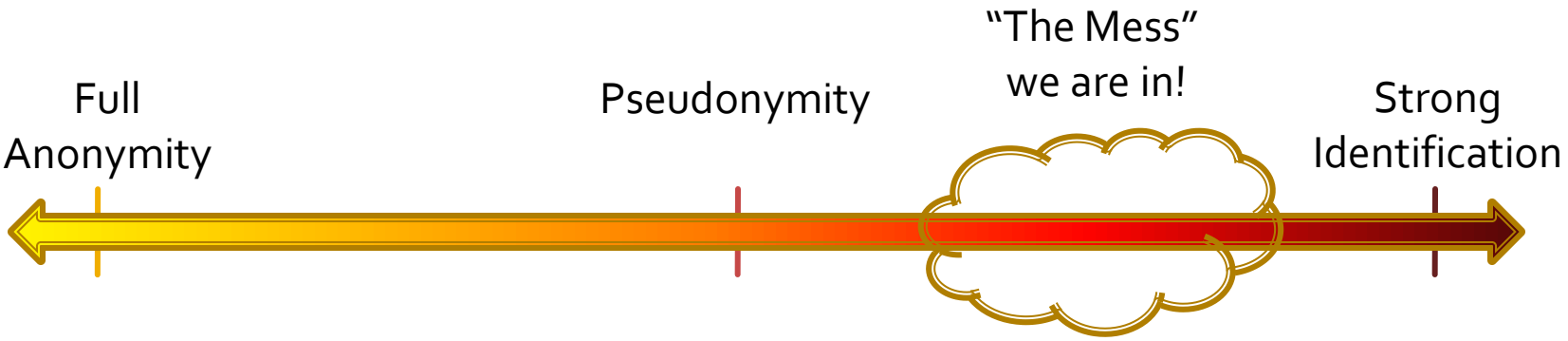
Anonymous communications: High latency systems

Anonymous email and messaging and their traffic analysis

Network identity today

- Networking
  - Relation between identity and efficient routing
  - Identifiers: MAC, IP, email, screen name
  - No network privacy = no privacy!
- The identification spectrum today



Network identity today (contd.)

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

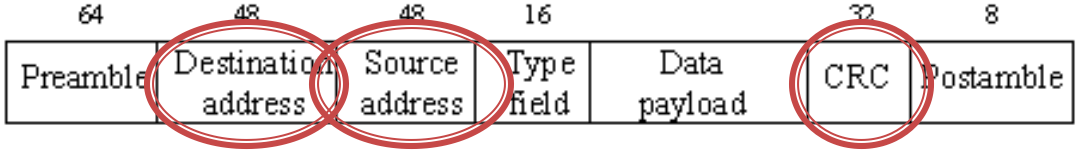
NO IDENTIFICATION

- Weak identifiers easy to modulate
  - Expensive / unreliable logs.
  - IP / MAC address changes
  - Open wifi access points
  - Botnets
- 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.

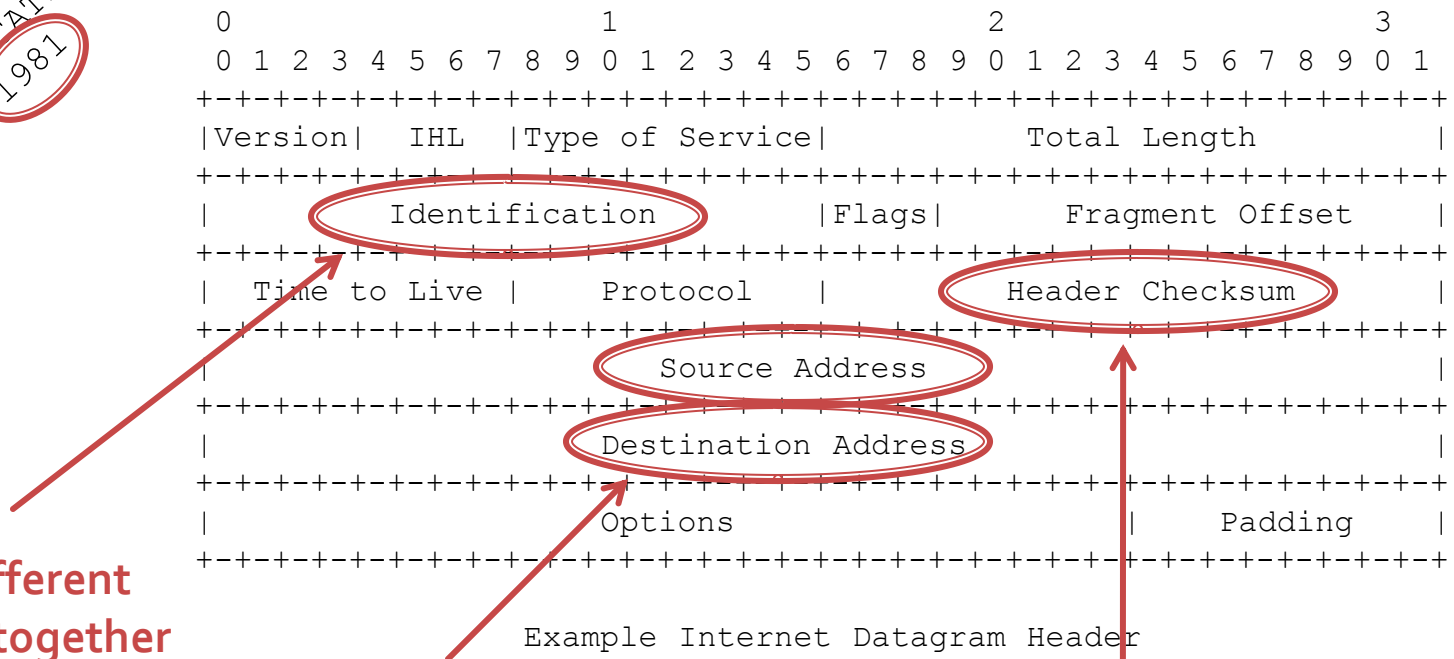
IP packet format

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, …

Outline

- Motivation and properties
- Constructions
  - Unconditional anonymity – DC nets
  - Practical anonymity – Mix networks
  - Practical robustness
- Traffic analysis
  - Measuring anonymity
  - Cryptographic attacks
  - Statistical disclosure attacks

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.

Pseudonymity properties

- 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)

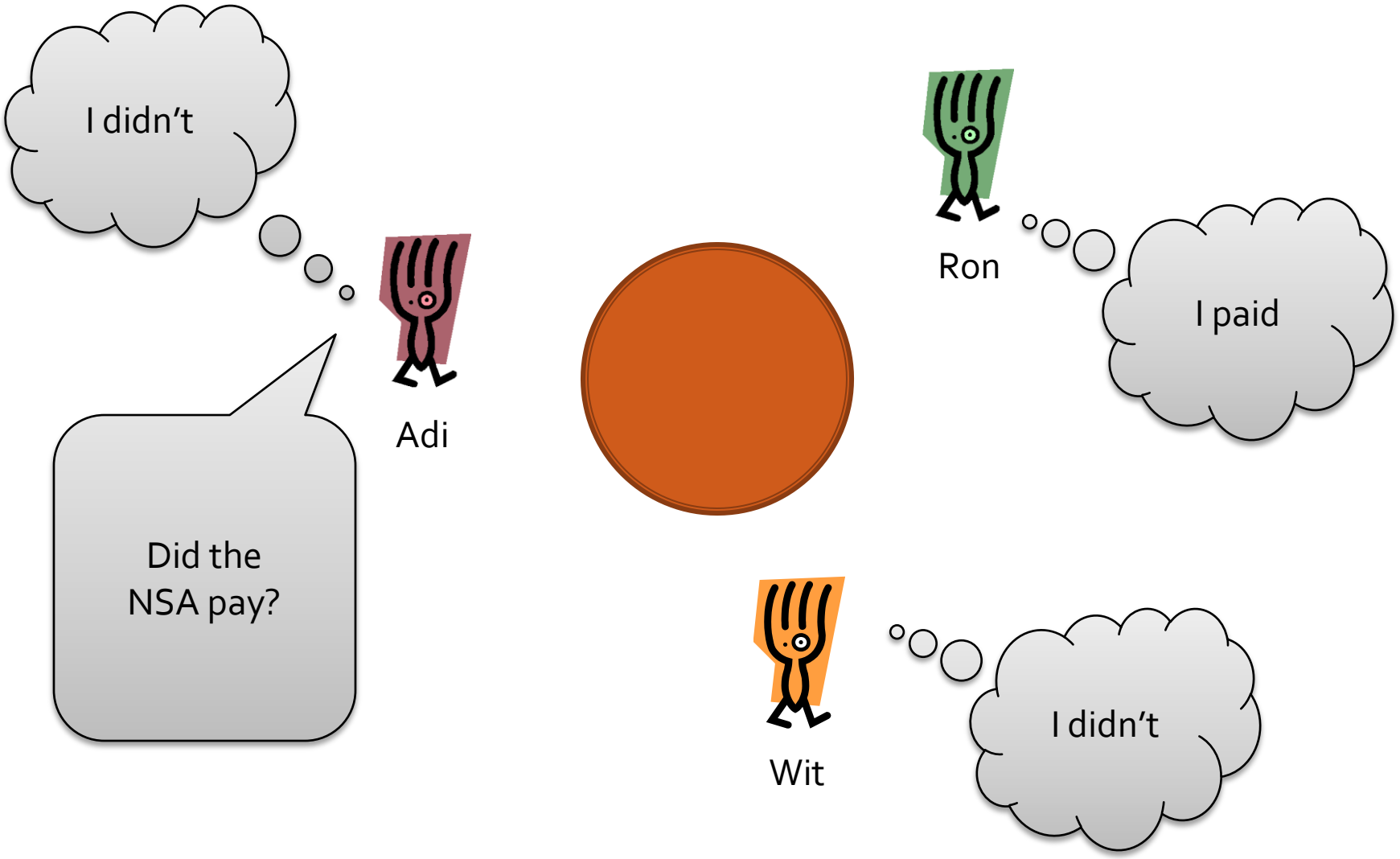
Unconditional anonymity

- DC-nets
  - Dining Cryptographers (David Chaum 1985)
- Multi-party computation resulting in a message being broadcast anonymously
  - No one knows from which party
  - How to avoid collisions
- Communication cost…

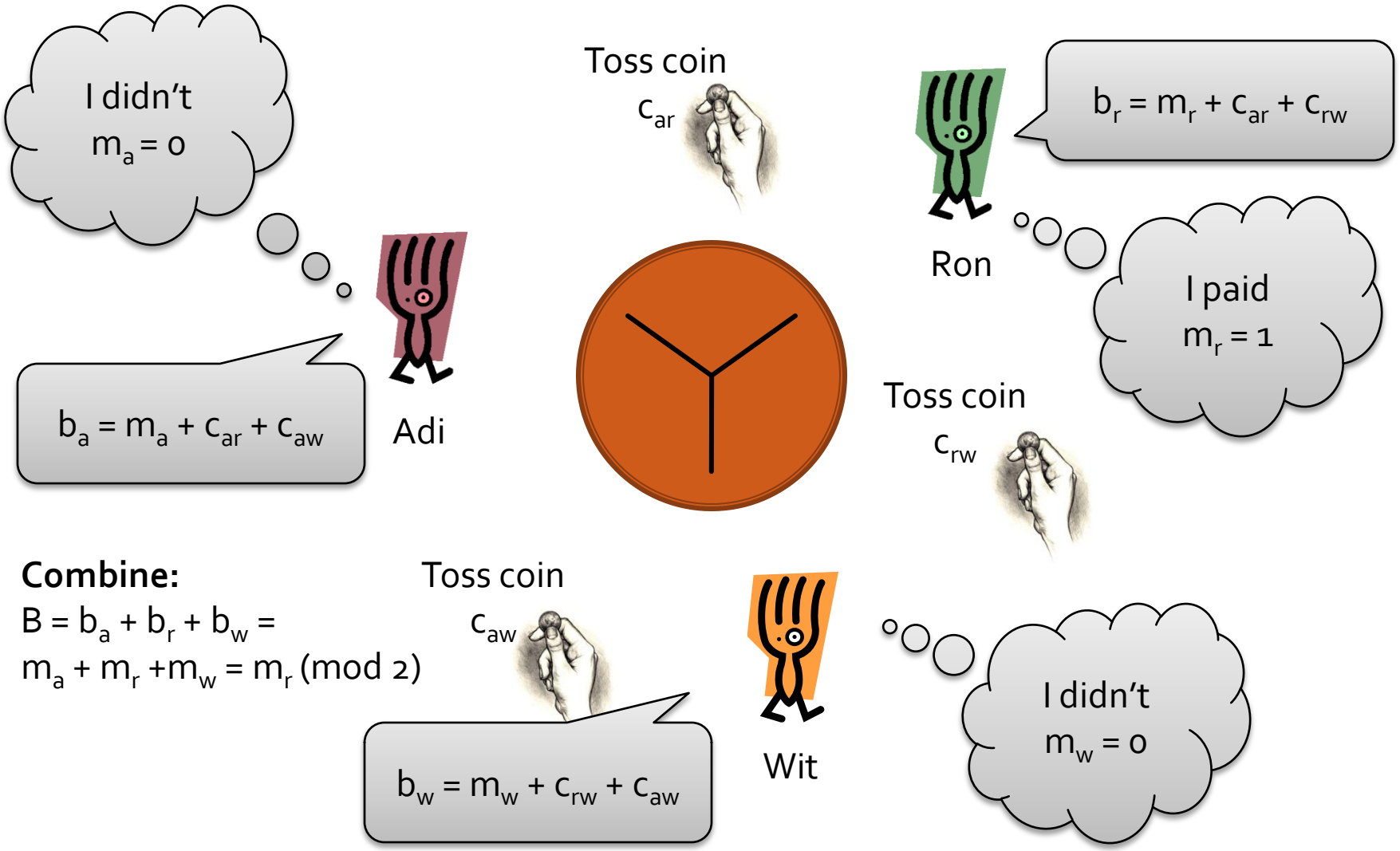
The Dining Cryptographers (1)

- "Three cryptographers are sitting down to dinner at their favourite three-star restaurant.
- Their waiter informs them that arrangements have been made with the maitre d'hotel for the bill to be paid anonymously.
- One of the cryptographers might be paying for the dinner, or it might have been NSA (U.S. National Security Agency).
- The three cryptographers respect each other's right to make an anonymous payment, but they wonder if NSA is paying."

The Dining Cryptographers (2)



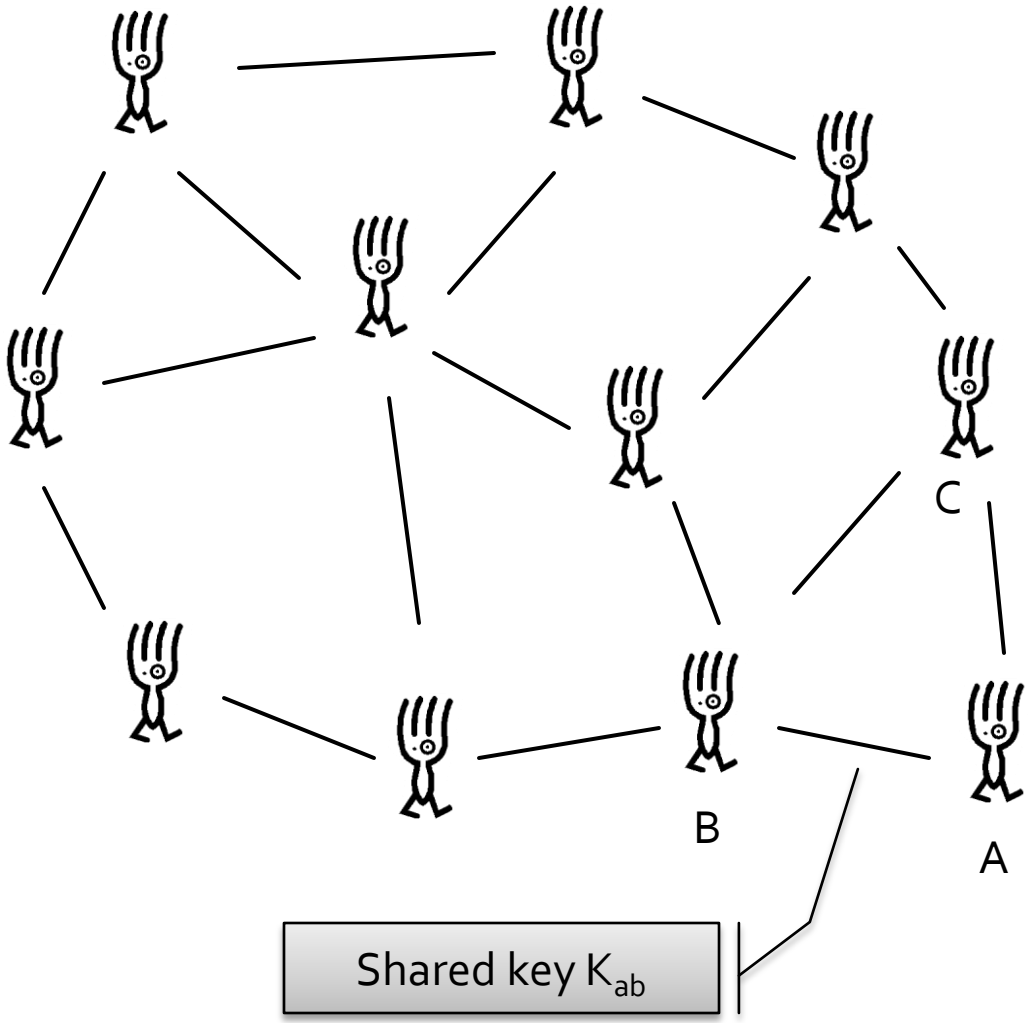
The Dining Cryptographers (2)



DC-nets

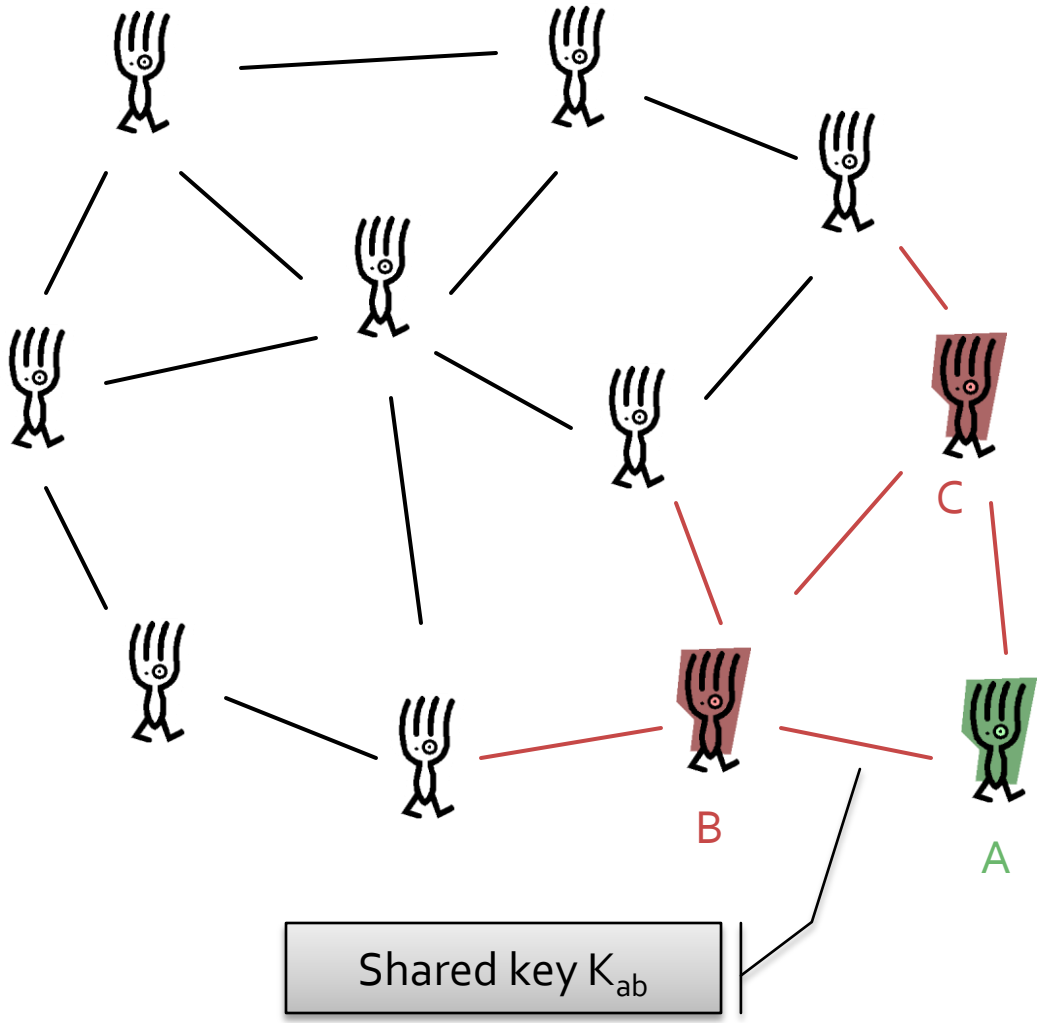
- Generalise
  - Many participants
  - Larger message size
    - Conceptually many coins in parallel (xor)
    - Or: use +/- (mod $2^{|m|}$)
  - Arbitrary key (coin) sharing
    - Graph G:
      - nodes - participants,
      - edges - keys shared
- What security?

Key sharing graph



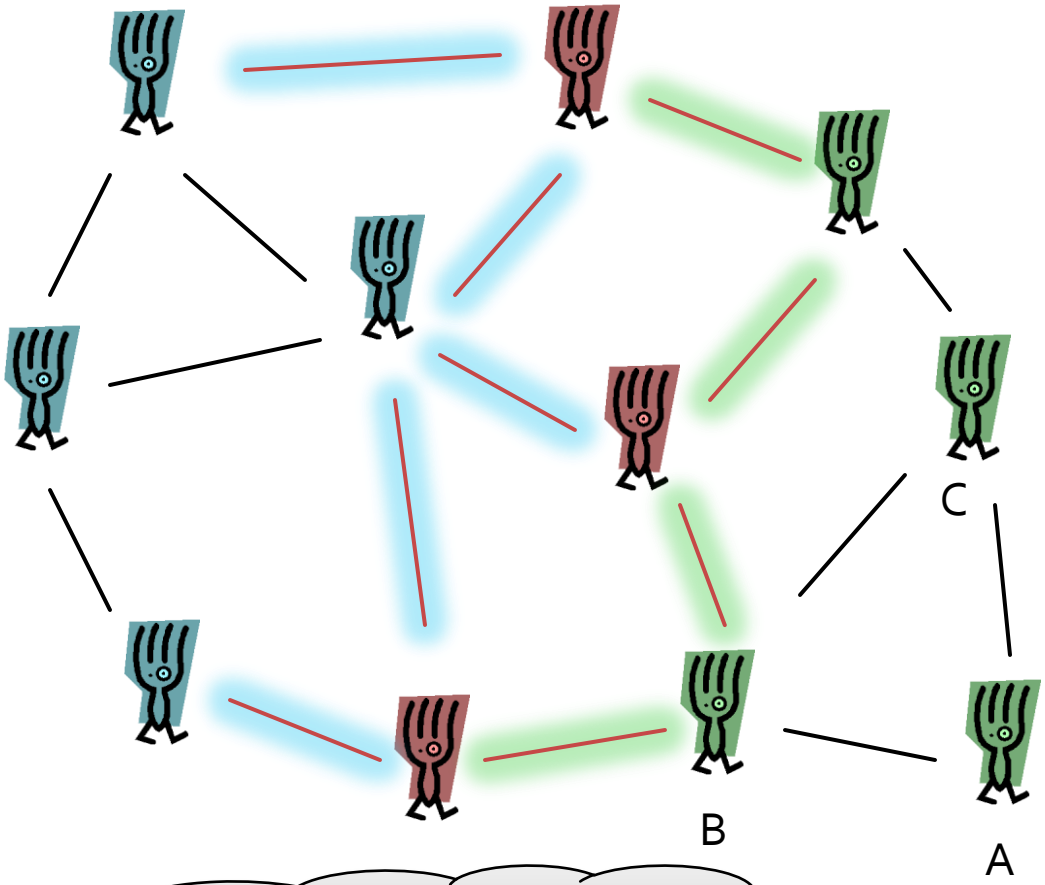
- Derive coins
  - $c_{abi} = H[K_{ab}, i]$ for round i
  - Stream cipher (K_{ab})
- Alice broadcasts
  - $b_a = c_{ab} + c_{ac} + m_a$

Key sharing graph – security (1)



- If B and C corrupt
- Alice broadcasts
  - $b_a = c_{ab} + c_{ac} + m_a$
- Adversary's view
  - $b_a = c_{ab} + c_{ac} + m_a$
- No Anonymity

Key sharing graph – security (2)



- Adversary nodes partition the graph into a blue and green sub-graph
- Calculate:
  - $B_{blue} = \sum b_j$, j is blue
  - $B_{green} = \sum b_i$, i is green
- Substract known keys
  - $B_{blue} + K_{red\text{-}blue} = \sum m_j$
  - $B_{green} + K'_{red\text{-}green} = \sum m_i$
- Discover the originating subgraph.
  - Reduction in anonymity

DC-net twists

- b_i broadcast graph
  - Tree – independent of key sharing graph
  - = Key sharing graph – No DoS unless split in graph
- Collisions
  - Alice says $m_A \neq 0$ and Bob says $m_B \neq 0$
  - N collisions only require N rounds to be resolved!
  - Intuition: collisions do destroy all information
    - Round 1: $B_1 = m_A + m_B$ Round 2: $B_2 = m_B$ $m_A = ?$
- Disruption?
  - Dining Cryptographers in a Disco

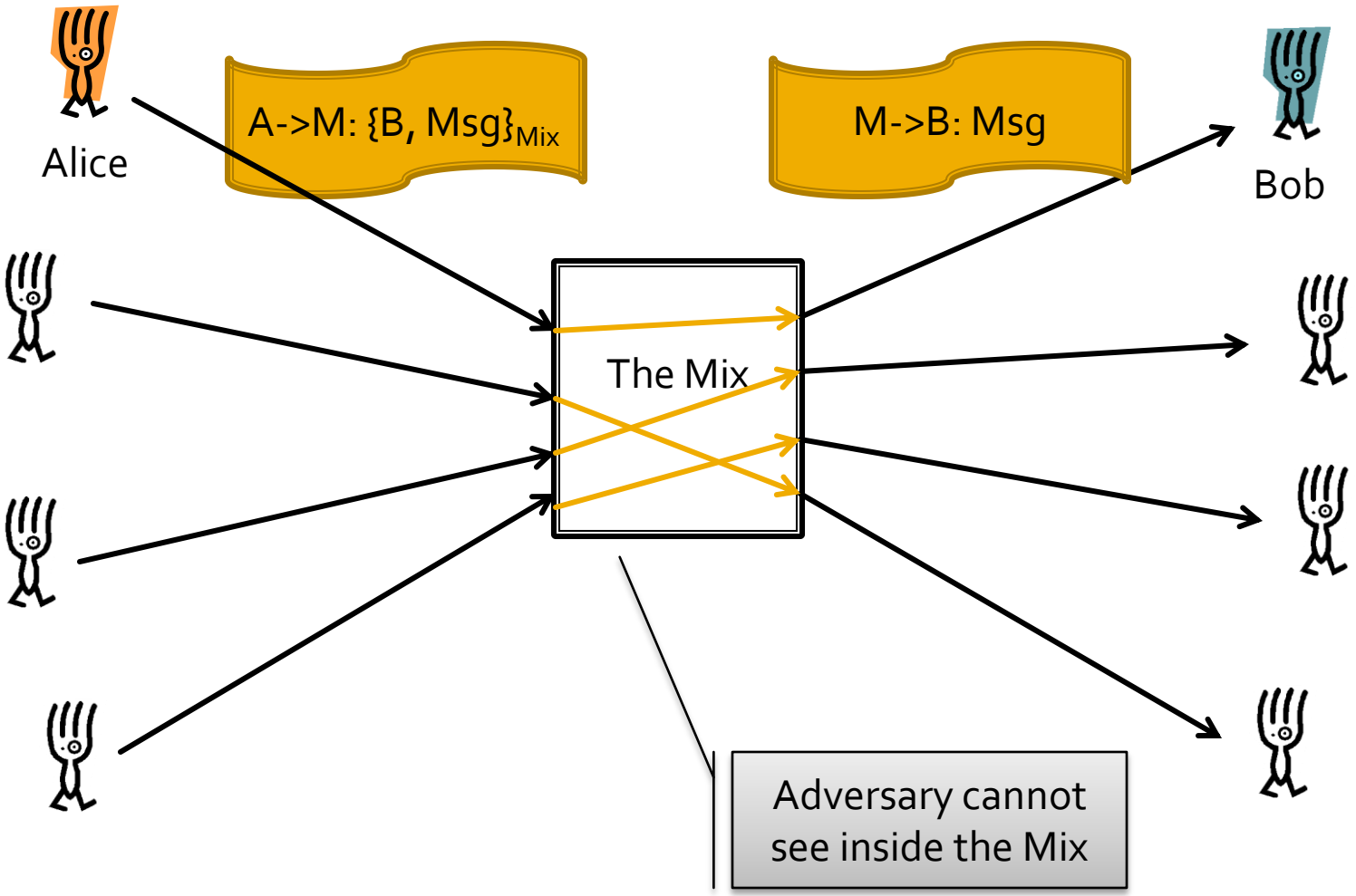
DC-net shortcommings

- Security is great!
  - Full key sharing graph ⇔ perfect anonymity
- Communication cost – BAD
  - (N broadcasts for each message!)
  - Naive: $O(N^2)$ cost, O(1) Latency
  - Not so naive: O(N) messages, O(N) latency
    - Ring structure for broadcast
  - Expander graph: O(N) messages, O(logN) latency?
  - Centralized: O(N) messages, O(1) latency
- Not practical for large(r) N! ☹
  - Local wireless communications?

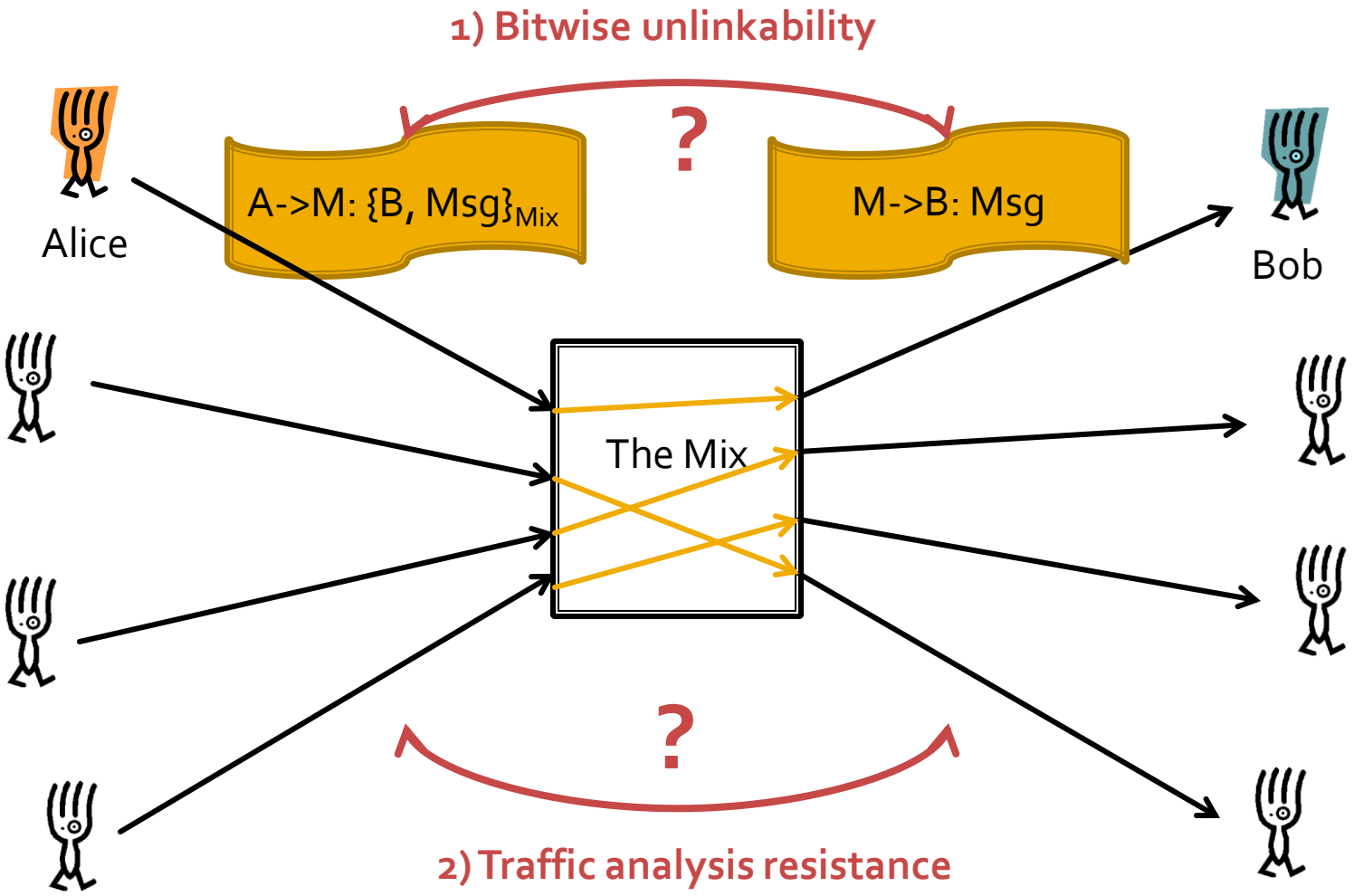
Mix – practical anonymity

- David Chaum (concept 1979 – publish 1981)
  - Ref 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

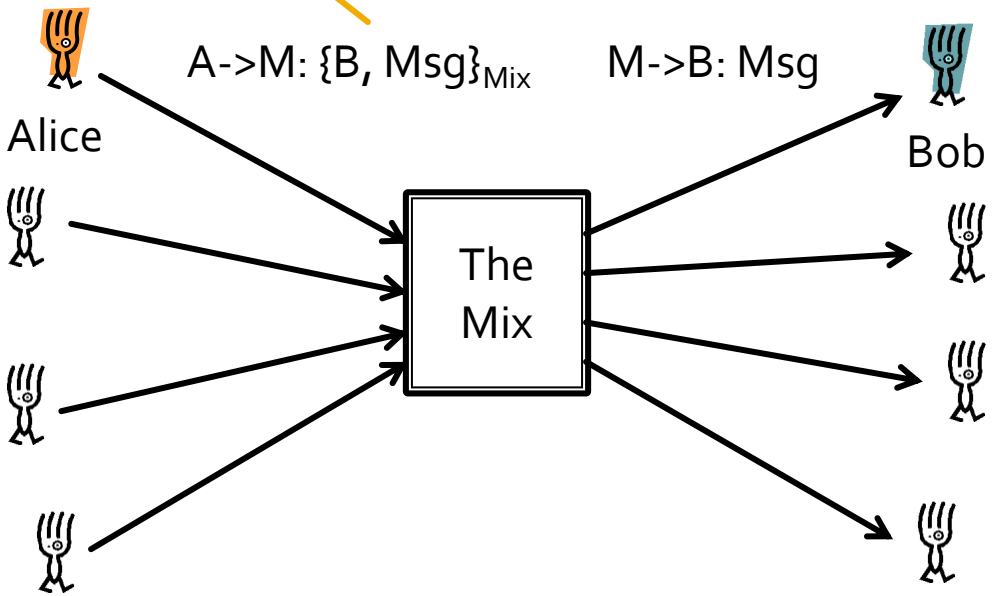


Mix security (contd.)

- 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 or inject traffic – both add cost!

Two broken mix designs (1)

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



- Active attack?

Tagging Attack

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

The mix outputs message:

M->B: Msg xor Y

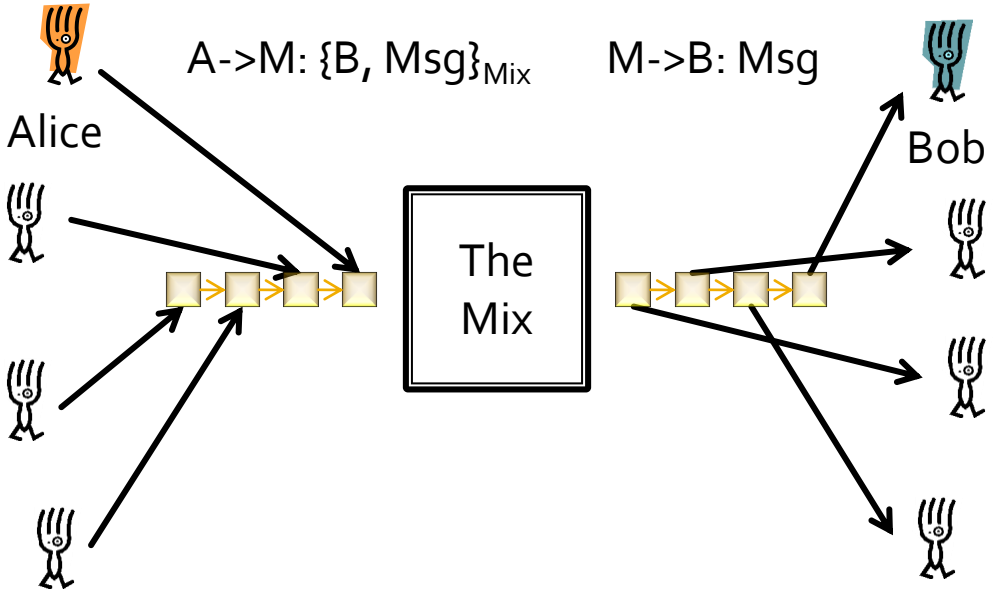
And the attacker can link them.

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: lose all information (Mixminion, Minx)

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.

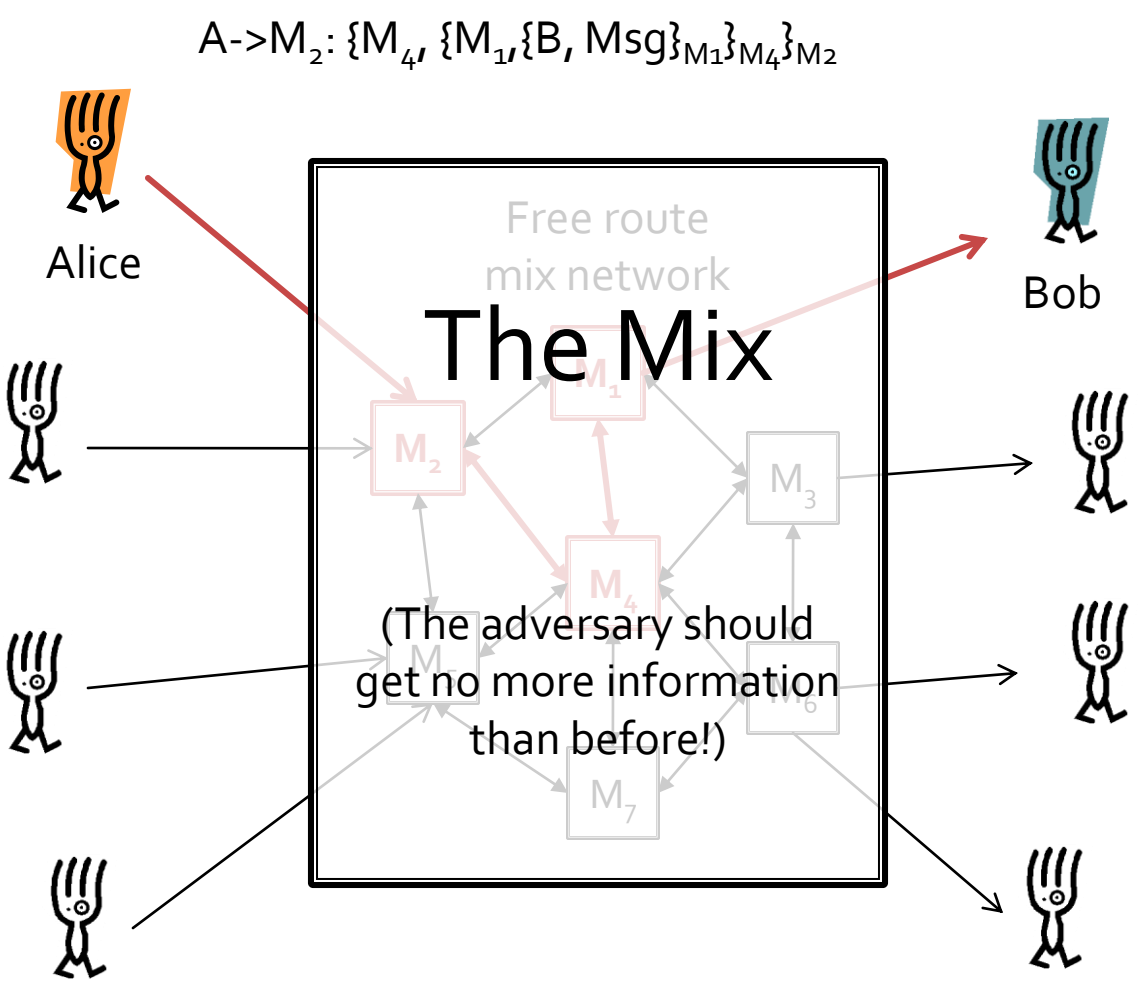
Lessons from broken design 2

- 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, …
- Anonymity security relies on *others*
  - Mix honest – Problem 1
  - Other sender-receiver pairs to hide amongst – Problem 2

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

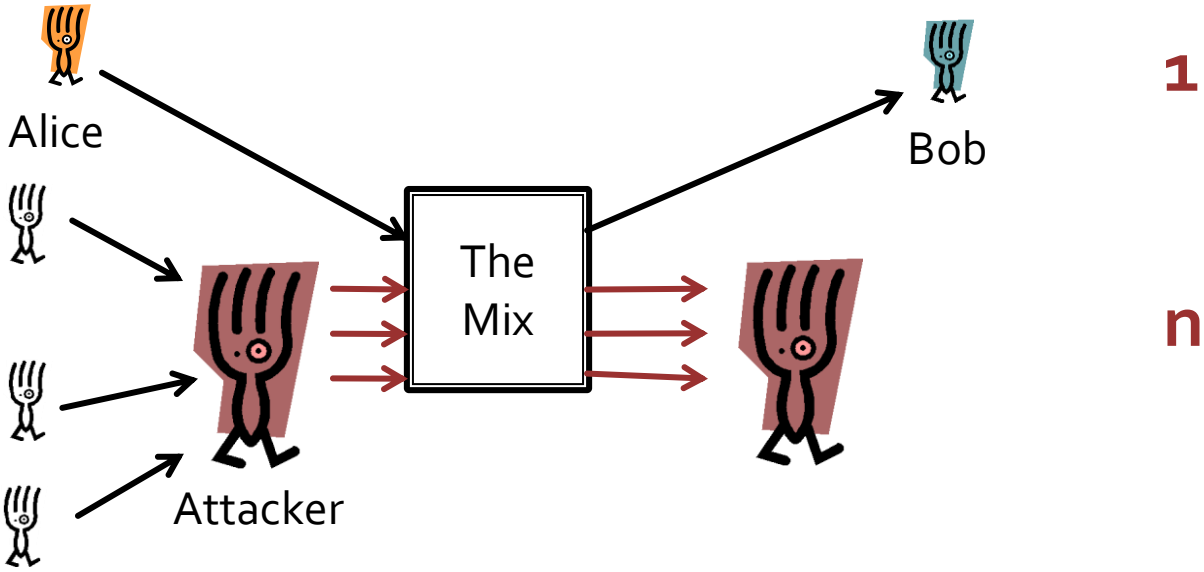


Free route mix networks

- Bitwise unlinkability
  - Length invariance
  - Replay prevention
- 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"

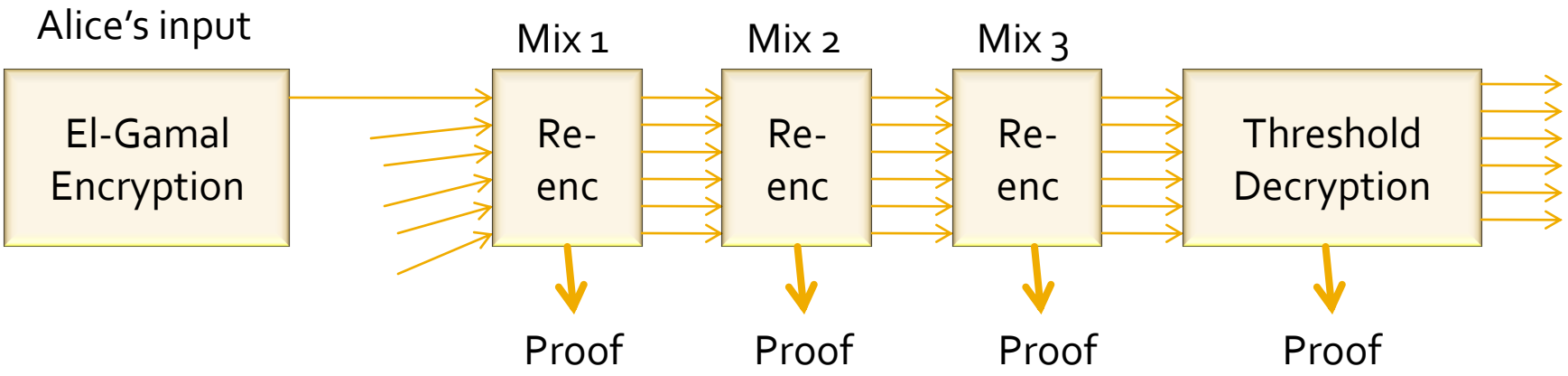
Robustness to DoS

- Malicious mixes may be dropping messages
  - Special problem in elections
- Original idea: receipts (unworkable)
- Two key strategies to prevent DoS
  - Provable shuffles
  - Randomized partial checking

Provable shuffles – overview

- Bitwise unlinkability: El-Gamal re-encryption
  - El-Gamal public key (g, g^x) for private x
  - El-Gamal encryption $(g^k, g^{kx} \cdot M)$
  - El-Gamal re-encryption $(g^{k'} \cdot g^k , g^{k'x} g^{kx} \cdot M)$
    - No need to know x to re-encrypt
    - Encryption and re-encryption unlinkable
- Architecture – re-encryption cascade
  - Output proof of correct shuffle at each step

Provable shuffles – illustrated

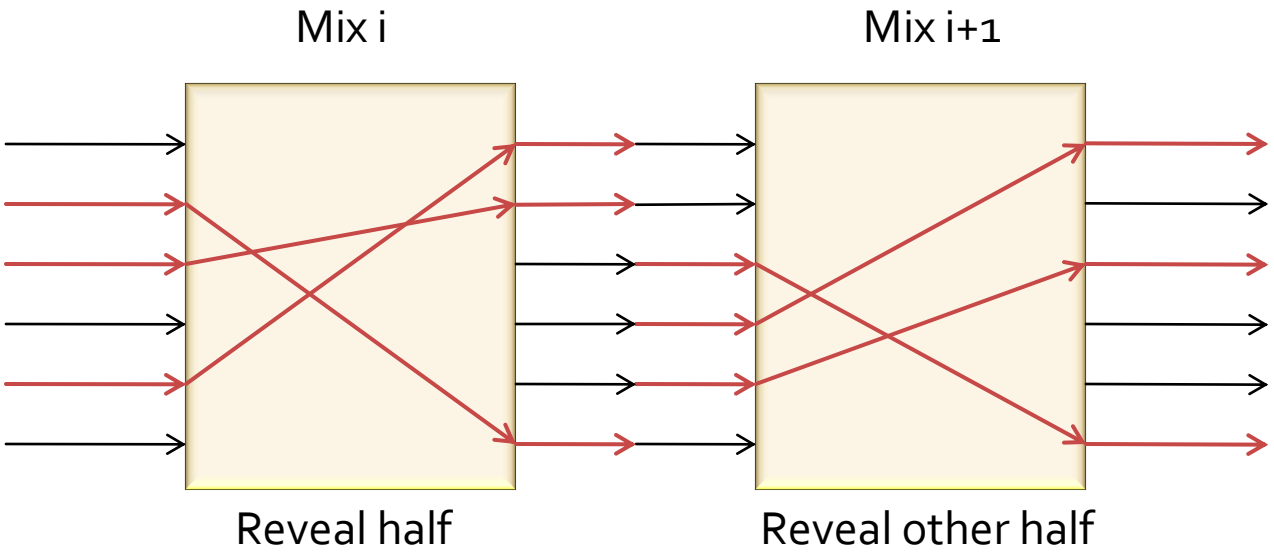


- Proof of correct shuffle
  - Outputs are a permutation of the decrypted inputs
  - (Nothing was inserted, dropped, otherwise modified!)
  - Upside: Publicly verifiable – Downside: expensive

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
- 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

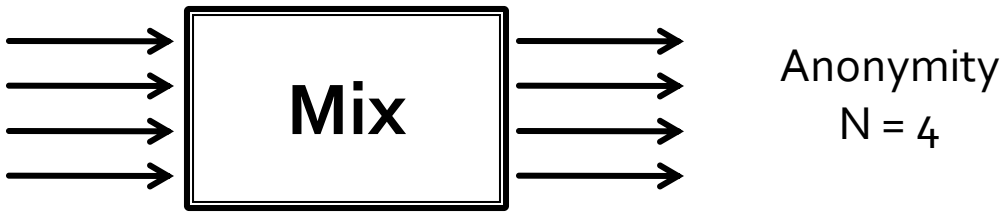
Summary of key concepts

- Anonymity requires a crowd
  - Difficult to ensure it is not simulated – (n-1) attack
- DC-nets – Unconditional anonymity at high communication cost
  - Collision resolution possible
- 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

Anonymity measures – old

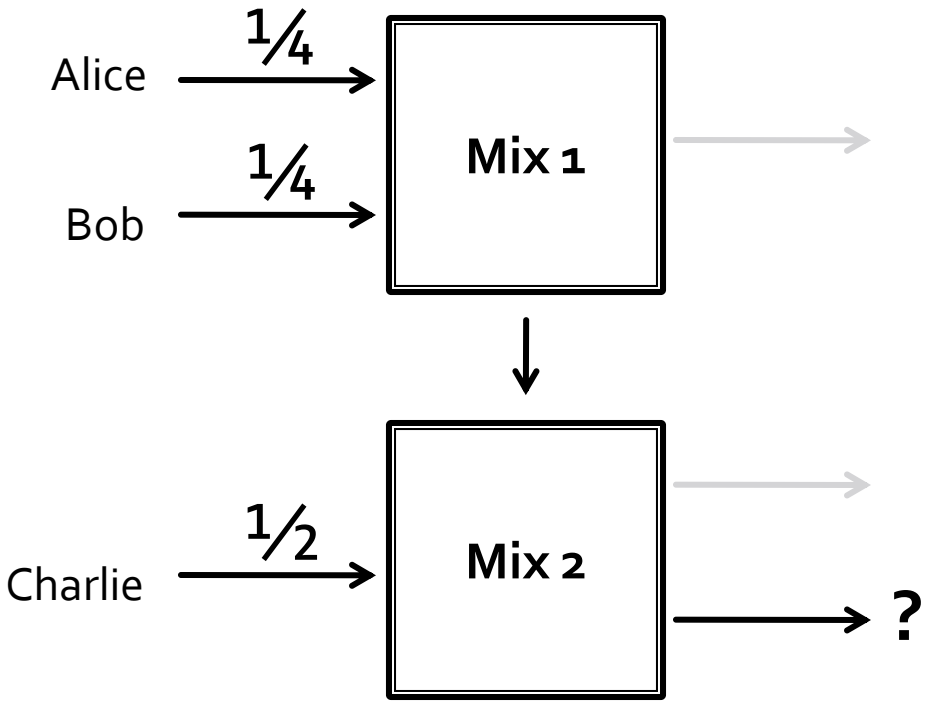
- The anonymity set (size)
  - Dining cryptographers
    - Full key sharing graph = (N - |Adversary|)
    - Non-full graph – size of graph partition
  - Assumption: all equally likely

- Mix network context
  - Threshold mix with N inputs: Anonymity = N



Anonymity set limitations

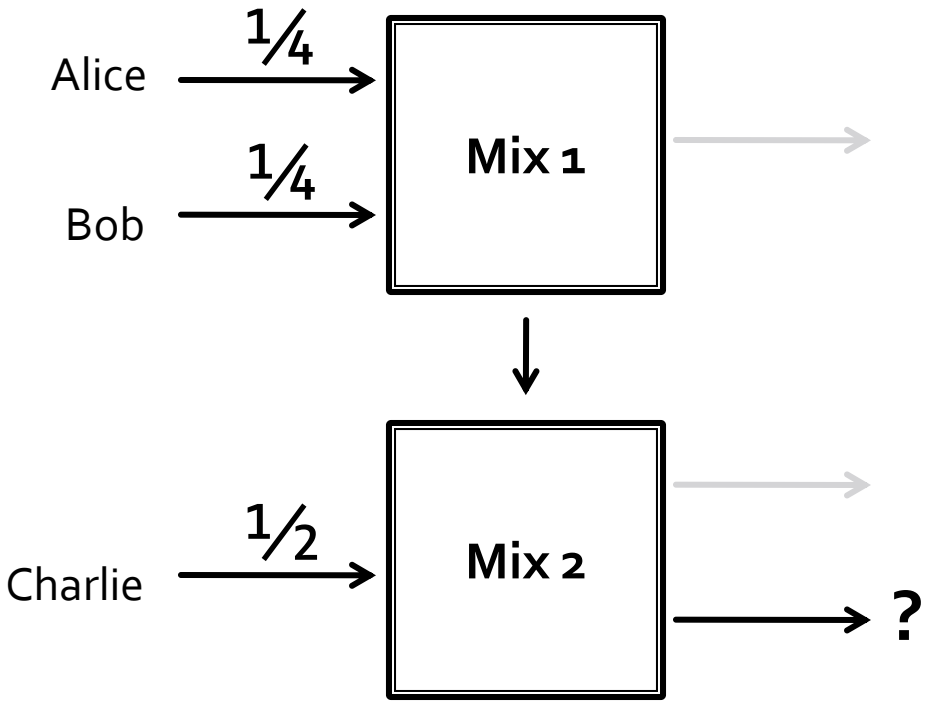
- Example: 2-stage mix



- Option 1:
  - 3 possible participants
  - => N = 3
  - Note probabilities!
- Option 2:
  - Arbitrary min probability
  - Problem: ad-hoc

Entropy as anonymity

- Example: 2-stage mix



- Define distribution of senders (as shown)
- Entropy of the distribution is anonymity
  - $E = -\sum p_i \log_2 p_i$
- Example:

  $E = -2\,\frac{1}{4}(-2) - (\frac{1}{2})(-1)$

  $= +1 + \frac{1}{2} = 1.5$ bits
- (NOT N=3 => E = -log3 = 1.58 bits)
- Intuition: missing information for full identification!

Anonymity measure pitfalls

- Only the attacker can measure the anonymity of a system.
  - Need to know which inputs, output, mixes are controlled
- Anonymity of single messages
  - How to combine to define the anonymity of a systems?
  - Min-anonymity of messages
- How do you derive the probabilities? (Hard!)
  - Complex systems – not just examples

What next? Patterns!

- Statistical Disclosure
  - Tracing persistent communications
- Low-latency anonymity
  - Onion-routing & Tor
    - Tracing streams
    - Restricted directories
    - (Going fully peer-to-peer…)
  - Crowds
    - Predecessor attack

References

- Core:
  - **The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability** by David Chaum. In Journal of Cryptology 1, 1988, pages 65-75.
  - **Mixminion: Design of a Type III Anonymous Remailer Protocol** by George Danezis, Roger Dingledine, and Nick Mathewson. In the Proceedings of the 2003 IEEE Symposium on Security and Privacy, May 2003, pages 2-15.
- More
  - **A survey of anonymous communication channels** by George Danezis and Claudia Diaz http://homes.esat.kuleuven.be/~gdanezis/anonSurvey.pdf
  - **The anonymity bibliography** http://www.freehaven.net/anonbib/

Anonymous communications: Low latency systems

Anonymous web browsing and peer-to-peer

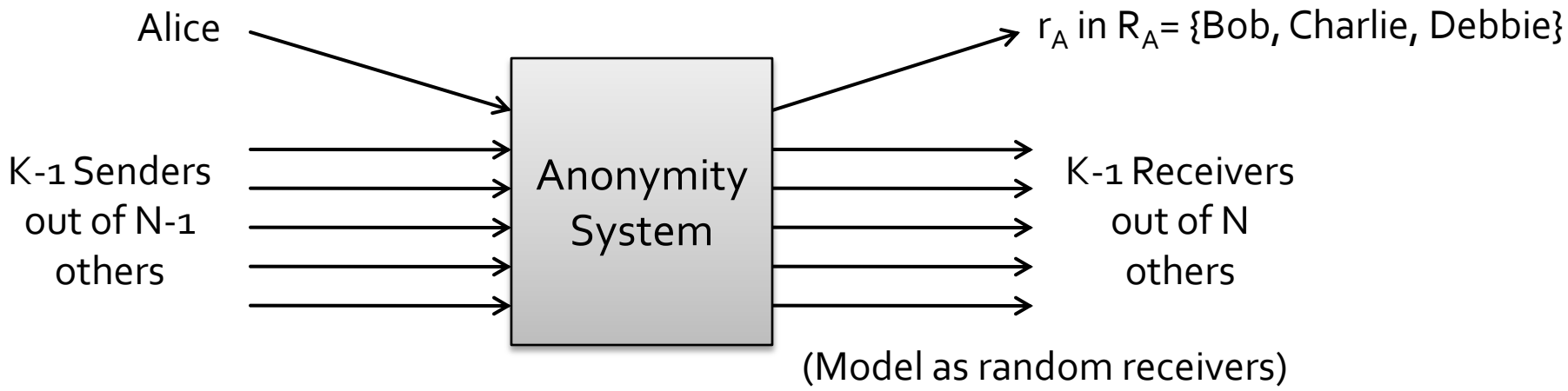
Anonymity so far…

- Mixes or DC-nets – setting
  - Single message from Alice to Bob
  - Replies
- Real communications
  - Alice has a few friends that she messages often
  - Interactive stream between Alice and Bob (TCP)
- Repetition – patterns -> Attacks

Fundamental limits

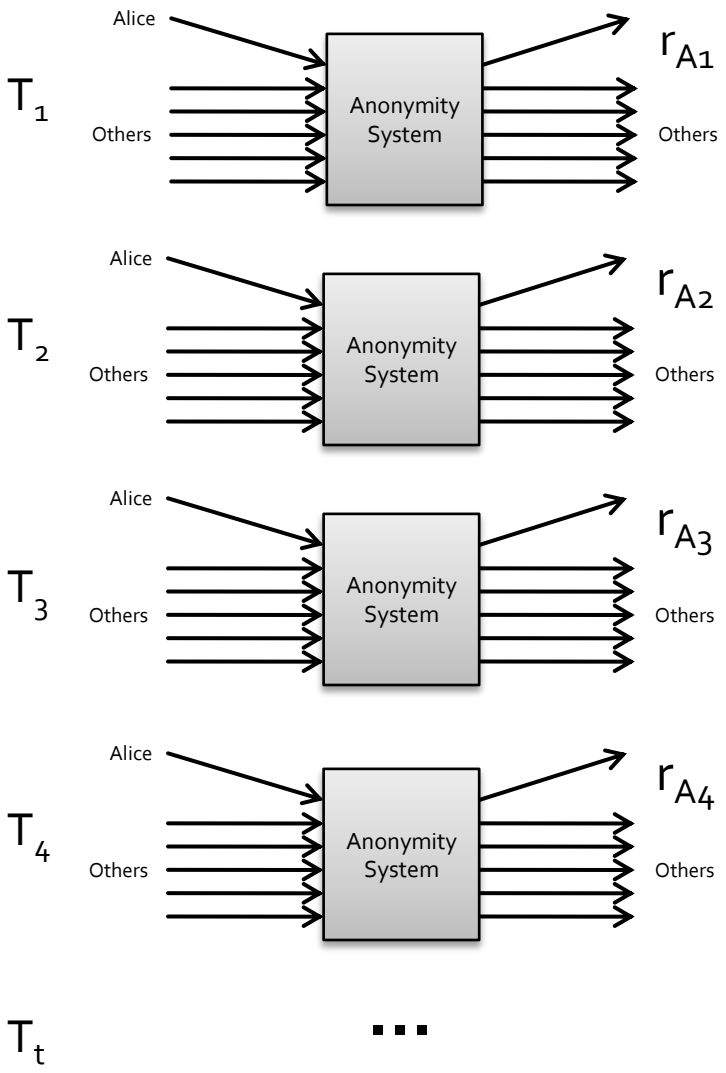
- Even perfect anonymity systems leak information when participants change
- Setting:
  - N senders / receivers – Alice is one of them
  - Alice messages a small number of friends:
    - R_A in {Bob, Charlie, Debbie}
    - Through a MIX / DC-net
    - Perfect anonymity of size K
  - Can we infer Alice's 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**	**1**
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		
23	[19, 13, 3, 17, 12]	[0, 13, 19]	0			
24	[8, 18, 0, 10, 18]	[0, 13, 18]	1	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

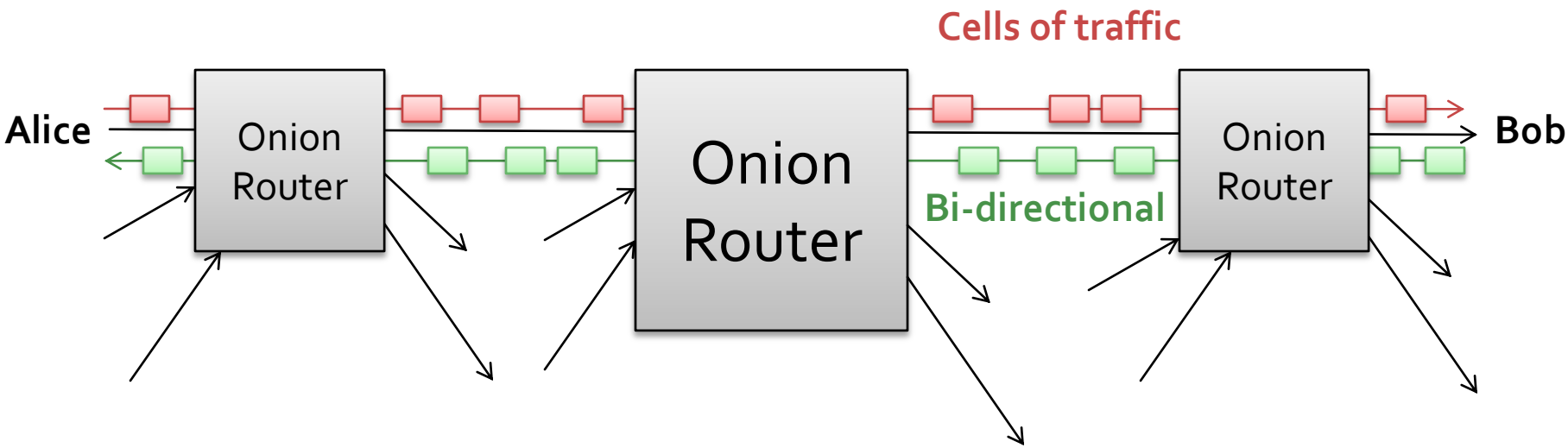
- Counter-intuitive
  - The larger N the easiest the attack
- Hitting-set attacks
  - More accurate, need less information
  - Slower to implement
  - Sensitive to Model
    - E.g. Alice sends dummy messages with probability p.
- Statistical disclosure attacks
  - Need more data
  - Very efficient to 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!

Onion Routing

- Anonymising streams of messages
  - Example: Tor
- 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 PK 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!

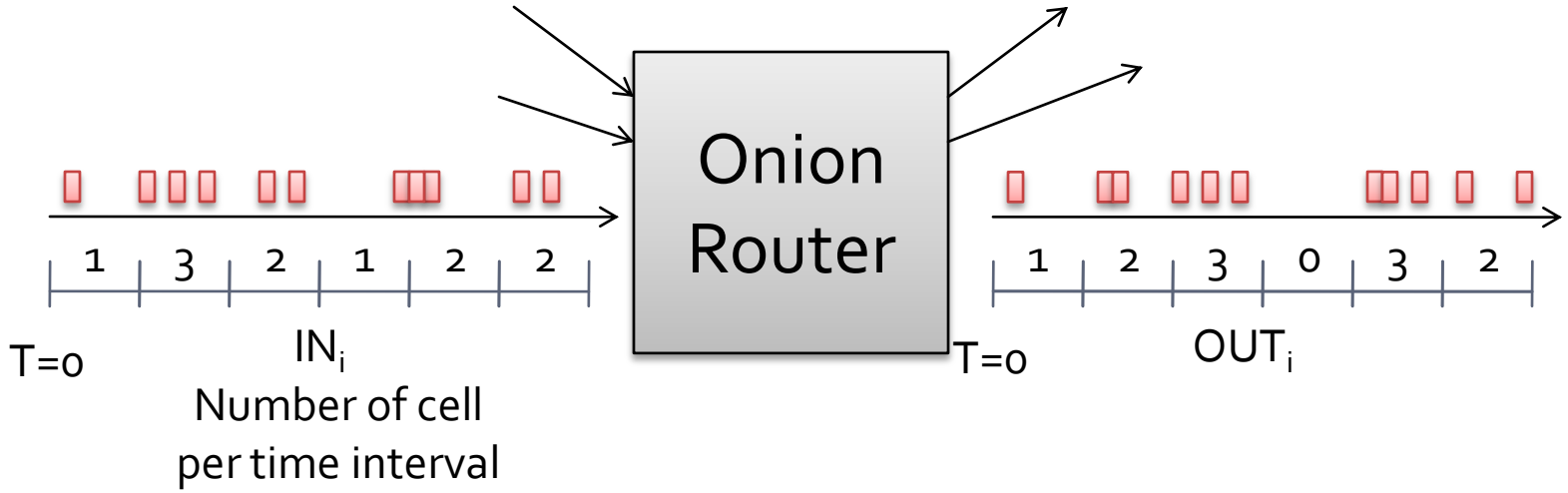
Stream Tracing

- Adversary observes all inputs and outputs of an onion router
- Objective link the ingoing and outgoing connections (to trace from Alice to Bob)

- Key: timing of packets are correlated

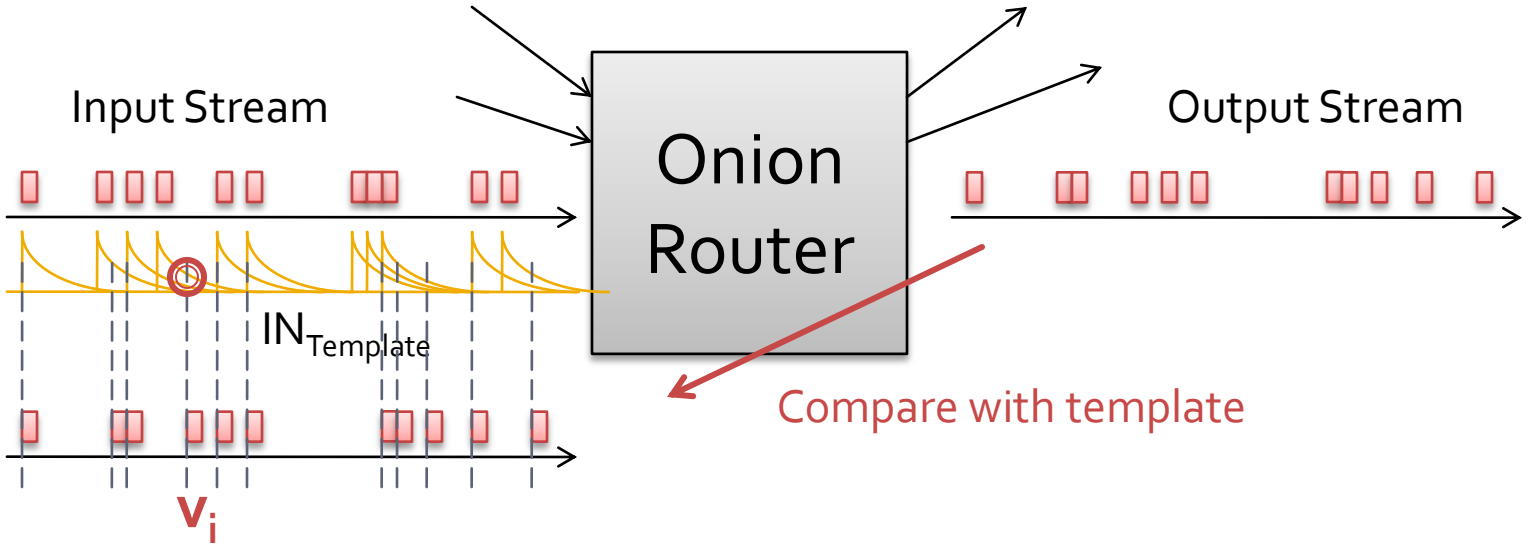
- Two techniques:
  - Correlation
  - Template matching

Tracing (1) – Correlation



- Quantise input and output load in time
- Compute:
  - Corr = $\sum_i IN_i \cdot OUT_i$
- Downside: lose precision by quantising

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 ($\prod_i v_i$)

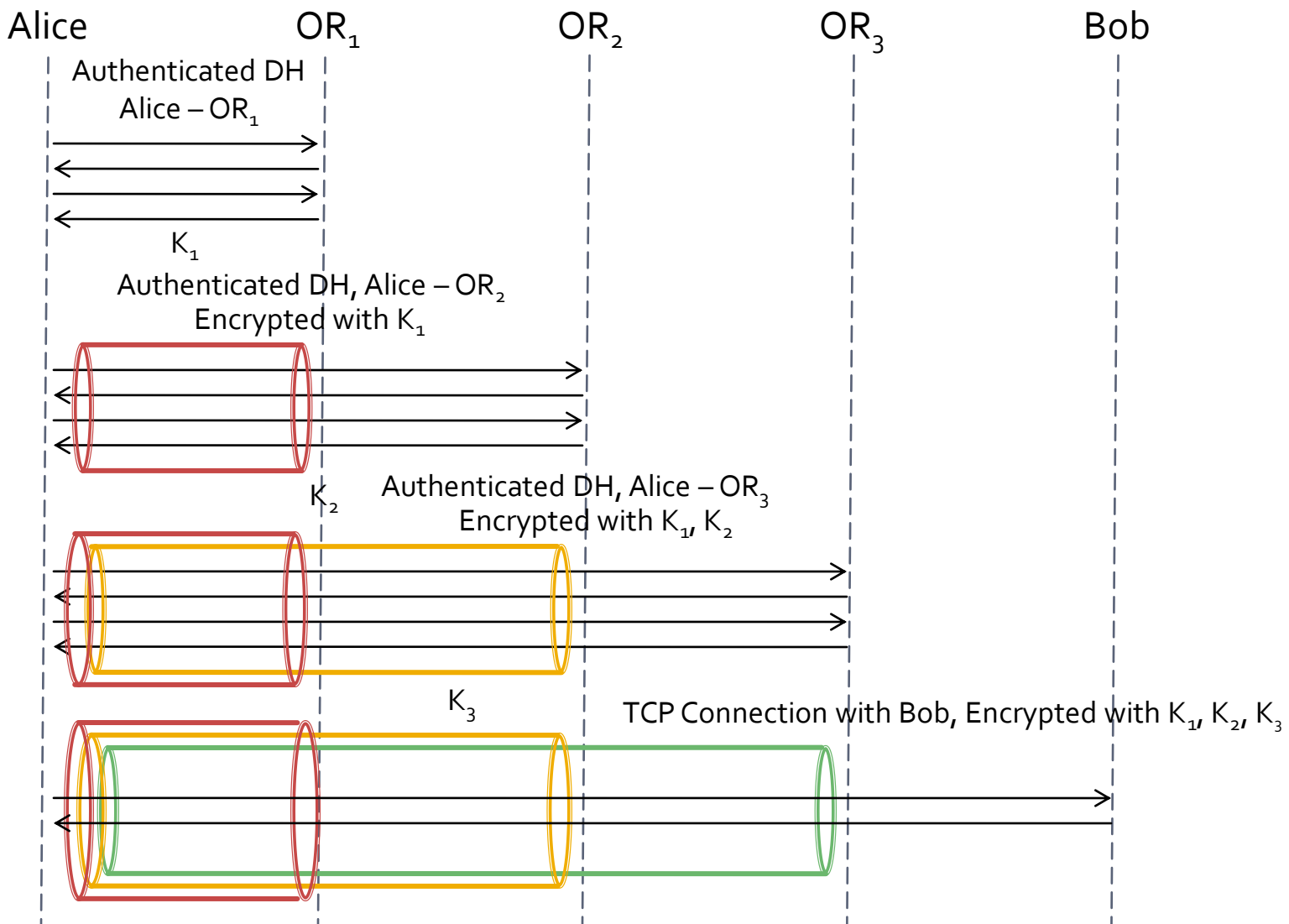
The security of Onion Routing

- Cannot withstand a global passive adversary
  - (Tracing attacks to 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
  - Compromize probability = c^2
- No point making routes too long

More Onion Routing security

- Forward secrecy
  - In mix networks Alice uses long term keys
    $A \rightarrow M_2: \{M_4, \{M_1, \{B, Msg\}_{M1}\}_{M4}\}_{M2}$
  - In Onion Routing a bi-directional channel is available
  - Can perform authenticated Diffie-Hellman to extend the anonymous channel
- OR provides better security against compulsion

Extending the route in OR



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

Exercise

- Show that:
  - If Alice knows only a small subset of all Onion Routers, the paths she creates using them are not anonymous.
  - Assume adversary knows Alice's subset of nodes.
  - Hint: Consider collusion between a corrupt middle and last node – then corrupt last node only.
- Real problem: need to ensure all clients know the full, most up-to-date list of routers.

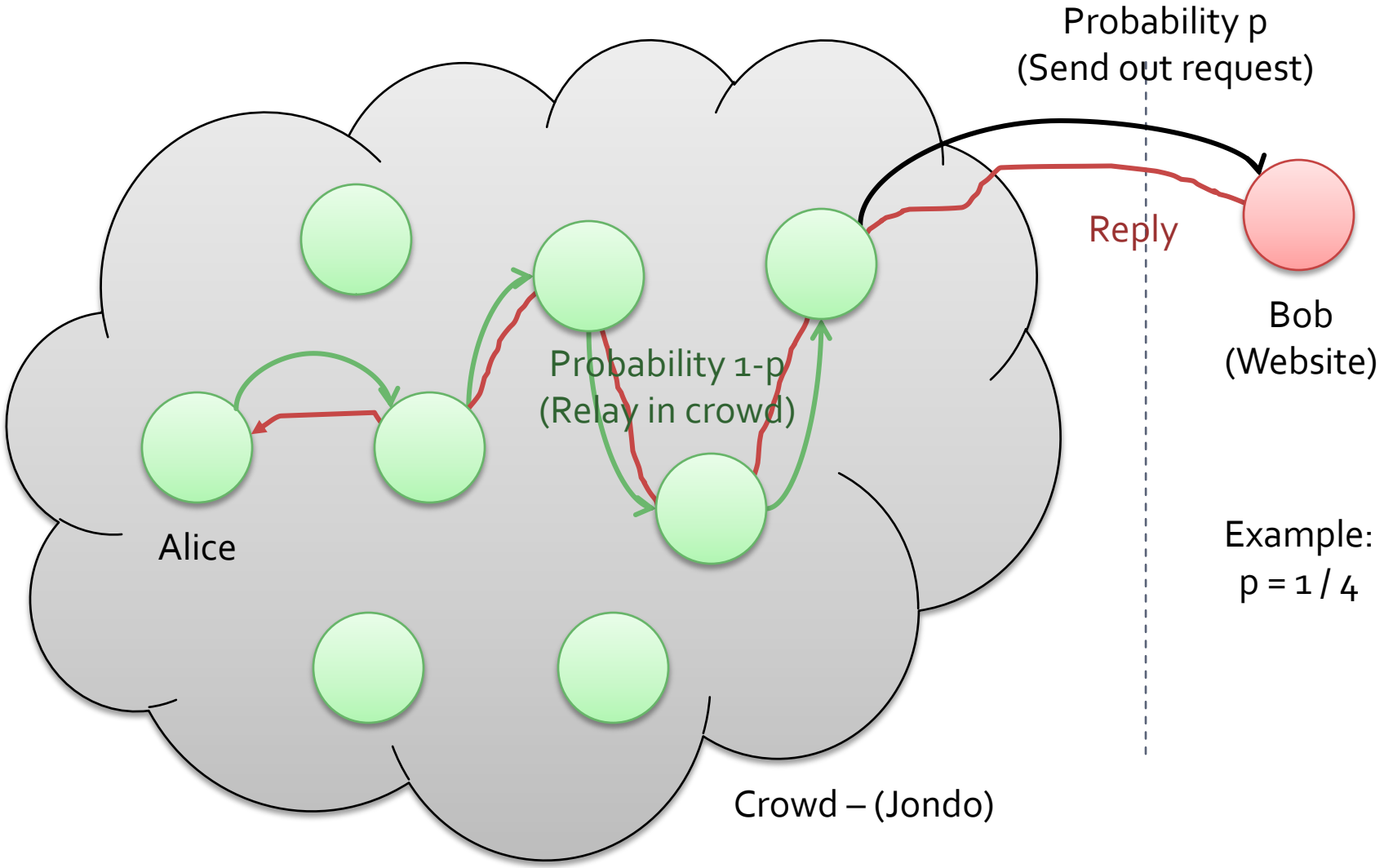
Future directions in OR

- Anonymous routing immune to tracing
  - Reasonable latency?
- Yes, we can!
  - Tracing possible because of input-output correlations
  - Strategy 1: fixed sending of cells (eg. 1 every 20-30ms)
  - Strategy 2: fix any sending schedule independently of the input streams

Crowds – lightweight anonymity

- Mixes and OR – heavy on cryptography
- Lighter threat model
  - No network adversary
  - Small fraction of corrupt nodes
  - Anonymity of web access
- Crowds: a groups of nodes cooperate to provide anonymous web-browsing

Crowds – illustrated



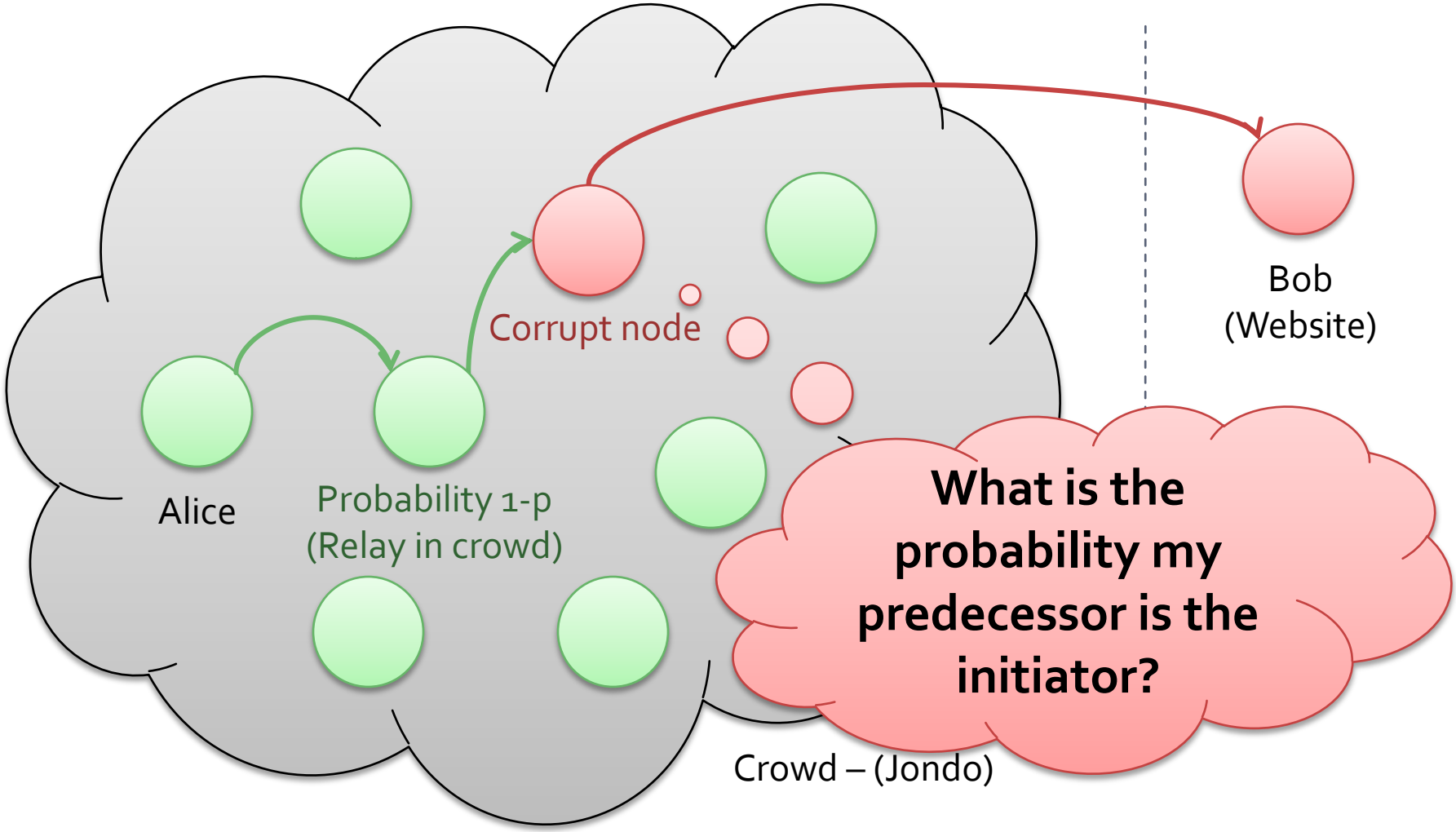
Crowds security

- Final website (Bob) or corrupt node does not know who the initiator is
  - Could be the node that passed on the request
  - Or one before
- How long do we expect paths to be?
  - Mean of geometric distribution
  - $L = 1 / p$ – (example: $L = 4$)
  - Latency of request / reply

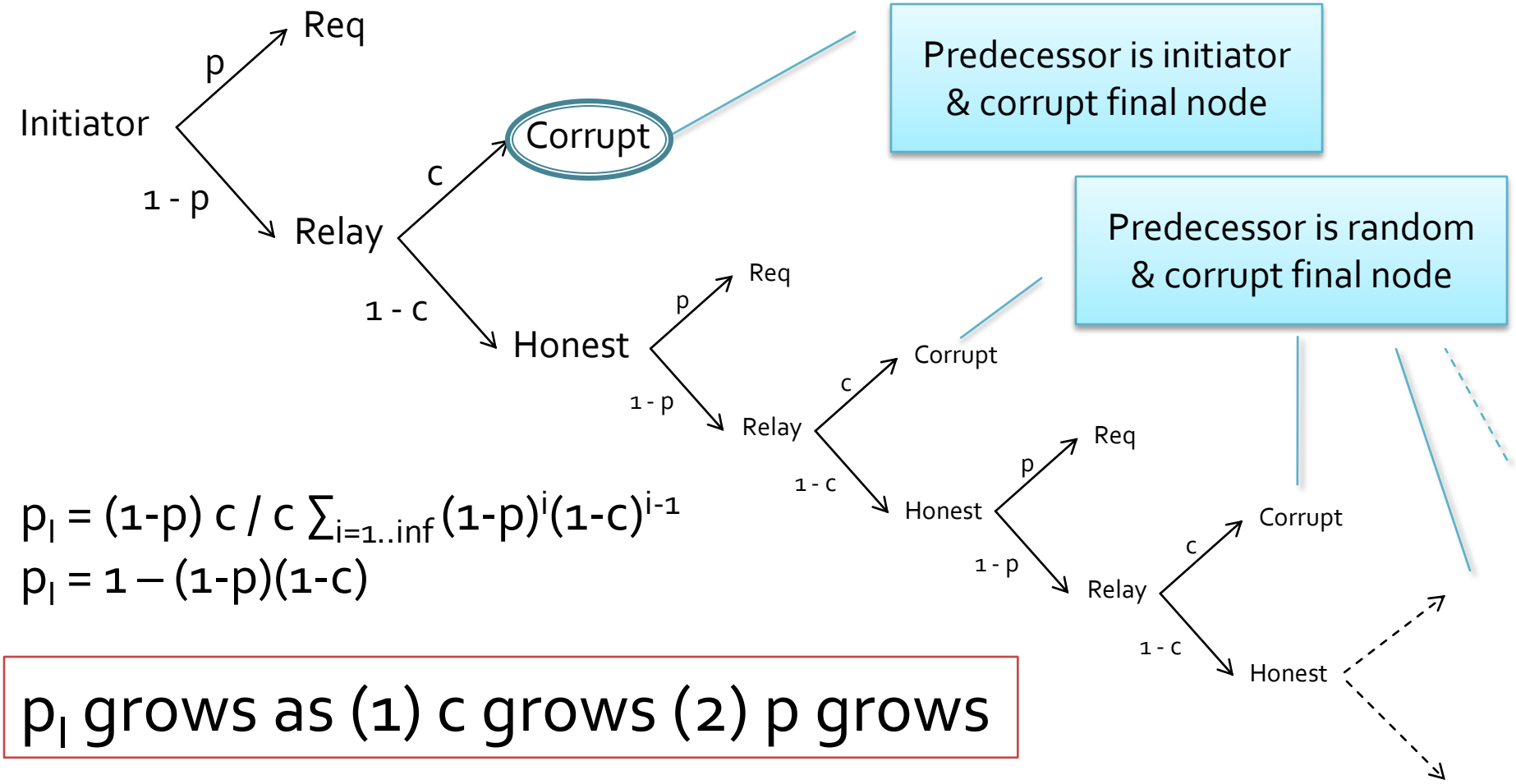
Crowds security (2)

- Consider the case of a corrupt insider
  - A fraction c of nodes are in fact corrupt
- When they see a request they have to decide whether
  - the predecessor is the initiator
  - or merely a relay
- Note: corrupt insiders will never pass the request to an honest node again!

Crowds – Corrupt insider



Calculate: initiator probability



$p_I = (1\text{-}p)\, c \,/\, c \sum_{i=1..inf} (1\text{-}p)^i (1\text{-}c)^{i\text{-}1}$

$p_I = 1 - (1\text{-}p)(1\text{-}c)$

p_I grows as (1) c grows (2) p grows

Exercise: What is the information theoretic amount of anonymity of crowds in this context

The predecessor attack

- What about repeated requests?
  - Alice always visits Bob
  - E.g. Repeated SMTP connection to microsoft.com
- Adversary can observe n times the tuple
  - 2 x (Alice, Bob)
  - Probability Alice is initiator (at least once)
    - $P = 1 - [(1-p)(1-c)]^n$
  - Probability of compromize reaches 1 very fast!

Summary of key points

- Fast routing = no mixing = traffic analysis attacks
- Weaker threat models
  - Onion routing: partial observer
  - Crowds: insiders and remote sites
- Repeated patterns
  - Onion routing: Streams vs. Time
  - Crowds: initiators-request tuples
- PKI overheads a barrier to p2p anonymity

References

- Core:
  - **Tor: The Second-Generation Onion Router** by Roger Dingledine, Nick Mathewson, and Paul Syverson. In the Proceedings of the 13th USENIX Security Symposium, August 2004.
  - **Crowds: Anonymity for Web Transactions** by Michael Reiter and Aviel Rubin. In ACM Transactions on Information and System Security 1(1), June 1998.
- More:
  - **An Introduction to Traffic Analysis** by George Danezis and Richard Clayton. http://homes.esat.kuleuven.be/~gdanezis/TAIntro-book.pdf
  - **The anonymity bibliography** http://www.freehaven.net/anonbib/