

Advanced Networking

Privacy defense on the Internet

Csaba Kiraly

Topics

→ Privacy

- ⇒ Anonymity
- ⇒ Pseudonymity
- ⇒ Unlinkability
- ⇒ Unobservability

→ Anonymity on the Internet

- ⇒ Chaum Mix
- ⇒ Mix network & Onion Routing
- ⇒ Low-latency anonymous routing

→ Traffic Analysis & Traffic Flow Confidentiality

- ⇒ Traffic Analysis attacks
- ⇒ The protection: Traffic Flow Confidentiality

Topics

→ **Privacy**

- ⇒ *Anonymity*
- ⇒ *Pseudonymity*
- ⇒ *Unlinkability*
- ⇒ *Unobservability*

→ **Anonymity on the Internet**

- ⇒ Chaum Mix
- ⇒ Mix network & Onion Routing
- ⇒ Low-latency anonymous routing

→ **Traffic Analysis & Traffic Flow Confidentiality**

- ⇒ Traffic Analysis attacks
- ⇒ The protection: Traffic Flow Confidentiality

What Privacy means

→ *Quality or state of being apart from company or observation*

[Webster]

→ *Ability of an individual (or group) to stop information about themselves from becoming known to people other than those they choose to give the information to*

[Wikipedia]

→ *the right of the individual to control his personal information by posing specific obligations on the subjects that process his data*

[legal]



What is not Privacy?

→ *“We are sorry to hear that you wish to delete your account.”*

...

→ *“We here at Skype are dedicated to each and every one of our customers. If there is anything at all we can do to help you get the most out of Skype rather than deleting your account please do not hesitate to contact us by replying to this email.”*

...

→ *“Due privacy and policy we are not able to delete your account completely.”*

[Skype]

Privacy vs. public/private needs

→ Personal information and its processing:

Mandatory for public authorities and bodies

→ to achieve public interests

» governmental, security, economic, tax purposes, etc.

Vital for private sector

→ To improve commercial/customer relationships, business activities and strategies, provisioning of services, etc.

But subject to abuse

→ profiling / data mining techniques

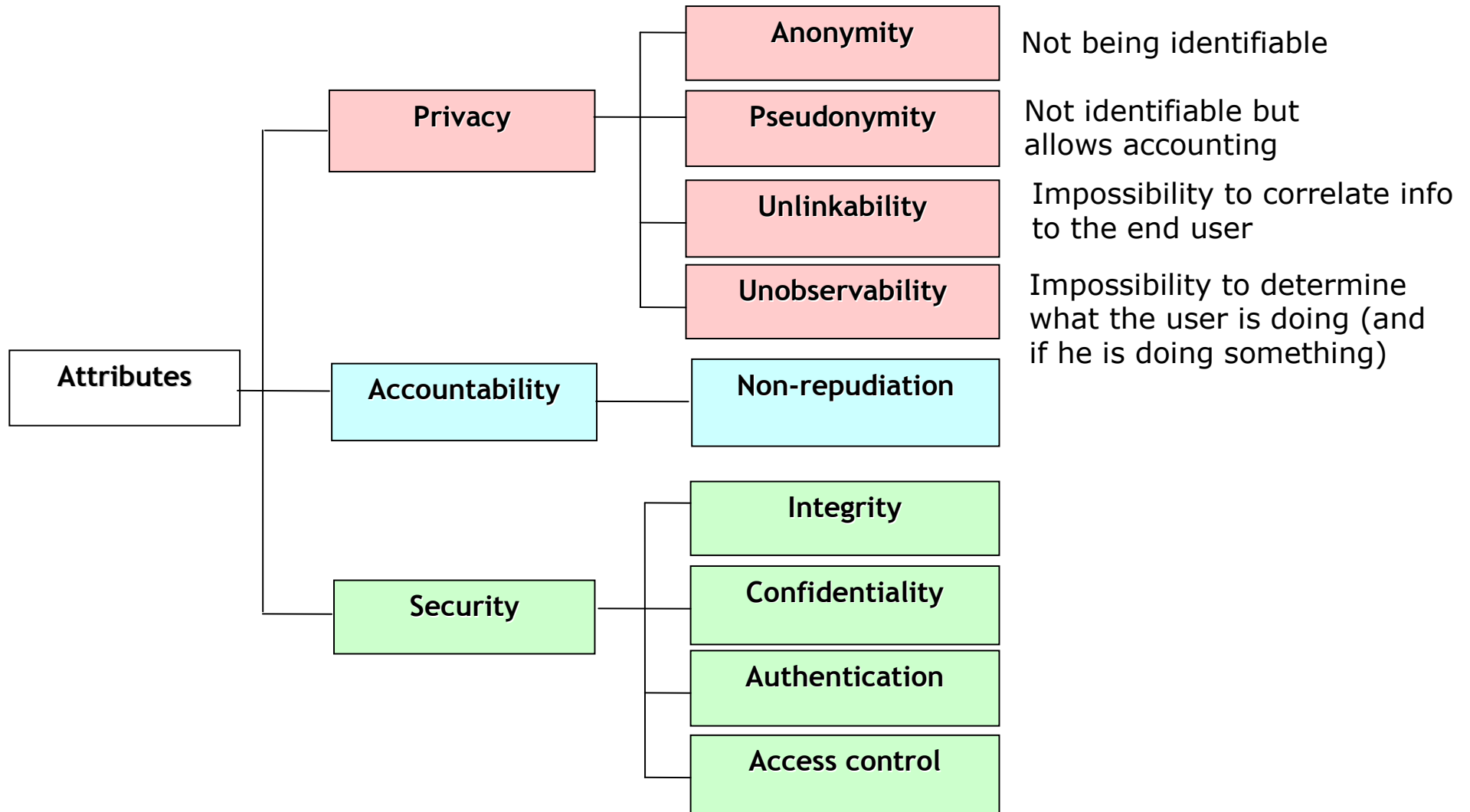
» precise definitions of the individual's preferences

→ Personal information on consumers perceived as asset

» often subject matter of commercial transactions

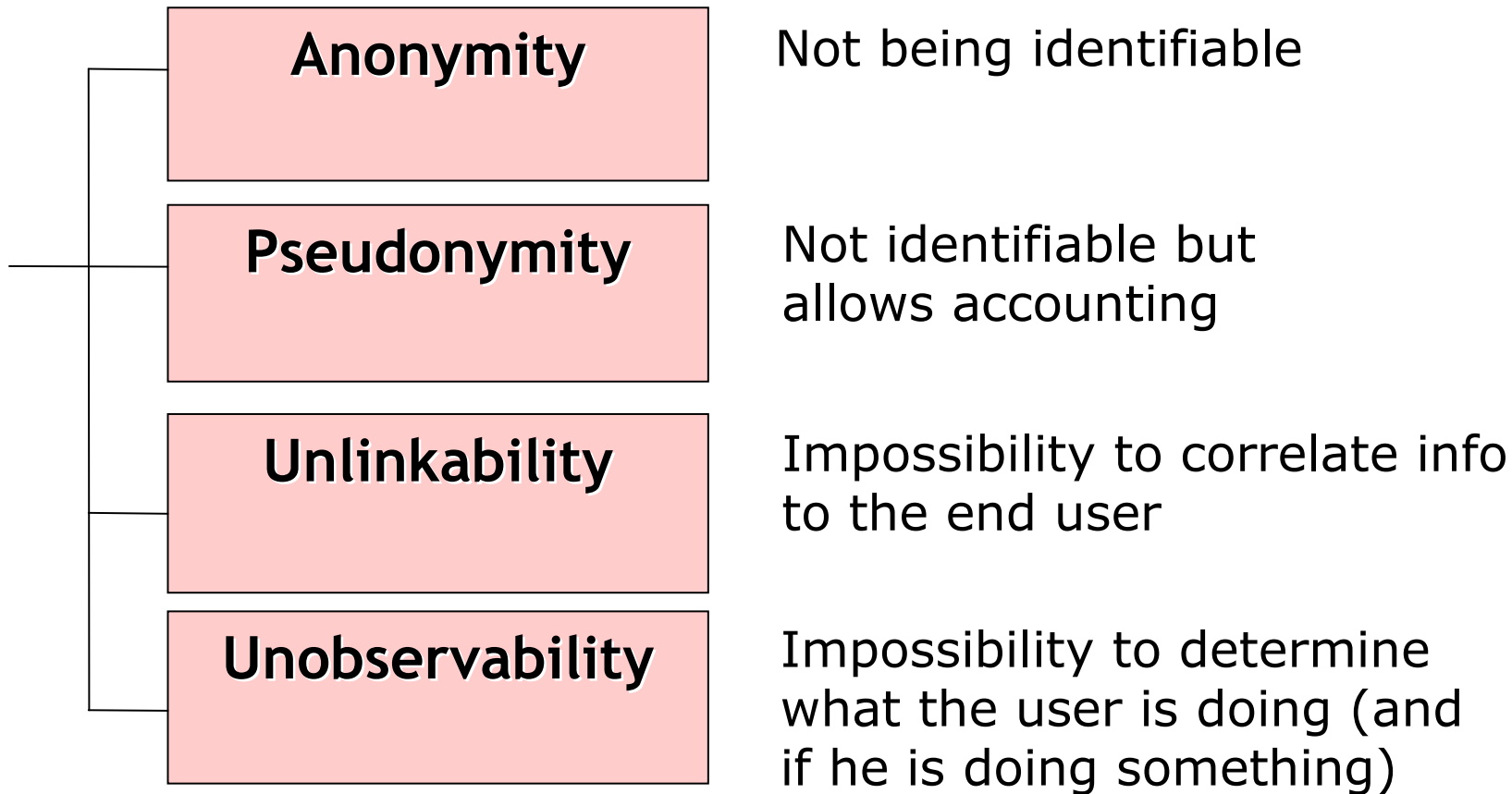


Privacy attributes



More precise definition @ <http://www.freehaven.net/anonbib/cache/terminology.pdf>

Privacy attributes



More precise definition @ <http://www.freehaven.net/anonbib/cache/terminology.pdf>

==== csaba.kiraly@disi.unitn.it =====

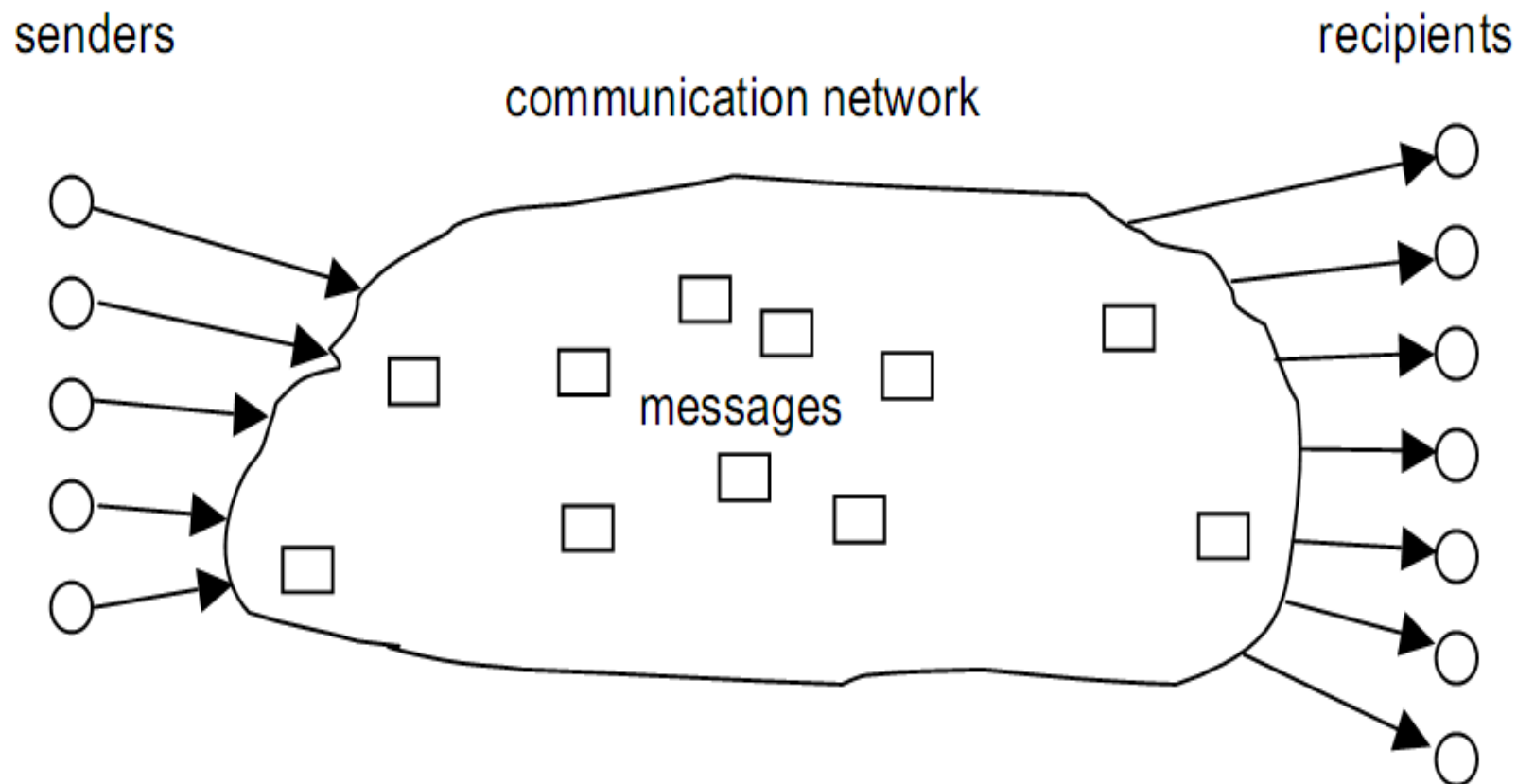
Setting

Privacy properties in different environments:

→ **Databases, Identity management systems, networking,**

⇒ In networking: we use the usual setting that senders send messages to recipients using a communication network.

→ **Easy to generalize to database access, shopping, etc.**



Possible attackers

→ Outsider

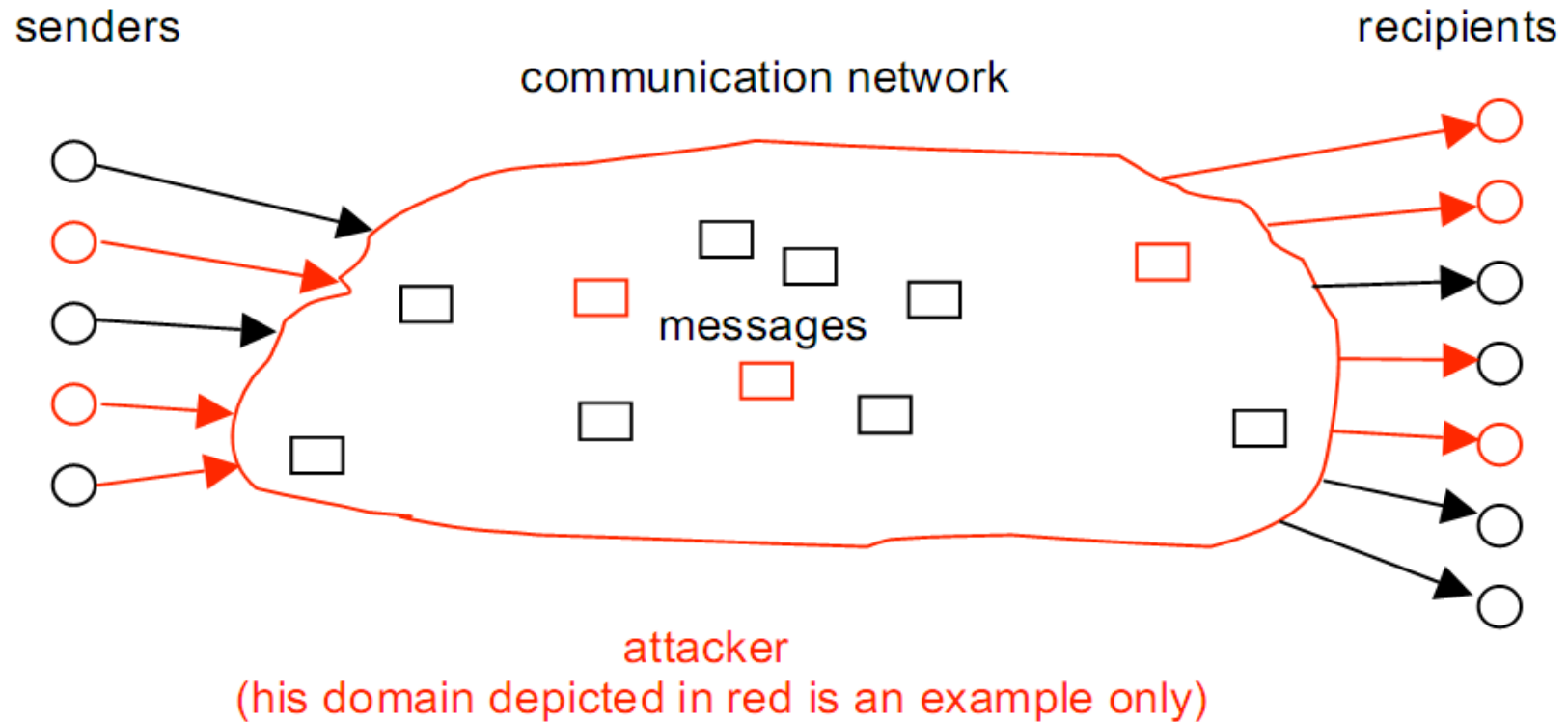
⇒ eavesdropping on communication lines

→ Insider

⇒ able to participate in normal communications and controlling at least some stations

→ Other side of the communication (sometimes)

⇒ Can be seen as special case of insider



Anonymity

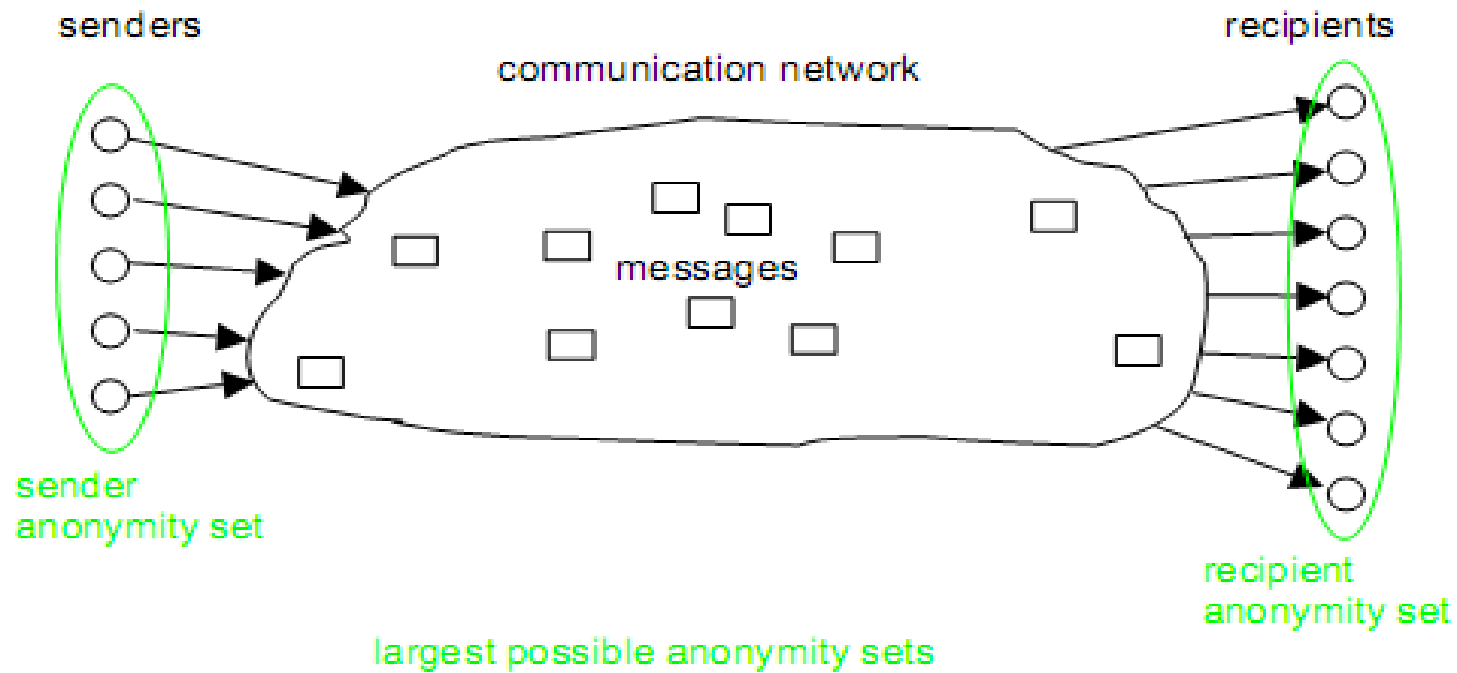
→ **Anonymity is the state of being not identifiable**

⇒ within a set of subjects: the **anonymity set**

→ Sender anonymity set

→ Recipient anonymity set

⇒ You can't be anonymous as the only user of a system!

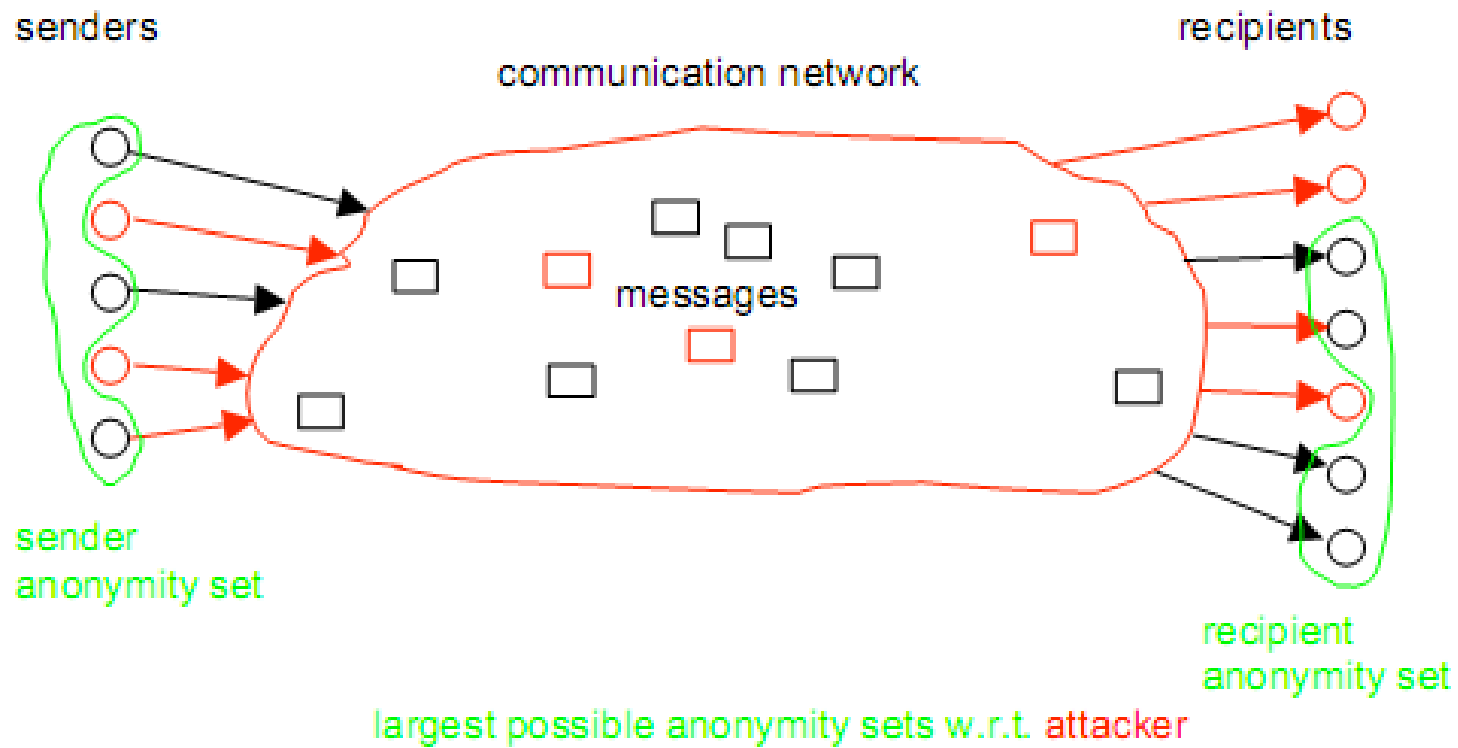


Anonymity

→ **If attacker can exclude hosts**

⇒ Anonymity set reducing

⇒ Up to the point when it contains only 1



Unlinkability

→ Unlinkability of two or more items:

⇒ from the attacker's perspective
no relation can be found between the items (no more and no less related after his observation than they are related concerning his a-priori knowledge).

→ Example for linkability:

⇒ **Alice** was paying **140,40** EUR to buy something at **14:47**

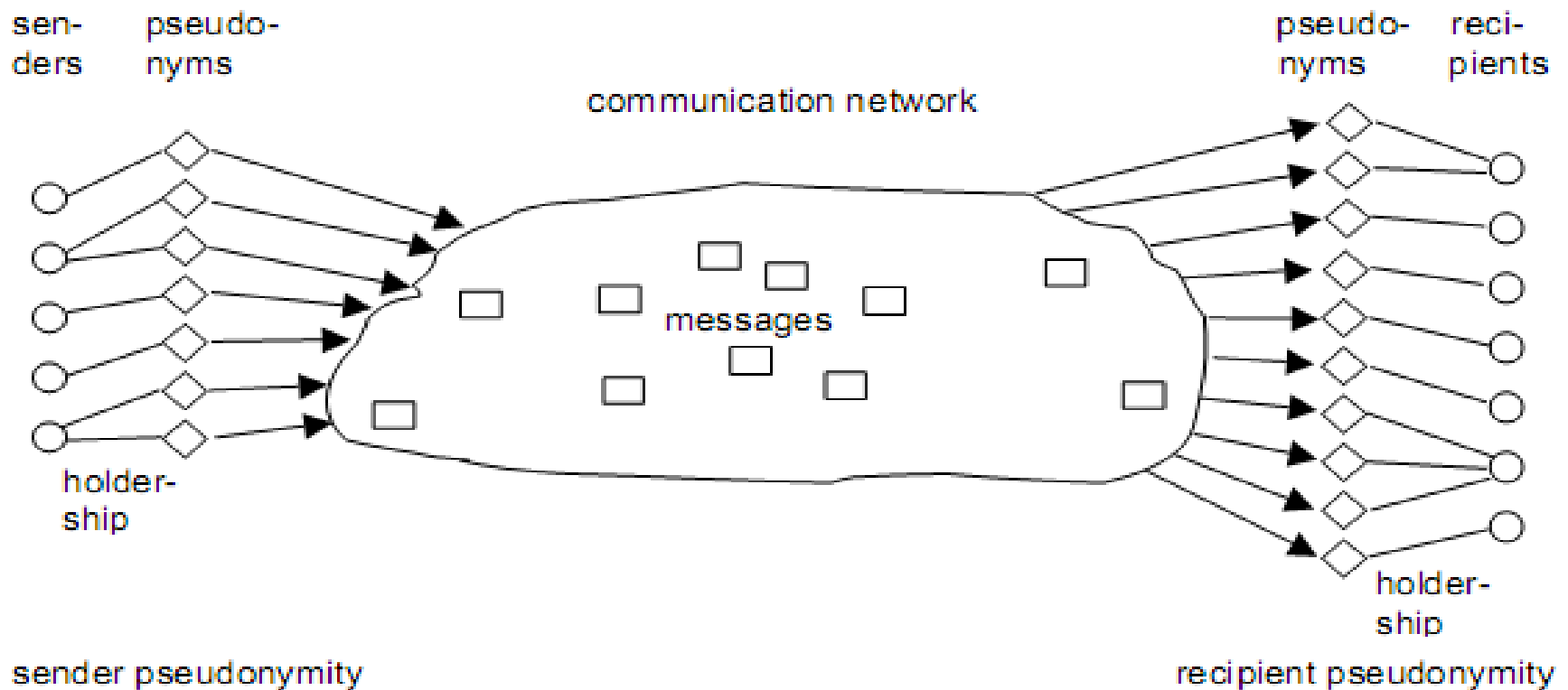
⇒ Someone was buying *airplane ticket to the Bahamas* at **14:47** for **140,40** EUR

⇒ If these are in the same hand → linkable

Pseudonymity

→ using a pseudonym as ID

- ⇒ To avoid linkage (using several different pseudonyms)
- ⇒ To be accountable (reversible, if needed)



Unobservability

→ **Impossible to observe whether an event was taking place, whether a property is true, etc.**

→ **Examples:**

⇒ Impossibility of determining what the user is doing (and if he is doing something)

→ browsing web pages, but impossible to know what

→ Using the network, but impossible to know what kind of application

⇒ Impossible to determine how much communication is

→ between embassy and home country

→ between military units.

Topics

→ Privacy

- ⇒ Anonymity
- ⇒ Pseudonymity
- ⇒ Unlinkability
- ⇒ Unobservability

→ **Anonymity on the Internet**

- ⇒ *Chaum Mix*
- ⇒ *Mix network & Onion Routing*
- ⇒ *Low-latency anonymous routing*

→ Traffic Analysis & Traffic Flow Confidentiality

- ⇒ Traffic Analysis attacks
- ⇒ The protection: Traffic Flow Confidentiality

Anonymity: Chaum mix

→ David L. Chaum (1981):

- ⇒ How to send anonymous e-mail ...
- ⇒ with return path

→ Designed for e-mail

- ⇒ Most of the concepts can be reused at packet level!

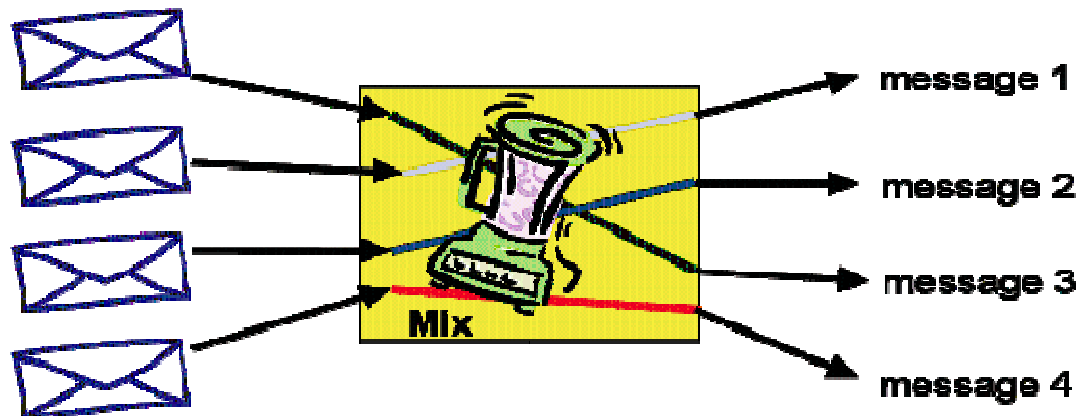
→ What anonymous means here:

- ⇒ Protect from external attacker
 - Someone eavesdropping on the communication should not understand who is communicating
- ⇒ From internal attacker
 - A mail server should not know who is communicating to whom
- ⇒ From other side
 - The recipient should not know who was sending the mail

Chaum mix: tricks

1. What if someone can eavesdrop on the communication?

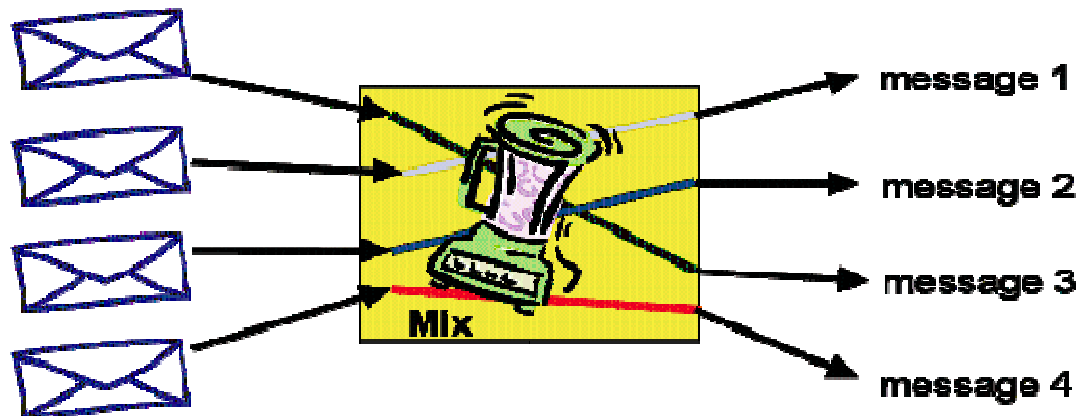
- ⇒ Solution: not sending directly
 - Proxy (**mix**) removes/changes source
- ⇒ Solution: use encryption
 - Send message to mix encrypted (with its public key)
 - Mix sends out message decrypted



Chaum mix: tricks

2. What if someone can inspect links of the mix?

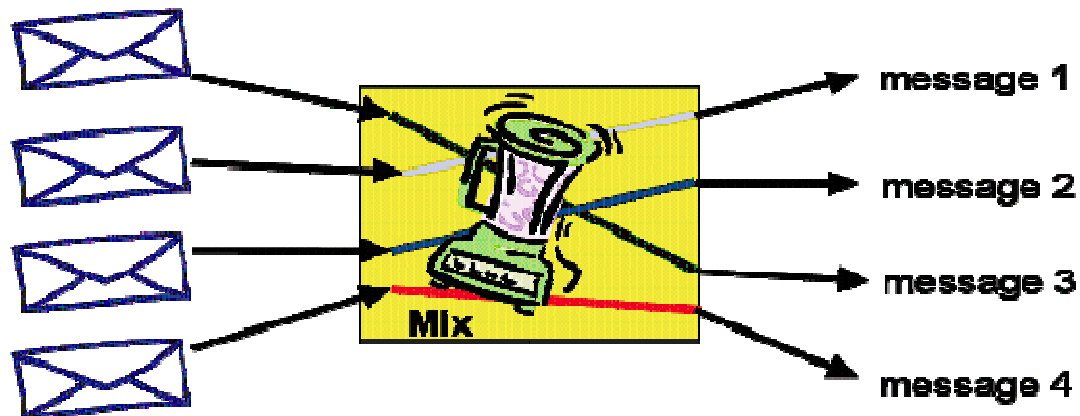
- ⇒ Can see incoming and outgoing message right after each other
- ⇒ Solution: delay messages
 - Form batches of messages
 - Send them out in random (or lexicographical) order



Chaum mix: tricks

3. What if someone can inspect links of the mix?

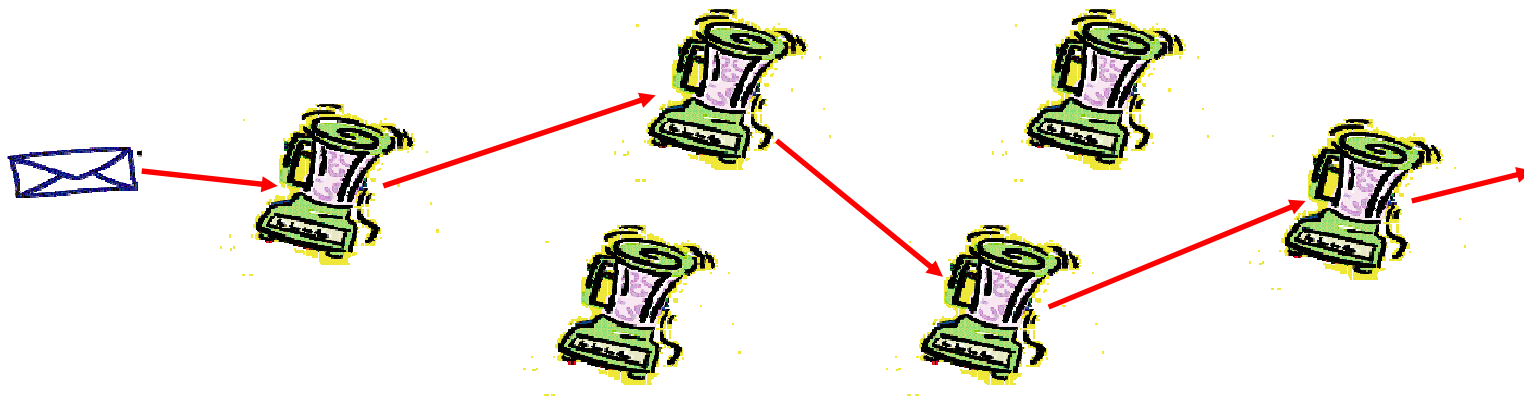
- ⇒ Can see the identical sized incoming and outgoing messages -> linkage
- ⇒ Solution: pad messages to fixed size



Chaum mix: tricks

Onion routing, steps

1. Source downloads list of Mix nodes & their public keys
2. Select route (source routing): $S \rightarrow M1 \rightarrow M2 \rightarrow M3 \rightarrow D$
3. Construct **onion** message
 1. $D, K_D(\text{Message})$
 2. $M3, K_{M3}(D, K_D(\text{Message}))$
 3. $M1, K_{M1}(M2, K_{M2}(M3, K_{M3}(D, K_D(\text{Message}}))))$



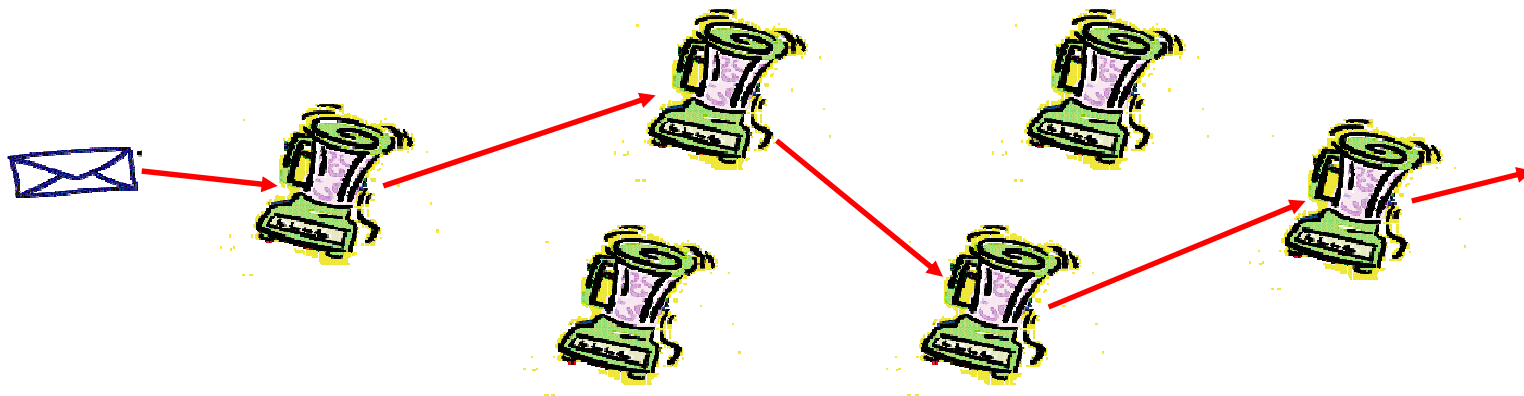
Chaum mix: tricks

3. Construct **onion** message

1. $D, K_D(\text{Message})$
2. $M3, K_{M3}(D, K_D(\text{Message}))$
3. $M1, K_{M1}(M2, K_{M2}(M3, K_{M3}(D, K_D(\text{Message}))))$

4. Onion travels

1. $S \rightarrow M1: M1, \text{cyphertext}$
2. $M1: M1, K_{M1}(M2, \text{cyphertext}) \rightarrow M2, \text{cyphertext}$
3. $M1 \rightarrow M2: M2, \text{cyphertext}, \dots$



Chaum mix: reply

→ Omitted for lack of time 😞

Low-latency anonymous routing

→ Chaum mix works for

⇒ large messages, high latency applications

⇒ E.g. e-mail

→ What can be done for low latency applications (like web browser)?

⇒ Messages (e-mail) -> packets

→ Why can't it work the same way?

⇒ Asymmetric (public/private key) encryption too slow, not feasible

⇒ Delaying in batches introduces too much latency

Low-latency anonymous routing: Circuit

→ Circuit: Why needed?

- ⇒ Public key cryptography too slow to encrypt each packet in an onion
- ⇒ Use public key cryptography (slow) only at the beginning, to negotiate symmetric keys (fast)
 - once with each node on the path
- ⇒ Use these symmetric keys for the whole flow of packets

→ Consequences:

- ⇒ 2 phases of communication
 - Circuit setup phase
 - Data communication phase

Circuit setup phase

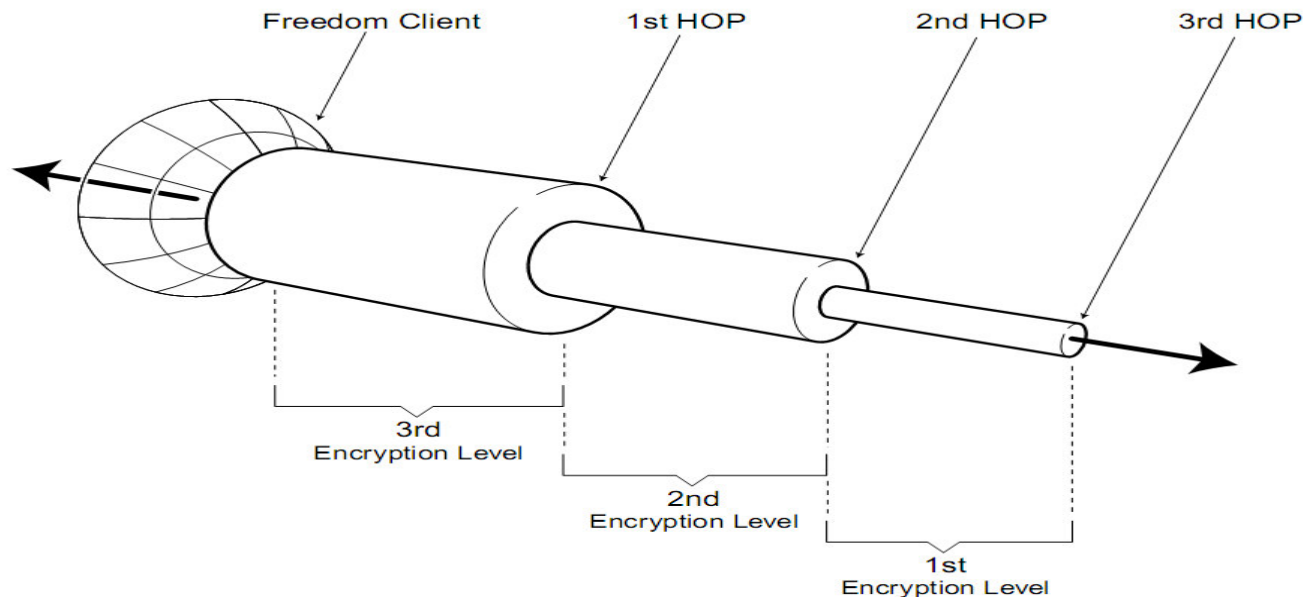
→ **Problem: When negotiating with the 2nd mix, the source must be anonymous**

→ **Idea: incremental circuit setup**

⇒ Negotiate circuit with the 1st mix

⇒ Use this circuit to speak to the 2nd anonymously and extend the circuit

→ **Circuit implemented like a Telescope**



Data on the Circuit

→ Label (circuit ID) switching

⇒ Circuit ID in each packet

⇒ ID swapped in each mix

→ Why?

» ID can't be same, otherwise the path would be easily uncovered

» ID must be unique in the mix

→ Re-encryption

⇒ Similar to Chaum, but now with symmetric keys.

Data return path

→ In case of Chaum, it was cumbersome (not discussed)

→ With circuit:

⇒ The same circuit can be used

⇒ In each hop

→ Swap ID back

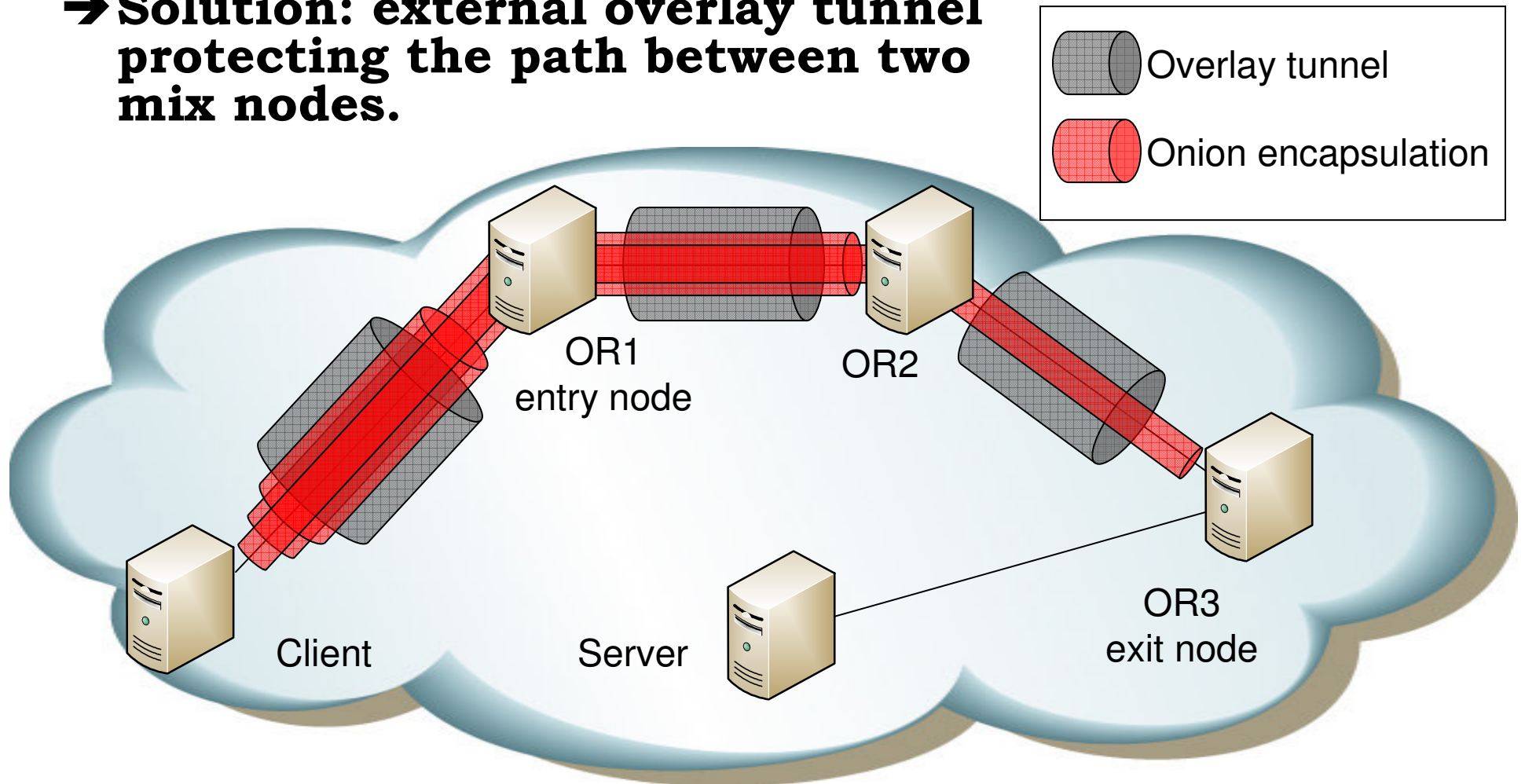
→ Re-encrypt using the symmetric key

→ Send it to “previous” node

⇒ Source knows all the keys, so it can decrypt at the end

Protecting the circuit

- **Problem: circuit ID seen on the path between two mix nodes**
- **Solution: external overlay tunnel protecting the path between two mix nodes.**

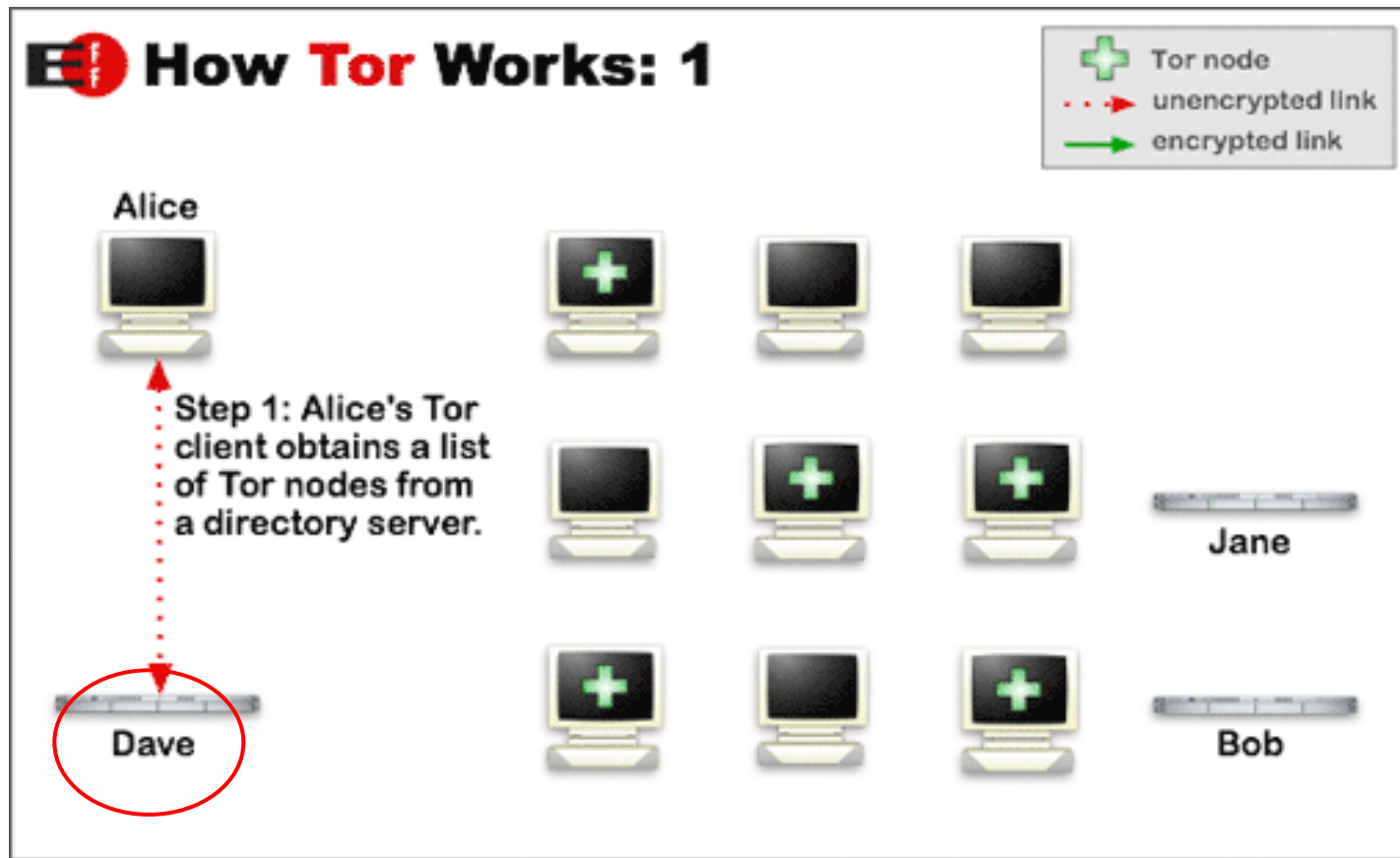


Steps: download list of mix nodes

→ The same as with e-mail

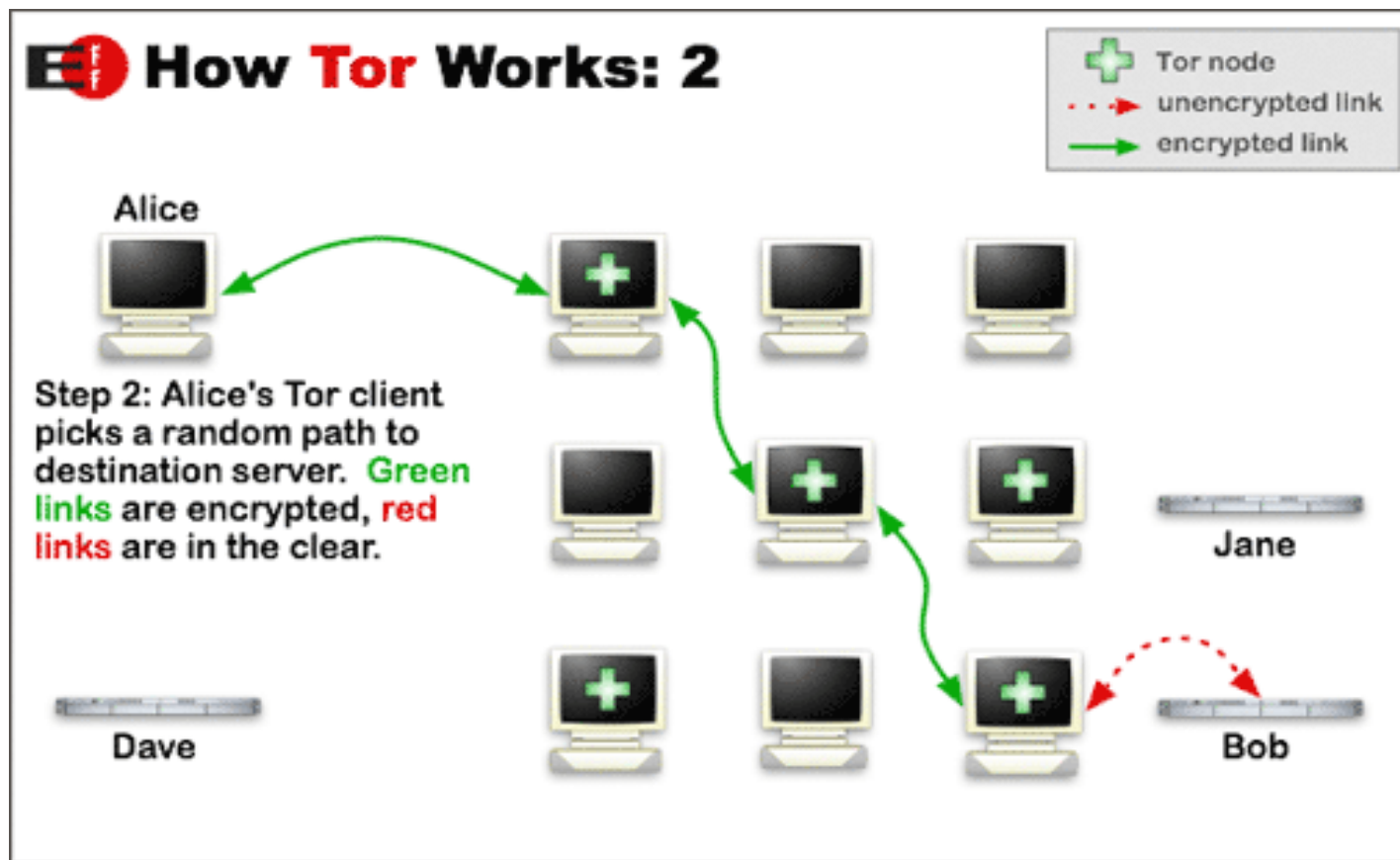
→ List of overlay routers (mix nodes)

⇒ Their IP address and Public Key



Steps: Routing with Circuits

- Decide about route
- Set up circuit for the whole connection!
- Send data through the circuit



Source routing

Decide about route

→ Our goal

- ⇒ Whole route known only by the source
- ⇒ A mix node knows only part of it
 - Previous mix
 - Next mix

→ Consequences

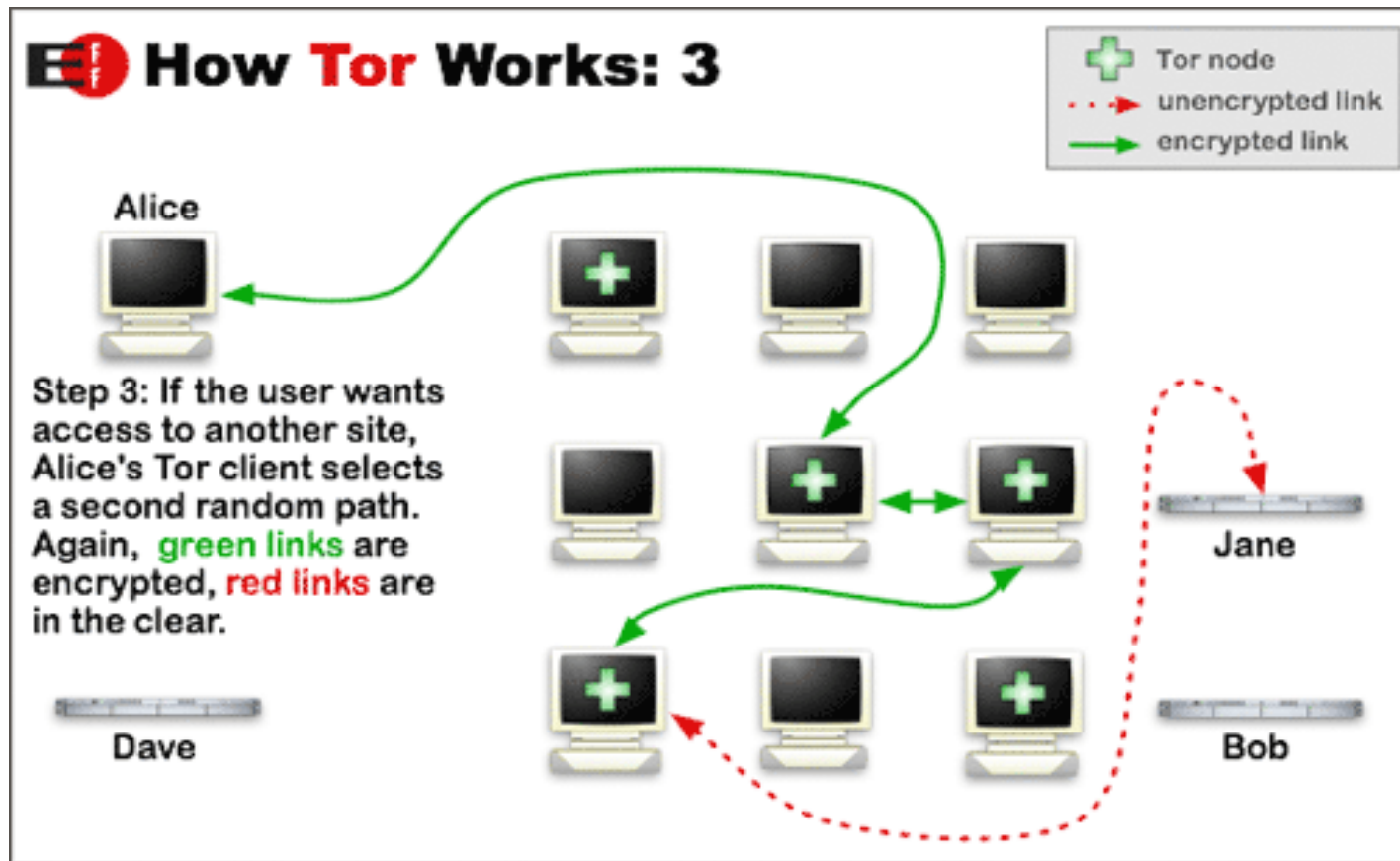
- ⇒ Mix shouldn't know the destination! => only source routing feasible
- ⇒ Route selection should be random
 - Otherwise easy to figure out the route
 - Theoretical optimum: uniform random selection
 - Practical considerations:
 - » In Tor: bandwidth weighted route selection

Steps: Changing circuits

→ **Following communications might use different path**

⇒ to avoid linkage with old data exchanges

⇒ to replace broken circuit



Circuit (dis)advantages

→ Circuit

⇒ Pros:

→ Fast encryption

→ Easy return path routing

⇒ Cons:

→ Two phase communications

→ External overlay tunnel to protect circuit ID

→ State information in each mix node for each circuit!

A working example: Tor

→ Deployed in the Internet

- ⇒ More than 2000 mix nodes, run by volunteers
- ⇒ Much more users

→ The infrastructure

- ⇒ Clients (called Onion Proxies, OP)
- ⇒ Mix nodes (called Onion Routers, OR)
 - Allow OP-OR and OR-OR traffic
- ⇒ Exit nodes
 - Special ORs that also allow traffic towards any server
- ⇒ Directory servers
 - Keep list of available OR nodes

Tor Network Map



Refresh



Zoom In



Zoom Out



Zoom To Fit



Help



Close

- Relay
- SURFnetTor1
 - desync
 - jalopy
 - blutmagie
 - Tonga
 - gpfTOR4
 - Lifuka
 - xanadu
 - atari
 - dotplex1
 - mnl
 - TSL
 - whistlersmother
 - BostonUCompSci
 - Arlequin
 - FoeBuD3
 - dizum
 - cuogh
 - tornodeviennasil
 - c03d9ebf
 - charlesbabbage
 - routor
 - itpol3
 - bettyboop
 - gpfTOR3
 - weltbank
 - nixnix
 - FreeDomOne
 - idbsgetyrubdgrety...
 - GuyMontag
 - HOLLY
 - Kryten
 - juliusagrippa
 - StoneHenge911
 - masterofall
 - gpfTOR2
 - diora
 - BTtec
 - betroffenheit
 - bach



Connection	Status
<Path Empty>	Failed
<Path Empty>	Failed
<Path Empty>	Failed
dizum,HappyFunTorRelay,blutmagie	Open
sb-ssl.google.com:443	Open
images.google.it:80	Closed
ocsp.thawte.com:80	Open
diora,cuogh	Building
dizum,oophikeejee,Wiia	Open
diora,ididnteditheconfig	Building

A working example: Tor

→ Tor works on L7

- ⇒ It does not work on IP packets, but on “cells” of TCP connection of an application
- ⇒ There is nothing like a TCP cell, but Tor splits up the application data to small cells (502 bytes)
- ⇒ These cells are onion encrypted individually, the circuit ID is set, and the cell is routed on the overlay network

→ Integration with the application

- ⇒ SOCKS proxy

→ Onion encryption

- ⇒ Tor uses AES encryption on each cell, re-encrypting for each OR

→ Tunnel between OR nodes

- ⇒ TLS/TCP tunnel
 - Multiplex several circuits going between the same 2 ORs
 - hide circuit IDs
 - send stream of cells

→ In each OR hop TLS/TCP is terminated -> L7 operation

Tor in itself is not enough

→ HTTP query contains information too

```
GET /spec.html HTTP/1.1
Host: www.example.org
User-Agent: Mozilla/5.0 (Macintosh; U; PPC Mac
  OS X Mach-0; en-US; rv:1.8) Gecko/20051130
  Firefox/1.5
Referer: http://comingfrom.com/
Accept: ...
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300 Connection: keep-alive
Cookie: yourtrackingid=123412
```

⇒ Leaks data or allows linkage

→ ***Solution: remove these***

⇒ *e.g. use Privoxy*

Tor in itself is not enough

→ DNS query contains information as well

⇒ Tor hides the destination IP address from Mix nodes and eavesdroppers

⇒ But, DNS query to figure out the IP address is in clear -> eavesdroppers see it

→ *Solution: force the DNS query through Tor as well*

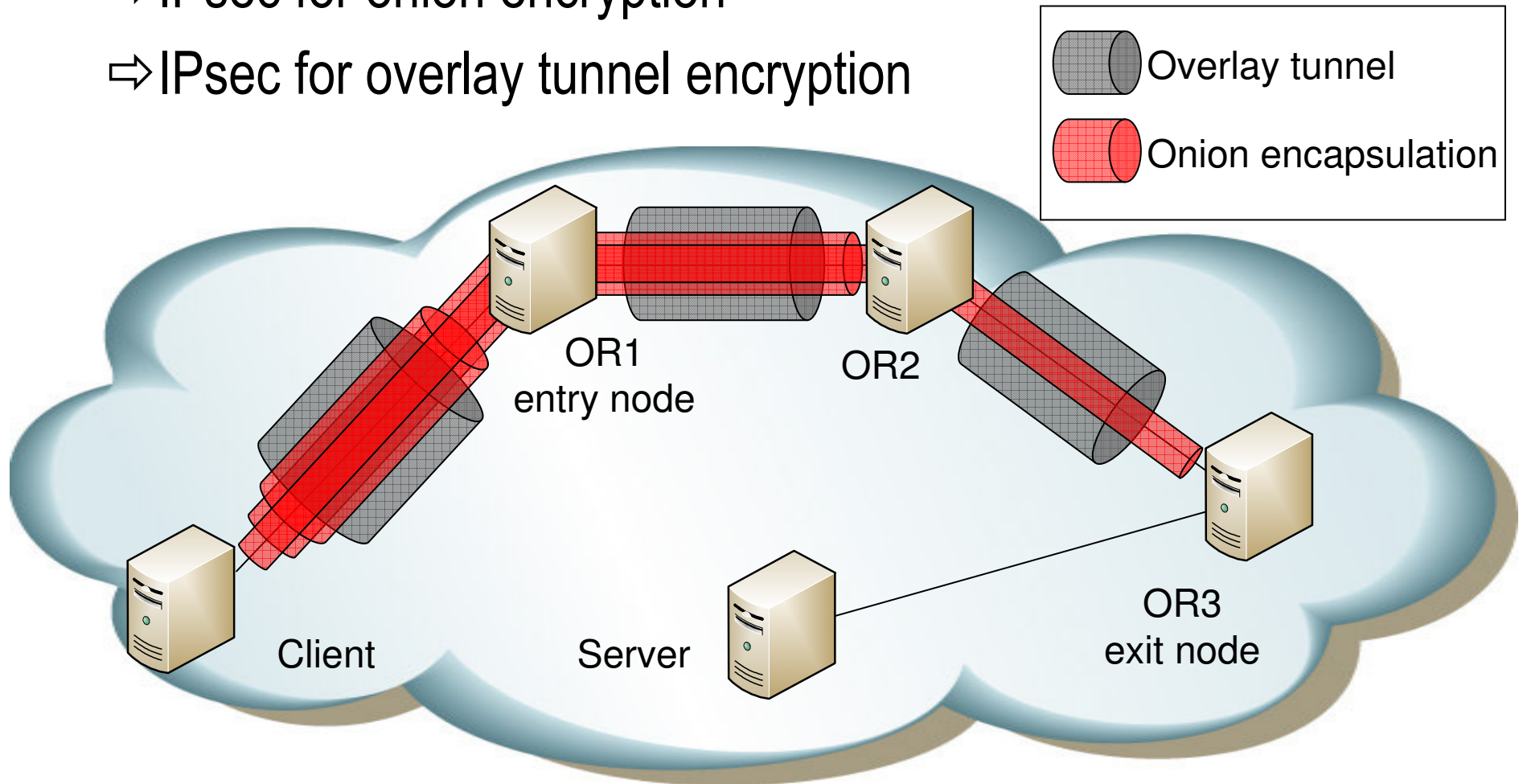
⇒ *Using a special version of SOCKS*

Another example: IPpriv

IPpriv works on L3, using IPsec

⇒ IPsec for onion encryption

⇒ IPsec for overlay tunnel encryption



L7 (Tor) vs. L3 (IPpriv)

L7 solution

- **Integration with the application: SOCKS proxy**
 - ⇒ Only TCP supported
 - ⇒ application support (or wrapper) needed
- **Tunnel between mix nodes: TLS/TCP**
 - ⇒ TCP congestion control and reliable transmission on each tunnel
- **In each hop TLS/TCP is terminated**
 - ⇒ Application level processing
- **Deployment: application**
 - ⇒ Easy to install anywhere
- **Mix operates at: L7**
 - ⇒ L3 and L4 characteristics hidden

L3 solution

- **Integration with the application: IP**
 - ⇒ Not needed
 - ⇒
- **Tunnel between mix nodes: IPsec tunnel**
 - ⇒ Best effort delivery
- **In each hop IPsec decryption and routing**
 - ⇒ Kernel processing with lower delays
- **Deployment: IPsec SP and SA**
 - ⇒ Root privileges needed
- **Mix operates at: L3**
 - ⇒ L3 and L4 characteristics are not hidden

Topics

→ Privacy

- ⇒ Anonymity
- ⇒ Pseudonymity
- ⇒ Unlinkability
- ⇒ Unobservability

→ Anonymity on the Internet

- ⇒ Chaum Mix
- ⇒ Mix network & Onion Routing
- ⇒ Low-latency anonymous routing

→ **Traffic Analysis & Traffic Flow Confidentiality**

- ⇒ *Traffic Analysis attacks*
 - *Web site fingerprinting*
 - *Traffic classification*
- ⇒ *The protection: Traffic Flow Confidentiality*
 - *Basic tools*
 - *Control logic*

(Statistical) Traffic Analysis

→ Traffic analysis

- ⇒ Looking inside the packet
 - Does not work if packet is encrypted
 - Can be really slow (or at least requires huge resources)
- ⇒ What can be done without looking inside the packet?
 - looking only at header (L3/L4)

→ Statistical Traffic Analysis

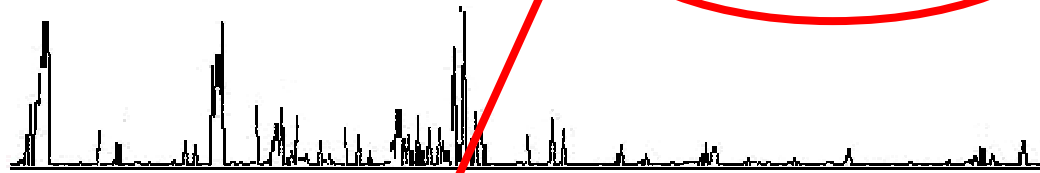
- ⇒ Looking only at some statistical properties of the packet stream / connections
 - Mean packet size -> VoIP?
 - Number of TCP connections opened -> P2P?

No protection: easy eavesdropping

192.168.100.45 72.21.206.5 TCP HTTP GET WWW.AMAZON.COM



client
192.168.100.45

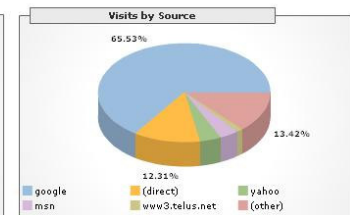
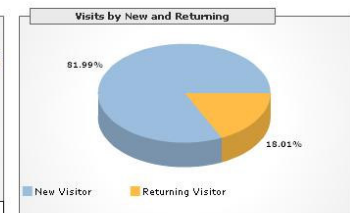
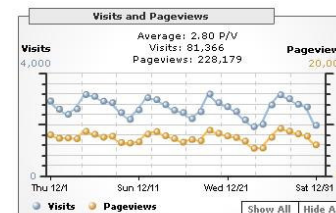


WWW.AMAZON.COM
72.21.206.5

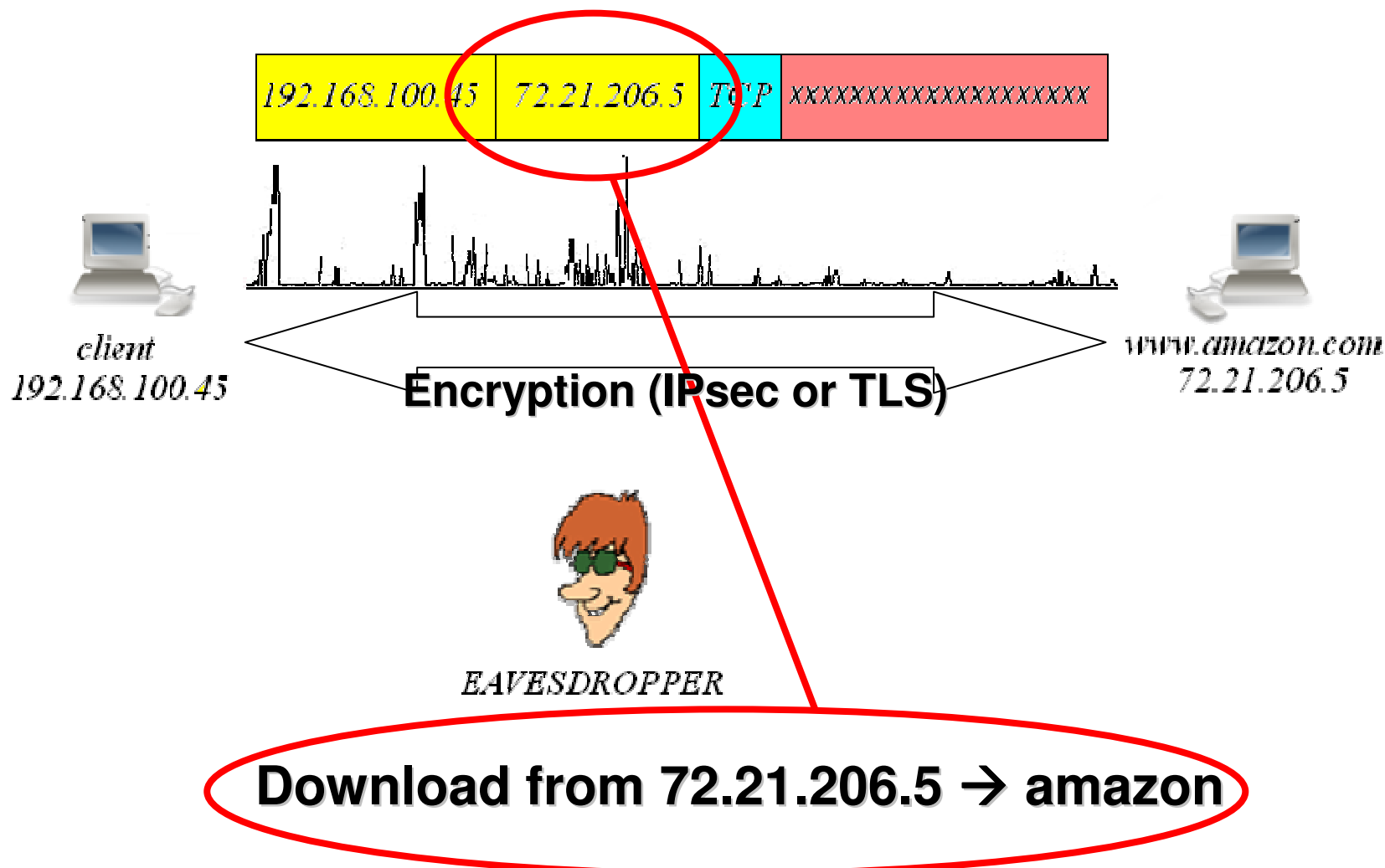


EAVESDROPPER

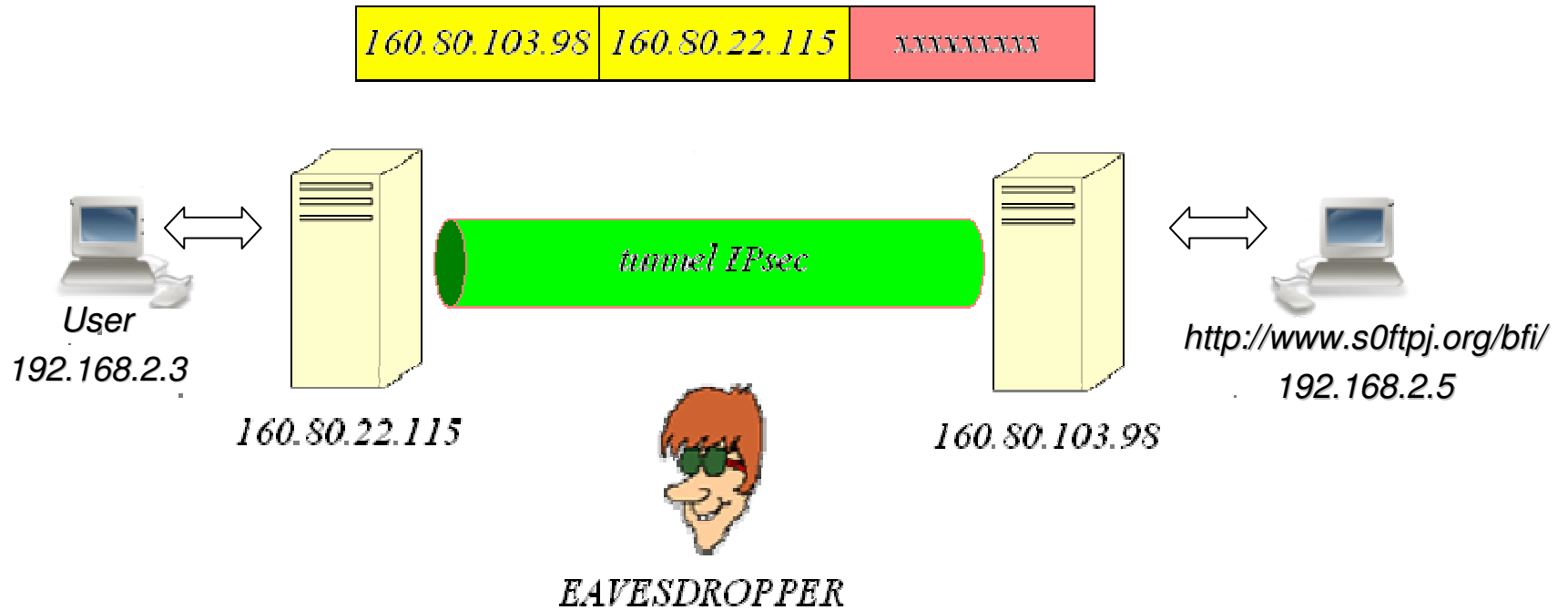
Download from amazon



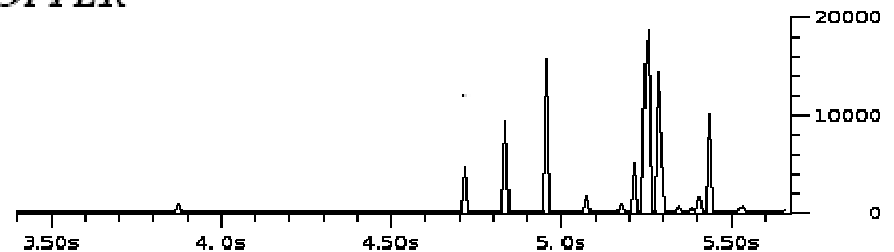
Encryption: not enough to hide everything



Protection by tunnel/onion routing?

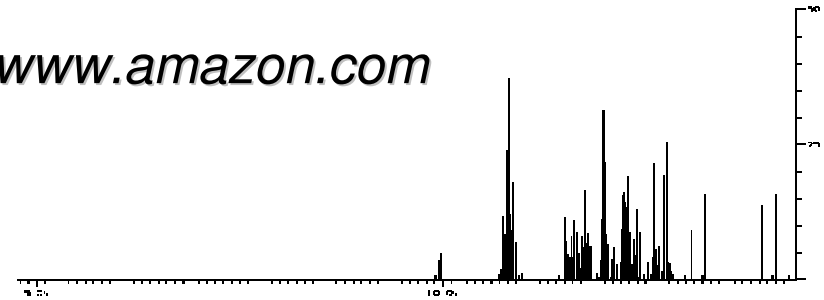


- Length
- Arrival time
- Packets direction

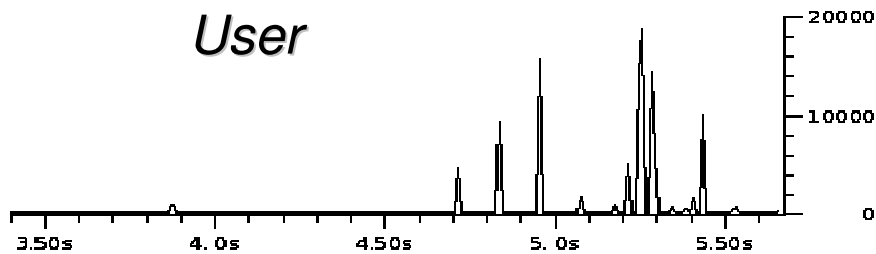


Web site fingerprinting

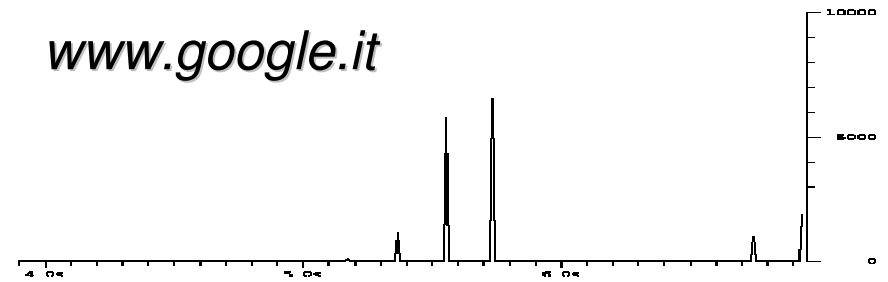
www.amazon.com



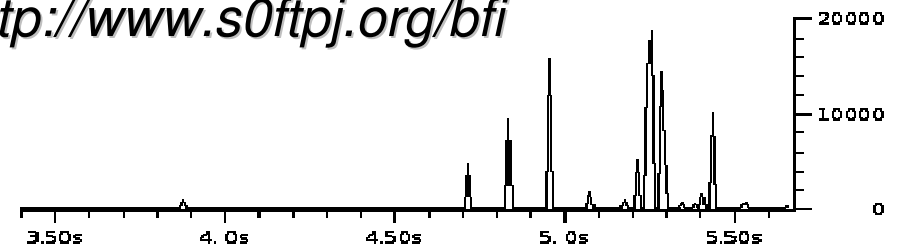
User



www.google.it



http://www.s0ftpj.org/bfi



Traffic Classification

Sort traffic (packets) into classes

→ E.g. ISP to understand what user does

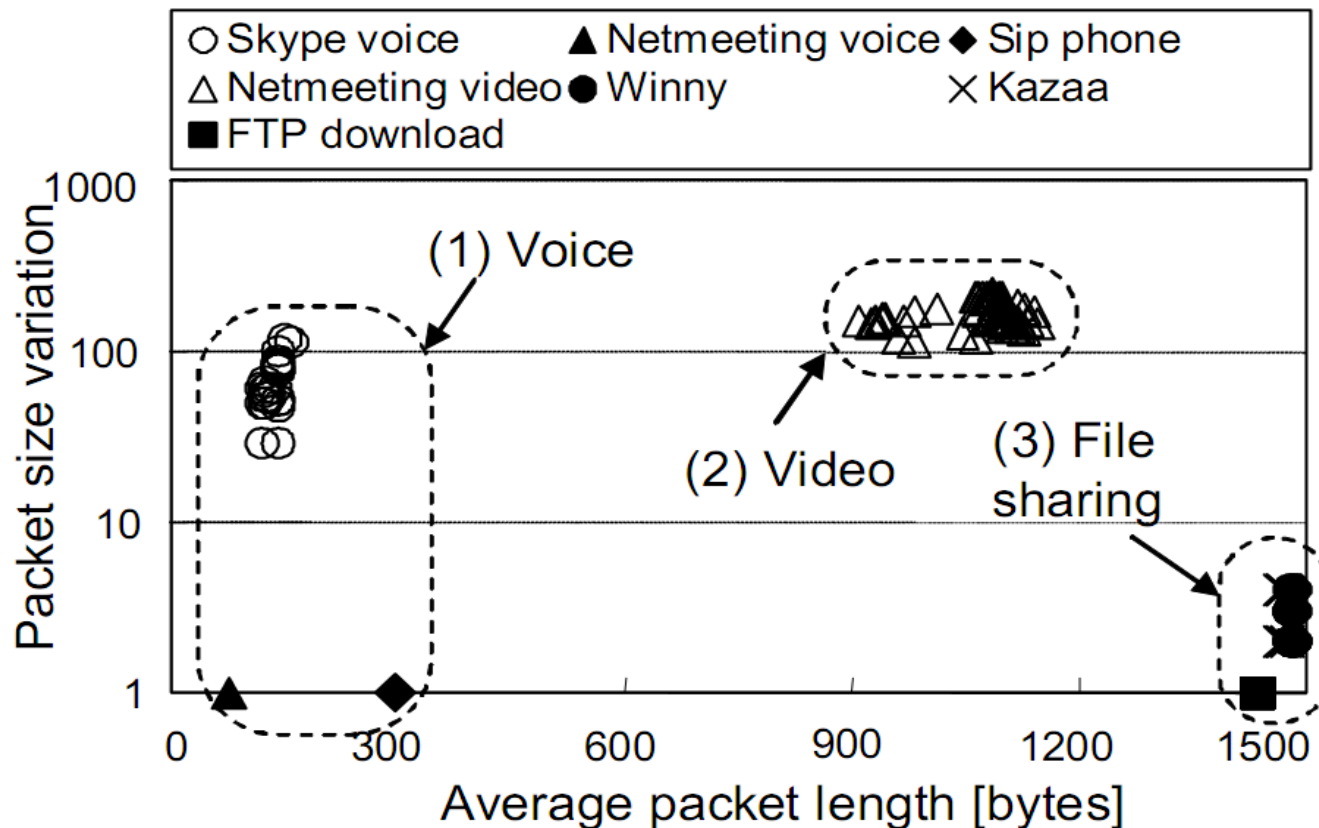


Image from Okabe et al.: Statistical Traffic Identification Method Statistical Traffic Identification Method

csaba.kiraly@disi.unitn.it

Traffic Flow Confidentiality (TFC)

Protection against (statistical) traffic analysis

⇒ Basic tools

→ packet padding, delaying, etc.

⇒ Control logic

→ how to modify traffic pattern

Traffic Flow Confidentiality

Basic tools

Traffic Flow Confidentiality = artificial alteration of traffic statistics

Basic Tools

→ changing packet size

- ⇒ Packet padding
- ⇒ Packet fragmentation

→ changing packet timing

- ⇒ Forwarding delay changes

→ Adding extra traffic

- ⇒ Dummy packet generation

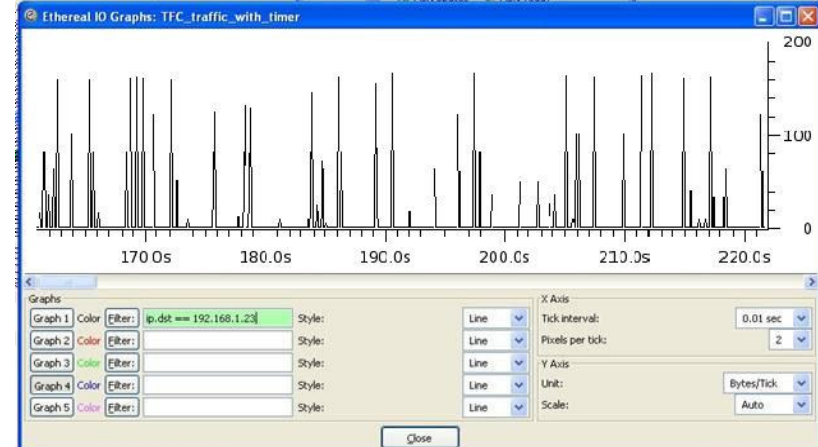
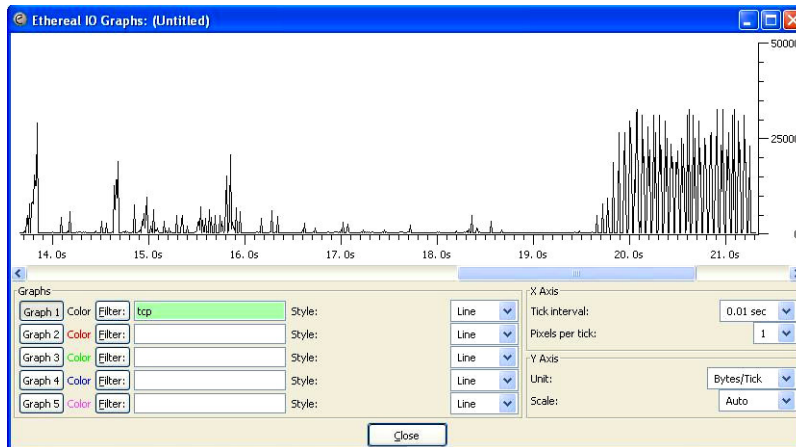
& drawbacks

- ⇒ Increase congestion
- ⇒ Increase latency, reduce reliability
- ⇒ Increase latency
- ⇒ Increase congestion

Performance/security trade-off!

Traffic Flow Confidentiality Control Logic

- **Algorithms combining basic tools to shape traffic, e.g.**
 - ⇒ Constant Packet Rate, padding to fixed size: protects well, large overhead
 - ⇒ Only padding: protects only packet size, relatively low overhead
 - ⇒ Etc.



- **Challenges:**
 - Devise low congestion TFC algorithms
 - Measure privacy/performance trade-offs