The state of academic research in Tor

Linus Nordberg, NORDUnet
linus@nordu.net 0x23291265

FRISC winter school, Finse, Norway

What is Tor?

How does Tor work?
- Overview
- Circuits and cell encryption
- The trust root
- Onion services

Academic research on Tor
- Circuit setup
- Routing
- Congestion
- Scalability
- Security

Research tools and community research

Three functions

- Anonymous and protected access to the internet
- Censorship circumvention
- Anonymous and protected publishing

Can mean many things

- A network
- A protocol and software
- A community
- An organisation

The network

- About 6500 relays – ordinary computers running the Tor software
- . . .at peoples homes, schools, workplaces
- Used for bouncing your traffic

The protocol and software

- A network protocol
- A program – free software, BSD licensed
- More software – a browser and 25 more sw projects
- Open mailing lists, source code repositories with specifications, proposals and code, a bug tracker, chat rooms
- Technique from NRL (U.S. Naval Research Laboratory) ==> “onion routing” 1996 ==> Tor 2002

The community

- Researchers
- Software developers
- Relay operators – 6.5k relays
- Users – 1-2M daily
- Support

The organisation

- A 501(c)(3) non-profit
- Employees – 10-15
- Contractors – 25
- Financing – EFF, Ford foundation, Google, HRW, NSF, private donors, RFA, Sida, US state department
- Turn-over – $ 3M 2013
- Infrastructure – metrics, atlas, compass, globe, ooni, support, torperf; bridges, check; exonerator, gettor, weather; deb, git, jenkins, track, people; lists, media, www, archive; backup, mail, nagios

What is Tor?

How does Tor work?

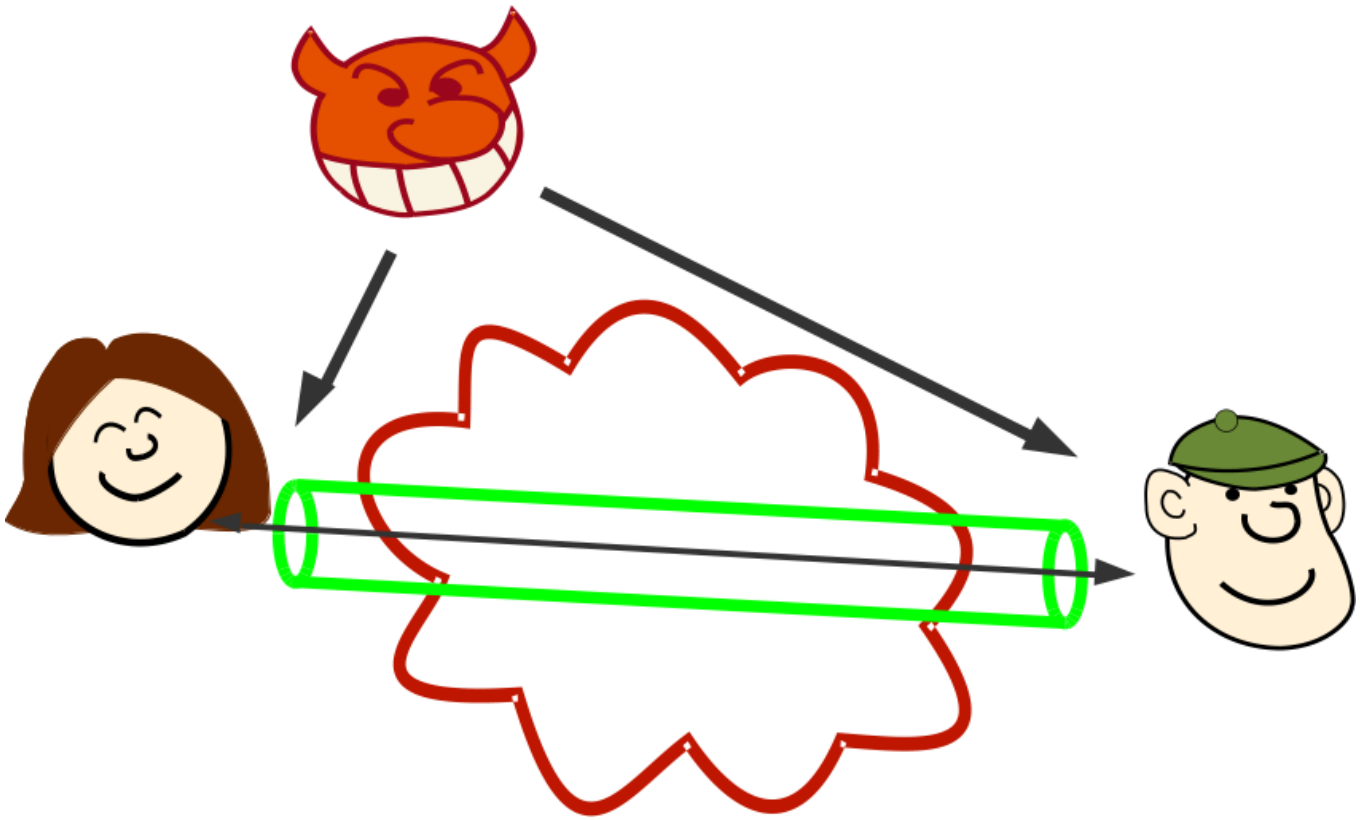
- Overview
- Circuits and cell encryption
- The trust root
- Onion services

Academic research on Tor

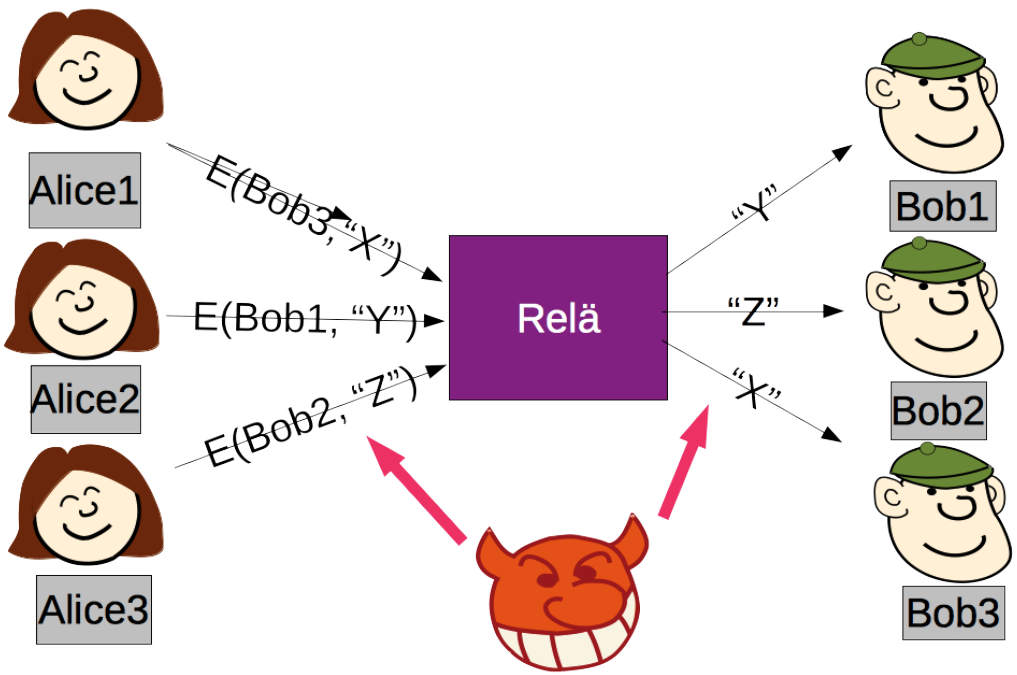
- Circuit setup
- Routing
- Congestion
- Scalability
- Security

Research tools and community research

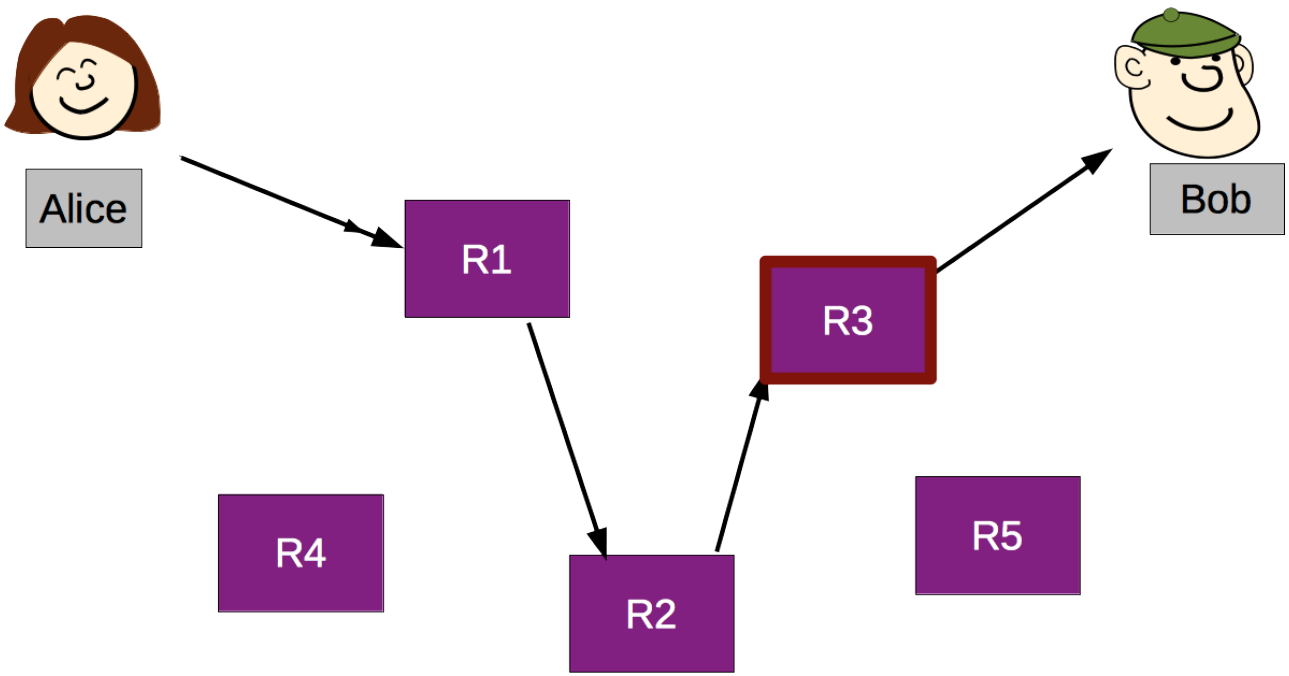
Encrypting payload doesn't protect against traffic analysis



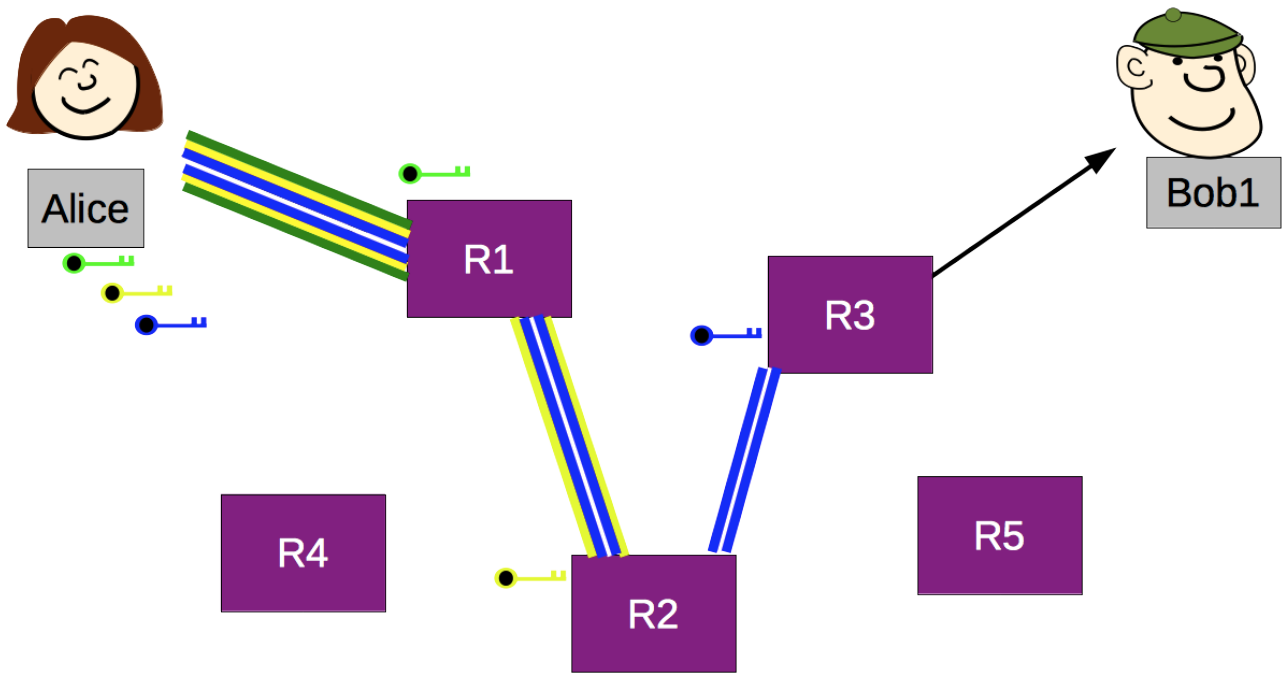
One hop isn't enough



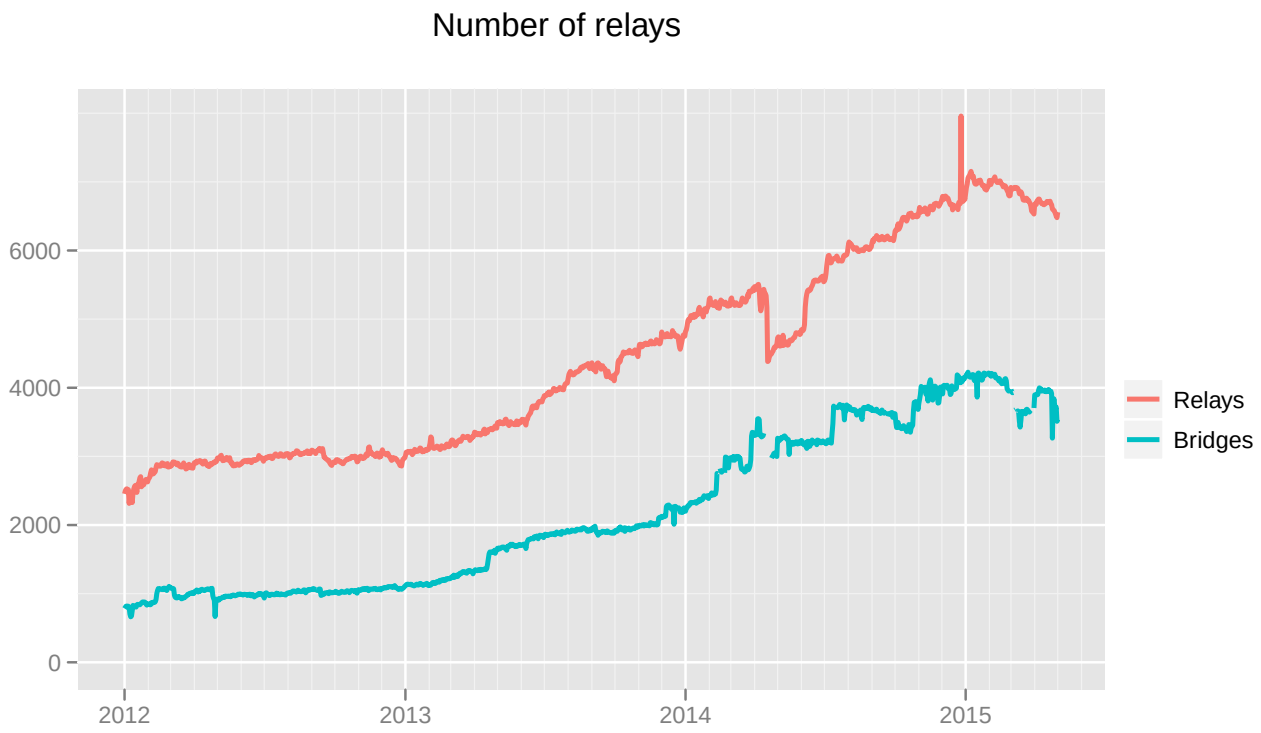
Three hops is good



Three layers of encryption

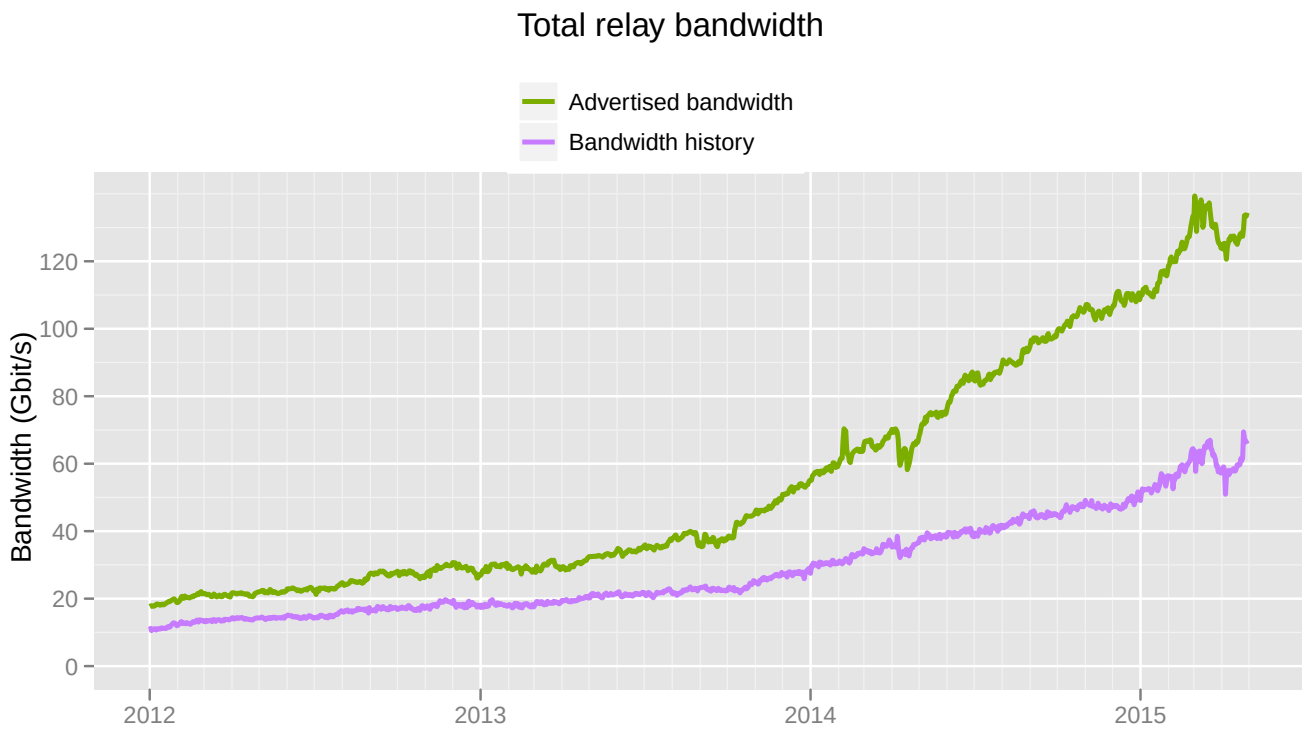


Relays in the network



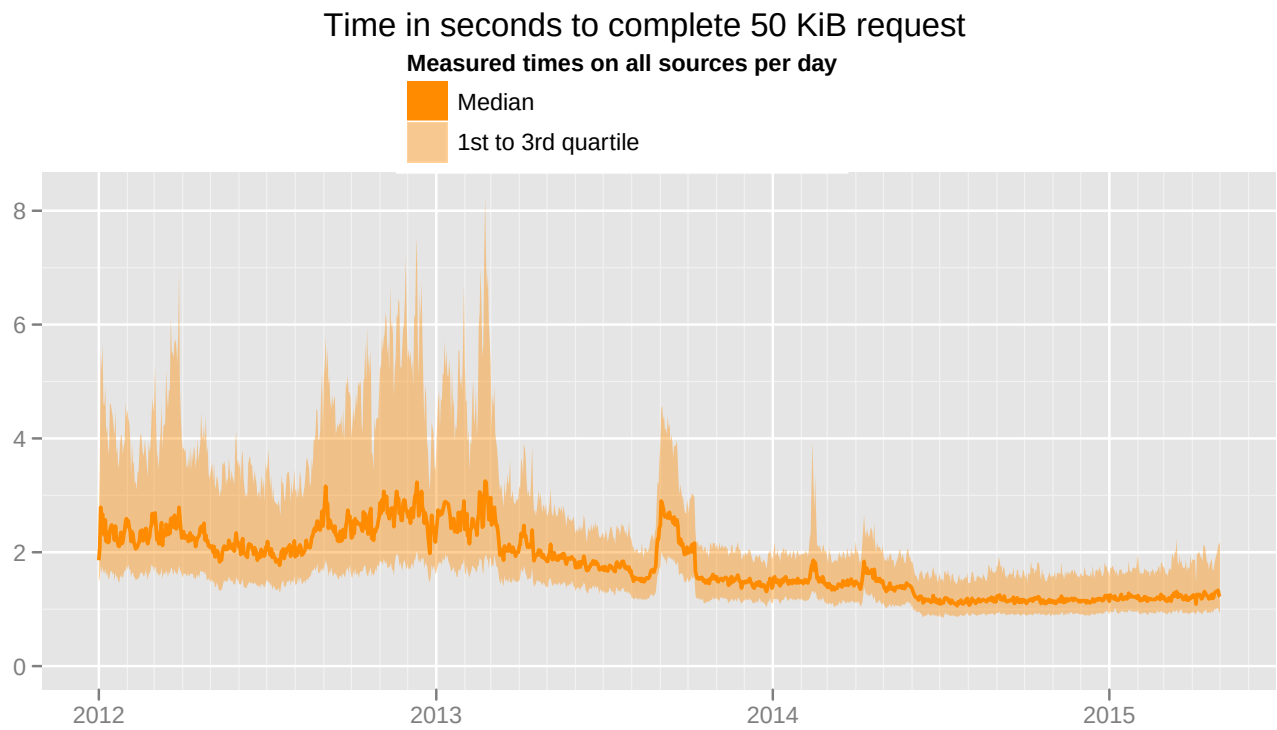
The Tor Project − https://metrics.torproject.org/

Network capacity



The Tor Project − https://metrics.torproject.org/

End-user performance



The Tor Project − https://metrics.torproject.org/

Link protocol

- Pairwise communication between nodes
- . . .client to first-hop
- . . .and relay to relay
- Authenticating one or both
- Uses TLS

Link protocol

- Pairwise communication between nodes
- . . .client to first-hop
- . . .and relay to relay
- Authenticating one or both
- Uses TLS

Link protocol

- Pairwise communication between nodes
- . . .client to first-hop
- . . .and relay to relay
- Authenticating one or both
- Uses TLS

Link protocol

- Pairwise communication between nodes
- . . .client to first-hop
- . . .and relay to relay
- Authenticating one or both
- Uses TLS

Link protocol

- Pairwise communication between nodes
- . . .client to first-hop
- . . .and relay to relay
- Authenticating one or both
- Uses TLS

Circuit protocol

- Setting up a tunnel between a client and an exit relay
- Client sharing a key with each relay
- Uses public-key crypto (DH with RSA or Curve25519)
- This tunnel is called a circuit

Circuit protocol

- Setting up a tunnel between a client and an exit relay
- Client sharing a key with each relay
- Uses public-key crypto (DH with RSA or Curve25519)
- This tunnel is called a circuit

Circuit protocol

- Setting up a tunnel between a client and an exit relay
- Client sharing a key with each relay
- Uses public-key crypto (DH with RSA or Curve25519)
- This tunnel is called a circuit

Circuit protocol

- Setting up a tunnel between a client and an exit relay
- Client sharing a key with each relay
- Uses public-key crypto (DH with RSA or Curve25519)
- This tunnel is called a circuit

Relay protocol

- Allowing a client to communicate with the nodes on a circuit
- Done by exchanging 512 byte cells
- One layer of symmetric crypto (AES128-CTR) for each relay processing the cell
- Cells contain a four byte digest (SHA1)

Relay protocol

- Allowing a client to communicate with the nodes on a circuit
- Done by exchanging 512 byte cells
- One layer of symmetric crypto (AES128-CTR) for each relay processing the cell
- Cells contain a four byte digest (SHA1)

Relay protocol

- Allowing a client to communicate with the nodes on a circuit
- Done by exchanging 512 byte cells
- One layer of symmetric crypto (AES128-CTR) for each relay processing the cell
- Cells contain a four byte digest (SHA1)

Relay protocol

- Allowing a client to communicate with the nodes on a circuit
- Done by exchanging 512 byte cells
- One layer of symmetric crypto (AES128-CTR) for each relay processing the cell
- Cells contain a four byte digest (SHA1)

Stream protocol

- Tunneled over the relay protocol
- Between clients and exit relays for
- . . .opening TCP connections to internet services
- . . .resolving DNS names
- . . .sending and receiving data
- . . .closing connections

Stream protocol

- Tunneled over the relay protocol
- Between clients and exit relays for
- . . .opening TCP connections to internet services
- . . .resolving DNS names
- . . .sending and receiving data
- . . .closing connections

Stream protocol

- Tunneled over the relay protocol
- Between clients and exit relays for
- . . .opening TCP connections to internet services
- . . .resolving DNS names
- . . .sending and receiving data
- . . .closing connections

Stream protocol

- Tunneled over the relay protocol
- Between clients and exit relays for
- . . .opening TCP connections to internet services
- . . .resolving DNS names
- . . .sending and receiving data
- . . .closing connections

Stream protocol

- Tunneled over the relay protocol
- Between clients and exit relays for
- . . .opening TCP connections to internet services
- . . .resolving DNS names
- . . .sending and receiving data
- . . .closing connections

Stream protocol

- Tunneled over the relay protocol
- Between clients and exit relays for
- . . .opening TCP connections to internet services
- . . .resolving DNS names
- . . .sending and receiving data
- . . .closing connections

Directory authorities

- A few semi-trusted relays with an authority identity key
- . . .known by clients (compiled into the program)
- Receive router descriptors from potential relays
- Vote every hour to form a signed consensus document
- . . .describing the current Tor network
- . . .giving various flags to relays, like Exit, Fast, Guard, Stable

Directory authorities

- A few semi-trusted relays with an authority identity key
- . . .known by clients (compiled into the program)
- Receive router descriptors from potential relays
- Vote every hour to form a signed consensus document
- . . .describing the current Tor network
- . . .giving various flags to relays, like Exit, Fast, Guard, Stable

Directory authorities

- A few semi-trusted relays with an authority identity key
- . . .known by clients (compiled into the program)
- Receive router descriptors from potential relays
- Vote every hour to form a signed consensus document
- . . .describing the current Tor network
- . . .giving various flags to relays, like Exit, Fast, Guard, Stable

Directory authorities

- A few semi-trusted relays with an authority identity key
- . . .known by clients (compiled into the program)
- Receive router descriptors from potential relays
- Vote every hour to form a signed consensus document
- . . .describing the current Tor network
- . . .giving various flags to relays, like Exit, Fast, Guard, Stable

Directory authorities

- A few semi-trusted relays with an authority identity key
- . . .known by clients (compiled into the program)
- Receive router descriptors from potential relays
- Vote every hour to form a signed consensus document
- . . .describing the current Tor network
- . . .giving various flags to relays, like Exit, Fast, Guard, Stable

Directory authorities

- A few semi-trusted relays with an authority identity key
- . . .known by clients (compiled into the program)
- Receive router descriptors from potential relays
- Vote every hour to form a signed consensus document
- . . .describing the current Tor network
- . . .giving various flags to relays, like Exit, Fast, Guard, Stable

The consensus

- Clients download the consensus and builds circuits through relays matching their needs
- . . .using relay identity keys from the consensus for the handshake

The consensus

- Clients download the consensus and builds circuits through relays matching their needs
- . . .using relay identity keys from the consensus for the handshake

Bob offers an onion service

- Creates a key pair from which an .onion address is made
- . . .by base32-encoding (parts of) a hash of the public key
- Picks a couple of “introduction points”, relays used as a meeting place
- Advertises the key and the list of introduction points in a distributed hash table (DHT) kept by Tor relays
- Creates circuits to the introduction points and tells them about the service

Bob offers an onion service

- Creates a key pair from which an .onion address is made
- . . .by base32-encoding (parts of) a hash of the public key
- Picks a couple of “introduction points”, relays used as a meeting place
- Advertises the key and the list of introduction points in a distributed hash table (DHT) kept by Tor relays
- Creates circuits to the introduction points and tells them about the service

Bob offers an onion service

- Creates a key pair from which an .onion address is made
- . . .by base32-encoding (parts of) a hash of the public key
- Picks a couple of “introduction points”, relays used as a meeting place
- Advertises the key and the list of introduction points in a distributed hash table (DHT) kept by Tor relays
- Creates circuits to the introduction points and tells them about the service

Bob offers an onion service

- Creates a key pair from which an .onion address is made
- . . .by base32-encoding (parts of) a hash of the public key
- Picks a couple of “introduction points”, relays used as a meeting place
- Advertises the key and the list of introduction points in a distributed hash table (DHT) kept by Tor relays
- Creates circuits to the introduction points and tells them about the service

Bob offers an onion service

- Creates a key pair from which an .onion address is made
- . . .by base32-encoding (parts of) a hash of the public key
- Picks a couple of “introduction points”, relays used as a meeting place
- Advertises the key and the list of introduction points in a distributed hash table (DHT) kept by Tor relays
- Creates circuits to the introduction points and tells them about the service

Alice connects to Bob's service

- Gets the list of Bob's introduction points through the DHT
- Builds a circuit to a "rendezvous point", a Tor relay
- Connects to one of Bob's introduction points and asks it to tell Bob about her rendezvous point
- If Bob likes the idea, he builds a circuit to Alices rendezvous point which joins their circuits together

Alice connects to Bob's service

- Gets the list of Bob's introduction points through the DHT
- Builds a circuit to a "rendezvous point", a Tor relay
- Connects to one of Bob's introduction points and asks it to tell Bob about her rendezvous point
- If Bob likes the idea, he builds a circuit to Alices rendezvous point which joins their circuits together

Alice connects to Bob's service

- Gets the list of Bob's introduction points through the DHT
- Builds a circuit to a "rendezvous point", a Tor relay
- Connects to one of Bob's introduction points and asks it to tell Bob about her rendezvous point
- If Bob likes the idea, he builds a circuit to Alices rendezvous point which joins their circuits together

Alice connects to Bob's service

- Gets the list of Bob's introduction points through the DHT
- Builds a circuit to a "rendezvous point", a Tor relay
- Connects to one of Bob's introduction points and asks it to tell Bob about her rendezvous point
- If Bob likes the idea, he builds a circuit to Alices rendezvous point which joins their circuits together

What is Tor?

How does Tor work?

- Overview
- Circuits and cell encryption
- The trust root
- Onion services

Academic research on Tor

- Circuit setup
- Routing
- Congestion
- Scalability
- Security

Research tools and community research

Tor research

- This section leans heavily on the “Performance and Security Improvements for Tor: A Survey” report by Mashael AlSabah and Ian Goldberg
- Will miss out on “community research” like the current HS work and all the censorship circumvention research
- . . .and various crypto work
- Except we’ll visit these briefly at the end

Tor research

- This section leans heavily on the “Performance and Security Improvements for Tor: A Survey” report by Mashael AlSabah and Ian Goldberg
- Will miss out on “community research” like the current HS work and all the censorship circumvention research
- . . .and various crypto work
- Except we’ll visit these briefly at the end

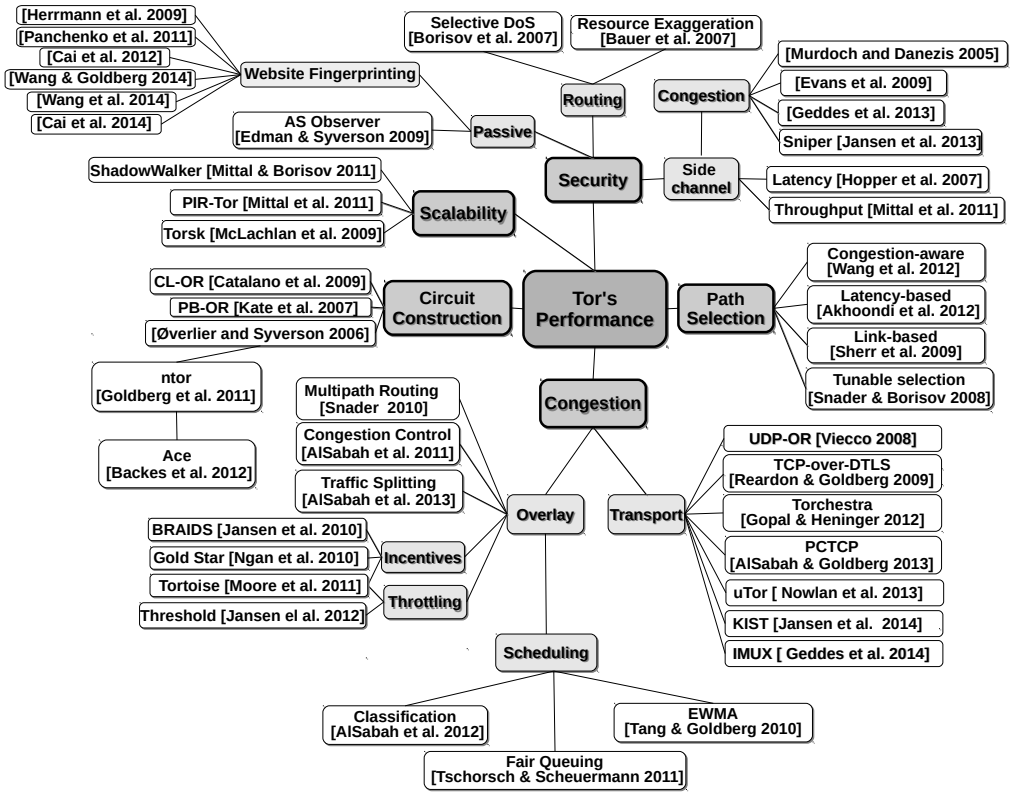
Tor research

- This section leans heavily on the “Performance and Security Improvements for Tor: A Survey” report by Mashael AlSabah and Ian Goldberg
- Will miss out on “community research” like the current HS work and all the censorship circumvention research
- . . .and various crypto work
- Except we’ll visit these briefly at the end

Tor research

- This section leans heavily on the “Performance and Security Improvements for Tor: A Survey” report by Mashael AlSabah and Ian Goldberg
- Will miss out on “community research” like the current HS work and all the censorship circumvention research
- . . .and various crypto work
- Except we’ll visit these briefly at the end

Academic Tor research mindmap



Circuit setup

Constructing circuits is costly both computationally and communication-wise.

- Several improved DH-based key agreement protocols have been presented (Øverlier and Syverson 2007)
- . . .of which one later became “ntor” (Goldberg et al. 2011), implemented and deployed in 2013 (tor-0.2.4)
- . . .which was then improved in “Ace” (Backes et al. 2012)

Circuit setup

Constructing circuits is costly both computationally and communication-wise.

- Several improved DH-based key agreement protocols have been presented (Øverlier and Syverson 2007)
- . . .of which one later became “ntor” (Goldberg et al. 2011), implemented and deployed in 2013 (tor-0.2.4)
- . . .which was then improved in “Ace” (Backes et al. 2012)

Circuit setup

Constructing circuits is costly both computationally and communication-wise.

- Several improved DH-based key agreement protocols have been presented (Øverlier and Syverson 2007)
- . . .of which one later became “ntor” (Goldberg et al. 2011), implemented and deployed in 2013 (tor-0.2.4)
- . . .which was then improved in “Ace” (Backes et al. 2012)

Circuit setup (cont.)

- Pairing-based onion routing (PB-OR, Kate et al. 2007) uses a trusted third party (TTP) to form a non-interactive key agreement protocol, with drawbacks like TTP knowledge of keys, SPOF and higher computational costs for relays
- Certificateless onion routing (Catalano et al. 2009) improves PB-OR by using partial secret keys from the TTP

Circuit setup (cont.)

- Pairing-based onion routing (PB-OR, Kate et al. 2007) uses a trusted third party (TTP) to form a non-interactive key agreement protocol, with drawbacks like TTP knowledge of keys, SPOF and higher computational costs for relays
- Certificateless onion routing (Catalano et al. 2009) improves PB-OR by using partial secret keys from the TTP

Routing

How to build a circuit through the network.

- Tunable selection (Snader and Borisov 2008) is the idea of letting the user indicate her willingness to trade anonymity for performance
- Link-based selection (Sherr et al. 2009) replaces the self-reporting bandwidth plus opportunistic bandwidth measurement with link-based metrics such as latency, jitter and number of traversed AS

Routing

How to build a circuit through the network.

- Tunable selection (Snader and Borisov 2008) is the idea of letting the user indicate her willingness to trade anonymity for performance
- Link-based selection (Sherr et al. 2009) replaces the self-reporting bandwidth plus opportunistic bandwidth measurement with link-based metrics such as latency, jitter and number of traversed AS

Routing (cont.)

- LASTor (Akhoondi et al. 2012) uses geographic location data of relays to minimise latency, as perceived by clients
- In Congestion-aware Path Selection (Wang et al. 2012), clients use observed latency to infer congestion and build paths based on this knowledge
- An evaluation of many path selection proposals can be found in Wacek et al. 2013.

Routing (cont.)

- LASTor (Akhoondi et al. 2012) uses geographic location data of relays to minimise latency, as perceived by clients
- In Congestion-aware Path Selection (Wang et al. 2012), clients use observed latency to infer congestion and build paths based on this knowledge
- An evaluation of many path selection proposals can be found in Wacek et al. 2013.

Routing (cont.)

- LASTor (Akhoondi et al. 2012) uses geographic location data of relays to minimise latency, as perceived by clients
- In Congestion-aware Path Selection (Wang et al. 2012), clients use observed latency to infer congestion and build paths based on this knowledge
- An evaluation of many path selection proposals can be found in Wacek et al. 2013.

Congestion control vs. flow control

- Flow control is about regulating the flow in a network between two endpoints so that the sender doesn't overrun the receiver (e.g. TCP window size)
- The goal of congestion control is to avoid that nodes in the network are overloaded by receiving more than they can get rid of (TCP slow start, fast retransmit and recovery)
- Tor has flow control but no congestion control

Congestion control vs. flow control

- Flow control is about regulating the flow in a network between two endpoints so that the sender doesn't overrun the receiver (e.g. TCP window size)
- The goal of congestion control is to avoid that nodes in the network are overloaded by receiving more than they can get rid of (TCP slow start, fast retransmit and recovery)
- Tor has flow control but no congestion control

Congestion control vs. flow control

- Flow control is about regulating the flow in a network between two endpoints so that the sender doesn't overrun the receiver (e.g. TCP window size)
- The goal of congestion control is to avoid that nodes in the network are overloaded by receiving more than they can get rid of (TCP slow start, fast retransmit and recovery)
- Tor has flow control but no congestion control

Types of congestion

We differentiate between overlay congestion (“application layer”) and transport congestion.

Application layer congestion

- N23 (AlSabah et al. 2011) provides congestion control by giving each circuit a credit balance which is influenced negatively by using downstream bandwidth and positively by recieving credits from downstream routers
- Multipath routing has shown to improve throughput (Snader 2010) where multiple circuits are used in parallell, f.ex. Conflux (AlSabah et al. 2013) which builds two circuits with the same exit and then the exit computes the latency on each path
- Incentive-based schemes is about turning clients into routers: Gold Star, PAR, BRAIDS, LIRA, TEARS and TorCoin (see Jansen 2014)

Application layer congestion

- N23 (AlSabah et al. 2011) provides congestion control by giving each circuit a credit balance which is influenced negatively by using downstream bandwidth and positively by recieving credits from downstream routers
- Multipath routing has shown to improve throughput (Snader 2010) where multiple circuits are used in parallell, f.ex. Conflux (AlSabah et al. 2013) which builds two circuits with the same exit and then the exit computes the latency on each path
- Incentive-based schemes is about turning clients into routers: Gold Star, PAR, BRAIDS, LIRA, TEARS and TorCoin (see Jansen 2014)

Application layer congestion

- N23 (AlSabah et al. 2011) provides congestion control by giving each circuit a credit balance which is influenced negatively by using downstream bandwidth and positively by recieving credits from downstream routers
- Multipath routing has shown to improve throughput (Snader 2010) where multiple circuits are used in parallell, f.ex. Conflux (AlSabah et al. 2013) which builds two circuits with the same exit and then the exit computes the latency on each path
- Incentive-based schemes is about turning clients into routers: Gold Star, PAR, BRAIDS, LIRA, TEARS and TorCoin (see Jansen 2014)

Application layer congestion (cont.)

- Scheduling and prioritisation
  - Scheduling based on number of cells sent per circuit based on a moving average (EWMA) (Tang and Goldberg 2010)
  - Fair queuing between connections on a given router + N23 (Tschorsch and Scheuermann 2011)
  - Hybrid technique with throttled clients as an incentive mechanism for running routers – Tortoise (Moore et al. 2011)
  - Classification of traffic into browsing, streaming and bulk with DiffTor (AlSabah et al. 2012)
  - Throttling client to entry relay connections based on EWMA using the “threshold” algorithm (Jansen et al. 2012)

Application layer congestion (cont.)

- Scheduling and prioritisation
  - Scheduling based on number of cells sent per circuit based on a moving average (EWMA) (Tang and Goldberg 2010)
  - Fair queuing between connections on a given router + N23 (Tschorsch and Scheuermann 2011)
  - Hybrid technique with throttled clients as an incentive mechanism for running routers – Tortoise (Moore et al. 2011)
  - Classification of traffic into browsing, streaming and bulk with DiffTor (AlSabah et al. 2012)
  - Throttling client to entry relay connections based on EWMA using the “threshold” algorithm (Jansen et al. 2012)

Application layer congestion (cont.)

- Scheduling and prioritisation
  - Scheduling based on number of cells sent per circuit based on a moving average (EWMA) (Tang and Goldberg 2010)
  - Fair queuing between connections on a given router + N23 (Tschorsch and Scheuermann 2011)
  - Hybrid technique with throttled clients as an incentive mechanism for running routers – Tortoise (Moore et al. 2011)
  - Classification of traffic into browsing, streaming and bulk with DiffTor (AlSabah et al. 2012)
  - Throttling client to entry relay connections based on EWMA using the “threshold” algorithm (Jansen et al. 2012)

Application layer congestion (cont.)

- Scheduling and prioritisation
  - Scheduling based on number of cells sent per circuit based on a moving average (EWMA) (Tang and Goldberg 2010)
  - Fair queuing between connections on a given router + N23 (Tschorsch and Scheuermann 2011)
  - Hybrid technique with throttled clients as an incentive mechanism for running routers – Tortoise (Moore et al. 2011)
  - Classification of traffic into browsing, streaming and bulk with DiffTor (AlSabah et al. 2012)
  - Throttling client to entry relay connections based on EWMA using the “threshold” algorithm (Jansen et al. 2012)

Application layer congestion (cont.)

- Scheduling and prioritisation
  - Scheduling based on number of cells sent per circuit based on a moving average (EWMA) (Tang and Goldberg 2010)
  - Fair queuing between connections on a given router + N23 (Tschorsch and Scheuermann 2011)
  - Hybrid technique with throttled clients as an incentive mechanism for running routers – Tortoise (Moore et al. 2011)
  - Classification of traffic into browsing, streaming and bulk with DiffTor (AlSabah et al. 2012)
  - Throttling client to entry relay connections based on EWMA using the “threshold” algorithm (Jansen et al. 2012)

Application layer congestion (cont.)

- Scheduling and prioritisation
  - Scheduling based on number of cells sent per circuit based on a moving average (EWMA) (Tang and Goldberg 2010)
  - Fair queuing between connections on a given router + N23 (Tschorsch and Scheuermann 2011)
  - Hybrid technique with throttled clients as an incentive mechanism for running routers – Tortoise (Moore et al. 2011)
  - Classification of traffic into browsing, streaming and bulk with DiffTor (AlSabah et al. 2012)
  - Throttling client to entry relay connections based on EWMA using the “threshold” algorithm (Jansen et al. 2012)

Transport congestion

The problem is the multiple circuits over a single TCP connection – TCP congestion control mechanisms will hit all circuits that share the relay-to-relay connection.

- UDP-OR (Viecco 2008) keeps using TCP from client to entry and from exit to destination but switches to UDP between relays
- TCP-over-DTLS (Reardon and Goldberg 2009) would need a performant userland TCP stack
- Torchestra (Gopal and Heninger 2012) splits traffic between routers onto two separate TCP connections – light and bulk using classification at exits based on EWMA of cells sent
- PCTCP (AlSabah and Goldberg 2013) uses one TCP connection per circuit and IPsec ESP instead of TLS, for bundling them together

Transport congestion

The problem is the multiple circuits over a single TCP connection – TCP congestion control mechanisms will hit all circuits that share the relay-to-relay connection.

- UDP-OR (Viecco 2008) keeps using TCP from client to entry and from exit to destination but switches to UDP between relays
- TCP-over-DTLS (Reardon and Goldberg 2009) would need a performant userland TCP stack
- Torchestra (Gopal and Heninger 2012) splits traffic between routers onto two separate TCP connections – light and bulk using classification at exits based on EWMA of cells sent
- PCTCP (AlSabah and Goldberg 2013) uses one TCP connection per circuit and IPsec ESP instead of TLS, for bundling them together

Transport congestion

The problem is the multiple circuits over a single TCP connection – TCP congestion control mechanisms will hit all circuits that share the relay-to-relay connection.

- UDP-OR (Viecco 2008) keeps using TCP from client to entry and from exit to destination but switches to UDP between relays
- TCP-over-DTLS (Reardon and Goldberg 2009) would need a performant userland TCP stack
- Torchestra (Gopal and Heninger 2012) splits traffic between routers onto two separate TCP connections – light and bulk using classification at exits based on EWMA of cells sent
- PCTCP (AlSabah and Goldberg 2013) uses one TCP connection per circuit and IPsec ESP instead of TLS, for bundling them together

Transport congestion

The problem is the multiple circuits over a single TCP connection – TCP congestion control mechanisms will hit all circuits that share the relay-to-relay connection.

- UDP-OR (Viecco 2008) keeps using TCP from client to entry and from exit to destination but switches to UDP between relays
- TCP-over-DTLS (Reardon and Goldberg 2009) would need a performant userland TCP stack
- Torchestra (Gopal and Heninger 2012) splits traffic between routers onto two separate TCP connections – light and bulk using classification at exits based on EWMA of cells sent
- PCTCP (AlSabah and Goldberg 2013) uses one TCP connection per circuit and IPsec ESP instead of TLS, for bundling them together

Transport congestion (cont.)

- uTor (Nowlan et al. 2013) uses Unordered TCP and Unordered TLS allowing the TCP connection to keep sending data even if a packet is missing for head-of-line
- KIST (Jansen et al. 2014) is a kernel space socket change making it possible for Tor to prioritise when handing data over to the kernel
- IMUX (Geddes et al. 2014) protects against socket exhaustion attacks against routers through a connection manager and scheduler

Transport congestion (cont.)

- uTor (Nowlan et al. 2013) uses Unordered TCP and Unordered TLS allowing the TCP connection to keep sending data even if a packet is missing for head-of-line
- KIST (Jansen et al. 2014) is a kernel space socket change making it possible for Tor to prioritise when handing data over to the kernel
- IMUX (Geddes et al. 2014) protects against socket exhaustion attacks against routers through a connection manager and scheduler

Transport congestion (cont.)

- uTor (Nowlan et al. 2013) uses Unordered TCP and Unordered TLS allowing the TCP connection to keep sending data even if a packet is missing for head-of-line
- KIST (Jansen et al. 2014) is a kernel space socket change making it possible for Tor to prioritise when handing data over to the kernel
- IMUX (Geddes et al. 2014) protects against socket exhaustion attacks against routers through a connection manager and scheduler

Scalability

- Peer-to-peer approaches
  - Try to find ways around the problems of using a DHT for finding peers to build a circuit through, problems including revealing too much during lookup and also sybil attacks
  - ShadowWalker (Mittal and Borisov 2009) uses “shadow nodes” to prevent route capture attacks
  - . . .which later showed to be susceptible to weaknesses (Schuchard et al. 2010)
  - Torsk (McLachlan et al. 2009) uses a “buddy selection protocol” for peer discovery where a Neighbourhood Authority issues certificates to joining routers
  - . . .which was shown (Want et al. 2010) to be vulnerable to attacks where the attacker can learn what routers a client is searching for

Scalability

- Peer-to-peer approaches
  - Try to find ways around the problems of using a DHT for finding peers to build a circuit through, problems including revealing too much during lookup and also sybil attacks
  - ShadowWalker (Mittal and Borisov 2009) uses “shadow nodes” to prevent route capture attacks
  - . . .which later showed to be susceptible to weaknesses (Schuchard et al. 2010)
  - Torsk (McLachlan et al. 2009) uses a “buddy selection protocol” for peer discovery where a Neighbourhood Authority issues certificates to joining routers
  - . . .which was shown (Want et al. 2010) to be vulnerable to attacks where the attacker can learn what routers a client is searching for

Scalability

- Peer-to-peer approaches
  - Try to find ways around the problems of using a DHT for finding peers to build a circuit through, problems including revealing too much during lookup and also sybil attacks
  - ShadowWalker (Mittal and Borisov 2009) uses “shadow nodes” to prevent route capture attacks
  - . . .which later showed to be susceptible to weaknesses (Schuchard et al. 2010)
  - Torsk (McLachlan et al. 2009) uses a “buddy selection protocol” for peer discovery where a Neighbourhood Authority issues certificates to joining routers
  - . . .which was shown (Want et al. 2010) to be vulnerable to attacks where the attacker can learn what routers a client is searching for

Scalability

- Peer-to-peer approaches
  - Try to find ways around the problems of using a DHT for finding peers to build a circuit through, problems including revealing too much during lookup and also sybil attacks
  - ShadowWalker (Mittal and Borisov 2009) uses “shadow nodes” to prevent route capture attacks
  - . . .which later showed to be susceptible to weaknesses (Schuchard et al. 2010)
  - Torsk (McLachlan et al. 2009) uses a “buddy selection protocol” for peer discovery where a Neighbourhood Authority issues certificates to joining routers
  - . . .which was shown (Want et al. 2010) to be vulnerable to attacks where the attacker can learn what routers a client is searching for

Scalability

- Peer-to-peer approaches
  - Try to find ways around the problems of using a DHT for finding peers to build a circuit through, problems including revealing too much during lookup and also sybil attacks
  - ShadowWalker (Mittal and Borisov 2009) uses “shadow nodes” to prevent route capture attacks
  - . . .which later showed to be susceptible to weaknesses (Schuchard et al. 2010)
  - Torsk (McLachlan et al. 2009) uses a “buddy selection protocol” for peer discovery where a Neighbourhood Authority issues certificates to joining routers
  - . . .which was shown (Want et al. 2010) to be vulnerable to attacks where the attacker can learn what routers a client is searching for

Scalability

- Peer-to-peer approaches
  - Try to find ways around the problems of using a DHT for finding peers to build a circuit through, problems including revealing too much during lookup and also sybil attacks
  - ShadowWalker (Mittal and Borisov 2009) uses “shadow nodes” to prevent route capture attacks
  - . . .which later showed to be susceptible to weaknesses (Schuchard et al. 2010)
  - Torsk (McLachlan et al. 2009) uses a “buddy selection protocol” for peer discovery where a Neighbourhood Authority issues certificates to joining routers
  - . . .which was shown (Want et al. 2010) to be vulnerable to attacks where the attacker can learn what routers a client is searching for

Scalability (cont.)

- Using private information retrieval (PIR) techniques, PIR-Tor (Mittal et al. 2011) have clients download a small portion of the network map without revealing which portion

Passive attacks

- AS-level adversaries are breaking an assumption about probability of an attacker seeing both entry and exit relays (Edman and Syverson 2009)
- Website fingerprinting is comparing traffic patterns of encrypted client-to-entry with known patterns of browsers visiting certain websites (numerous papers, but see also Perry 2013)

Passive attacks

- AS-level adversaries are breaking an assumption about probability of an attacker seeing both entry and exit relays (Edman and Syverson 2009)
- Website fingerprinting is comparing traffic patterns of encrypted client-to-entry with known patterns of browsers visiting certain websites (numerous papers, but see also Perry 2013)

Path selection

- Selective denial of service (SDoS, Borisov et al. 2007) is where an attacker denies a client a service in order to decrease client security, f.ex. a malicious middle relay refusing to extend to honest exits
- Low-resource routing attacks (Bauer et al. 2007) fool clients to use a set of relays under the control of the attacker by posing as faster than other relays
- The sniper attack (Jansen et al. 2014) makes clients DoS relays and can be used for things like deanonymisation of onion services

Path selection

- Selective denial of service (SDoS, Borisov et al. 2007) is where an attacker denies a client a service in order to decrease client security, f.ex. a malicious middle relay refusing to extend to honest exits
- Low-resource routing attacks (Bauer et al. 2007) fool clients to use a set of relays under the control of the attacker by posing as faster than other relays
- The sniper attack (Jansen et al. 2014) makes clients DoS relays and can be used for things like deanonymisation of onion services

Path selection

- Selective denial of service (SDoS, Borisov et al. 2007) is where an attacker denies a client a service in order to decrease client security, f.ex. a malicious middle relay refusing to extend to honest exits
- Low-resource routing attacks (Bauer et al. 2007) fool clients to use a set of relays under the control of the attacker by posing as faster than other relays
- The sniper attack (Jansen et al. 2014) makes clients DoS relays and can be used for things like deanonymisation of onion services

Side channel

- Throughput fingerprinting attacks
  - use the distinctive characteristics of the throughput in circuits over a given relay to determine if two circuits share the same path, or sometimes even just the bottleneck relay of a path (Mittal et al. 2011)
  - can also be used for confirming wether a certain traffic flow goes over a given router
  - also also useful for malicious routers to confirm if two streams belong to the same circuit and thus user

Side channel

- Throughput fingerprinting attacks
  - use the distinctive characteristics of the throughput in circuits over a given relay to determine if two circuits share the same path, or sometimes even just the bottleneck relay of a path (Mittal et al. 2011)
  - can also be used for confirming wether a certain traffic flow goes over a given router
  - also also useful for malicious routers to confirm if two streams belong to the same circuit and thus user

Side channel

- Throughput fingerprinting attacks
  - use the distinctive characteristics of the throughput in circuits over a given relay to determine if two circuits share the same path, or sometimes even just the bottleneck relay of a path (Mittal et al. 2011)
  - can also be used for confirming wether a certain traffic flow goes over a given router
  - also also useful for malicious routers to confirm if two streams belong to the same circuit and thus user

Side channel

- Throughput fingerprinting attacks
  - use the distinctive characteristics of the throughput in circuits over a given relay to determine if two circuits share the same path, or sometimes even just the bottleneck relay of a path (Mittal et al. 2011)
  - can also be used for confirming wether a certain traffic flow goes over a given router
  - also also useful for malicious routers to confirm if two streams belong to the same circuit and thus user

Side channel (cont.)

- Congestion attacks
  - could back in the day be mounted by a malicious internet service feeding its client a specialised pattern and then probe Tor relays and distinguish them by the latency caused by the pattern (Murdoch and Danezis 2005)
  - more recently (Evans et al. 2009), a malicious exit could identify the entry guard of a given client by injecting javascript generating a measurable latency which would change if the guard was hit by a “long path circuit” congestion attack

Side channel (cont.)

- Congestion attacks
  - could back in the day be mounted by a malicious internet service feeding its client a specialised pattern and then probe Tor relays and distinguish them by the latency caused by the pattern (Murdoch and Danezis 2005)
  - more recently (Evans et al. 2009), a malicious exit could identify the entry guard of a given client by injecting javascript generating a measurable latency which would change if the guard was hit by a “long path circuit” congestion attack

Side channel (cont.)

- Congestion attacks
  - could back in the day be mounted by a malicious internet service feeding its client a specialised pattern and then probe Tor relays and distinguish them by the latency caused by the pattern (Murdoch and Danezis 2005)
  - more recently (Evans et al. 2009), a malicious exit could identify the entry guard of a given client by injecting javascript generating a measurable latency which would change if the guard was hit by a “long path circuit” congestion attack

Side channel (cont.)

- Network latency attacks include
  - the circuit linkability attack (Hopper et al. 2010) in which two malicious internet services each measure the latency of one connection from a Tor exit relay and determine weather they're from the same client
  - estimating a clients location by using a congestion attack to find the relays used by the client and constructing an identical circuit to infer the latency of the link between the victims client and its first hop (Hopper et al. 2010)

Side channel (cont.)

- Network latency attacks include
  - the circuit linkability attack (Hopper et al. 2010) in which two malicious internet services each measure the latency of one connection from a Tor exit relay and determine weather they're from the same client
  - estimating a clients location by using a congestion attack to find the relays used by the client and constructing an identical circuit to infer the latency of the link between the victims client and its first hop (Hopper et al. 2010)

Side channel (cont.)

- Network latency attacks include
  - the circuit linkability attack (Hopper et al. 2010) in which two malicious internet services each measure the latency of one connection from a Tor exit relay and determine weather they're from the same client
  - estimating a clients location by using a congestion attack to find the relays used by the client and constructing an identical circuit to infer the latency of the link between the victims client and its first hop (Hopper et al. 2010)

What is Tor?

How does Tor work?

- Overview
- Circuits and cell encryption
- The trust root
- Onion services

Academic research on Tor

- Circuit setup
- Routing
- Congestion
- Scalability
- Security

Research tools and community research

The metrics site

- Relays and bridges, with flags and by version and platform
- Network capacity, relay bandwidth by flags, consumed bandwidth
- Advertised bandwidth distribution
- Users by country and top-10 countries by direct connect, bridge connect and censorship events
- Onion service traffic and unique addresses

Simulators and emulators

A number network emulators exist, running the real Tor software, simulating Tor network topologies. Several of them are capable of running large network simulations for measuring performance.

- Shadow
- ExperimenTor
- SNEAC

Chutney is a tool for running a small Tor network on a single host, typically used for testing.

Crypto

Crypto related work include

- Proposal 202 presents two possible new relay encryption protocols for Tor cells
- . . .see also Fu et al. 2009 and the “The 23 Raccoons” thread from 2012 and for more on tagging attacks
- Proposal 220 describes how to migrate server identity keys from RSA1024 to Ed25519

Crypto

Crypto related work include

- Proposal 202 presents two possible new relay encryption protocols for Tor cells
- . . .see also Fu et al. 2009 and the “The 23 Raccoons” thread from 2012 and for more on tagging attacks
- Proposal 220 describes how to migrate server identity keys from RSA1024 to Ed25519

Crypto

Crypto related work include

- Proposal 202 presents two possible new relay encryption protocols for Tor cells
- . . .see also Fu et al. 2009 and the “The 23 Raccoons” thread from 2012 and for more on tagging attacks
- Proposal 220 describes how to migrate server identity keys from RSA1024 to Ed25519

Crypto (cont.)

- Proposal 223 replaces the ntor handshake with Ace for improved performance
- Proposal 224 outlines a revised version of hidden services (see Kadianakis 2013)
- Proposal 228 shows how routers prove ownership of their onion keys

Crypto (cont.)

- Proposal 223 replaces the ntor handshake with Ace for improved performance
- Proposal 224 outlines a revised version of hidden services (see Kadianakis 2013)
- Proposal 228 shows how routers prove ownership of their onion keys

Crypto (cont.)

- Proposal 223 replaces the ntor handshake with Ace for improved performance
- Proposal 224 outlines a revised version of hidden services (see Kadianakis 2013)
- Proposal 228 shows how routers prove ownership of their onion keys

Circumvention research

- Ordinary “bridge relays” can be used to bypass blocking by destination address
- In order to avoid being blocked by DPI devices, “pluggable transports” were introduced in 2011 (tor-0.2.3)
- Pluggable transports transform the Tor traffic flow between a client and its first hop, the bridge

Circumvention research

- Ordinary “bridge relays” can be used to bypass blocking by destination address
- In order to avoid being blocked by DPI devices, “pluggable transports” were introduced in 2011 (tor-0.2.3)
- Pluggable transports transform the Tor traffic flow between a client and its first hop, the bridge

Circumvention research

- Ordinary “bridge relays” can be used to bypass blocking by destination address
- In order to avoid being blocked by DPI devices, “pluggable transports” were introduced in 2011 (tor-0.2.3)
- Pluggable transports transform the Tor traffic flow between a client and its first hop, the bridge

Deployed pluggable transports

- obfs3 – random looking
- flashproxy – browser-based short-lived proxies using WebSockets
- Format-Transforming Encryption – arbitrary format
- ScrambleSuit – random looking and adapting
- meek – “domain fronting” using third-party services (Google App Engine, Amazon CloudFront, MS Azure)
- obfs4 – adapting, server authentication and public key obfuscation

Deployed pluggable transports

- obfs3 – random looking
- flashproxy – browser-based short-lived proxies using WebSockets
- Format-Transforming Encryption – arbitrary format
- ScrambleSuit – random looking and adapting
- meek – “domain fronting” using third-party services (Google App Engine, Amazon CloudFront, MS Azure)
- obfs4 – adapting, server authentication and public key obfuscation

Deployed pluggable transports

- obfs3 – random looking
- flashproxy – browser-based short-lived proxies using WebSockets
- Format-Transforming Encryption – arbitrary format
- ScrambleSuit – random looking and adapting
- meek – “domain fronting” using third-party services (Google App Engine, Amazon CloudFront, MS Azure)
- obfs4 – adapting, server authentication and public key obfuscation

Deployed pluggable transports

- obfs3 – random looking
- flashproxy – browser-based short-lived proxies using WebSockets
- Format-Transforming Encryption – arbitrary format
- ScrambleSuit – random looking and adapting
- meek – “domain fronting” using third-party services (Google App Engine, Amazon CloudFront, MS Azure)
- obfs4 – adapting, server authentication and public key obfuscation

Deployed pluggable transports

- obfs3 – random looking
- flashproxy – browser-based short-lived proxies using WebSockets
- Format-Transforming Encryption – arbitrary format
- ScrambleSuit – random looking and adapting
- meek – “domain fronting” using third-party services (Google App Engine, Amazon CloudFront, MS Azure)
- obfs4 – adapting, server authentication and public key obfuscation

Deployed pluggable transports

- obfs3 – random looking
- flashproxy – browser-based short-lived proxies using WebSockets
- Format-Transforming Encryption – arbitrary format
- ScrambleSuit – random looking and adapting
- meek – “domain fronting” using third-party services (Google App Engine, Amazon CloudFront, MS Azure)
- obfs4 – adapting, server authentication and public key obfuscation

Sources and credit I

- Mindmap of research from Performance and Security Improvements for Tor: A Survey by Mashael AlSabah, Ian Goldberg, ePrint Tech Report 2015/235, March 2015

  https://eprint.iacr.org/2015/235.pdf
- “anonbib” – Selected Papers in Anonymity

  http://freehaven.net/anonbib/
- Graphs from Tor metrics https://metrics.torproject.org/
- A Critique of Website Traffic Fingerprinting Attacks by Mike Perry, 2013

  https://blog.torproject.org/blog/critique-website-traffic-fingerprinting-attacks

Sources and credit II

- Tor incentives research roundup: GoldStar, PAR, BRAIDS, LIRA, TEARS, and TorCoin by Rob G. Jansen, 2014

  https://blog.torproject.org/blog/tor-incentives-research-roundup-goldstar-par-braids-lira-tears-and-torcoin
- Analysis of the Relative Severity of Tagging Attacks by The 23 Raccoons, 2012

  https://lists.torproject.org/pipermail/tor-dev/2012-March/003347.html
- Hidden Services need some love by George Kadianakis, 2013

  https://blog.torproject.org/blog/hidden-services-need-some-love
- Charming figures by Ilja Hallberg

Questions and discussion