Privacy Enhancing Technologies Anonymous Communications.

George Danezis (g.danezis@ucl.ac.uk)

With help from:

Luca Melis (luca.melis.14@ucl.ac.uk)

Steve Dodier-Lazaro (s.dodier-lazaro.12@ucl.ac.uk)

Administration & Labs

- Enrol into the moodle for M067/GA17.
  - Enrolment key “pets”.
  - Information, slides & links …
- State of the Labs.
  - More documentation on petlib: http://petlib.readthedocs.org/en/latest/
  - A “readme” with command line help and hints and help: https://github.com/gdanezis/PET-Exercises/blob/master/Lab01Basics/Lab01Readme.txt
  - Unit tests grouped by task to help you manage them.

  (“git pull” will update your exercises directory in the VM)
- Labs: How are you doing?

Network identity today
Neither privacy nor authenticity / integrity

No anonymity

- Weak identifiers everywhere:
  - IP, MAC
  - Logging at all levels
  - Login names / authentication
  - PK certificates in clear
- Also:
  - Location data leaked
  - Application data leakage

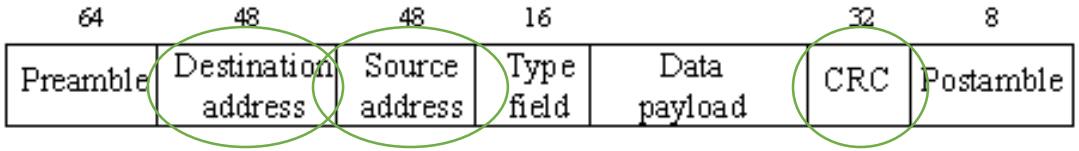
No identification

- Weak identifiers easy to modulate
  - Expensive / unreliable logs.
  - IP / MAC address changes
  - Open Wi-Fi access points
  - Bot-nets
- Partial solution
  - Authentication
- Open issues:
  - DoS and network level attacks

Ethernet packet format

Anthony F. J. Levi - `http://www.usc.edu/dept/engineering/eleceng/Adv_Network_Tech/Html/datacom/`

Ethernet Frame Format



No integrity or authenticity

64 bit preamble is sequence of alternating 1 and 0 for receiver synchronization with signal.

MAC Address

Every Ethernet adapter attached to a host has a unique 6-Byte address e.g. 8:0:2b:e4:b1:2 is 0001000:00000000:00101011:11100100:10110001:00000010

Ethernet standard defined by Xerox, DEC and Intel in 1978 uses 16 bit type field for demultiplexing to frame to higher level protocols. IEEE 802.3 standard uses this field to determine how long the frame is.

Maximum data payload is 1500 Byte.

Cyclic Redundancy Code (CRC-32) is used for error checking.

Postamble indicates end of frame.

IP packet format

RFC: 791

INTERNET PROTOCOL

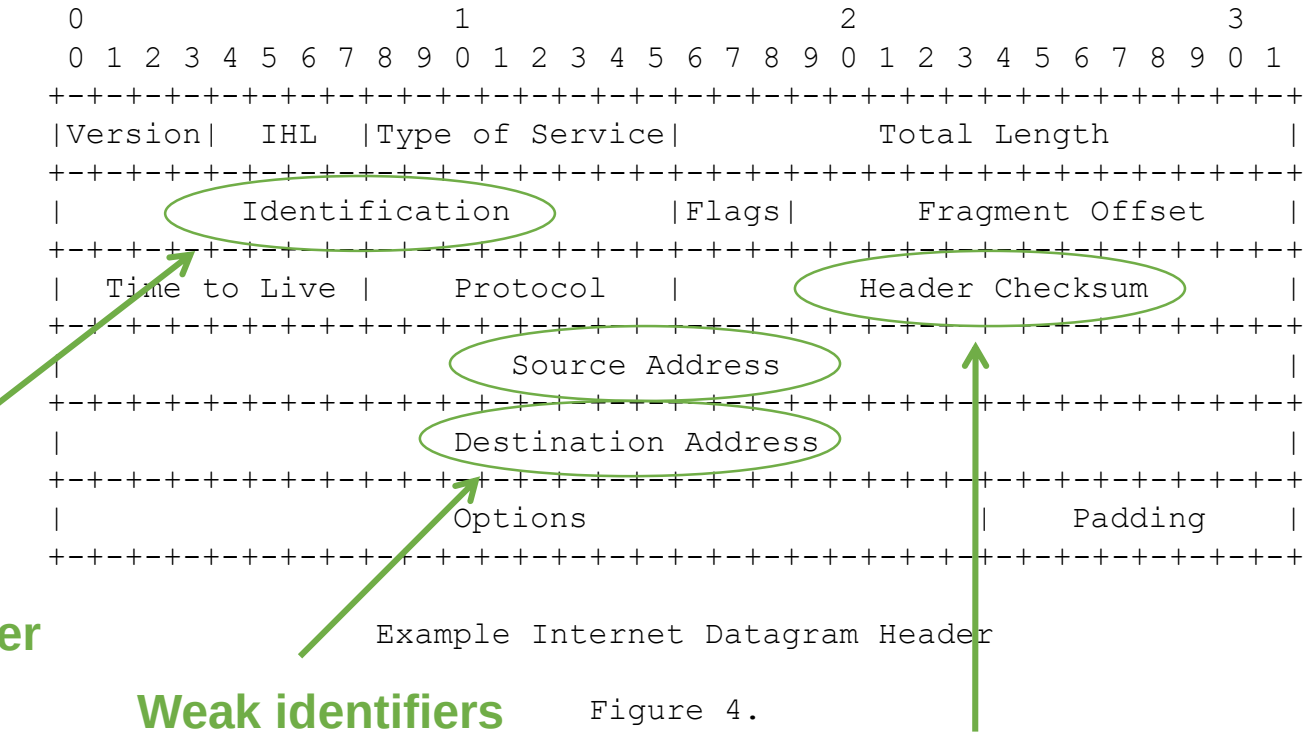
DARPA INTERNET PROGRAM

PROTOCOL SPECIFICATION

September 1981

3.1. Internet Header Format

A summary of the contents of the internet header follows:



Example Internet Datagram Header

Figure 4.

Link different packets together

Weak identifiers

No integrity / authenticity

Same for TCP, SMTP, IRC, HTTP, ...

Anonymity in communications

- Specialized applications
  - Electronic voting
  - Auctions / bidding / stock market
  - Incident reporting
  - Witness protection / whistle blowing
  - Showing anonymous credentials!

- General applications
  - Freedom of speech
  - Profiling / price discrimination
  - Spam avoidance
  - Investigation / market research
  - Censorship resistance

Anonymity properties (1)

- Sender anonymity
  - Alice sends a message to Bob. Bob cannot know who Alice is.
- Receiver anonymity
  - Alice can send a message to Bob, but cannot find out who Bob is.
- Bi-directional anonymity
  - Alice and Bob can talk to each other, but neither of them know the identity of the other.

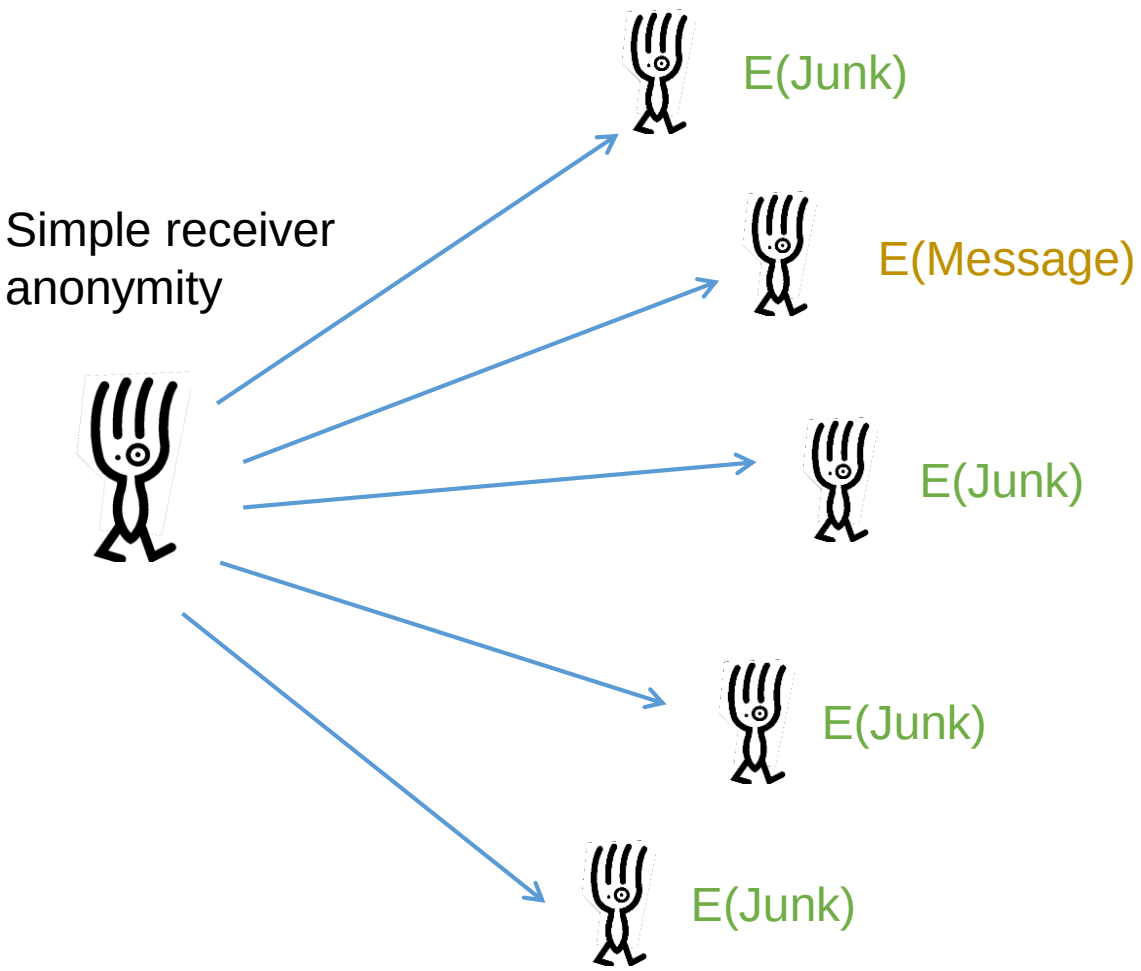
Anonymity properties (2)

- 3rd party anonymity
  - Alice and Bob converse and know each other, but no third party can find this out.
- Unobservability
  - Alice and Bob take part in some communication, but no one can tell if they are transmitting or receiving messages.
- Unlinkability
  - Two messages sent (received) by Alice (Bob) cannot be linked to the same sender (receiver).
- Pseudonymity
  - All actions are linkable to a pseudonym, which is unlinkable to a principal (Alice)

High-Latency Anonymity Systems

Mix Networks

Anonymity through Broadcast



Point 1: Do not re-invent this

Point 2: Many ways to do broadcast

- Ring
- Trees

It has all been done (Buses)

Point 3: Is your anonymity system better than this?

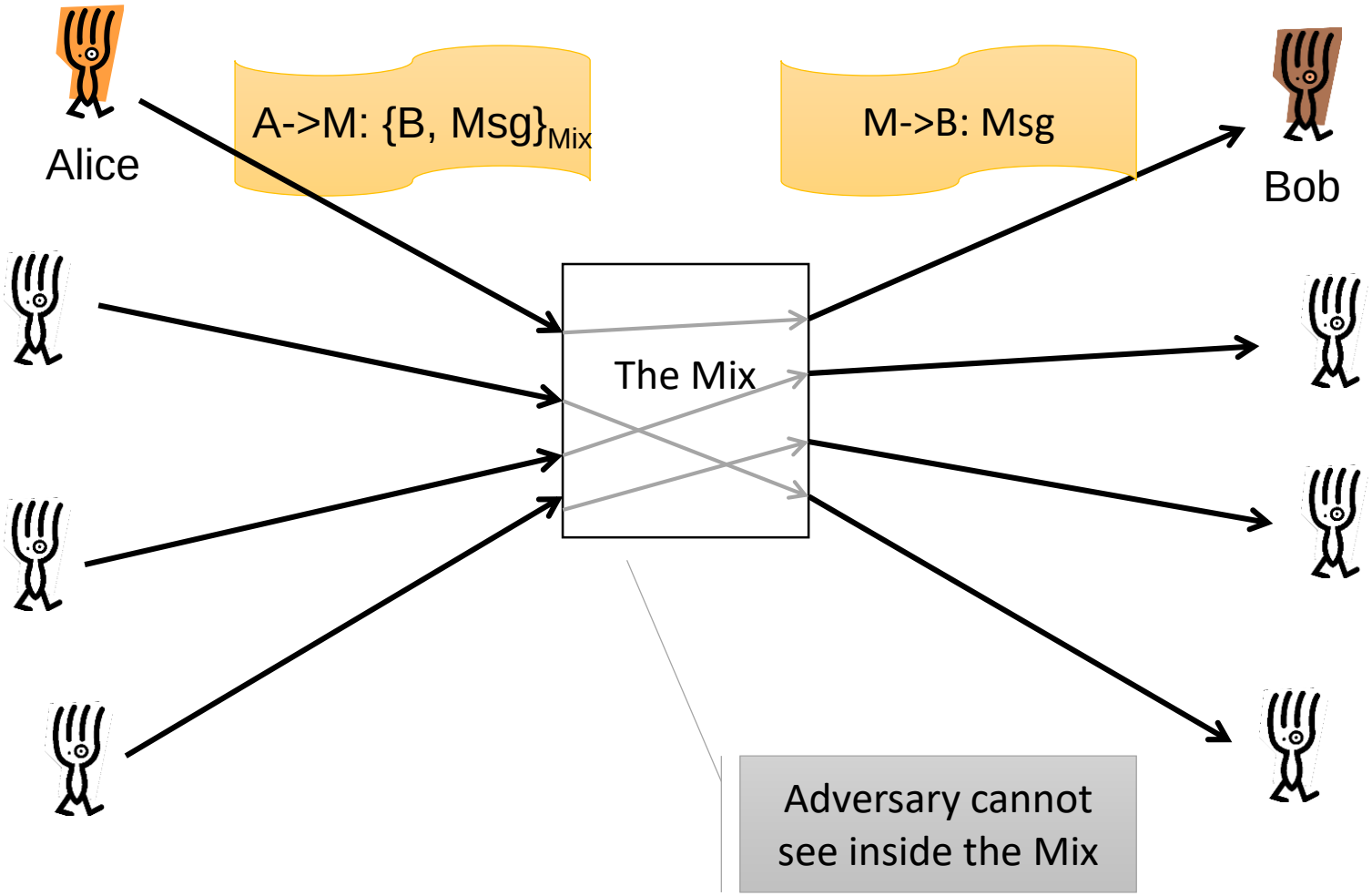
Point 4: What are the problems here?

Coordination
Sender anonymity
Latency
Bandwidth

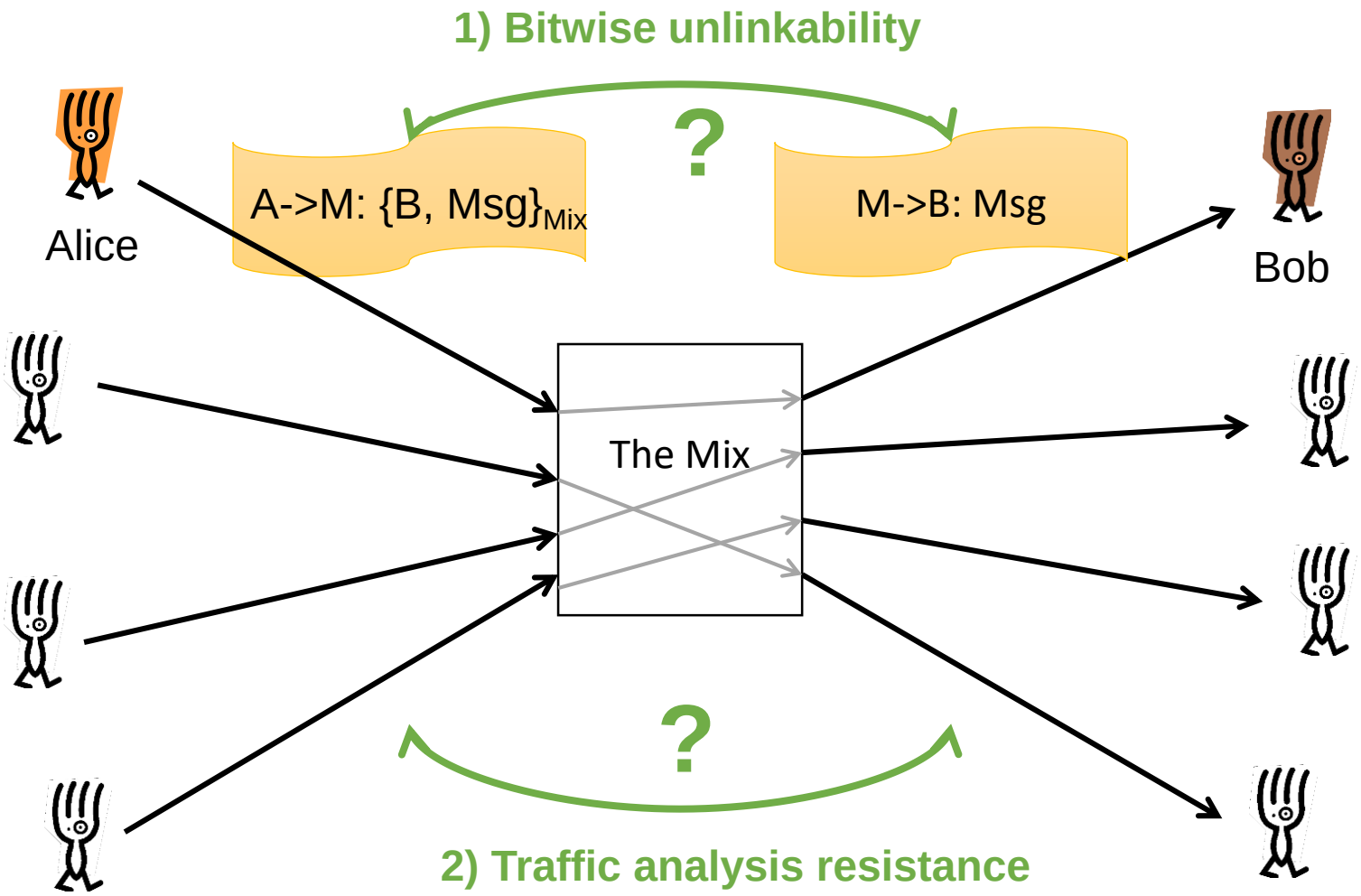
Mix – practical anonymity

- David Chaum (concept 1979 – publish 1981)
  - Reference is marker in anonymity bibliography
- Makes uses of cryptographic relays
  - Break the link between sender and receiver
- Cost
  - O(1) – O(logN) messages
  - O(1) – O(logN) latency
- Security
  - Computational (public key primitives must be secure)
  - Threshold of honest participants

The mix – illustrated



The mix – security issues

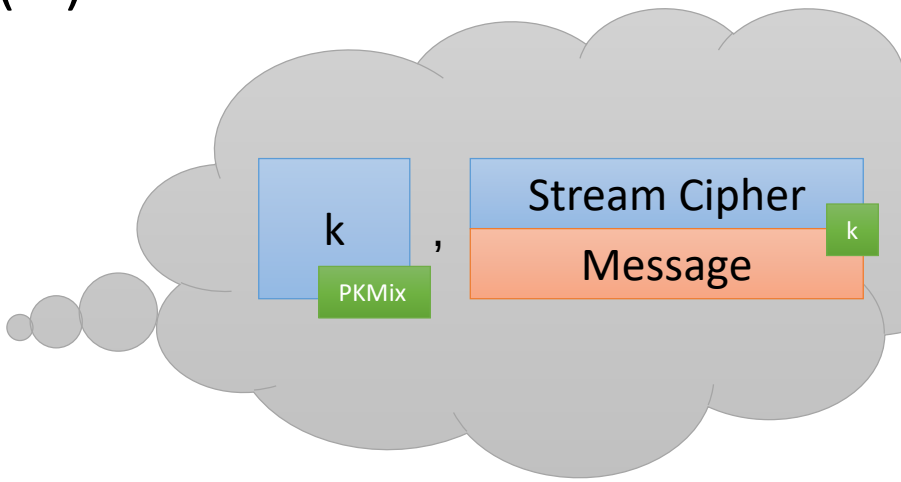


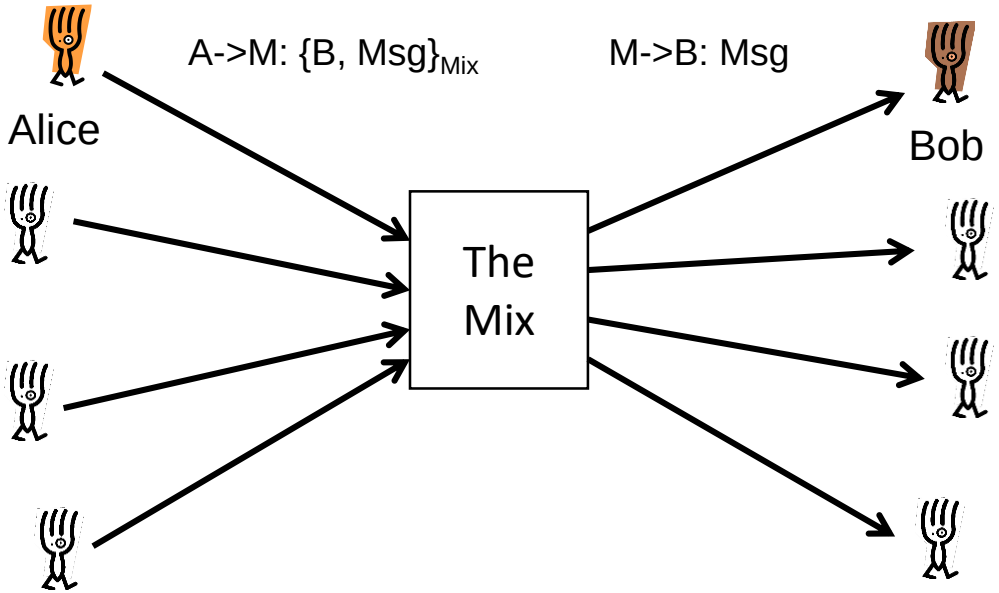
Mix security (contd.)

- Bitwise unlinkability
  - Ensure adversary cannot link messages in and out of the mix from their bit pattern
  - Cryptographic problem
- Traffic analysis resistance
  - Ensure the messages in and out of the mix cannot be linked using any meta-data (timing, ...)
  - Two tools: delay , inject or drop traffic –add cost!

Two broken mix designs (1)

- Broken bitwise unlinkability
  - The `stream cipher' mix (Design 1)
  - $\{M\}_{Mix} = \{\text{fresh } k\}_{PKmix}, M \oplus \text{Stream}_k$





- Active attack?

Tagging Attack

Adversary intercepts $\{B, Msg\}_{Mix}$ and injects $\{B, Msg\}_{Mix} \oplus (0, Y)$.

The mix outputs message:

M->B: Msg $\oplus$ Y

And the attacker can link them.

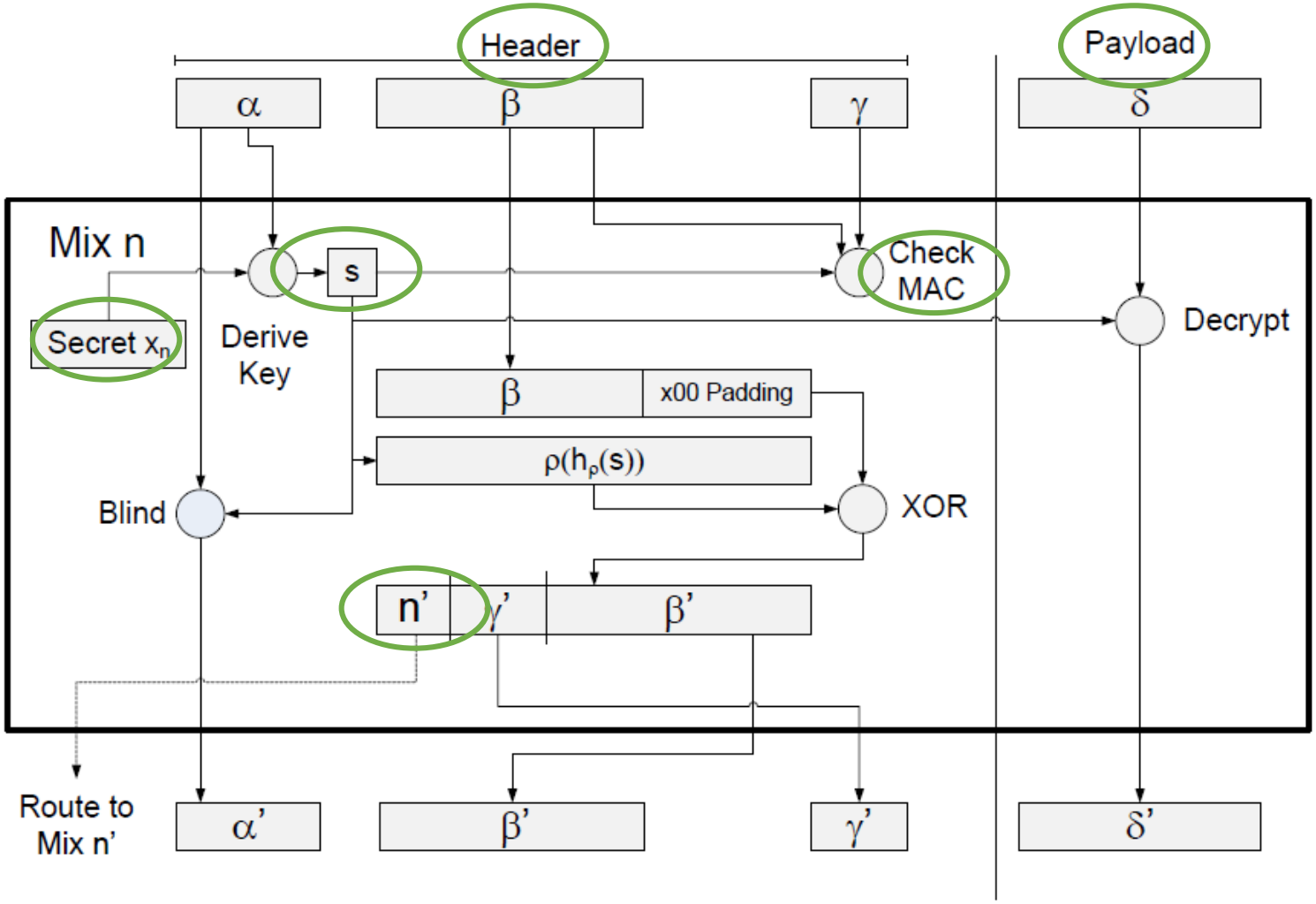
Lessons from broken design 1

- Mix acts as a service
  - Everyone can send messages to it; it will apply an algorithm and output the result.
  - That includes the attacker – decryption oracle, routing oracle, ...
- (Active) Tagging attacks
  - Defence 1: detect modifications (CCA2)
  - Defence 2: destroy all information (Mixminion, Minx)

GA17 Lab 2 – Implement a simple mix client

- Note: the second lab will be on implementing a simple mix client, and fixing aspects of a mix server.

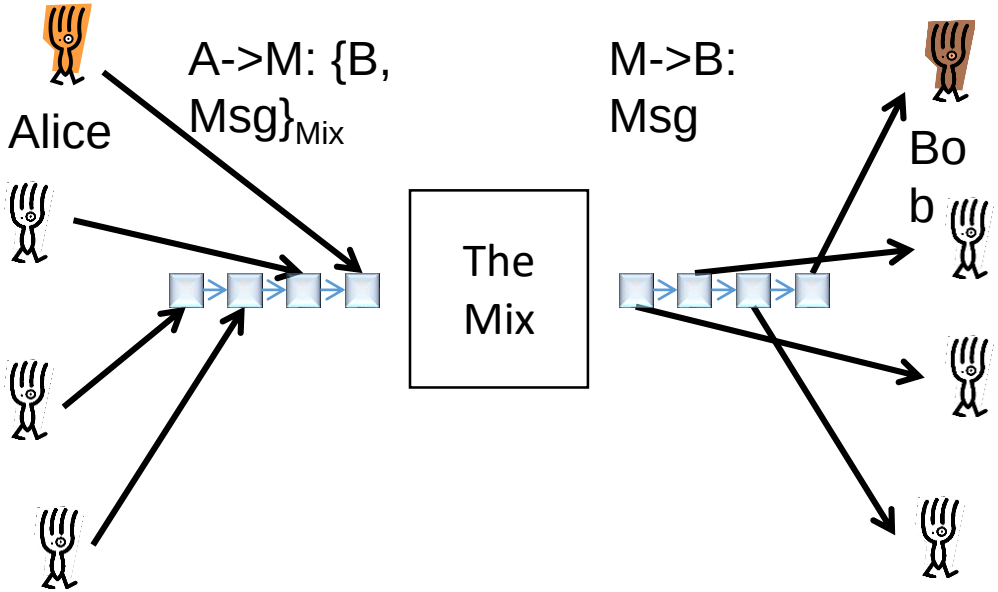
Insight into a Modern message format



George Danezis & Ian Goldberg. Sphinx: A Compact and Provably Secure Mix Format. IEEE S&P '09.

Two broken mix designs (2)

- Broken traffic analysis resistance
  - The `FIFO*' mix (Design 2)
  - Mix sends messages out in the order they came in!



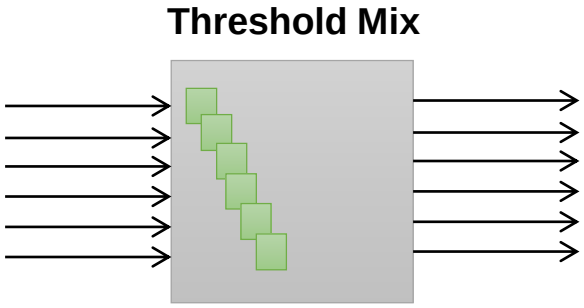
* FIFO = First in, First out

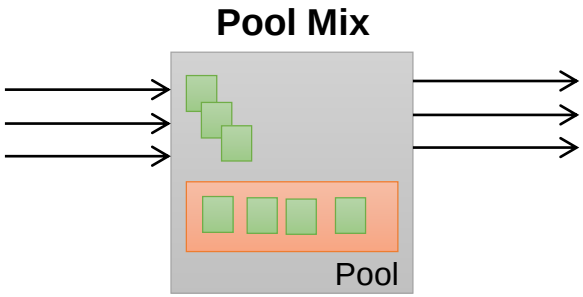
- Passive attack?

The adversary simply counts the number of messages, and assigns to each input the corresponding output.

Lessons from broken design 2

- Mix strategies – 'mix' messages together
  - Threshold mix: wait for N messages and output them in a random order.
  - Pool mix: Pool of n messages; wait for N inputs; output N out of N+n; keep remaining n in pool.
  - Timed, random delay, ...



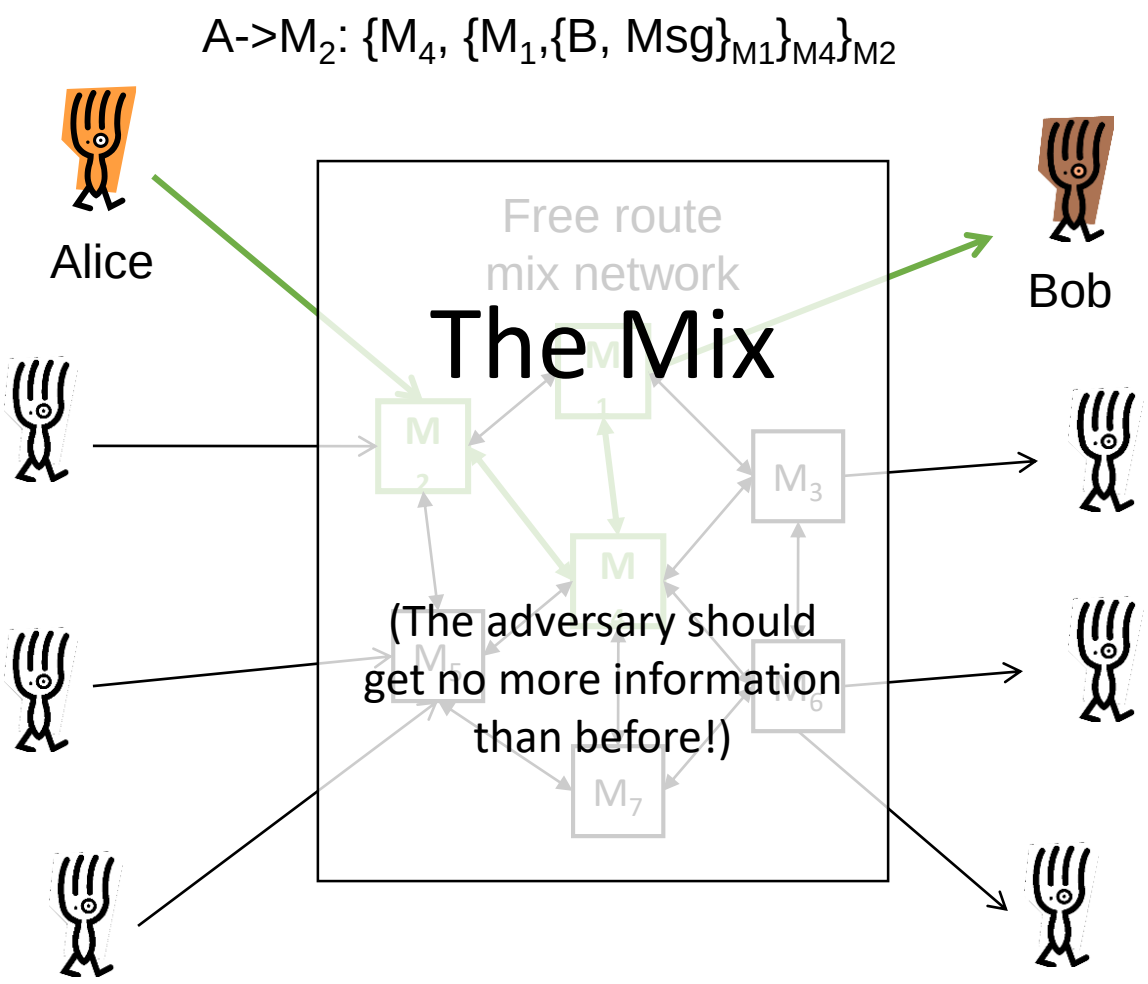


- "Hell is other people" – J.P. Sartre
  - Anonymity security relies on *others*
  - Problem 1: Mix must be honest
  - Problem 2: Other honest sender-receiver pairs to hide amongst

Distributing mixing

- Rely on more mixes – good idea
  - Distributing trust – some could be dishonest
  - Distributing load – fewer messages per mix
- Two extremes
  - Mix Cascades
    - All messages are routed through a preset mix sequence
    - Good for anonymity – poor load balancing
  - Free routing
    - Each message is routed through a random sequence of mixes
    - Security parameter: L then length of the sequence

The free route example

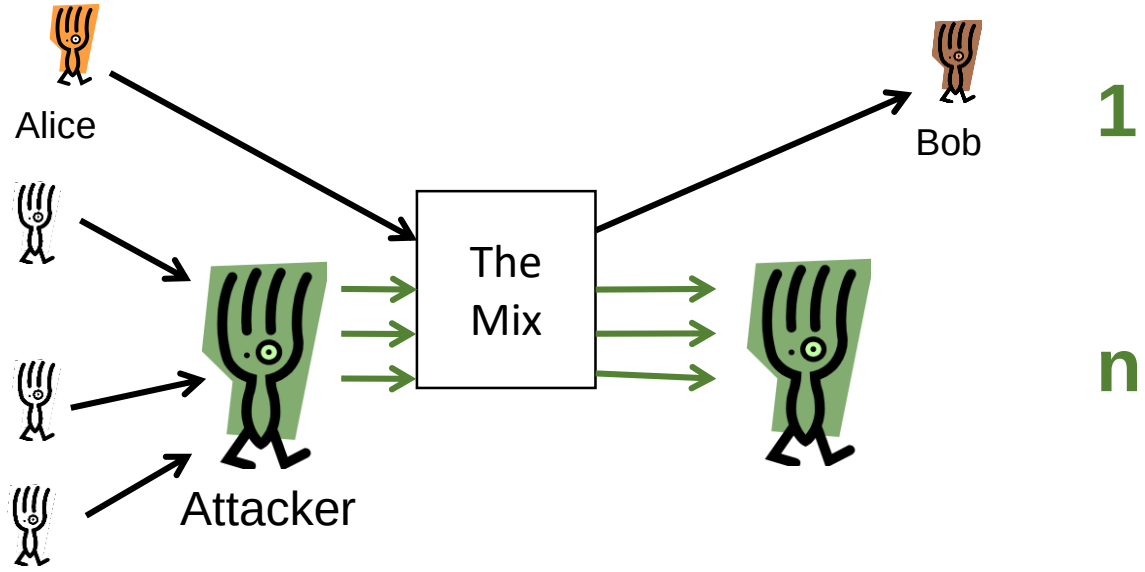


Free route mix networks

- Bitwise unlinkability
  - Length invariance
  - Replay prevention
- How to find mixes?
  - Lists need to be authoritative, comprehensive & common
- Additional requirements – corrupt mixes
  - Hide the total length of the route
  - Hide the step number
  - (From the mix itself!)
- Length of paths?
  - Good mixing in O(log(|Mix|)) steps = log(|Mix|) cost
  - Cascades: O(|Mix|)
- We can manage “Problem 1 – trusting a mix”

Problem 2 – who are the others?

- The (n-1) attack – active attack
  - Wait or flush the mix.
  - Block all incoming messages (trickle) and injects own messages (flood) until Alice's message is out.



Mitigating the (n-1) attack

- Strong identification to ensure distinct identities
  - Problem: user adoption
- Message expiry
  - Messages are discarded after a deadline
  - Prevents the adversary from flushing the mix, and injecting messages unnoticed
- Heartbeat traffic
  - Mixes route messages in a loop back to themselves
  - Detect whether an adversary is blocking messages
  - Forces adversary to subvert everyone, all the time
- General instance of the “Sybil Attack”

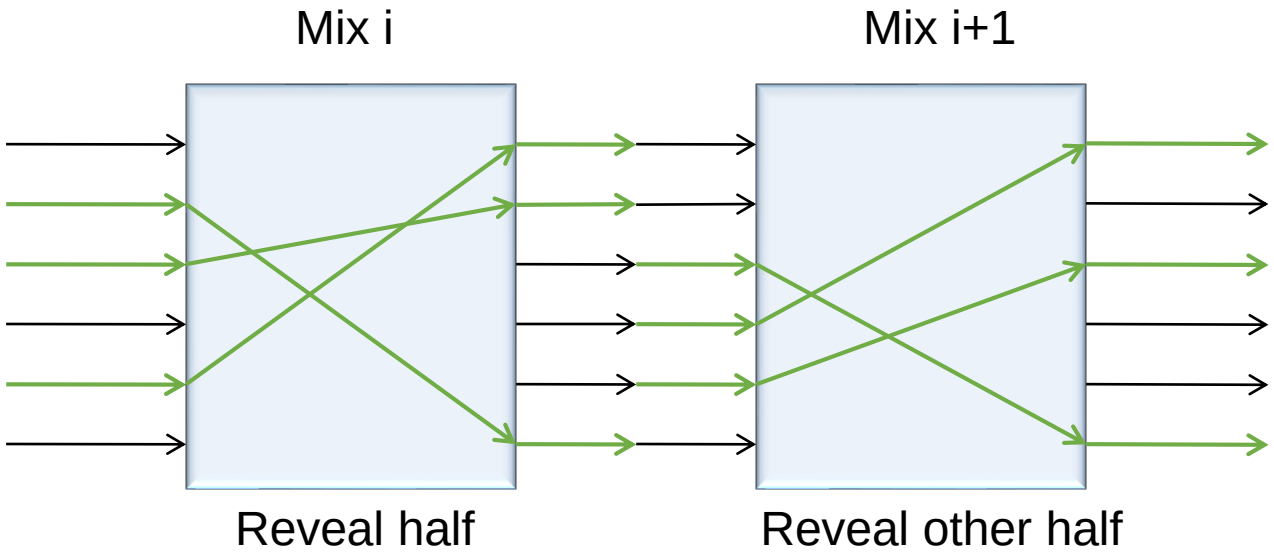
Robustness to Denial of Service (DoS)

- Malicious mixes may be dropping messages
  - Special problem in elections
- Original idea: receipts (unworkable)
- Two key strategies to prevent DoS
  - Provable shuffles – will not cover here.
  - Randomized partial checking

Randomized partial checking

- Applicable to any mix system
- Two round protocol
  - Mix commits to inputs and outputs
  - Gets challenge
  - Reveals half of correspondences at random
  - Everyone checks correctness
- Pair mixes to ensure messages get some anonymity

Partial checking – illustrated



- Rogue mix can cheat with probability at most ½

Slight lie

- Messages are anonymous with overwhelming probability in the length L
  - Even if no pairing is used – safe for L = O(logN)

Receiver anonymity

- Cryptographic reply address
  - Alice sends to bob: $M_1,\{M_2, k_1,\{A,\{K\}_A\}_{M2}\}_{M1}$
    - Memory-less: $k_1 = H(K, 1)$ $k_2 = H(K, 2)$
  - Bob replies:
    - B->M1: $\{M_2, k_1, \{A,\{K\}_A\}_{M2}\}_{M1}$, Msg
    - M1->M2: $\{A,\{K\}_A\}_{M2}$, $\{Msg\}_{k1}$
    - M2->A: $\{K\}_A$, $\{\{Msg\}_{k1}\}_{k2}$
  - Security: indistinguishable from other messages

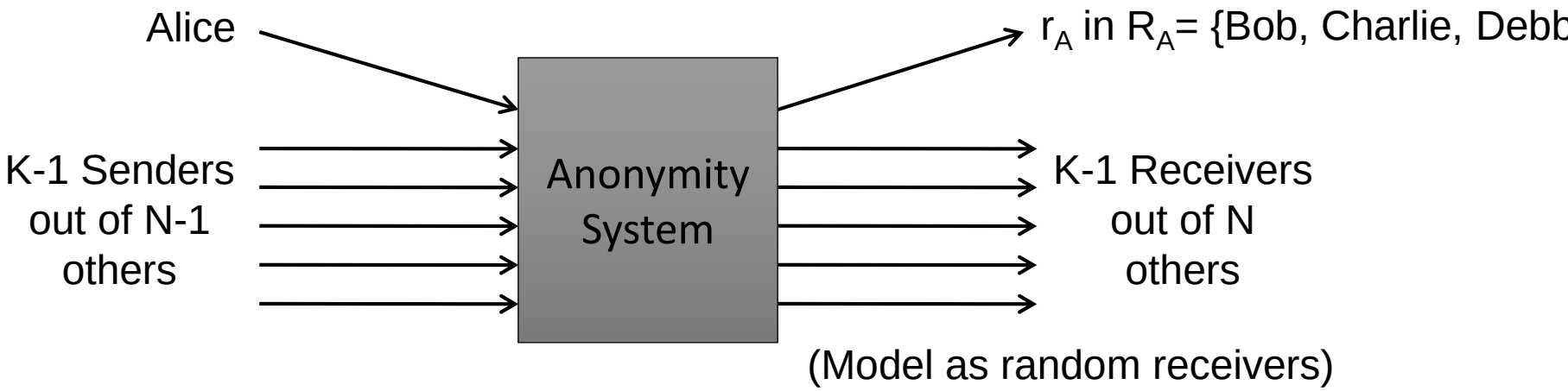
Basic Traffic Analysis

Anonymity is more fragile than communications privacy!

Fundamental limits

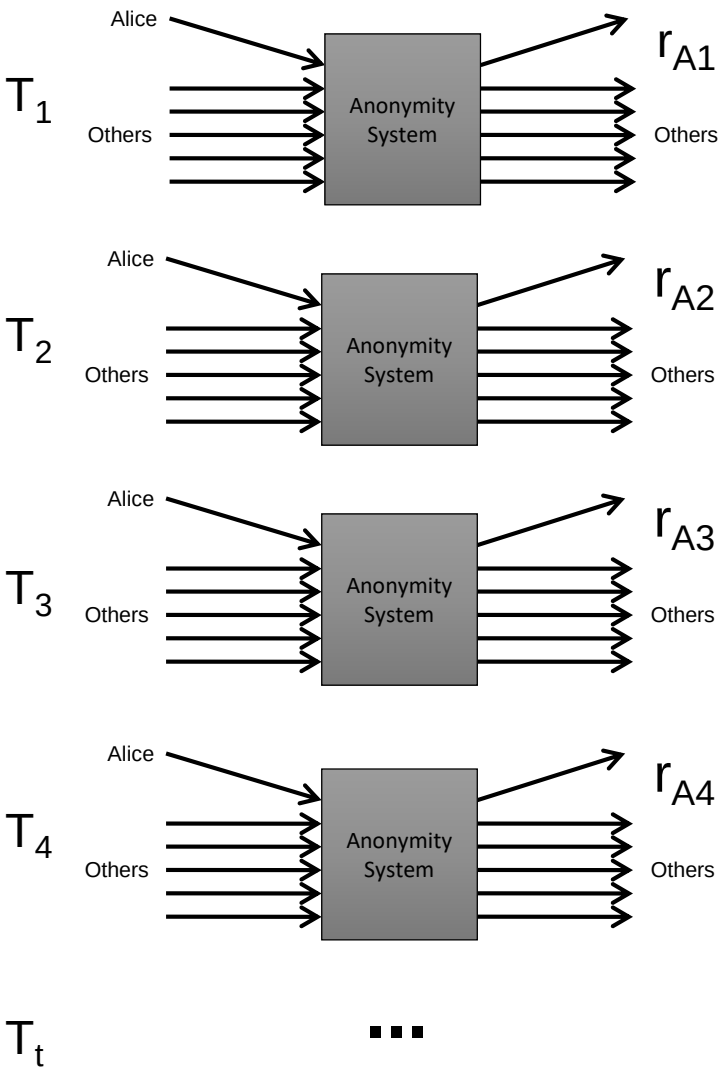
- What is traffic analysis:
  - The discipline of inferring information from patterns of traffic.
  - Applied to security: assume content is encrypted; traffic is anonymized: Traffic analysis of “hardened targets”
- Even perfect anonymity systems leak information when participants change
- Setting:
  - N senders / receivers – Alice is one of them
  - Alice messages a small number of friends:
    - R_A in {Bob, Charlie, Debbie}
    - Through a MIX / DC-net
    - Perfect anonymity of size K
  - Can we infer Alice’s friends?

Setting



- Alice sends a single message to one of her friends
- Anonymity set size = K
  Entropy metric E_A = log K
- Perfect!

Many rounds



- Observe many rounds in which Alice participates
- Rounds in which Alice participates will output a message to her friends!
- Infer the set of friends!

Hitting set attack (1)

- Guess the set of friends of Alice (R_A')
  - Constraint $|R_A'| = m$
- Accept if an element is in the output of each round
- Downside: Cost
  - N receivers, m size – (N choose m) options
  - Exponential – Bad
- Good approximations...

Statistical disclosure attack

- Note that the friends of Alice will be in the sets more often than random receivers
- How often? Expected number of messages per receiver:
  - $\mu_{other} = (1 / N) \cdot (K-1) \cdot t$
  - $\mu_{Alice} = (1 / m) \cdot t + \mu_{other}$
- Just count the number of messages per receiver when Alice is sending!
  - $\mu_{Alice} > \mu_{other}$

Comparison: HS and SDA

- **Parameters: N=20 m=3 K=5 t=45** **KA={[0, 13, 19]}**

	Round Receivers	SDA		SDA_error	#Hitting sets
1	**[15, 13, 14, 5, 9]**	**[13, 14, 15]**	**2**	**685**	
2	[19, 10, 17, 13, 8]	[13, 17, 19]	1	395	
3	[0, 7, 0, 13, 5]	[0, 5, 13]	1	257	
4	[16, 18, 6, 13, 10]	[5, 10, 13]	2	203	
5	[1, 17, 1, 13, 6]	[10, 13, 17]	2	179	
6	[18, 15, 17, 13, 17]	[13, 17, 18]	2	175	
7	[0, 13, 11, 8, 4]	[0, 13, 17]	1	171	
8	[15, 18, 0, 8, 12]	[0, 13, 17]	1	80	
9	[15, 18, 15, 19, 14]	[13, 15, 18]	2	41	
10	[0, 12, 4, 2, 8]	[0, 13, 15]	1	16	
11	[9, 13, 14, 19, 15]	[0, 13, 15]	1	16	
12	[13, 6, 2, 16, 0]	[0, 13, 15]	1	16	
13	[1, 0, 3, 5, 1]	[0, 13, 15]	1	4	
14	[17, 10, 14, 11, 19]	[0, 13, 15]	1	2	
15	[12, 14, 17, 13, 0]	[0, 13, 17]	1	2	
16	**[18, 19, 19, 8, 11]**	**[0, 13, 19]**	**0**		
17	[4, 1, 19, 0, 19]	[0, 13, 19]	0	1	
18	[0, 6, 1, 18, 3]	[0, 13, 19]	0	1	
19	[5, 1, 14, 0, 5]	[0, 13, 19]	0	1	
20	[17, 18, 2, 4, 13]	[0, 13, 19]	0	1	
21	[8, 10, 1, 18, 13]	[0, 13, 19]	0	1	
22	[14, 4, 13, 12, 4]	[0, 13, 19]	0	1	

Round 16:
Both attacks give correct result

SDA: Can give wrong results – need more evidence

HS and SDA (continued)

25	[19, 4, 13, 15, 0]	[0, 13, 19]	0	1
26	[13, 0, 17, 13, 12]	[0, 13, 19]	0	1
27	[11, 13, 18, 15, 14]	[0, 13, 18]	1	1
28	[19, 14, 2, 18, 4]	[0, 13, 18]	1	1
29	[13, 14, 12, 0, 2]	[0, 13, 18]	1	1
30	[15, 19, 0, 12, 0]	[0, 13, 19]	0	1
31	[17, 18, 6, 15, 13]	[0, 13, 18]	1	1
32	[10, 9, 15, 7, 13]	[0, 13, 18]	1	1
33	[19, 9, 7, 4, 6]	[0, 13, 19]	0	1
34	[19, 15, 6, 15, 13]	[0, 13, 19]	0	1
35	[8, 19, 14, 13, 18]	[0, 13, 19]	0	1
36	[15, 4, 7, 13, 13]	[0, 13, 19]	0	1
37	[3, 4, 16, 13, 4]	[0, 13, 19]	0	1
38	[15, 13, 19, 15, 12]	[0, 13, 19]	0	1
39	[2, 0, 0, 17, 0]	[0, 13, 19]	0	1
40	[6, 17, 9, 4, 13]	[0, 13, 19]	0	1
41	[8, 17, 13, 0, 17]	[0, 13, 19]	0	1
42	[7, 15, 7, 19, 14]	[0, 13, 19]	0	1
43	[13, 0, 17, 3, 16]	[0, 13, 19]	0	1
44	[7, 3, 16, 19, 5]	[0, 13, 19]	0	1
45	[13, 0, 16, 13, 6]	[0, 13, 19]	0	1

SDA: Can give wrong results – need more evidence

Disclosure attack family

- Counter-intuitive
  - The larger N the easiest the attack
- Hitting-set attacks
  - More accurate, need less information
  - Slower to implement
  - Sensitive to Model
    - E.g. Alice sends dummy messages with probability p.
- Statistical disclosure attacks
  - Need more data
  - Very efficient to implement (vectorised) – Faster partial results
  - Can be extended to more complex models (pool mix, replies, ...)
- The Future: Bayesian modelling of the problem

Summary of key points

- Near-perfect anonymity is not perfect enough!
  - High level patterns cannot be hidden for ever
  - Unobservability / maximal anonymity set size needed
- Flavours of attacks
  - Very exact attacks – expensive to compute
    - Model inexact anyway
  - Statistical variants – wire fast!

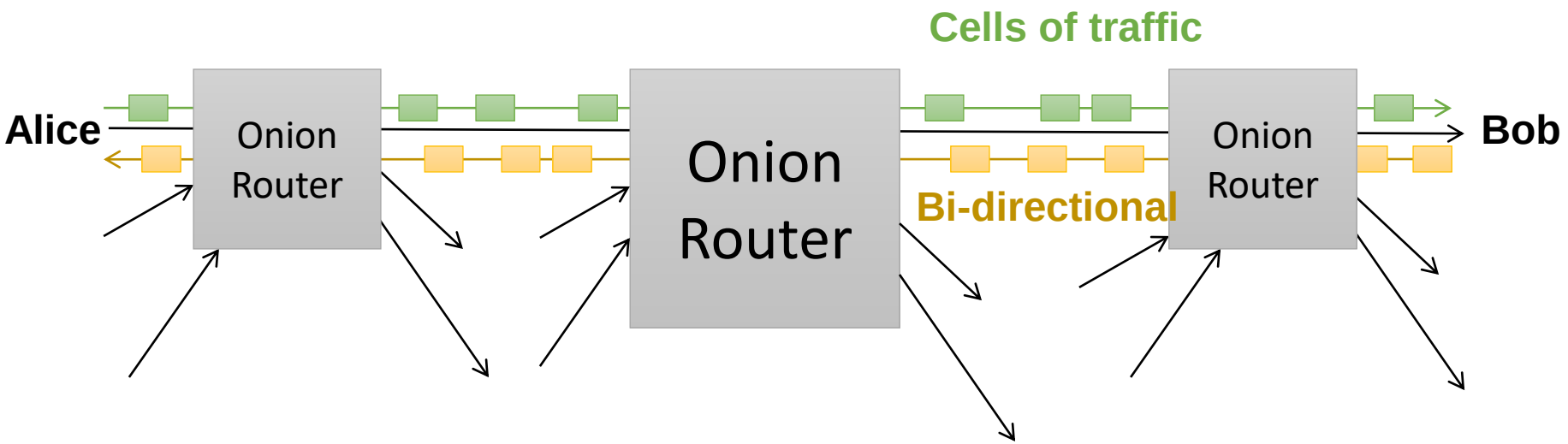
Low Latency Anonymity Systems

Tor, and all that …

Onion Routing

- Anonymising streams of messages
  - Example: Tor (The onion router)

- As for mix networks
  - Alice chooses a (short) path
  - Relays a bi-directional stream of traffic to Bob



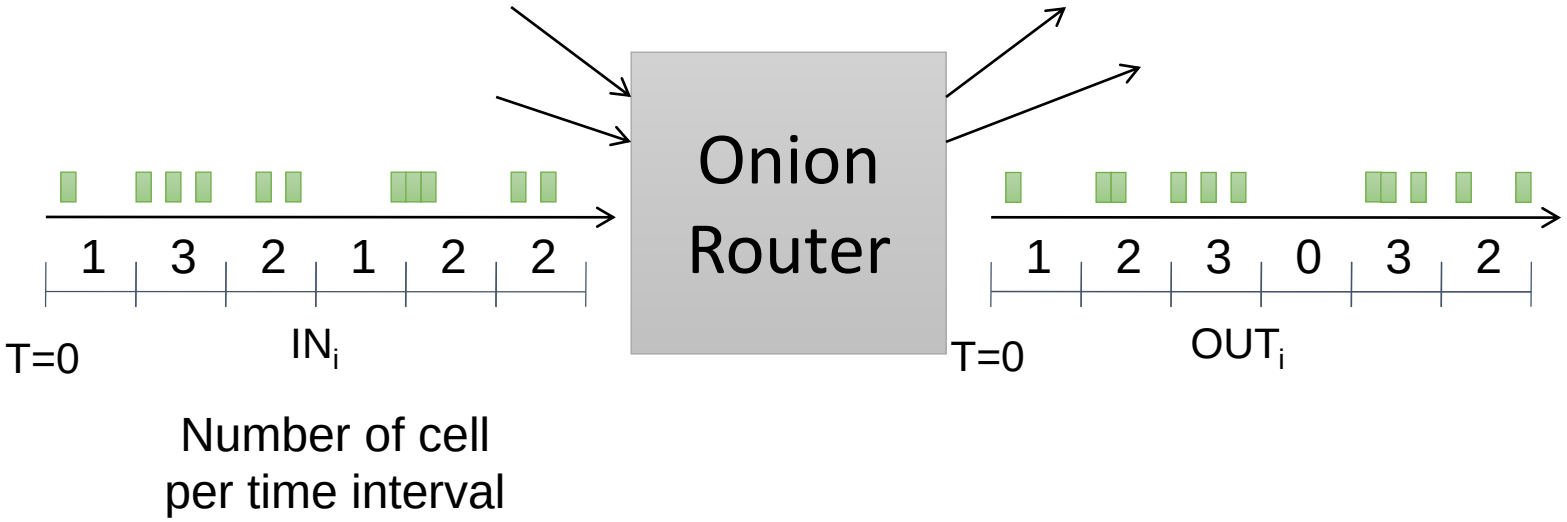
Onion Routing vs. Mixing

- Setup route once per connection
  - Use it for many cells – save on public key operations
- No time for delaying
  - Usable web latency 1—2 sec round trip
  - Short routes – Tor default 3 hops
  - No batching (no threshold , ...)
- Passive attacks!

Stream Tracing

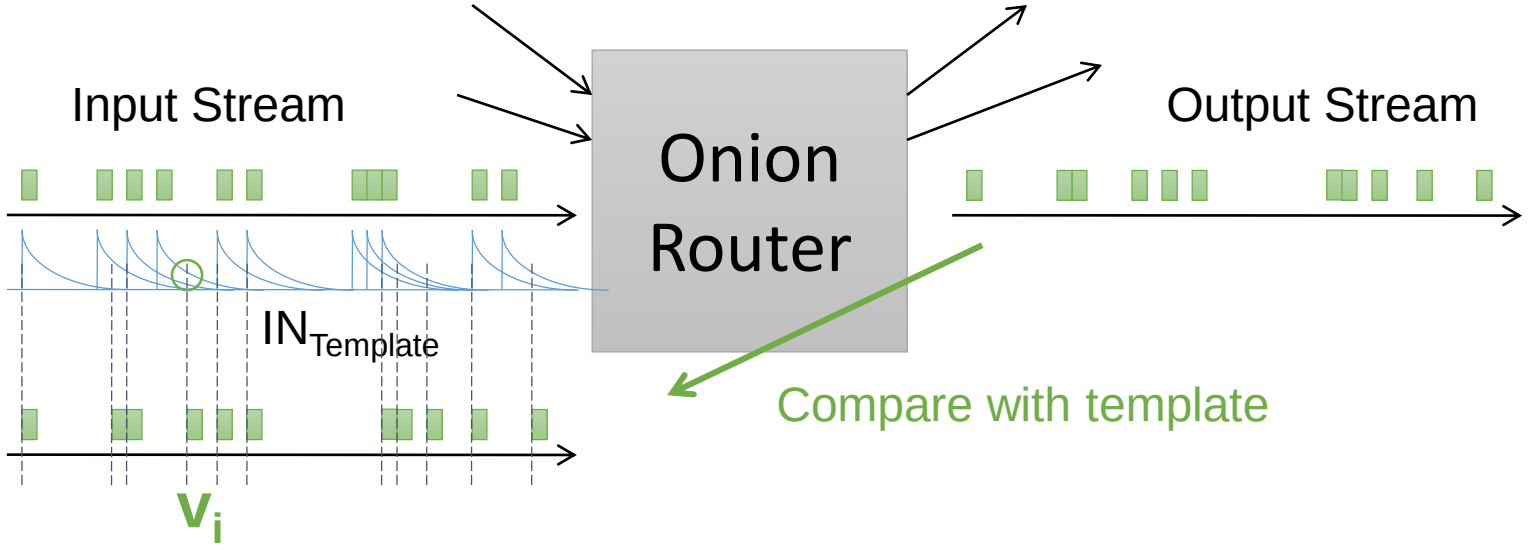
- Adversary observes all inputs and outputs of an onion router
- Objective: link the ingoing and outgoing connections (to trace from Alice to Bob)
- Key insight: timing of packets are correlated
- Two techniques:
  - Correlation
  - Template matching

Tracing (1) – Correlation



- Bucket input and output packets over time bins. Normalize around zero.
- Compute:
  - Corr = $\sum_i IN_i \cdot OUT_i$
- Downside: lose precision by bucketing.

Tracing (2) – Template matching

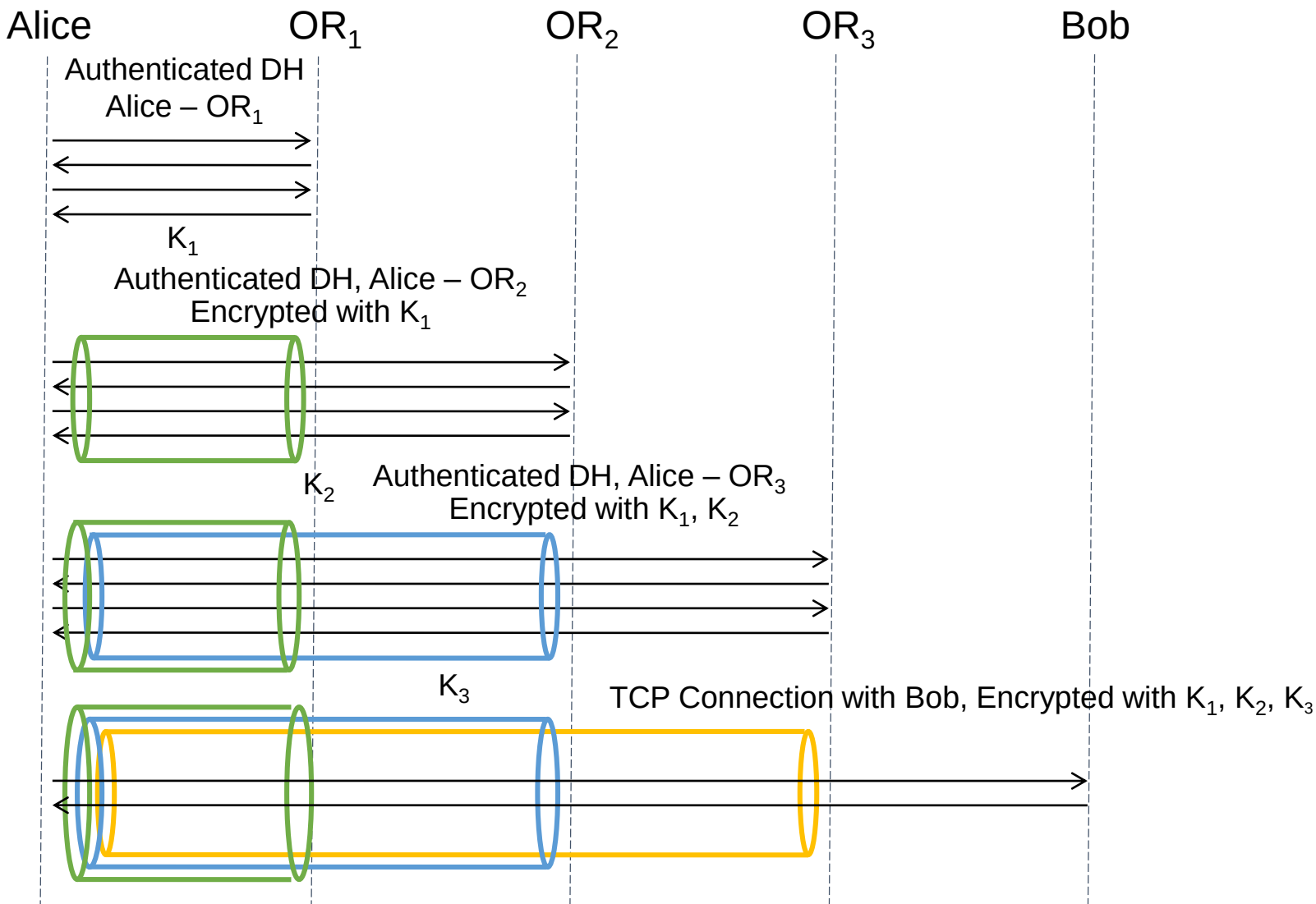


- Use input and delay curve to make template
  - Prediction of what the output will be
- Assign to each output cell the template value (v_i) for its output time
- Multiply them together to get a score ($\prod_i v_i$)

The security of Onion Routing

- Cannot withstand a global passive adversary
  - (Tracing attacks too expensive to foil)
- Partial adversary
  - Can see *some* of the network
  - Can control *some* of the nodes
- Secure if adversary cannot see first and last node of the connection
  - If c is fraction of corrupt servers
  - Compromise probability = $O(c^2)$
- No point making routes too long

Extending the route in Tor



Some remarks

- Encryption of input and output streams under different keys provides bitwise unlinkability
  - As for mix networks
  - Is it really necessary?

- Authenticated Diffie-Hellman
  - One-sided authentication: Alice remains anonymous
  - Alice needs to know the signature keys of the Onion Routers
  - Scalability issue – 1000 routers x 2048 bit keys
  - Advantage: **Perfect Forward Secrecy!**

Summary of key concepts on Anonymity

- Anonymity requires a crowd
  - Difficult to ensure it is not simulated – (n-1) attack
  - Making one on your own expensive (broadcast)

- Mix networks – Practical anonymous messaging
  - Bitwise unlinkability / traffic analysis resistance
  - Crypto: Decryption vs. Re-encryption mixes
  - Distribution: Cascades vs. Free route networks
  - Robustness: Partial checking

- Onion Routing – Supports interactive streams
  - Only withstands a partial adversary.
  - Very widely deployed (Tor: The onion router)