Prof. Alfredo De Santis

The Onion Router

"L'instradatore di *cipolle*"

Sicurezza su Reti II

Studente: Michele Falso matr. 0521/000436
11/03/2010

1. Introduzione

1.1. Privacy ed Anonimato

Anonimato: "Lo stato di non essere identificabile in un insieme" (Gianni Bianchini).

Il tema dell'anonimato e della Privacy, legato alla navigazione oppure alla posta elettronica, è estremamente ampio ed alquanto tecnico. Basti sapere che quando si naviga in Internet ci lasciamo dietro una traccia specifica, *l'indirizzo IP*, che identifica in maniera univoca in tutto il mondo il nostro PC. Conoscendo l'indirizzo IP, è possibile individuare la rete, la sottorete ed il provider a cui siamo connessi e, in molti casi, anche la nostra città. In Rete, senza accorgimenti, non esiste privacy. L'origine, la destinazione ed anche il contenuto di ogni comunicazione sono identificabili presso ogni nodo di rete intermedio con elementari tecniche di analisi del traffico. Infatti, i pacchetti dati su internet sono divisi in due parti: il *blocco dati* e l'*intestazione*. Mentre questa viene utilizzata per l'instradamento dei pacchetti il blocco dati, invece, contiene le informazioni che vengono inviate, siano esse una e-mail, una pagina web o un file musicale. Anche se il blocco dati viene cifrato, l'analisi del traffico continua a rivelare informazioni su quello che si sta facendo e, possibilmente, su quello che si sta dicendo. Questo perché tale tipo di analisi si concentra sull'intestazione del pacchetto dati, che fornisce sorgente, destinazione, dimensione e tempi. Un problema basilare per coloro che sono attenti alla privacy è che il destinatario di una comunicazione può sapere, attraverso l'analisi dell'intestazione del pacchetto, chi lo sta mandando. La stessa cosa possono fare gli intermediari che ricevono il flusso dei pacchetti, come ad esempio gli Internet Service Provider (ISP) e, talvolta, anche gli intermediari non autorizzati. Una forma molto semplice di analisi del traffico consiste nel porsi in un punto qualsiasi tra la sorgente e il destinatario della comunicazione e studiare le intestazioni dei pacchetti.

Vi sono però altri e più potenti metodi di analisi del traffico. Alcuni attaccanti spiano molte aree di Internet e usano sofisticate tecniche statistiche per carpire schemi di comunicazione tra diversi individui e organizzazioni. Cifrare i messaggi non serve molto in caso di un attacco del genere, poiché in questo modo si nasconde solo il contenuto del traffico Internet e non le intestazioni dei pacchetti. Inoltre, nessuno è completamente libero di esprimersi; la pubblicazione sul web è facilmente censurabile attaccando, per via informatica o legale, un singolo server. In Rete senza adeguate misure di sicurezza nessuno è anonimo; ogni connessione ad un provider comporta l'archiviazione di ora e numero di telefono in file di log oltre che di altre informazioni che contengono traccia del nostro passaggio e di cosa siamo andati a cercare. Inoltre, recenti disposizioni di legge obbligano i gestori degli internet point ad identificare i client, altre obbligano i fornitori di accesso a conservare a lungo i file di log. Preservare la privacy non significa soltanto nascondere il contenuto dei messaggi, ma anche nascondere l'emittente al ricevente.

Dal punto di vista dello sviluppo fisico, una semplice applicazione di crittografia entro una rete "packet-switched" nasconde i messaggi che vengono spediti, l'emittente, il ricevente e la frequenza delle comunicazioni.

È possibile evitare di essere individuati con tanta precisione e di lasciarsi dietro questo tipo di tracce ?

La risposta risiede nell'utilizzo attento delle tecnologie per la comunicazione anonima, in particolare l'*Onion routing*, una infrastruttura general purpose per la comunicazione privata su una rete pubblica che fornisce connessioni anonime fortemente resistenti sia per coloro che spiano la comunicazione sia per l'analisi del traffico. Le connessioni sono bidirezionali e possono essere usate sia per il traffico connection - based sia per il traffico connection-less. L'Onion Routing si interfaccia con tutte le piattaforme ed i sistemi tramite proxy specializzati, che lo rendono facilmente integrabile nei sistemi esistenti.

Comunicazione anonima

In una comunicazione si definisce:

- Accesso *anonimo in avanti*: Alice accede al servizio senza che nessuno possa risalire all'indirizzo IP da cui ella ha generato la richiesta.
- Accesso *anonimo all'indietro*: Alice accede al servizio senza che nessuno possa risalire all'indirizzo IP del server che ospita il servizio stesso.
- Accesso *riservato*: Nessuno, tranne Alice e Bob, può conoscere il contenuto dei dati scambiati.

1.2. La riservatezza passa per la crittografia

La riservatezza della comunicazione viene garantita attraverso:

1. **sistemi crittografici a chiave pubblica (o asimmetrica)**. Esempio: Alice possiede due chiavi, una segreta K_S^A ed una pubblica K_P^A ,che ella condivide con Bob, ogni messaggio che Bob cifra con K_P^A può essere decifrato solo impiegando K_S^A, quindi solo Alice può decifrarlo. Non è possibile risalire da K_P^A a K_S^A in tempi ragionevoli.
2. **Sistemi crittografici a chiave simmetrica**. Esempio: Alice e Bob condividono un segreto K, usato da entrambi come chiave per cifrare e decifrare messaggi.

Gli algoritmi di ogni buon sistema crittografico devono essere pubblici, la riservatezza deve essere affidata alle sole chiavi.

1.3. Classificazione dei sistemi di comunicazione anonima

I Sistemi di comunicazione anonima si dividono in due grandi categorie che si differenziano in base alla latenza della comunicazione.

Essi si distinguono in:

1. **Sistemi anonimi ad alta latenza**

   1.1. Anonymous remailer

2. **Sistemi anonimi a bassa latenza**

   2.1. The Onion Routing

   2.2. Anonymizer

   2.3. Java Anon Proxy

   2.4. FreeNet

Sistemi anonimi ad alta latenza

1.1 Per **anonymous remailer** si intende un particolare tipo di mail-server il cui compito è ricevere ed inoltrare un messaggio di posta elettronica in modo che, partendo dal messaggio giunto al destinatario finale, sia impossibile risalire al mittente originario.

Fondamentalmente un anonymous remailer inoltra il messaggio ad una qualunque destinazione richiesta dal mittente, soltanto dopo aver effettuato una rimozione e sostituzione degli header che possono identificare il mittente stesso del messaggio.

Le diverse tipologie di anonymous remailer sono:

Tipo 0 - Anonymous remailer pseudoanonimi:

Gli anonymous remailer di tipo 0, detti *pseudoanonimi*, hanno ormai un interesse più che altro storico.

Questo tipo di remailer (chiamato anche "penet" dal più famoso di essi, *anon.penet.fi* , da tempo inattivo), forniva all'utente un account al quale veniva assegnato casualmente uno pseudonimo di identificazione.

Il remailer memorizzava in un database la coppia "pseudonimo - indirizzo di posta elettronica" dell'utente. Nella posta in uscita gli *header* di identificazione del messaggio venivano sostituiti con lo pseudonimo; in questo modo era possibile per il destinatario finale del messaggio rispondere ad esso in modo facile e diretto utilizzando la funzione *reply-to* del proprio programma di posta elettronica. La posta in entrata destinata a uno pseudonimo veniva inoltrata al vero indirizzo e-mail corrispondente.

Purtroppo, memorizzare le corrispondenze nomi-pseudonimi costituiva una notevole debolezza; infatti, l'operatore del sistema ed i suoi assistenti avevano accesso ai veri indirizzi degli utenti, la cui riservatezza dipendeva, quindi, dalla loro capacità e volontà di mantenerli segreti. Inoltre, la presenza del database rendeva anche il sistema particolarmente vulnerabile ad hackeraggi ed intrusioni, teoricamente sempre possibili.

Tipo I - Cypherpunk: Questo tipo di remailer è più sicuro del precedente, poiché non esiste un database degli utenti e gli operatori dichiarano di evitare con la massima cura di tenere *log* che potrebbero permettere di identificare gli utenti.

Inoltre un anonymous remailer Cypherpunk dispone della propria chiave pubblica PGP[1], reperibile spedendo una e-mail con subject "remailer-key" al server che si desidera utilizzare. Il remailer è in grado di accettare posta cifrata mediante questa chiave, di decifrare la posta ricevuta e di inoltrarla all'indirizzo richiesto.

Il sistema offre la possibilità di concatenare diversi remailer in una successione (*chaining*), in cui ciascun remailer conosce soltanto il punto da cui arriva il messaggio e la destinazione di re-invio. Tale sistema fornisce funzioni come il riordino casuale dei messaggi (*reordering*) e comandi per richiedere un ritardo nella trasmissione degli stessi (*latent time*) così da evitare la possibilità di correlare i messaggi in uscita a quelli in entrata basandosi sull'ora di arrivo e di partenza.

Inoltre, è possibile costruire un indirizzo anonimo *(reply-block*) che permette di ricevere risposte senza perdere l'anonimato. Per indirizzo anonimo si intende un recapito presso il quale possiamo ricevere posta, tenendo nascosto il nostro nome o il nostro vero indirizzo di posta elettronica a coloro che ci rispondono.

E' possibile, quindi, utilizzare gli anonymous remailer Cypherpunk per preparare un reply-block, cioè un file cifrato con la chiave pubblica di uno o più remailer, contenente una richiesta di re-invio, attraverso una catena di remailer, al nostro vero indirizzo di posta elettronica.

I remailer Cypherpunk potrebbero risultare vulnerabili ad un attacco basato sull'analisi della dimensione dei messaggi. Infatti, anche se il nostro messaggio è correttamente cifrato, riordinato dagli anonymous remailer della catena e ritardato su ognuno di essi, comunque il messaggio spedito in partenza è

[1] Pretty Good Privacy (PGP) è un programma che permette di usare autenticazione e privacy crittografica. Nelle sue varie versioni è probabilmente il crittosistema più usato al mondo. In Applied Cryptography, il crittografo Bruce Schneier lo ha descritto come il modo per arrivare "probabilmente il più vicino alla crittografia di livello militare"PGP è stato originariamente sviluppato da Phil Zimmermann nel 1991. Il nome gli è stato suggerito da una drogheria di Lake Wobegon, l'immaginaria città natale dello speaker radio Garrison Keillor. La drogheria si chiamava "Ralph's Pretty Good Grocery" ("la drogheria piuttosto buona di Ralph") e il suo slogan era "se non lo puoi trovare da Ralph, probabilmente puoi anche farne a meno".

costituito da un numero preciso di byte, difficilmente identico a quello di un altro, che diminuirà a ogni successivo passaggio della catena di remailer in modo prevedibile.

Un attaccante che dispone delle risorse opportune e di una notevole capacità di calcolo potrebbe tentare di identificarlo e tracciarlo basandosi proprio su questa caratteristica.

Un'altra possibilità di attacco potrebbe basarsi sulla tecnica del cosiddetto *reply attack* in cui l'attaccante registra il nostro messaggio nel momento in cui lascia la nostra macchina e ne invia un gran numero di copie al primo remailer della catena. Un gran numero di messaggi identici sarà così re-inviato al remailer successivo, identificabile mediante l'analisi del volume del traffico scambiato tra i due remailer. In questo modo sarebbe possibile seguire il percorso del nostro messaggio sino al destinatario finale.

Tipo II – Mixmaster: Il Mixmaster è il tipo più recente e sicuro di remailer operativo su Internet e rappresenta lo stato dell'arte nel campo dell'anonymous remailing. Concatenamento e cifratura sono predisposti in modo automatico al momento della preparazione del messaggio con un apposito client. Un remailer Mixmaster non usa la cifratura PGP, ma il pacchetto RSAREF[2] e alcuni formati proprietari in modo da utilizzare un processo di cifratura\decifratura estremamente più complesso di quello degli anonymous remailer Cypherpunk.

In pratica, un remailer Mixmaster scompone i messaggi, li sottopone a cifratura multipla e li incapsula in uno o più pacchetti di dati di uguale dimensione, rendendo così impossibile un'analisi efficace basata su questa caratteristica.

I pacchetti vengono inviati separatamente lungo la rete dei remailer e sottoposti a un sofisticato *reordering*. Ogni pacchetto è interamente cifrato ad ogni re-invio con una chiave 3DES, in modo che nessuna informazione sia visibile ad

[2] http://www.csm.ornl.gov/~dunigan/rsaref.txt

un osservatore esterno. Come per gli anonymous remailer Cypherpunk, ogni remailer della catena, anche se è stato compromesso, può conoscere solo il punto da cui arriva un pacchetto e la destinazione di re-invio. Tuttavia, solo l'ultimo remailer della catena è in grado di vedere quali pacchetti compongono un singolo messaggio. Per tutti gli altri remailer i pacchetti sono completamente indipendenti tra loro.

A ogni pacchetto di dati in transito viene casualmente assegnato un numero di identificazione, che il remailer conserva in memoria.

Il sistema è così in grado di riconoscere un pacchetto già transitato e rifiutarne il re-invio, proteggendosi da un possibile "reply attack", basato sulla cattura e ritrasmissione dei pacchetti. Infine, come ulteriore protezione, l'architettura di un remailer Mixmaster prevede un traffico costante "di copertura" generato casualmente, in modo da nascondere la trasmissione dei pacchetti veri e propri.

I remailer Mixmaster, a differenza degli anonymous remailer di tipo Cypherpunk, necessitano di un apposito client per la preparazione dei messaggi.

Sistemi anonimi a bassa latenza

2.1 Tra i sistemi anonimi **a bassa latenza** abbiamo identificato "**The Onion Routing".** Tale sistema mira a rendere non tracciabile la fruizione di servizi di rete generici (WWW, IM, ecc.) in modo trasparente all'utente e a garantire latenze tollerabili. Questo tipo di tecnologia anonima è resistente alle comuni tecniche di analisi del traffico ed effettua, attraverso circuiti virtuali, tunneling di stream TCP, l'incapsulamento dei pacchetti TCP in strutture dati *(onion)* ripetutamente cifrate e l'instradamento delle onion attraverso nodi successivi.

2.2 **Anonymizer**

Sistema a single-hop proxy, in cui un proxy elimina le informazioni sul mittente della richiesta prima di inviarle sulla rete. La struttura di questi sistemi è molto semplice da analizzare, ma gli utenti devono credere nell'affidabilità del proxy che anonimizza la comunicazione. Concentrare il traffico in un singolo punto incrementa l'insieme anonimo (l'insieme di persone in cui un dato utente viene nascosto), ma è vulnerabile ad un attacco proveniente da un avversario che può osservare tutto il traffico in uscita e in entrata da un proxy.

2.3 **Java Anon Proxy (JAP)**

JAP[3] 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:

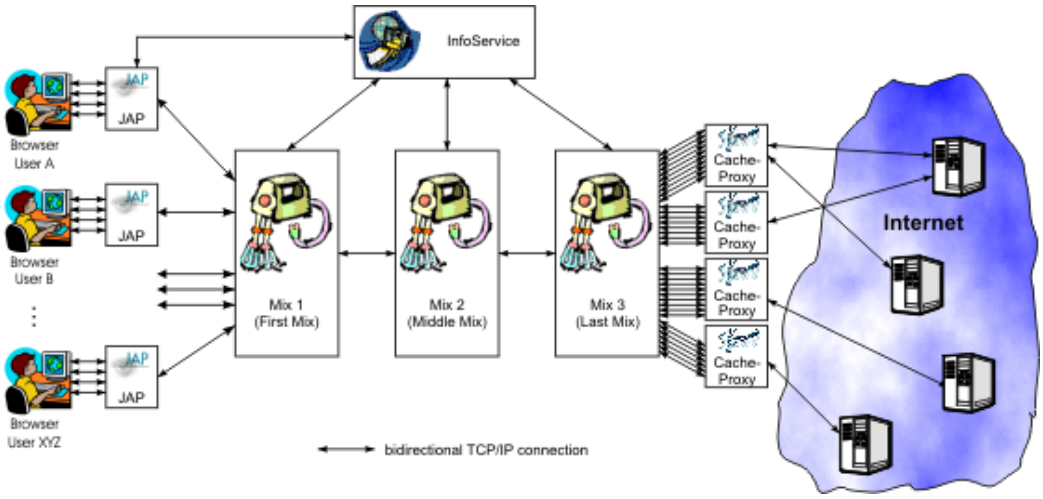
- un programma client che deve essere installato sulla macchina dell'utente detto *JAP client*;
- un *info service* che fornisce meta-informazioni sulle mix-cascade disponibili, numero di utenti che usano attualmente le mix-cascade carico corrente sul mix e chiavi pubbliche di ciascun mix;

[3] http://anon.inf.tu-dresden.de

- alcuni *cache proxy* che ricevono i dati dai mix e li inviano attraverso la rete Internet.

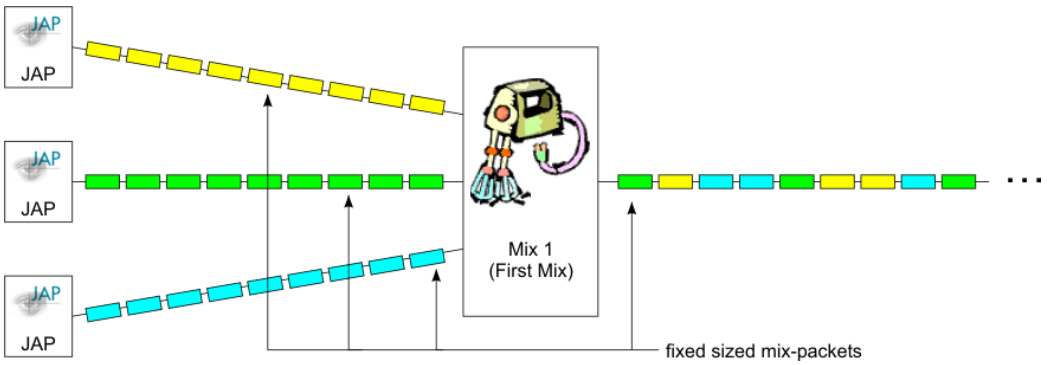
Gli utenti, mediante il programma client, possono scegliere una sequenza predeterminata di mix, detta *mix-cascade* all'interno della quale ciascun mix conosce soltanto quello precedente e quello successivo. La cascata scelta verrà utilizzata per spedire dati sino al log-off dell'utente.

Le connessioni di un singolo utente sono nascoste attraverso le connessioni di tutti gli altri utenti, poiché ogni utente può utilizzare le stesse mix-cascade.



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

Il primo mix di una cascata può ricevere dati da più utenti e perciò è in grado di multiplexare/demultiplexare il canale e spedire i dati ricevuti al mix successivo in modo serializzato all'interno di celle di taglia fissata come nella figura seguente.



JAP utilizza cifratura ibrida (RSA/AES-128)[4] per i messaggi scambiati tra i computer degli utenti ed i mix della cascata.

Dopo che l'utente ha selezionato una cascata, il JAP client si connette all' info service e scarica le chiavi pubbliche di tutti i mix della cascata.

Con tali chiavi verranno cifrati i pacchetti scambiati tra la cascata ed il client.

In tal modo saranno concordate, con ogni mix, le chiavi simmetriche per cifrare i dati spediti dal JAP client, attraverso la cascata, verso Internet.

Successivamente il JAP client cifrerà iterativamente, con le chiavi simmetriche concordate con ciascun mix, il dato da spedire partendo dall'ultimo mix della catena sino al primo. Sarà così costruita una struttura dati simile ad una cipolla.

Il primo mix decifrerà il primo strato con la chiave simmetrica concordata e lo inoltrerà al successivo mix che ne decifrerà lo strato seguente e così via sino all'ultimo mix della cascata. Sarà quest'ultimo mix a poter accedere al dato vero e proprio e ad inoltrarlo ad un cache proxy che, infine, lo invierà su Internet.

Le risposte provenienti dalla rete Internet seguono il processo in ordine inverso.

[4] Vedi:

1- http://anon.inf.tu-dresden.de/develop/doc/mix_short/index.html#docCascadeInit

2- http://anon.inf.tu-dresden.de/desc/encr_jap_en.html

2.4 **FreeNet**

Freenet è un sistema P2P per lo storage distribuito e il prelievo di informazioni progettato per risolvere i problemi legati alla privacy degli utenti e alla sopravvivenza dei dati. Tale sistema viene descritto nella pubblicazione “A distributed decentralized information storage and retrieval system”, redatto da Ian Clarke nel Luglio 1999, quando si trovava come studente presso l’Università di Edinburgh in Scozia. Freenet è un Software libero rilasciato sotto licenza GPL e scritto in linguaggio Java, pubblicamente distribuito per la prima volta in versione 0.1 nel Marzo 2000. Il sistema realizza un filesystem location-independent distribuito attraverso molti nodi paritetici che donano le proprie risorse hardware per permettere di inserire, memorizzare e prelevare in modo anonimo i file, ma senza garantire che i dati rimangano permanentemente nello storage.

Freenet è formato da un insieme di nodi che scambiano tra di loro messaggi. Un nodo è semplicemente un calcolatore che sta eseguendo il software Freenet e tutti i nodi della rete sono identici e trattati allo stesso modo. Ogni nodo mette a disposizione una certa quantità di spazio di memorizzazione per mantenere una propria porzione locale del data store distribuito. Gli ideatori di Freenet desiderano offrire a qualsiasi autore la possibilità di pubblicare i propri documenti a prescindere dalle sue disponibilità finanziarie e di spazio libero su disco e per questo motivo il sistema non richiede né di condividere una soglia minima di memoria di massa, né alcun pagamento per fruire del servizio. Questo giustifica il fatto che Freenet non garantisce che i documenti siano presenti in modo permanente nello storage ed implementa, invece, una politica probabilistica per lo scarto dei file meno richiesti, quando è necessario liberare spazio per l’inserimento di nuovi dati.

I nodi di Freenet seguono un protocollo packet-oriented, detto Freenet Network Protocol (FNP), per organizzarsi spontaneamente in una configurazione di rete efficiente.

In Freenet, pertanto, i nodi stabiliscono automaticamente connessioni gli uni con gli altri: una rete di questo tipo è definita Opennet. Gli indirizzi dei nodi sono composti da indirizzo IP e un numero di porta. Per aggiungere un nuovo file alla rete l'utente invia un messaggio di inserimento contenente il file ed il suo identificatore. Questo identificatore si chiama GUID-key (Globally Unique IDentifier). Il file verrà memorizzato su un insieme di nodi il cui numero viene indicato dall'utente mediante il software Freenet. Durante tutto il tempo in cui il file resterà sulla rete (ovvero fino a quando non verrà automaticamente cancellato perché nessuno lo richiede più), esso potrà migrare su un altro insieme di nodi oppure essere duplicato. Per scaricare quel file un altro utente dovrà inviare alla rete un messaggio di download contenente la GUID-key del file. Quando la richiesta raggiunge uno dei nodi in cui è memorizzato quel file, i dati vengono inviati indietro lungo la catena verso colui che aveva inoltrato la richiesta. Le GUID – key vengono calcolate usando la funzione di hash SHA1.

Tali GUID-key[5] si dividono in tre tipologie: la keyword-signed key (KSK), la content-hash key (CHK), usata per la memorizzazione dei dati, e la signed-subspace key (SSK) pensata per un uso di più alto livello.

- **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

[5] Per la trattazione specifica relativa alle GUID-key consultare i seguenti link:
1) http://www.s0ftpj.org/docs/BFi14-dev-06.pdf
2) http://www.ecse.rpi.edu/Homepages/shivkuma/teaching/sp2001/readings/freenet.pdf

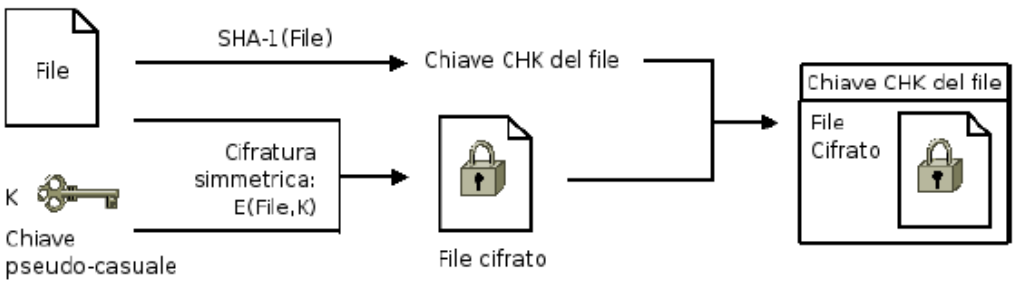
che serviranno da identificativo e una breve descrizione del file. La chiave pubblica e la stringa descrittiva verranno utilizzate per generare la chiave SSK, mentre la chiave privata firmerà il documento.

Quindi per quanto riguarda le SSK possiamo dire che:

1) Per scaricare un file occorre avere la chiave pubblica e la stringa descrittiva attribuita al file. Con queste due informazioni è possibile ricreare la chiave SSK.

2) Per aggiornare un file occorre anche la chiave privata per generare una firma valida. Questo assicura che solo il proprietario di quel file possa modificarlo successivamente al rilascio sulla rete.

La GUID-key più comune è la *Content-Hash Key* (CHK) utilizzata per la memorizzazione dei dati e viene generata calcolando un codice hash in base ai contenuti del file. Questo processo fornisce ad ogni file un identificatore unico. Inoltre, il file viene cifrato simmetricamente con una chiave generata in modalità casuale. Le CHK riducono la ridondanza delle informazioni perché dati uguali avranno una medesima CHK e al momento dell'inserimento si riscontrerà che il documento è già presente in rete.



Affinché gli altri utenti possano prelevare e visualizzare il documento, chi inserisce il file deve pubblicare insieme alla sua CHK e alla descrizione del file, anche la chiave simmetrica per la decifratura. Tali chiavi possono essere pubblicizzate inviando un messaggio all'autore di uno dei siti già presenti in FreeNet. Molti dei portali i cui collegamenti si trovano sull'interfaccia web di FreeNet (fproxy) hanno uno spazio in cui è possibile inviare messaggi anonimamente. Le chiavi possono essere inviate ad altre persone anche usando la mailing list di Freenet o attraverso il canale *irc irc.freenode.net#freenet*[6].

In FreeNet i messaggi, anziché viaggiare direttamente dal mittente al destinatario, attraversano una sequenza di nodi. Poiché ogni nodo della catena conosce soltanto i suoi vicini, il nodo mittente e il nodo destinatario potrebbero trovarsi ovunque tra migliaia di nodi.

Ogni nodo mantiene un tabella di instradamento (routing table), che elenca gli indirizzi degli altri nodi, le GUID-key e gli eventuali dati cifrati.

Chiave	Dato	Indirizzo
8e0109xb87wkhkujhs98k	99usbkjhgd7333khjgs763	tcp/5.34.27.4:6473
uushs89763kjhx7w732722	yy6254231gsyw4GGcwhgs	tcp/89.34.36.1:24855
kjhks872228x0982876jjhd	TTRos384hgygduybv1111n	tcp/194.44.62.66:9897
878772kx762776xbv8622		tcp/64.28.67.48:43653
222764kjhx8f63wkbkjs77w		tcp/4.18.49.35:65466
57765xkjhd72729jnbck01kj		tcp/55.18.4.1:3895

Quando un nodo riceve una richiesta, analizza per prima cosa il suo spazio disco per controllare se il file è presente. In tal caso, lo re-invia al nodo dal quale ha ricevuto la richiesta, affiancandovi un tag che lo identifica come possessore di quel file.

In caso contrario, il nodo invia la richiesta ai nodi presenti nella sua tabella aventi la chiave più "vicina" (in base alla distanza lessicografica) a quella cercata.

[6] Vedi http://freenetproject.org/faq.html#find

Se la richiesta ha successo, ogni nodo nella catena spedisce il file indietro verso il mittente. Inoltre, ogni nodo potrebbe conservare sul proprio spazio-disco una copia del file richiesto per rendere più veloce le richieste successive verso quel file. Infine, per nascondere l'identità del possessore dei dati e, quindi, mantenerne l'anonimato, i nodi occasionalmente alterano i messaggi di risposta, fingendo di esserne i reali possessori.

2. Tor. Panoramica

Tor (The Onion Routing) è un progetto open-source sviluppato da "The Free Haven Project". Esso è un servizio di comunicazione circuit-based anonimo a bassa latenza, che consiste in un overlay network progettata per anonimizzare applicazioni TCP-based.

Alcune sue caratteristiche sono:

- **segretezza nel forward dei messaggi**

  Tor usa un design di creazione del cammino incrementale, o meglio *"telescopico"*, dove l'iniziatore negozia una chiave di sessione con ogni hop successivo nel circuito. Una volta che queste chiavi sono cancellate, nodi sequenzialmente compromessi non possono decifrare il vecchio traffico. In questo modo il processo di creazione del circuito è più affidabile, in quanto possiamo sapere quando e se un hop fallisce e possiamo cercare di estendere il circuito con un'altro nodo.

- **controllo della congestione e directory server:**

  Tor effettua controlli di congestione usando acknowledgment end-to-end per mantenere l'anonimato, e allo stesso tempo , permette ai nodi alla fine del network di rilevare le congestioni o i flooding.

  Inoltre, alcuni nodi agiscono da directory server che hanno il compito di fornire informazioni fidate sui nodi della rete Tor ed il loro stato corrente.

- **controllo di integrità**

  Per evitare che un nodo possa cambiare il contenuto dei dati di una cella, per esempio alterare una richiesta di connessione o “taggare” traffico cifrato per poi analizzare il traffico modificato in uscita dal network, Tor verifica l'integrità dei dati prima che loro abbandonino il circuito.

- **Topologia del circuito “leaky pipe”**

  Grazie a segnali “in-band” all'interno del circuito, gli utenti possono dirigere il traffico su nodi diversi lungo il circuito. Questo approccio permette al traffico di uscire dal circuito da qualsiasi parte , in maniera tale da far divenire veramente difficoltosa l'analisi del traffico e i “volume attacks” basati sull'osservazione del traffico alla fine del circuito.

Tor è una rete di tunnel virtuali che permette alle persone ed alle organizzazioni di aumentare la propria privacy e sicurezza su Internet. L’idea alla base dell'onion routing è quella di proteggere non solo la privacy del mittente e del ricevente del messaggio, ma anche il contenuto del messaggio stesso quando attraversa la rete.

Inoltre, l’onion routing consente agli sviluppatori di software di creare nuovi strumenti di comunicazione con caratteristiche intrinseche di privacy e fornisce le basi per una gamma di applicazioni con cui i singoli individui e le organizzazioni possono condividere informazioni sulla rete pubblica senza compromettere la propria privacy. L'onion routing, pertanto, risponde al principio delle *chaum's mix*[7] *cascades*: i messaggi viaggiano dalla sorgente alla destinazione attraverso una sequenza di proxy *("onion routers")* che

[7]Un mix è un dispositivo di memorizzazione e di inoltro che accetta un numero di messaggi di lunghezza fissata da numerose sorgenti, attuando trasformazioni crittografiche sui messaggi ,epoi inoltrarli alla successiva destinazione in ordine casuale. Un singolo Mix tiene traccia di un particolare messaggio o attraverso una specifica bit-pattern ,o mediante la taglia, o ordinando rispetto ad altri messaggi

reindirizzano i messaggi in un path non predicibile. L'onion routing non fornisce una perfetta anonimia del mittente o del ricevente contro tutti i possibili intrusi: è possibile per un intruso osservare che un individuo ha spedito o ricevuto un messaggio. L'onion routing possiede un forte grado di *unlinkability*: un intruso non può determinare facilmente sia il mittente che il ricevente di un determinato messaggio.

Anche entro questi limiti, l'onion routing non fornisce alcuna garanzia di privacy assoluta; al contrario determina un continuum in cui il grado di privacy è generalmente una funzione del numero di routers partecipanti contro il numero di routers compromessi o maliziosi.

2.1. 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".

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.

2.2. Cui prodest ?

2.2.1. Le persone normali

Le persone normali che vogliono proteggere la loro privacy dal marketing senza scrupoli e dal furto di identità. Gli Internet Service Provider (ISP) vendono il registro delle navigazioni in Internet a società di marketing o a chiunque sia disposto a pagare. Gli ISP di solito dicono di anonimizzare i dati e di non fornire informazioni in grado di identificare le persone, ma non è così. Questi dati possono ancora contenere il registro completo di ogni sito visitato, il testo di ogni ricerca fatta e potenzialmente anche la username e la password.

Oltre all’ISP, i siti web (ed i motori di ricerca) che si visitano hanno i loro registri (log), contenenti le informazioni stesse o anche di più. Su Internet Tor rappresenta uno degli strumenti per chi oggi si preoccupa per la propria privacy di fronte a crescenti intrusioni e a perdite di dati personali. Infatti, i dati spesso non sono ben protetti da coloro a cui sono stati affidati e può accadere di dare informazioni in grado di identificare online l’identità dell’utente, semplicemente fornendo il proprio l'indirizzo IP.

Sempre più spesso gli indirizzi IP si possono collegare con precisione a una città, a una certa via e possono rivelare altre informazioni su come ci si connette a Internet.

2.2.2. I militari

Agenti in missione: non è difficile per delle forze ostili sorvegliare il traffico Internet e scoprire gli hotel e le località da cui le persone si collegano a server militari conosciuti. Il personale militare in missione potrebbe utilizzare Tor per nascondere i siti che visita, per proteggere così gli interessi e le operazioni militari, oltre a proteggere fisicamente se stessi.

Hidden service: Quando la rete Internet fu progettata dalla DARPA, il suo scopo principale era permettere comunicazioni robuste e distribuite in caso di attacchi locali. Tuttavia alcune funzioni devono essere centralizzate come i siti di comando e controllo.

E' nella natura dei protocolli Internet rivelare la località geografica di ogni server raggiungibile online. Se utilizzati, gli hidden service (servizi nascosti) di Tor consentirebbero al comando militare di essere fisicamente al sicuro e di non essere scoperti e neutralizzati.

2.2.3. I giornalisti ed i loro lettori

Reporter senza frontiere: si occupa di obiettori di coscienza su Internet e di giornalisti incarcerati o minacciati in tutto il mondo. I giornalisti, fonti corrispondenti, blogger e dissidenti potrebbero avvalersi dell'uso di Tor per la propria sicurezza e riservatezza.

L'International Broadcasting Bureau degli Stati Uniti (Voice of America/Radio Free Europe/Radio Free Asia) sostiene lo sviluppo di Tor per aiutare gli utenti di Internet nei paesi senza un accesso sicuro ai mezzi di informazione liberi.

Tor potrebbe consentire a chi vive dietro firewall nazionali o sotto la sorveglianza di regimi repressivi, di avere una prospettiva globale su argomenti controversi, come la democrazia, l'economia e le religioni.

2.2.4. Polizia e magistratura

Sorveglianza online: Gli operatori di pubblica sicurezza potrebbero utilizzare Tor per visitare siti web e servizi sospetti senza lasciare tracce rivelatrici.

Operazioni sotto copertura: Analogamente l'anonimato permette alle forze di pubblica sicurezza di svolgere operazioni online “sotto copertura ”.

Per quanto sia buona la reputazione di un agente sotto copertura, se nelle sue comunicazioni vi sono indirizzi IP della polizia, la sua copertura salta.

Linee di denuncia davvero anonime: Le linee ed i servizi online per raccogliere denunce anonimamente sono poco utili senza software per l'anonimato. Le fonti più evolute sanno che, nonostante un nome o un indirizzo e-mail non sia collegato ad altre informazioni, i log del server possono identificarli rapidamente.

2.2.5. I dirigenti d'impresa

Banche dati sugli incidenti di sicurezza: Supponiamo che una istituzione finanziaria partecipi ad una banca dati collettiva di informazioni su attacchi Internet. Queste banche dati richiedono che i membri riferiscano ad un gruppo centrale le intrusioni subite, così che si possano correlare gli attacchi per scoprire schemi coordinati e per potere inviare segnalazioni. Se, però, una certa banca dati subisce un'intrusione, vorrà evitare che un attaccante che osserva il traffico verso la banca dati, possa scoprire da dove provengono queste informazioni.

Anche se ogni pacchetto fosse cifrato, l'indirizzo IP tradirebbe la posizione di un sistema compromesso. Tor potrebbe essere utile per permette a simili banche dati di informazioni sensibili di resistere contro questo tipo di attacchi.

Mantenere confidenziali le strategie aziendali: una banca d'investimenti, ad esempio, non vuole che degli osservatori nel settore possano capire quali sono i siti web letti dai suoi analisti. L'importanza strategica degli schemi di traffico e la vulnerabilità alla sorveglianza di questi dati stanno iniziando ad essere riconosciuti in molte aree del mondo degli affari.

2.2.6. I blogger

Si sente ogni giorno di blogger che vengono denunciati o licenziati per aver pubblicato online, nei propri blog, cose perfettamente legali.

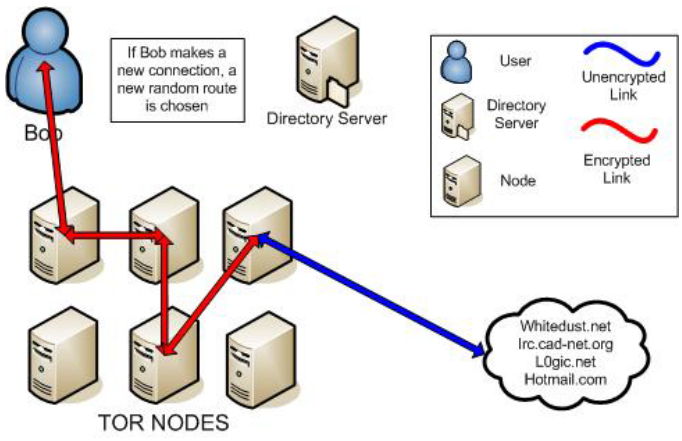
Queste persone potrebbero utilizzare uno strumento anonimo come Tor.

3. Tor: una rete anonima distribuita

3.1. La soluzione

Tor aiuta a ridurre i rischi derivati dall'analisi del traffico, sia semplice che sofisticata, distribuendo le transazioni attraverso molti nodi della rete Internet, in modo che nessun singolo punto possa collegare una transazione alla sua destinazione. L'idea è quella di costruire un percorso tortuoso e difficile per depistare un inseguitore, cancellando periodicamente le proprie orme. Infatti, anziché prendere un percorso diretto dalla sorgente alla destinazione, i pacchetti di dati nella rete Tor prendono un percorso casuale attraverso molti relay che ne coprono le tracce, in modo che nessun osservatore situato in un singolo punto possa dire da dove venga o dove sia diretto un certo traffico.

Per creare un percorso di rete privato con Tor, il software crea incrementalmente un circuito di connessioni cifrate attraverso i relay della rete Tor. Il circuito viene esteso un salto alla volta e ogni relay, lungo il percorso, 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 così da assicurarsi che ciascun nodo non possa tracciare queste connessioni durante il passaggio dei messaggi.

Una volta che un circuito è stato stabilito, si possono scambiare diversi tipi di dati e usare molti tipi di applicazioni attraverso una rete Tor.

Poiché ogni relay non vede altro che un singolo salto nel circuito, accade che né un intercettatore e neppure un relay compromesso possono utilizzare le tecniche di analisi del traffico per collegare la sorgente con la destinazione della connessione.

Inoltre, tale rete è tollerante agli errori e sotto il controllo di più domini amministrativi, in modo che un singolo onion router non può far cadere la rete o compromettere la privacy dell'utente. Tor funziona solo con i flussi TCP e può essere usato da ogni applicazione che abbia il supporto SOCKS. Per ragioni di efficienza Tor utilizza lo stesso circuito per le connessioni che avvengono nell'arco di dieci minuti. Le richieste successive vengono servite su un nuovo circuito, per evitare che nessuno possa collegare le azioni precedenti con le successive.

3.2. Tipi di nodi Tor

Client

In questa configurazione normale di base, Tor 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. Quando funge da *entrance node*, gestisce anche le connessioni dell'utente, garantendo un maggiore anonimato. Tutti i relay sono pubblicamente noti per scelta progettuale.

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 dalla rete Tor. Esso offre una protezione maggiore all'utente che lo usa per le proprie connessioni. Come tutti i relay Tor, esso è noto.

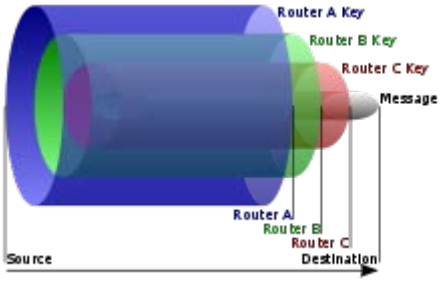
Bridge relay

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

3.3. Connessioni anonime in uscita

Poiché un client, per collegarsi ad un server, crea un circuito composto da nodi (macchine TOR) ognuno dei quali conosce soltanto quello precedente ed il successivo, è necessario specificare quali sono i passi per la creazione di una *connessione anonima in avanti*.

Le informazioni viaggiano in maniera cifrata, tranne nel tratto tra l'ultimo nodo Tor (detto exit node) e la destinazione finale che è normalmente in chiaro, all'interno di pacchetti (celle) contenenti vari strati incapsulati l'uno nell'altro così come avviene tra i livelli dello stack TCP/IP. Ogni nodo estrae l'involucro più esterno, operando sul payload il quale viene eventualmente inoltrato al nodo successivo. Ogni nodo (Onion Router - OR) mantiene una connessione TLS con altri OR. L'utente che vuole utilizzare la rete TOR deve avere a disposizione un apposito client, detto Onion Proxy (OP). Ogni cella ha una dimensione fissa di 512 byte e si compone di un identificatore di circuito (CIRCID), un comando (Command) che indica l'operazione da effettuare sul campo successivo (Data) contenente l'informazione vera e propria.



Per quanto riguarda le celle con comando Relay, il campo Data viene a sua volta suddiviso in vari campi, i primi dei quali forniscono ulteriori informazioni al ricevente sulle operazioni da effettuare sul payload in esso presente.

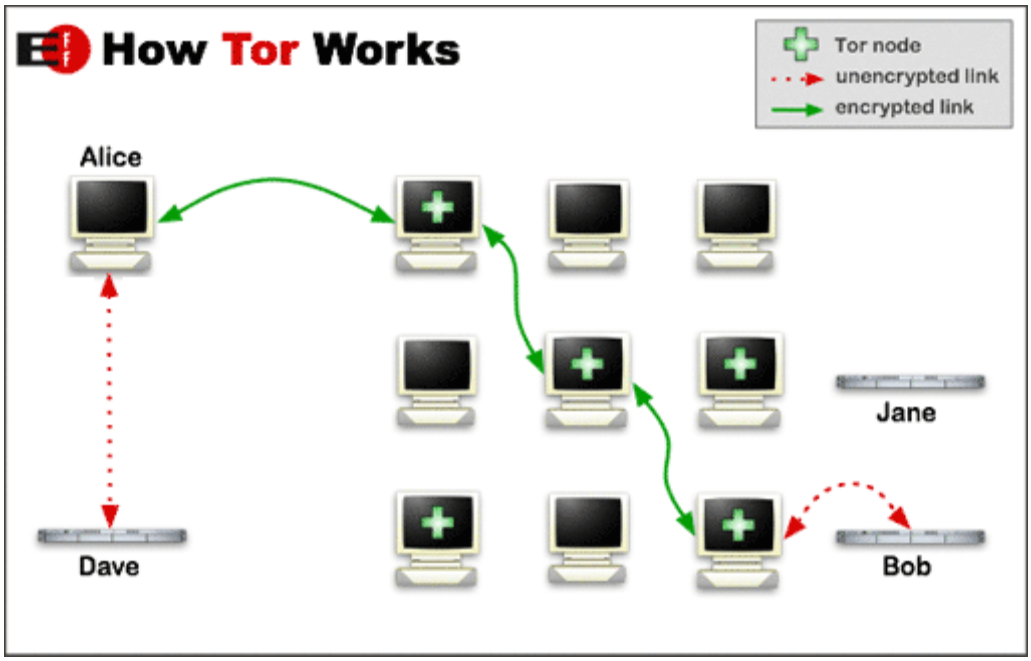
3.4. Tor: La tecnologia

Il Circuito Tor:

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[8].

Un circuito è generalmente un percorso di tre OR (ma questo numero è configurabile) sulla rete Tor dall'OP al server di destinazione desiderato.

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.



[8]"Clients choose a path through the network and build a circuit, in which each node (or "onion router" or "OR") in the path knows its predecessor and successor, but no other nodes in the circuit" - http://www.torproject.org/, R. Dingledine, N. Mathewson, P. Syverson.

Ogni utente esegue un software locale chiamato Onion Proxy (OP) che:

- Stabilisce i circuiti nella rete Tor.
- Gestisce le connessioni con l'applicazione utente.
- Effettua il recupero degli onion descriptor (un insieme di informazioni riguardanti le chiavi, gli indirizzi, la banda, le exit polices e altro…) dai directory server.
- Accetta Stream TCP e li multiplexa lungo il circuito.

Premessa

Durante la trattazione del design di TOR useremo queste abbreviazioni e convenzioni:

Tor utilizza uno "stream cipher", un cifrario a chiave pubblica, il protocollo di Diffie Hellman (D.H.) e una funzione Hash.

Per lo stream cipher, TOR utilizza AES a 128-bit in counter mode[9].

Per il cifrario a chiave pubblica, si usa RSA[10] con chiave a 1024 bit e un esponente fisso pari a 65537.

Per Diffie-Helmann, si usa un generatore (g) pari a 2, mentre per il modulo si usa un numero primo di 1024 bit.

Per la funzione hash viene utilizzato SHA-1.

Gli OR (Onion Router) comunicano l'uno con l'altro e con l'OP (Onion Proxy) dell'utente via TLS con ephimeral key.

Le chiavi pubbliche ed i descriptor degli OR che partecipano alla rete Tor vengono comunicati, attraverso connessioni TLS, alle autorità di directory che memorizzano ed approvano tali informazioni.

[9] Si veda http://it.wikipedia.org/wiki/AES
[10] Si veda http://it.wikipedia.org/wiki/RSA

Le chiavi pubblica/privata associate ad ogni Onion Router si distinguono in:

- **identity key a lungo termine**, utilizzate per firmare i certificati TLS, l'onion router descriptor e, in caso di directory server, per firmare le directory.
- **onion key a breve termine**, utilizzate per decifrare i messaggi inviati dall'utente per costruire un circuito e negoziare la *chiave effimera*.

Il protocollo **TLS** viene utilizzato per creare delle chiavi temporanee (link key) per cifrare la comunicazione tra gli OR della rete. Le chiavi di breve durata vengono cambiate periodicamente ed indipendentemente per limitare l'impatto di chiavi compromesse.

Celle e comandi

Il traffico Tor è costituito da celle di grandezza fissata. Ogni cella ha una taglia di 512 bytes, ed è formata da un header ed un payload.

L'header include un *circuit identifier* (CIRCID) che specifica a quale circuito la cella appartiene (più circuiti possono essere aperti sulla stessa connessione TLS), e un *command* che descrive cosa fare con il payload della cella.

Il CIRCID è specifico per la connessione, quindi ogni circuito ha un differente CIRCID per ogni OP/OR o OR/OR che la connessione attraversa.

Basandosi sul *command* presente nella cella, le celle si distinguono in:

- celle di controllo;
- relay cell;

Le prime vengono sempre interpretate dal nodo che le riceve, mentre le seconde trasportano i data stream end-to-end.

I comandi possibili che possiamo ritrovare per una cella di controllo sono:

- padding, utilizzato per mantenere in vita il circuito ma viene anche utilizzato per il link padding;
- CREATE e CREATED, usati per instaurare un circuito o segnalare che il circuito è stato creato correttamente;
- DESTROY, utilizzato per distruggere il circuito.

Le celle di relay possiedono un header aggiuntivo (relay header) posizionato all'inizio del payload, contenente:

- uno streamID, per identificare il singolo stream, dato che più stream possono essere multiplexati sullo stesso circuito;
- un end-to-end checksum per il controllo dell'integrità;
- la lunghezza del relay payload;
- un comando di relay.

L'intero contenuto del relay header e del relay cell payload vengono cifrati o decifrati, a seconda di come la cella si muove lungo il circuito, mediante cifrario AES con chiave a 128-bit in counter mode.

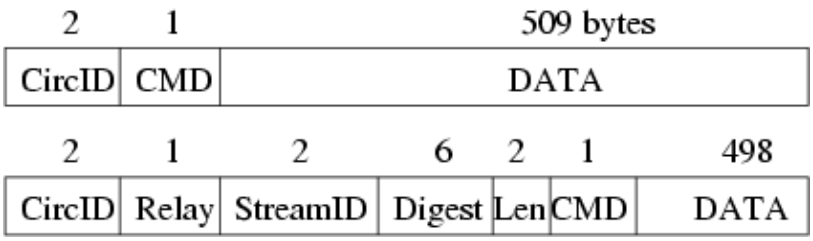
I possibili comandi presenti in una cella di relay sono:

- RELAY DATA, indica che il payload contiene dati dello stream aperto;
- RELAY BEGIN, utilizzato per aprire uno stream sul circuito;
- RELAY END, per chiudere uno stream;
- RELAY TEARDOWN, inviato quando si vuole chiudere uno stream interrotto bruscamente;

- RELAY CONNECTED, per notificare all'OP che il comando RELAY BEGIN è stato eseguito con successo;
- RELAY EXTEND e RELAY EXTENDED, utilizzati per estendere il circuito di un altro hop e confermare l'avvenuta estensione del circuito stesso;
- RELAY TRUNCATE e RELAY TRUNCATED,

  utilizzati per interrompere solo parte del circuito ma continuare a mantenerlo aperto;
- RELAY SENDME, utilizzato per il controllo della congestione.

Nella figura qui sotto viene data una rappresentazione visuale di quanto è stato detto finora riguardo la struttura delle celle.



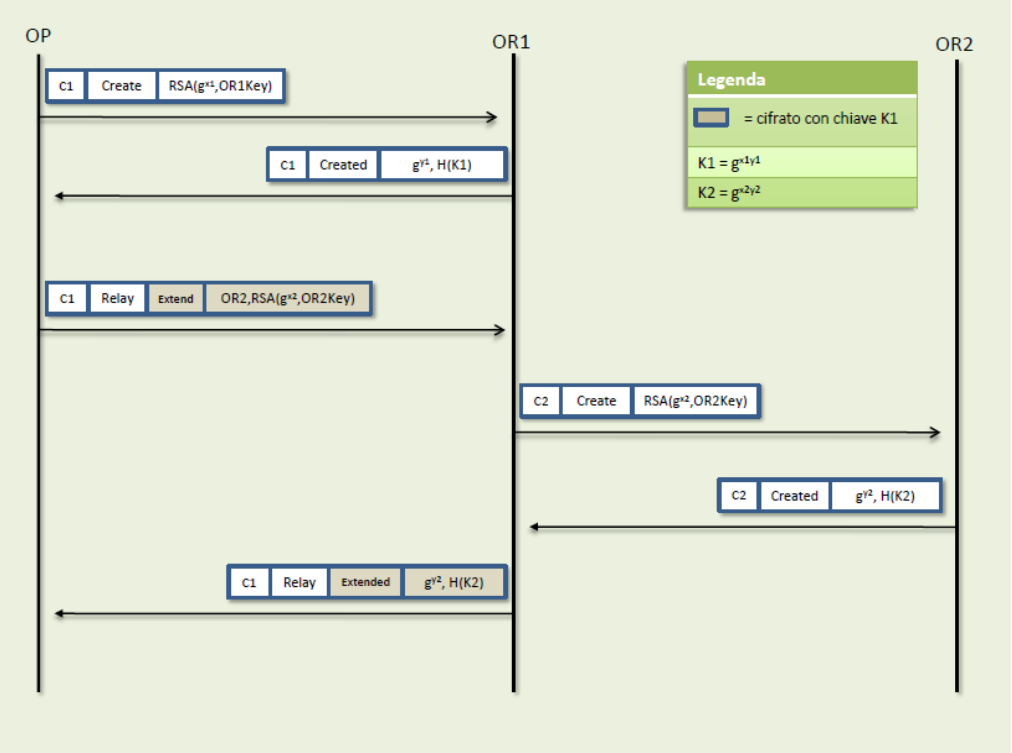
Creazione del circuito

Gli utenti creano il circuito un hop alla volta, negoziando una chiave simmetrica (K1,K2,…) con ogni OR del circuito.

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.



Creazione di un circuito

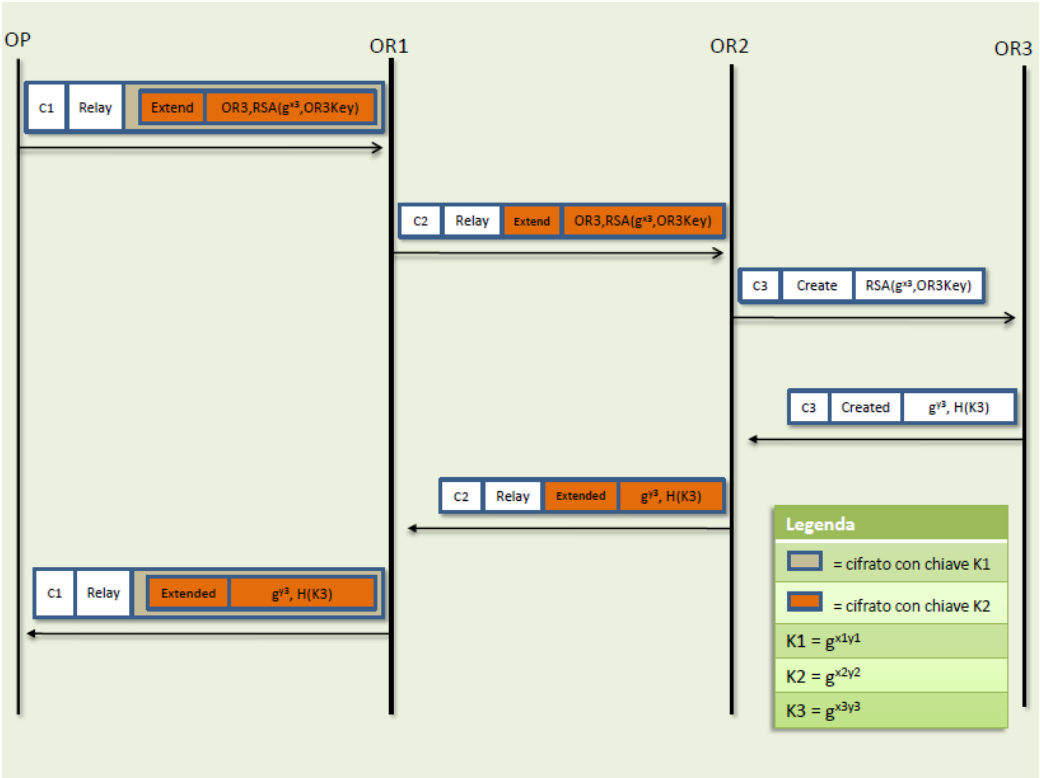
Per estendere il circuito:

- 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, $K2=g^{x2y2}$.

Per estendere ulteriormente il circuito:

Assumendo che siano state calcolate le chiavi di sessione K1 (condivisa con OR1) e K2 (condivisa con OR2), la figura qui sotto mostra l'estensione del circuito, precedentemente creato, verso OR3. Il procedimento è analogo a quello esposto in precedenza, con la differenza che questa volta, la cell – RELAY spedita dall'OP viene cifrata iterativamente, prima con la chiave K2 e poi con la chiave K1.

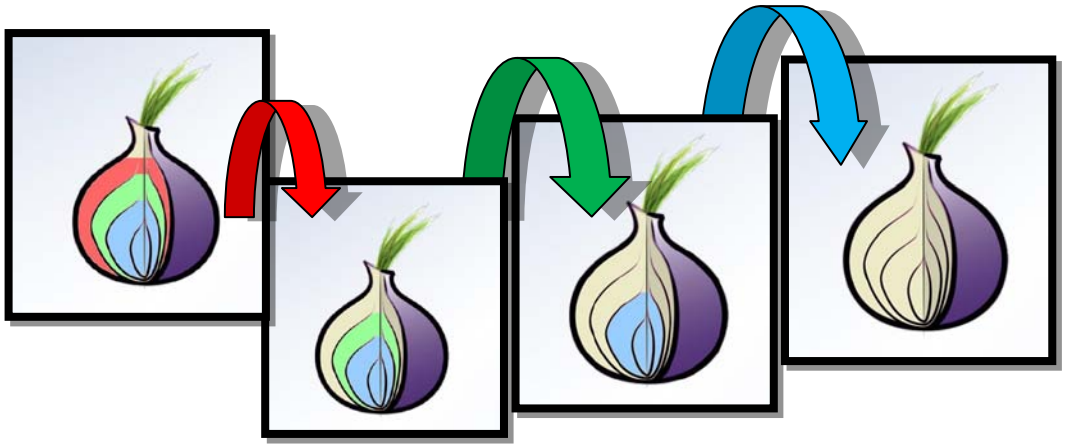
La cella viene inoltrata verso OR1, il quale ne decifra il primo strato mediante la propria chiave di sessione (K1) e la invia verso OR2 che, dopo aver decifrato lo strato successivo (mediante K2), infine, invia una cell-CREATE verso OR3 estendendo ulteriormente il circuito.



Dunque, tenendo presente la figura sopra, una volta che l'OP ha stabilito il circuito (quindi condivide una chiave con ogni OR del circuito), può mandare una cell – RELAY in questo modo:

- l'OP seleziona l'OR destinatario (OR2) e calcola il digest del campo *Data*;
- l'OP iterativamente cifra il RELAY header e payload con la chiave simmetrica di ogni hop del circuito dall'OR destinatario al più vicino;
- Ogni OR del circuito che riceve una cell - RELAY, controlla il CIRCID, decifra il RELAY header e il payload con la chiave di sessione negoziata per quel circuito e verifica il digest del campo *Data*;
- se il digest è valido significa che l'OR è l'esatto destinatario della cella e ne processa il contenuto, altrimenti inoltra la cella lungo il circuito verso il prossimo OR;

Gli strati di una cipolla



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

- l'OR mittente calcola il digest del campo *Data* e cifra l'header e il payload con la chiave che condivide con l'OP ;
- Ogni OR sul circuito cifra l'header e il payload della cella e manda il messaggio così formato indietro;
- l'OP quando riceve la cell-RELAY, iterativamente decifra ciascuno strato con la chiave relativa ad ogni OR del circuito, dal più vicino al più lontano;
- Ad ogni decifratura, l'OP verifica il digest e, se esso è corretto, è sicuro che la cella proviene dall'OR corrispondente all'ultima chiave utilizzata per la decifratura.

3.5. Hidden Services

3.5.1. Motivazioni

Anche se la sua funzionalità più diffusa è quella di fornire anonimità ai client, Tor può fornire anche anonimità ai server. Usando la rete Tor, è possibile ospitare dei server in modo che la loro localizzazione nella rete sia sconosciuta, questo perché:

1. un server costituisce tipicamente un single point of failure: se esso viene attaccato, isolato, rimosso o distrutto, contenuti e servizi non sono più disponibili.
2. 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.
3. la conoscenza della collocazione di un server e dell'insieme dei servizi da esso offerti aumenta notevolmente la probabilità di successo di un attacco.

Un servizio nascosto può essere ospitato da qualsiasi nodo della rete Tor, non importa che esso sia un relay o solo un client; per accedere ad un servizio nascosto, però, è necessario l'uso di Tor da parte del client. E', dunque, possibile offrire un servizio (ad esempio un sito web) in modo totalmente anonimo, come hidden service Tor.

Agli hidden services si accede attraverso uno pseudo-dominio di primo livello ".onion". La rete Tor comprende la richiesta e apre una connessione con il server desiderato.

3.5.2. 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

3.5.3. Funzionamento

Premessa:

Entità coinvolte: TOR garantisce la possibilità di fornire un hidden service introducendo due nuove entità all'interno del proprio sistema:

- **introduction point:** uno o più OR che il provider seleziona e pubblicizza come punti d'accesso al proprio servizio.

- **rendezvous point:** un OR selezionato dal client e utilizzato come punto d'incontro tra il provider ed il client stesso.

Affinché un provider possa pubblicizzare i propri introduction point e un client possa efficacemente venirne a conoscenza, è necessario l'utilizzo di un lookup system: la specifica Tor[11] suggerisce la scelta di un robusto ed efficiente servizio di tipo chiave-valore con aggiornamenti autenticati, come una hash table distribuita (DHT).

Tor garantisce al provider un ottimo livello di trasparenza nel deploy del proprio servizio, infatti:

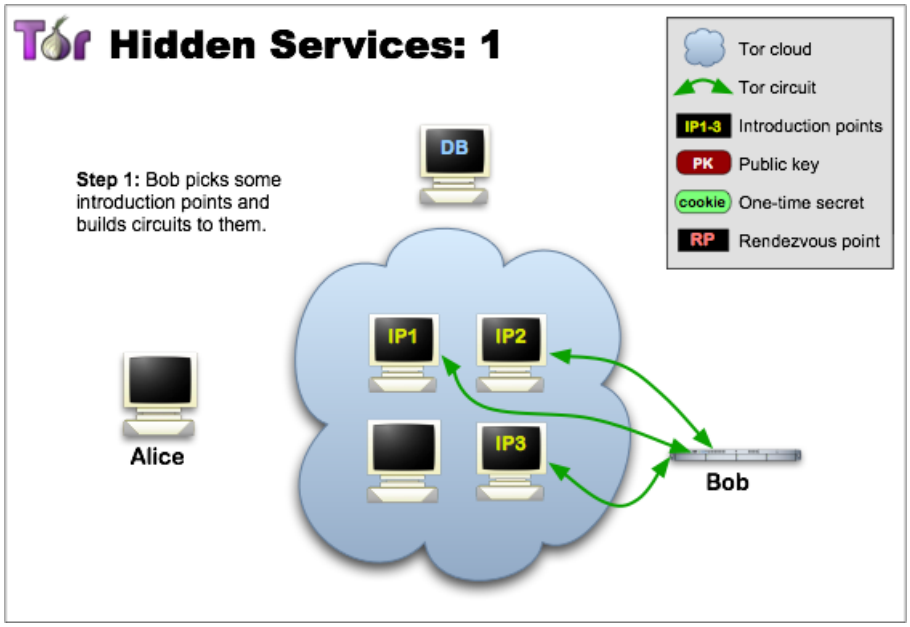
- Il provider dovrà generare una coppia di chiavi pubblica e privata a lungo termine, utilizzata per identificare il suo servizio.
- Il provider dovrà configurare il proprio onion proxy con l'indirizzo IP locale e la porta del servizio, una policy di autorizzazione per i client e la propria chiave pubblica. Inoltre indicherà gli introduction point selezionati.
- L'OP del provider provvederà a pubblicare in maniera anonima una dichiarazione firmata con la chiave pubblica, che indichi tra le altre cose l'expiration time del servizio e gli introduction point correnti presso il servizio di lookup. Quest'ultimo provvederà ad indicizzare il tutto con l'hash della chiave pubblica.
- L'OP del provider anonimamente costruirà un circuito Tor verso ognuno dei suoi introduction point e chiederà loro di aspettare eventuali richieste; inoltre, fornirà loro la chiave pubblica che identifica il suo servizio.
- Il client, una volta venuto a conoscenza del servizio, può ottenerne i dettagli eseguendo un'interrogazione al servizio di lookup: se il client è interessato ad accedere al servizio anonimamente, dovrà eseguire tale interrogazione attraverso Tor.

[11] Tor: The Second-Generation Onion Router – Roger Dingledine, Nick Mathewson, Paul Syverson. Official TOR Design Documentation.Pubblicata al 13th USENIX Security Symposium (San Diego, CA, USA; 9-13 August 2004). Reperibile presso <http://tor.eff.org/tor-design.pdf>

Costruzione di un Hidden Service in breve

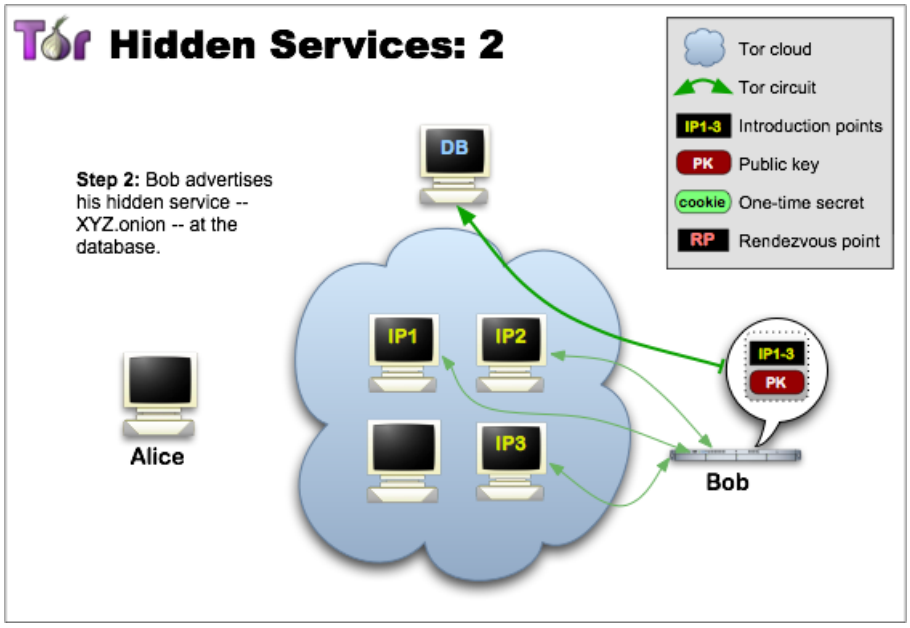
Primo passo:

Il provider sceglie alcuni relay a caso con i quali stabilisce dei circuiti Tor, rappresentati in figura dai link in verde, e chiede loro di fungere da *introduction point*, comunicando loro la sua chiave pubblica.



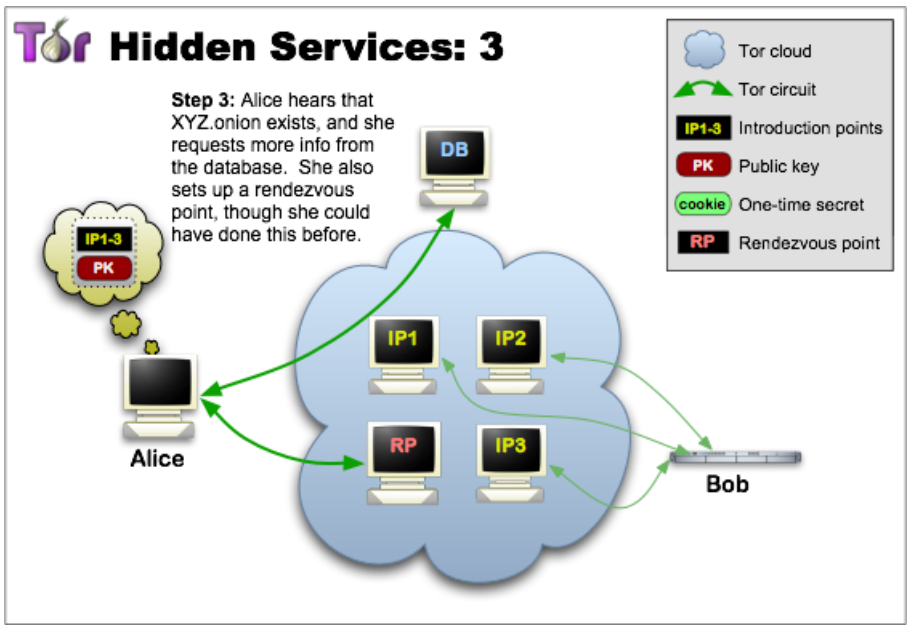
Secondo passo: l'hidden service costruisce un hidden service descriptor, contenente la sua chiave pubblica ed un sommario degli introduction point, e firma questo descriptor con la sua chiave privata. Invia il descriptor a un gruppo di directory server, usando sempre un circuito completo Tor per celare il collegamento tra il directory server contenente il descriptor e l'indirizzo IP dell'hidden server. Il descriptor verrà trovato dai client che richiederanno XYZ.onion, dove XYZ è un nome di 16 caratteri derivato in modo unico dalla chiave pubblica dell'hidden service.

Dopo questo passo, l'hidden service è attivo. Anche se usare un nome del servizio generato automaticamente sembra scomodo, ciò ha uno scopo importante: tutti – compresi gli introduction point, i directory server e, naturalmente, i client – possono verificare di comunicare con l'hidden service corretto.

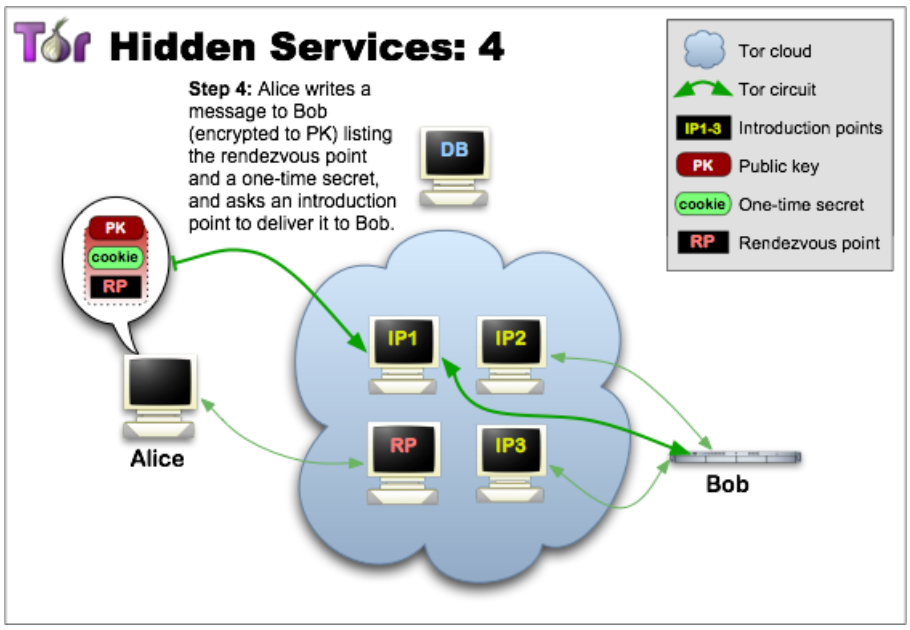


Terzo passo: il client, una volta venuto a conoscenza del servizio, deve iniziare a stabilire la connessione scaricandone il descrittore dai directory server.

Se esiste un descrittore per XYZ.onion, il client conosce il gruppo di introduction point e la corretta chiave pubblica. In questo momento il client crea anche un circuito verso un altro relay scelto a caso e gli chiede di fungere da rendez-vous point comunicandogli un "One-time secret" (rendez-vous cookie).

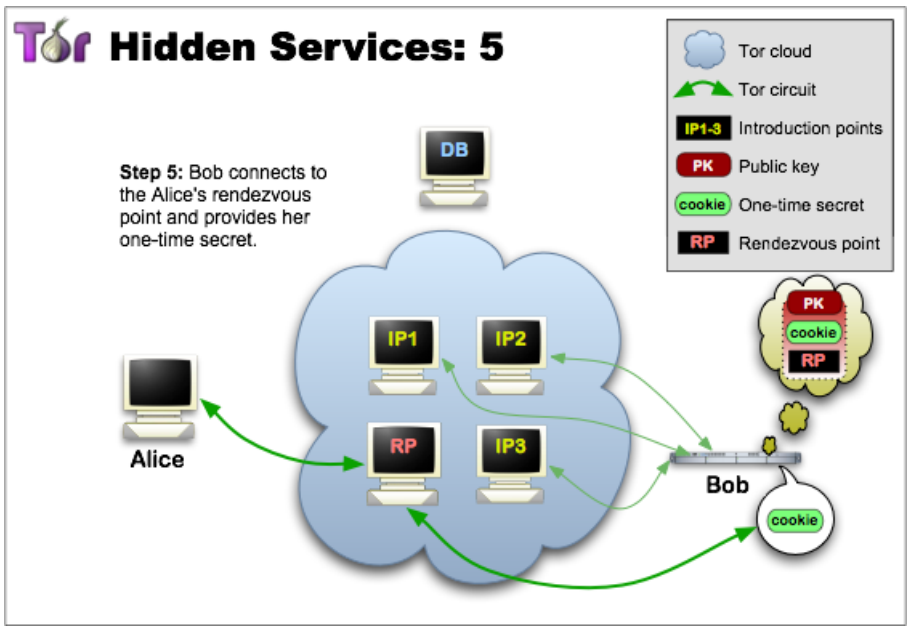


Quarto passaggio: una volta ottenuto il descriptor e pronto il rendez-vous point, il client costruisce un “introduce message” (cifrato con la chiave pubblica dell'hidden service) contenente l'indirizzo del rendez-vous point ed il “One-time secret”. Il client invia questo messaggio a uno degli introduction point, chiedendo che venga consegnato all'hidden service. La comunicazione avviene sempre tramite un circuito Tor: in questo modo nessuno può collegare l'invio dell'introduce message all'indirizzo IP del client che rimane così anonimo.

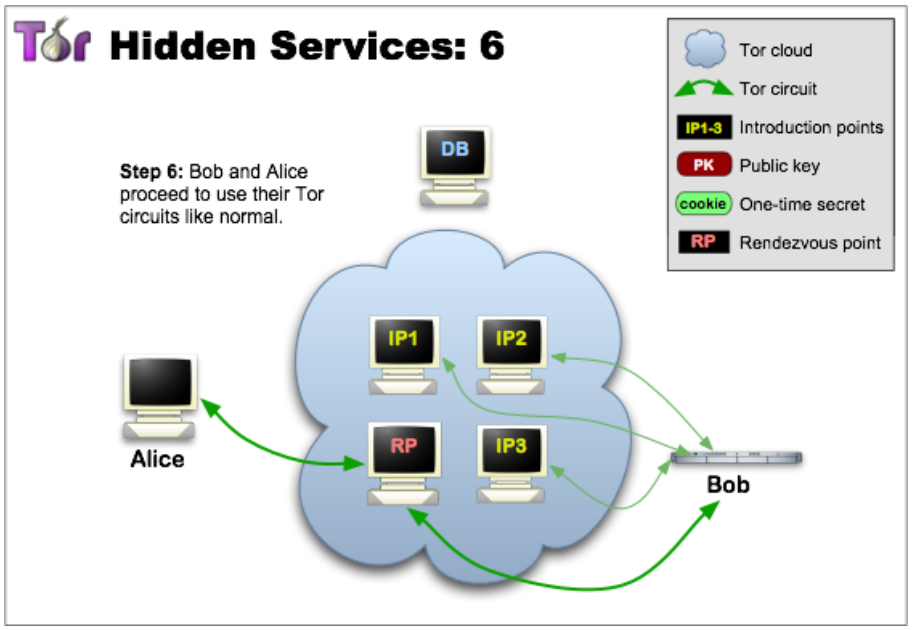


Quinto passaggio: l'hidden service decifra l'introduce message del client e scopre l'indirizzo del rendez-vous point ed il “One-time secret” contenuto. Il service crea un circuito verso il rendez-vous point e gli invia il “One-time secret” in un rendez-vous message.

Il circuito tra l’introduction point e il provider comunque non sarà distrutto, permettendo così al primo di continuare ad accettare connessioni per mezzo del secondo.



Nell'ultimo passaggio il rendez-vous 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 rendez-vous point per comunicare tra di loro. Il rendez-vous point inoltra semplicemente i messaggi (cifrati end-to-end) dal client al service e viceversa.



4. Etiquette e abuso

Gli eventuali abusi della rete Tor vengono mitigati dalla possibilità, per ciascun nodo Tor di uscita (exit node), di definire una politica d'uscita (exit policy) che definisca quale tipo di traffico può uscire o meno attraverso l'exit node. Usando una combinazione di indirizzi e porte, è possibile combattere la maggior parte degli abusi.

I potenziali abusi includono:

4.1. Esaurimento della banda

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

4.2. E-mail

L'uso anonimo del protocollo SMTP può portare allo Spam. Di conseguenza la politica di uscita di default dei nodi Tor rifiuta le connessioni in uscita verso la porta 25, cioè quella usata per SMTP.

4.3. 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. Wikipedia ad esempio inibisce la modifica delle proprie pagine agli utenti anonimi che si collegano via Tor, tramite un'estensione chiamata *"TorBlock"*. L'uso da parte di utenti registrati resta comunque soggetto a dei limiti.

5. Attacchi e Difese

Vengono descritti brevemente in questo capitolo alcuni possibili attacchi che possono essere eseguiti contro la rete Tor; tali attacchi si distinguono in passivi ed attivi.

5.1. Attacchi passivi

5.1.1. Osservazione dei pattern di traffico

Attacco

Osservare le connessioni di un utente non svela la sua destinazione o il contenuto dei messaggi trasmessi, ma può indicare pattern di traffico sia del mittente che del ricevente; infatti, il traffico di un torrent, ad esempio, è voluminoso e continuo mentre il traffico http è intervallato.

Difesa

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

5.1.2. Estrazione dei contenuti

Attacco

Mentre il traffico che viene trasmesso lato client viene cifrato dalla rete Tor, quello in uscita da un exit node verso la destinazione finale non è detto che lo sia; ciò comporta per un attaccante il poter osservare il traffico in chiaro in uscita da un exit node e inferire sul client che ha inoltrato la richiesta.

Pensiamo per esempio agli header del protocollo http:

- User agent
- Cookies
- Parametri GET e POST

Difesa

Anche se il filtering non è l'obiettivo principale di Tor, esso utilizza Privoxy ed i relativi servizi di filtering per anonimizzare i data stream delle applicazioni.

5.1.3. 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 ciò che ha inviato, si può capire ciò a cui era rivolto l'interesse della vittima. Se il protocollo era in chiaro l'attaccante ottiene anche il contenuto.

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.

5.1.4. Correlazione sulla dimensione dei pacchetti

Attacco

Tale attacco è simile al precedente, ma anziché 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.

Difesa

L'utilizzo del padding delle connessioni sarebbe ottimo, ma 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 causando volumi di traffico differenti tra pacchetti in ingresso e in uscita.

5.1.5. Fingerprinting

Attacco

Simile al precedente ma elimina la necessità di osservare l'uscita. L'attaccante conosce un set di dati riguardo la taglia dei file e dei 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.

Difesa

Questo tipo di attacco è reso più difficile da attuare sulla rete Tor, poiché essa multiplexa stream multipli su di una stessa connessione e scambia celle di taglia fissata.

5.2. Attacchi attivi

5.2.1. Compromissione di chiavi

Attacco

- Se un attaccante conosce la chiave di sessione TLS di un nodo, è in grado di vedere le celle di controllo e le relay-cells cifrate su quella connessione per tutto il lifetime della chiave.
- 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.

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.

5.2.2. Tagging

Attacco

Un attaccante potrebbe 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 di dati previene questo tipo di attacco.

5.2.3. 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

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/

5.2.4. 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.

5.3. Attacchi alle autorità di directory

5.3.1. Destroy directory server

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.

5.3.2. Compromettere la maggioranza dei Directory server

Attacco

Un attaccante che controlla più della metà dei directory server può introdurre degli OR maligni all'interno della rete.

Difesa

Per difendersi da questi attacchi Tor deve assicurare:

- Indipendenza dei directory server.
- la resistenza dei directory server agli attacchi.

5.3.3. Splitting

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

Non vi è alcuna difesa possibile a questo tipo di attacco poiché Tor è una rete di fiducia con un numero di autorità limitato (generalmente 5).

5.3.4. Tricking

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, ma si sta sviluppando un modo per la ricerca automatica di nodi di uscita mal funzionanti.

5.4. Attacchi contro i nodi di rendezvous

5.4.1. Flooding

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

Le possibili soluzioni a tali attacchi sono:

- Token di autorizzazione delle richieste.
- Limitazione dei rate di richiesta.

5.4.2. Compromissione di nodi

Attacco

Un circuito verso un hidden service è esattamente identico a un circuito verso un host esterno alla rete torificata.

Se un attaccante compromette un nodo di introduzione può rigirare il traffico dove vuole oppure effettuare flood di introduction request.

Un attaccante potrebbe distruggere un hidden service disabilitando tutti i suoi introduction point

Difesa

Il servizio nota il flood di richieste e chiude il circuito con l'introduction point.

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.

6. Tor in 4 semplici passi

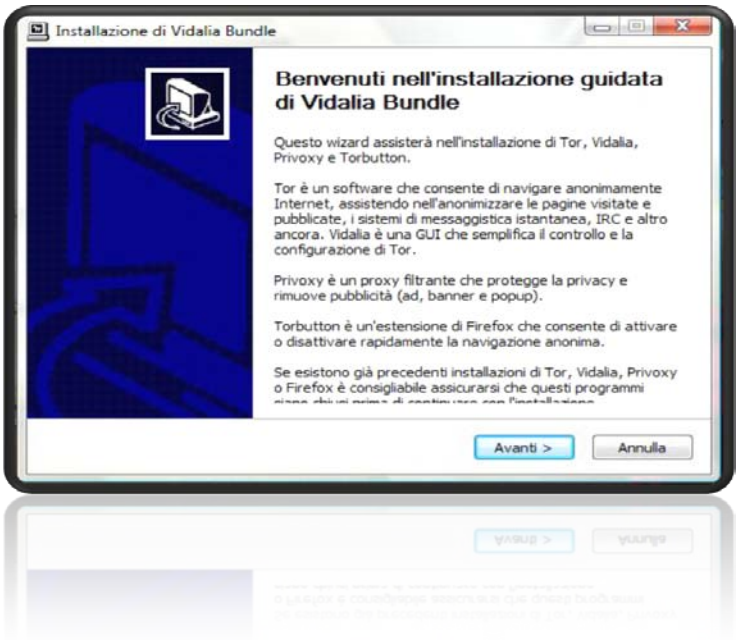
In questo capitolo vedremo come configurare il servizio Tor sulla nostra macchina.

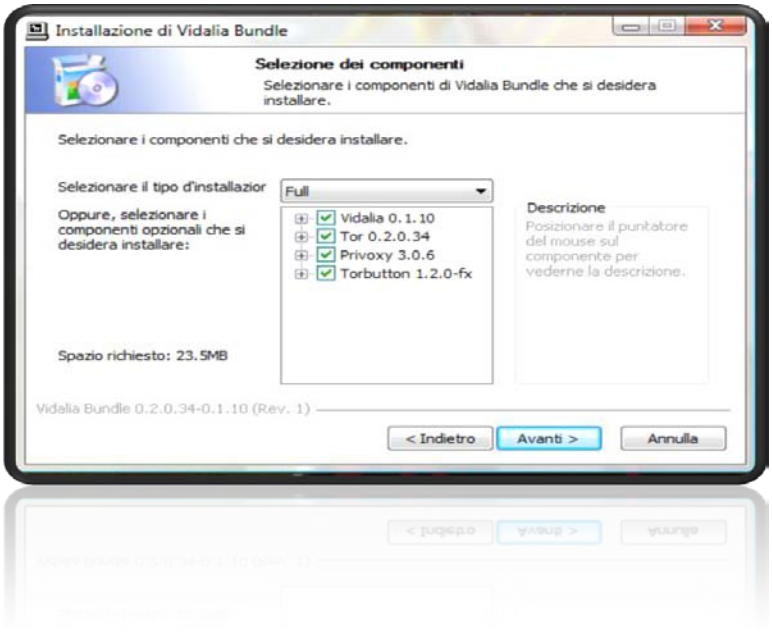
I passi da eseguire per realizzare una tale operazione sono i seguenti:

- 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”.

Installazione del pacchetto Vidalia Bundle

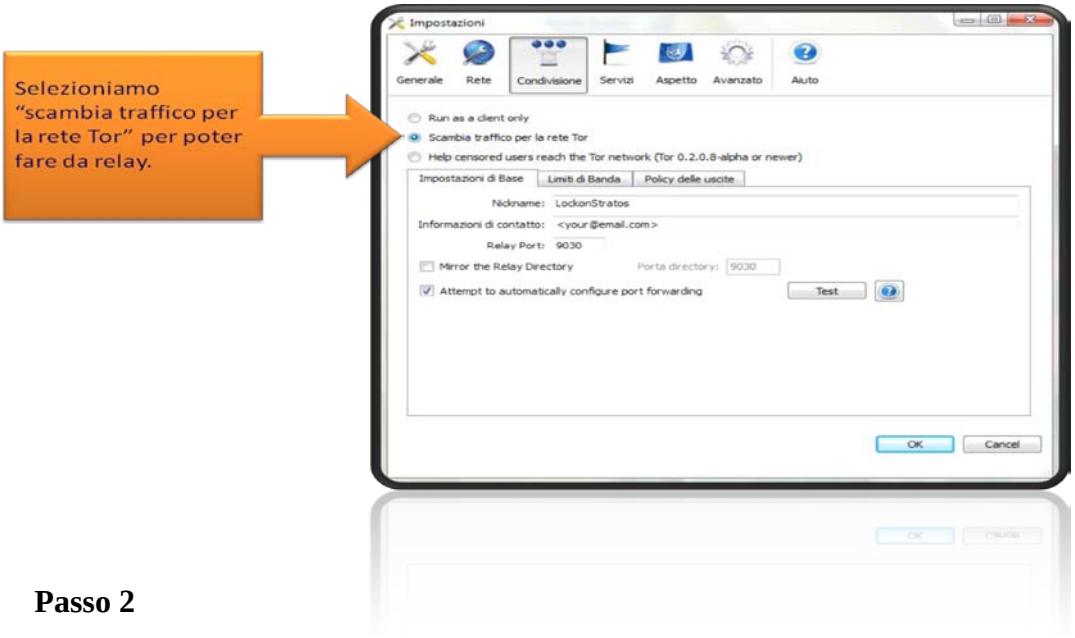
Avviamo l'installazione del software facendo doppio click sul file .exe scaricato dal sito tor.eff.org e selezioniamo l'installazione completa (Full) come si vede nella figura che segue.



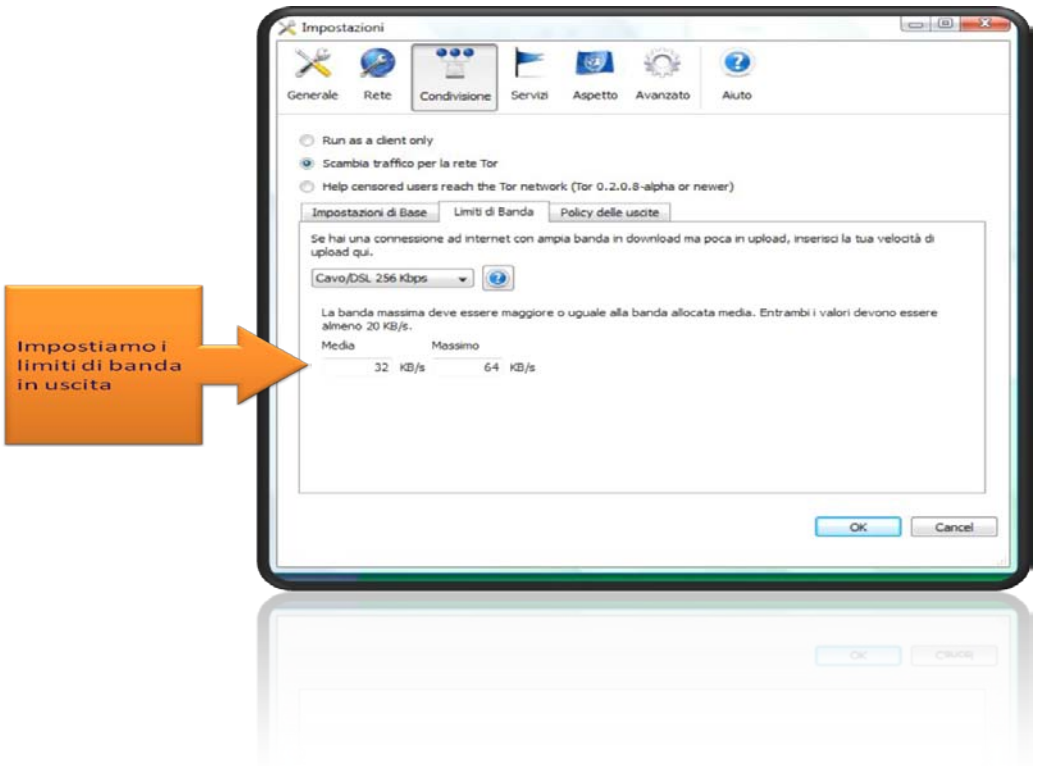


Configurazione della macchina come Relay Tor

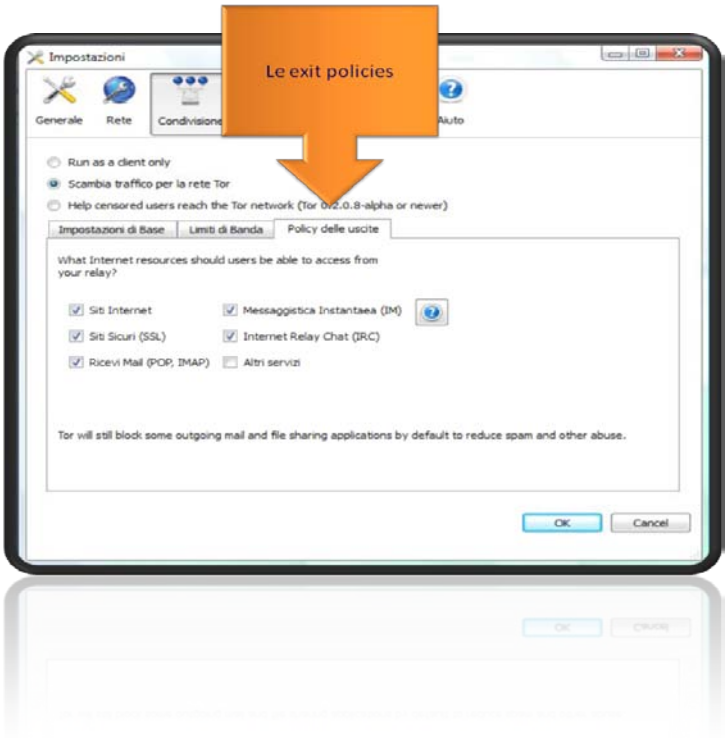
Passo 1



Passo 2



Passo 3



Passo 4

Navigazione anonima con tor button

Cliccando sul pulsante “tor button” presente nella zona inferiore del browser si potrà sfruttare la rete Tor per la navigazione web. La navigazione è, inoltre, resa sicura dalla presenza di privoxy (il proxy installato con il pacchetto Vidalia)

7. Conclusioni

Per quanto riguarda il mantenimento della rete Tor potremmo chiederci se un approccio basato sul volontariato potrà sostenere la rete in un futuro a lungo termine, inoltre, 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 ed è necessario che impari a coesistere con la varietà di servizi internet ed i loro meccanismi di autenticazione. L'attuale architettura del sistema non è sufficientemente scalabile per gestire la domanda di utenti attuale.

In conclusione, Tor rappresenta senza dubbio un utile servizio pubblico di anonimato esistente, ma poiché esso si regge sulla "fidatezza" dei Directory server e questi sugli OR fidati, un suo utilizzo poco coscienzioso potrebbe rivelarsi di poca utilità per la tutela della privacy.

Bibliografia

1. Marco Bonetti, *Attacchi a Tor: come funzionano, come difendersi*, MOCA 2008 – Pescara.

   http://sid77.slackware.it/tor/Attacchi_a_Tor_come_funzionano_come_difendersi.pdf

   http://camp.olografix.org/home.php?goto=1&lng=#torattack

2. Roger Dingledine, Nick Mathewson and Paul Syverson,

   Challenges in deploying low-latency anonymity (DRAFT),

   http://www.torproject.org/documentation.html.it.

3. Marco A. Calamari, *Condivisione anonima di informazioni: remailer anonimi, server di pseudonimi, darknet*, Infosecurity 2006.

   http://www.sikurezza.org/wiki/Risorse/Infosecurity06?action=download&upname=Infosecurity_2006_Condivisione_anonima_delle_informazioni.pdf

4. Roger Dingledine, Nick Mathewson, Paul Syverson,

   Tor: The Second-Generation Onion Router

   http://www.torproject.org/documentation.html.it , Usenix Security 2004.

5. Gianni Bianchini, *Comunicazione anonima in retemediante tecniche di onion routing: TOR*, Comunicazione e Sicurezza @ LILiK Firenze, Facoltà di Ingegneria - 16 novembre 2006.

   http://www.giannibi.net/lilik.pdf

6. Francesco Poli, *Introduzione ai remailer anonimi*, Versione 2.1 2003-2004

   http://e-privacy.winstonsmith.info/2005/2004/atti/ep2004-Poli-remailers_printable.pdf

7. David Goldschlag, Michael Reedy, Paul Syversony,

   Onion Routing for Anonymous and Private Internet Connections January 28-1999, http://www.onion-router.net/Publications/CACM-1999.pdf.

8. Privoxy: navigazione sicura a prova di spam -Guide@Debianizzati.Org http://guide.debianizzati.org/index.php/Privoxy:_navigazione_sicura_a_prova_di_spam

9. Chi usa Tor?, Anonimato in rete, Tor: panoramica, Il protocollo hidden service, http://www.torproject.org/index.html.it.

10. Tor Software di anonimato, http://it.wikipedia.org/wiki/Tor_(software_di_anonimato).

11. TOR Onion Routing internals: funzionamento e analisi dell'architettura,

    http://www.makeinstall.it/tor-internals-funzionamento-e-analisi-dellarchitettura.html.

12. Anonymous remailer http://it.wikipedia.org/wiki/Anonymous_remailer.

13. Daniele Masini, *Informatica e GNU/Linux versione 2006.7.29,*

    29 luglio 2006. http://vandali.org/DanieleMasini/infolinux.php.