# Network Security

## Anonymous Networks & Communications

Isis Agora Lovecruft
Core Developer
The Tor Project

Raboud University, The Netherlands


21 March 2016

# About

- Disclaimer: IANACSNAC **(I Am Neither A Computer Scientist Nor A Cryptographer)**

# About

- Disclaimer: IANACSNAC **(I Am Neither A Computer Scientist Nor A Cryptographer)**
- Studied Physics, specialising in HEP Theory, and English Literature, specialising in Feminist Critical Theory

# About

- Disclaimer: IANACSNAC **(I Am Neither A Computer Scientist Nor A Cryptographer)**
- Studied Physics, specialising in HEP Theory, and English Literature, specialising in Feminist Critical Theory
- Developer for The Tor Project since 2010

# About

- Disclaimer: IANACSNAC **(I Am Neither A Computer Scientist Nor A Cryptographer)**
- Studied Physics, specialising in HEP Theory, and English Literature, specialising in Feminist Critical Theory
- Developer for The Tor Project since 2010
- Previous projects include design and development of the Open Observatory of Network Interference (OONI), an open-source, open-methodology platform for large-scale tests for online censorship and other global network anomalies.

# About

- Disclaimer: IANACSNAC **(I Am Neither A Computer Scientist Nor A Cryptographer)**
- Studied Physics, specialising in HEP Theory, and English Literature, specialising in Feminist Critical Theory
- Developer for The Tor Project since 2010
- Previous projects include design and development of the Open Observatory of Network Interference (OONI), an open-source, open-methodology platform for large-scale tests for online censorship and other global network anomalies.
- Current work includes development on Tor Bridges and Pluggable Transports, development on backend infrastructure within the Tor Network, distribution of Tor Bridges, Tor circuit-level crypto and path manipulation.

# About

- Disclaimer: IANACSNAC **(I Am Neither A Computer Scientist Nor A Cryptographer)**
- Studied Physics, specialising in HEP Theory, and English Literature, specialising in Feminist Critical Theory
- Developer for The Tor Project since 2010
- Previous projects include design and development of the Open Observatory of Network Interference (OONI), an open-source, open-methodology platform for large-scale tests for online censorship and other global network anomalies.
- Current work includes development on Tor Bridges and Pluggable Transports, development on backend infrastructure within the Tor Network, distribution of Tor Bridges, Tor circuit-level crypto and path manipulation.
- More information: https://fyb.patternsinthevoid.net
  Including asciiart and My Little Ponies which bounce around in your browser (if you trust me enough to enable Javascript).
  It also has useful things: technical blogposts, project ideas and updates, my public keys and fingerprints, and, of course, links to all my code.

# Why should you care about privacy?

*"There is an entire genre of YouTube videos devoted to an experience which I'm certain that everyone in this room has had. It entails an individual, who, thinking they're alone, engages in some expressive behaviour – wild singing, girating dancing, some mild sexual activity – only to discover that, in fact, they are not alone, that there's a person watching and lurking, the discovery of which causes them to immediately cease what they're doing in horror. The sense of shame and humiliation in their face is palpable: it's the sense of 'this is something I'm willing to do only if no one else is watching.' This is the crux of the work on which I have been singularly focused for the sixteen months: the question of why privacy matters."*

*—Glenn Greenwald, TED Talk, October 2014*

# But I have nothing to hide...

*"Arguing that you don't care about the right to privacy because you have nothing to hide is no different than saying you don't care about free speech because you have nothing to say"*
*—Ed Snowden, Reddit AMA, 21 May 2015*

# Privacy is necessary for all other rights

## Even if you don't care about your own right to privacy . . .

- . . . it's incredibly antisocial to claim that the right to privacy as a whole should be reliquished, that others should also be expected to have nothing to hide.

# Privacy is necessary for all other rights

## Even if you don't care about your own right to privacy . . .

- . . . it's incredibly antisocial to claim that the right to privacy as a whole should be reliquished, that others should also be expected to have nothing to hide.
- . . . it's saying, "I don't care about *your* right to privacy; I don't care about your rights to freedom of expression and freedom of speech. I don't care about anyone different to me."

# Privacy is necessary for all other rights

## Even if you don't care about your own right to privacy . . .

- . . . it's incredibly antisocial to claim that the right to privacy as a whole should be reliquished, that others should also be expected to have nothing to hide.
- . . . it's saying, "I don't care about *your* right to privacy; I don't care about your rights to freedom of expression and freedom of speech. I don't care about anyone different to me."

> *"Privacy is the right from which all others are derived. Privacy is the fountainhead of individuality. Without privacy, there is only the collective, there is only society, there is only influence from groups, from large powers, that shape every person to bring them into that fold and to make them all alike."*
>
> —*Ed Snowden, CIJ Logan Symposium, 12 March 2016*

# Privacy is necessary because laws and norms change

## So let's assume . . .

- . . . You have nothing to hide.

# Privacy is necessary because laws and norms change

## So let's assume . . .

- . . . You have nothing to hide.
- . . . You have never done anything wrong.

# Privacy is necessary because laws and norms change

## So let's assume . . .

- . . . You have nothing to hide.
- . . . You have never done anything wrong.
- . . . You have never done anything illegal.

# Privacy is necessary because laws and norms change

## So let's assume . . .

- . . . You have nothing to hide.
- . . . You have never done anything wrong.
- . . . You have never done anything illegal.
- . . . You don't care about anyone else's rights.

# Privacy is necessary because laws and norms change

## So let's assume . . .

- . . . You have nothing to hide.
- . . . You have never done anything wrong.
- . . . You have never done anything illegal.
- . . . You don't care about anyone else's rights.
- **You should still care about your own right to privacy, because laws and societal norms change, or may be interpreted differently over time.**

# Privacy is necessary because laws and norms change

## So let's assume . . .

- . . . You have nothing to hide.
- . . . You have never done anything wrong.
- . . . You have never done anything illegal.
- . . . You don't care about anyone else's rights.
- **You should still care about your own right to privacy, because laws and societal norms change, or may be interpreted differently over time.**

In the United States, it's not only unknowable *which* laws might apply: it's actually unknowable *how many* laws might apply at a given place and time – let alone how they may be interpreted by a particular court.

# Privacy is necessary because laws and norms change

*"Estimates of the current size and body of the [United States] federal criminal law vary. It has been reported that the Congressional Research Service cannot even count the current number of federal crimes. And these laws are scattered in over fifty titles of the United States Code, encompassing roughly 27,000 pages. Worse yet, the statutory code sections often incorporate, by reference, the provisions and sanctions of administrative regulations promulgated by various regulatory agencies. Estimates of how many such regulations exist are even less well settled, but the American Bar Association thinks there are nearly 10,000."*

*—James Duane, Regent Law Professor, in a lecture entitled "Don't Talk to Police", May 2012*

# Privacy is necessary for scientific progress

Privacy is essential for continued open progression of scientific understanding and human knowledge.

## From Juice Rap News, "Big Brother is WWWatching You":

*We're told we need safety; which is precious, yes,*
*But can a society that can enforce all it's laws ever progress?*
*Hindsight shows that many figures guilty of "thoughtcrime"*
*Turned out to be luminaries and heroes, before their time.*
*[images of Martin Luther King, Galileo, Huey P. Newton, others]*
*But if the surveillance state had reigned then, in this form and design,*
*Just think of all the progress we may've been denied:*
*Could lobbies for women's or gay rights have appeared and thrived?*
*Would revolutionary ideals have materialised?*
*Would science have pioneered or even survived,*

*If every word had been monitored by thought police and spies?*

https://youtu.be/o66FUc61MvU

# But most web traffic now is encrypted, right?

# But most web traffic now is encrypted, right?

# Actually, no, everything is not encrypted, not really

From the article:

> *"Early last year–before the Snowden revelations–encrypted traffic accounted for 2.29 percent of all peak hour traffic in North America, according to Sandvine's report. Now, it spans 3.8 percent. But that's a small jump compared to other parts of the world. In Europe, encrypted traffic went from 1.47 percent to 6.10 percent, and in Latin America, it increased from 1.8 percent to 10.37 percent."*
>
> *—Klint Finley on wired.com, May 16, 2014*

# . . . update from 2015

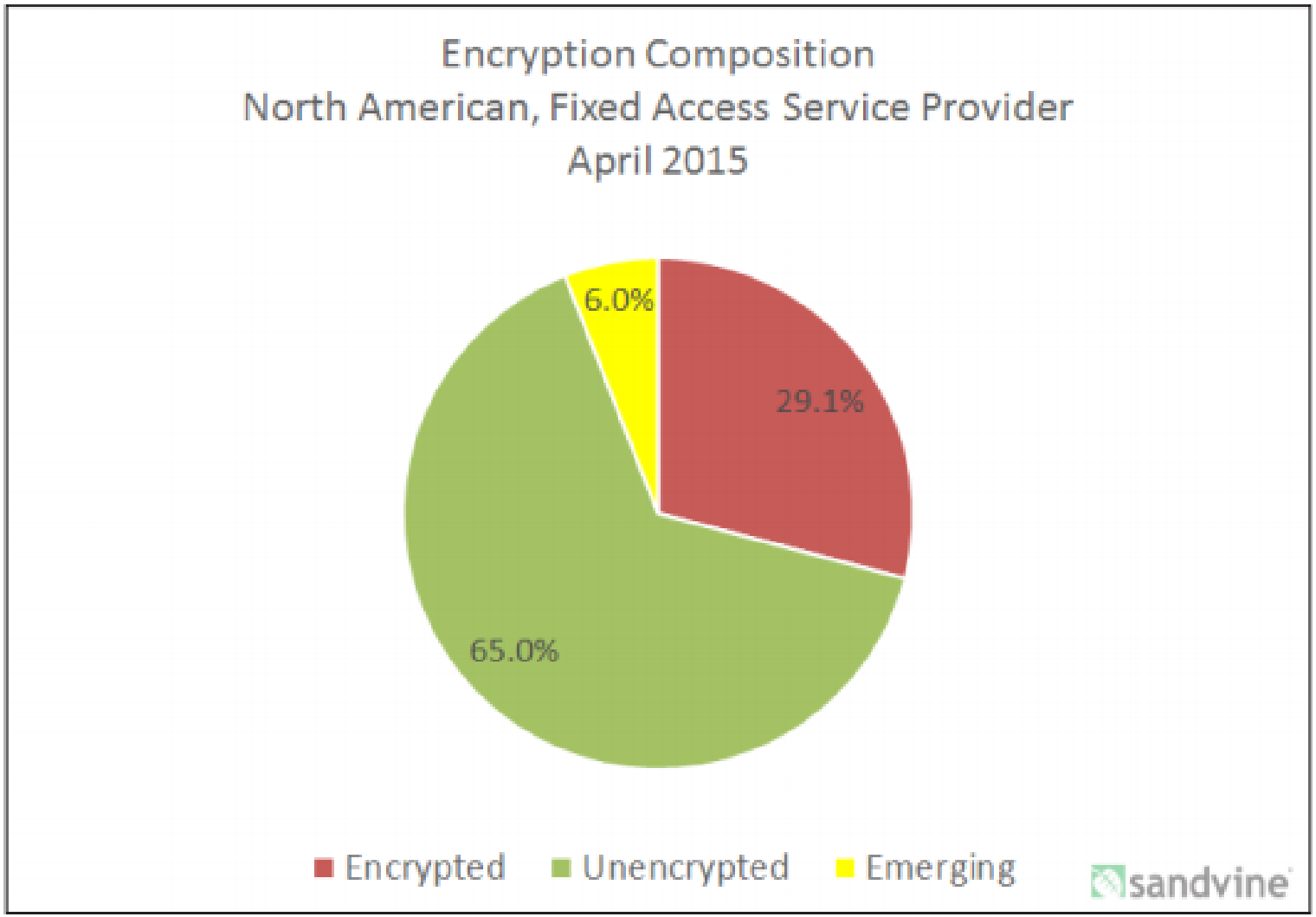



Figure 1 - Encryption Composition - North America, Fixed Access - April 2015

# . . . estimated for 2016

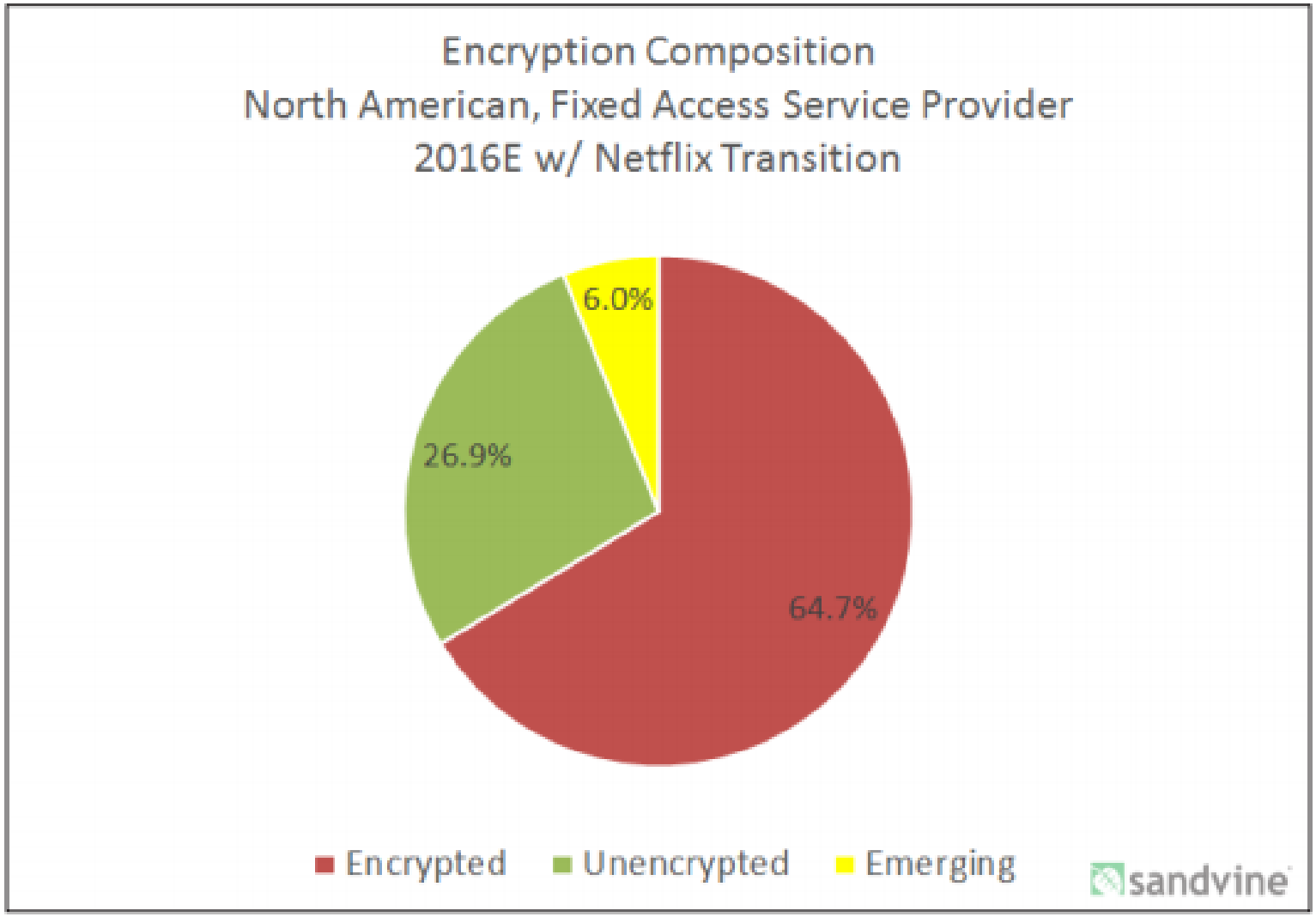



Figure 2 - Encryption Composition - North America, Fixed Access - 2016 Estimate

# Okay, but! if everything were encrypted . . .

## Imagine a world in which . . .

- . . . all Internet traffic is encrypted and authenticated,

# Okay, but! if everything were encrypted . . .

## Imagine a world in which . . .

- . . . all Internet traffic is encrypted and authenticated,
- . . . emails are all signed and encrypted end-to-end,

# Okay, but! if everything were encrypted . . .

## Imagine a world in which . . .

- . . . all Internet traffic is encrypted and authenticated,
- . . . emails are all signed and encrypted end-to-end,
- . . . everybody is using ciphersuites that offer high security,

# Okay, but! if everything were encrypted . . .

## Imagine a world in which . . .

- . . . all Internet traffic is encrypted and authenticated,
- . . . emails are all signed and encrypted end-to-end,
- . . . everybody is using ciphersuites that offer high security,

# Okay, but! if everything were encrypted . . .

## Imagine a world in which . . .

- . . . all Internet traffic is encrypted and authenticated,
- . . . emails are all signed and encrypted end-to-end,
- . . . everybody is using ciphersuites that offer high security,
- . . . crypto implementations are correct, secure, side-channel resistant, formally verified, and expose impossible-to-misuse APIs to applications developers,

# Okay, but! if everything were encrypted . . .

## Imagine a world in which . . .

- . . . all Internet traffic is encrypted and authenticated,
- . . . emails are all signed and encrypted end-to-end,
- . . . everybody is using ciphersuites that offer high security,
- . . . crypto implementations are correct, secure, side-channel resistant, formally verified, and expose impossible-to-misuse APIs to applications developers,
- . . . applied cryptographers have trouble finding jobs.

# Okay, but! if everything were encrypted . . .

## Imagine a world in which . . .

- . . . all Internet traffic is encrypted and authenticated,
- . . . emails are all signed and encrypted end-to-end,
- . . . everybody is using ciphersuites that offer high security,
- . . . crypto implementations are correct, secure, side-channel resistant, formally verified, and expose impossible-to-misuse APIs to applications developers,
- . . . applied cryptographers have trouble finding jobs.

**They still get the "metadata".**

# What even is this “metadata” stuff?

# What even is this “metadata” stuff?

## EU’s Data Retention Directive

(Annulled in 2014, when the European Parliament found blanket data rentention of unsuspicious persons to generally violate the EU Charter of Fundamental Rights.)

# What even is this "metadata" stuff?

## EU's Data Retention Directive

(Annulled in 2014, when the European Parliament found blanket data rentention of unsuspicious persons to generally violate the EU Charter of Fundamental Rights.)

*"Member States shall ensure that the categories of data specified in Article 5 are retained for periods of not less than six months and not more than two years from the date of the communication."*

# What even is this "metadata" stuff?

## EU's Data Retention Directive

(Annulled in 2014, when the European Parliament found blanket data rentention of unsuspicious persons to generally violate the EU Charter of Fundamental Rights.)

*"Member States shall ensure that the categories of data specified in Article 5 are retained for periods of not less than six months and not more than two years from the date of the communication."*

From Article 5:

- data necessary to trace and identify both the source and destination of a communication

# What even is this "metadata" stuff?

## EU's Data Retention Directive

(Annulled in 2014, when the European Parliament found blanket data rentention of unsuspicious persons to generally violate the EU Charter of Fundamental Rights.)

*"Member States shall ensure that the categories of data specified in Article 5 are retained for periods of not less than six months and not more than two years from the date of the communication."*

From Article 5:

- data necessary to trace and identify both the source and destination of a communication
- data necessary to identify the date, time and duration of a communication

# What even is this "metadata" stuff?

## EU's Data Retention Directive

(Annulled in 2014, when the European Parliament found blanket data rentention of unsuspicious persons to generally violate the EU Charter of Fundamental Rights.)

*"Member States shall ensure that the categories of data specified in Article 5 are retained for periods of not less than six months and not more than two years from the date of the communication."*

From Article 5:

- data necessary to trace and identify both the source and destination of a communication
- data necessary to identify the date, time and duration of a communication
- data necessary to identify the type of communication

# What even is this "metadata" stuff?

## EU's Data Retention Directive

(Annulled in 2014, when the European Parliament found blanket data rentention of unsuspicious persons to generally violate the EU Charter of Fundamental Rights.)

*"Member States shall ensure that the categories of data specified in Article 5 are retained for periods of not less than six months and not more than two years from the date of the communication."*

From Article 5:

- data necessary to trace and identify both the source and destination of a communication
- data necessary to identify the date, time and duration of a communication
- data necessary to identify the type of communication
- data necessary to identify users' communication equipment or what purports to be their equipment

# What even is this "metadata" stuff?

## EU's Data Retention Directive

(Annulled in 2014, when the European Parliament found blanket data rentention of unsuspicious persons to generally violate the EU Charter of Fundamental Rights.)

*"Member States shall ensure that the categories of data specified in Article 5 are retained for periods of not less than six months and not more than two years from the date of the communication."*

From Article 5:

- data necessary to trace and identify both the source and destination of a communication
- data necessary to identify the date, time and duration of a communication
- data necessary to identify the type of communication
- data necessary to identify users' communication equipment or what purports to be their equipment
- data necessary to identify the location of mobile communication equipment

# What even is this "metadata" stuff?

## EU's Data Retention Directive

(Annulled in 2014, when the European Parliament found blanket data rentention of unsuspicious persons to generally violate the EU Charter of Fundamental Rights.)

*"Member States shall ensure that the categories of data specified in Article 5 are retained for periods of not less than six months and not more than two years from the date of the communication."*

From Article 5:

- data necessary to trace and identify both the source and destination of a communication
- data necessary to identify the date, time and duration of a communication
- data necessary to identify the type of communication
- data necessary to identify users' communication equipment or what purports to be their equipment
- data necessary to identify the location of mobile communication equipment

**Encrypting and authenticating content doesn't prevent any of this!**

# But "it's just metadata"!

> *"Metadata absolutely tells you everything about somebody's life. If you have enough metadata you don't really need content. . . [It's] sort of embarrassing how predictable we are as human beings."*
>
> —*Stewart Baker, former general counsel of the NSA*

# But "it's just metadata"!

*"Metadata absolutely tells you everything about somebody's life. If you have enough metadata you don't really need content. . . [It's] sort of embarrassing how predictable we are as human beings."*

*—Stewart Baker, former general counsel of the NSA*

*"Metadata is our context. And that can reveal far more about us than the words we speak. Context yields insights into who we are and the implicit, hidden relationships between us."*

*—Matt Blaze on wired.com, June 2013*

# But "it's just metadata"!

*"Metadata absolutely tells you everything about somebody's life. If you have enough metadata you don't really need content. . . [It's] sort of embarrassing how predictable we are as human beings."*

*—Stewart Baker, former general counsel of the NSA*

*"Metadata is our context. And that can reveal far more about us than the words we speak. Context yields insights into who we are and the implicit, hidden relationships between us."*

*—Matt Blaze on wired.com, June 2013*

*"We kill people based on metadata."*

*—Michael Hayden, former director of the NSA and the CIA*

# Is "metadata" all an attacker gets?

- Common assumption: this "metadata" only includes who is talking to whom, or which website is being requested, but we assume "metadata" does include any information about the content.
- Example, interview with Jimmy Wales (Wikipedia founder):

  "***You've said that you're going to start encrypting communications on Wikipedia as a result***...

  *We have done. It's not completely finished yet but the only thing that GCHQ, hopefully, can see is that you're looking at Wikipedia. They can't see which article you're reading. It's not the government's business to know what everybody is reading."*

# Is "metadata" all an attacker gets?

- Common assumption: this "metadata" only includes who is talking to whom, or which website is being requested, but we assume "metadata" does include any information about the content.
- Example, interview with Jimmy Wales (Wikipedia founder):

  "***You've said that you're going to start encrypting communications on Wikipedia as a result***...

  *We have done. It's not completely finished yet but the only thing that GCHQ, hopefully, can see is that you're looking at Wikipedia. They can't see which article you're reading. It's not the government's business to know what everybody is reading."*
- Small experiment: $10$ Wikipedia pages, load one at random through HTTPS
- Attacker sniffs the network, tries to figure out which one

# Is "metadata" all an attacker gets?

- Common assumption: this "metadata" only includes who is talking to whom, or which website is being requested, but we assume "metadata" does include any information about the content.
- Example, interview with Jimmy Wales (Wikipedia founder):

  "***You've said that you're going to start encrypting communications on Wikipedia as a result***...

  *We have done. It's not completely finished yet but the only thing that GCHQ, hopefully, can see is that you're looking at Wikipedia. They can't see which article you're reading. It's not the government's business to know what everybody is reading."*
- Small experiment: $10$ Wikipedia pages, load one at random through HTTPS
- Attacker sniffs the network, tries to figure out which one

# Is "metadata" all an attacker gets?

- Common assumption: this "metadata" only includes who is talking to whom, or which website is being requested, but we assume "metadata" does include any information about the content.
- Example, interview with Jimmy Wales (Wikipedia founder):

  "***You've said that you're going to start encrypting communications on Wikipedia as a result***...

  *We have done. It's not completely finished yet but the only thing that GCHQ, hopefully, can see is that you're looking at Wikipedia. They can't see which article you're reading. It's not the government's business to know what everybody is reading."*
- Small experiment: $10$ Wikipedia pages, load one at random through HTTPS
- Attacker sniffs the network, tries to figure out which one

# Is "metadata" all an attacker gets?

- Common assumption: this "metadata" only includes who is talking to whom, or which website is being requested, but we assume "metadata" does include any information about the content.
- Example, interview with Jimmy Wales (Wikipedia founder):

  "***You've said that you're going to start encrypting communications on Wikipedia as a result***...

  *We have done. It's not completely finished yet but the only thing that GCHQ, hopefully, can see is that you're looking at Wikipedia. They can't see which article you're reading. It's not the government's business to know what everybody is reading."*
- Small experiment: 10 Wikipedia pages, load one at random through HTTPS
- Attacker sniffs the network, tries to figure out which one
- In this (contrived, closed-world) scenario, it's not that hard:
  - 10 webpages have different amount of pictures
  - 10 webpages have different (length of) URLs to pictures
  - Attacker can count bytes of requests to image server

# Is "metadata" all an attacker gets?

- Common assumption: this "metadata" only includes who is talking to whom, or which website is being requested, but we assume "metadata" does include any information about the content.
- Example, interview with Jimmy Wales (Wikipedia founder):

  "***You've said that you're going to start encrypting communications on Wikipedia as a result***...

  *We have done. It's not completely finished yet but the only thing that GCHQ, hopefully, can see is that you're looking at Wikipedia. They can't see which article you're reading. It's not the government's business to know what everybody is reading."*
- Small experiment: $10$ Wikipedia pages, load one at random through HTTPS
- Attacker sniffs the network, tries to figure out which one
- In this (contrived, closed-world) scenario, it's not that hard:
  - $10$ webpages have different amount of pictures
  - $10$ webpages have different (length of) URLs to pictures
  - Attacker can count bytes of requests to image server
  - This is not the only thing an attacker sees: number of requests, timings, delays, responses

# What can we do to protect our privacy?

If encryption alone is not enough, what more can we do to protect our data?

# What can we do to protect our privacy?

If encryption alone is not enough, what more can we do to protect our data?

Report of the Special Rapporteur on the promotion and protection of the right to freedom of opinion and expression:

> *"States should promote strong encryption and anonymity. National laws should recognize that individuals are free to protect the privacy of their digital communications by using encryption technology and tools that allow anonymity online. Legislation and regulations protecting human rights defenders and journalists should also include provisions enabling access and providing support to use the technologies to secure their communications."*
>
> —*David Kaye, Report, May 2015*

# Terminology: Anonymity Set

- **Anonymity Set.** "To enable the anonymity of a subject, there always has to be an appropriate set of subjects with potentially the same attributes. Anonymity is thus defined as the state of being not identiable within a set of subjects, the anonymity set.
  The anonymity set is the set of all possible subjects. With respect to acting entities, the anonymity set consists of the subjects who might cause an action. With respect to addressees, the anonymity set consists of the subjects who might be addressed. Both anonymity sets may be disjoint, be the same, or they may overlap. The anonymity sets may vary over time."

# Terminology: Unlinkability

# Terminology: Unlinkability

- **Absolute Unlinkability.** "Absolute unlinkability ensures that a user may make multiple uses of resources or services without others being able to link these uses together. [...] Unlinkability requires that users and/or subjects are unable to determine whether the same user caused certain specic operations in the system."

# Terminology: Unlinkability

- **Absolute Unlinkability.** "Absolute unlinkability ensures that a user may make multiple uses of resources or services without others being able to link these uses together. [...] Unlinkability requires that users and/or subjects are unable to determine whether the same user caused certain specic operations in the system."
- **Relative Unlinkability.** "Unlinkability of two or more items of interest (e.g., subjects, messages, events, actions, ...) means that within the system (comprising these and possibly other items), from the attacker's perspective, these items of interest are no more and no less related after her observation than they were related concerning her a-priori knowledge."

# Trusted and Semi-Trusted relays

There are two broad categories for designs of anonymous communications networks:

# Trusted and Semi-Trusted relays

There are two broad categories for designs of anonymous communications networks:

- Those that use trusted relays, i.e. the privacy/anonymity guarantees of the system rely on one centralised node.

# Trusted and Semi-Trusted relays

There are two broad categories for designs of anonymous communications networks:

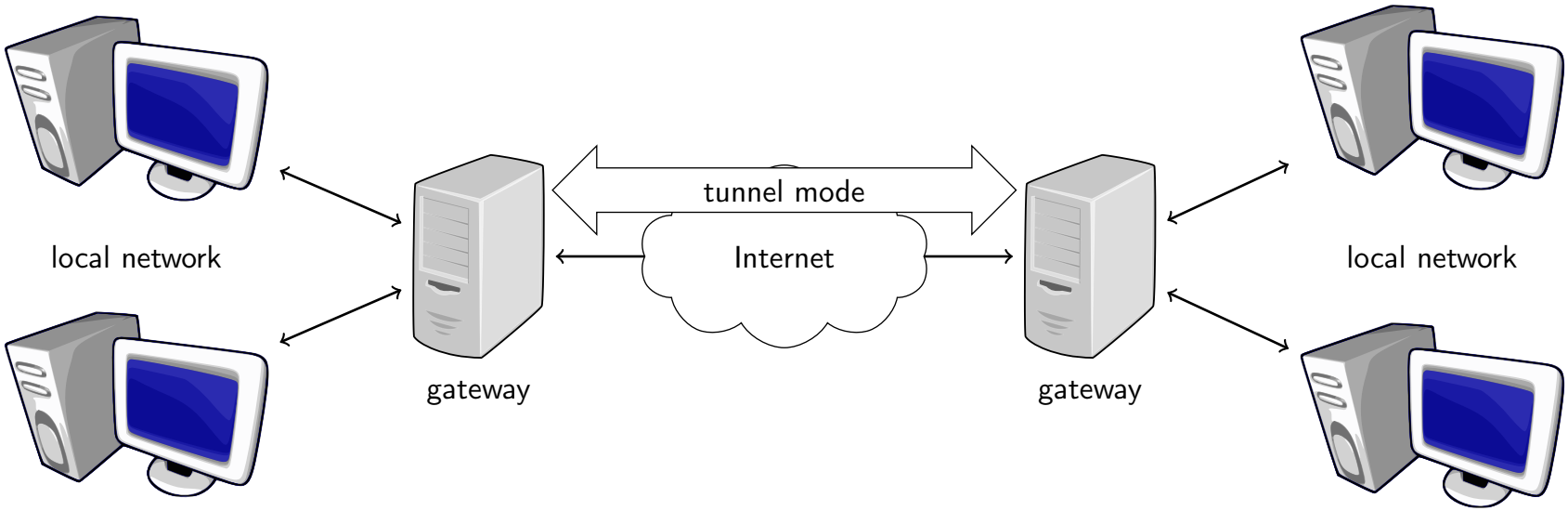
- Those that use trusted relays, i.e. the privacy/anonymity guarantees of the system rely on one centralised node.
- Those that use semi-trusted relays, i.e. compromise of any one node in the network should not degrade the privacy/anonymity of the user.

# Trusted-Relays Systems: Email

Danezis, George, and Claudia Diaz.
A survey of anonymous communication channels.
Technical Report MSR-TR-2008-35, Microsoft Research, 2008.

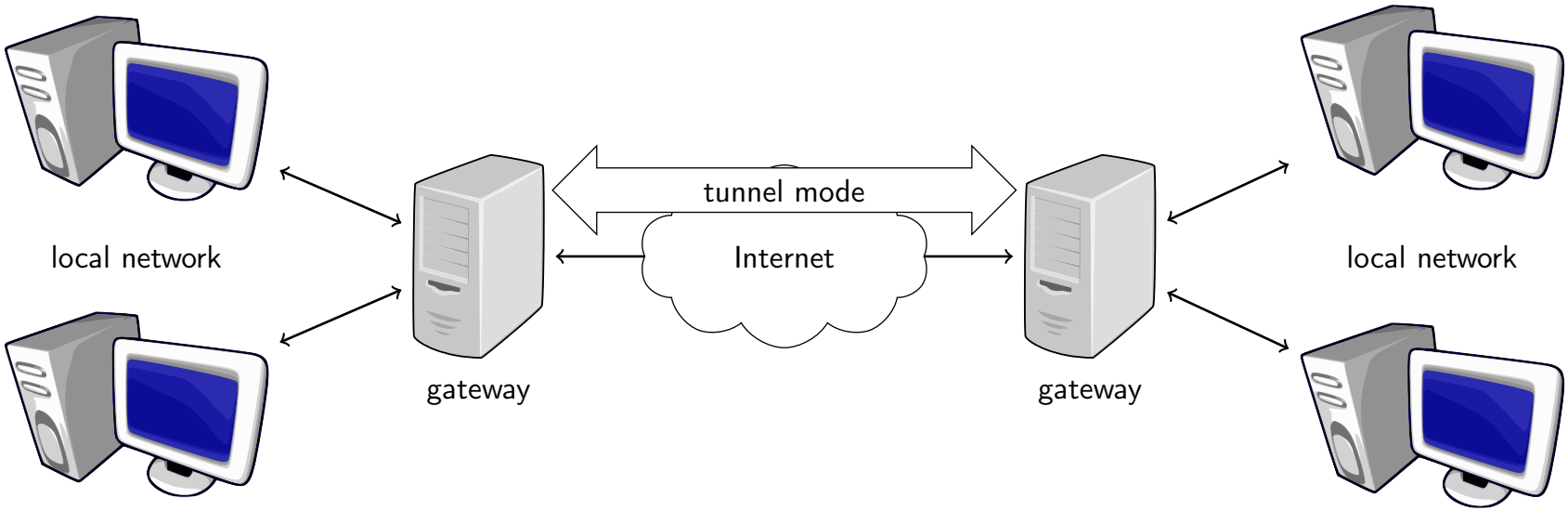
> *"Johan Helsingius started running a trusted mail relay, anon.penet.fi, providing anonymous and pseudonymous email accounts in 1993. The technical principle behind the service was a table of correspondences between real email addresses and pseudonymous addresses, kept by the server. Email to a pseudonym would be forwarded to the real user. Email from a pseudonym was stripped of all identifying information and forwarded to the recipient. While users receiving or sending email to a pseudonym would not be able to nd out the real email address of their anonymous correspondent, it would be trivial for a local passive attacker or the service itself to uncover the correspondence by correlating the timing of incoming and outgoing email traffic."*

# Trusted-Relay Systems: IPsec Tunneling



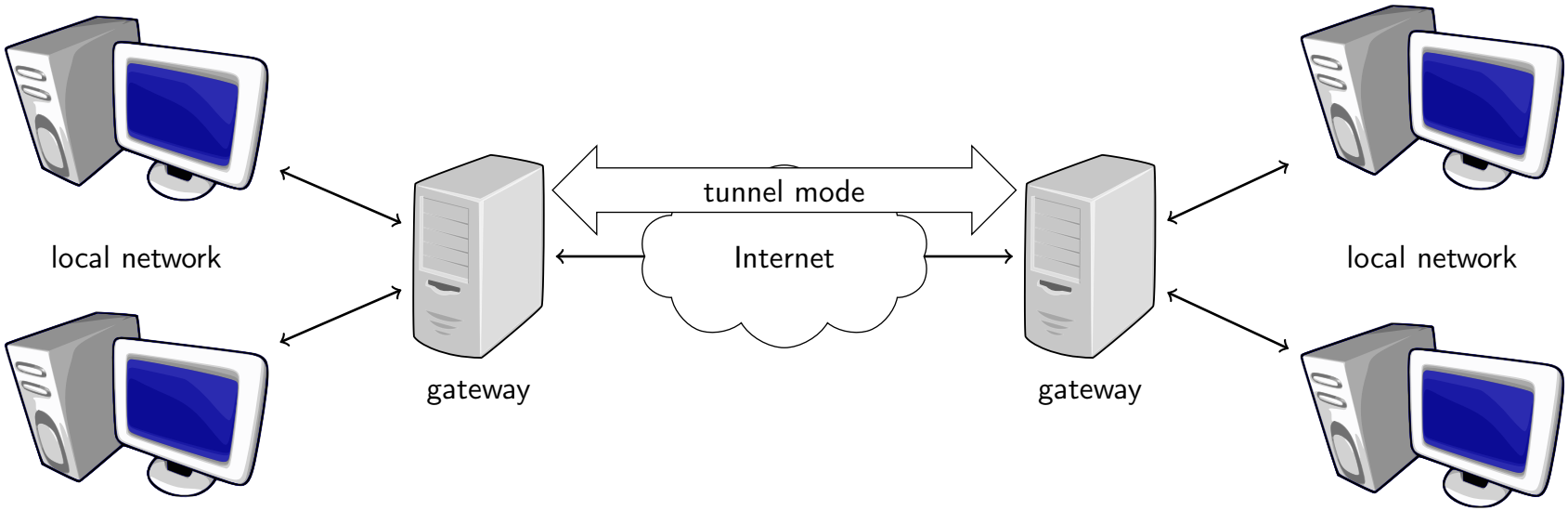

- Everything between the gateways has the *gateways' addresses*
- Nodes behind the gateways are indistinguishable
- RFC 2406 calls this "limited traffic flow confidentiality"

# Trusted-Relay Systems: IPsec Tunneling




- Everything between the gateways has the *gateways' addresses*
- Nodes behind the gateways are indistinguishable
- RFC 2406 calls this "limited traffic flow confidentiality"
- Problem 1: Does not help against state-level attacker who can request gateway's logfiles

# Trusted-Relay Systems: IPsec Tunneling




- Everything between the gateways has the *gateways' addresses*
- Nodes behind the gateways are indistinguishable
- RFC 2406 calls this "limited traffic flow confidentiality"
- Problem 1: Does not help against state-level attacker who can request gateway's logfiles
- Problem 2: Potentially small anonymity set

# Trusted-Relay Systems: Anonymizing proxies

- Somewhat similar idea (without crypto): use a proxy server
- Typically: application-specific proxies (e.g., HTTP proxies) or generic request-based proxies (e.g. SOCKS proxies)
- Requests to websites come from proxy
- All users behind the proxy are indistinguishable
- Various problems:
  1. Single point of failure against state-level attackers

# Trusted-Relay Systems: Anonymizing proxies

- Somewhat similar idea (without crypto): use a proxy server
- Typically: application-specific proxies (e.g., HTTP proxies) or generic request-based proxies (e.g. SOCKS proxies)
- Requests to websites come from proxy
- All users behind the proxy are indistinguishable
- Various problems:
  1. Single point of failure against state-level attackers
  2. Proxy somewhere in the Internet: easy to correlate ingoing/outgoing traffic

# Trusted-Relay Systems: Anonymizing proxies

- Somewhat similar idea (without crypto): use a proxy server
- Typically: application-specific proxies (e.g., HTTP proxies) or generic request-based proxies (e.g. SOCKS proxies)
- Requests to websites come from proxy
- All users behind the proxy are indistinguishable
- Various problems:
  1. Single point of failure against state-level attackers
  2. Proxy somewhere in the Internet: easy to correlate ingoing/outgoing traffic
  3. No crypto protection from the user to the proxy

# Trusted-Relay Systems: Anonymizing proxies

- Somewhat similar idea (without crypto): use a proxy server
- Typically: application-specific proxies (e.g., HTTP proxies) or generic request-based proxies (e.g. SOCKS proxies)
- Requests to websites come from proxy
- All users behind the proxy are indistinguishable
- Various problems:
  1. Single point of failure against state-level attackers
  2. Proxy somewhere in the Internet: easy to correlate ingoing/outgoing traffic
  3. No crypto protection from the user to the proxy
- Can add crypto to the proxy (e.g., OpenVPN Service)
- That still does not solve problems 1 and 2, that is: Virtual Private Networks increase user privacy but do not substantially improve anonymity. Additional problems are added: in the case of VPNs, the VPN fails open, meaning that when the tunnel to the private network breaks down, user traffic goes out in the clear.

# Trusted-Relay Systems

We can see from the previous examples that designs which place ultimate trust in any single node in the network cannot provide any strong guarantees to anonymity, because these single points of failure can be exploited (legally or otherwise) to deanonymise users.

# Trusted-Relay Systems: Mix Networks

Mix networks are routing protocols that create hard-to-trace communications by using a chain of nodes, known as mixes, which receive messages from multiple senders, shuffle them in some manner, and send them to the next destination.

# Trusted-Relay Systems: Mix Networks

Mix networks are routing protocols that create hard-to-trace communications by using a chain of nodes, known as mixes, which receive messages from multiple senders, shuffle them in some manner, and send them to the next destination.

## Chaum's Original Mix Networks

- Idea for anonymous electronic mail by David Chaum, in 1981.

# Trusted-Relay Systems: Mix Networks

Mix networks are routing protocols that create hard-to-trace communications by using a chain of nodes, known as mixes, which receive messages from multiple senders, shuffle them in some manner, and send them to the next destination.

## Chaum's Original Mix Networks

- Idea for anonymous electronic mail by David Chaum, in 1981.
- There should exist some well-known public key for each mix.

# Trusted-Relay Systems: Mix Networks

Mix networks are routing protocols that create hard-to-trace communications by using a chain of nodes, known as mixes, which receive messages from multiple senders, shuffle them in some manner, and send them to the next destination.

## Chaum's Original Mix Networks

- Idea for anonymous electronic mail by David Chaum, in 1981.
- There should exist some well-known public key for each mix.
- Messages are split up into blocks and encrypted to the mix's public key.

# Trusted-Relay Systems: Mix Networks

Mix networks are routing protocols that create hard-to-trace communications by using a chain of nodes, known as mixes, which receive messages from multiple senders, shuffle them in some manner, and send them to the next destination.

## Chaum's Original Mix Networks

- Idea for anonymous electronic mail by David Chaum, in 1981.
- There should exist some well-known public key for each mix.
- Messages are split up into blocks and encrypted to the mix's public key.
- Each mix only knows the nodes immediately before and after it, making the network resistant to malicious mix nodes.

# Trusted-Relay Systems: Mix Networks

Mix networks are routing protocols that create hard-to-trace communications by using a chain of nodes, known as mixes, which receive messages from multiple senders, shuffle them in some manner, and send them to the next destination.

## Chaum's Original Mix Networks

- Idea for anonymous electronic mail by David Chaum, in 1981.
- There should exist some well-known public key for each mix.
- Messages are split up into blocks and encrypted to the mix's public key.
- Each mix only knows the nodes immediately before and after it, making the network resistant to malicious mix nodes.
- Supposed to achieve *bitwise unlinkability* between the source of the message and the destination, making it difficult for an adversary to trace end-to-end communications end-to-end.

# Trusted-Relay Systems: Mix Networks

Mix networks are routing protocols that create hard-to-trace communications by using a chain of nodes, known as mixes, which receive messages from multiple senders, shuffle them in some manner, and send them to the next destination.

## Chaum's Original Mix Networks

- Idea for anonymous electronic mail by David Chaum, in 1981.
- There should exist some well-known public key for each mix.
- Messages are split up into blocks and encrypted to the mix's public key.
- Each mix only knows the nodes immediately before and after it, making the network resistant to malicious mix nodes.
- Supposed to achieve *bitwise unlinkability* between the source of the message and the destination, making it difficult for an adversary to trace end-to-end communications end-to-end.
- Message shuffling in order to achieve unlinkability.

# Trusted-Relay Systems: Mix Networks

## Problems with Chaum's Original Mix Network Scheme

- Pfitzmann and Pfitzmann (1990) demonstrated that Chaum's original work did not acheive the desired unlinkability property.

# Trusted-Relay Systems: Mix Networks

## Problems with Chaum's Original Mix Network Scheme

- Pfitzmann and Pfitzmann (1990) demonstrated that Chaum's original work did not acheive the desired unlinkability property.
- *Tagging attacks* are possible: each encrypted message block, using RSA, is not dependent on those before or after it, and thus can be substituted or reused.

# Trusted-Relay Systems: Mix Networks

## Problems with Chaum's Original Mix Network Scheme

- Pfitzmann and Pfitzmann (1990) demonstrated that Chaum's original work did not acheive the desired unlinkability property.
- *Tagging attacks* are possible: each encrypted message block, using RSA, is not dependent on those before or after it, and thus can be substituted or reused.
- Most of Chaum's work was done in the late 1970s. Unsurprisingly, it used RSA in ways now known to be unsafe, i.e. without padding, applying the modular exponations directly to messages.

# Trusted-Relay Systems: Mix Networks

## Problems with Chaum's Original Mix Network Scheme

- Pfitzmann and Pfitzmann (1990) demonstrated that Chaum's original work did not acheive the desired unlinkability property.
- *Tagging attacks* are possible: each encrypted message block, using RSA, is not dependent on those before or after it, and thus can be substituted or reused.
- Most of Chaum's work was done in the late 1970s. Unsurprisingly, it used RSA in ways now known to be unsafe, i.e. without padding, applying the modular exponations directly to messages.
- Because the RSA operation was applied directly, an adversary could trick a mix into signing a message by applying the decryption operation.

# Trusted-Relay Systems: Mix Networks

## Problems with Chaum's Original Mix Network Scheme

- Pfitzmann and Pfitzmann (1990) demonstrated that Chaum's original work did not acheive the desired unlinkability property.
- *Tagging attacks* are possible: each encrypted message block, using RSA, is not dependent on those before or after it, and thus can be substituted or reused.
- Most of Chaum's work was done in the late 1970s. Unsurprisingly, it used RSA in ways now known to be unsafe, i.e. without padding, applying the modular exponations directly to messages.
- Because the RSA operation was applied directly, an adversary could trick a mix into signing a message by applying the decryption operation.
  - This attack could be further hidden from the mix by applying a signature blinding technique.

# Trusted-Relay Systems: Mix Networks

## Problems with Chaum's Original Mix Network Scheme

- Pfitzmann and Pfitzmann (1990) demonstrated that Chaum's original work did not acheive the desired unlinkability property.
- *Tagging attacks* are possible: each encrypted message block, using RSA, is not dependent on those before or after it, and thus can be substituted or reused.
- Most of Chaum's work was done in the late 1970s. Unsurprisingly, it used RSA in ways now known to be unsafe, i.e. without padding, applying the modular exponations directly to messages.
- Because the RSA operation was applied directly, an adversary could trick a mix into signing a message by applying the decryption operation.
  - This attack could be further hidden from the mix by applying a signature blinding technique.
  - To be fair, Chaum invented RSA blind signing two years later, in 1983.

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Sending a message

- Assume that Alice wants to anonymously send message $M$ to Bob

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Sending a message

- Assume that Alice wants to anonymously send message $M$ to Bob
- Uses intermediate computer called *mix* and public keys
  - $K_B$ of Bob, and
  - $K_M$ of the mix

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Sending a message

- Assume that Alice wants to anonymously send message $M$ to Bob
- Uses intermediate computer called *mix* and public keys
  - $K_B$ of Bob, and
  - $K_M$ of the mix
- Sends to mix: $K_M(R_1, K_B(R_0, M))$ for random $R_0, R_1$

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Sending a message

- Assume that Alice wants to anonymously send message $M$ to Bob
- Uses intermediate computer called *mix* and public keys
  - $K_B$ of Bob, and
  - $K_M$ of the mix
- Sends to mix: $K_M(R_1, K_B(R_0, M))$ for random $R_0, R_1$
- Mix collects many such mails, decrypts to $K_B(R_0, M)$

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Sending a message

- Assume that Alice wants to anonymously send message $M$ to Bob
- Uses intermediate computer called *mix* and public keys
  - $K_B$ of Bob, and
  - $K_M$ of the mix
- Sends to mix: $K_M(R_1, K_B(R_0, M))$ for random $R_0, R_1$
- Mix collects many such mails, decrypts to $K_B(R_0, M)$
- Sends mails in lexicographic order to receivers
- Receiver Bob decrypts and obtains $M$

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Sending a message

- Assume that Alice wants to anonymously send message $M$ to Bob
- Uses intermediate computer called *mix* and public keys
  - $K_B$ of Bob, and
  - $K_M$ of the mix
- Sends to mix: $K_M(R_1, K_B(R_0, M))$ for random $R_0, R_1$
- Mix collects many such mails, decrypts to $K_B(R_0, M)$
- Sends mails in lexicographic order to receivers
- Receiver Bob decrypts and obtains $M$
- Achieves anonymity if encrypted messages are indistinguishable
- Very important: never repeat input and output!
- No protection against *tagging attacks* and *replay attacks*
- Has high communication latency (the mix should wait for enough messages to be within the mixing pool so as to provide some sufficient anonymity set for the clients)

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Return addresses

- Need a way for Bob to reply without revealing Alice's address/identity

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Return addresses

- Need a way for Bob to reply without revealing Alice's address/identity
- Alice includes a return address her message encrypted to Bob:

$$K_M(R_1, A_X), K_X$$

- $R_1, K_X$ are random one-time symmetric keys
- $A_X$ is Alice's real address

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Return addresses

- Need a way for Bob to reply without revealing Alice's address/identity
- Alice includes a return address her message encrypted to Bob:

$$K_M(R_1, A_X), K_X$$

- $R_1, K_X$ are random one-time symmetric keys
- $A_X$ is Alice's real address
- Bob can send response $M$ as

$$K_M(R_1, A_X), K_X(R_0, M)$$

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Return addresses

- Need a way for Bob to reply without revealing Alice's address/identity
- Alice includes a return address her message encrypted to Bob:

$$K_M(R_1, A_X), K_X$$

- $R_1, K_X$ are random one-time symmetric keys
- $A_X$ is Alice's real address
- Bob can send response $M$ as

$$K_M(R_1, A_X), K_X(R_0, M)$$

- Mix receives and recovers $R_1, A_X$, sends to Alice

$$R_1(K_X(R_0, M))$$

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Return addresses

- Need a way for Bob to reply without revealing Alice's address/identity
- Alice includes a return address her message encrypted to Bob:

$$K_M(R_1, A_X), K_X$$

- $R_1, K_X$ are random one-time symmetric keys
- $A_X$ is Alice's real address
- Bob can send response $M$ as

$$K_M(R_1, A_X), K_X(R_0, M)$$

- Mix receives and recovers $R_1, A_X$, sends to Alice

$$R_1(K_X(R_0, M))$$

- Only Alice can decrypt, because only she knows both $K_X$ and $R_1$

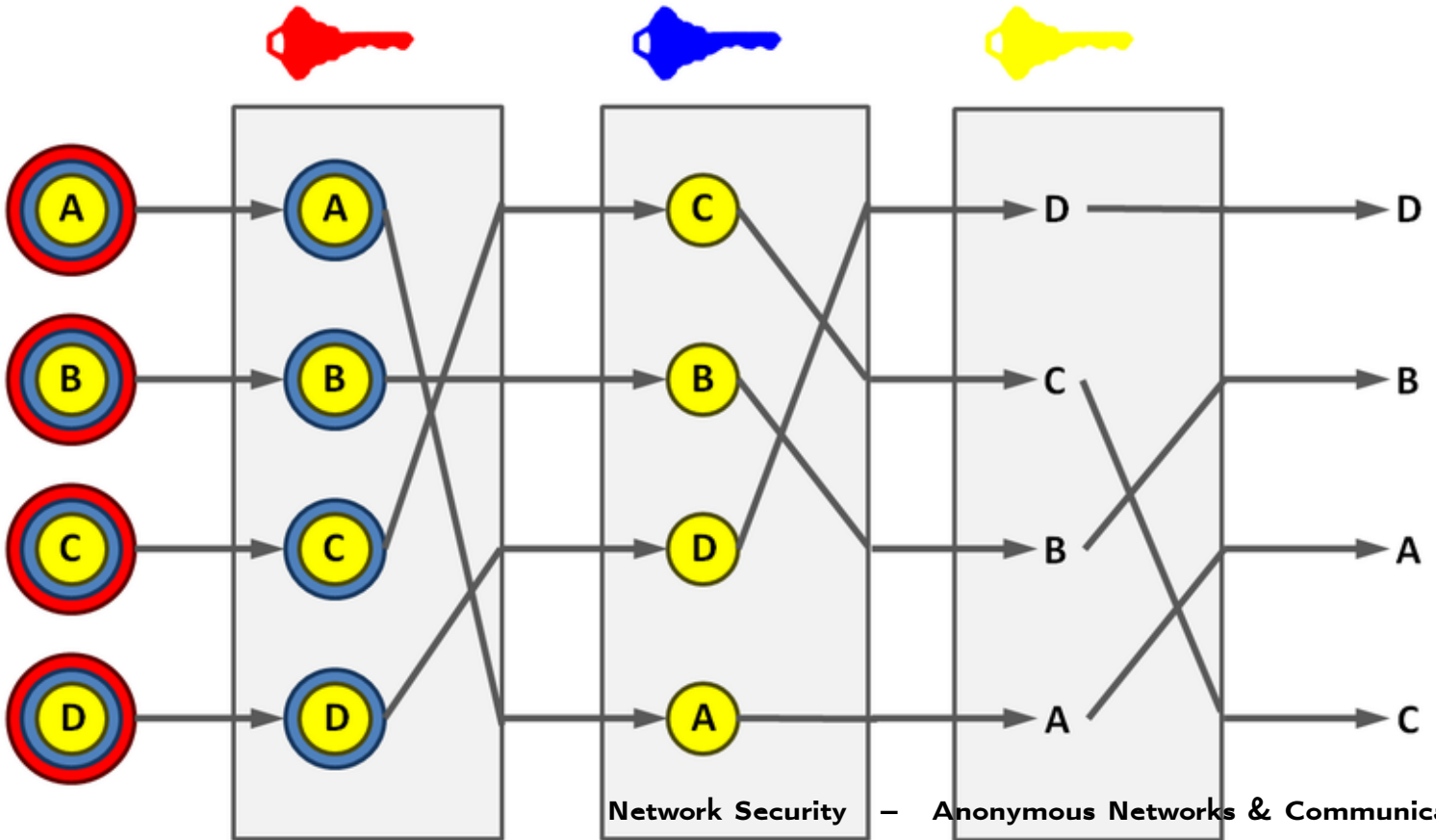
# Mix Network Designs: Cascading Mixes

Chaum noted that relying on only one mix is not resilient against malicious nodes, so the function of mixing should distributed. Mixes can be chained to ensure that, even if just one of them remains honest, some anonymity is provided.

# Mix Network Designs: Cascading Mixes

Chaum noted that relying on only one mix is not resilient against malicious nodes, so the function of mixing should distributed. Mixes can be chained to ensure that, even if just one of them remains honest, some anonymity is provided.

First proposed way to chain mixes together is called *cascade mixing*, and uses all nodes in the network, in a specific order:



# Mix Network Designs: Mix Networks

The second way is to allow users to arbitrarily select which mixes their message will pass through, and is what is now generally referred to as a *mix network*.

# Mix Network Designs: Mix Networks

The second way is to allow users to arbitrarily select which mixes their message will pass through, and is what is now generally referred to as a *mix network*.

## This design has some problems:

- Berthold, Pfitzmann, and Standtke (2000) argue that mix networks do not offer some properties that cascades offer.

# Mix Network Designs: Mix Networks

The second way is to allow users to arbitrarily select which mixes their message will pass through, and is what is now generally referred to as a *mix network*.

## This design has some problems:

- Berthold, Pfitzmann, and Standtke (2000) argue that mix networks do not offer some properties that cascades offer.
- They illustrate a number of attacks to show that, if *only one* mix is honest in the network, the anonymity of the messages going through it can be compromised.

# Mix Network Designs: Mix Networks

The second way is to allow users to arbitrarily select which mixes their message will pass through, and is what is now generally referred to as a *mix network*.

## This design has some problems:

- Berthold, Pfitzmann, and Standtke (2000) argue that mix networks do not offer some properties that cascades offer.
- They illustrate a number of attacks to show that, if *only one* mix is honest in the network, the anonymity of the messages going through it can be compromised.
- These attacks rely on compromised mixes which exploit some knowledge of their position in the chain ...

# Mix Network Designs: Mix Networks

The second way is to allow users to arbitrarily select which mixes their message will pass through, and is what is now generally referred to as a *mix network*.

## This design has some problems:

- Berthold, Pfitzmann, and Standtke (2000) argue that mix networks do not offer some properties that cascades offer.
- They illustrate a number of attacks to show that, if *only one* mix is honest in the network, the anonymity of the messages going through it can be compromised.
- These attacks rely on compromised mixes which exploit some knowledge of their position in the chain ...
- ... or multiple messages using the same sequence of mixes through the network.

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Mixmaster

- Mixmaster was created in 1995 and has some differences to previous Type II anonymous remailers.

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Mixmaster

- Mixmaster was created in 1995 and has some differences to previous Type II anonymous remailers.
- Supports only *forward path*, i.e. sender, anonymity.

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Mixmaster

- Mixmaster was created in 1995 and has some differences to previous Type II anonymous remailers.
- Supports only *forward path*, i.e. sender, anonymity.
- Messages are bitwise unlinkable via hybrid RSA and EDE 3DES.

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Mixmaster

- Mixmaster was created in 1995 and has some differences to previous Type II anonymous remailers.
- Supports only *forward path*, i.e. sender, anonymity.
- Messages are bitwise unlinkable via hybrid RSA and EDE 3DES.
- Messages can be divided in smaller chunks and sent via independent paths. If all parts end up at a common mix, then reconstruction happens transparently.

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Mixmaster

- Mixmaster was created in 1995 and has some differences to previous Type II anonymous remailers.
- Supports only *forward path*, i.e. sender, anonymity.
- Messages are bitwise unlinkable via hybrid RSA and EDE 3DES.
- Messages can be divided in smaller chunks and sent via independent paths. If all parts end up at a common mix, then reconstruction happens transparently.
- v2: The integrity of the RSA-encrypted header is protected by a hash, making tagging attacks on the header impossible.

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Mixmaster

- Mixmaster was created in 1995 and has some differences to previous Type II anonymous remailers.
- Supports only *forward path*, i.e. sender, anonymity.
- Messages are bitwise unlinkable via hybrid RSA and EDE 3DES.
- Messages can be divided in smaller chunks and sent via independent paths. If all parts end up at a common mix, then reconstruction happens transparently.
- v2: The integrity of the RSA-encrypted header is protected by a hash, making tagging attacks on the header impossible.
- v3: The appended noise is deterministically generated using a secret shared between the mix and the sender of the message, and this is included in the header.

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Mixmaster

- Mixmaster was created in 1995 and has some differences to previous Type II anonymous remailers.
- Supports only *forward path*, i.e. sender, anonymity.
- Messages are bitwise unlinkable via hybrid RSA and EDE 3DES.
- Messages can be divided in smaller chunks and sent via independent paths. If all parts end up at a common mix, then reconstruction happens transparently.
- v2: The integrity of the RSA-encrypted header is protected by a hash, making tagging attacks on the header impossible.
- v3: The appended noise is deterministically generated using a secret shared between the mix and the sender of the message, and this is included in the header.
  - Allows an additional header containing a hash of the entire message.

# Trusted-Relay Systems: Mix Networks

## Type II Anonymous Remailers: Mixmaster

- Mixmaster was created in 1995 and has some differences to previous Type II anonymous remailers.
- Supports only *forward path*, i.e. sender, anonymity.
- Messages are bitwise unlinkable via hybrid RSA and EDE 3DES.
- Messages can be divided in smaller chunks and sent via independent paths. If all parts end up at a common mix, then reconstruction happens transparently.
- v2: The integrity of the RSA-encrypted header is protected by a hash, making tagging attacks on the header impossible.
- v3: The appended noise is deterministically generated using a secret shared between the mix and the sender of the message, and this is included in the header.
  - Allows an additional header containing a hash of the entire message.
  - Makes replies impossible to construct: the body of the reply would not be known to the creator of the anonymous address block, so it isn't possible to compute in the hash.

# Mix Nets vs. Anonymizing proxies

## Mix Nets

+ No single point of failure (with cascading)
+ Inbound/output-traffic analysis does not deanonymise
+ Generally good anonymity guarantees

# Mix Nets vs. Anonymizing proxies

## Mix Nets

+ No single point of failure (with cascading)
+ Inbound/output-traffic analysis does not deanonymise
+ Generally good anonymity guarantees
− Slow public-key cryptography (at least in vanilla mix nets)
− High latency

# Mix Nets vs. Anonymizing proxies

## Mix Nets

+ No single point of failure (with cascading)
+ Inbound/output-traffic analysis does not deanonymise
+ Generally good anonymity guarantees
– Slow public-key cryptography (at least in vanilla mix nets)
– High latency

## Anon. Proxies

+ Low latency
+ No overhead from slow crypto

# Mix Nets vs. Anonymizing proxies

## Mix Nets

+ No single point of failure (with cascading)
+ Inbound/output-traffic analysis does not deanonymise
+ Generally good anonymity guarantees
– Slow public-key cryptography (at least in vanilla mix nets)
– High latency

## Anon. Proxies

+ Low latency
+ No overhead from slow crypto
– Single point of failure
– Inbound/output-traffic analysis deanonymises
– No strong anonymity guarantees

# Mix Nets vs. Anonymizing proxies

## Mix Nets

+ No single point of failure (with cascading)
+ Inbound/output-traffic analysis does not deanonymise
+ Generally good anonymity guarantees
– Slow public-key cryptography (at least in vanilla mix nets)
– High latency

## Anon. Proxies

+ Low latency
+ No overhead from slow crypto
– Single point of failure
– Inbound/output-traffic analysis deanonymises
– No strong anonymity guarantees

**Idea of Onion Routing: Combine advantages:**

- Use cascade of "proxies", called *Tor relays* or *Tor nodes*
- Use asymmetric crypto for establishing an authenticated and encrypted channel, then use fast symmetric crypto.

# Onion Routing and Tor

- Assume user has public keys for all relays.

# Onion Routing and Tor

- Assume user has public keys for all relays.
- Use relay public keys to setup an authenticated and encrypted channel, which is used to establish symmetric keypairs for the *forward* and *reverse paths*:
  - Entry relay $R_1$ (keys $KB_{R_1}$, $KF_{R_1}$)
  - Middle relay $R_2$ (keys $KB_{R_2}$, $KF_{R_2}$)
  - Exit relay $R_3$ (keys $KB_{R_3}$, $KF_{R_3}$)

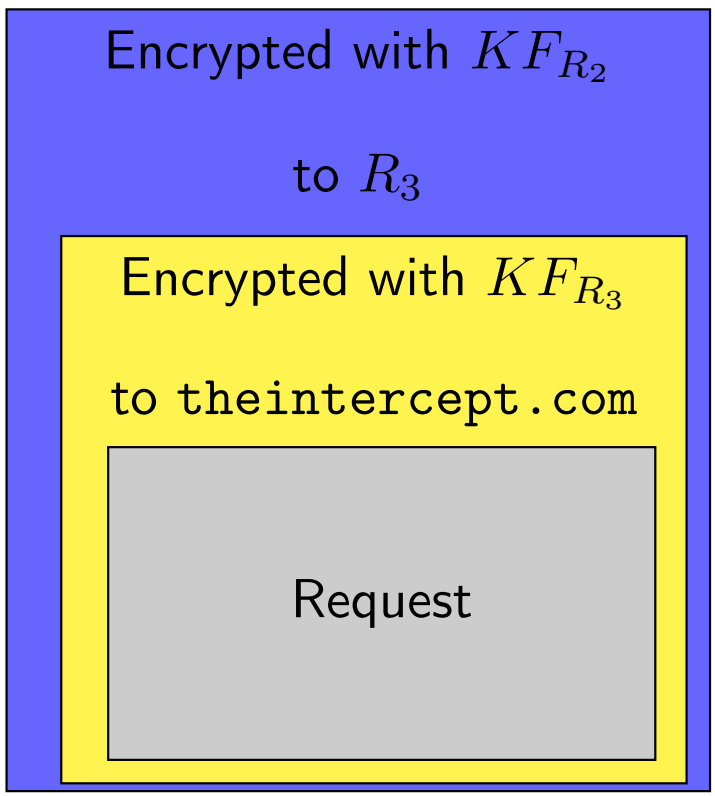
# Onion Routing and Tor

- Assume user has public keys for all relays.
- Use relay public keys to setup an authenticated and encrypted channel, which is used to establish symmetric keypairs for the *forward* and *reverse paths*:
  - Entry relay $R_1$ (keys $KB_{R_1}$, $KF_{R_1}$)
  - Middle relay $R_2$ (keys $KB_{R_2}$, $KF_{R_2}$)
  - Exit relay $R_3$ (keys $KB_{R_3}$, $KF_{R_3}$)
- Wants to anonymously send request to `theintercept.com`

# Onion Routing and Tor

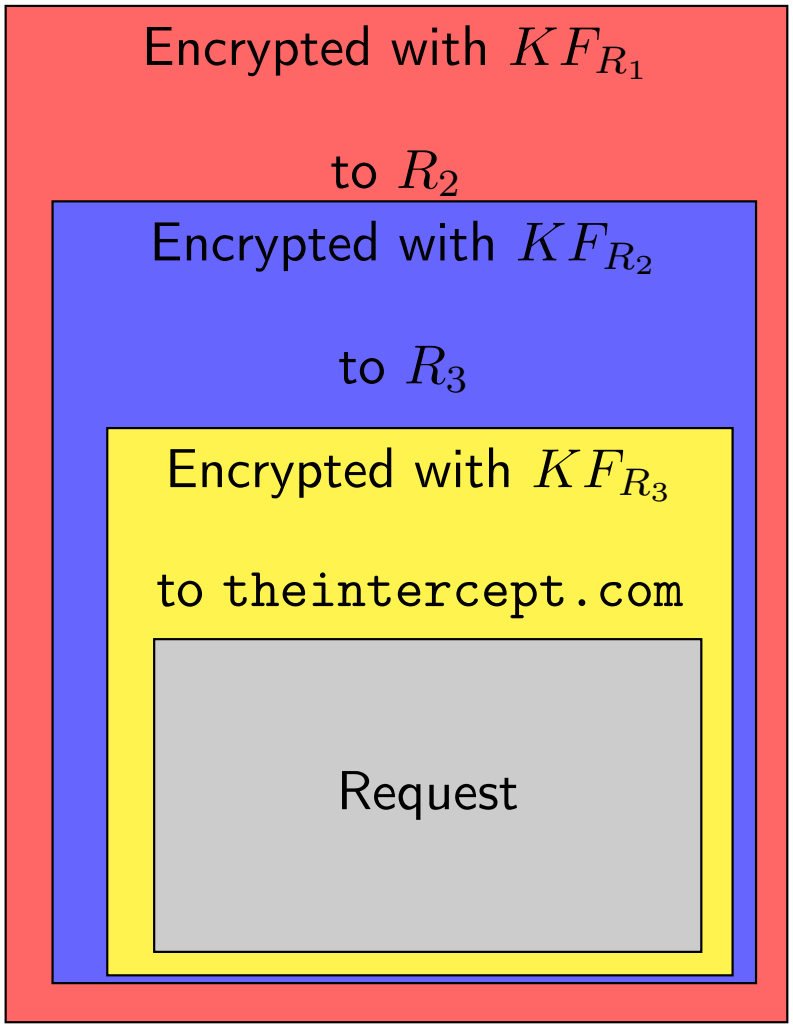

- Assume user has public keys for all relays.
- Use relay public keys to setup an authenticated and encrypted channel, which is used to establish symmetric keypairs for the *forward* and *reverse paths*:
  - Entry relay $R_1$ (keys $KB_{R_1}$, $KF_{R_1}$)
  - Middle relay $R_2$ (keys $KB_{R_2}$, $KF_{R_2}$)
  - Exit relay $R_3$ (keys $KB_{R_3}$, $KF_{R_3}$)
- Wants to anonymously send request to `theintercept.com`
- Prepares packet as follows:
  - Write dest. `theintercept.com`, encrypt with $KF_{R_3}$

# Onion Routing and Tor



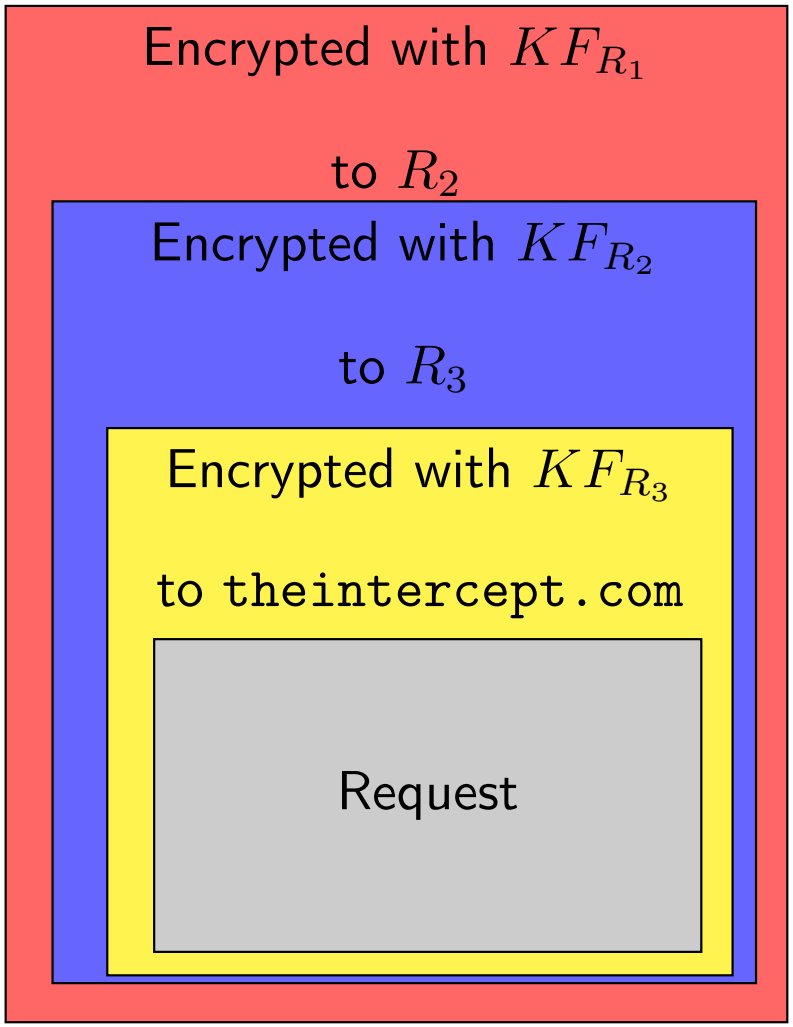

- Assume user has public keys for all relays.
- Use relay public keys to setup an authenticated and encrypted channel, which is used to establish symmetric keypairs for the *forward* and *reverse paths*:
  - Entry relay $R_1$ (keys $KB_{R_1}$, $KF_{R_1}$)
  - Middle relay $R_2$ (keys $KB_{R_2}$, $KF_{R_2}$)
  - Exit relay $R_3$ (keys $KB_{R_3}$, $KF_{R_3}$)
- Wants to anonymously send request to `theintercept.com`
- Prepares packet as follows:
  - Write dest. `theintercept.com`, encrypt with $KF_{R_3}$
  - Write dest. $R_3$ encrypt with $KF_{R_2}$

# Onion Routing and Tor



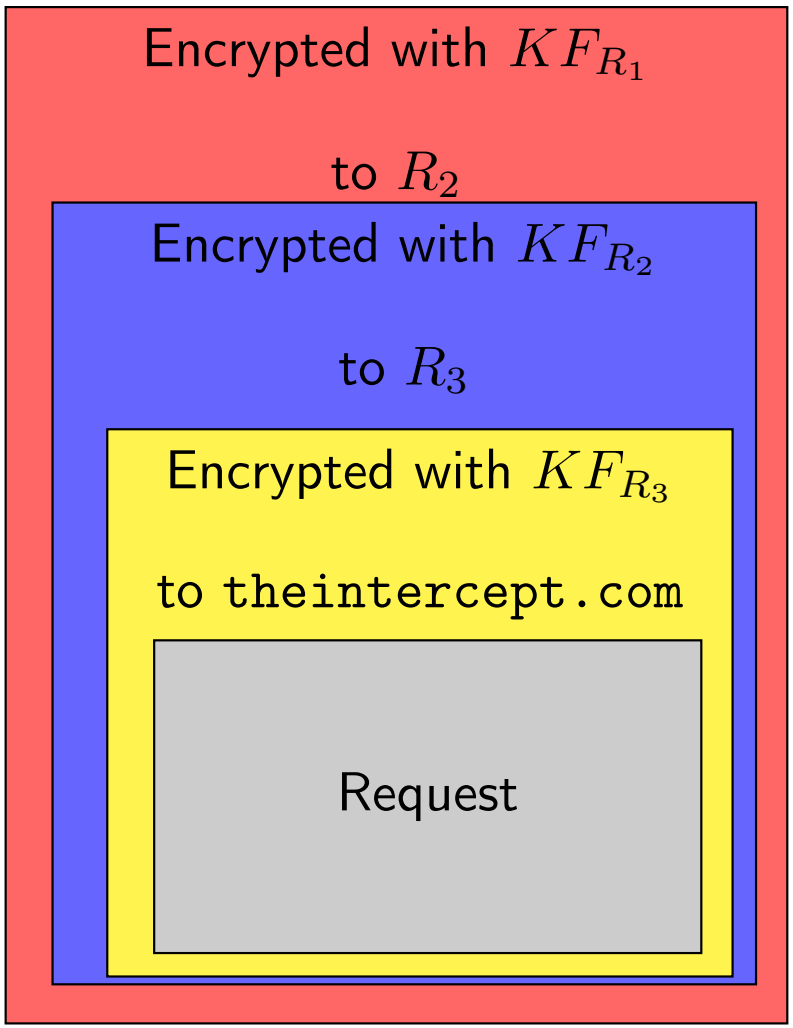

- Assume user has public keys for all relays.
- Use relay public keys to setup an authenticated and encrypted channel, which is used to establish symmetric keypairs for the *forward* and *reverse paths*:
  - Entry relay $R_1$ (keys $KB_{R_1}$, $KF_{R_1}$)
  - Middle relay $R_2$ (keys $KB_{R_2}$, $KF_{R_2}$)
  - Exit relay $R_3$ (keys $KB_{R_3}$, $KF_{R_3}$)
- Wants to anonymously send request to `theintercept.com`
- Prepares packet as follows:
  - Write dest. `theintercept.com`, encrypt with $KF_{R_3}$
  - Write dest. $R_3$ encrypt with $KF_{R_2}$
  - Write dest. $R_2$ encrypt with $KF_{R_1}$

# Onion Routing and Tor




- Assume user has public keys for all relays.
- Use relay public keys to setup an authenticated and encrypted channel, which is used to establish symmetric keypairs for the *forward* and *reverse paths*:
  - Entry relay $R_1$ (keys $KB_{R_1}$, $KF_{R_1}$)
  - Middle relay $R_2$ (keys $KB_{R_2}$, $KF_{R_2}$)
  - Exit relay $R_3$ (keys $KB_{R_3}$, $KF_{R_3}$)
- Wants to anonymously send request to `theintercept.com`
- Prepares packet as follows:
  - Write dest. `theintercept.com`, encrypt with $KF_{R_3}$
  - Write dest. $R_3$ encrypt with $KF_{R_2}$
  - Write dest. $R_2$ encrypt with $KF_{R_1}$
- Send this packet to $R_1$

# Onion Routing and Tor

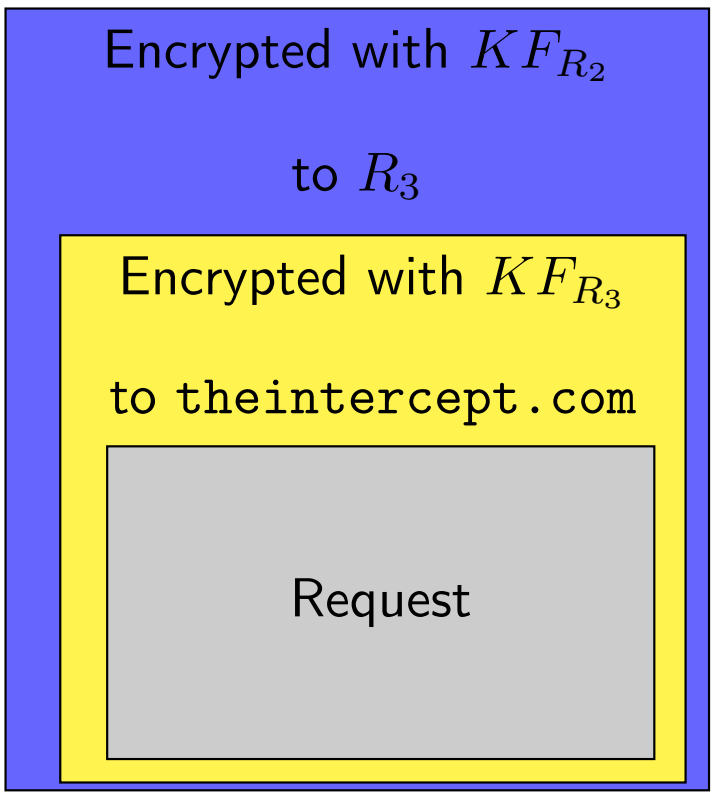



- $R_1$ receives packet, removes encryption with $KF_{R_1}$

# Onion Routing and Tor

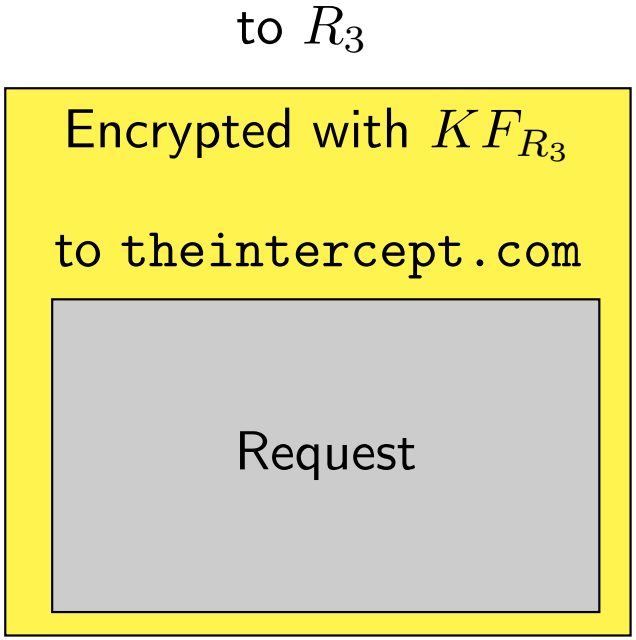

- $R_1$ receives packet, removes encryption with $KF_{R_1}$
- Sees next destination: $R_2$, forwards

# Onion Routing and Tor




- $R_1$ receives packet, removes encryption with $KF_{R_1}$
- Sees next destination: $R_2$, forwards
- $R_2$ receives packet, removes encryption with $KF_{R_2}$

# Onion Routing and Tor




- $R_1$ receives packet, removes encryption with $KF_{R_1}$
- Sees next destination: $R_2$, forwards
- $R_2$ receives packet, removes encryption with $KF_{R_2}$
- Sees next destination: $R_3$, forwards

# Onion Routing and Tor

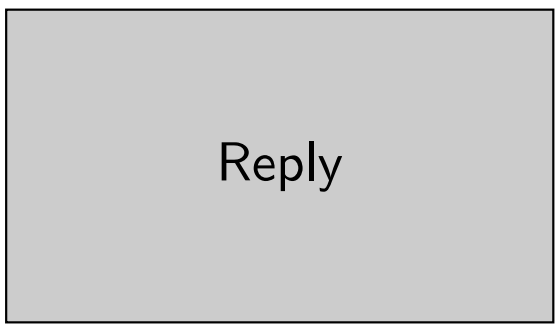

- $R_1$ receives packet, removes encryption with $KF_{R_1}$
- Sees next destination: $R_2$, forwards
- $R_2$ receives packet, removes encryption with $KF_{R_2}$
- Sees next destination: $R_3$, forwards
- $R_3$ receives packet, removes encryption with $KF_{R_3}$

# Onion Routing and Tor

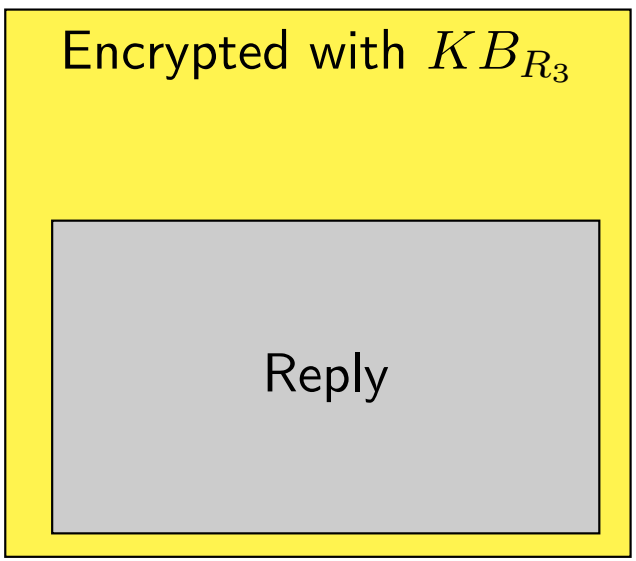

- $R_1$ receives packet, removes encryption with $KF_{R_1}$
- Sees next destination: $R_2$, forwards
- $R_2$ receives packet, removes encryption with $KF_{R_2}$
- Sees next destination: $R_3$, forwards
- $R_3$ receives packet, removes encryption with $KF_{R_3}$
- Sees next destination: `theintercept.com`, sends request

# Response from `theintercept.com`

- $R_3$ receives reply from `theintercept.com`

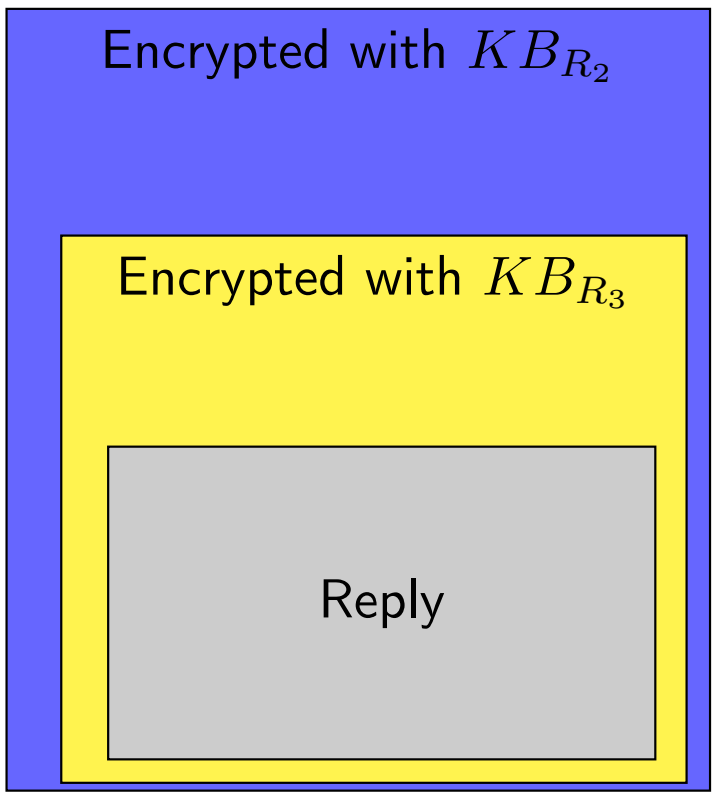


# Response from `theintercept.com`



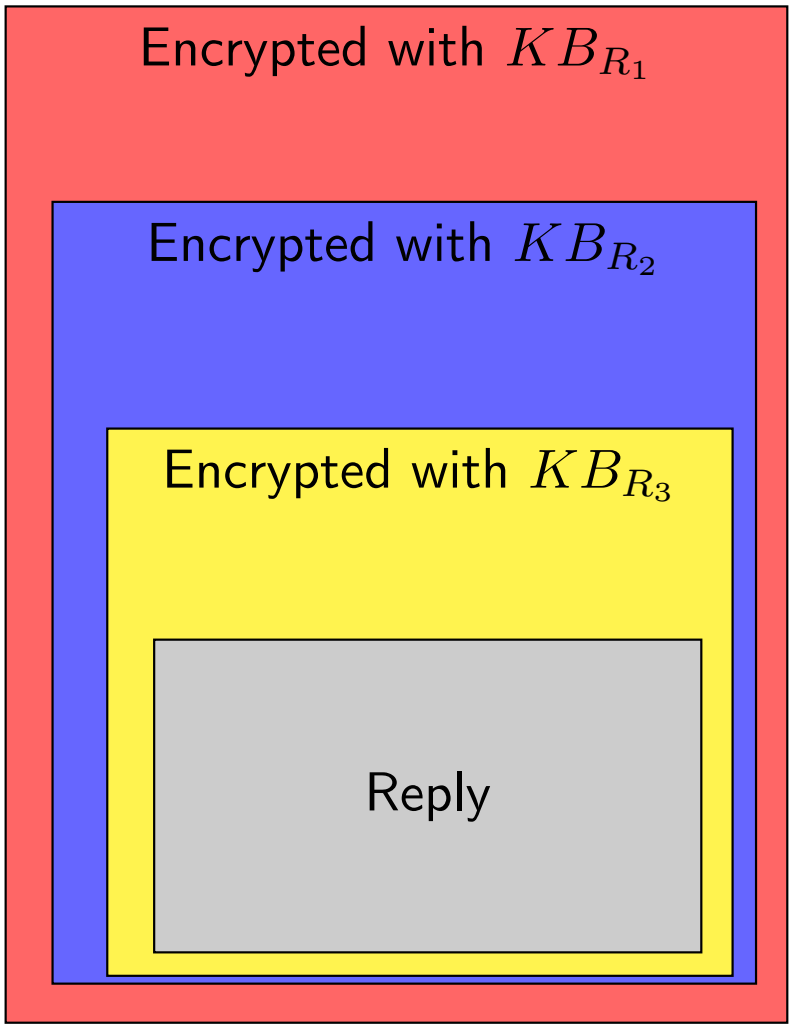

- $R_3$ receives reply from `theintercept.com`
- $R_3$ encrypts with $KB_{R_3}$, sends to $R_2$

# Response from `theintercept.com`



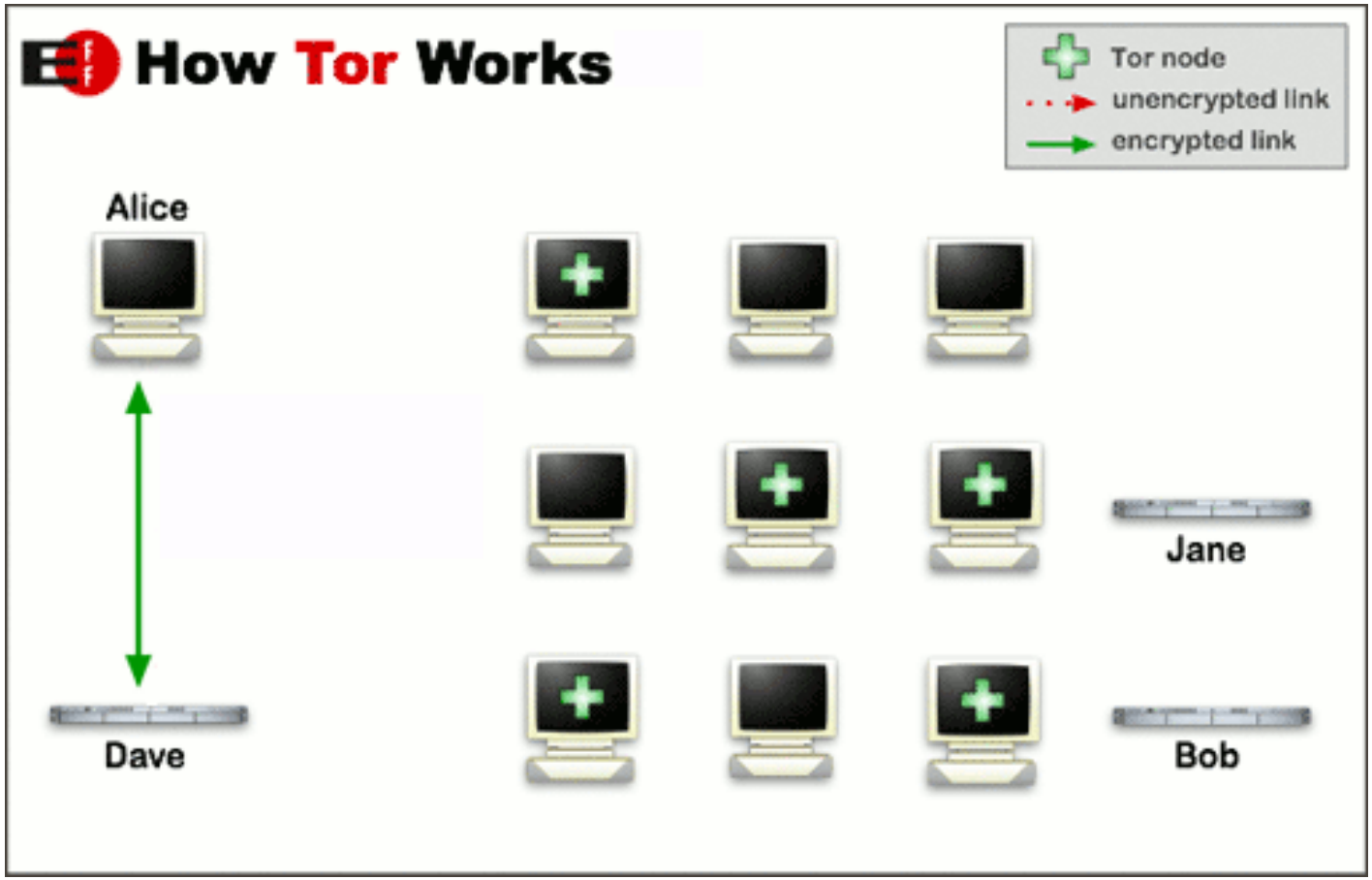

- $R_3$ receives reply from `theintercept.com`
- $R_3$ encrypts with $KB_{R_3}$, sends to $R_2$
- $R_2$ encrypts with $KB_{R_2}$, sends to $R_1$

# Response from `theintercept.com`



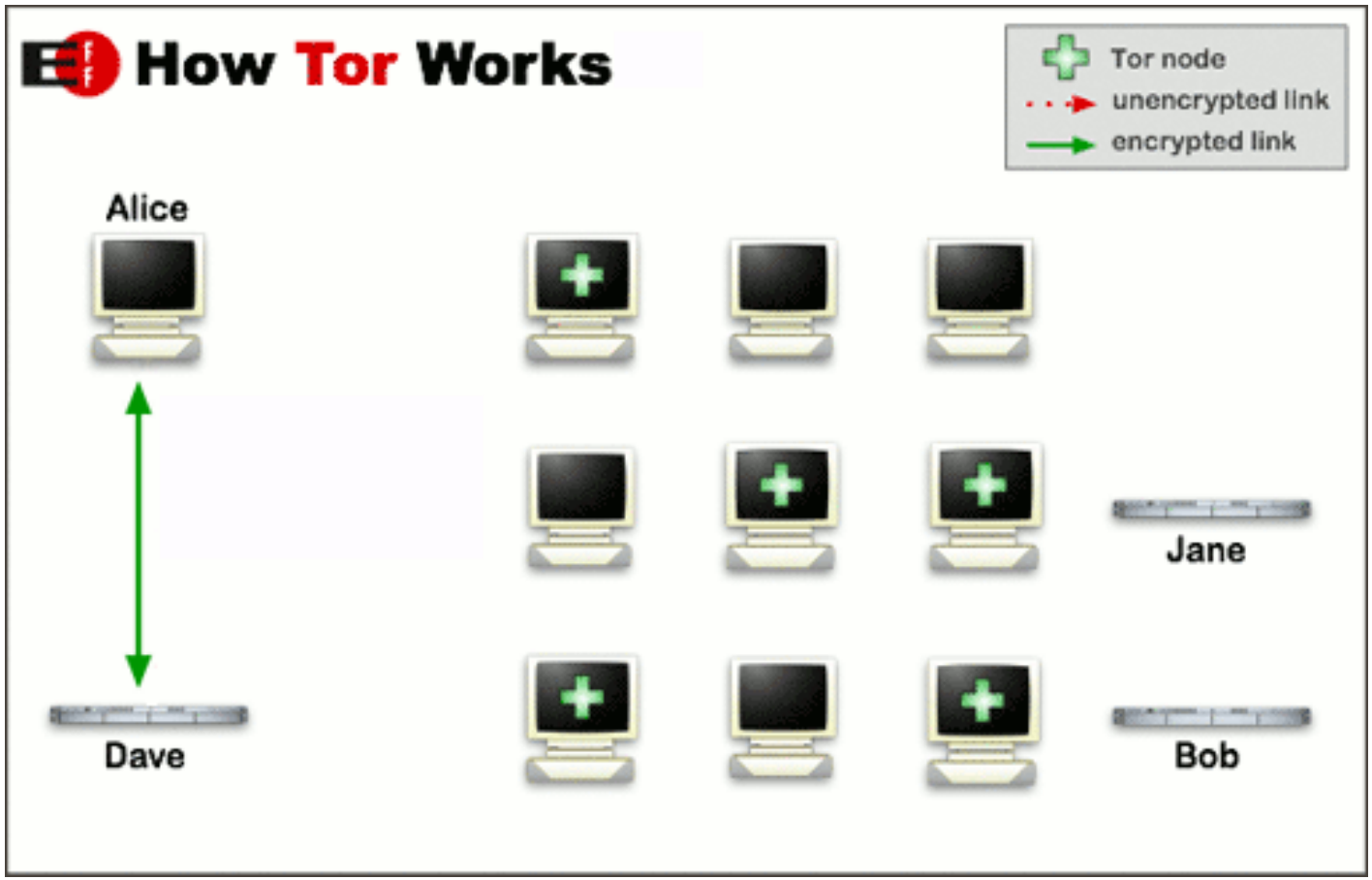

- $R_3$ receives reply from `theintercept.com`
- $R_3$ encrypts with $KB_{R_3}$, sends to $R_2$
- $R_2$ encrypts with $KB_{R_2}$, sends to $R_1$
- $R_1$ encrypts with $KB_{R_1}$, sends to Tor client

# Establishing a Circuit



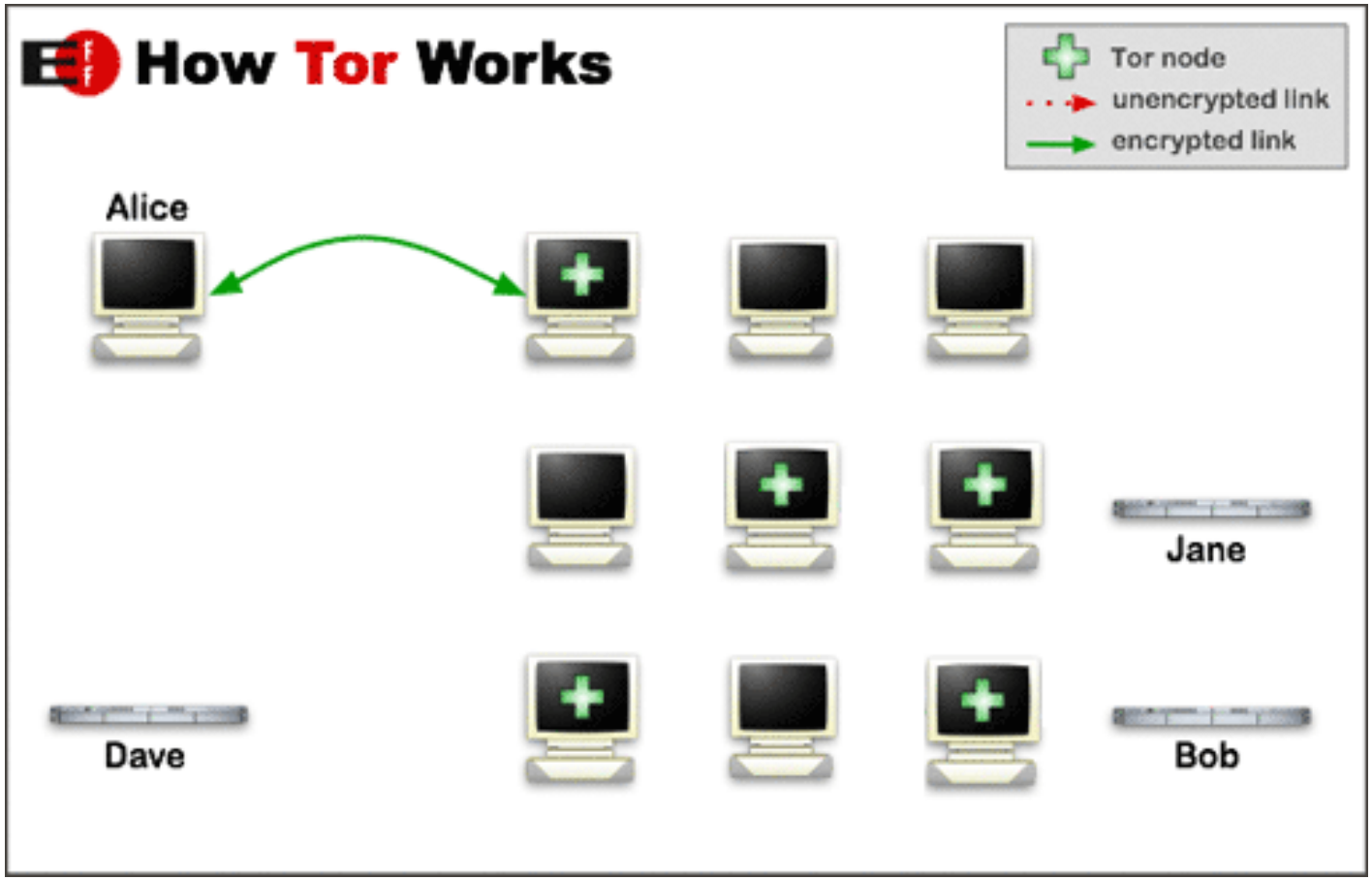

Request listing of Tor nodes from Directory Authorities (DirAuths)

# Establishing a Circuit



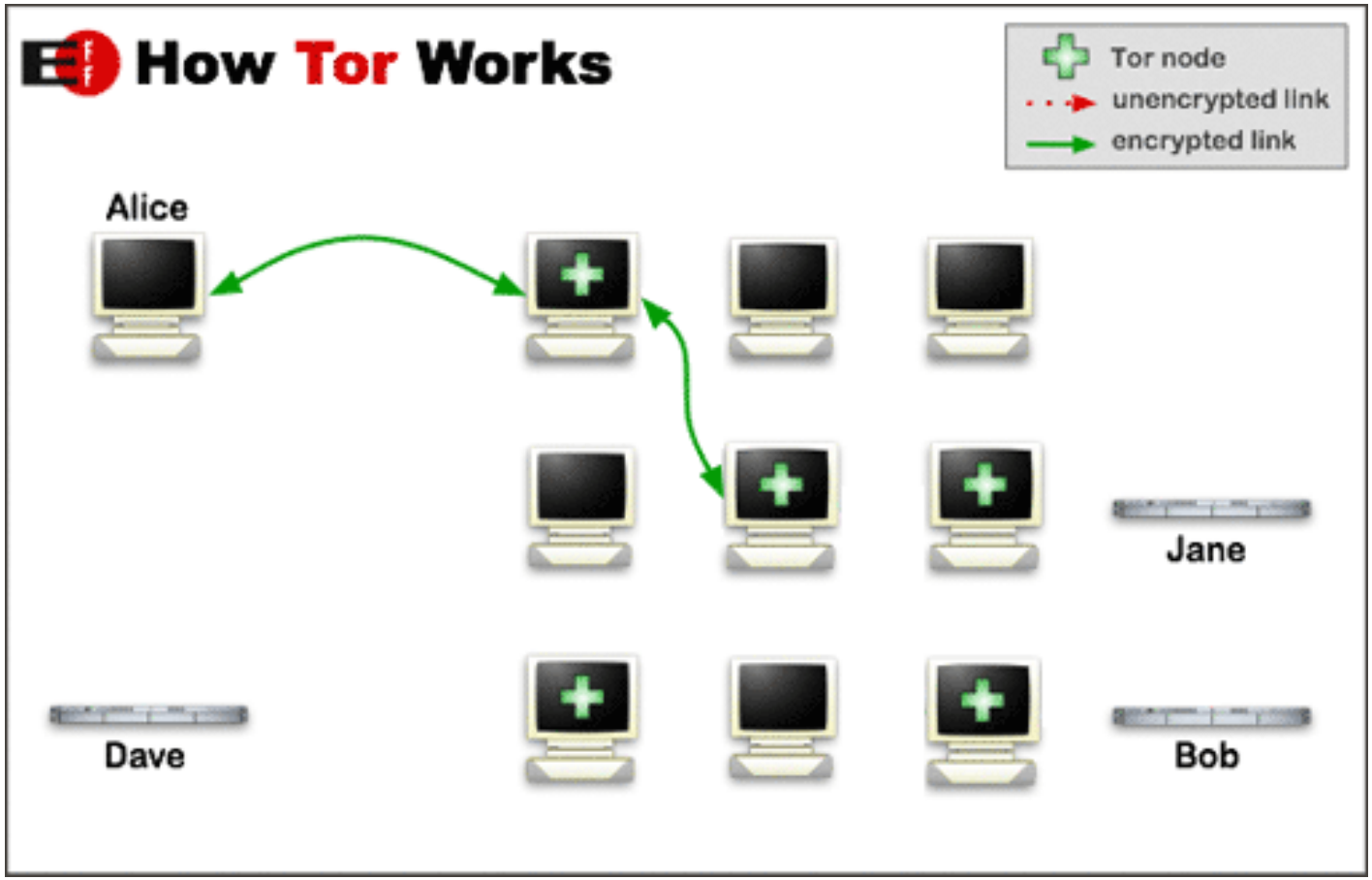

Pick entry, middle, and exit node; obtain their public keys from directory mirror (DirServ)

# Establishing a Circuit



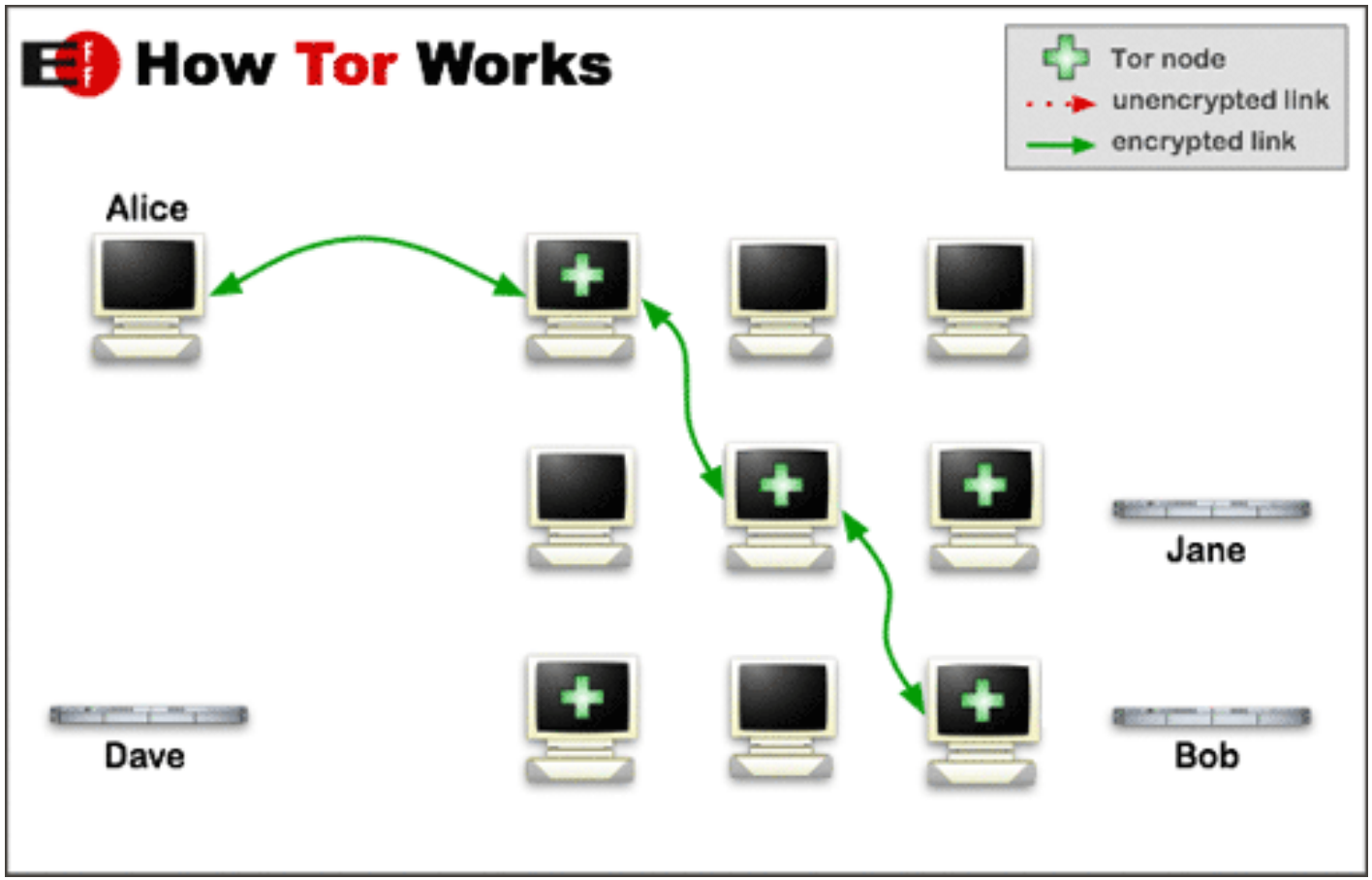

Exchange symmetric key with entry node (Diffie-Hellman)

# Establishing a Circuit



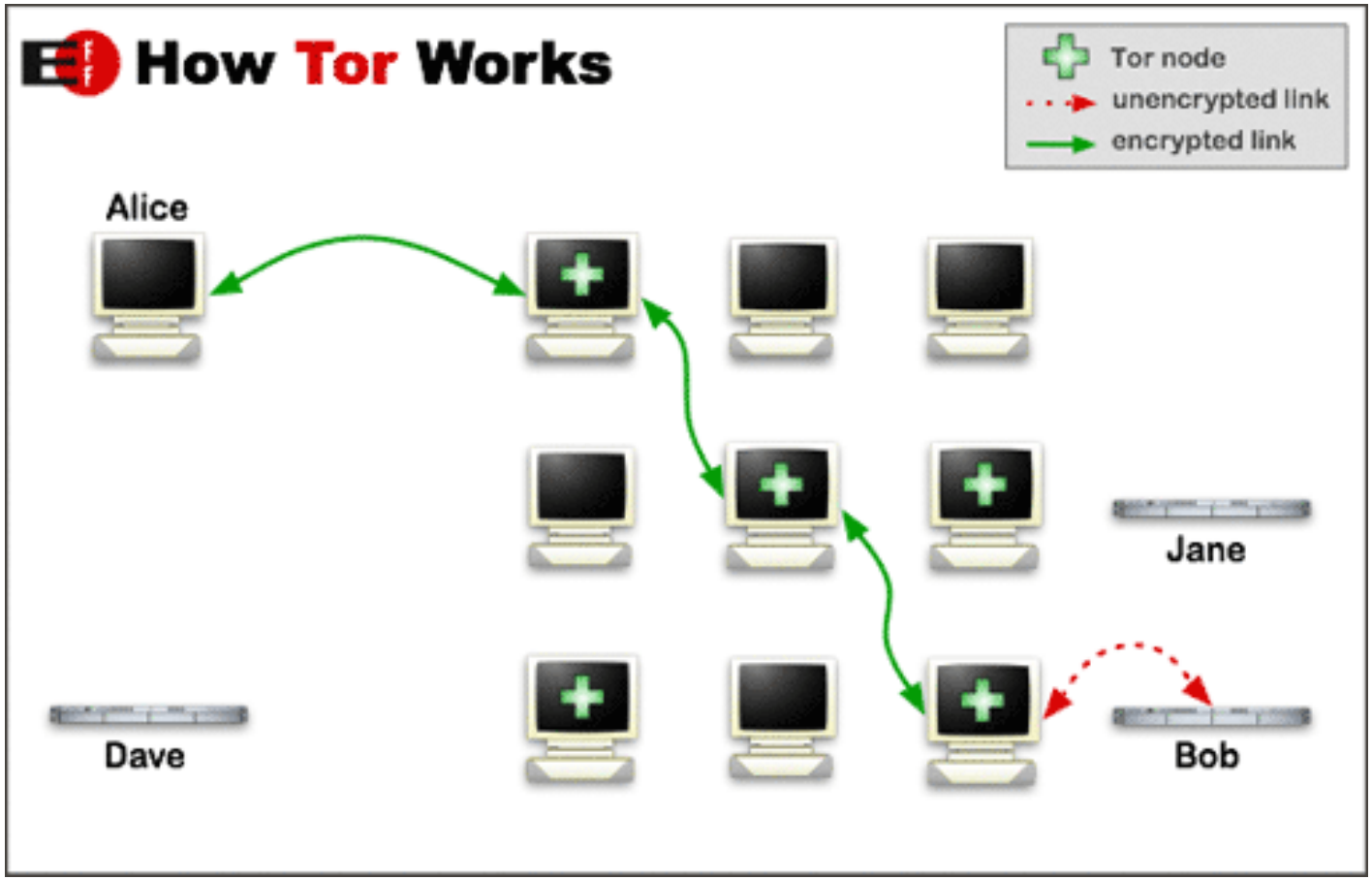

Exchange key with middle node (tunnelled through entry node)

# Establishing a Circuit




Exchange key with exit node (tunnelled through middle node, tunnelled through entry node)

# Establishing a Circuit




Communicate with Bob (`theintercept.com`)

# Some problems with Tor, part I

- Tor offers anonymity up to the transport layer

# Some problems with Tor, part I

- Tor offers anonymity up to the transport layer
- Tor cannot offer application-level anonymity
- Example: I connect through Tor and send a tweet as `@isislovecruft`:

  "My name is Isis Agora Lovecruft, but my *real* name is Urte Pfiffer, I'm 25-years-old, and my IP address is 106.187.37.158."

# Some problems with Tor, part I

- Tor offers anonymity up to the transport layer
- Tor cannot offer application-level anonymity
- Example: I connect through Tor and send a tweet as `@isislovecruft`:

  "My name is Isis Agora Lovecruft, but my *real* name is Urte Pfiffer, I'm 25-years-old, and my IP address is 106.187.37.158."
- Various Bittorrent send the IP address as part of application data
- Conclusion: Bittorrent over Tor isn't a good idea (It's also a bad idea for other reasons, e.g. the torrent bandwidth allocation algorithm doesn't play nicely with Tor)

# Some problems with Tor, part I

- Tor offers anonymity up to the transport layer
- Tor cannot offer application-level anonymity
- Example: I connect through Tor and send a tweet as `@isislovecruft`:

  "My name is Isis Agora Lovecruft, but my *real* name is Urte Pfiffer, I'm 25-years-old, and my IP address is 106.187.37.158."
- Various Bittorrent send the IP address as part of application data
- Conclusion: Bittorrent over Tor isn't a good idea (It's also a bad idea for other reasons, e.g. the torrent bandwidth allocation algorithm doesn't play nicely with Tor)
- Browsers are easily identifiable, see Panopticlick by EFF
- Conclusion: Use the Tor browser (modified Firefox)

# Some problems with Tor, part I

- Tor offers anonymity up to the transport layer
- Tor cannot offer application-level anonymity
- Example: I connect through Tor and send a tweet as `@isislovecruft`:

  "My name is Isis Agora Lovecruft, but my *real* name is Urte Pfiffer, I'm 25-years-old, and my IP address is 106.187.37.158."
- Various Bittorrent send the IP address as part of application data
- Conclusion: Bittorrent over Tor isn't a good idea (It's also a bad idea for other reasons, e.g. the torrent bandwidth allocation algorithm doesn't play nicely with Tor)
- Browsers are easily identifiable, see Panopticlick by EFF
- Conclusion: Use the Tor browser (modified Firefox)
- Tor re-uses an existing circuit for new TCP connections for 10 minutes, but Tor Browser isolates (i.e. uses different) circuits, per second-level domain name in the URL bar.

# Some problems with Tor, part I

- Tor offers anonymity up to the transport layer
- Tor cannot offer application-level anonymity
- Example: I connect through Tor and send a tweet as `@isislovecruft`:

  "My name is Isis Agora Lovecruft, but my *real* name is Urte Pfiffer, I'm 25-years-old, and my IP address is 106.187.37.158."
- Various Bittorrent send the IP address as part of application data
- Conclusion: Bittorrent over Tor isn't a good idea (It's also a bad idea for other reasons, e.g. the torrent bandwidth allocation algorithm doesn't play nicely with Tor)
- Browsers are easily identifiable, see Panopticlick by EFF
- Conclusion: Use the Tor browser (modified Firefox)
- Tor re-uses an existing circuit for new TCP connections for 10 minutes, but Tor Browser isolates (i.e. uses different) circuits, per second-level domain name in the URL bar.
- If you transparently proxy several applications through Tor simultaneously, and one leaks your IP address (bad apple attack), this activity may be linkable to the activity of other applications and decrease anonymity.

# Some problems with Tor, part II

- Most attacks in practice attempt, or assume, control of both the entry and exit (*traffic confirmation attacks*)

# Some problems with Tor, part II

- Most attacks in practice attempt, or assume, control of both the entry and exit (*traffic confirmation attacks*)
- Tor has a leaky-pipe topology, meaning that some types of commands and cells can be sent between relays in a circuit, not just from the user to the edge relay.

# Some problems with Tor, part II

- Most attacks in practice attempt, or assume, control of both the entry and exit (*traffic confirmation attacks*)
- Tor has a leaky-pipe topology, meaning that some types of commands and cells can be sent between relays in a circuit, not just from the user to the edge relay.
  - This is how the recent CMU attack worked:

# Some problems with Tor, part II

- Most attacks in practice attempt, or assume, control of both the entry and exit (*traffic confirmation attacks*)
- Tor has a leaky-pipe topology, meaning that some types of commands and cells can be sent between relays in a circuit, not just from the user to the edge relay.
  - This is how the recent CMU attack worked:
  - Two specific types of command ($relay$ and $relay\_early$) are permitted to go down both the forward and reverse paths

# Some problems with Tor, part II

- Most attacks in practice attempt, or assume, control of both the entry and exit (*traffic confirmation attacks*)
- Tor has a leaky-pipe topology, meaning that some types of commands and cells can be sent between relays in a circuit, not just from the user to the edge relay.
  - This is how the recent CMU attack worked:
  - Two specific types of command ($relay$ and $relay\_early$) are permitted to go down both the forward and reverse paths
  - We fixed a bug ($\#1038$) where infinite-length circuits could be created (which allowed a *resource-exhaustion attack*), by limiting the number of $relay_early$s in the forward direction.

# Some problems with Tor, part II

- Most attacks in practice attempt, or assume, control of both the entry and exit (*traffic confirmation attacks*)
- Tor has a leaky-pipe topology, meaning that some types of commands and cells can be sent between relays in a circuit, not just from the user to the edge relay.
  - This is how the recent CMU attack worked:
  - Two specific types of command ($relay$ and $relay\_early$) are permitted to go down both the forward and reverse paths
  - We fixed a bug ($\#1038$) where infinite-length circuits could be created (which allowed a *resource-exhaustion attack*), by limiting the number of $relay_early$s in the forward direction.
  - We didn't limit the number of $relay_early$s in the reverse direction, which allowed the CMU researchers to encode tags by alternating $relay$ and $relay\_early$ cells.

# Some problems with Tor, part II

- Most attacks in practice attempt, or assume, control of both the entry and exit (*traffic confirmation attacks*)
- Tor has a leaky-pipe topology, meaning that some types of commands and cells can be sent between relays in a circuit, not just from the user to the edge relay.
  - This is how the recent CMU attack worked:
  - Two specific types of command ($relay$ and $relay\_early$) are permitted to go down both the forward and reverse paths
  - We fixed a bug ($\#1038$) where infinite-length circuits could be created (which allowed a *resource-exhaustion attack*), by limiting the number of $relay_early$s in the forward direction.
  - We didn't limit the number of $relay_early$s in the reverse direction, which allowed the CMU researchers to encode tags by alternating $relay$ and $relay\_early$ cells.
  - CMU researchers simultaneously did a *Sybil attack* by running several high-bandwidth relays in both entry and exit positions.

# Correlation attacks

- Tor is aiming at low latency (for web browsing etc.)
- Tor does not wait for traffic to do mixing

# Correlation attacks

- Tor is aiming at low latency (for web browsing etc.)
- Tor does not wait for traffic to do mixing
- *Timing correlation attacks* are still theoretically possible:
  - Think of the whole Tor network as one big proxy
  - Correlate traffic going into and out of this proxy

# Correlation attacks

- Tor is aiming at low latency (for web browsing etc.)
- Tor does not wait for traffic to do mixing
- *Timing correlation attacks* are still theoretically possible:
  - Think of the whole Tor network as one big proxy
  - Correlate traffic going into and out of this proxy
- Tor currently doesn't really do padding:
  - There are patches for the upcoming (0.2.8) version to use padding in the outer TLS connection layer (not circuit-layer), in order to decrease the resolution of timing correlation attacks in netflow records.
  - There are plans to implement adaptive padding techniques (see the "WTF-PAD" paper) at the circuit-level in the future to defend against *website traffic fingerprinting* correlation attacks.

# NSA stinks

- Snowden leaked NSA slides "Tor stinks" from 2007
- Quotes from these slides:

  *"We will never be able to de-anonymize all Tor users all the time."*

  *"With manual analysis we can de-anonymize a **very small fraction** of Tor users, however **no** success de-anonymizing a user in response to a TOPI request/on demand."*

# NSA stinks

- Snowden leaked NSA slides "Tor stinks" from 2007
- Quotes from these slides:

  *"We will never be able to de-anonymize all Tor users all the time."*

  *"With manual analysis we can de-anonymize a **very small fraction** of Tor users, however **no** success de-anonymizing a user in response to a TOPI request/on demand."*
- Our assumption is that they meant they were sometimes able to control both the entry and exit relays for some circuits (but couldn't do it on demand), however that they otherwise cannot deanonymise users.

# NSA stinks

- Snowden leaked NSA slides "Tor stinks" from 2007
- Quotes from these slides:

  *"We will never be able to de-anonymize all Tor users all the time."*

  *"With manual analysis we can de-anonymize a **very small fraction** of Tor users, however **no** success de-anonymizing a user in response to a TOPI request/on demand."*
- Our assumption is that they meant they were sometimes able to control both the entry and exit relays for some circuits (but couldn't do it on demand), however that they otherwise cannot deanonymise users.
- Not-so-secretly, we think the NSA stinks too. :P

# Tor as censorship circumvention

- Various countries filter Internet traffic by destination address
- Most prominent example: Great Firewall of China

# Tor as censorship circumvention

- Various countries filter Internet traffic by destination address
- Most prominent example: Great Firewall of China
- Firewalls and gateways cannot see the true destination of Tor traffic
- Tor is a powerful tool to circumvent online censorship (e.g., in China, Iran, Turkey, Kazakhstan, Ethiopia, others)

# Tor as censorship circumvention

- Various countries filter Internet traffic by destination address
- Most prominent example: Great Firewall of China
- Firewalls and gateways cannot see the true destination of Tor traffic
- Tor is a powerful tool to circumvent online censorship (e.g., in China, Iran, Turkey, Kazakhstan, Ethiopia, others)
- Can also use Tor to circumvent country filters:
  - Need an IP address that isn't in Germany (e.g. because of GEMA restrictions on YouTube): can use Tor access YouTube from a non-German IP address.

# Censorship of Tor

- Easy solution for censors:

# Censorship of Tor

- Easy solution for censors:
  - Obtain list of Tor nodes from the Directory Authorities

# Censorship of Tor

- Easy solution for censors:
  - Obtain list of Tor nodes from the Directory Authorities
  - Block access to the Tor network (all relays)

# Censorship of Tor

- Easy solution for censors:
  - Obtain list of Tor nodes from the Directory Authorities
  - Block access to the Tor network (all relays)
- Solution: Tor Bridges

# Tor Bridges

- *Bridge relays* or *bridges* are secret entrances to the Tor network.

# Tor Bridges

- *Bridge relays* or *bridges* are secret entrances to the Tor network.
- Bridge IP address and other connection information must be distributed out-of-band

# Tor Bridges

- *Bridge relays* or *bridges* are secret entrances to the Tor network.
- Bridge IP address and other connection information must be distributed out-of-band
- DPI or an active adversary is required to identify Bridges

# Tor Bridges

- *Bridge relays* or *bridges* are secret entrances to the Tor network.
- Bridge IP address and other connection information must be distributed out-of-band
- DPI or an active adversary is required to identify Bridges
- Distributed via a centralised system called BridgeDB. Users can currently obtain bridges by:
  - visiting `https://bridges.torproject.org/`
  - writing e-mail to bridges@torproject.org

# Tor Bridges

- *Bridge relays* or *bridges* are secret entrances to the Tor network.
- Bridge IP address and other connection information must be distributed out-of-band
- DPI or an active adversary is required to identify Bridges
- Distributed via a centralised system called BridgeDB. Users can currently obtain bridges by:
  - visiting `https://bridges.torproject.org/`
  - writing e-mail to bridges@torproject.org
- Yes, that whole system sucks. One of my current projects is redesigning and rewriting it.

# Pluggable Transports

- Censors can also block Tor by identifying Tor traffic

# Pluggable Transports

- Censors can also block Tor by identifying Tor traffic
- Tor traffic is relatively easy to identify:
  - Disguised as HTTPS traffic, but
  - uses random domain names
  - has a characteristic packet-size distribution

# Pluggable Transports

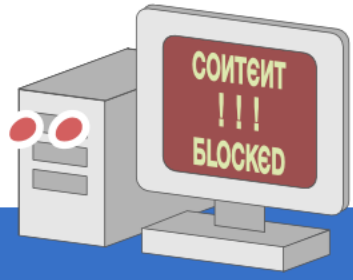
- Censors can also block Tor by identifying Tor traffic
- Tor traffic is relatively easy to identify:
  - Disguised as HTTPS traffic, but
  - uses random domain names
  - has a characteristic packet-size distribution
- Solution: disguise Tor traffic as other traffic
- Pluggable Transport API allows communication between obfuscating SOCKS proxy and Tor client

★ YOU ★

Can Help Protect

Freedom Of Speech

Online...




Run a

TOR RELAY

Today!

★ ★ TorProject.org ★ ★