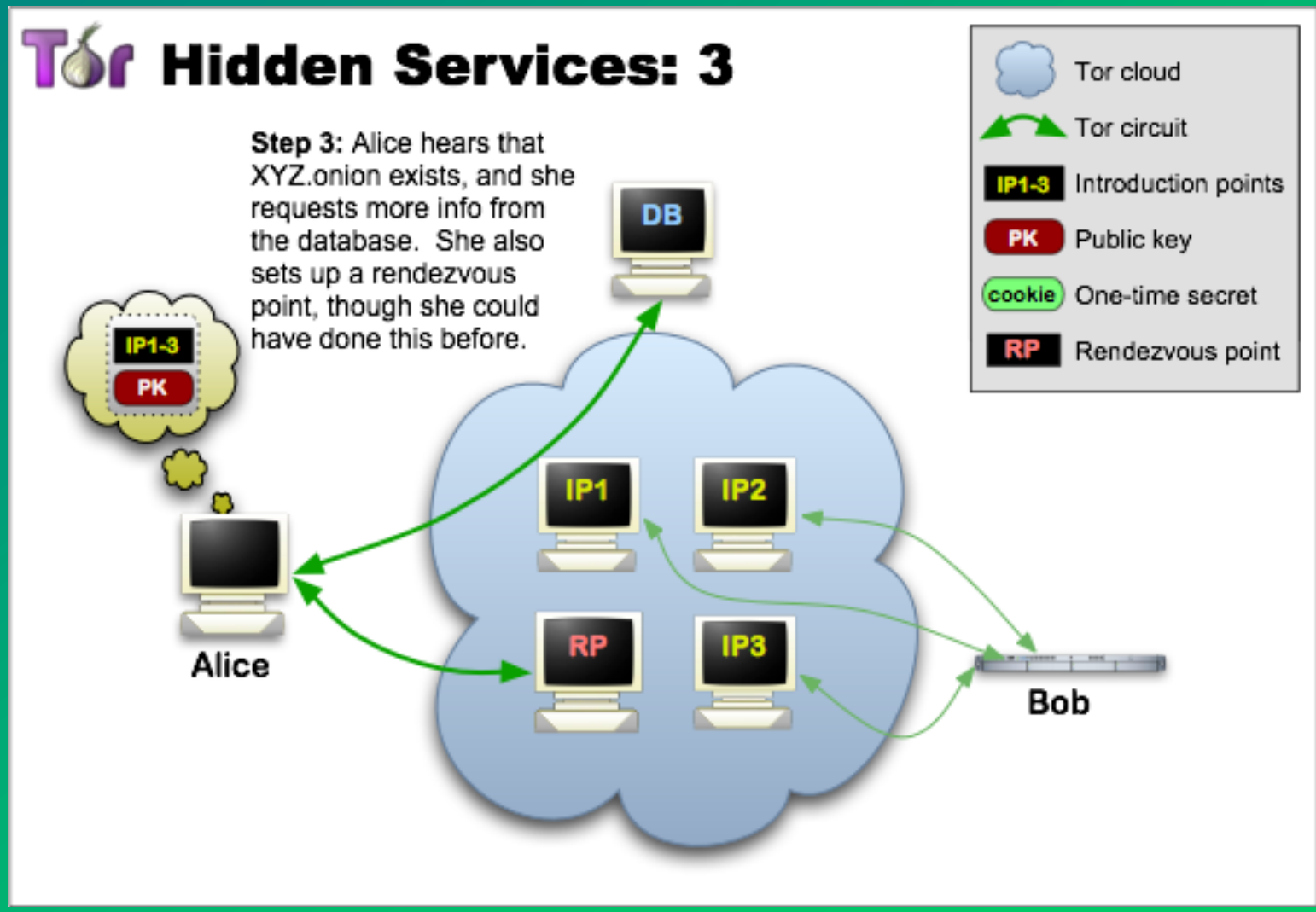
Hidden Services: Funzionamento (3)

6. Il client può iniziare a stabilire la connessione scaricando il descrittore dal directory server.

   Se esiste un descrittore per XYZ.onion , il client conoscerà il gruppo di introduction point e la corretta chiave pubblica.

7. Nello stesso momento il client crea un circuito verso un altro relay scelto a caso e gli chiede di fungere da *rendezvous point* comunicandogli un one-time secret.

Hidden Services: Funzionamento (3)

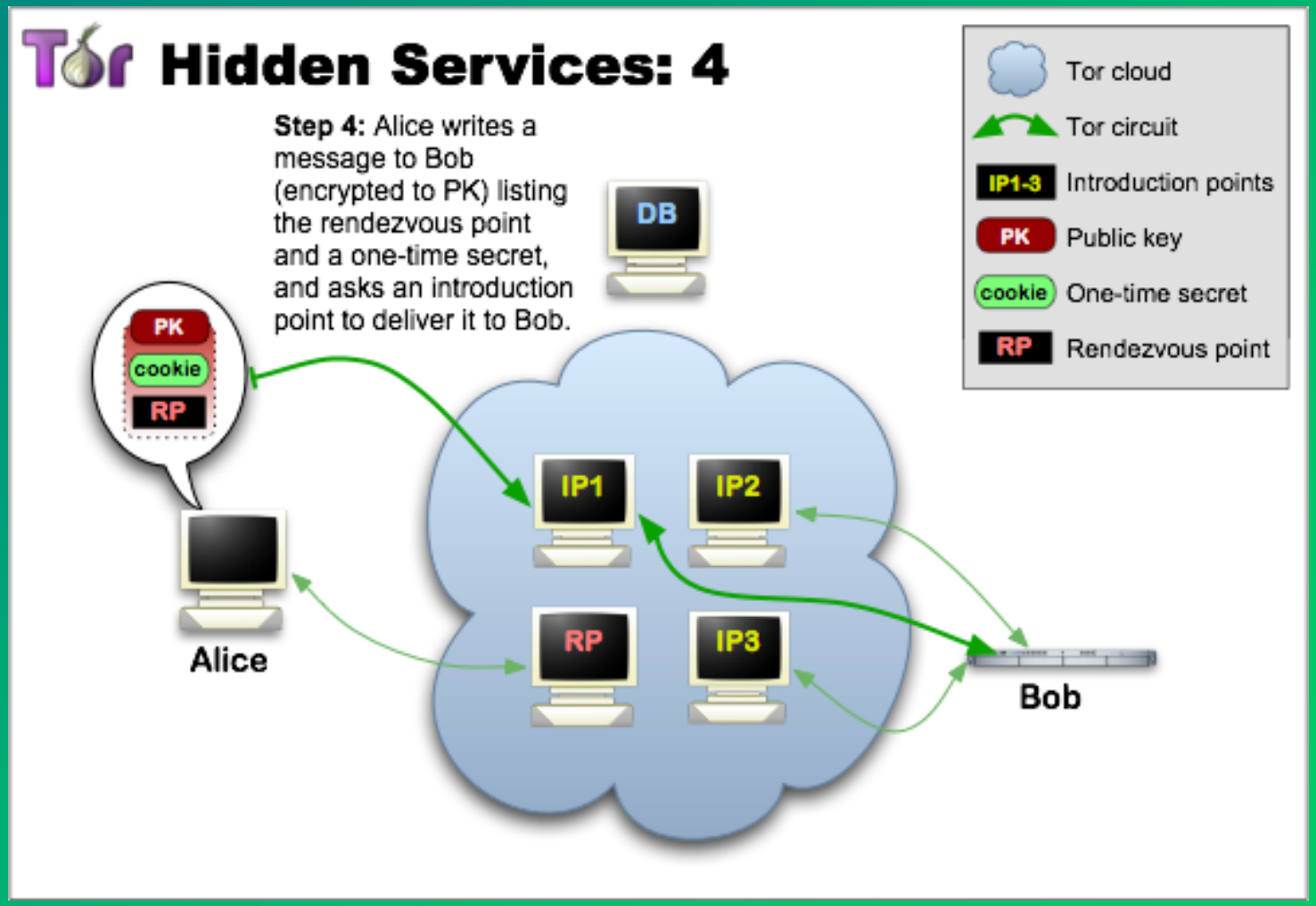


Hidden Services: Funzionamento (4)

8. Una volta ottenuto il descriptor e pronto il rendezvous point, il client crea un *introduce message* (cifrato con la chiave pubblica dell'hidden service) contenente l'indirizzo del rendezvous point ed il one-time secret.

9. Il client invia questo messaggio a uno degli introduction point, chiedendo che venga consegnato all'hidden service.

Hidden Services: Funzionamento (4)

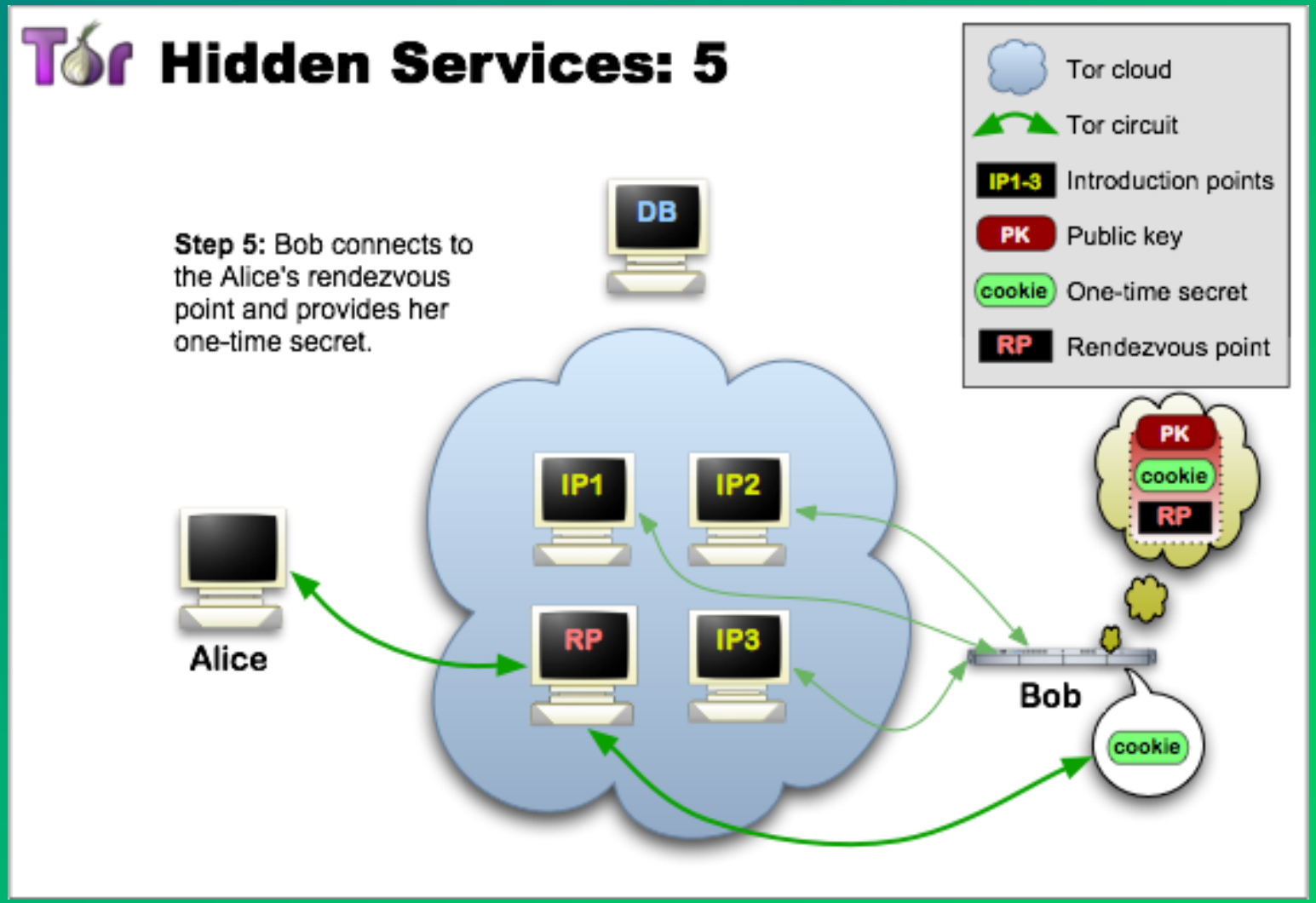


Hidden Services: Funzionamento (5)

10. L'hidden server decifra *l'introduce message* del client ed ottiene:
    - *l'indirizzo del rendezvous point*
    - *il one-time secret*

    Il server crea un circuito verso il rendezvous point e gli invia il one-time secret in un *rendezvous message*.

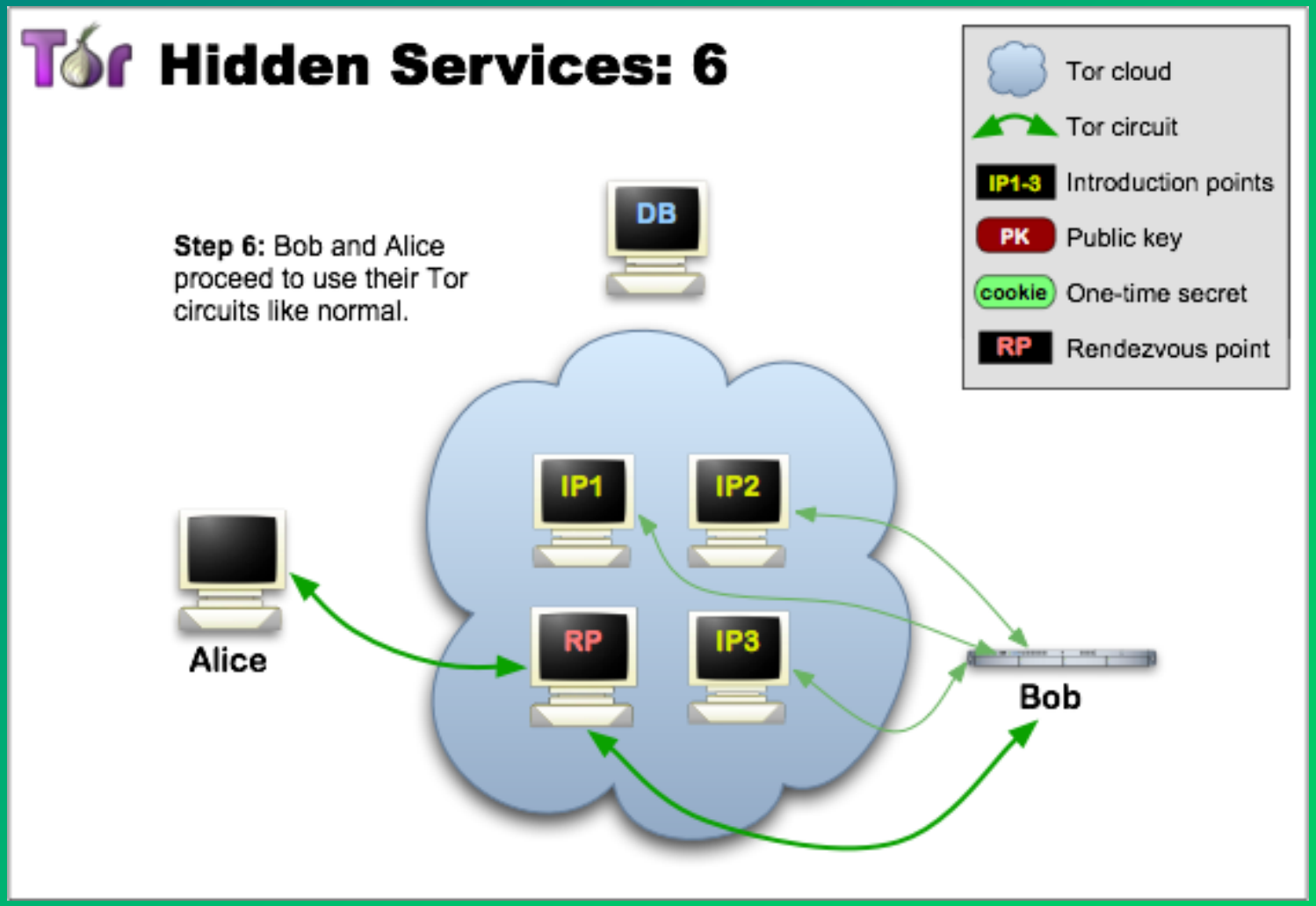
Hidden Services: Funzionamento (5)



Hidden Services: Funzionamento (6)

11. Nell'ultimo passaggio, il rendezvous point notifica al client che la connessione è stata stabilita con successo; Dopodiché il client e l'hidden service possono usare i loro circuiti verso il rendezvous point per comunicare tra di loro. Il rendezvous point inoltra semplicemente i messaggi (cifrati end-to-end) dal client al service e viceversa.

Hidden Services: Funzionamento (6)



Hidden Services: Configurazione

- E’ necessario disporre di un nodo Tor funzionante, non necessariamente in modalità server.
- La configurazione di un hidden service avviene mediante redirezione di indirizzi fittizi *[indirizzo.onion:porta]* verso indirizzi fisici *[indirizzoIP:porta]*, locali o remoti, su cui *“ascoltano”* i servizi di rete.
- Se il servizio è correttamente configurato, Tor genera automaticamente l’indirizzo *“.onion”*, che rappresenta anche la chiave pubblica del servizio, e la relativa chiave privata.

Attacchi e Difese

- Attacchi passivi
- Attacchi attivi
- Attacchi alle autorità di directory
- Attacchi contro i nodi di rendezvous

Attacchi e Difese

ATTACCHI PASSIVI

Osservazione dei pattern di traffico – Attacco

- Se osserviamo il traffico in uscita da un utente Tor non possiamo ricavarne la destinazione nè il contenuto
- Possiamo, però, profilare l'azione compiuta dall'utente
  - il traffico di un torrent è voluminoso e continuo
  - Il traffico http è intervallato

Osservazione dei pattern di traffico – Difesa

- Non è un attacco facile da portare a termine.
- Tor accorpa più flussi di dati in un singolo circuito.

Estrazione dei contenuti - Attacco

- Mentre il traffico viaggia attraverso la rete Tor, viene cifrato.
- Non è detto che lo sia anche all'uscita!
- Un attaccante può osservare il traffico in chiaro in uscita da un exit node e inferire sul client che ha inoltrato la richiesta.
- Pensiamo al protocollo http:
  - User agent
  - Cookies
  - Parametri GET e POST

Estrazione dei contenuti - Difesa

- Il filtering non è l'obiettivo principale di Tor...
- Tor utilizza Privoxy per filtrare e anonimizzare i data streams delle applicazioni (information stripping)

Correlazione temporale - Attacco

- Un attaccante, che è in grado di osservare sia il traffico del nodo di ingresso alla rete che quello del nodo di uscita, è in grado di correlare i tempi di invio e ricezione dei pacchetti.
- Conoscendo il mittente e cosa ha inviato, si può capire ciò a cui era rivolto l'interesse della vittima.
- Se il protocollo era in chiaro l'attaccante ha pure il contenuto.

Correlazione temporale - Difesa

- Per impedire all'attaccante di distinguere chi genera una connessione verso la rete Tor occorre:
  - Porre più client dietro un firewall/router così da farli sembrare uno solo
  - Elevare il ruolo del proprio client anche solo a quello di relay permette di generare diverse connessioni.
  - Evitare di scegliere nodo di entrata e di uscita nella stessa sottorete o in reti molto vicine tra loro.

Correlazione sulla dimensione dei pacchetti - Attacco

- Tale attacco è simile al precedente, ma anzichè osservare i tempi di risposta tra ingresso e uscita, l'attaccante controlla i volumi di traffico scambiati:

  se riesce a inferire pattern comportamentali simili, allora è possibile correlare i due nodi.

Correlazione sulla dimensione dei pacchetti - Difesa

- L'utilizzo del padding delle connessioni sarebbe ottimo.
- Non è ancora utilizzato per evitare di impattare sulle performance di rete.
- L'attacco viene mitigato dal comportamento stesso dei nodi:
  - Essi accorpano più flussi di informazione nei circuiti che costruiscono
  - Volumi di traffico differenti tra pacchetti in ingresso e in uscita.

Fingerprinting – Attacco/Difesa

- **Attacco:**
  - Simile al precedente ma elimina la necessità di osservare l'uscita.
  - L'attaccante conosce un set di dati e patterns di accesso. (websites)
  - L’attaccante 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 patterns.
- **Difesa**:
  - Stessa del punto precedente

Attacchi e Difese

ATTACCHI ATTIVI

Compromissione di chiavi - Attacco

- Se un attaccante conosce la chiave di sessione di un nodo, è in grado di vedere le celle di controllo e le relay-cells cifrate su quella connessione.
- Se l’attaccante conosce la chiave di sessione di un circuito potrebbe essere capace di eliminare un layer della cifratura.
- Compromettere un OR potrebbe permettere all’attaccante di proseguire lungo il circuito compromettendo anche gli altri nodi fino a raggiungere la fine.

Compromissione di chiavi - Difesa

- La rotazione rapida delle chiavi di sessione TLS limita questo tipo di attacchi.
- Conoscere la chiave di sessione di un circuito consente di decifrare solo uno strato della onion.
- Conoscere la chiave di sessione TLS di un nodo comporta necessariamente di apprendere anche la onion key per poter decifrare le cell-CREATE.
- La compromissione dei nodi lungo un determinato circuito deve avvenire necessariamente entro il tempo di vita del circuito stesso, altrimenti gli OR avranno già cancellato le informazioni necessarie all'attaccante per linkare la sorgente con la destinazione.

Tagging – Attacco/Difesa

- **Attacco**
  - Un attaccante può modificare il traffico generato dai propri nodi inserendo dei tag arbitrari nei flussi di informazione scambiati.
  - La registrazione e ricerca dei tag permette di svelare il cammino di un client.
- **Difesa**
  - Il controllo di integrità dei dati previene questo tipo di attacco.

Nodi e client – Attacco

- Puntano a rivelare l'identità di un nodo attaccandone l'ambiente circostante.
- L'utilizzo di plugin per i browser web permette di bypassare l'utilizzo di Tor:
  - Java, Javascript, Actionscript permettono tutti di eseguire connessioni, ignorando totalmente il proxy configurato
  - I plugin multimediali soffrono di problemi simili. Inoltre possono filtrare informazioni eseguendo risoluzioni dns dirette, bypassando le impostazioni del browser

Nodi e client – Difesa

- Utilizzare l’estensione TorButton per Firefox permette di ridurre i rischi:
  - Disabilita i plugins
  - Isola le sessioni di navigazione
  - Ripulisce le informazioni sensibili registrate
  - Esegue spoofing di user agent, locale e timezone
- Utilizzare un proxy trasparente delle connessioni
- Aggiornare sempre all'ultima versione di Tor rilasciata dal sito ufficiale: https://www.torproject.org/

Altri tipi di attacchi attivi

- **Onion Proxy remoti**
  - In alcune configurazioni l'OP gira in remoto piuttosto che in locale come nelle istituzioni che desiderano monitorare l'attività di coloro che si connettono al proxy. Quindi, compromettendo un OP, si comprometteranno tutte le future connessioni che passano attraverso di esso.

- **Smear attacks**
  - Un attaccante potrebbe usare la rete Tor per commettere azioni non approvate allo scopo di abbassare il grado di reputazione della rete e costringere gli operatori a chiuderla. Le **exit policies** riducono la possibilità di abusi.

- **Dos non-observed node:**
  - Un attaccante, che può osservare solo una parte della rete Tor, potrebbe incrementare il volume di traffico verso i nodi non controllati per chiuderli così da ridurre la loro affidabilità oppure convincere gli utenti che tali nodi non sono fidati.

Attacchi e difese

ATTACCHI ALLE AUTORITA' DI DIRECTORY

Destroy directory servers – Attacco/Difesa

- **Attacco:**
  - Il consenso sullo stato della rete viene generato dalle autorità di directory votando in base alla loro visione della rete.
  - L'obiettivo dell'attaccante è abbattere alcune di queste autorità per indurre la pubblicazione di uno stato non valido o non funzionante.
- **Difesa:**
  - Se più della metà cade è necessario un intervento manuale dell'operatore del client per accettare il consenso.

Compromettere la maggioranza dei directory servers - Attacco/Difesa

- **Attacco:**
  - Un attaccante che controlla più della metà dei directory servers può introdurre degli OR maligni all'interno della rete.
- **Difesa:**
  - Indipendenza dei directory servers.
  - Resistenza agli attacchi.

Splitting – Attacco/Difesa

- **Attacco**
  - L'attaccante mira a raggirare uno o più operatori in modo che tolgano la fiducia verso un terzo (o terzi)
  - Così facendo si ottiene uno split del consenso generato dalle autorità
  - Si possono isolare gli utenti a seconda di quale consenso abbiano ricevuto
- **Difesa:**
  - Nessuna
  - Rete di fiducia
  - Numero autorità limitato

Tricking – Attacco/Difesa

- **Attacco**
  - L'attaccante convince un’ autorità a listare come non validi uno o più nodi perfettamente funzionanti.
  - O viceversa.
- **Difesa:**
  - Il blacklisting è attualmente manuale
  - Si sta sviluppando un modo per la ricerca automatica di nodi di uscita malfunzionanti

Attacchi e difese

ATTACCHI CONTRO I NODI DI RENDEZVOUS

Flooding – Attacco/difesa

- **Attacco**
  - La tecnica più comune per attaccare i servizi nascosti è floodare di richieste fittizie i nodi che si incaricano di eseguire il rendezvous tra client e servizio.
  - Un nodo floodato non può più svolgere il proprio lavoro, impedendo la localizzazione dei servizi.
- **Difesa:**
  - Token di autorizzazione delle richieste
  - Limitazione dei rate di richiesta

Compromissione nodi - Attacco

- Un circuito a un hidden service è esattamente identico a un circuito verso un host esterno alla rete torificata cambia solo la destinazione.
- Se un attaccante compromette un nodo di introduzione può rigirare il traffico dove vuole oppure effettuare flood di introduction request.
- Se un attaccante compromette il nodo di rendezvous può iniettare contenuti nella comunicazione.
- Un attaccante potrebbe distruggere un location-hidden service disabilitando tutti i suoi introduction point

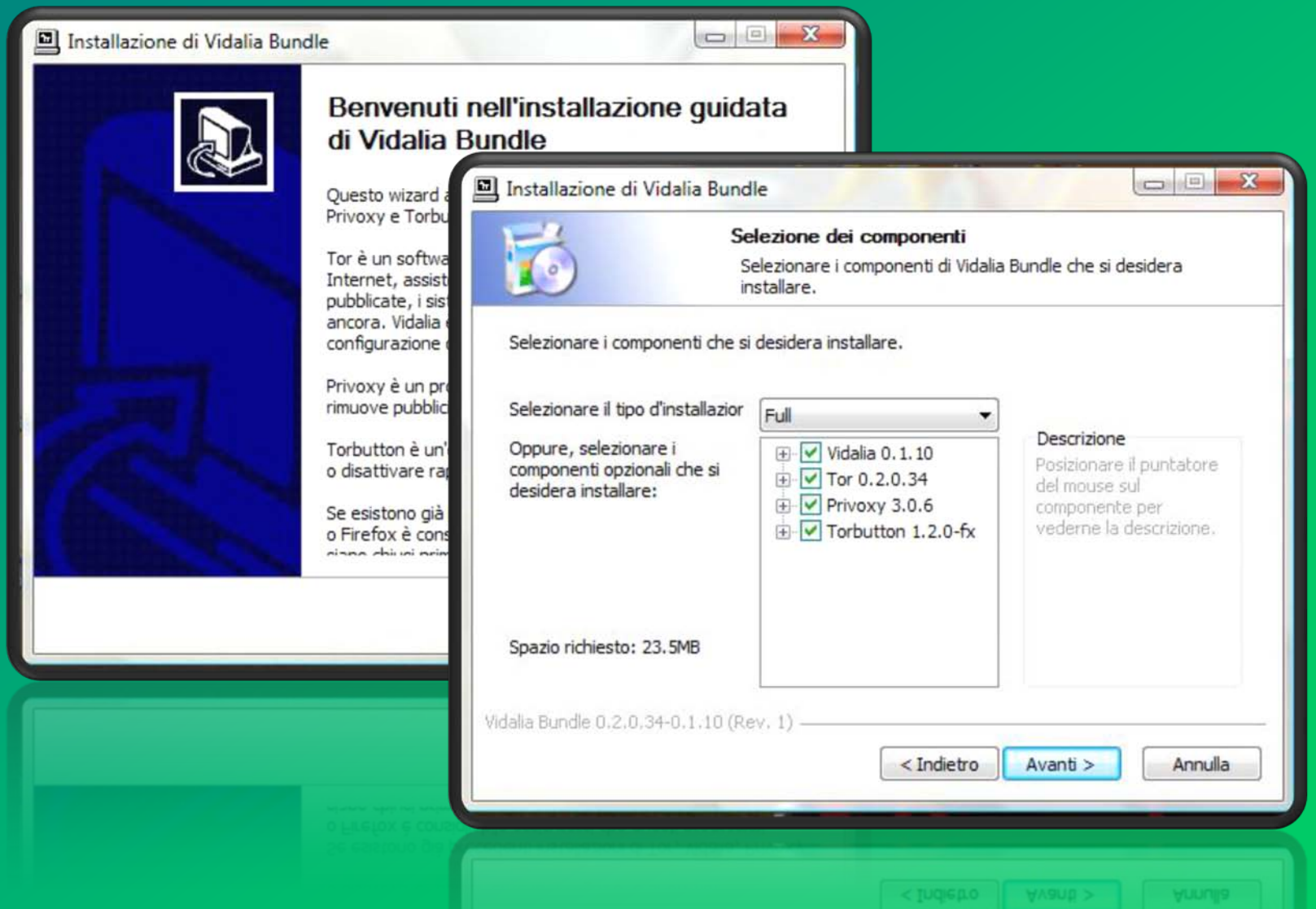
Compromissione nodi - Difesa

- Il servizio nota il flood di richieste e chiude il circuito con l'introduction point.
- Un rendezvous point non è diverso dagli altri OR, dunque si applicano le stesse difese viste in precedenza.
- Poiché l'identità del servizio viene definita dalla sua chiave pubblica, il servizio può semplicemente riannunciarsi verso introduction points diversi; ciò costringe l'attaccante a disabilitare tutti i possibili Introduction points.

Tor in 4 semplici passi

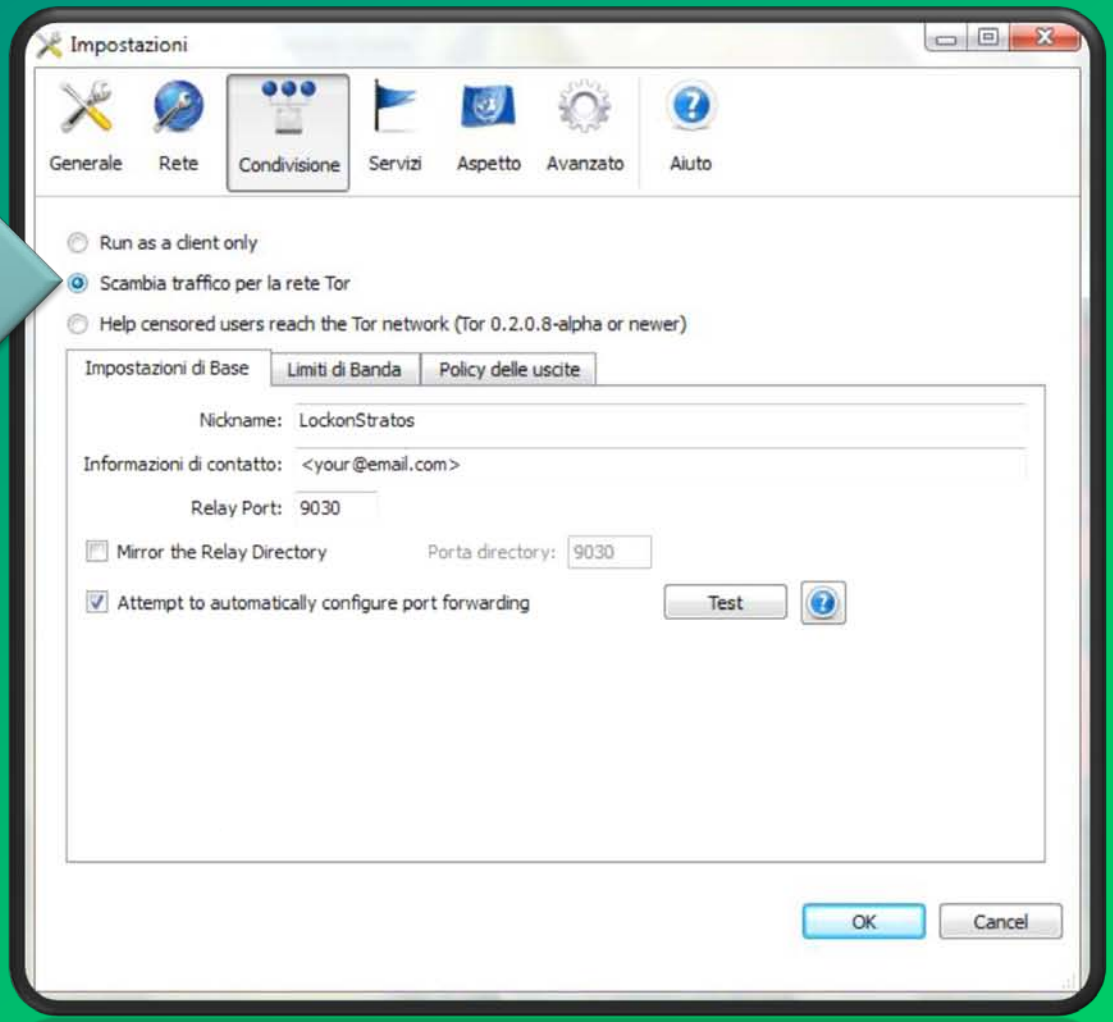
- Installazione pacchetto *vidalia bundle* reperibile nella sezione download di *tor.eff.org*
- Configurazione della macchina come relay Tor
- Connessione alla rete Tor
- Navigazione anonima mediante il plug-in “tor button”
- E’ possibile seguire un tutorial a questo indirizzo: http://www.youtube.com/watch?v=wBNwIIZwXwU

Installazione Vidalia Bundle



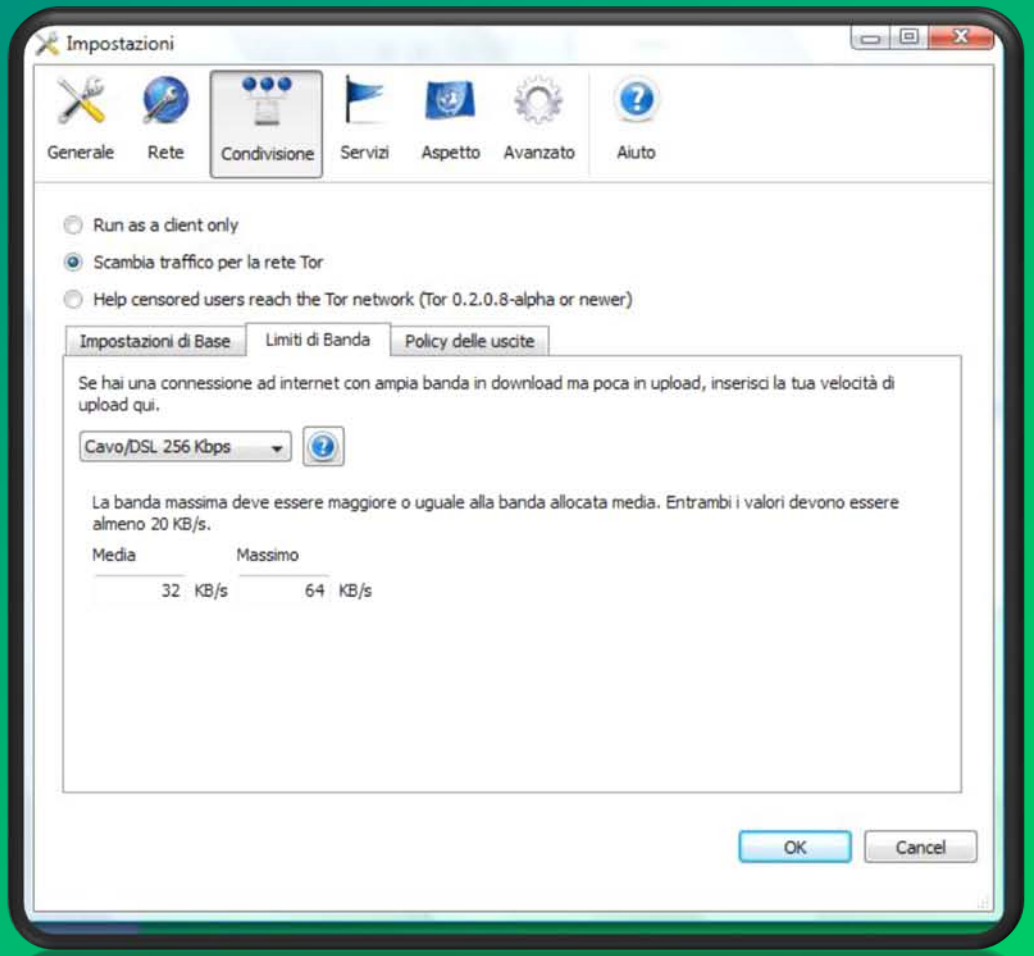
Configurazione della macchina come relay Tor

Selezioniamo “scambia traffico per la rete Tor” per poter fare da relay.



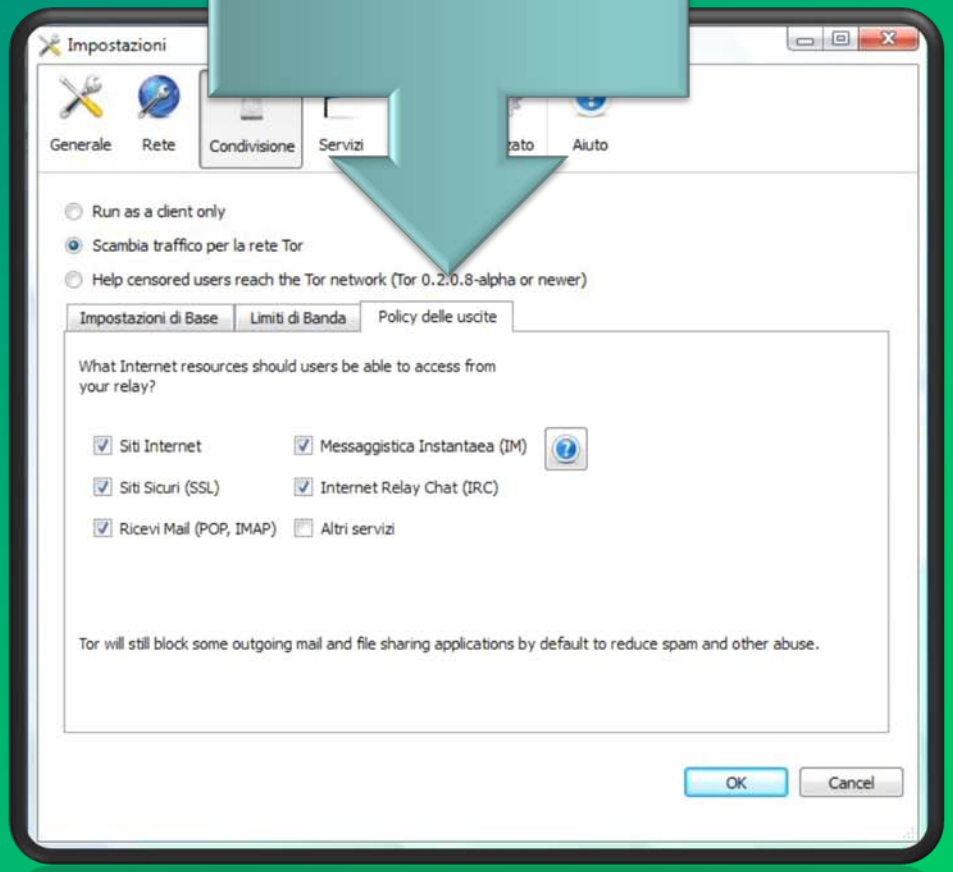
Configurazione della macchina come relay Tor (2)

Impostiamo i limiti di banda in uscita



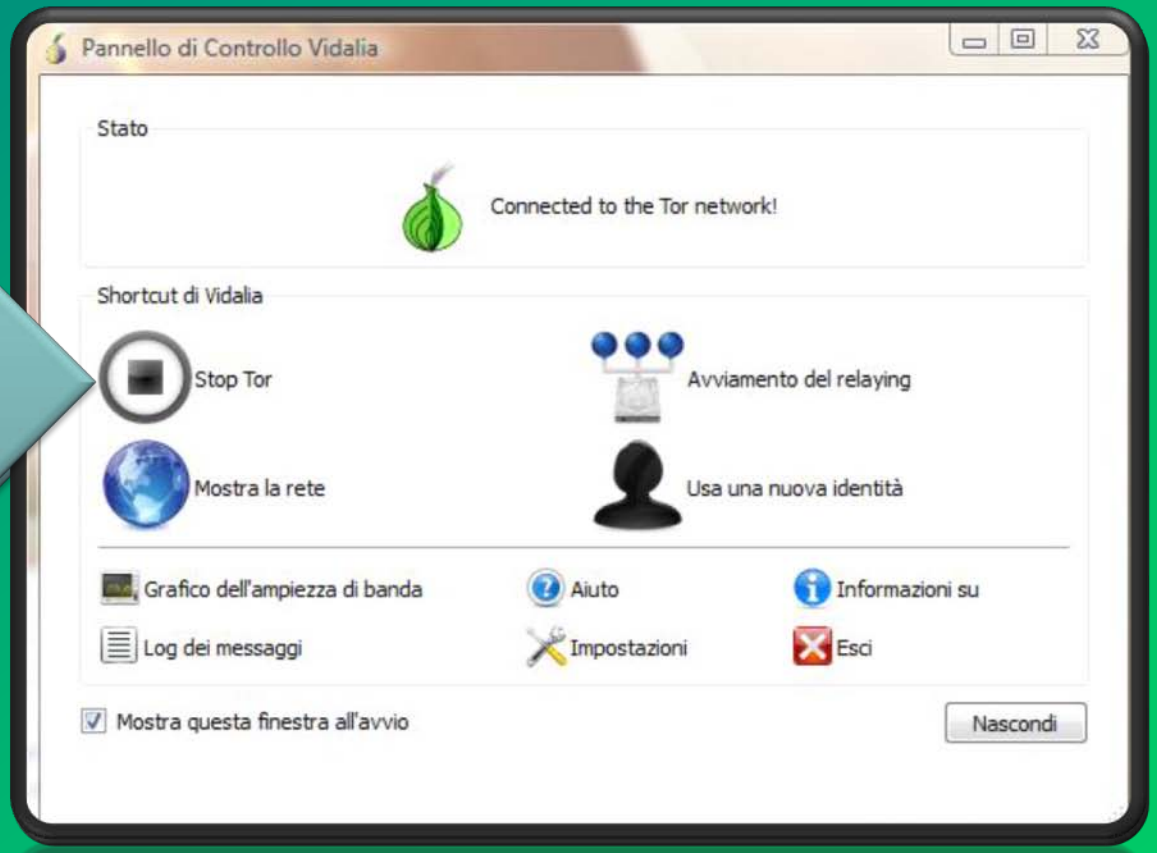
Configurazione della macchina come relay Tor (2)

Le exit policies

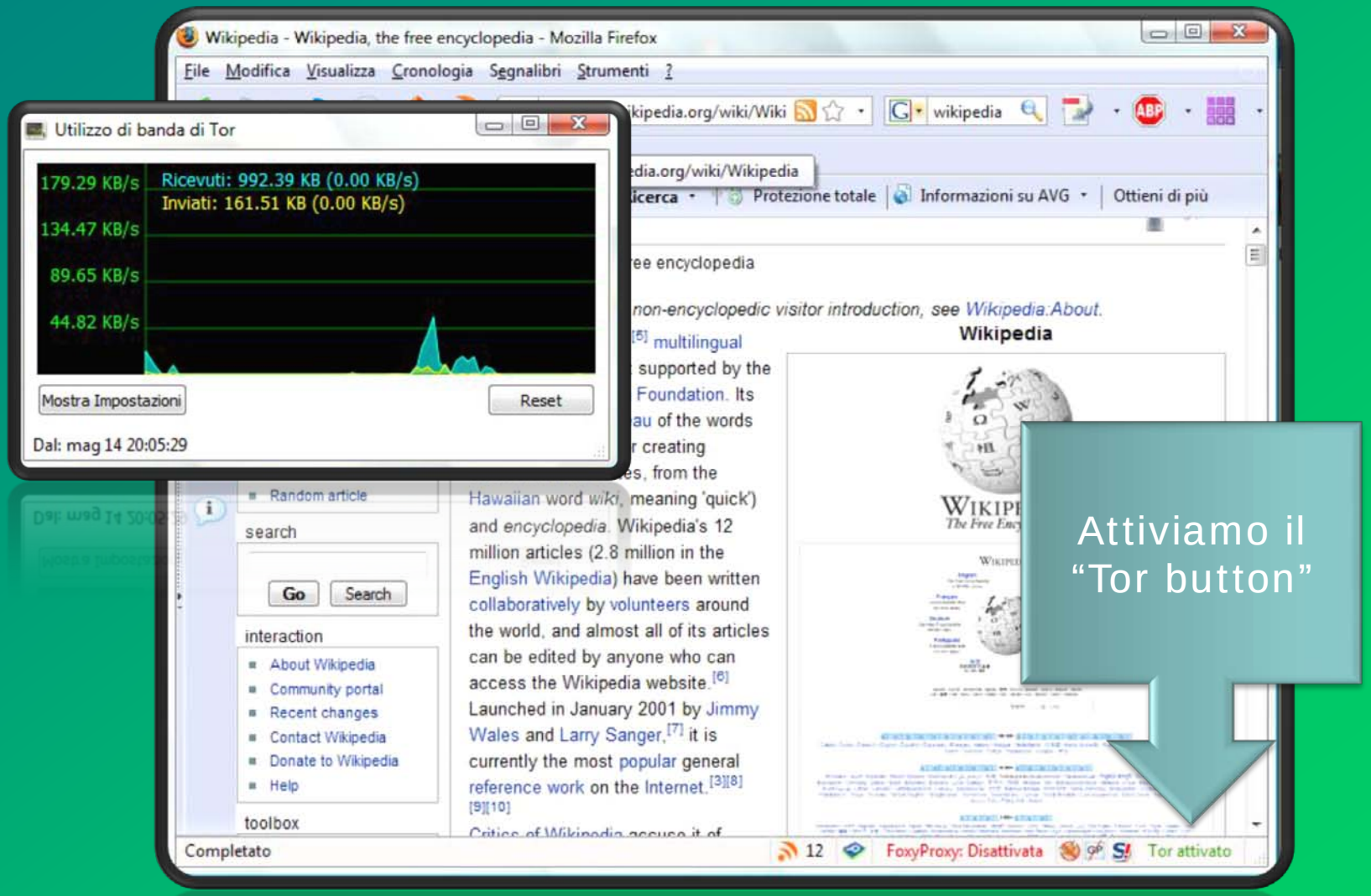


Connessione alla rete Tor

Avviamo la connessione utilizzando il pannello di controllo



Navigazione anonima



Etiquette e abuso

Le Exit policies, definite sugli exit nodes, regolamentano il tipo di traffico in uscita mitigando gli eventuali abusi della rete Tor.

Potenziali abusi sono:

- **Esaurimento della banda**

  È scortese e improprio trasferire grandi quantità di dati attraverso la rete Tor come con software peer-to-peer, poichè gli onion router sono mantenuti da volontari che donano la propria banda.

- **E-mail**

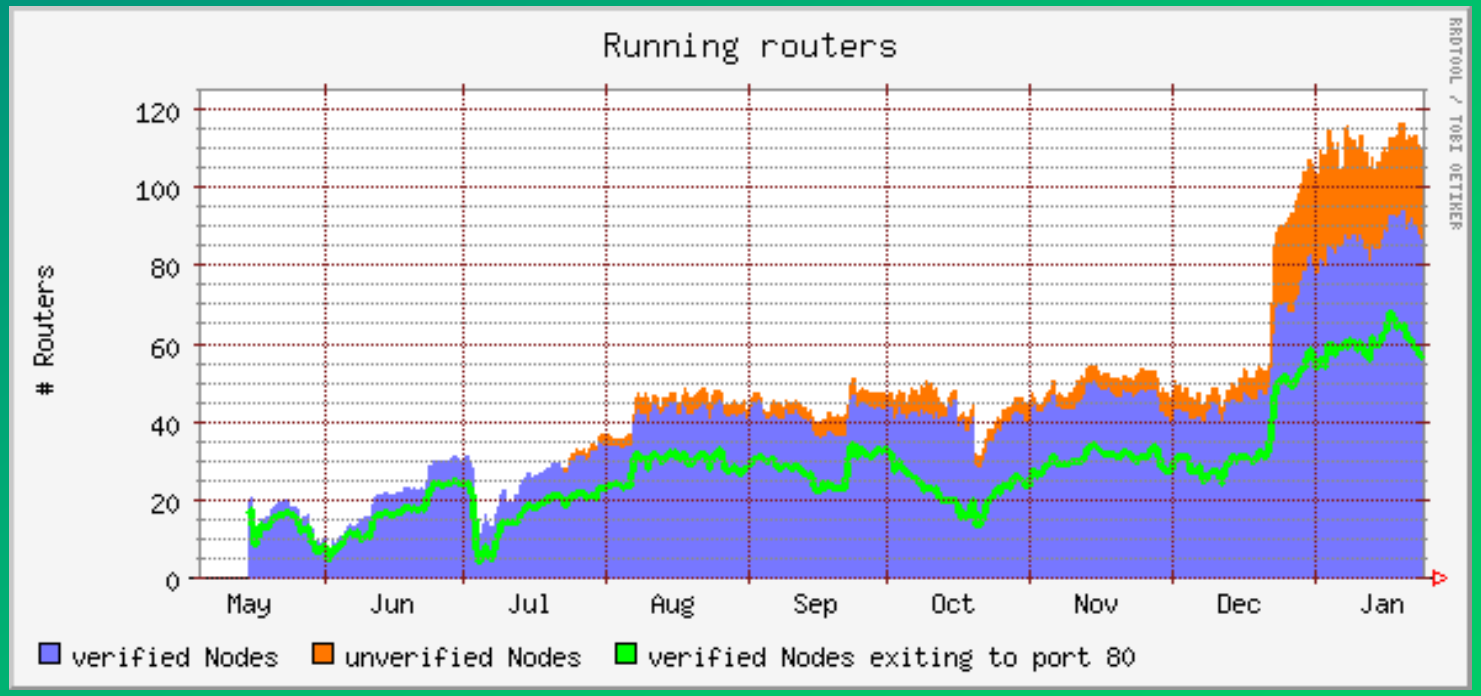
  L'uso anonimo del protocollo SMTP può portare allo Spam.

- **Vandalismo**

  Sicuri del fatto di non poter essere rintracciati, alcuni utenti Tor scrivono messaggi ritenuti impropri su Forum, wiki, o chat-room. Come risultato, molti grandi provider di questi servizi impediscono agli utenti anonimizzati di utilizzare i propri servizi.

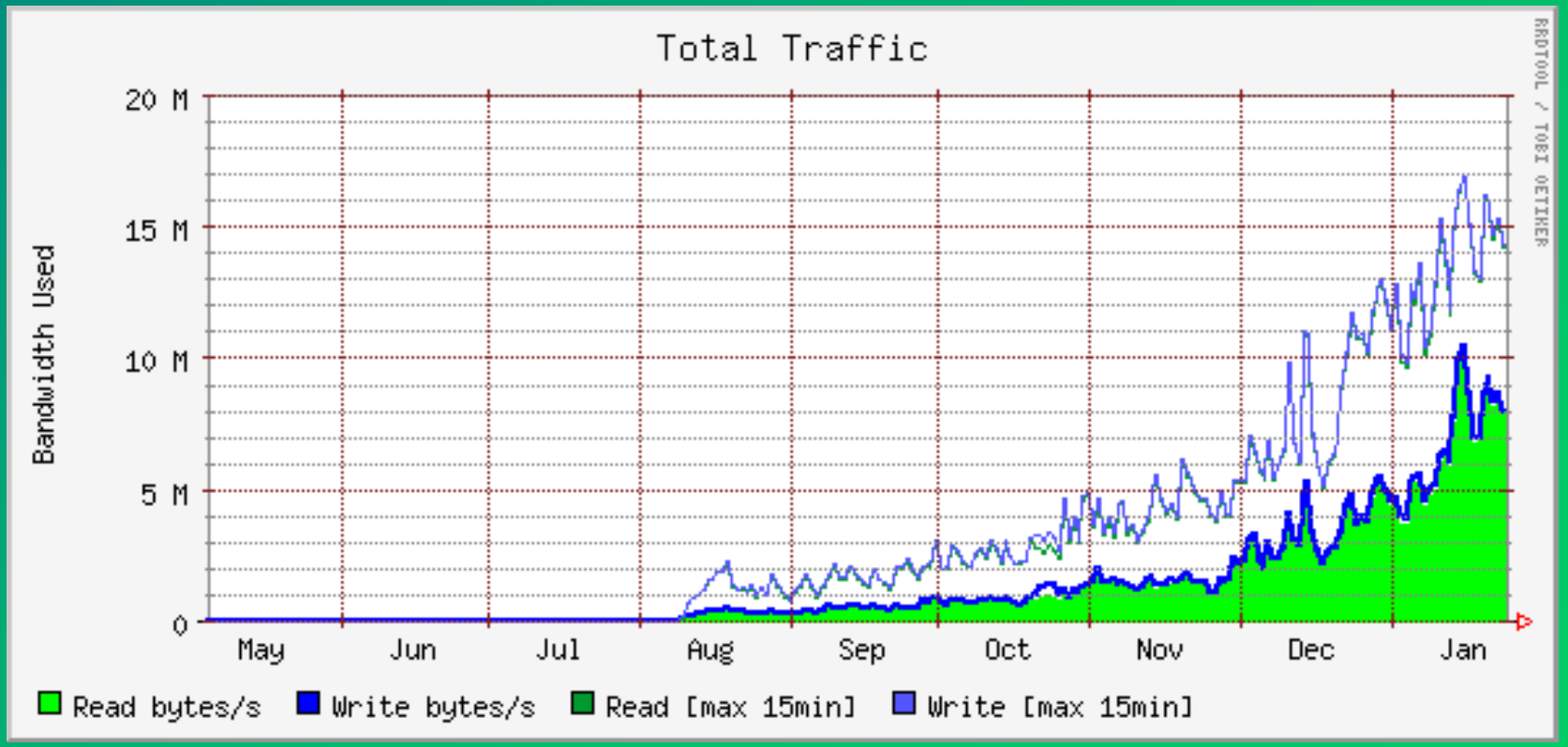
Sviluppo Tor nel tempo

- Nell'ottobre 2003 la rete Tor diviene funzionante
- Da quel momento raggiunge centinaia di nodi operativi con un traffico medio di 80 megabits.
- A partire da gennaio 2005, Tor ha raggiunto centinaia di nodi nel mondo con una capacità totale pari a 1 Gbit/s.



Sviluppo Tor nel tempo (2)

Somma del traffico riportato da ciascun nodo (gennaio2005)



Il futuro di Tor

- Un approccio basato sul volontariato potrà sostenere la rete in un futuro a lungo termine ?
- Essendo un software libero, la rete potrebbe cessare di esistere se gli sviluppatori smettessero di lavorarci.
- Tor è soltanto uno dei molti componenti che preservano la privacy online. Si dovrebbero implementare molti e migliori proxy “protocol-aware” usabili dalle persone comuni.
- E’ necessario che Tor impari a coesistere con la varietà di servizi internet ed i loro meccanismi di autenticazione.
- L’attuale architettura Tor non è sufficientemente scalabile per gestire la domanda di utenti attuale.

Conclusioni

La rete Tor

- Rappresenta senza dubbio il miglior servizio pubblico di anonimato esistente, ma un suo utilizzo poco coscienzioso puo’ rivelarsi di nessuna utilità per la tutela della privacy.

- Si regge sulla “fidatezza” dei Directory server e questi sugli OR fidati.

Piattaforme supportate

- Gnu/linux
- MacOS X
- MS Windows

Partecipazione attiva

Il sito di riferimento è ***http://tor.eff.org***

Diversi modi per collaborare:

- Installare e usare Tor
- Condividere la propria banda con un server “exit node”
- Sviluppare il software
- Fare una donazione per supportare lo sviluppo

The Onion Router

GRAZIE PER L’ATTENZIONE