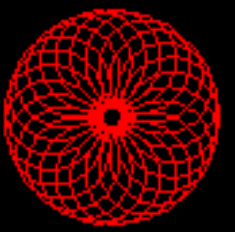
Lo Spirografo

www.spirografo.net



RETI DISTRIBUITE

Tor

Introduzione

- Tor è un sistema di comunicazione anonima per internet basato sulla seconda generazione del protocollo “onion routing”.
- Cosa vedremo:
  - offerta
  - funzionamento
  - problematiche

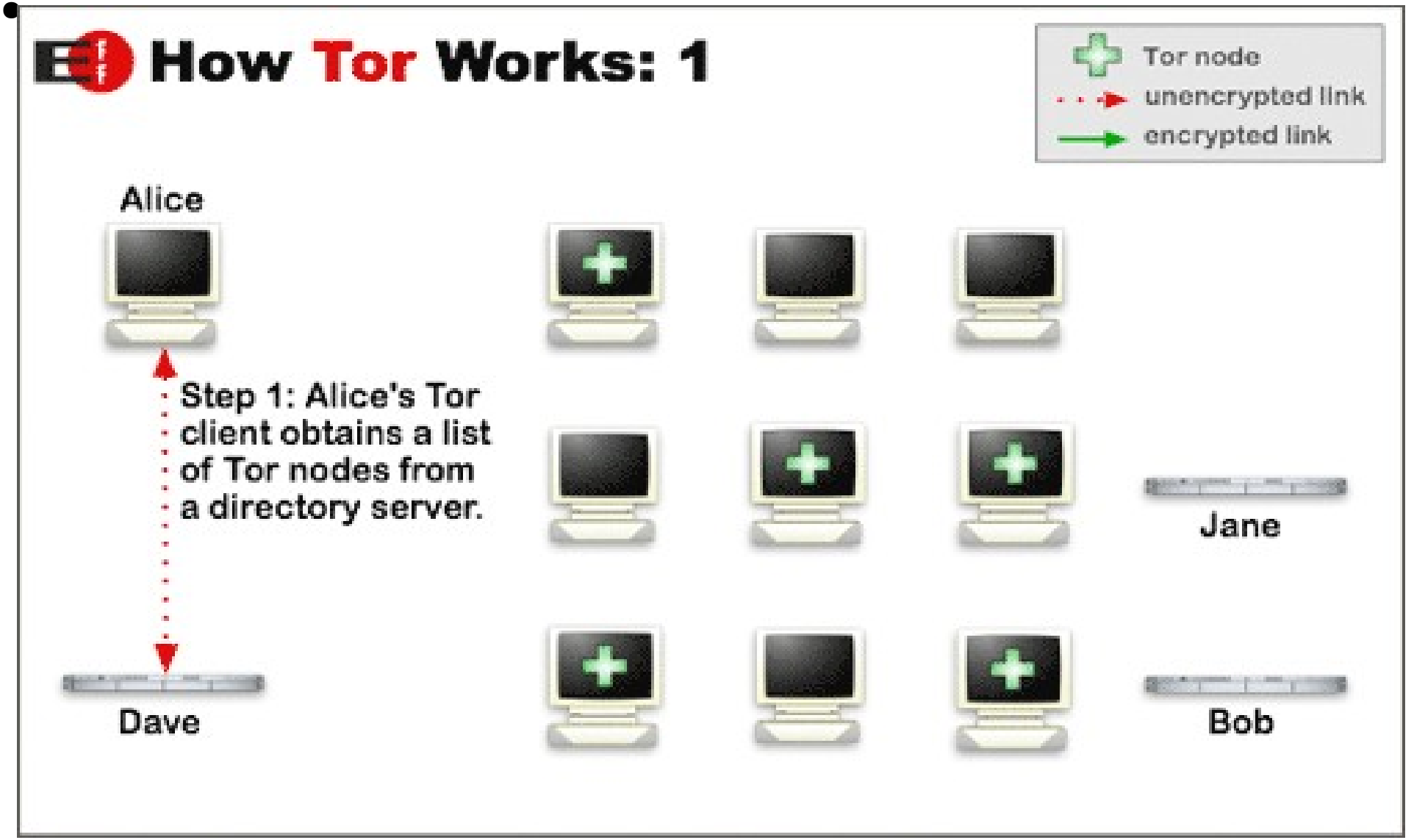
Scheda riassuntiva

- Tor è free software (brevetti?)
- Sito -> http://tor.eff.org
- Originariamente sponsorizzato dalla US Naval Research Laboratory, è diventato un progetto della Electronic Frontier Foundation (EFF) alla fine del 2004.
- Progetto originario:
  - Roger Dingledine
  - Nick Mathewson
  - Paul Syverson

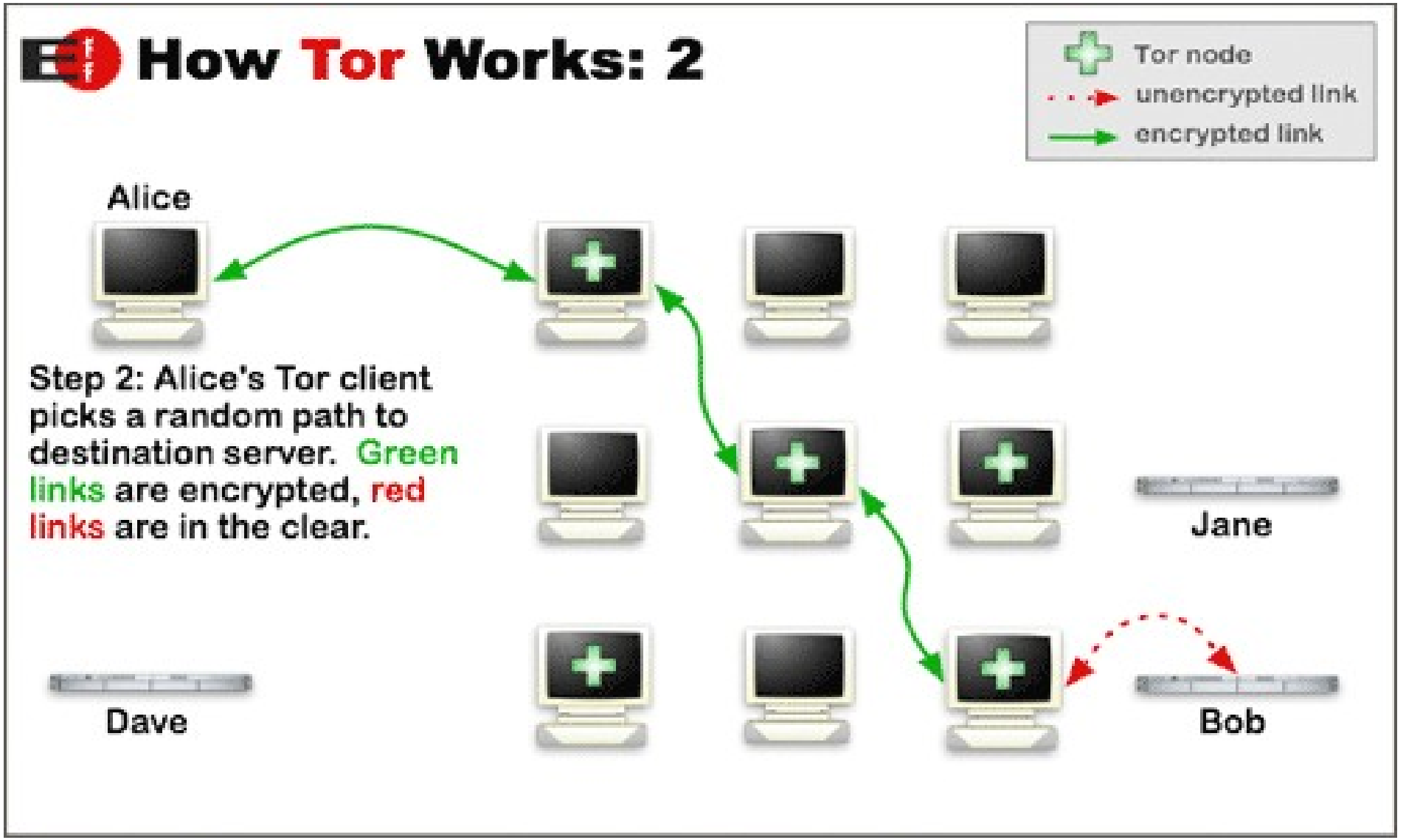
Funzionamento

- **Un applicazione può ridirigere il proprio traffico attraverso un Onion Proxy (OP) che solitamente è eseguito localmente**
- **L'OP funga da SOCKS proxy per l'applicazione**
- **Un OP crea un canale passante per vari Onion Router (OR) fino a raggiungere la destinazione voluta**
- **Ogni comunicazione fra OP e OR e fra OR e OR è cifrata**
- **Il server di destinazione “vede” come indirizzo del client l'ultimo OR del percorso e non quello del mittente**
- **L'elenco degli OR è ottenuto da “directory server”**

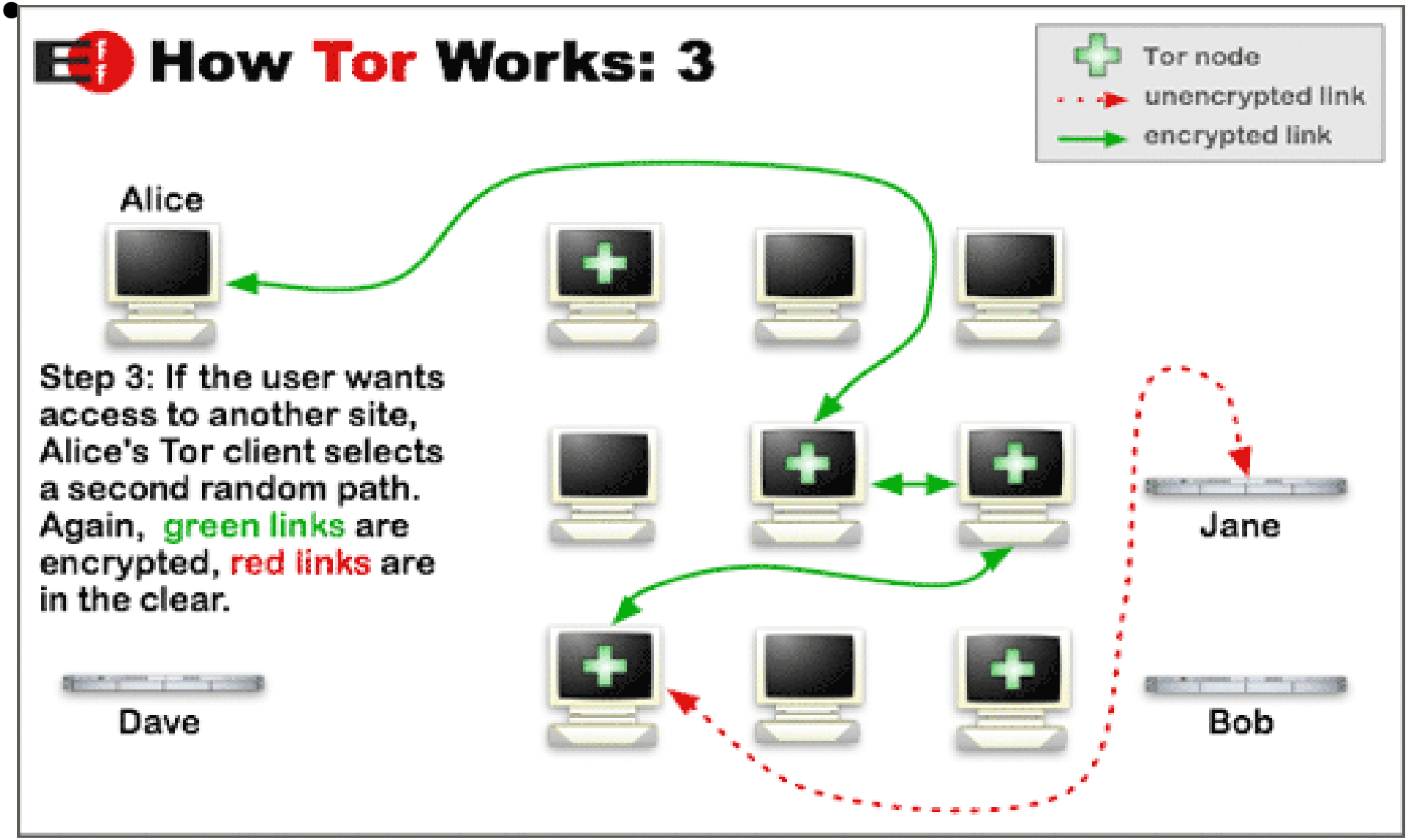
Funzionamento



Funzionamento



Funzionamento



I Directory Server

- Per prima cosa il client OP si connette al “directory server” e scarica la lista dei nodi OR disponibili e delle chiavi di identificazione di quest'ultimi.
- I “directory server” offrono una lista (autenticata) di tutti gli OR, del loro stato, delle loro “exit policies”, ...
- E' possibile un intervento “umano” sui “directory server” da parte dei gestori
- Per guadagnare banda non sempre viene scaricata una lista completa
- Gli indirizzi dei “directory server” (attualmente 5) sono hardcoded nei sorgenti di Tor (file /src/or/config.c). Si trovano 3 in USA (Mit, Harvard) e 2 in Europa
- Sono chiaramente un punto vulnerabile dell'intero meccanismo,
  nell'ultima versione di Tor (0.2.0.x alpha) cambiano alcune cose

Guardando dentro al sorgente...

```
const char *dirservers[] = {
  "moria1 v1 orport=9001 v3ident=5420FD8EA46BD4290F1D07A1883C9D85ECC486C4 "
    "128.31.0.34:9031 FFCB 46DB 1339 DA84 674C 70D7 CB58 6434 C437 0441",
  "moria2 v1 orport=9002 128.31.0.34:9032 "
    "719B E45D E224 B607 C537 07D0 E214 3E2D 423E 74CF",
  "tor26 v1 orport=443 v3ident=A9AC67E64B200BBF2FA26DF194AC0469E2A948C6 "
    "86.59.21.38:80 847B 1F85 0344 D787 6491 A548 92F9 0493 4E4E B85D",
  "lefkada orport=443 140.247.60.64:80 "
    "38D4 F5FC F7B1 0232 28B8 95EA 56ED E7D5 CCDC AF32",
  "dizum 194.109.206.212:80 "
    "7EA6 EAD6 FD83 083C 538F 4403 8BBF A077 587D D755",
  "Tonga orport=443 bridge no-v2 82.94.251.206:80 "
    "4A0C CD2D DC79 9508 3D73 F5D6 6710 0C8A 5831 F16D",
  NULL
};
```

Routing

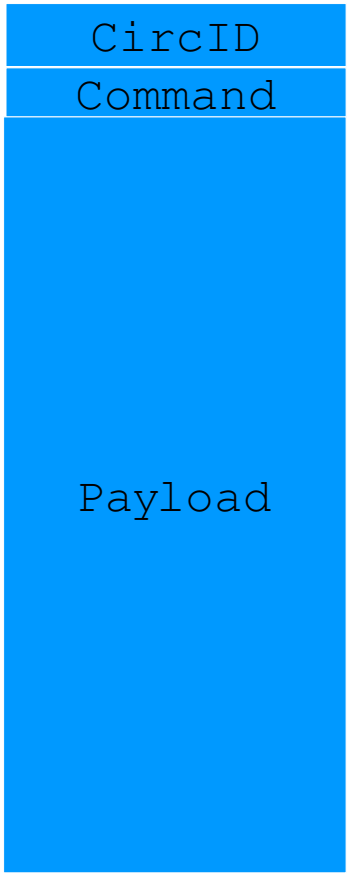
- Un applicazione si connette all' OP come se fosse un proxy SOCKS
- OP guarda la destinazione e stabilisce il percorso attraverso gli OR. Solo alcuni OR permettono il traffico verso l'esterno (cioe' NON verso OR) e comunque questo è regolato da “exit policies” configurabili dal gestore dell' OR
- Una volta stabilito l'ultimo OR si l'OP sceglie gli altri OR del percorso
- A questo punto bisogna comunicare coi vari OR scelti e creare il percorso, vediamo come...

Celle

- Senza entrare troppo nei dettagli diciamo che:
  - i dati viaggiano in celle da 512 Byte
  - ogni trasmissione è criptata con una chiave comune (protocollo TLS)
  - lo scambio delle chiave (che costituisce il “segreto comune” fra OP-OR e OR-OR) avviene con protocollo Diffie-Hellman all'interno di una cella di tipo Create. Dalla chiave scambiata si ricavano le chiavi simmetriche (2, una per senso di comunicazione) usate per criptare le celle.

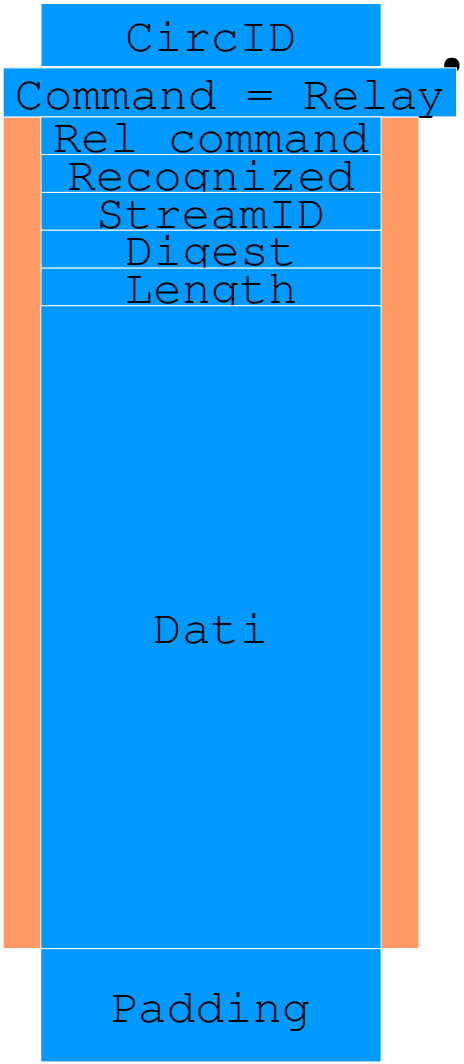
- Diffie-Hellman, come funziona?
  - Alice e Bob (e chiunque altro) conoscono un valore g e p
  - tutte le operazioni avvengono in modulo p
  - Alice sceglie un valore casuale X, Bob un valore casuale Y
  - Alice invia g^X a Bob e Bob invia g^Y ad Alice
  - Alice calcola (g^Y)^X e Bob calcola (g^X)^Y, questi 2 valori risultano identici!
  - la chiave comune è quindi (g^X)^Y=(g^Y)^X=g^(X*Y) (proprietà delle potenze!)
  - se qualcuno intercetta le comunicazioni fra Alice e Bob può conoscere solo: g, p, g^X, g^Y : non può calcolare g^(X*Y)!
  - potrebbe farlo calcolando il logaritmo in base g di g^X , ma non esistono ad ora algoritmi efficienti (polinomiali)

Struttura delle celle



- CircID
  - identifica il circuito, ogni OR infatti può avere a che fare con più circuiti ed è quindi necessario un meccanismo con cui distinguerli
  - è specifico per la connessione, cioè è diverso per ogni connessione OP-OR e OR-OR che il circuito attraversa
- Command
  - 0 -- PADDING (Padding)
  - 1 -- **CREATE** (Crea un circuito)
  - 2 -- **CREATED** (Conferma creazione)
  - 3 -- **RELAY** (Invio dati, vediamo dopo)
  - 4 -- DESTROY (Chiude un circuito)
  - 5 -- CREATE_FAST (Crea un circuito, senza generare una nuova chiave)
  - 6 -- CREATED_FAST (Conferma creazione, senza generare una nuova chiave)

Struttura delle celle



- Celle di tipo Relay
- Rel command
  - 1 -- RELAY_BEGIN
  - 2 -- RELAY_DATA
  - 3 -- RELAY_END
  - 4 -- RELAY_CONNECTED
  - 5 -- RELAY_SENDME
  - 6 -- RELAY_EXTEND
  - 7 -- RELAY_EXTENDED
  - 8 -- RELAY_TRUNCATE
  - 9 -- RELAY_TRUNCATED
  - 10 -- RELAY_DROP
  - 11 -- RELAY_RESOLVE
  - 12 -- RELAY_RESOLVED

Celle relay

- Il payload dele celle Relay dirette verso OR può essere letto e quindi “processato” dal nodo che lo riceve solo se è quello desiderato dall'OP. Questo perchè:
  - il payload di ogni cella è criptato dall'OP con tutte le chiavi condivise coi nodi che la cella dovrà attraversare
  - il payload di ogni cella è decriptato da ogni OR con la chiave che condivide con l'OP e inviato all'OR successivo
  - il pacchetto verrà quindi decriptato correttamente solo se avrà attraversato tutti gli OR che l'OP ha stabilito
  - in quel caso “recognized” varrà = 0

Celle relay

- Il payload delle celle Relay dirette verso l'OP può essere letto solo dall'OP Questo perchè:
  - ogni cella è criptata da ogni OR che attraversa con la chiave che questo condivide con l'OP
  - quando la cella raggiunge l'OP sarà quindi criptata da tutte le chiavi che l'OP condivide con gli OR che la cella ha attraversato
  - solo l'OP conosce tutte queste chiavi e quindi solo l'OP può decriptare la cella

Celle relay

- Celle Relay EXTEND
  - Servono per costruire un circuito estendendolo di un OR
  - supponiamo che l'attuale ultimo nodo di un circuto sia OR1 e che l'OP voglia estendere il circuito verso OR2
  - Quando la cella viene processata (ha raggiunto OR1) il suo payload è utilizzato per creare una cella Create da inviare a OR2
  - OR2 risponde con una cella Created a OR1
  - La cella Created raggiunge l'OP all'interno di una cella Relay EXTENDED
  - Notare che l'OR2 non può sapere se OR1 è un OR o un OP, un OR conosce l'OR precedente e successivo, ma non l'intero percorso

Celle relay

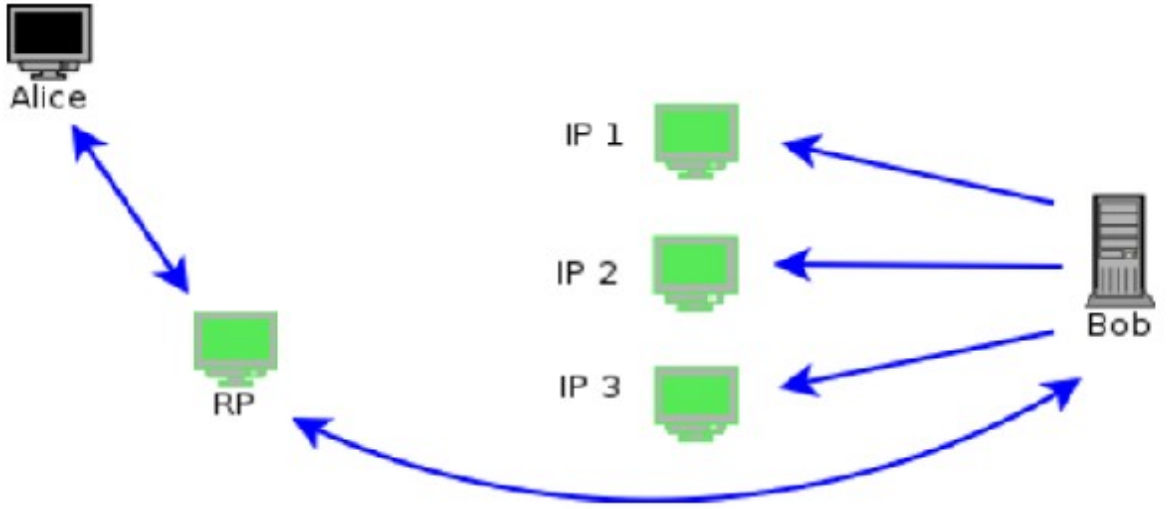
- Celle Relay BEGIN
  - Comunicano all'ultimo OR del circuito (“exit node”) di creare una connessione TCP verso un server
- Celle Relay CONNECTED
  - Comunicano all'OP l'avvenuta connessione ad un server
- Celle Relay DATA
  - Usate per il traffico dati

Altri meccanismi

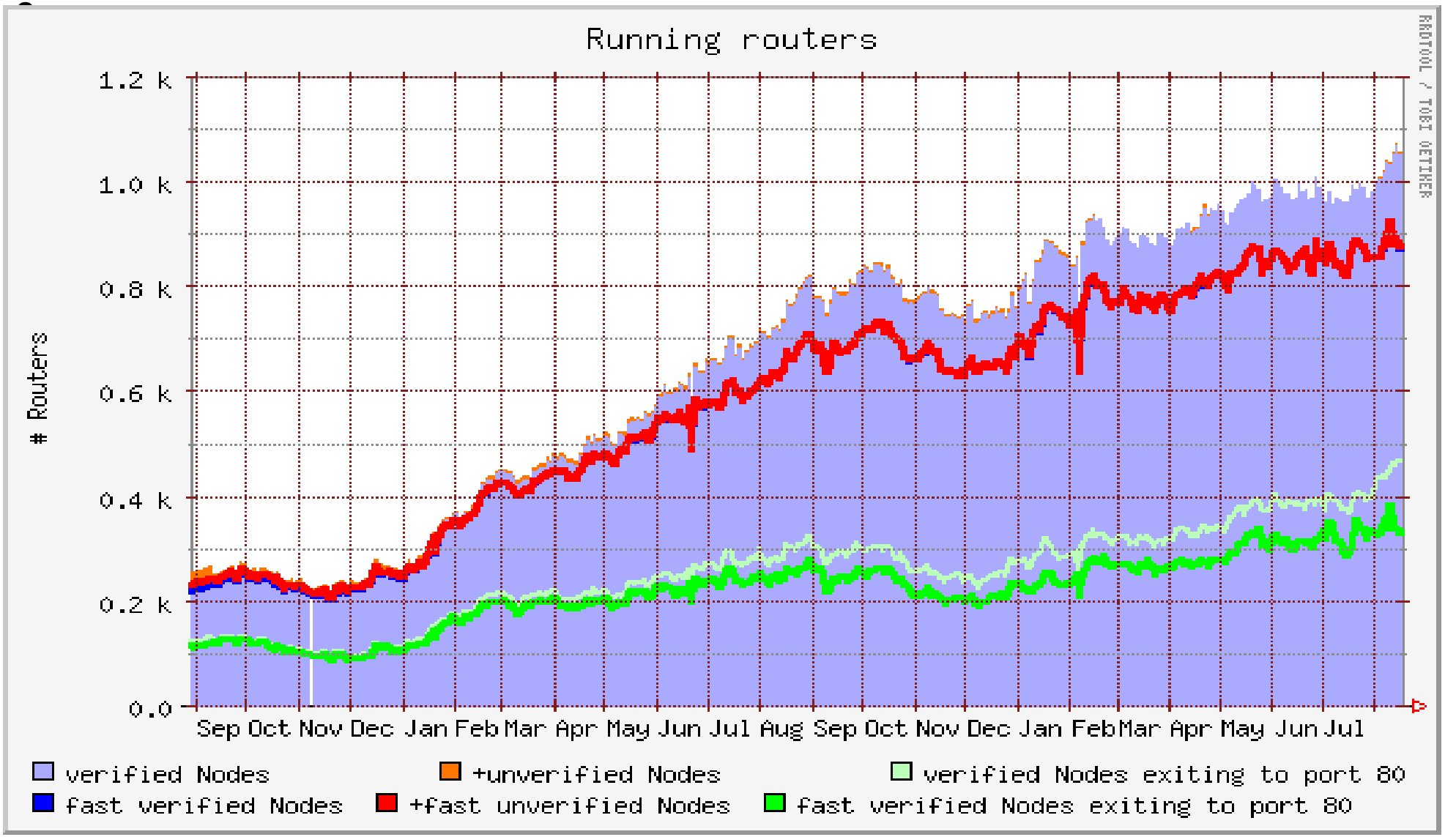
- Tor presenta numerosi altri meccanismi, vediamoli brevemente:
  - I circuiti vengono generalmente creati prima di quando sia effettivamente necessario
  - E' possibile cambiare il percorso di un circuito
  - Lo stesso circuito può trasportare diversi flussi TCP
  - Controllo di integrità sulle celle
  - Controllo di congestione
  - Possibilità di limitare il traffico generato dagli OR
  - Hidden service (domini .onion)
  - ...

Hidden service

- E' possibile rendere accessibile all'interno della rete Tor un server che offra qualunque tipo di servizio TCP
- Viene garantita l'impossibilità di identificare chi fornisce il servizio e chi ci accede
- La comunicazione fra client e server avviene infatti tramite 2 circuiti anonimi. Uno dal client diretto verso un OR definito come “punto di ingresso”. L'altro fra questo OR e il server.

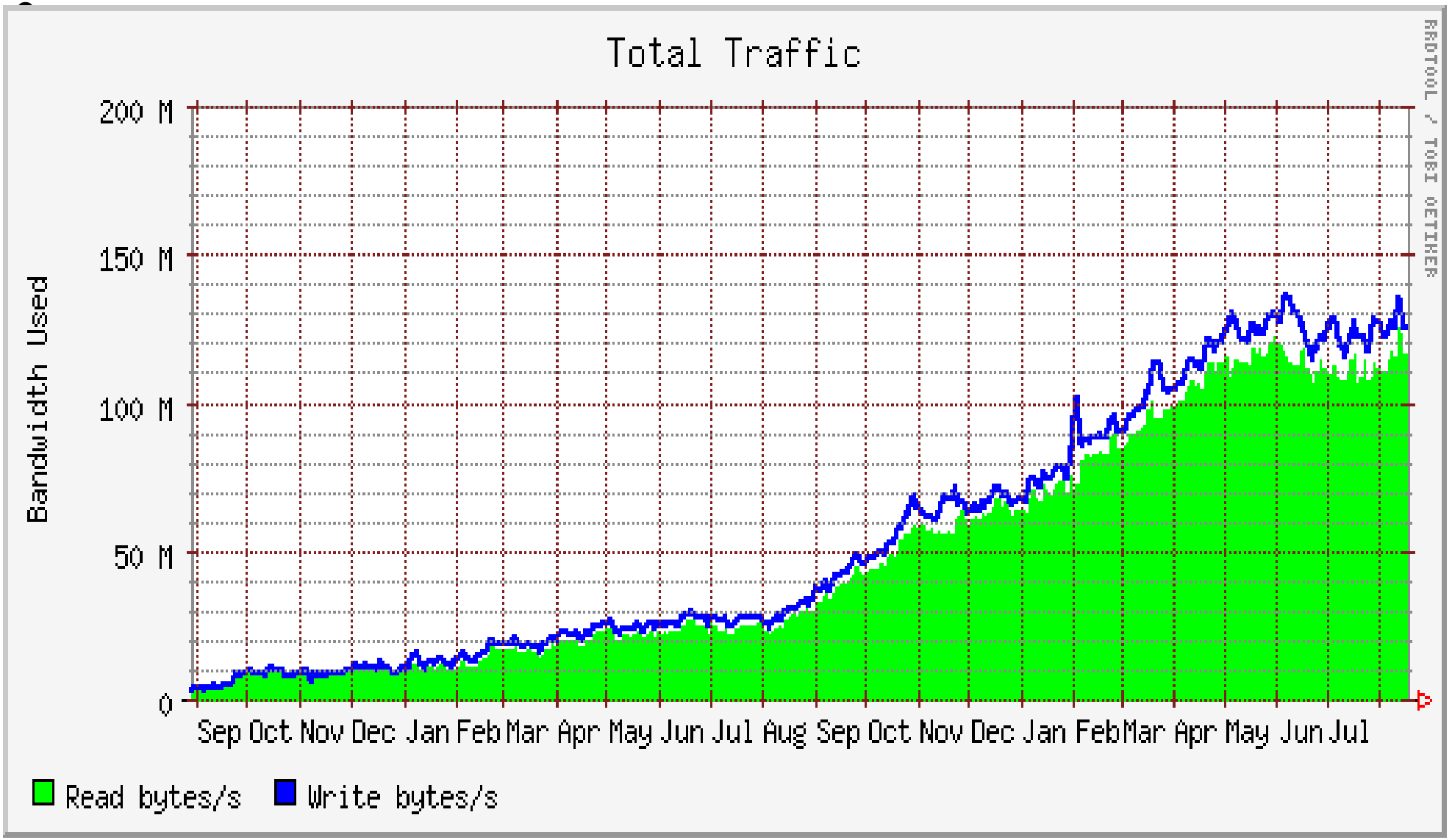


Statistiche



Statistiche

(ultimi 24 mesi)



Problematiche

- Tor, da solo, NON garantisce l'anonimato, infatti:
  - l'applicazione che utilizza la rete Tor può comunque rivelare la posizione del client sul quale è eseguita. Molti protocolli ad esempio comunicano l'IP del client sul quale sono eseguiti, oppure sono comunicate informazioni che permettono di distinguere un client da un altro, ...

    Per quanto riguarda la navigazione Web esiste un proxy software (Privoxy) che serve a eliminare dal traffico tutte quelle informazioni che possono essere utili al server per riconoscere e individuare un client

Problematiche

- Tor, da solo, NON garantisce l'anonimato, infatti:
  - Tor non è efficace contro l'analisi del traffico end-to-end
    Se qualcuno può intercettare le comunicazioni all'inizio e alla fine del circuito, analizzando la quantità di traffico in entrata e in uscita dal circuito, può facilmente capire se il client e il server stanno comunicando.
    Questo problema è difficilmente risolvibile in modo completo, ed è comune a tutte le reti a bassa latenza, in cui cioè, non vengono introdotti ritardi

Problematiche

- Tor, da solo, NON garantisce l'anonimato, infatti:
  - Non tutte le applicazioni utilizzano solamente pacchetti TCP
  - Non tutte le applicazioni possono essere configurate per dirigere il proprio traffico verso un server SOCKS
    E' possibile utilizzare tool come tsocks, SocksCap,FreeCap
    Per Firefox esiste il comodo plug-in SwitchProxy
  - Questo vale soprattutto per quanto riguarda il modo in cui le applicazioni sono solite a contattare i DNS
  - Il traffico fra “exit node” e server di destinazione NON è cifrato (a meno che si utilizzino protocolli cifrati come https)
    Un “exit node” può leggere o modificare il traffico
  - Se una stessa entità controlla una percentuale significativa degli OR può compromettere l'anonimato della rete

Problematiche

- Tor NON è sempre molto performante (10KB/s - 50 KB/s)
- Gli “exit nodes” sono pochi, spesso sovraccarichi
- Far girare un “exit nodes” può essere rischioso (numerosi “exit nodes” chiusi e sequestrati in Germania)
- Bannare gli utenti Tor è facilissimo: la lista degli “exit nodes” è disponibile a tutti contattando i “directory server”
- ...

Novità in arrivo

- la versione 0.2 è attualmente alpha
  - version 3 directory server
    - Gli OR possono funzionare come “directory cache”
    - i “directory server” votano uno stato della rete
    - ...
  - IPv6
  - Tor agisce anche come proxy per le richieste DNS

Informazioni varie

- http://tor.eff.org/documentation.html.it
  - *pagina di documentazione principale, contiene i link a svariate altre risorse*
- le specifiche si trovano anche nella cartella di tor, dopo averlo installato
- http://torproxy.net/ *(Onion proxy con privoxy accessibile da http)*
- http://e-privacy.winstonsmith.info/2006/atti/e-privacy_2006_Bianchini_Servizi_anonimi_Tor.pdf
  - *slide riguardanti gli hidden service*
- http://www5.autistici.org/eazy/paper/tor.sxi
  - *slide dettagliate sulla creazione e l'utilizzo dei circuiti*
- per contattarci: info@spirografo.net