Lecture #24: Tor
what to do when secure channels aren’t enough
this week, we’re going to turn to adversaries that are observing data on the network.

Alice

\[ c = \text{encrypt}(k, m) \]
\[ h = \text{MAC}(k, c) \]

\[ \text{MAC}(k, c) == h \] ?
\[ m = \text{decrypt}(k, c) \]
policy: provide **anonymity** (only the client should know that they’re communicating with the server)

threat model: adversary is on the path between the client and the server

**public-key cryptography:** a message to \( x \) is encrypted with \( x \)'s public key; only \( x \)'s secret key can decrypt the message

\[
\begin{align*}
\text{encrypt}(\text{PK}_x, m) &= c \\
\text{decrypt}(\text{SK}_x, c) &= m
\end{align*}
\]

things to avoid

no packet should say “from: A; to: S”

**problem:** packet header exposes to the adversary that A is communicating with S

alice is encrypting data to S using its public key

\[
\begin{align*}
A & \quad \{\text{PK}_A, \text{SK}_A\} \\
S & \quad \{\text{PK}_S, \text{SK}_S\}
\end{align*}
\]
**policy:** provide **anonymity** (only the client should know that they’re communicating with the server)

**threat model:** adversary is on the path between the client and the server

**public-key cryptography:** a message to $x$ is encrypted with $x$’s public key; only $x$’s secret key can decrypt the message

- $\text{encrypt}(PK_x, m) = c$
- $\text{decrypt}(SK_x, c) = m$

**things to avoid**

no packet should say “from: A; to: S”
**policy:** provide **anonymity** (only the client should know that they’re communicating with the server)

**threat model:** adversary is on the path between the client and the server

**public-key cryptography:** a message to $x$ is encrypted with $x$’s public key; only $x$’s secret key can decrypt the message

- $\text{encrypt}(PK_x, m) = c$
- $\text{decrypt}(SK_x, c) = m$

**things to avoid**

no packet should say “from: A; to: S”
**policy:** provide **anonymity** (only the client should know that they’re communicating with the server)

**threat model:** adversary is on the path between the client and the server

**public-key cryptography:** a message to \( x \) is encrypted with \( x \)'s public key; only \( x \)'s secret key can decrypt the message

\[
\text{encrypt}(\text{PK}_x, m) = c \\
\text{decrypt}(\text{SK}_x, c) = m
\]

**things to avoid**

no packet should say “from: A; to: S”
**policy:** provide **anonymity** (only the client should know that they’re communicating with the server)

**threat model:** adversary is on the path between the client and the server

**public-key cryptography:** a message to x is encrypted with x’s public key; only x’s secret key can decrypt the message

\[
\text{encrypt}(PK_x, m) = c \\
\text{decrypt}(SK_x, c) = m
\]

**things to avoid**

👍 no packet should say “from: A; to: S”

no entity in the network should receive a packet from A and send it directly to S

no entity in the network should keep state that links A to S

**problem:** P knows that A is communicating with S

a single proxy alone can be useful for other things; we’ll return to this later in the lecture.
policy: provide **anonymity** (only the client should know that they’re communicating with the server)

threat model: adversary is on the path between the client and the server

**public-key cryptography:** a message to \( x \) is encrypted with \( x \)'s public key; only \( x \)'s secret key can decrypt the message

\[
\begin{align*}
\text{encrypt}(PK_x, m) &= c \\
\text{decrypt}(SK_x, c) &= m
\end{align*}
\]

**things to avoid**

no packet should say “from: A; to: S”

no entity in the network should receive a packet from A and send it directly to S

no entity in the network should keep state that links A to S
policy: provide **anonymity** (only the client should know that they’re communicating with the server)

threat model: adversary is on the path between the client and the server

**public-key cryptography:** a message to \(x\) is encrypted with \(x\)'s public key; only \(x\)'s secret key can decrypt the message

\[
\text{encrypt}(\text{PK}_x, \text{m}) = \text{c} \\
\text{decrypt}(\text{SK}_x, \text{c}) = \text{m}
\]

**things to avoid**

no packet should say “from: A; to: S”

no entity in the network should receive a packet from A and send it directly to S

no entity in the network should keep *state* that links A to S

5 is the (random) “circuit ID”. A proxy may be involved in more than one circuit, so it needs a way to differentiate
**threat model:** adversary is on the path between the client and the server

**policy:** provide **anonymity** (only the client should know that they’re communicating with the server)

**public-key cryptography:** a message to \( x \) is encrypted with \( x \)'s public key; only \( x \)'s secret key can decrypt the message

\[
\text{encrypt}(PK_x, m) = c \\
\text{decrypt}(SK_x, c) = m
\]

**things to avoid**

no packet should say “from: A; to: S”

no entity in the network should receive a packet from A and send it directly to S

no entity in the network should keep state that links A to S
**policy:** provide **anonymity** (only the client should know that they’re communicating with the server)

**threat model:** adversary is on the path between the client and the server

**public-key cryptography:** a message to x is encrypted with x’s public key; only x’s secret key can decrypt the message

- \( \text{encrypt}(\text{PK}_x, m) = c \)
- \( \text{decrypt}(\text{SK}_x, c) = m \)

**things to avoid**

- no packet should say “from: A; to: S”
- no entity in the network should receive a packet from A and send it directly to S
- no entity in the network should keep state that links A to S

5 is the (random) “circuit ID”. A proxy may be involved in more than one circuit, so it needs a way to differentiate

\[
\text{from:} P_3 \quad \text{to:} S
\]

XXXXXX

PKs
**policy:** provide **anonymity** (only the client should know that they’re communicating with the server)

**threat model:** adversary is on the path between the client and the server

---

**public-key cryptography:** a message to \( x \) is encrypted with \( x \)'s public key; only \( x \)'s secret key can decrypt the message

\[
\text{encrypt}(PK_x, m) = c \\
\text{decrypt}(SK_x, c) = m
\]

---

**things to avoid**

- no packet should say “from: A; to: S”
- no entity in the network should receive a packet from A and send it directly to S
- no entity in the network should keep state that links A to S
- data should not appear the same across packets

---

**problem:** an adversary with multiple vantage points can observe the same data traveling from A to S

---

5 is the (random) “circuit ID”. a proxy may be involved in more than one circuit, so it needs a way to differentiate
policy: provide **anonymity** (only the client should know that they’re communicating with the server)

threat model: adversary is on the path between the client and the server

**public-key cryptography:** a message to \( x \) is encrypted with \( x \)'s public key; only \( x \)'s secret key can decrypt the message

\[
\text{encrypt}(PK_x, m) = c \\
\text{decrypt}(SK_x, c) = m
\]

**things to avoid**

no packet should say “from: A; to: S”

no entity in the network should receive a packet from A and send it directly to S

no entity in the network should keep state that links A to S

data should not appear the same across packets

5 is the (random) “circuit ID”. a proxy may be involved in more than one circuit, so it needs a way to differentiate

tor adds layers of encryption that nodes on the path can strip off as the packet traverses the network
**policy:** provide **anonymity** (only the client should know that they’re communicating with the server)

**threat model:** adversary is on the path between the client and the server

**public-key cryptography:** a message to $x$ is encrypted with $x$’s public key; only $x$’s secret key can decrypt the message

$$\text{encrypt}(PK_x, m) = c$$
$$\text{decrypt}(SK_x, c) = m$$

**things to avoid**

no packet should say “from: A; to: S”

no entity in the network should receive a packet from A and send it directly to S

no entity in the network should keep state that links A to S

data should not appear the same across packets

5 is the (random) “circuit ID”. A proxy may be involved in more than one circuit, so it needs a way to differentiate

**tor** adds layers of encryption that nodes on the path can strip off as the packet traverses the network

Katrina LaCurts | lacurts@mit.edu | 6.033 2022
policy: provide **anonymity** (only the client should know that they’re communicating with the server)

threat model: adversary is on the path between the client and the server

**public-key cryptography:** a message to x is encrypted with x’s public key; only x’s secret key can decrypt the message

\[
\text{encrypt}(\text{PK}_x, m) = c \\
\text{decrypt}(\text{SK}_x, c) = m
\]

**things to avoid**

no packet should say “from: A; to: S”

no entity in the network should receive a packet from A and send it directly to S

no entity in the network should keep state that links A to S

data should not appear the same across packets

5 is the (random) “circuit ID”. a proxy may be involved in more than one circuit, so it needs a way to differentiate

\[
\begin{array}{cccc}
5: & P_1 & 5: & A \rightarrow P_2 \\
A & {PK_A, SK_A} & {PK_{P_1}, SK_{P_1}} & {PK_{P_2}, SK_{P_2}} \\
5: & P_1 \rightarrow P_3 & 5: & P_2 \rightarrow S \\
{PK_{P_1}, SK_{P_1}} & {PK_{P_2}, SK_{P_2}} & {PK_{P_3}, SK_{P_3}} & {PK_S, SK_S} \\
\end{array}
\]

tor adds layers of encryption that nodes on the path can strip off as the packet traverses the network
**Policy:** provide **anonymity** (only the client should know that they’re communicating with the server)

**Threat model:** adversary is on the path between the client and the server

**Public-key cryptography:** a message to \( x \) is encrypted with \( x \)'s public key; only \( x \)'s secret key can decrypt the message

\[
\text{encrypt}(\text{PK}_x, m) = c \\
\text{decrypt}(\text{SK}_x, c) = m
\]

**Things to avoid**

- no packet should say “from: A; to: S”
- no entity in the network should receive a packet from A and send it directly to S
- no entity in the network should keep state that links A to S
- data should not appear the same across packets

**Tor** adds layers of encryption that nodes on the path can strip off as the packet traverses the network.
**policy:** provide **anonymity** (only the client should know that they’re communicating with the server)

**threat model:** adversary is on the path between the client and the server

**public-key cryptography:** a message to $x$ is encrypted with $x$’s public key; only $x$’s secret key can decrypt the message

$$\text{encrypt}(PK_x, m) = c$$

$$\text{decrypt}(SK_x, c) = m$$

**things to avoid**

- no packet should say “from: A; to: S”
- no entity in the network should receive a packet from A and send it directly to S
- no entity in the network should keep state that links A to S
- data should not appear the same across packets

5 is the (random) “circuit ID”. A proxy may be involved in more than one circuit, so it needs a way to differentiate.

**tor** adds layers of encryption that nodes on the path can strip off as the packet traverses the network.
**policy:** provide **anonymity** (only the client should know that they’re communicating with the server)

**threat model:** adversary is on the path between the client and the server

**public-key cryptography:** a message to \( x \) is encrypted with \( x \)'s public key; only \( x \)'s secret key can decrypt the message

\[
\text{encrypt}(PK_x, m) = c \\
\text{decrypt}(SK_x, c) = m
\]

**things to avoid**

- no packet should say “from: A; to: S”
- no entity in the network should receive a packet from A and send it directly to S
- no entity in the network should keep state that links A to S
- data should not appear the same across packets

\[
\begin{align*}
5: & \quad P_1 \\
A & \rightarrow P_1 \\
\{PK_A, SK_A\} & \rightarrow \{PK_{P1}, SK_{P1}\}
\end{align*}
\]

\[
\begin{align*}
5: & \quad A \rightarrow P_2 \\
5: & \quad P_1 \rightarrow P_3 \\
5: & \quad P_2 \rightarrow S
\end{align*}
\]

\[
\begin{align*}
\{PK_{P1}, SK_{P1}\} & \rightarrow \{PK_{P2}, SK_{P2}\} \\
\{PK_{P2}, SK_{P3}\} & \rightarrow \{PK_S, SK_S\}
\end{align*}
\]

**tor** adds layers of encryption that nodes on the path can strip off as the packet traverses the network

In practice, **tor** uses public-key cryptography to securely exchange **symmetric keys** between A and each node in the circuit, and the layers of encryption use those symmetric keys; this is what allow traffic to travel in both directions
policy: provide **anonymity** (only the client should know that they’re communicating with the server)

threat model: adversary is on the path between the client and the server

---

Am I totally anonymous if I use Tor?

Generally it is impossible to have perfect anonymity, even with Tor. Though there are some things you can practice to improve your anonymity while using Tor and offline.

Use Tor Browser and software specifically configured for Tor

Tor does not protect all of your computer's Internet traffic when you run it. Tor only protects applications that are properly configured to send their Internet traffic through Tor.

Web browsing:

- Safe: Tor Browser
- Unsafe: Any other browser configured to use Tor as a proxy

File sharing:

- Safe: OnionShare
- Unsafe: BitTorrent over Tor

---

https://support.torproject.org/faq/staying-anonymous/
**policy:** provide **anonymity** (only the client should know that they’re communicating with the server)

**threat model:** adversary is on the path between the client and the server

**public-key cryptography:** a message to $x$ is encrypted with $x$’s public key; only $x$’s secret key can decrypt the message

$\text{encrypt}(PK_x, m) = c$

$\text{decrypt}(SK_x, c) = m$

\[
\begin{align*}
A & \quad \{PK_A, SK_A\} \\
\rightarrow & \quad \rightarrow \\
\{PK_p, SK_p\}
\end{align*}
\]
policy: provide **anonymity** (only the client should know that they’re communicating with the server)

threat model: adversary is on the path between the client and the server

**public-key cryptography:** a message to \( x \) is encrypted with \( x \)'s public key; only \( x \)'s secret key can decrypt the message

\[
\text{encrypt}(PK_x, m) = c \\
\text{decrypt}(SK_x, c) = m
\]

assuming you trust the proxy, this type of service can be useful if you care about confidentiality on a local network

what we’ve shown here is a simplified version of some of the functionality you get when you use a VPN
tor provides some level of anonymity for users, preventing adversaries from linking a sender to its receiver.

There are still ways to attack tor, namely by correlating traffic from various points in the network.

A larger takeaway here is that a secure channel alone only provides confidentiality and integrity of the message data; packet headers can reveal information that may be sensitive in certain contexts.

Much like when we discussed certificate authorities, there are interesting questions about who should run tor. How do we trust that the relay nodes are behaving as they should?

As system designers, it's important to think about what traffic you're sending over the network to clients, and whether that traffic can be sent in a more secure way (and what the trade-offs would be).