Lecture 10:
Strong Anonymity

CPSC 4440/5440, 2026 Spring

Instructor: Fan Zhang

Yale

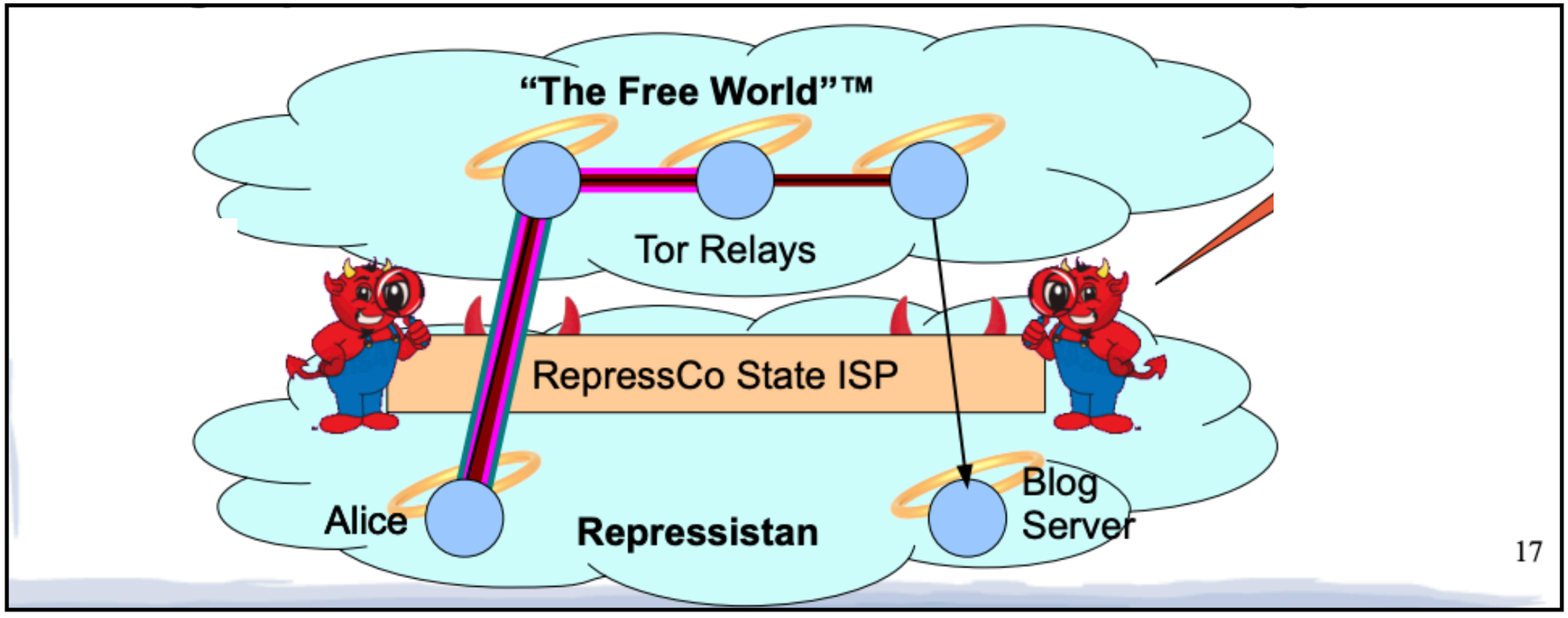
Computer Science

Fundamental problems with Tor

- Susceptible to **active attacks**
  - Will see more today
- Hard to **quantify**/analyze anonymity
  - Partly due to individualistic application-level behavior

Fingerprinting Attack

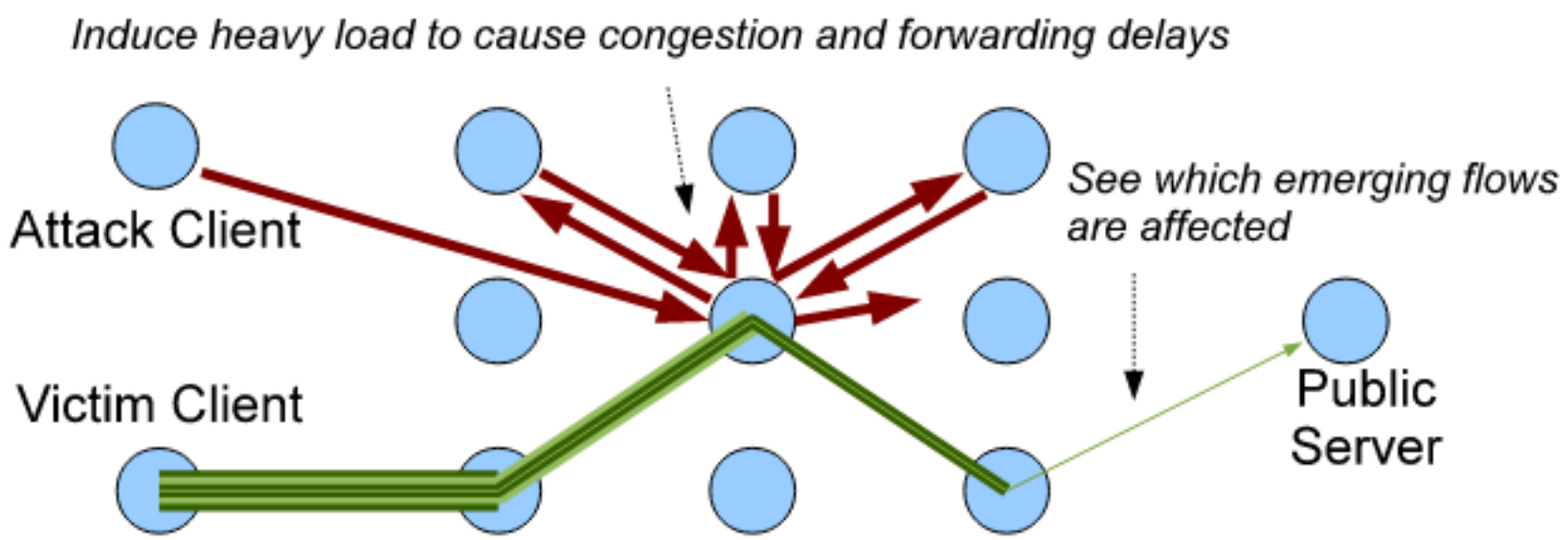
- Alice in Repressistan uses Tor to post on blog server hosted in Repressistan.
- State ISP controls both **entry** and **exit** hops
- Fingerprint & correlated traffic to deanonymize Alice



Slides borrowed from https://dedis.cs.yale.edu/dissent/pres/131024-austin.pdf

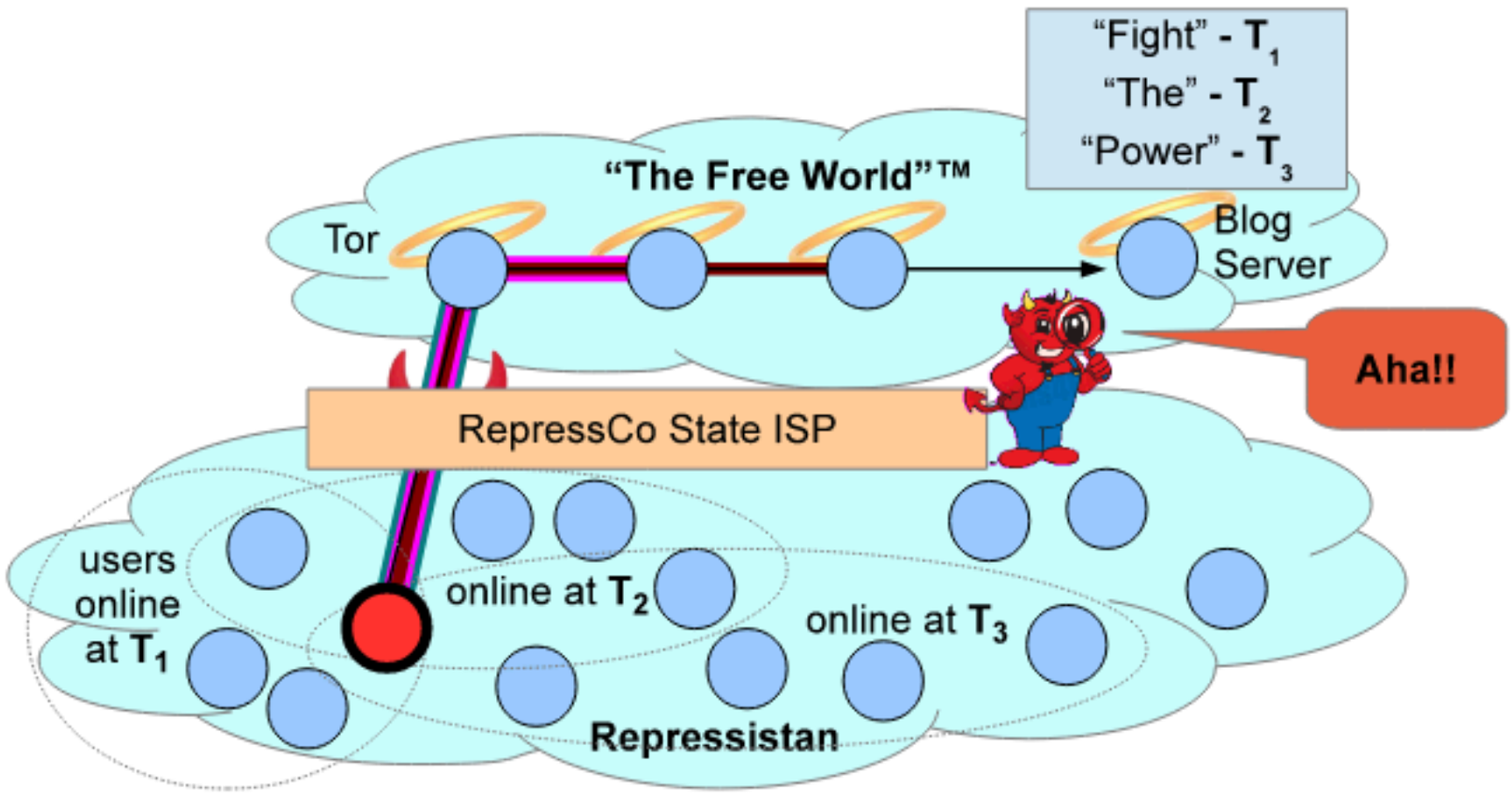
Congestion Attack

- Consider an attacker controls **only the exit hop**
  - E.g., the attacker located a pirate server, and wants to figure out which client is streaming movie from it
- Congestion attack: Create fake Tor users to **slow down** relays one by one; Observe which relay affects the exit hop's throughput.



Intersection Attack

- Dissent posts blog posts at different times
- Attacker (e.g., government) can request access log from ISP
- The intersection of users online at the time of blog posting is the victim



What does such logs look like?

- In 2019, hours before Epstein was officially found dead, an anonymous user on 4chan posted that Epstein had already died. The SDNY Attorney's office wanted to know who the poster was.
- Subpoenaed data from Apple, AT&T, and 4chan.
- Records now released as part of Epstein files: https://www.justice.gov/epstein/files/DataSet%209/EFTA00133349.pdf

RIDER
(Grand Jury Subpoena to AT&T, dated August 14, 2019)
Ref. No. 2019R01059

REQUEST:

For the Internet Protocol ("IP") addresses listed below at the following dates and times, please provide the following information:

[redacted] **on 08/10/2019 08:47:56 EDT**
on 08/10/2019 08:16:36 EDT
on 08/10/2019 08:21:58 EDT

1. All subscriber identifying information including, but not limited to:

   a. Name;
   b. User name or other subscriber identity or number;
   c. Address;
   d. Primary and alternate telephone numbers;
   e. Primary and alternate email addresses;
   f. Date of birth;
   g. Social Security number;
   h. Any temporarily assigned network address;
   i. MAC address; and
   j. Browser and operating system information.

2. Records of session times and durations and any IP addresses used by the subscriber at the beginning, end, and anytime during these sessions.

3. Length of service, including start date and types of service utilized.

4. Means and source of payment for services (including any credit card or bank account numbers).

5. Subscriber's call detail information for the period from January 1, 2014 to the present.

6. Account notes and logs, including any customer service communications or other correspondence with the subscriber.

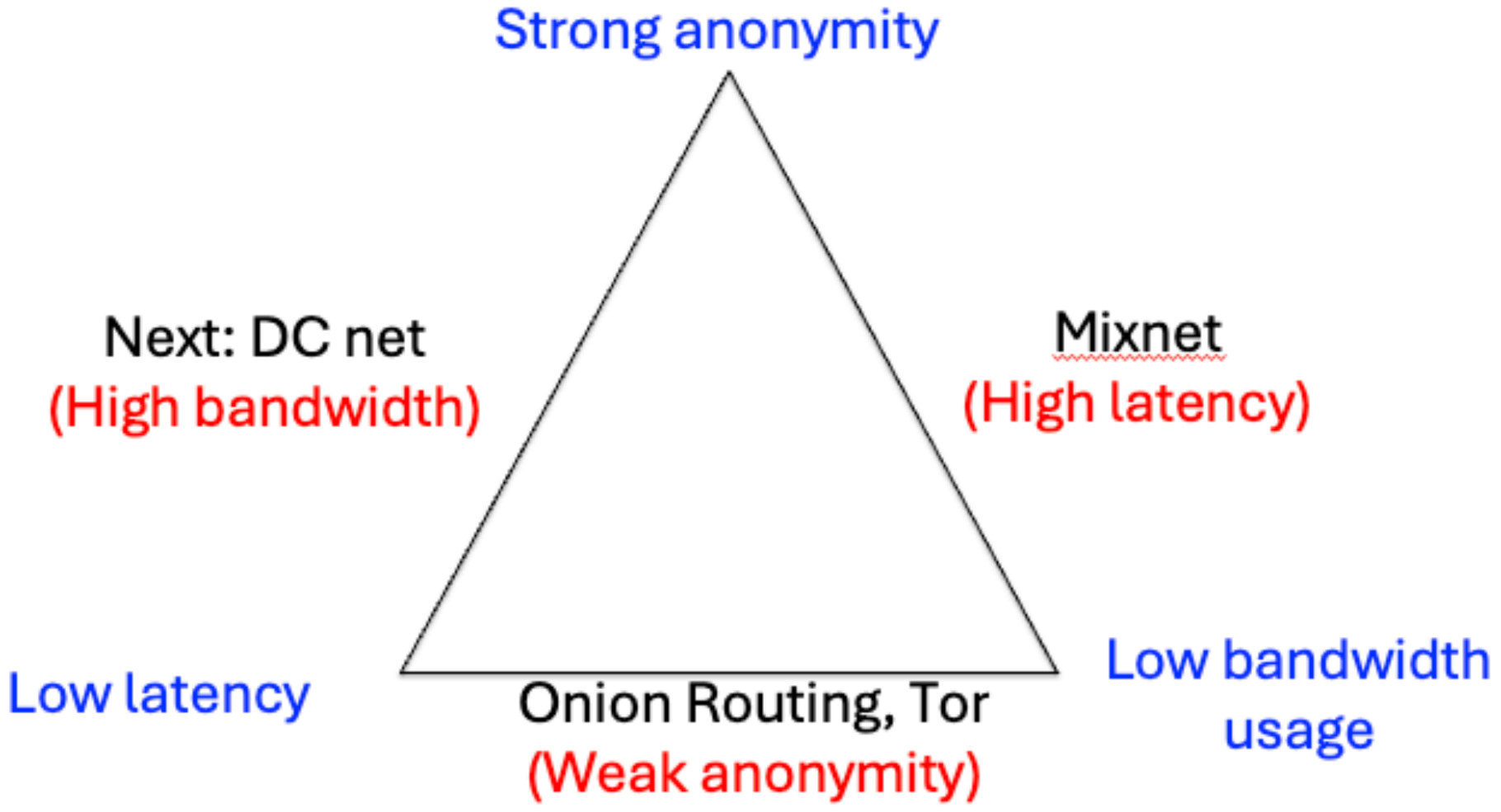
7. Investigative files or user complaints concerning the subscriber.

- The information requested from AT&T

Hard to analyze

- "Considerable theoretical work analyzes onion routing, but relies on idealized formal models making assumptions that are unenforceable and may be untrue in real systems – such as that users choose relays and communication partners at random – or depending on parameters that are unknown in practice, such as probability distributions representing user behavior."
- As a result, OR or Tor appear **unable to offer accurate, principled measurement of the quality of anonymity** a user might obtain.

Recall the trilemma



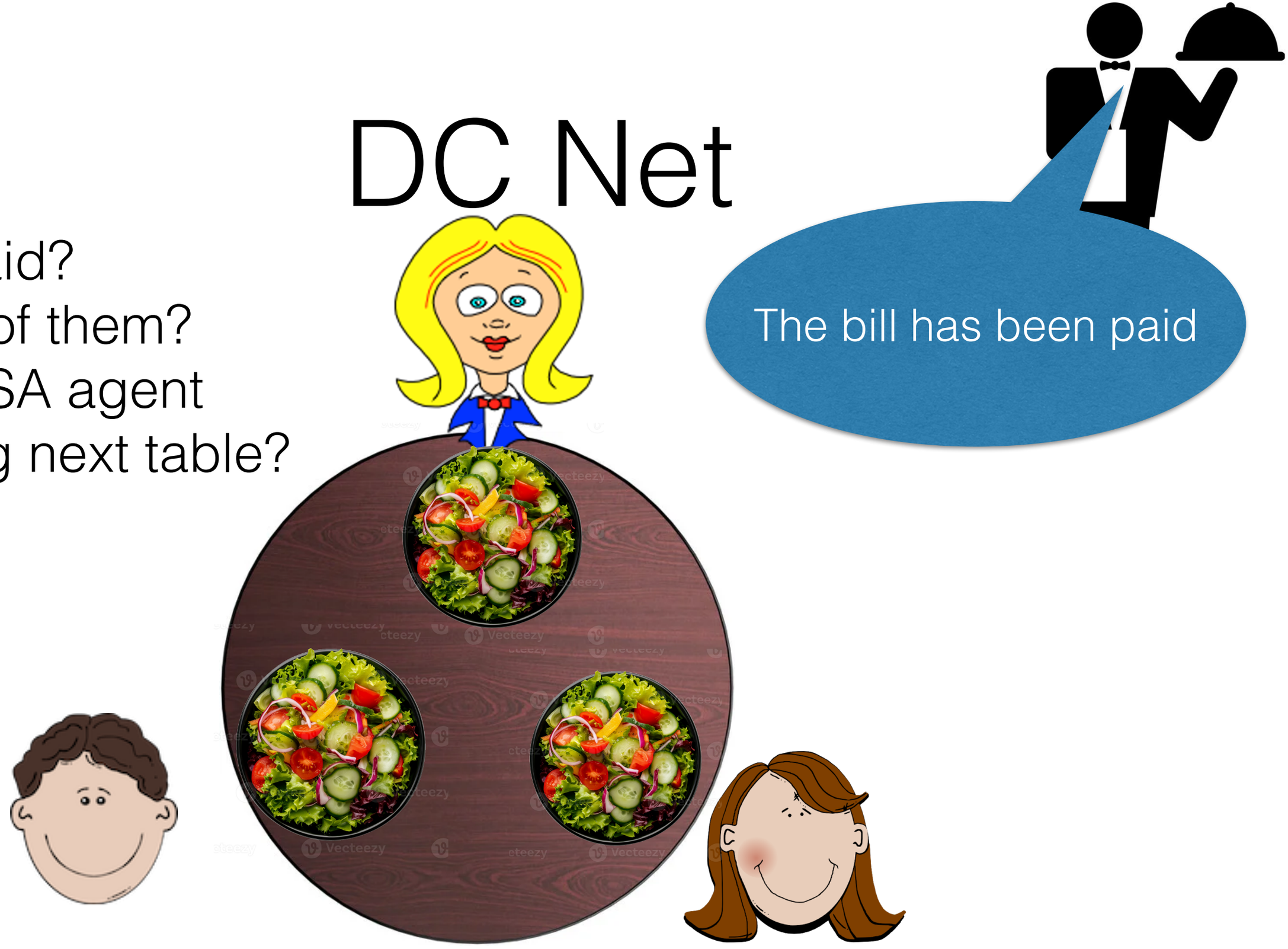
Dining Cryptographer

- The ***only*** well-studied anonymity protocol *not* based on sequential relaying.
- Invented by David Chaum in 80s
  - also invented Blind Signature (e-cash)
  - anonymous credentials
  - and mix nets
- Don’t be confused with Dining philosophers problem (which is much more boring)

DC Net

Who paid?

- One of them?
- Or NSA agent sitting next table?



Goal: figure out if one of the cryptographers paid or not w/o revealing who paid.

DC Net

Suppose Alice paid…

$C_{AB} \in \{0,1\}$

Alice

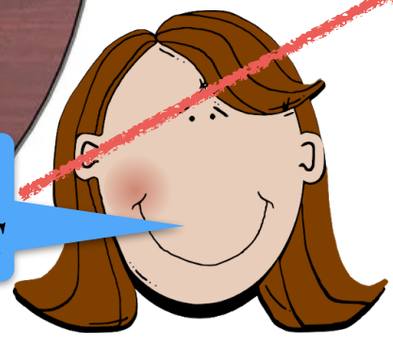
$m_A = 1 \oplus C_{AB} \oplus C_{AC}$

C_{AC}

$m_B = 0 \oplus C_{AB} \oplus C_{BC}$

$\oplus = 1$

Bob



Carmela

$m_c = 0 \oplus C_{BC} \oplus C_{AC}$

C_{BC}

Everyone computes

$m = m_A \oplus m_B \oplus m_C$

DC Net

$C_{AB} \in \{0,1\}$

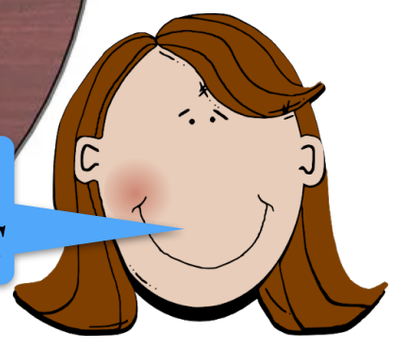
Alice

$m_A = 1 \oplus C_{AB} \oplus C_{AC}$

C_{AC}

Bob

$m_c = 0 \oplus C_{BC} \oplus C_{AC}$



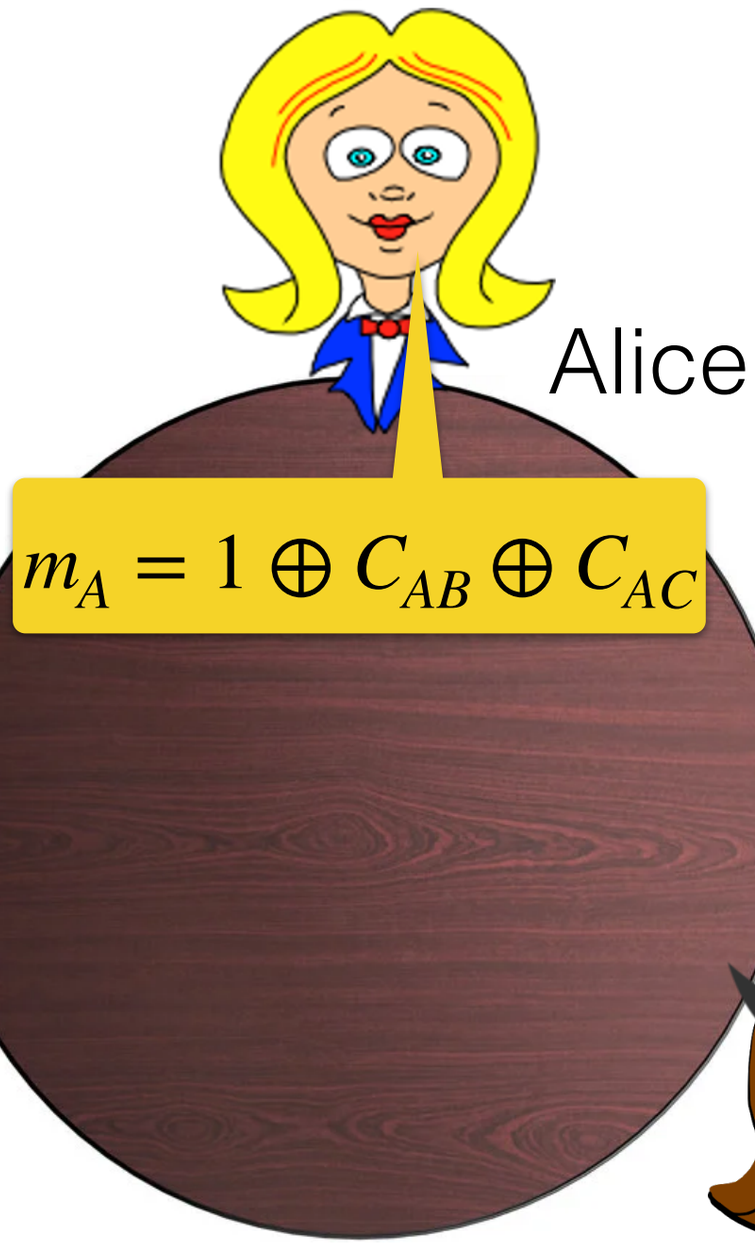
Carmela

C_{BC}

Bob still doesn't know C_{AC} and thus can't decrypt.

DC Net

$C_{AB} \in \{0,1\}$



Alice

C_{AC}

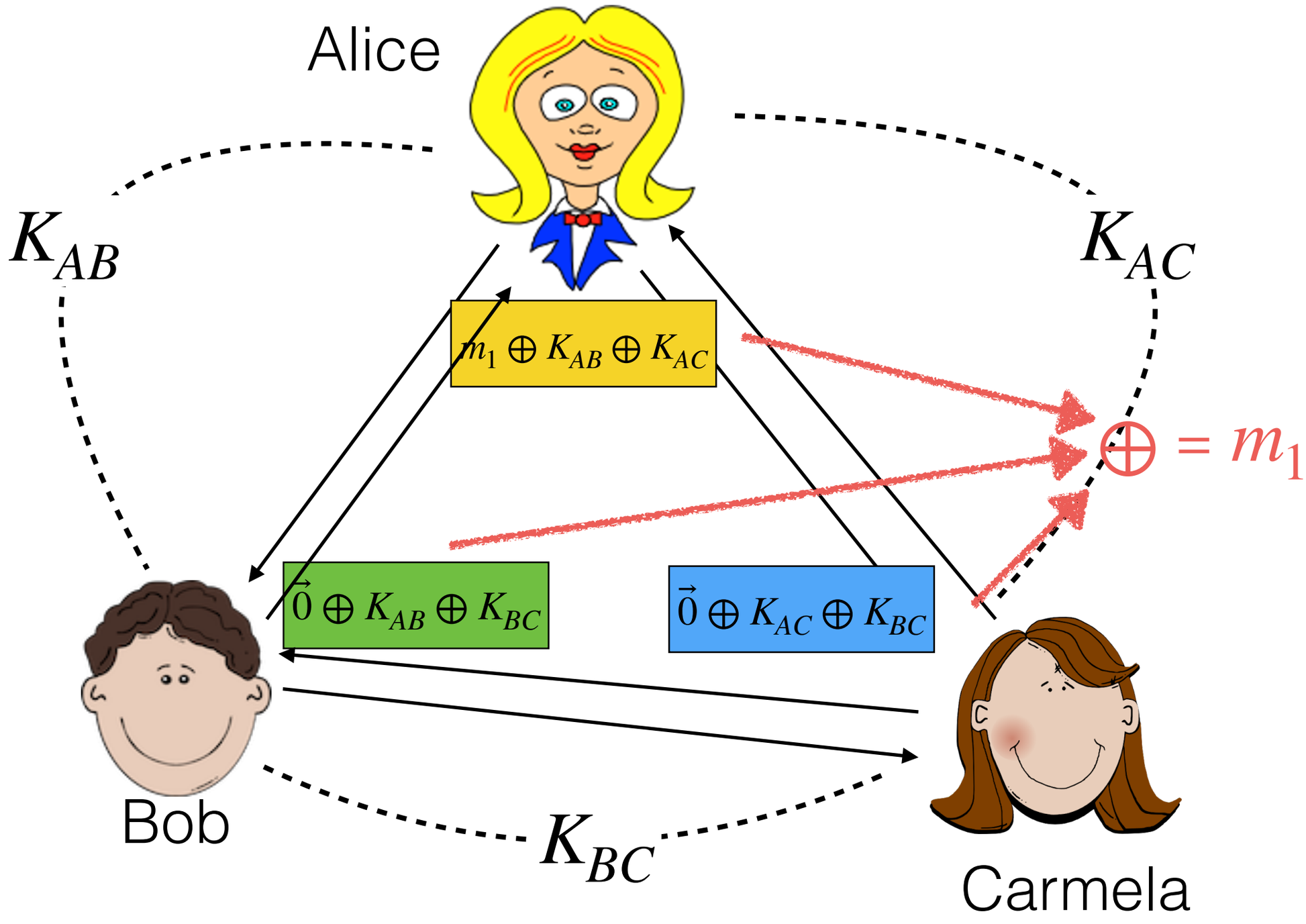
Carmela

Bob

Bob and Carmela **together** can break Alice's anonymity (hopeless anyway).

DC Net (generalized to n-bit)

- Parties share pair-wise keys $K_{A,B}, K_{A,C}, K_{B,C}$ (n-bit)

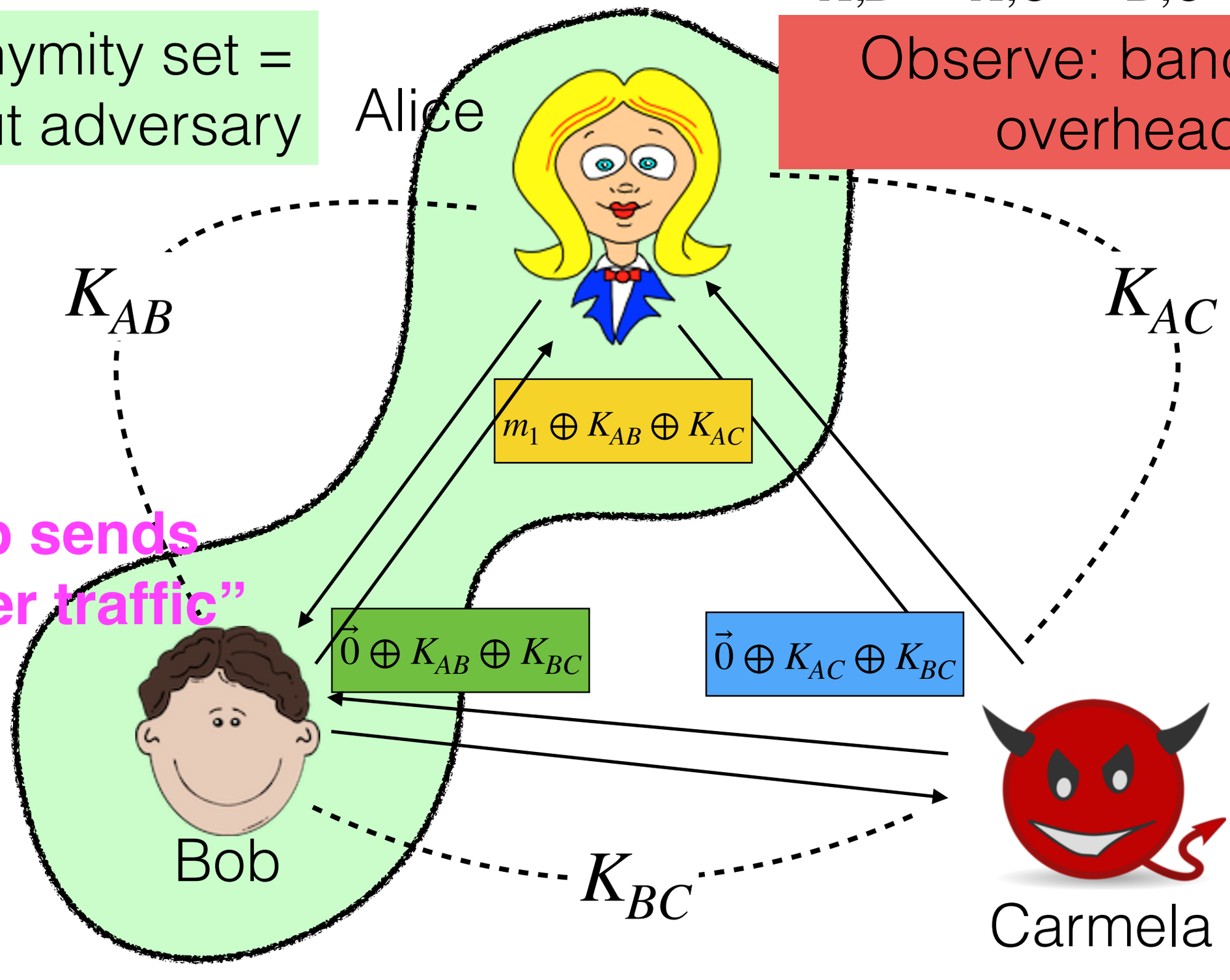


DC Net

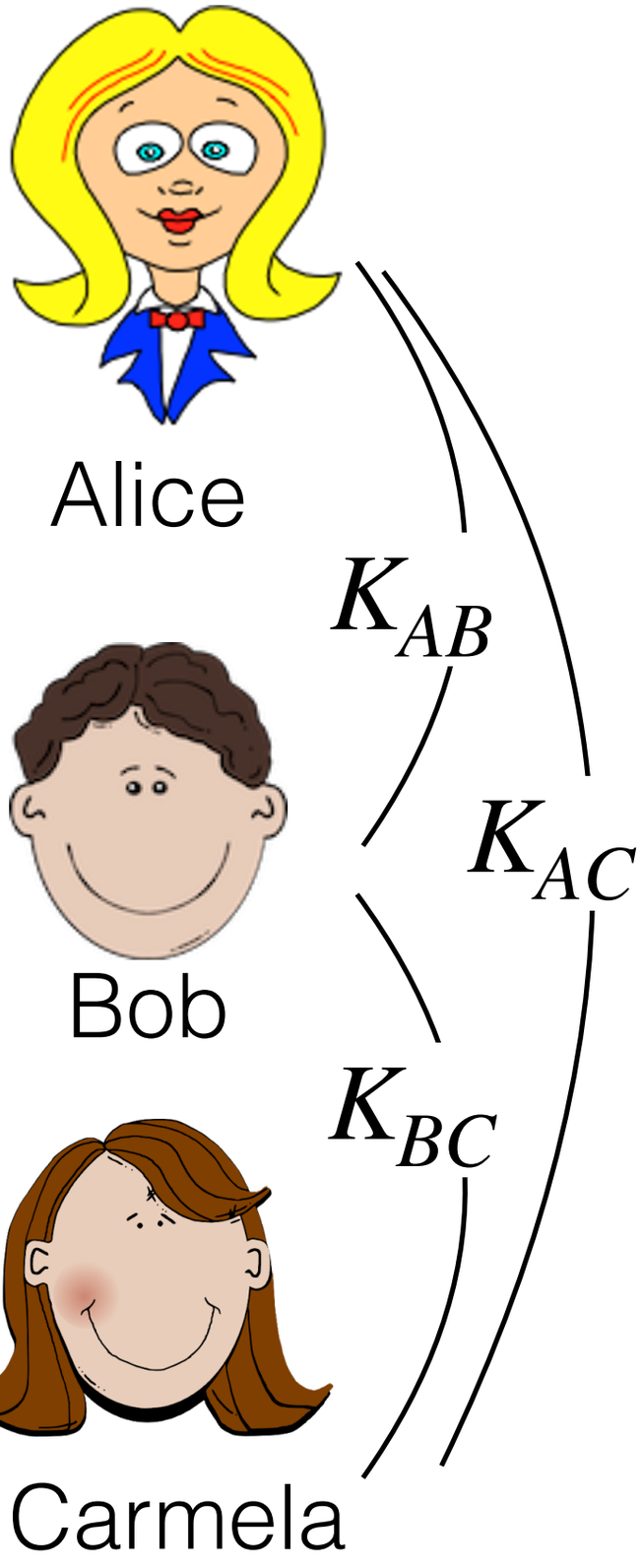
- Parties share pair-wise keys $K_{A,B}, K_{A,C}, K_{B,C}$ (n-bit)

Anonymity set = all but adversary

Observe: bandwidth overhead.



Multiple speakers



Divide channel into ***slots***.
Assuming **scheduling** with no collision.

broadcast
$m_a =$ | $\vec{0}$ | m_1 | $\vec{0}$ | $\oplus K_{AB} \oplus K_{AC}$

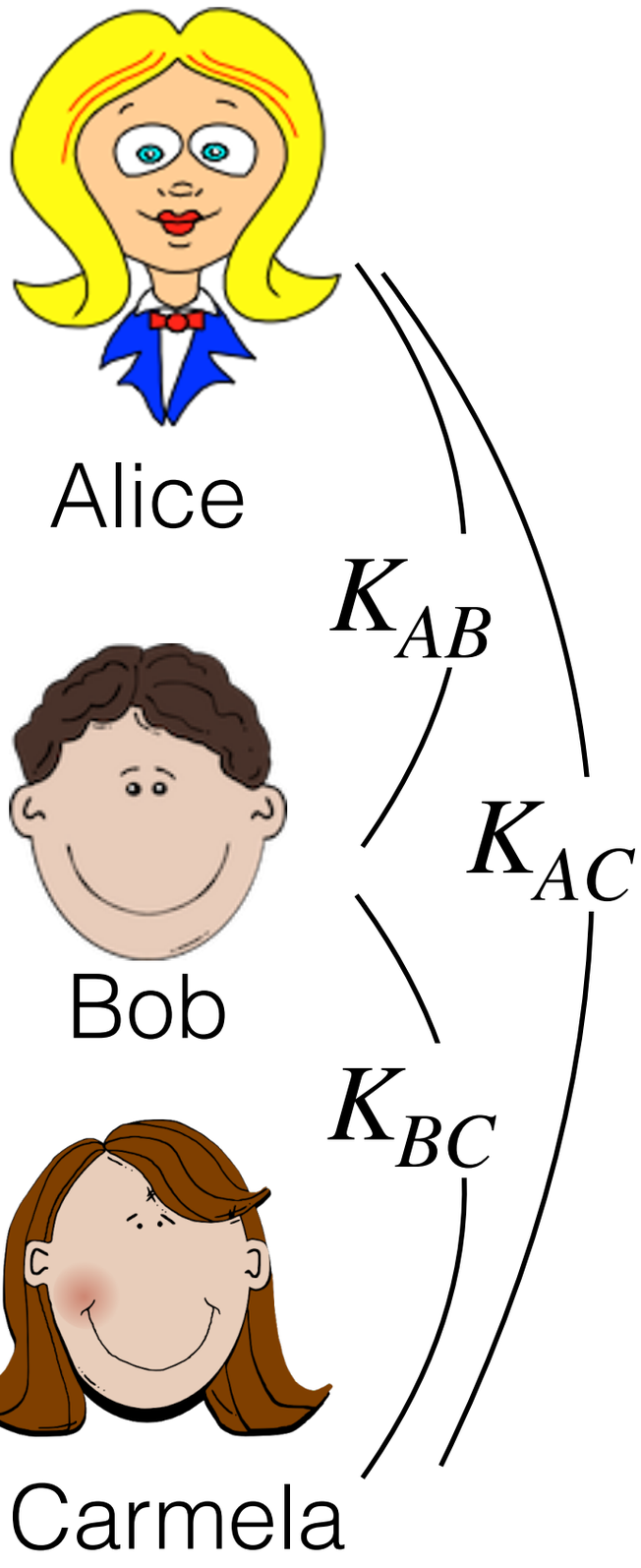
broadcast
$m_b =$ | m_2 | $\vec{0}$ | $\vec{0}$ | $\oplus K_{AB} \oplus K_{BC}$

broadcast
$m_c =$ | $\vec{0}$ | $\vec{0}$ | m_3 | $\oplus K_{BC} \oplus K_{AC}$

---

output: | m_2 | m_1 | m_3 |

Multiple speakers



(Case of **scheduling conflict**)

broadcast $m_a =$ | $\vec{0}$ | m_1 | $\vec{0}$ | $\oplus K_{AB} \oplus K_{AC}$

broadcast $m_b =$ | $\vec{0}$ | m_2 | $\vec{0}$ | $\oplus K_{AB} \oplus K_{BC}$

broadcast $m_c =$ | $\vec{0}$ | $\vec{0}$ | m_3 | $\oplus K_{BC} \oplus K_{AC}$

---

output: | $\vec{0}$ | $m_1 \oplus m_2$ | m_3

DC Net

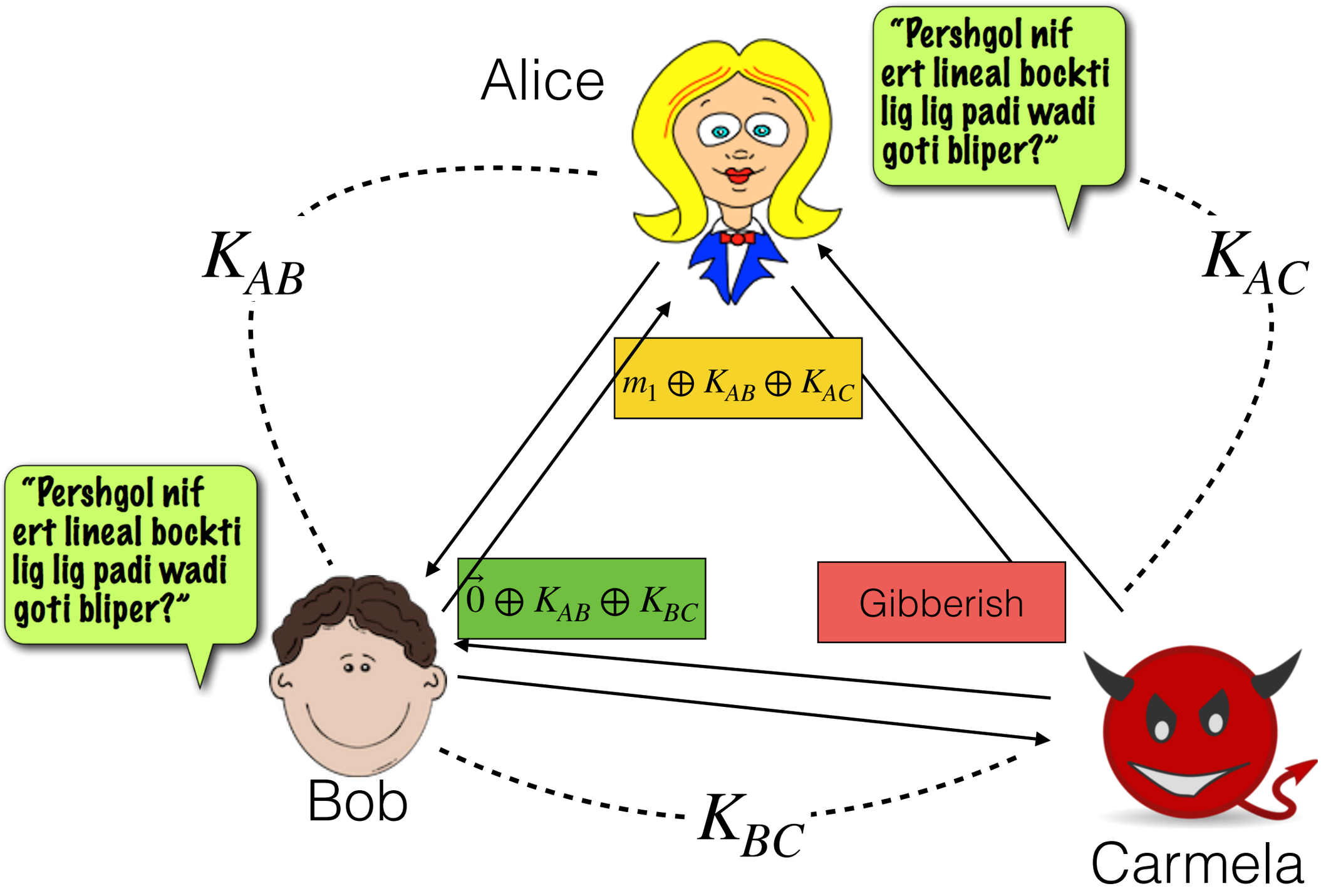
- Anonymous broadcast (ABC) among m parties
  - Assuming a closed group with fixed membership
  - Can also send *encrypted* messages
  - a scheduling protocol + a send protocol

- Key advantage: **Strong anonymity**
  - Immune from all attacks seen previously
  - all parties send same # of bits in the same round

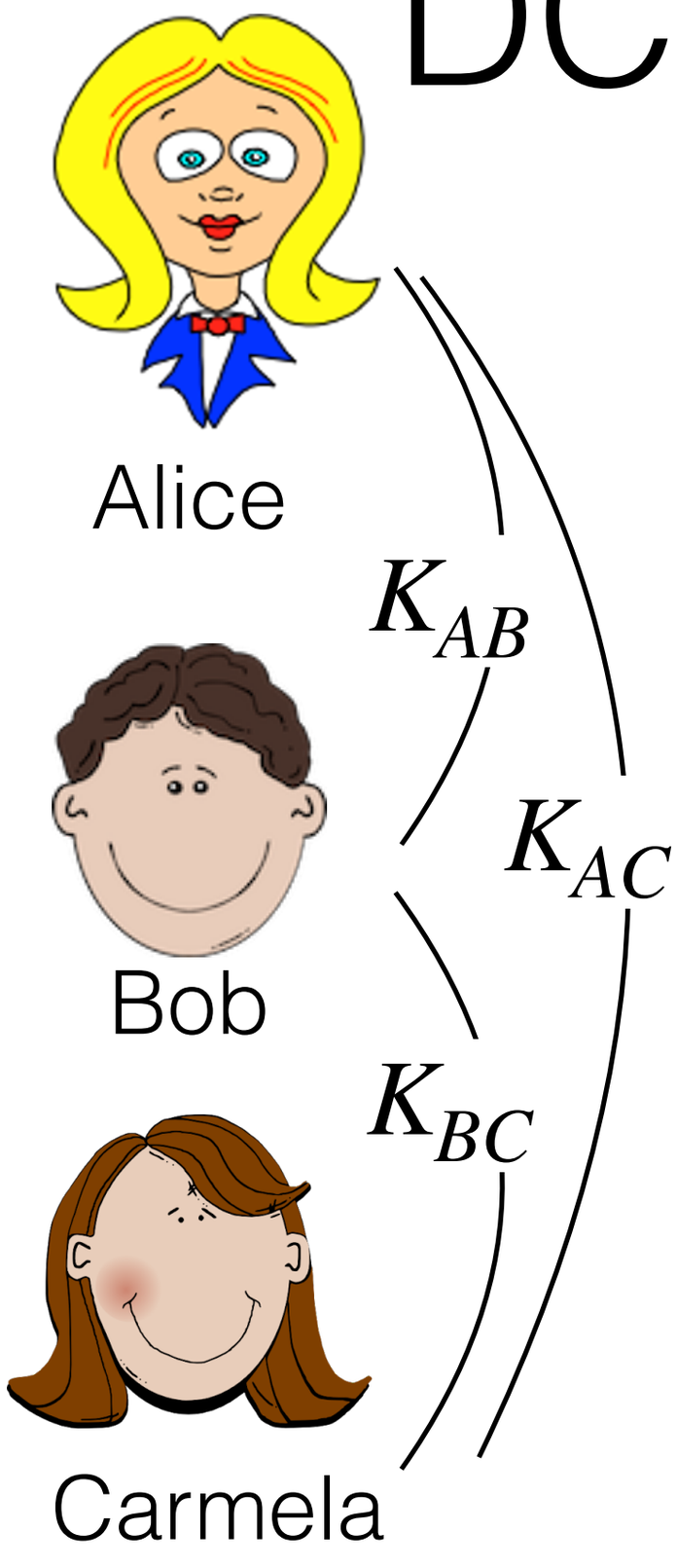
- Limitations?

Major one: Prone to disruption

- Anyone can disrupt the channel **anonymously**.



DC Net Limitations



- **Disruption**: Any party can DDoS the channel *anonymously*.
- **Computation and communication cost**: $O(n^2)$ keys in total; All-to-all communication $O(n^2)$ bits in total.
- **Node churn**: If *any* party leaves, *all* parties must start over.

Modern DC net pioneered by **Dissent**

dedis@yale Determinator Dissent EverCloud Tng

Dissent accountable anonymous group communication

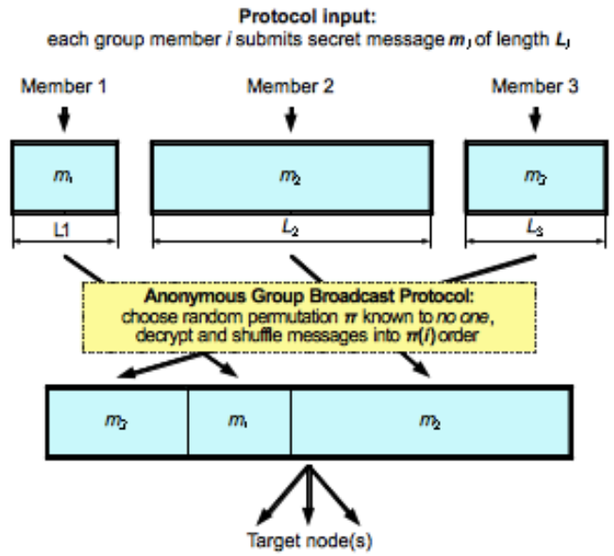
Background

The **Dissent** project is a research collaboration between Yale University and UT Austin to create a powerful, practical anonymous group communication system offering strong, provable security guarantees with reasonable efficiency. Dissent's technical approach differs in two fundamental ways from the traditional relay-based approaches used by systems such as Tor:

- Dissent builds on dining cryptographers and verifiable shuffle algorithms to offer *provable* anonymity guarantees, even in the face of traffic analysis attacks, of the kinds likely to be feasible for authoritarian governments and their state-controlled ISPs for example.
- Dissent seeks to offer *accountable anonymity*, giving users strong guarantees of anonymity while also protecting online groups or forums from anonymous abuse such as spam, Sybil attacks, and sockpuppetry. Unlike other systems, Dissent can guarantee that each user of an online forum gets *exactly* one bandwidth share, one vote, or one pseudonym, which other users can block in the event of misbehavior.

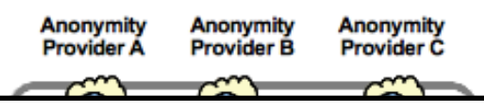
Dissent offers group-oriented anonymous communication best suited for broadcast communication: for example, bulletin boards, wikis, auctions, or voting. Members of a group obtain cryptographic guarantees of sender and receiver anonymity, message integrity, disruption resistance, proportionality, and location hiding.

Communication Model



Dissent conceptually builds a "shuffled send" anonymous multicast primitive

Anytrust cloud architecture



Principal Investigators

- Bryan Ford
- Joan Feigenbaum
- Vitaly Shmatikov

Project Leaders

- David Wolinsky
- Henry Corrigan-Gibbs
- Amir Houmansadr
- Ramakrishna Gummadi

PhD Students

- Ewa Syta
- Chad Brubaker
- Michael F. Nowlan
- Shu-Chun Weng
- John Maheswaran
- Hongda Xiao

Key Collaborators

- Aaron Johnson
- Paul Syverson
- Michael J. Fischer
- Avi Silberschatz

Dissent is a DARPA project led by Bryan Ford (Yale→EPFL), Joan Feigenbaum, Vitaly Shmatikov (UT Austin→Cornell Tech)

Dissent: Accountable Anonymous Group Messaging

Henry Corrigan-Gibbs and Bryan Ford
Department of Computer Science
Yale University
New Haven, CT, USA
henry.corrigan-gibbs@aya.yale.edu, bryan.ford@yale.edu

Dissent in Numbers: Making Strong Anonymity Scale

David Isaac Wolinsky, Henry Corrigan-Gibbs, and Bryan Ford
Yale University

Aaron Johnson
U.S. Naval Research Laboratory

- Dissent family of protocols: Dining-cryptographers Shuffled-Send Network
- **Accountability:** users can't disrupt group's operation without leaving evidence.
- **Scalability:** scale to many users

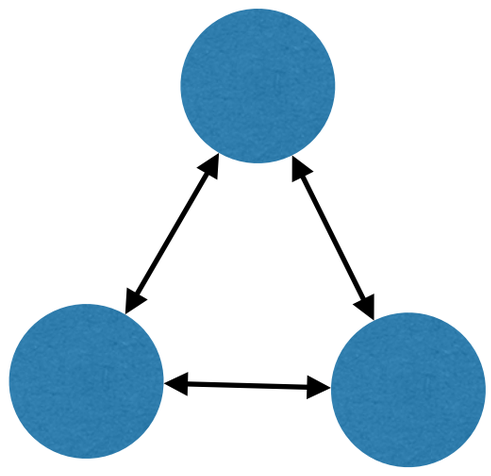
Client/server model

User 1

User 2

…

User 1000



Servers

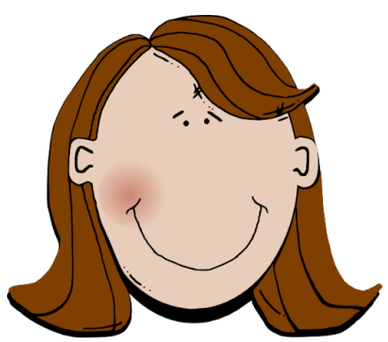
- Dissent system consists of
  - **Large** number of clients (thousands)
  - **A few** servers (e.g. <10)
- Each client connects to *one* server
  - E.g., one in their local region.
- Servers coordinate with each other.

$K_{1,1}, K_{1,2}$ and $K_{1,3}$

User 1

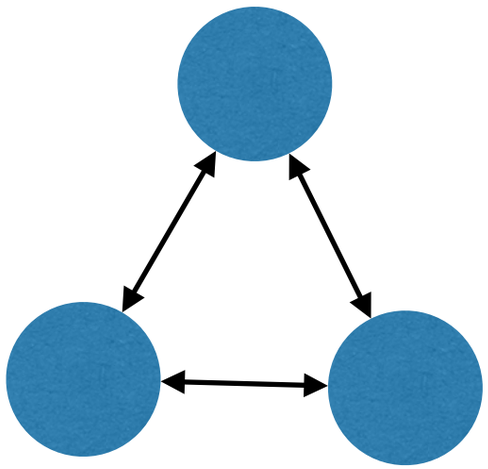
User 2

…



User 1000

Coin Sharing



Servers

- Each user exchanges pair-wise keys with all servers
- $O(nm)$ keys in total for n users and m servers
  - $m \ll n$ in practice
  - 3 Keys per user, instead of 1000!

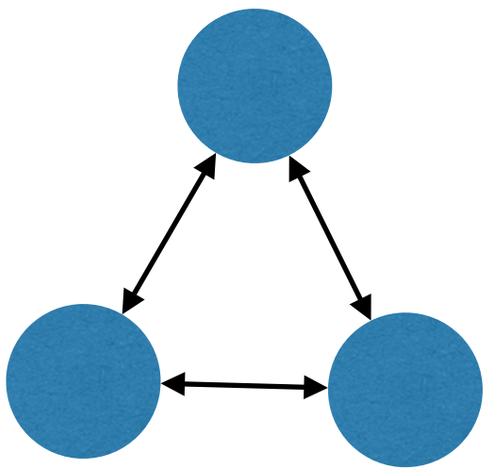
One Broadcast Round

User 1

User 2

…

User 1000



Servers

- Every user i sends $m_i \oplus_{j=1}^{3} K_{i,j}$ to a server
  - $K_{i,j}$ is user i's key with server j
- Every server j adds in $\oplus_{i=1}^{1000} K_{i,j}$
  - Server does more work but they are beefier than users.
- Servers produce the round output: XOR of all msgs

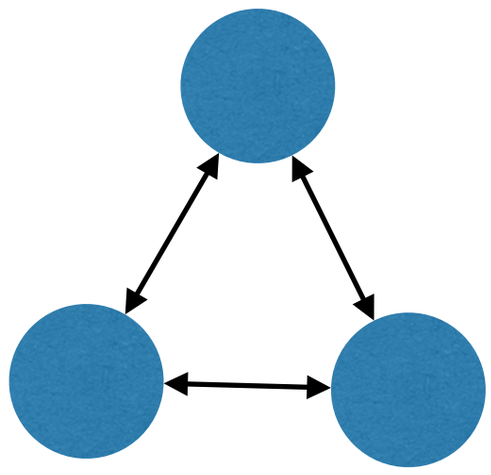
Benefits of client/server model

User 1

User 2

…

User 1000

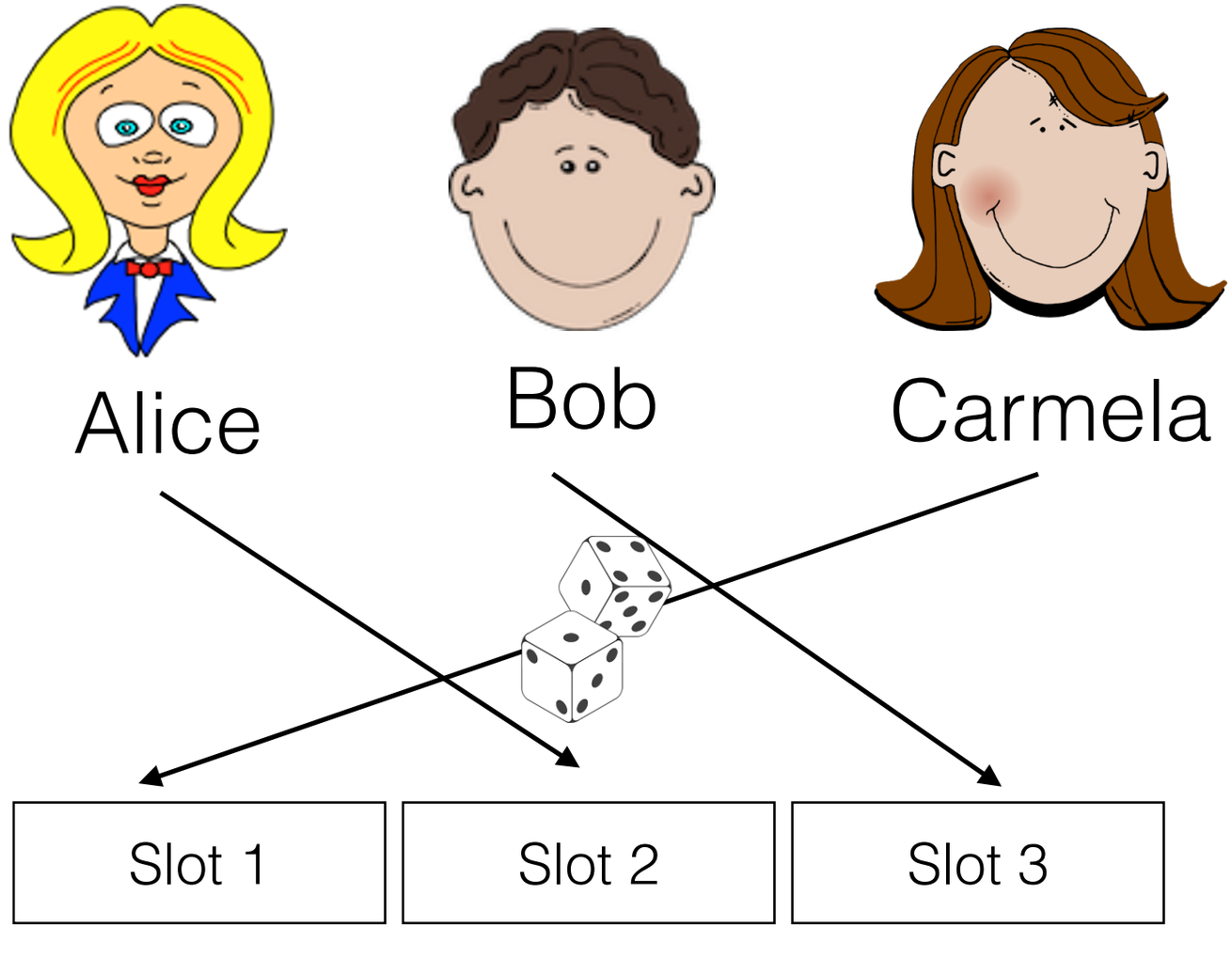


Servers

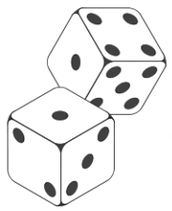
- Reduce client **computation**
  - From XORing with 1000 keys to 3 keys
- Reduce **communication**
  - From sending 999 msgs to 1
- [New] Servers can adapt to slow or offline clients
  - Simply time them out
- Anonymity is guaranteed if any server is honest **(any trust)**

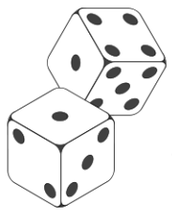
Accountability

- The core of accountability is a **scheduling problem**: assign parties to slots anonymously

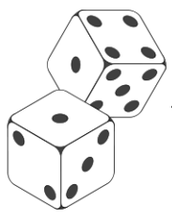


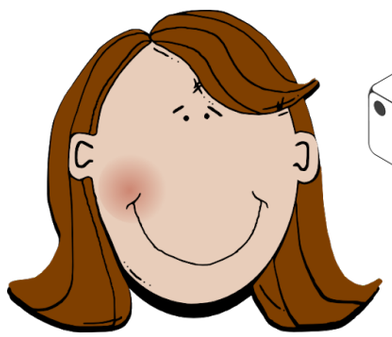
Idea in Dissent: **Verifiable Shuffle**

Alice  PK_A

 PK_B

Bob

 PK_C



Carmela

Mixnet run by servers

Shuffle outputs determines the schedule in next round.

PK_A

PK_C

PK_B

Slot 1

Slot 2

Slot 3

Accountability of sending

- Malicious users or servers may XOR non-zero bits in others' slots.
- Dissent: accusation scheme
  - adds little overhead when no disruptors
  - Enable servers to **identify** and **expel** disruptors quickly

Accusation phase

- Roughly three steps:
  - Identify **witness bit**: 0 bit in victim's message that got flipped in the output
  - Anonymously broadcast an accusation: "Bit #10 in Slot 2 of round 5 is corrupted, signed by owner of slot 2"
  - Servers **publish OTPs** that contribute to that particular bit.

1) Witness Bits

- **Witness bit**: 0 bit in victim's message that got flipped in the output
- What if all bits are 1? What if attacker can predict which bits are 1?

- Solution: randomize msg: $r \| PRG(r) \oplus m$
- Separately, Need to make sure even the last server can't choose keys adaptively (by commit-and-reveal)
- Each malicious bit-flip produce a witness bit with probability 1/2

2) Transmit Accusations

- Now the PKs are useful…
- Accusation “`Bit #10 in slot #2 is a witness bit`” — signed by owner of slot #2
- This msg needs to be broadcast anonymously. How?
- Can’t use the DC net, b/c the channel is being clobbered…
- Fall back to less efficient verifiable shuffle mixnet.

3) Identifying the malicious actors

- Upon receiving **“`Bit #10 in slot #2 is a witness bit`”** servers first check the basic correctness of accusation, then a complicated protocol…
- A few cases
  - Only malicious clients — easy: servers can decrypt client inputs and identify **who sent 1** in bit #10 of slot#2.
  - Only malicious servers — easy: clients have correct keys
  - Colluding clients and servers — read the paper
  - Fake accusation — read the paper carefully

Good ideas in Dissent

- Client/server model for scalability
- Trust in server is distributed
- Accountability instead of prevent (though quite complicated)

- Many works improving the client-server model of DC nets
- **Riposte (S&P'15)** improves on Dissent by having stronger trust assumptions and faster disrupter tracing techniques. Doesn't scale to many servers.
- **Blinder (CCS'20)** shifts traitor-sensitive behavior to a setup phase of its protocol.
- **Spectrum (NSDI'22)** is an anonymous broadcast system tailored specifically for the case of few, large-payload broadcasters.
- **Express (USENIX Security'21)** provides enhancements and a change of model. Expensive server computation.
- **Clarion (NDSS'22)** "[we target settings where] low latency anonymous broadcast is needed for a short period, e.g., during a live event, that can be planned ahead of time."

Some recent thinking

- Main improvements in ZIPNet
  - **Server cost**: diversification of trust requires low server cost
- How to increase the cost of jamming?

ZIPNet: Low-bandwidth anonymous broadcast from (dis)Trusted Execution Environments

Michael Rosenberg
University of Maryland
College Park, MD, USA
micro@umd.edu

Maurice Shih
University of Maryland
College Park, MD, USA
maurices@umd.edu

Zhenyu Zhao*
Tsinghua University
Beijing, China
jasonzhao404@gmail.com

Rui Wang*
Purdue University
West Lafayette, IN, USA
rui.wang.rw683@yale.edu

Ian Miers
University of Maryland
College Park, MD, USA
imiers@umd.edu

Fan Zhang
Yale University
New Haven, CT, USA
f.zhang@yale.edu

PETS 2025.