Design and Analysis of Efficient Anonymous-Communication Protocols

Thesis Defense

Aaron Johnson
Department of Computer Science
Yale University

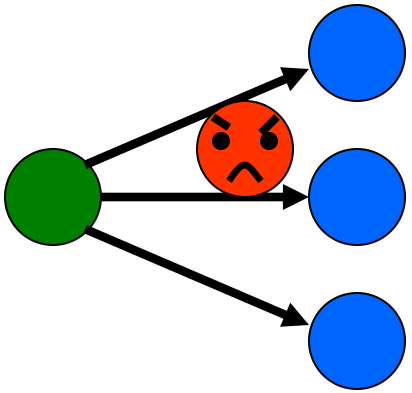
7/1/2009

Acknowledgements

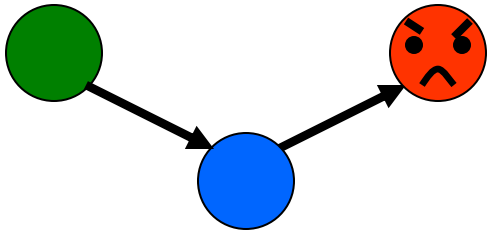
- Joan Feigenbaum
- Paul Syverson

Simple Anonymous-Communication Protocols

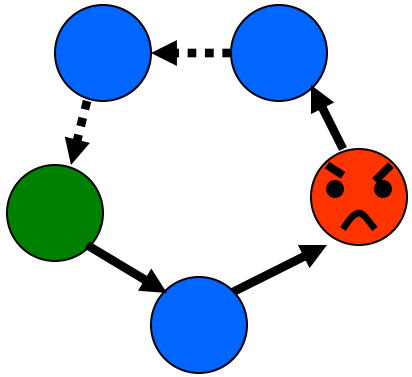
- **Broadcast:** Send an encrypted message to everyone



- **Proxy:** Send message via a proxy



- **Bus/Ring:** Send an encrypted message on a ring



Anonymous Communication in Practice

- Anonymous proxies
  - Anonymizer.net
  - SafeWeb
- anon.penet.fi
- Mixmaster
- Tor
- AN.ON

Anonymous Communication in Research

- Mix networks (Chaum, *CACM*, 1981)
- Dining cryptographers (Chaum, *Journal of Cryptology*, 1988)
- Onion routing (Goldschlag et al., *Information Hiding*, 1996)
- Crowds (Reiter and Rubin, *ACM TISSEC*, 1998)
- PipeNet (Dai, Cypherpunks mailing list, 1998)
- Xor-trees (Dolev and Ostrovsky, *ACM TISSEC*, 2000)
- Hordes (Levine and Shields, *JCS*, 2002)
- Tarzan (Freedman and Morris, *ACM CCS*, 2002)
- P5 (Sherwood et al., *IEEE S&P*, 2002)
- Anonymous buses (Beimel and Dolev, *JCS*, 2003)
- AP3 (Mislove et al., *ACM SIGOPS European Workshop* 2004)
- Salsa (Nambiar and Wright, *ACM CCS*, 2006)

Problem 1: Efficient protocols are useful but not rigorously understood.

Problem 1: Efficient protocols are useful but not rigorously understood.

Solution: We model and analyze *onion routing*.

Problem 1: Efficient protocols are useful but not rigorously understood.

Solution: We model and analyze *onion routing*.

Problem 2: Onion routing provides weak anonymity.

Solution: We design and analyze two protocols that provide stronger anonymity and similar efficiency.

Thesis Publications

1. ***A Model of Onion Routing with Provable Anonymity***
   with Joan Feigenbaum and Paul Syverson
   In *Proceedings of Financial Cryptography and Data Security '07* (*FC 2007*).
2. ***Probabilistic Analysis of Onion Routing in a Black-box Model***
   with Joan Feigenbaum and Paul Syverson
   In *Proceedings of the 2007 ACM Workshop on Privacy in Electronic Society* (*WPES 2007*).
3. **More Anonymous Onion Routing Through Trust**
   with Paul Syverson
   To appear in the *22nd IEEE Computer Security Foundations Symposium* (*CSF 2009*).
4. **Preventing Active Timing Attacks in Low-Latency Anonymous Communication**
   with Joan Feigenbaum and Paul Syverson
   (*under submission*)

Other Publications

1. ***Private Web Search***
   with Felipe Saint-Jean, Dan Boneh, and Joan Feigenbaum
   In *Proceedings of the 2007 ACM Workshop on Privacy in Electronic Society* (*WPES 2007*).
2. ***Online and Offline Selling in Limit Order Markets***
   with Kevin L. Chang
   In *Proceedings of the 4th International Workshop on Internet and Network Economics* (*WINE 2008*).

Outline

1. Introduction
2. Model & Definitions
3. Onion Routing – Possibilistic Analysis
4. Onion Routing – Probabilistic Analysis
5. Improved Anonymity Through Trust
6. Improved Anonymity by Eliminating Timing Attacks

Outline

1. Introduction
2. Model & Definitions
3. Onion Routing – Possibilistic Analysis
4. Onion Routing – Probabilistic Analysis
5. Improved Anonymity Through Trust
6. Improved Anonymity by Eliminating Timing Attacks

Model

- Network
  - Fully-connected
  - Asynchronous
  - Bounded message delay
- $\{m\}_k$ denotes private-key encryption by k.
- Users U, $|U| = n$
- Routers R
- Destinations D, $|D| = \delta$
- Adversary $A \subseteq U \cup R \cup D$
  $|A \cap R| / |R| = b$

Model

- Users *U* all try to send an anonymous message at the same time.
- Destinations *D* send response within a bounded time.
- Adversary *A*
  - **Active:** Can run arbitrary automata
  - **Local:** Three different situations to consider
    - Controls links from users (*e.g.* malicious ISP)
    - Controls destination
    - Controls some of the routers

Criteria

- **Definition (AnonymousMessage):** **AnonymousMessage(*d*,*m*):** This operation delivers message *m* to destination *d* and returns the response.
- **Definition (Relationship Anonymity):** **Relationship Anonymity** measures how well the adversary can determine the communication partners of a user.
- **Definition (Latency):** **Latency** is time between calling AnonymousMessage(*d*,*m*) and the receipt of *m* by *d*.
- **Definition (Message Complexity):** **Message complexity** is the ratio of total messages to the number of calls to AnonymousMessage.

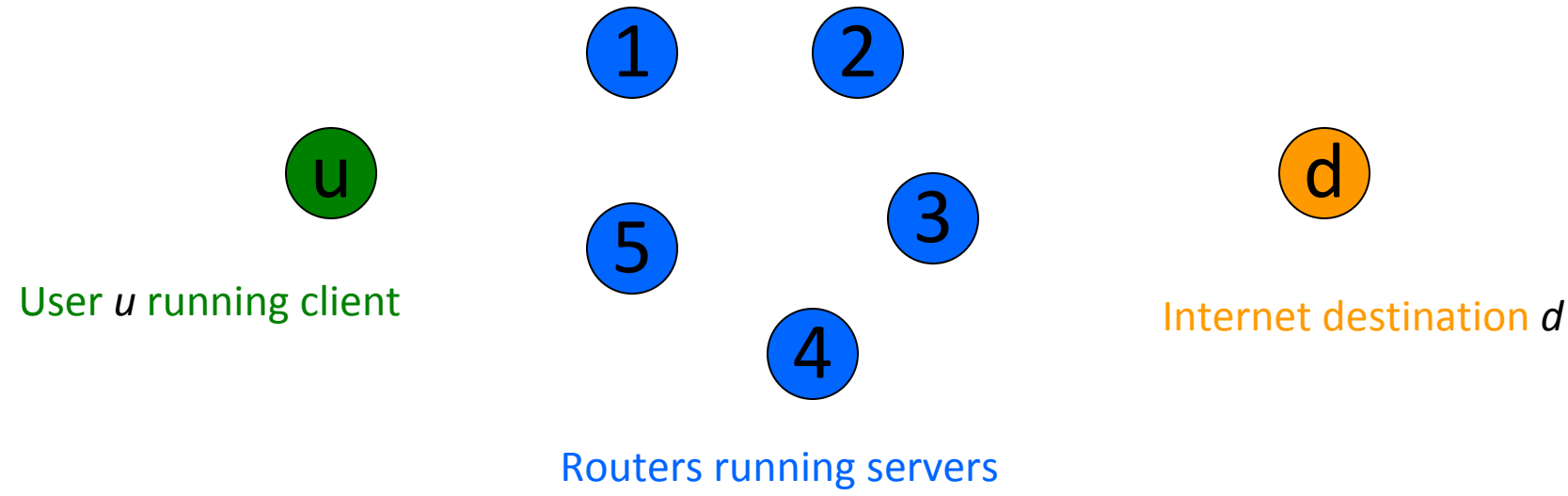
Outline

1. Introduction
2. Model & Definitions
3. Onion Routing – Possibilistic Analysis
4. Onion Routing – Probabilistic Analysis
5. Improved Anonymity Through Trust
6. Improved Anonymity by Eliminating Timing Attacks

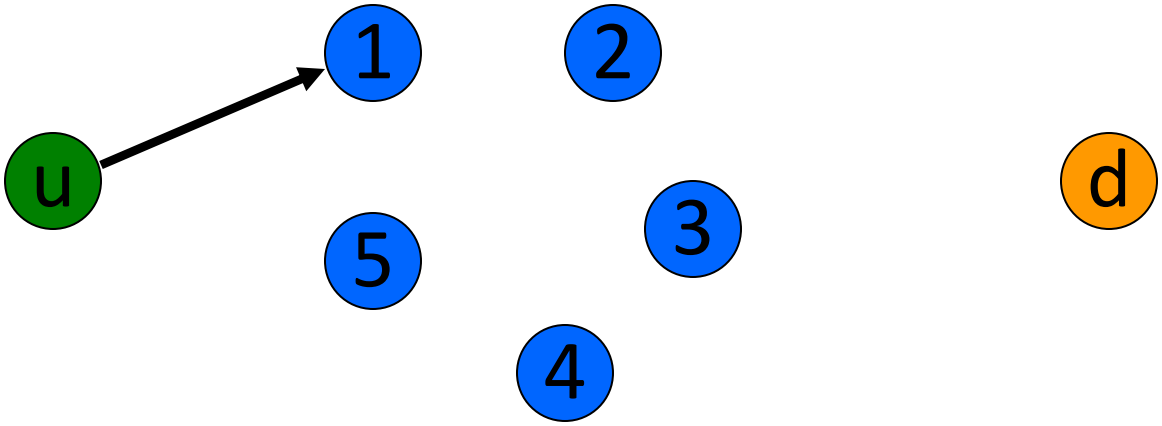
Contributions

- Create an I/O-automata model of onion routing
- Characterize those situations in which anonymity is provided

How Onion Routing Works

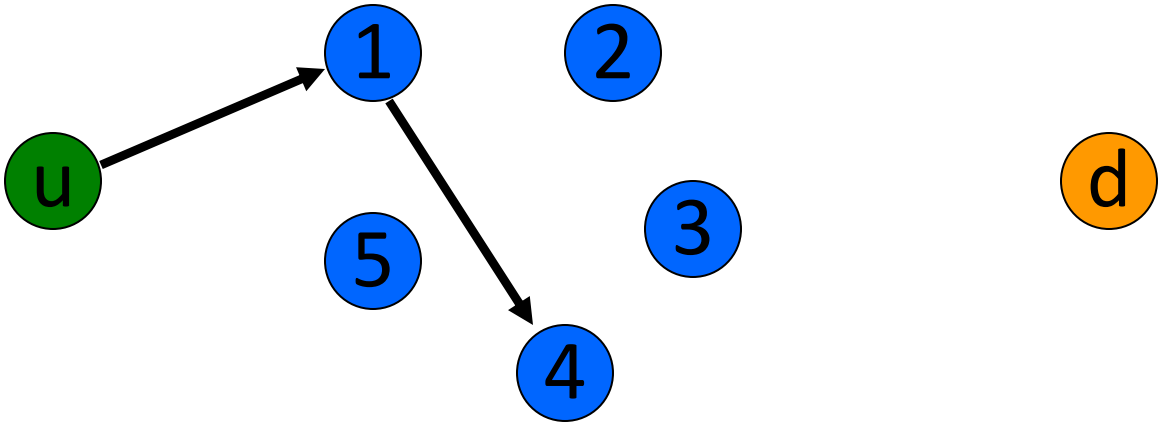


How Onion Routing Works



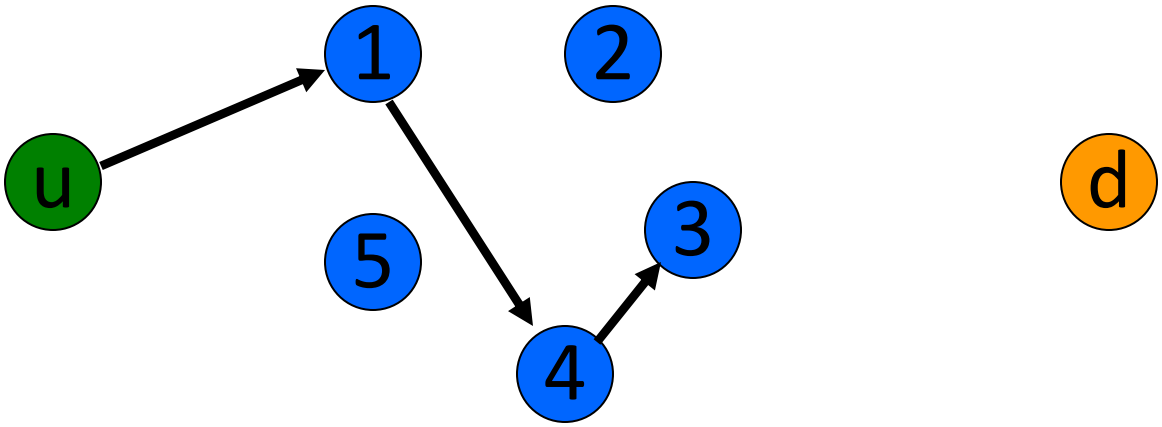
1. *u* creates *l*-hop **circuit** through routers

How Onion Routing Works



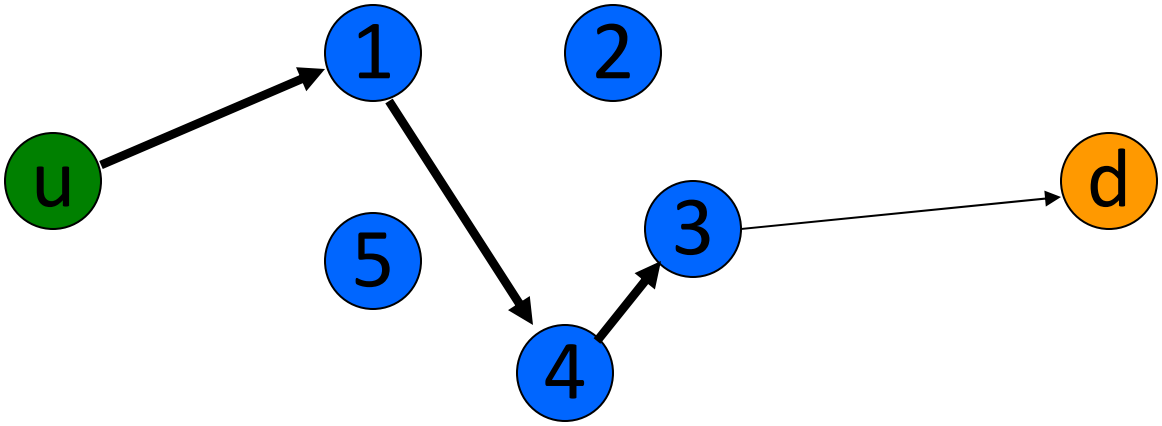
1. *u* creates *l*-hop **circuit** through routers

How Onion Routing Works



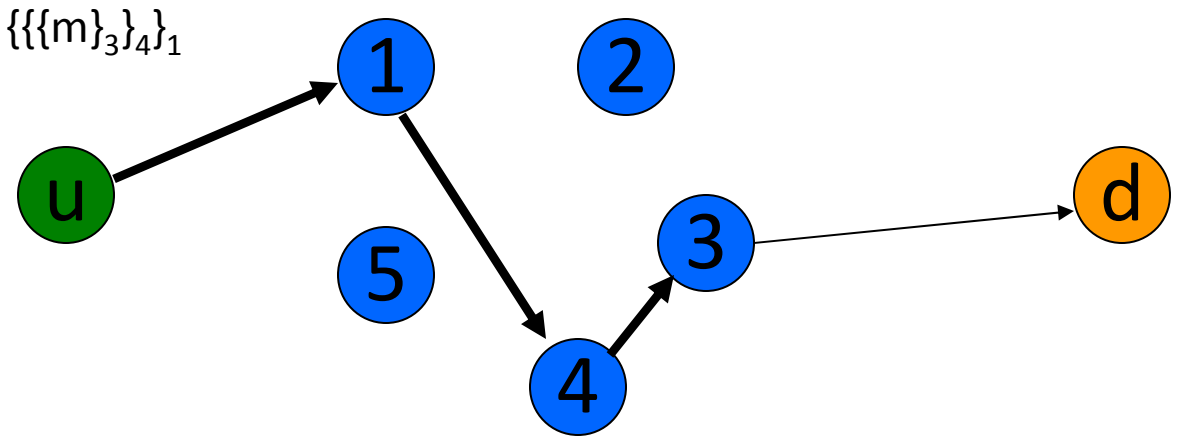
1. *u* creates *l*-hop **circuit** through routers

How Onion Routing Works



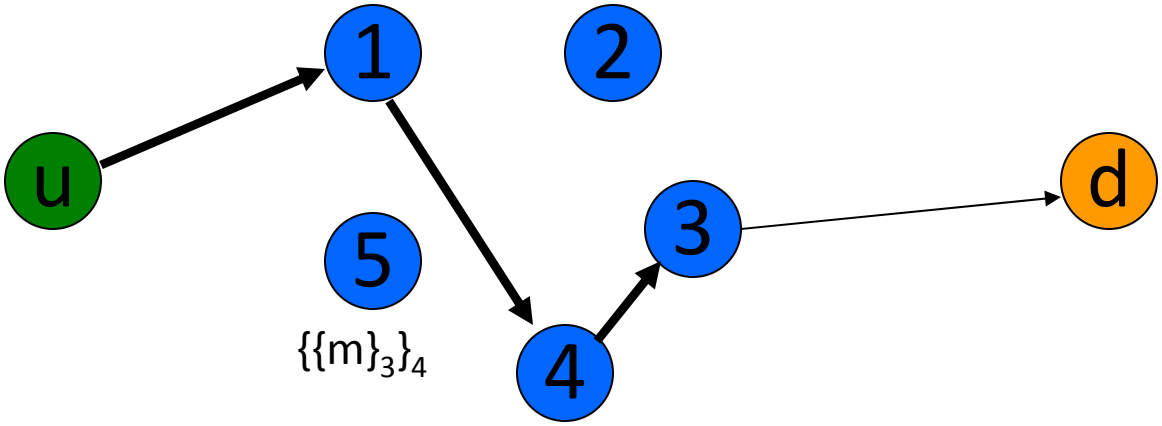
1. *u* creates *l*-hop **circuit** through routers
2. *u* opens a stream in the circuit to *d*

How Onion Routing Works



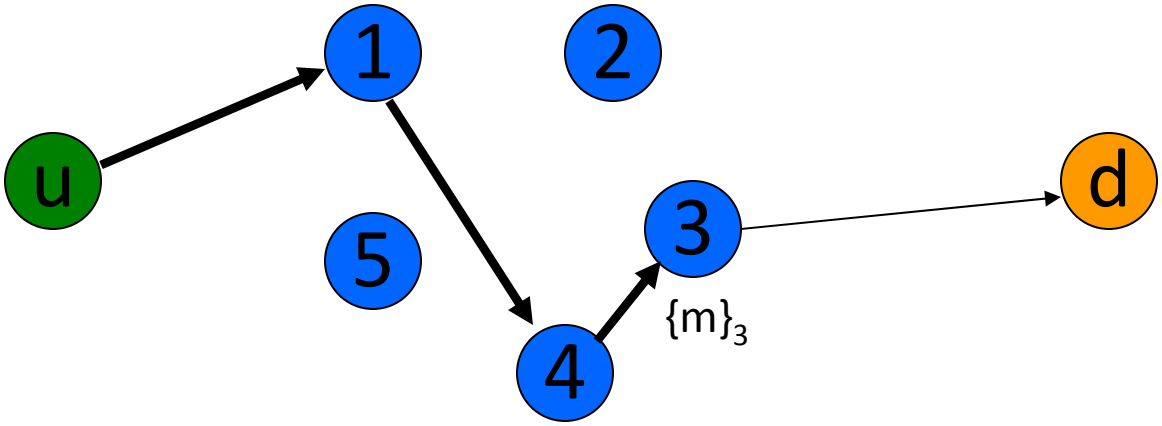
1. *u* creates *l*-hop **circuit** through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

How Onion Routing Works



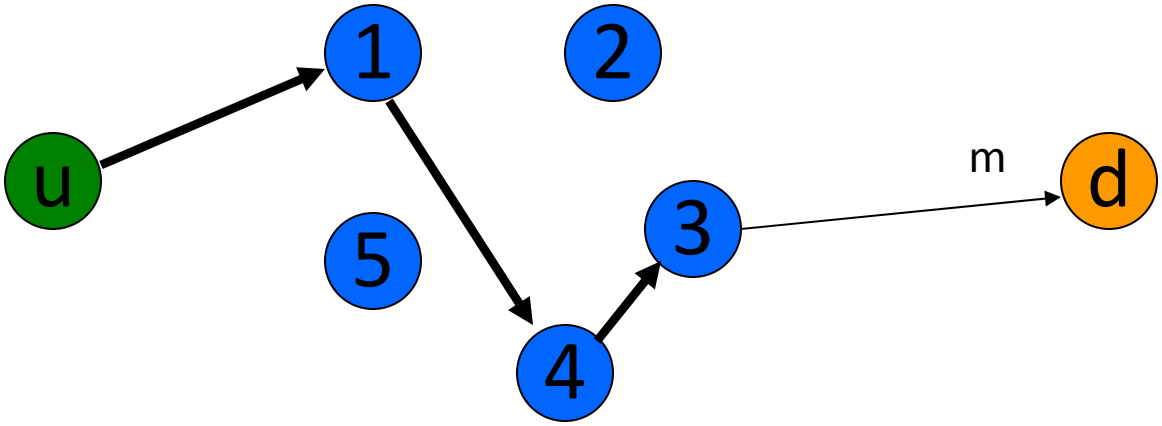
1. *u* creates *l*-hop **circuit** through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

How Onion Routing Works



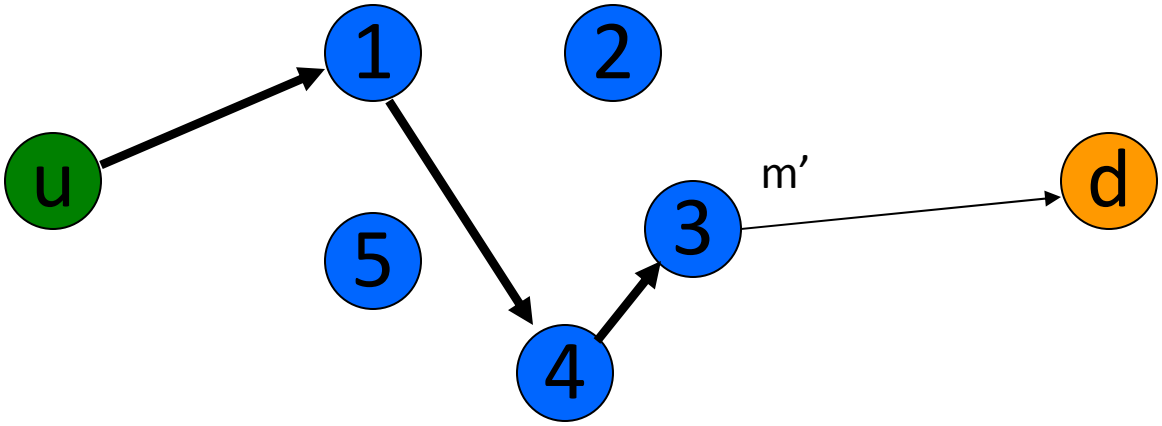
1. *u* creates *l*-hop **circuit** through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

How Onion Routing Works



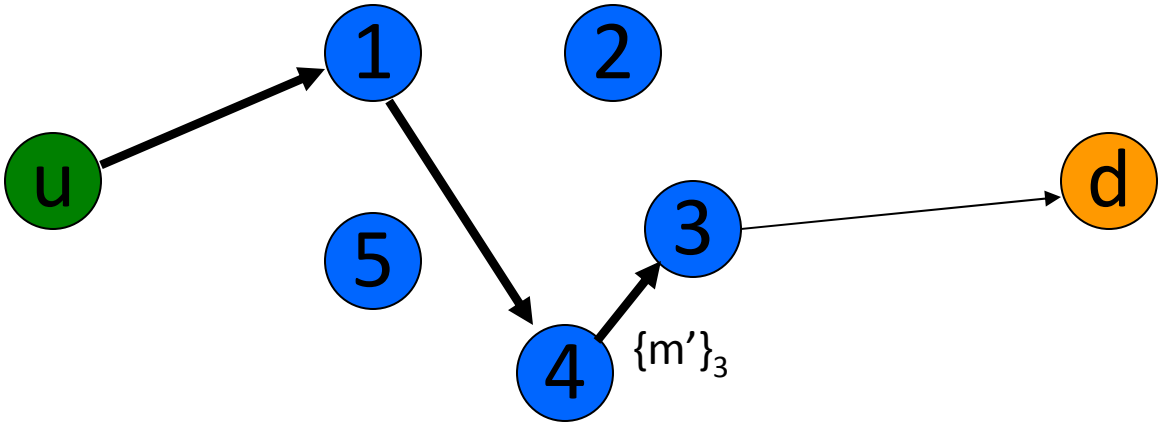
1. *u* creates *l*-hop **circuit** through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

How Onion Routing Works



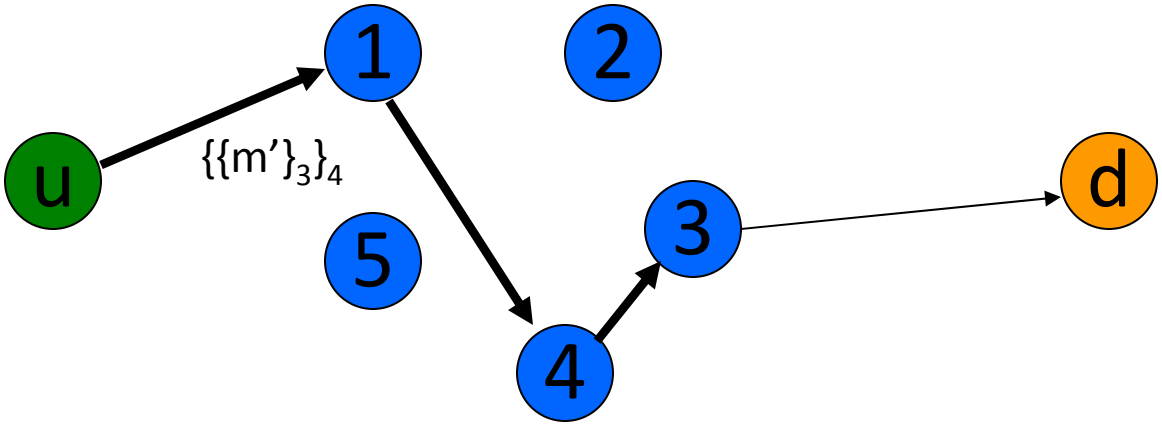
1. *u* creates *l*-hop **circuit** through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

How Onion Routing Works



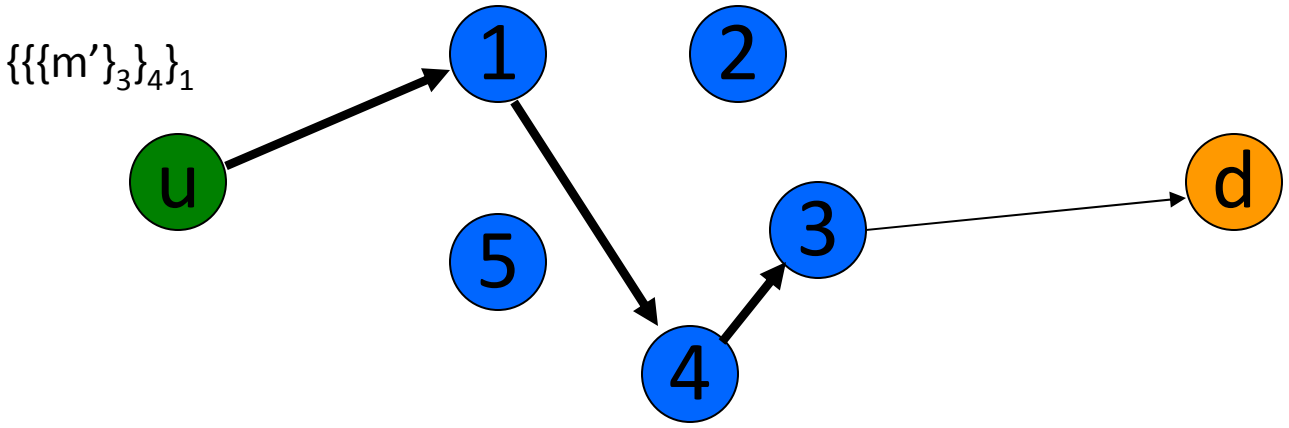
1. *u* creates *l*-hop **circuit** through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

How Onion Routing Works



1. *u* creates *l*-hop **circuit** through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

How Onion Routing Works



1. *u* creates *l*-hop **circuit** through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

Onion Routing – Possibilistic Analysis

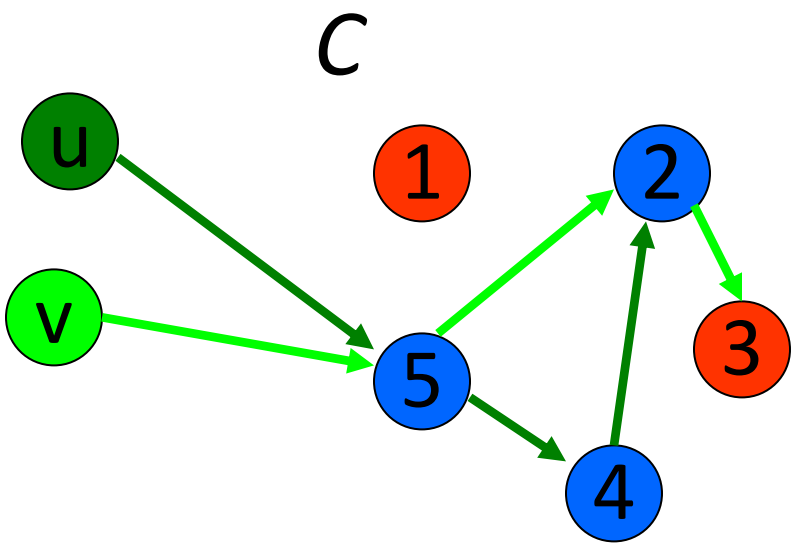
- Formalized protocol using I/O automata
  - Relies on abstract properties of cryptosystem
  - Simplification of actual protocol
    - No key exchange
    - No circuit teardown
    - No stream ciphers
    - No streams

Definition (relationship anonymity):

User *u* performs and destination *d* have **relationship anonymity** in execution α if there exists an indistinguishable execution β in which *u* does not communicate with *d*.

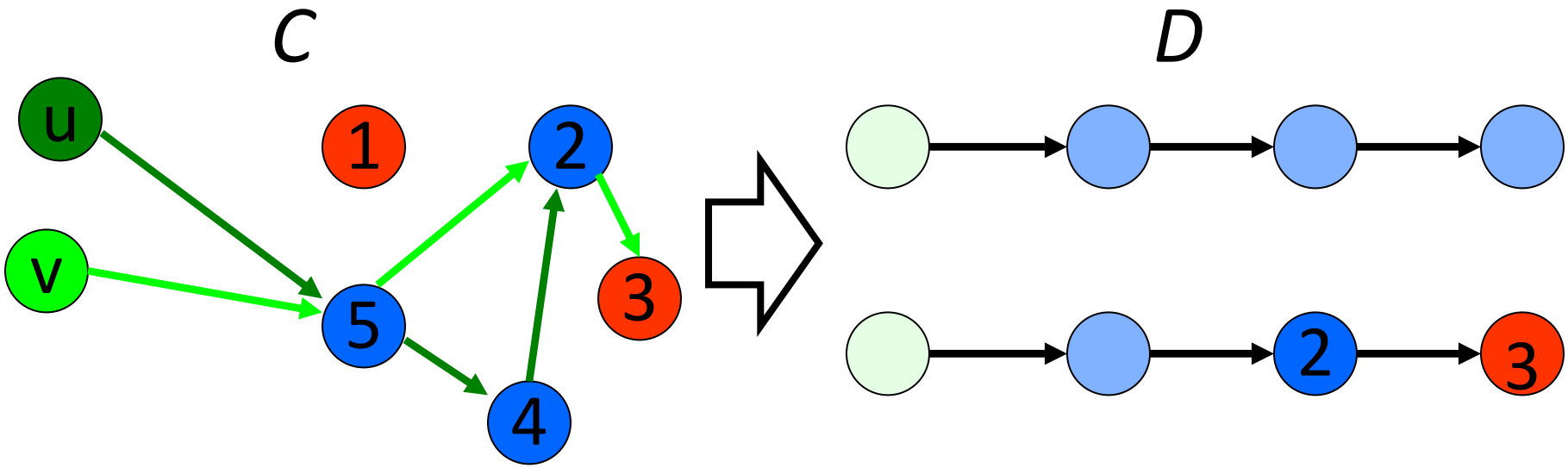
Onion Routing – Possibilistic Analysis

Theorem: Routers can only determine the parts of circuits they control or are adjacent to.



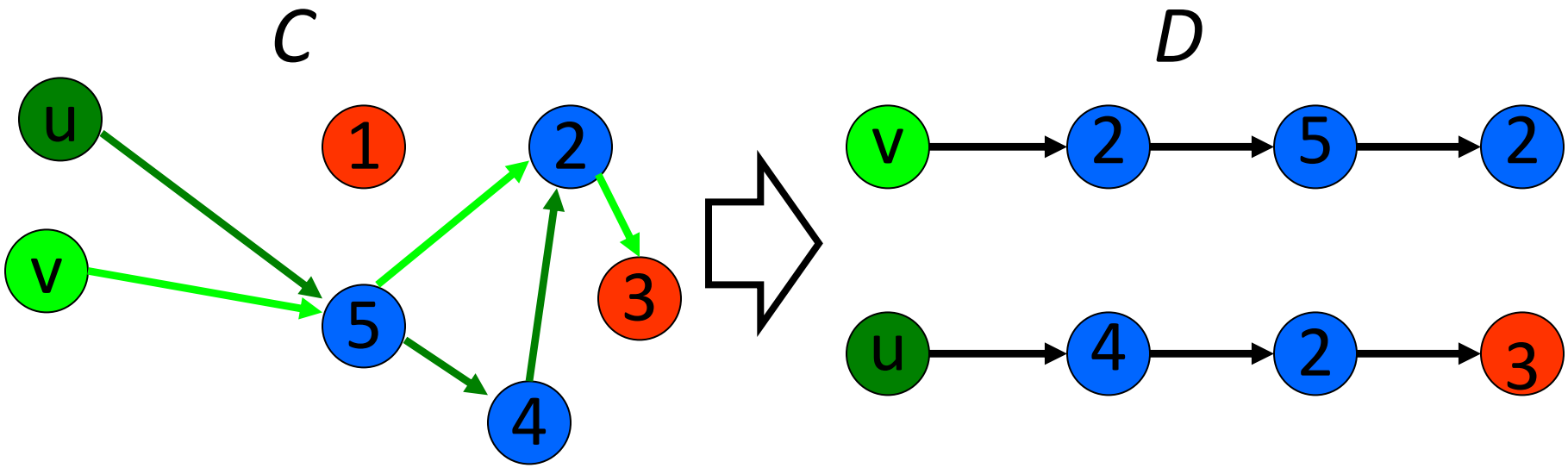
Onion Routing – Possibilistic Analysis

Theorem: Routers can only determine the parts of circuits they control or are adjacent to.



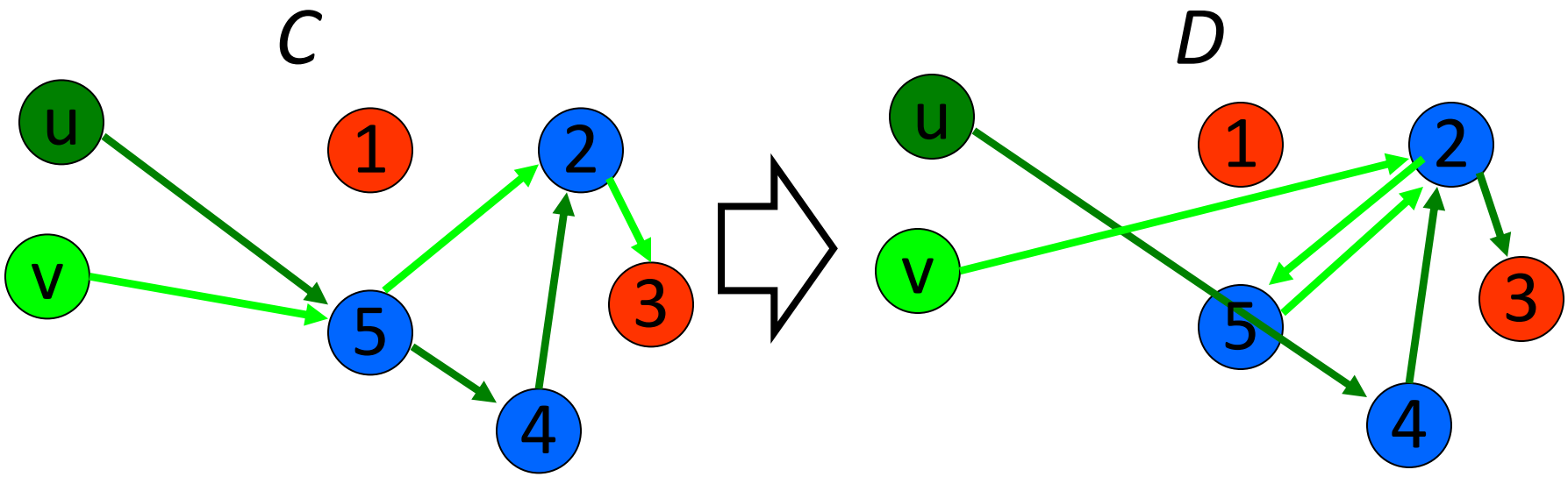
Onion Routing – Possibilistic Analysis

Theorem: Routers can only determine the parts of circuits they control or are adjacent to.



Onion Routing – Possibilistic Analysis

Theorem: Routers can only determine the parts of circuits they control or are adjacent to.

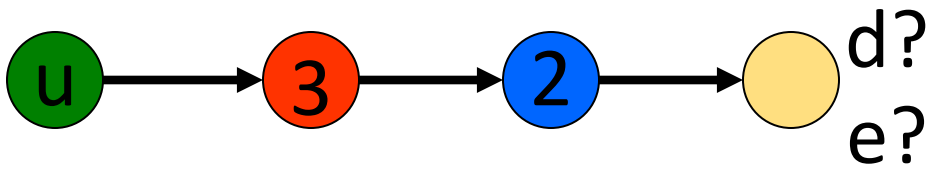


Relationship Anonymity

Corollary: A user and destination have relationship anonymity when:

Relationship Anonymity

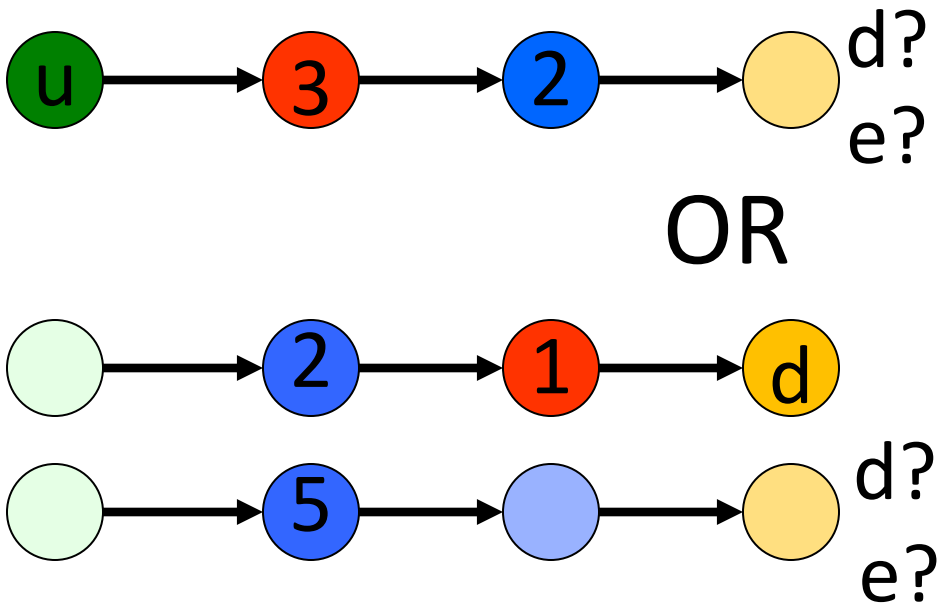
Corollary: A user and destination have relationship anonymity when:



The destination is unknown.

Relationship Anonymity

Corollary: A user and destination have relationship anonymity when:

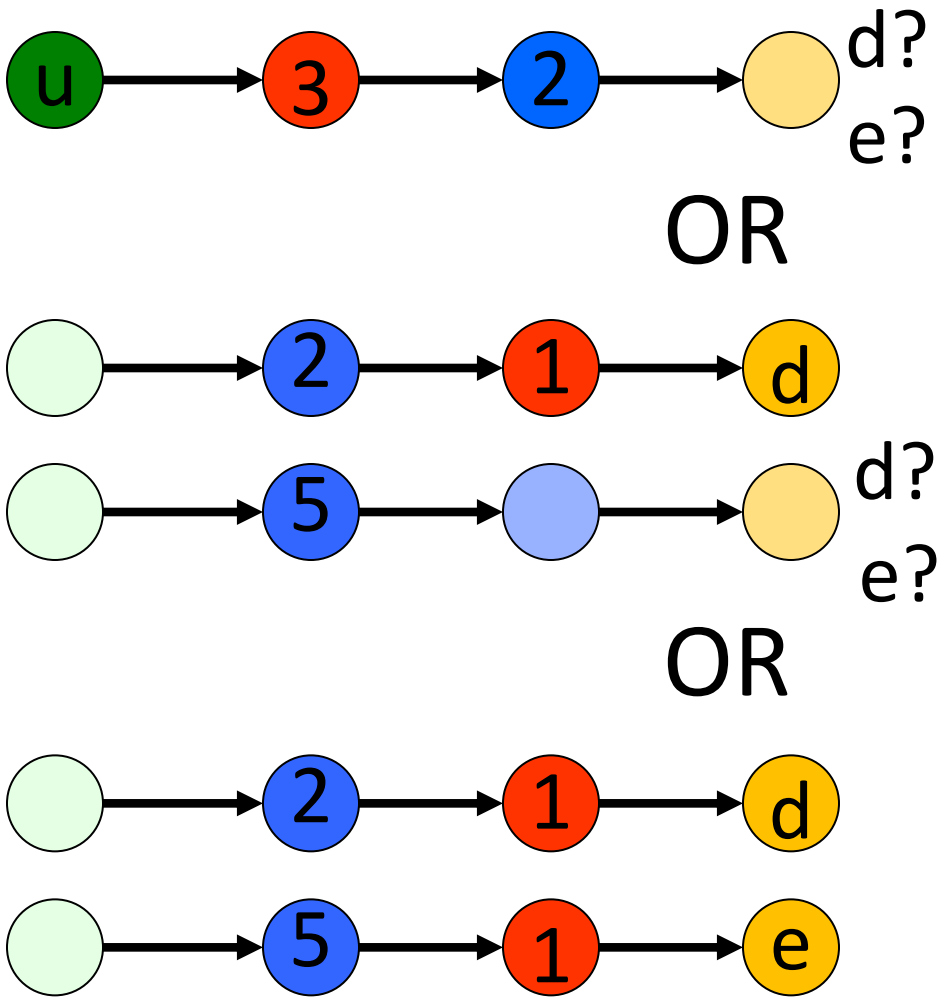


The destination is unknown.

The user is unknown and another unknown user has an unknown destination.

Relationship Anonymity

Corollary: A user and destination have relationship anonymity when:



The destination is unknown.

The user is unknown and another unknown user has an unknown destination.

The user is unknown and another unknown user has a different destination.

Protocol Efficiency

- Onion routing
  - Latency: *l*+1
  - Message complexity: *l*+1

Outline

1. Introduction
2. Model & Definitions
3. Onion Routing – Possibilistic Analysis
4. Onion Routing – Probabilistic Analysis
5. Improved Anonymity Through Trust
6. Improved Anonymity by Eliminating Timing Attacks

Contributions

1. Create a probabilistic model of onion routing
2. Analyze relationship anonymity
   a. Provide worst-case bounds
   b. Examine a typical case

Model with Probability

- Users choose routers uniformly at random
- User *u* chooses destination *d* with probability

  p^u_d
- Events in executions occur uniformly at random

Probabilistic Anonymity

Let C be the selection of routers and destinations.

Let X be a random selection, with X_D the destinations.

The metric Y for the relationship anonymity of u and d in C is:

$$Y(C) = \Pr[X_D(u)=d \mid X \approx C]$$

Probabilistic Anonymity

Let C be the selection of routers and destinations.

Let X be a random selection, with X_D the destinations.

The metric Y for the relationship anonymity of u and d in C is:

$$Y(C) = \Pr[X_D(u)=d \mid X \approx C]$$

Note: There are several other candidates for a probabilistic anonymity metric, *e.g.* **entropy**

Probabilistic Anonymity

Let C be the selection of routers and destinations.

Let X be a random selection, with X_D the destinations.

The metric Y for the relationship anonymity of u and d in C is:

$$Y(C) = \Pr[X_D(u)=d \mid X \approx C]$$

Relationship anonymity given that u visits d:

$$\mathbf{E}[Y \mid X_D(u)=d]$$

Worst-case Anonymity

Worst-case Anonymity

Let $p^u_1 \geq p^u_2 \geq p^u_{d-1} \geq p^u_{d+1} \geq \ldots \geq p^u_{\delta}$

Theorem: The maximum of $\mathbf{E}[Y \mid X_D(u)=d]$ over $(p^v)_{v \neq u}$ occurs when

1. $p^v_{\delta}=1$ for all $v \neq u$ OR
2. $p^v_{d}=1$ for all $v \neq u$

Worst-case Anonymity

Let $p^u_1 \geq p^u_2 \geq p^u_{d-1} \geq p^u_{d+1} \geq \ldots \geq p^u_\delta$

Theorem: The maximum of $\mathbf{E}[Y \mid X_D(u)=d]$ over $(p^v)_{v \neq u}$ occurs when

1. $p^v_\delta = 1$ for all $v \neq u$ OR
2. $p^v_d = 1$ for all $v \neq u$

Theorem: When $p^v_\delta = 1$ for all $v \neq u$:

$$\mathrm{E}[Y \mid X_D(u)=d] = b + b(1-b)p^u_d + (1-b)^2 p^u_d \left[(1-b)/(1-(1-p^u_\delta)b)\right) + O(\sqrt{\log n/n})]$$

$$\approx b + (1-b)\, p^u_d$$

Worst-case Anonymity

Let $p^u_1 \geq p^u_2 \geq p^u_{d-1} \geq p^u_{d+1} \geq \ldots \geq p^u_\delta$

Theorem: The maximum of $\mathbf{E}[Y \mid X_D(u)=d]$ over $(p^v)_{v \neq u}$ occurs when

1. $p^v_\delta = 1$ for all $v \neq u$ OR
2. $p^v_d = 1$ for all $v \neq u$

Theorem: When $p^v_\delta = 1$ for all $v \neq u$:

$$E[Y \mid X_D(u)=d] = b + b(1-b)p^u_d + (1-b)^2 p^u_d [(1-b)/(1-(1- p^u_\delta)b)) + O(\sqrt{\log n/n})]$$

$$\approx b + (1-b)\, p^u_d$$

$$E[Y \mid X_D(u)=d] \geq b^2 + (1-b^2)\, p^u_d$$

Typical Case

Let each user select from the Zipfian distribution:

$p_{d_i} = 1/(\mu_i^s)$

Theorem:

$\mathbf{E}[Y \mid X_D(u)=d] = b^2 + (1 - b^2)p^u{}_d + O(1/n)$

Typical Case

Let each user select from the Zipfian distribution:

$p_{d_i} = 1/(\mu_i^s)$

Theorem:

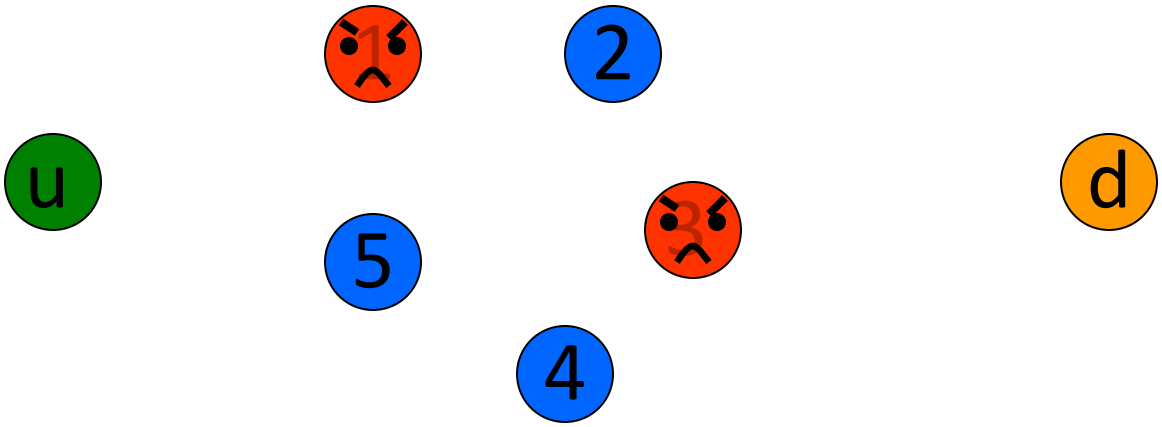
$\mathbf{E}[Y \mid X_D(u)=d] = b^2 + (1 - b^2)p^u{}_d + O(1/n)$

$\mathbf{E}[Y \mid X_D(u)=d] \geq b^2 + (1 - b^2)p^u{}_d$

Outline

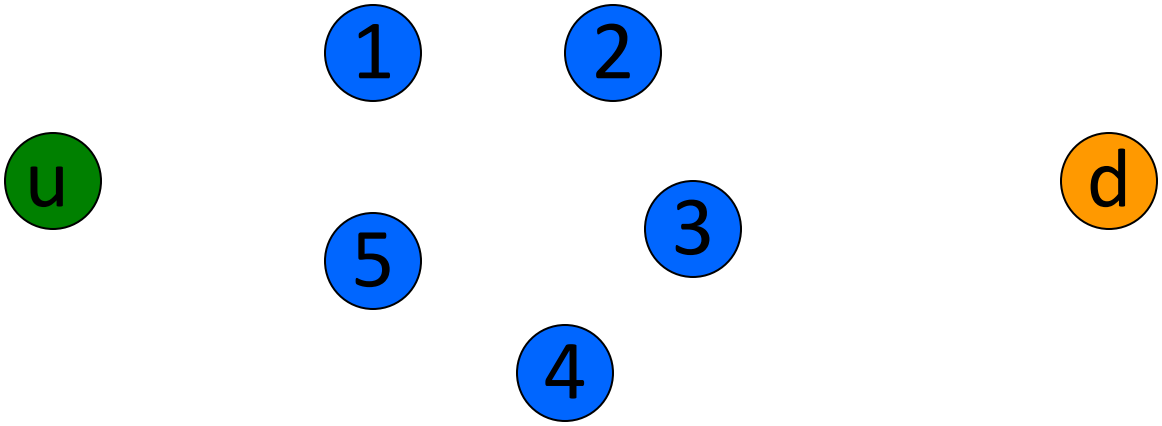
1. Introduction
2. Model & Definitions
3. Onion Routing – Possibilistic Analysis
4. Onion Routing – Probabilistic Analysis
5. Improved Anonymity Through Trust
6. Improved Anonymity by Eliminating Timing Attacks

Using Trust



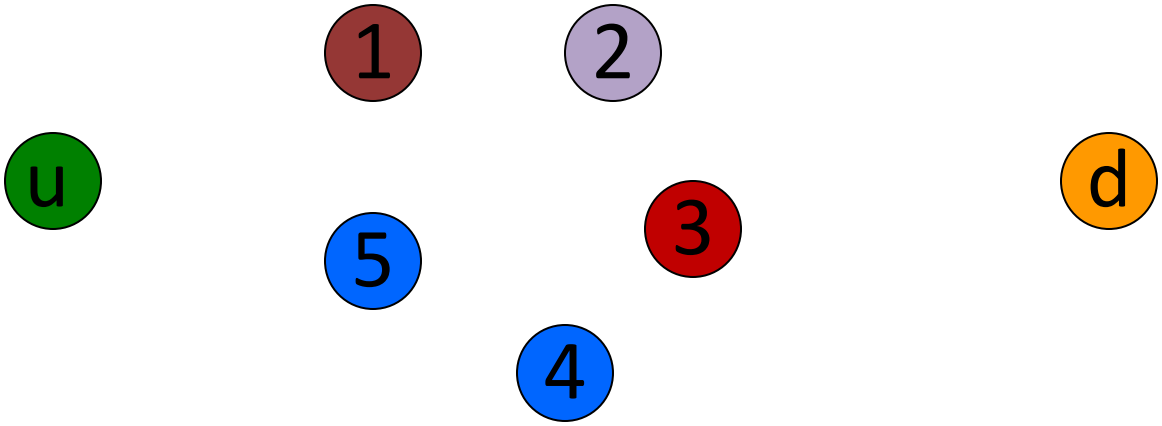
- Adversarial routers

Using Trust



- Adversarial routers
- User doesn't know where the adversary is.

Using Trust



- Adversarial routers
- User doesn't know where the adversary is.
- User may have some idea of which routers are likely to be adversarial.

Model

- Router r_i has **trust** t_i. An attempt to compromise a router succeeds with probability $c_i = 1\text{-}t_i$.
- User will choose circuits using a known distribution.
- Adversary attempts to compromise at most k routers, $K \subseteq R$.
- After attempts, users actually choose circuits.

Model

- For anonymity, just consider case that adversary controls first and last routers.
- Probability of compromise:

$$c(p,K) = \sum_{r,s \in K} p_{rs}\, c_r\, c_s$$

- **Problem:**
  - **Input:** Trust values $t_1,\ldots,t_n$
  - **Output:** Distribution p^* on router pairs such that

$$p^* \in \operatorname{argmin}_p \max_{K \subseteq R : |K| = k} c(p,K)$$

Algorithm

- Turn into a linear program
- Variables: $p_{rs} \quad \forall r,s \in R$
  t (slack variable)
- Constraints:
  - Probability distribution:
    $0 \leq p_{rs} \leq 1$
    $\Sigma_{r,s \in R} p_{rs} = 1$
  - Minimax:
    $t - c(p,K) \geq 0 \quad \forall K \subseteq R : |K| = k$
- Objective function : t

Algorithm

- Turn into a linear program
- Variables: $p_{rs} \quad \forall r,s \in R$
  t (slack variable)
- Constraints:
  - Probability distribution:
    $0 \leq p_{rs} \leq 1$
    $\Sigma_{r,s \in R} p_{rs} = 1$
  - Minimax:
    $t - c(p,K) \geq 0 \quad \forall K \subseteq R : |K| = k$
- Objective function : t

Problem: Exponential-size linear program

Independent-Choice Approximation

1. Let $c(p) = \max_{K \subseteq R : |K| = k} \sum_{r \in K} p_r \, c_r$.
2. Choose routers independently using

$$p^* \in \operatorname{argmin}_p c(p)$$

Independent-Choice Approximation

1. Let $c(p) = \max_{K \subseteq R : |K| = k} \sum_{r \in K} p_r \, c_r$.
2. Choose routers independently using

$$p^* \in \operatorname{argmin}_p c(p)$$

Let $\mu = \operatorname{argmin}_i c_i$.

Let $p^1(r_\mu) = 1$.

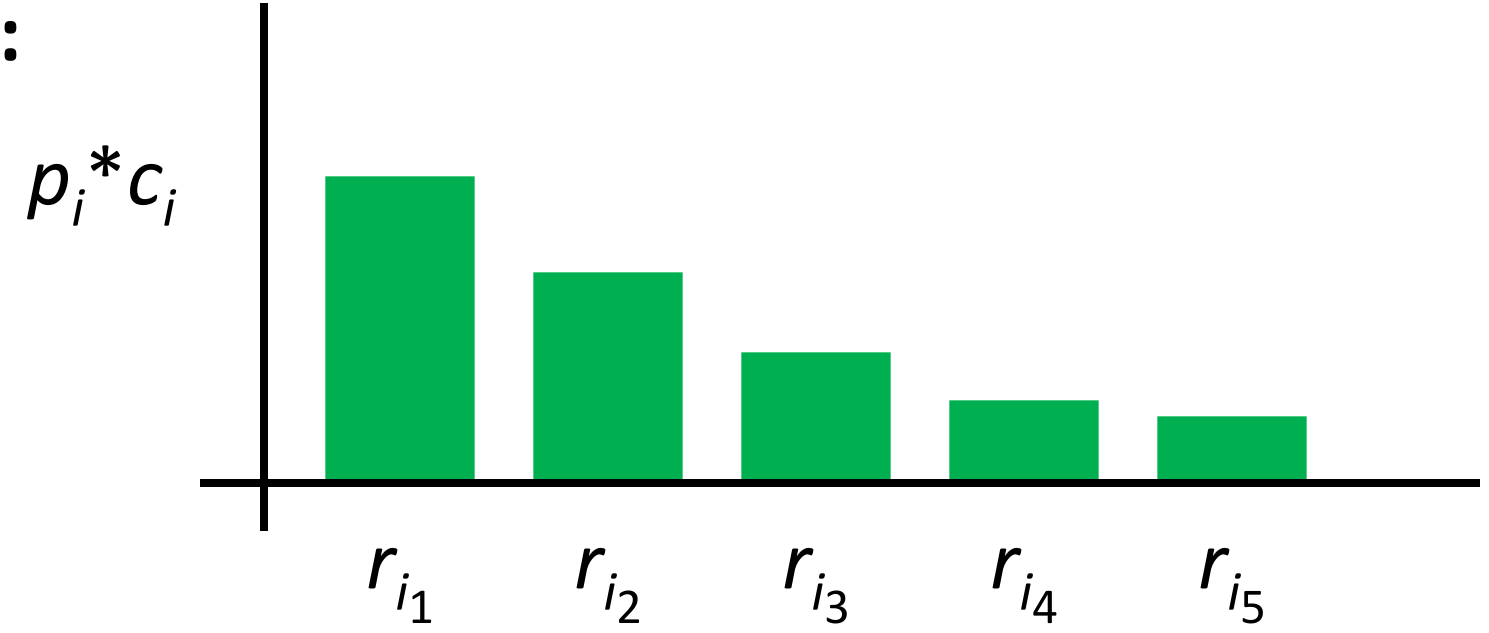
Let $p^2(r_i) = \alpha / c_i$, where $\alpha = (\sum_i 1/c_i)^{-1}$.

Theorem:

$$c(p^*) = \begin{cases} c(p^1) \text{ if } c_\mu \leq k\alpha \\ c(p^2) \text{ otherwise} \end{cases}$$

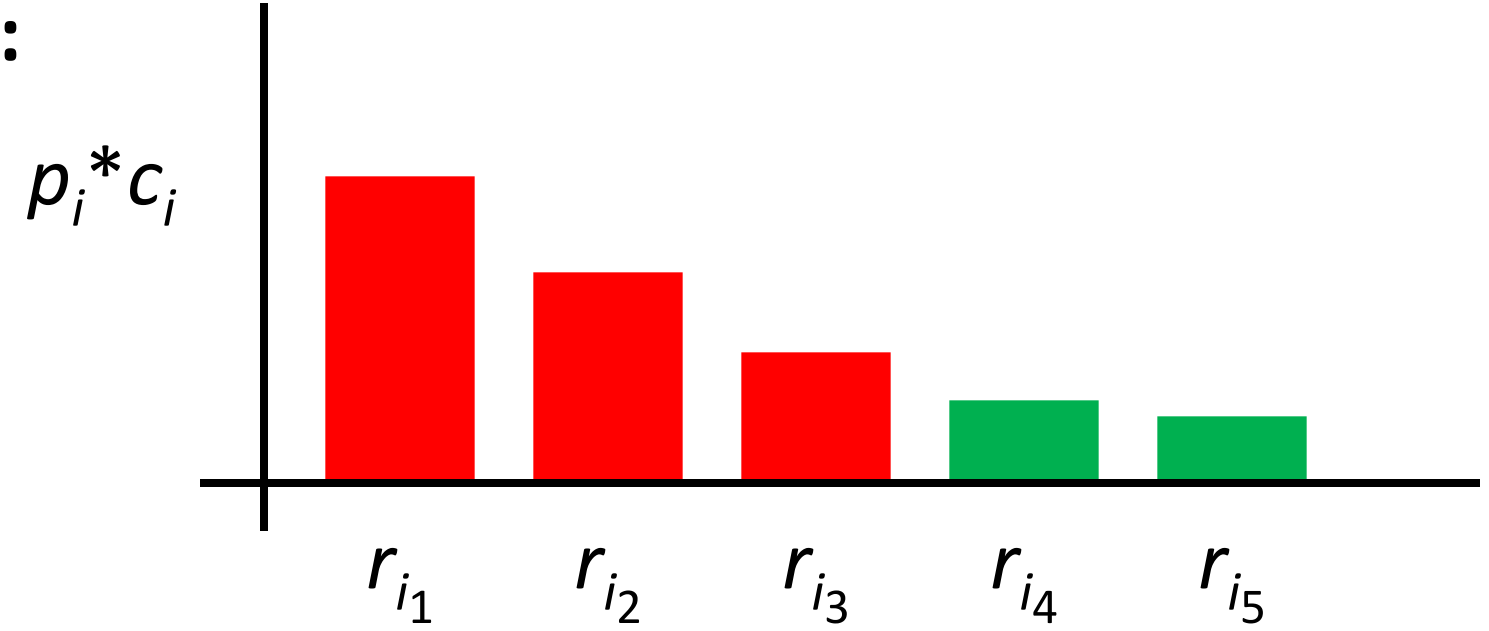
Independent-Choice Approximation

Proof:



Independent-Choice Approximation

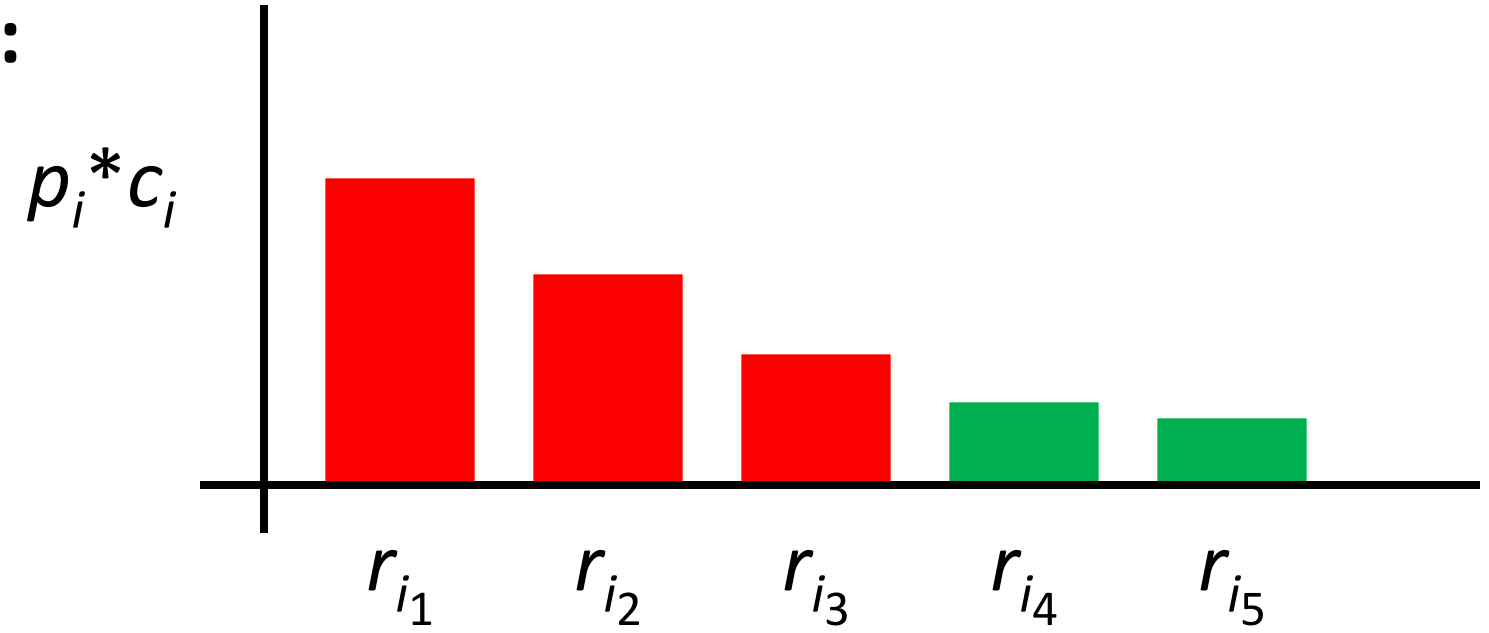
Proof:



1. Adversary chooses k routers with largest $p_i c_i$.

Independent-Choice Approximation

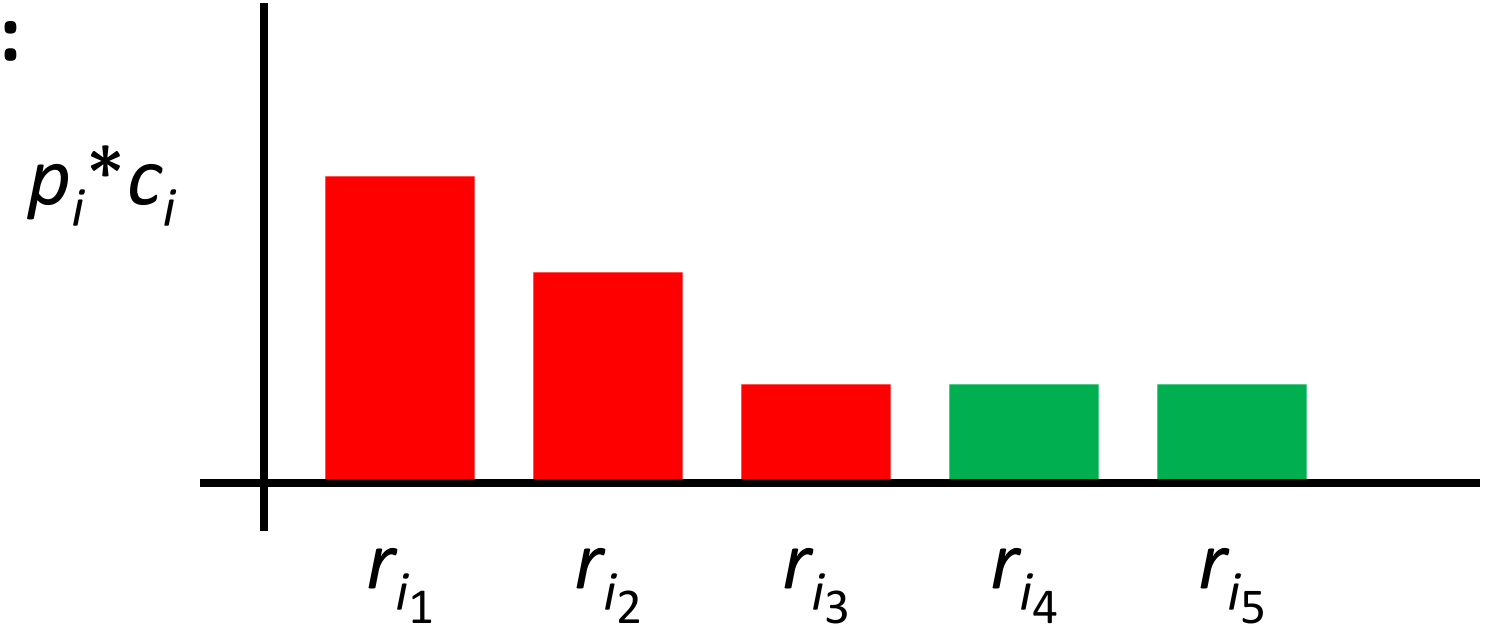
Proof:



1. Adversary chooses k routers with largest $p_i c_i$.
2. $c_{i_j} \leq c_{i_{j+1}}$ or swapping would be an improvement.

Independent-Choice Approximation

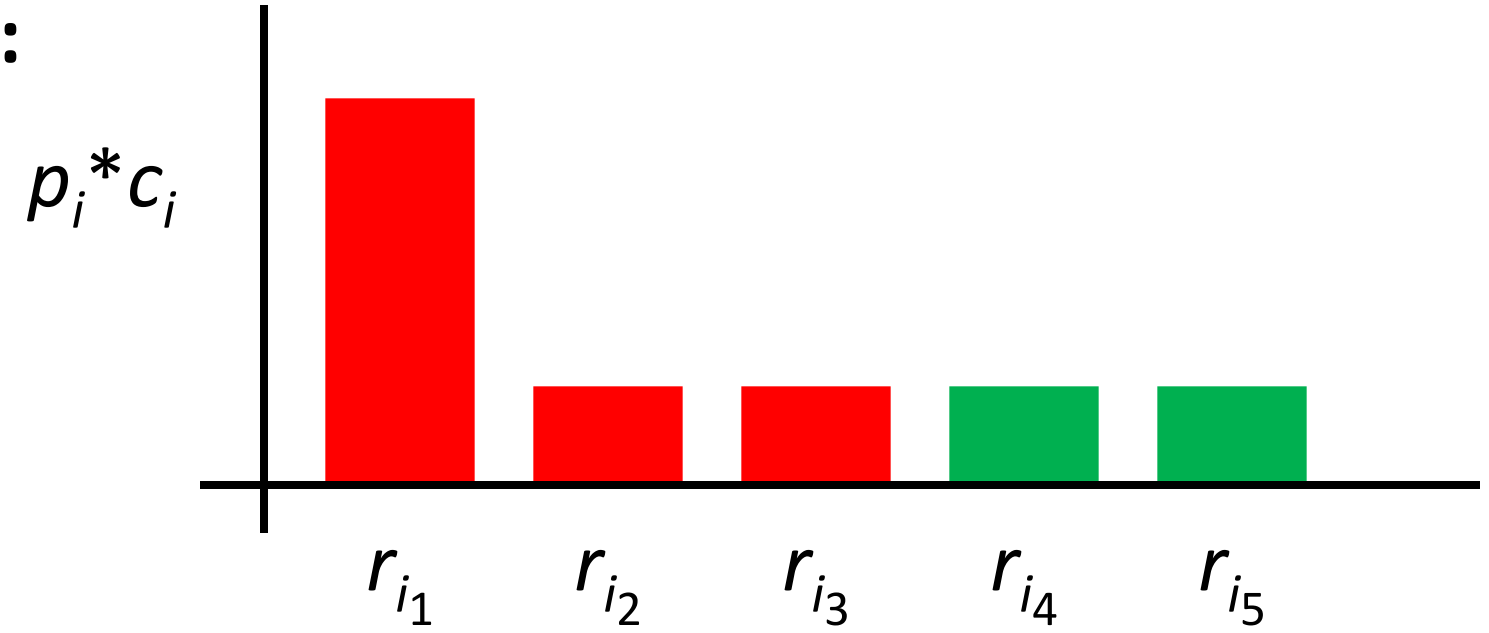
Proof:



1. Adversary chooses k routers with largest $p_i c_i$.
2. $c_{i_j} \leq c_{i_{j+1}}$ or swapping would be an improvement.
3. Can assume that $p_i\, c_i = p_j c_{j;}\ i,j >= k$.

Independent-Choice Approximation

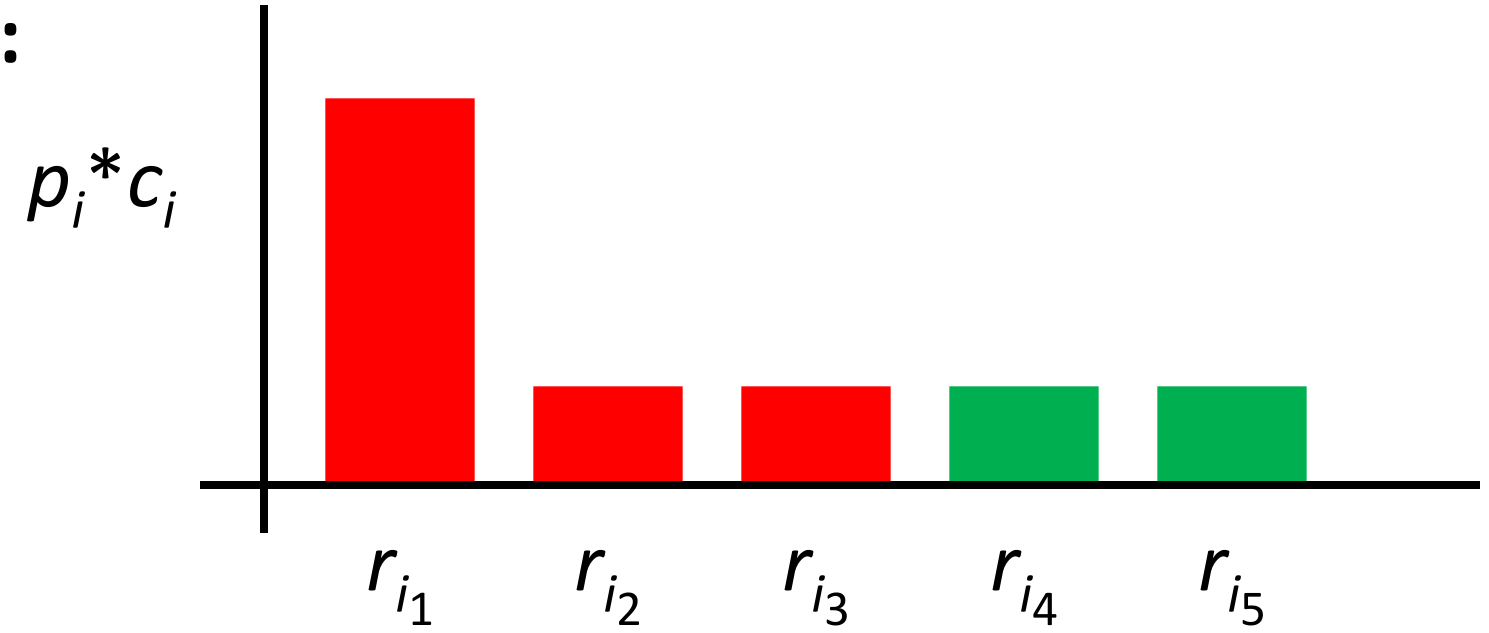
Proof:



1. Adversary chooses k routers with largest $p_i c_i$.
2. $c_{i_j} \leq c_{i_{j+1}}$ or swapping would be an improvement.
3. Can assume that $p_i\, c_i = p_j c_j$; $i,j >= k$.
4. Can assume that $p_i\, c_i = p_j c_j$; $i,j >= 2$.

Independent-Choice Approximation

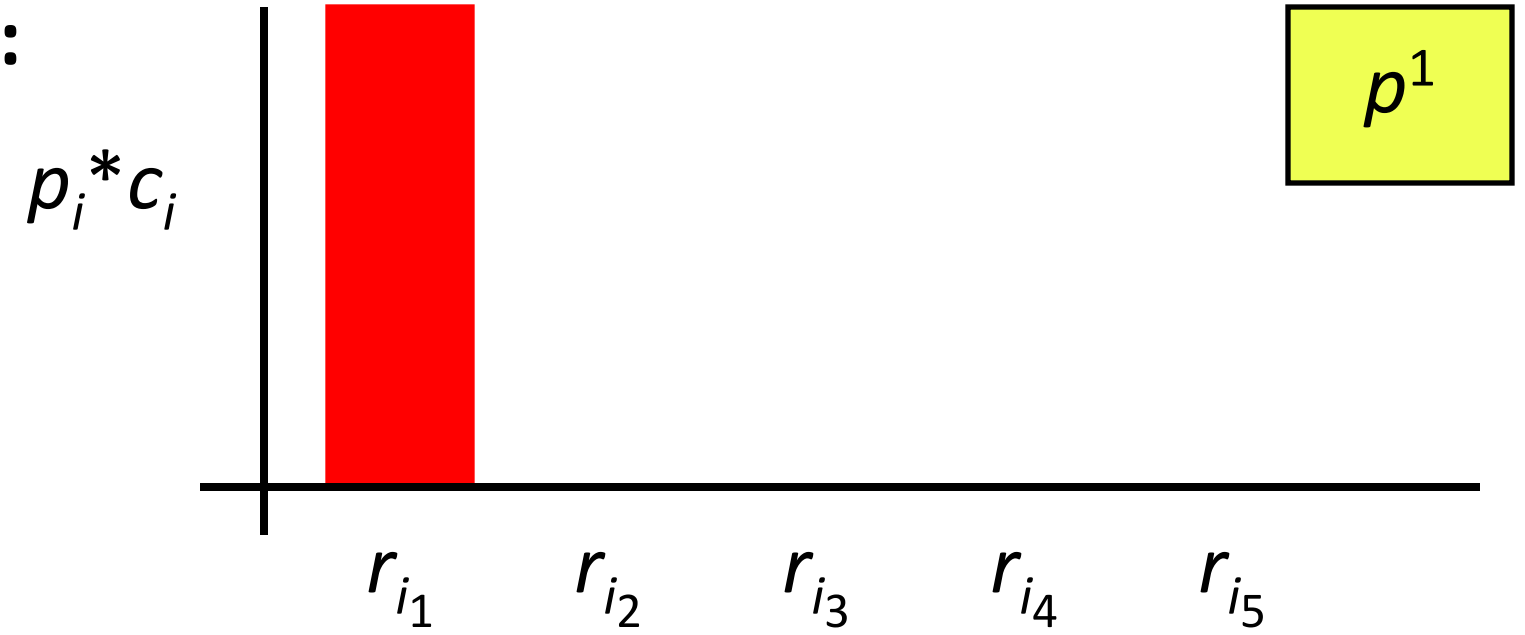
Proof:



1. Adversary chooses k routers with largest $p_i c_i$.
2. $c_{i_j} \leq c_{i_{j+1}}$ or swapping would be an improvement.
3. Can assume that $p_i\, c_i = p_j c_j$; $i,j >= k$.
4. Can assume that $p_i\, c_i = p_j c_j$; $i,j >= 2$.
5. Adjusting p_1 changes $c(p)$ linearly. Therefore one extreme is a minimum.

Independent-Choice Approximation

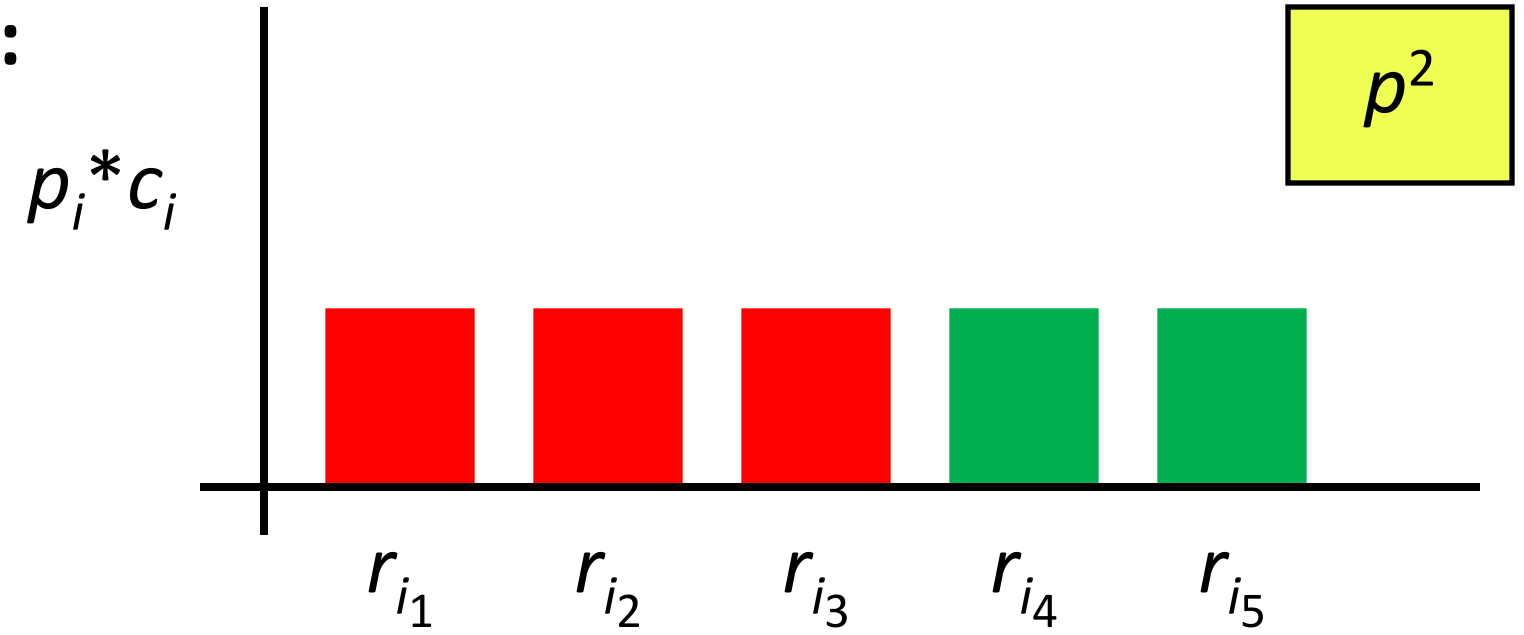
Proof:



1. Adversary chooses k routers with largest $p_i c_i$.
2. $c_{i_j} \leq c_{i_{j+1}}$ or swapping would be an improvement.
3. Can assume that $p_i\, c_i = p_j c_j$; $i,j >= k$.
4. Can assume that $p_i\, c_i = p_j c_j$; $i,j >= 2$.
5. Adjusting p_1 changes $c(p)$ linearly. Therefore one extreme is a minimum.

Independent-Choice Approximation

Proof:



1. Adversary chooses k routers with largest $p_i c_i$.
2. $c_{i_j} \leq c_{i_{j+1}}$ or swapping would be an improvement.
3. Can assume that $p_i\, c_i = p_j c_j$; $i,j >= k$.
4. Can assume that $p_i\, c_i = p_j c_j$; $i,j >= 2$.
5. Adjusting p_1 changes $c(p)$ linearly. Therefore one extreme is a minimum.

Independent-Choice Approximation

Theorem: The approximation ratio of independent selection is $\Omega(\sqrt{n})$.

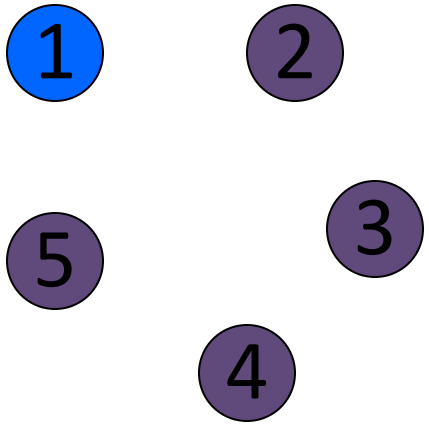
Independent-Choice Approximation

Theorem: The approximation ratio of independent selection is $\Omega(\sqrt{n})$.

Proof sketch:

Let $I_n = (c_1, \ldots, c_n, k)$ be such that

1. $c_1 = O(1/n)$
2. $c_2 > c$, $c \in (0, 1)$
3. $k = o(n)$
4. $k = \omega(1)$



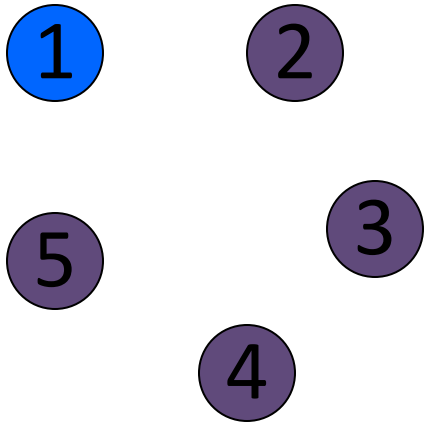
Independent-Choice Approximation

Theorem: The approximation ratio of independent selection is $\Omega(\sqrt{n})$.

Proof sketch:

Let $I_n = (c_1, \ldots, c_n, k)$ be such that

1. $c_1 = O(1/n)$
2. $c_2 > c, c \in (0, 1)$
3. $k = o(n)$
4. $k = \omega(1)$



Let $p^*(r_1, r_i) \propto 1/(c_{r_1} c_{r_i})$.

Then $c(I_n, p^1)/c(I_n, p^*) = \Omega(n/k)$

and $c(I_n, p^2)/c(I_n, p^*) = \Omega(k)$.

Independent-Choice Approximation

Theorem: The approximation ratio of independent selection is $\Omega(\sqrt{n})$.

Proof sketch:

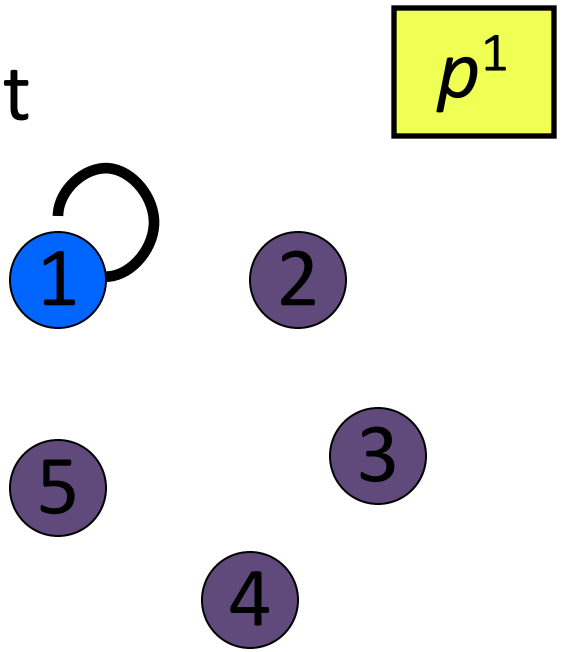
Let $I_n = (c_1, \ldots, c_n, k)$ be such that

1. $c_1 = O(1/n)$
2. $c_2 > c$, $c \in (0, 1)$
3. $k = o(n)$
4. $k = \omega(1)$

Let $p^*(r_1, r_i) \propto 1/(c_{r_1} c_{r_i})$.

Then $c(I_n, p^1)/c(I_n, p^*) = \Omega(n/k)$

and $c(I_n, p^2)/c(I_n, p^*) = \Omega(k)$.



Independent-Choice Approximation

Theorem: The approximation ratio of independent selection is $\Omega(\sqrt{n})$.

Proof sketch:

Let $I_n = (c_1, \ldots, c_n, k)$ be such that

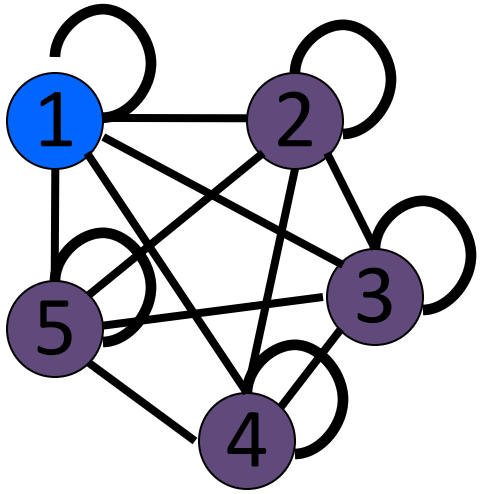
1. $c_1 = O(1/n)$
2. $c_2 > c$, $c \in (0, 1)$
3. $k = o(n)$
4. $k = \omega(1)$

Let $p^*(r_1, r_i) \propto 1/(c_{r_1} c_{r_i})$.

Then $c(I_n, p^1)/c(I_n, p^*) = \Omega(n/k)$

and $c(I_n, p^2)/c(I_n, p^*) = \Omega(k)$.

p^2



Independent-Choice Approximation

Theorem: The approximation ratio of independent selection is $\Omega(\sqrt{n})$.

Proof sketch:

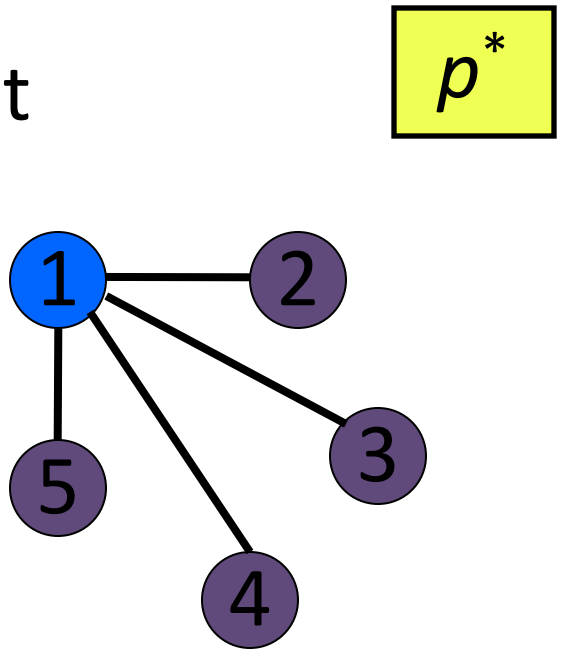
Let $I_n = (c_1, \ldots, c_n, k)$ be such that

1. $c_1 = O(1/n)$
2. $c_2 > c$, $c \in (0, 1)$
3. $k = o(n)$
4. $k = \omega(1)$

Let $p^*(r_1, r_i) \propto 1/(c_{r_1} c_{r_i})$.

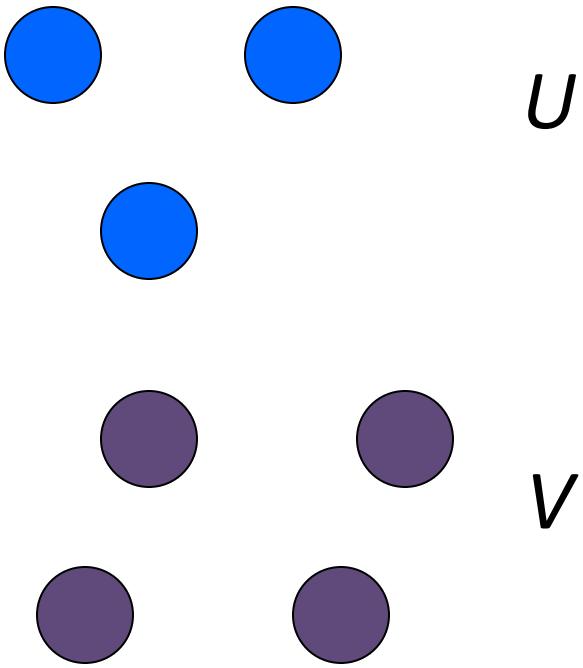
Then $c(I_n, p^1)/c(I_n, p^*) = \Omega(n/k)$

and $c(I_n, p^2)/c(I_n, p^*) = \Omega(k)$.



Trust Model

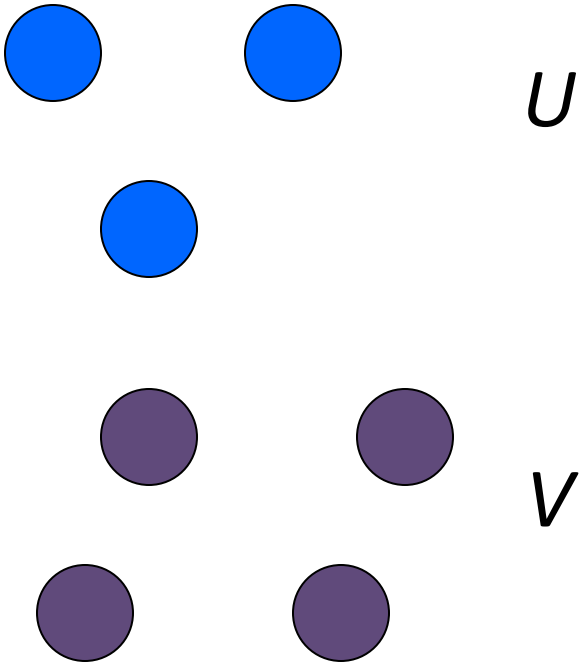
- Two trust levels: $t_1 \geq t_2$
- U = $\{r_i \mid t_i = t_1\}$, V = $\{r_i \mid t_i = t_2\}$



Trust Model

- Two trust levels: $t_1 \geq t_2$
- U = $\{r_i \mid t_i=t_1\}$, V = $\{r_i \mid t_i=t_2\}$

Theorem: Three distributions can be optimal:

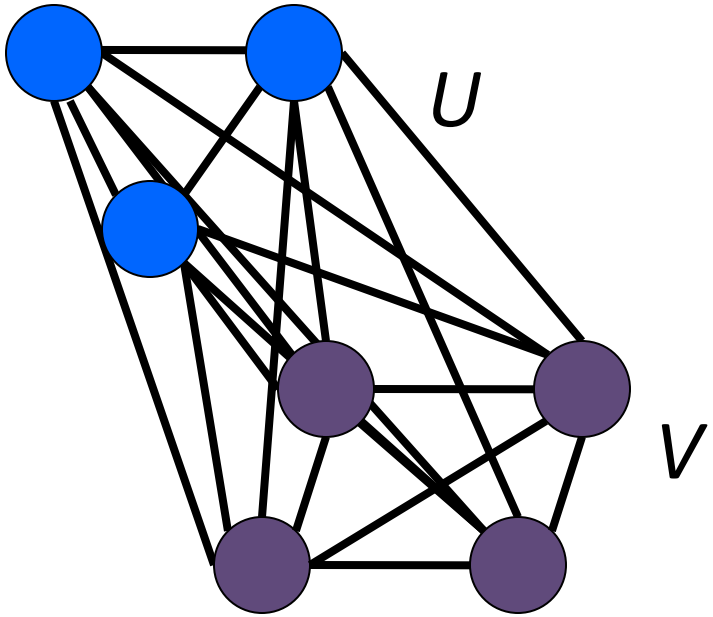


Trust Model

- Two trust levels: $t_1 \geq t_2$
- U = $\{r_i \mid t_i = t_1\}$, V = $\{r_i \mid t_i = t_2\}$

Theorem: Three distributions can be optimal:

1. $p(r,s) \propto c_r c_s$ for $r,s \in R$

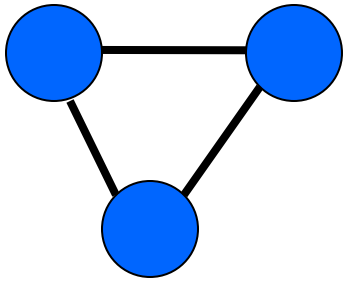


Trust Model

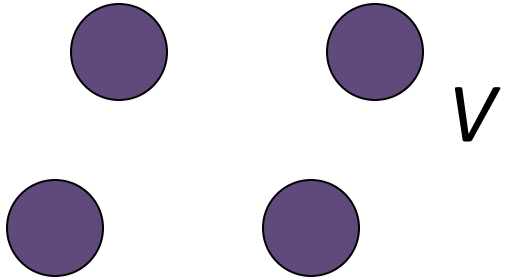
- Two trust levels: $t_1 \geq t_2$
- U = $\{r_i \mid t_i = t_1\}$, V = $\{r_i \mid t_i = t_2\}$

Theorem: Three distributions can be optimal:

1. $p(r,s) \propto c_r c_s$ for $r,s \in R$
2. $p(r,s) \propto \begin{cases} c_1^2 & \text{if } r,s \in U \\ 0 & \text{otherwise} \end{cases}$



U



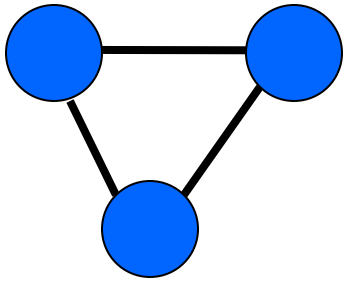
V

Trust Model

- Two trust levels: $t_1 \geq t_2$
- U = $\{r_i \mid t_i=t_1\}$, V = $\{r_i \mid t_i=t_2\}$

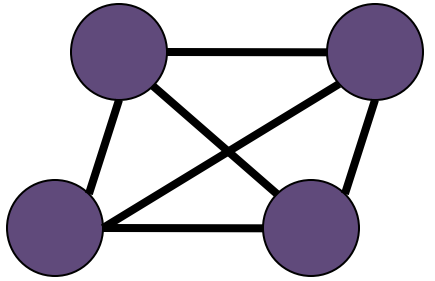
Theorem: Three distributions can be optimal:

1. $p(r,s) \propto c_r c_s$ for $r,s \in R$
2. $p(r,s) \propto \begin{cases} c_1^2 & \text{if } r,s \in U \\ 0 & \text{otherwise} \end{cases}$



U

3. $p(r,s) \propto \begin{cases} c_1^2(n(n\text{-}1)\text{-}v_0(v_0\text{-}1)) & \\ \quad \text{if } r,s \in U & \\ c_2^2(m(m\text{-}1)\text{-}v_1(v_1\text{-}1)) & \\ \quad \text{if } r,s \in V & \\ 0 \text{ otherwise} & \end{cases}$



V

where $v_0 = \max(k\text{-}m,0)$ and $v_1 = (\max(k\text{-}n,0))$

Protocol Efficiency

- Onion routing
  - Latency: *l+1*
  - Message complexity: *l*+1
- Onion routing with trust
  - Latency: *l+1*
  - Message complexity: *l*+1

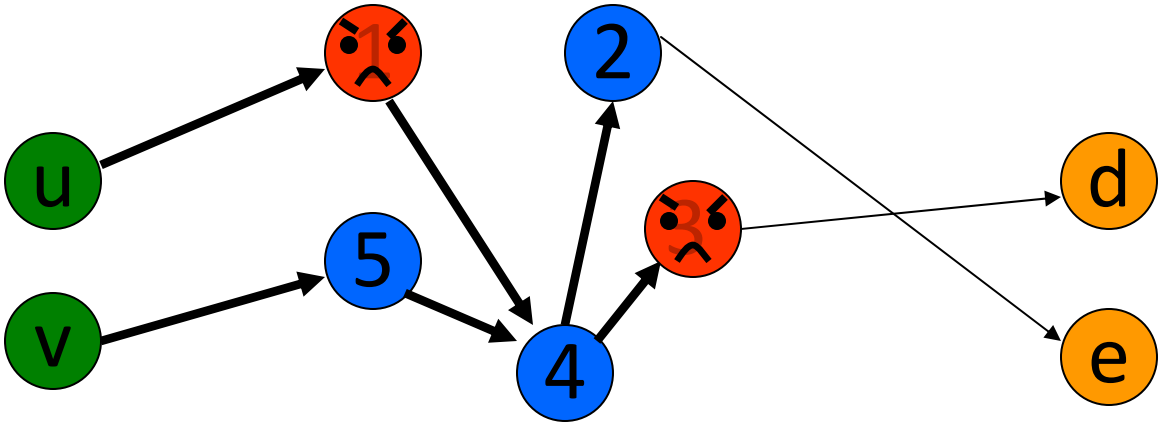
Outline

1. Introduction
2. Model & Definitions
3. Onion Routing – Possibilistic Analysis
4. Onion Routing – Probabilistic Analysis
5. Improved Anonymity Through Trust
6. Improved Anonymity by Eliminating Timing Attacks

Contributions

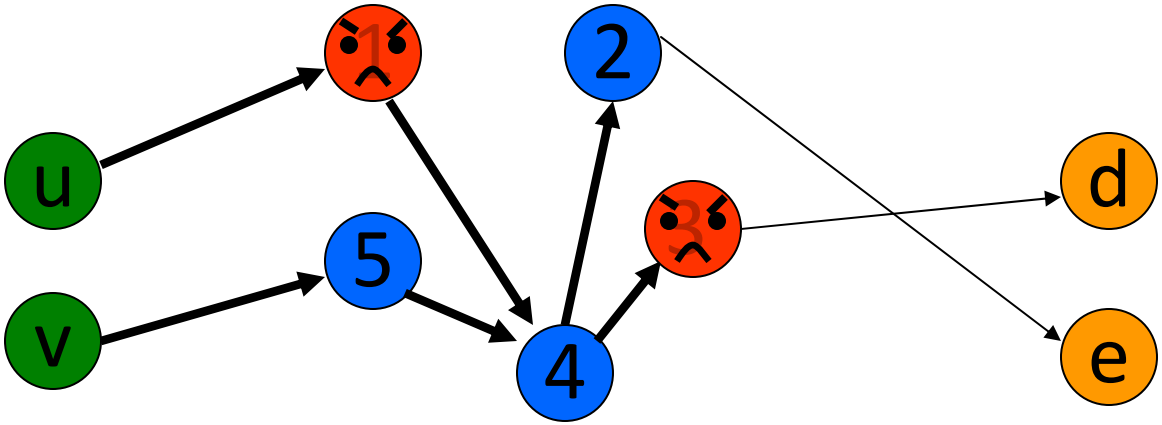
1. Give an efficient protocol that provides arbitrarily good anonymity
2. Extend the model improve time and user models
3. Perform measurements on the Tor network to evaluate performance in practice

Preventing Timing Attacks



Problem: Adversary can delay messages.

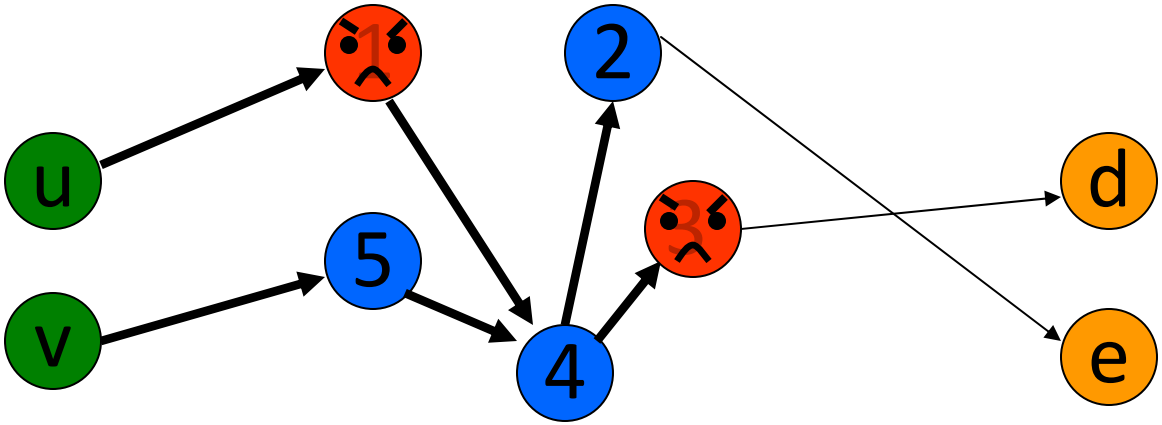
Preventing Timing Attacks



Problem: Adversary can delay messages.

Solution: Use redundancy to prevent blocked messages.

Preventing Timing Attacks

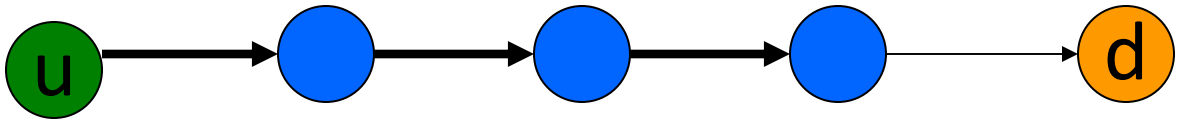


Problem: Adversary can delay messages.

Solution: Use redundancy to prevent blocked messages.

Difficulty: Balancing blocking and passive observation.

Preventing Timing Attacks

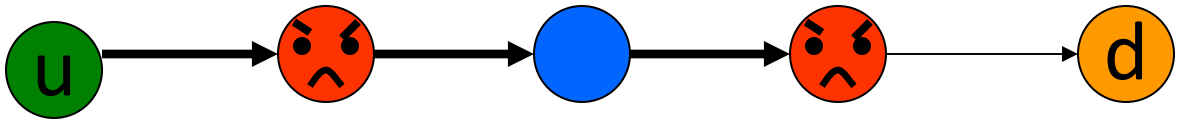


Problem: Adversary can delay messages.

Solution: Use redundancy to prevent blocked messages.

Difficulty: Balancing blocking and passive observation.

Preventing Timing Attacks

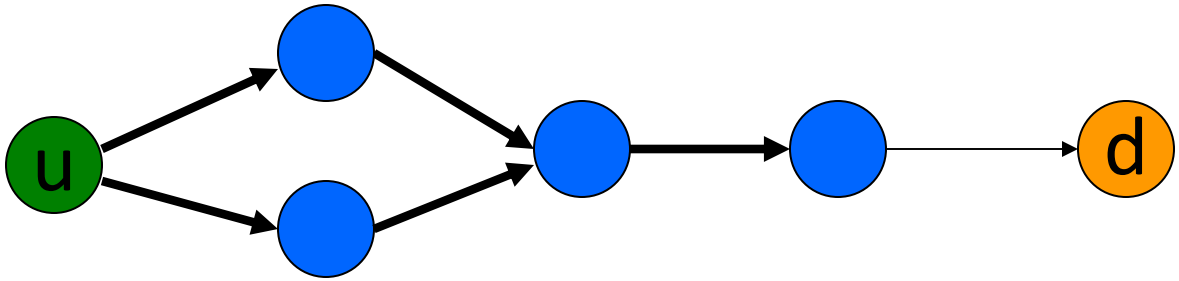


Problem: Adversary can delay messages.

Solution: Use redundancy to prevent blocked messages.

Difficulty: Balancing blocking and passive observation.

Preventing Timing Attacks

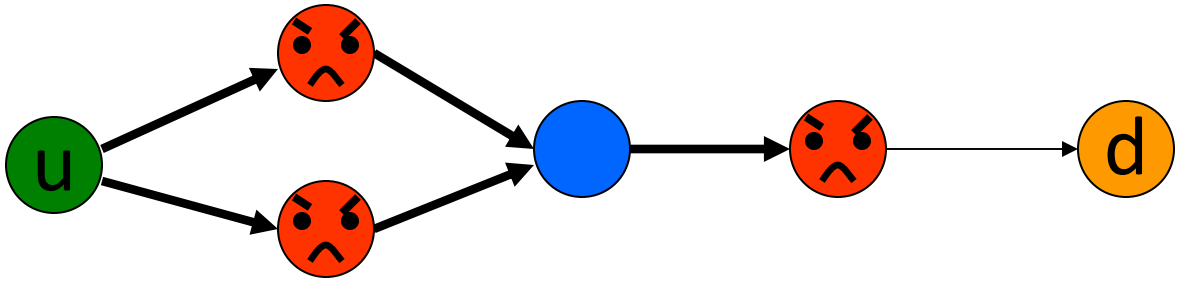


Problem: Adversary can delay messages.

Solution: Use redundancy to prevent blocked messages.

Difficulty: Balancing blocking and passive observation.

Preventing Timing Attacks

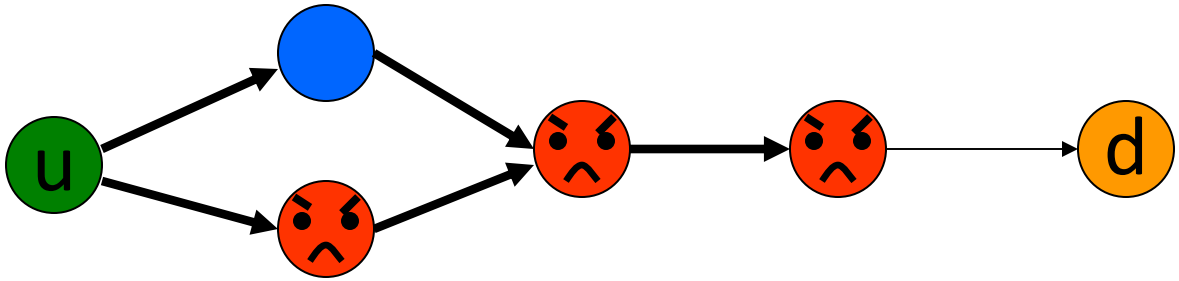


Problem: Adversary can delay messages.

Solution: Use redundancy to prevent blocked messages.

Difficulty: Balancing blocking and passive observation.

Preventing Timing Attacks

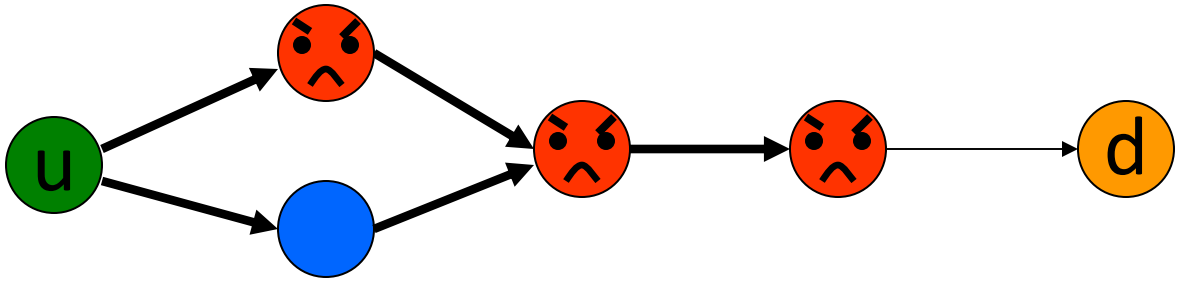


Problem: Adversary can delay messages.

Solution: Use redundancy to prevent blocked messages.

Difficulty: Balancing blocking and passive observation.

Preventing Timing Attacks

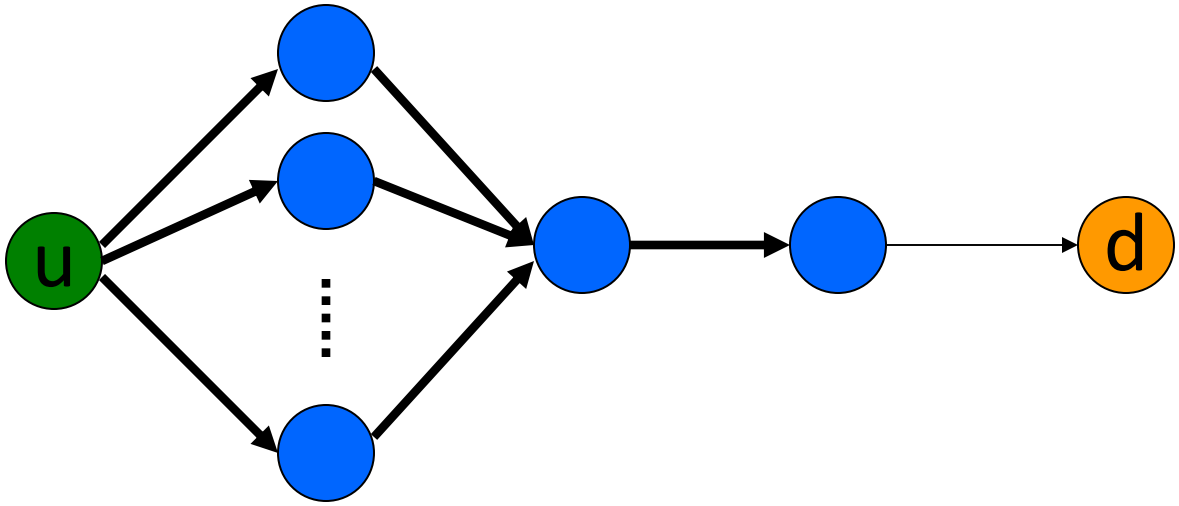


Problem: Adversary can delay messages.

Solution: Use redundancy to prevent blocked messages.

Difficulty: Balancing blocking and passive observation.

Preventing Timing Attacks

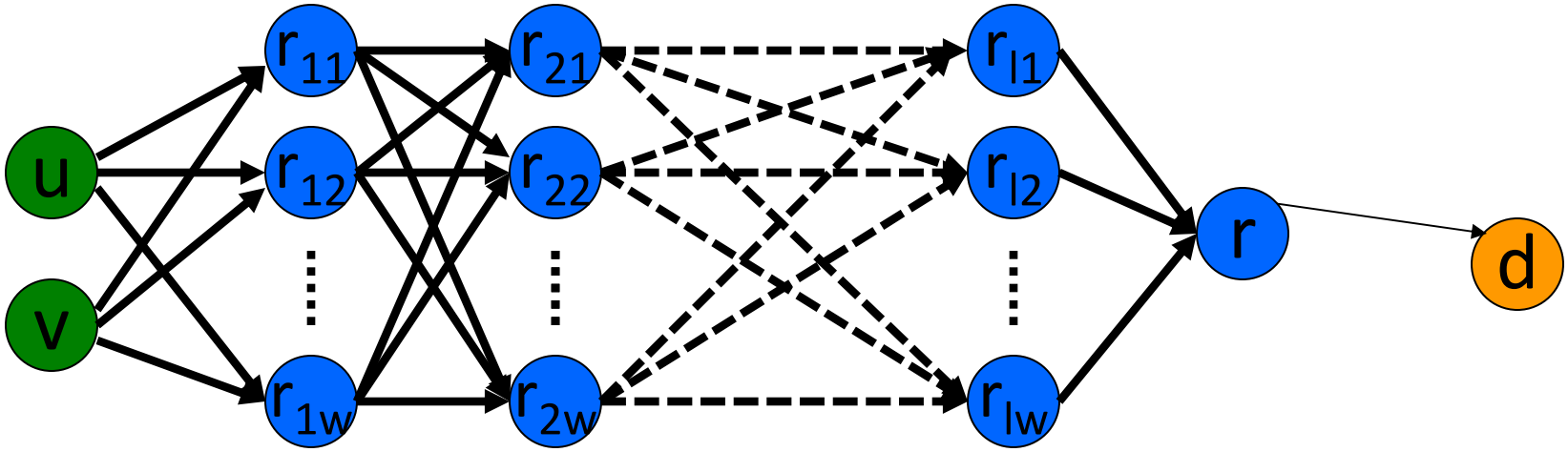


Problem: Adversary can delay messages.

Solution: Use redundancy to prevent blocked messages.

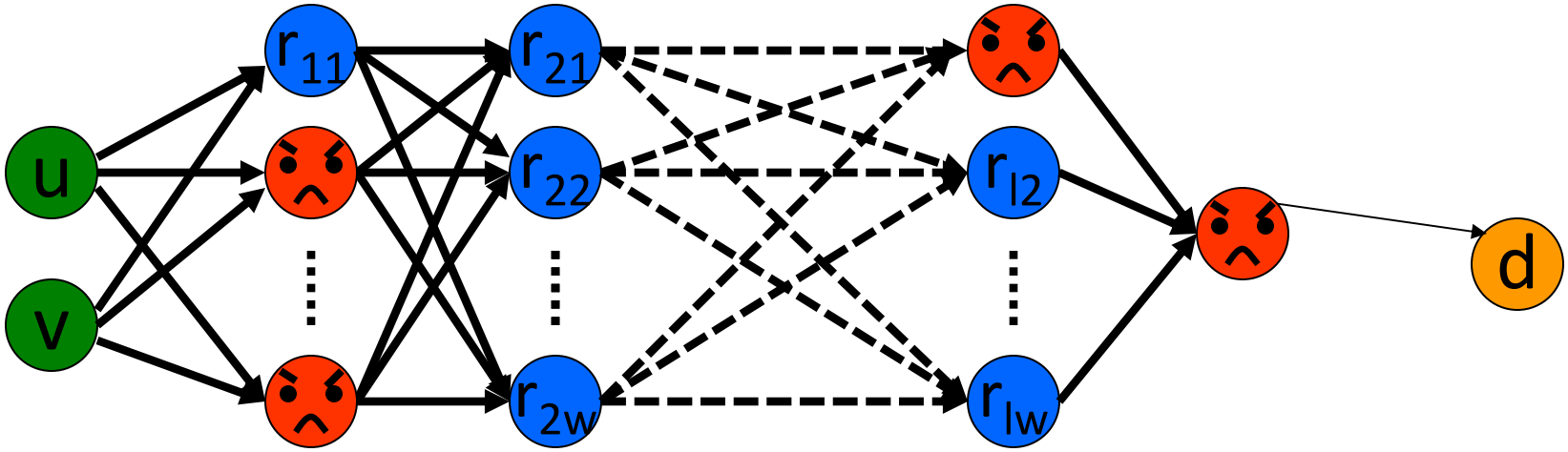
Difficulty: Balancing blocking and passive observation.

Preventing Timing Attacks



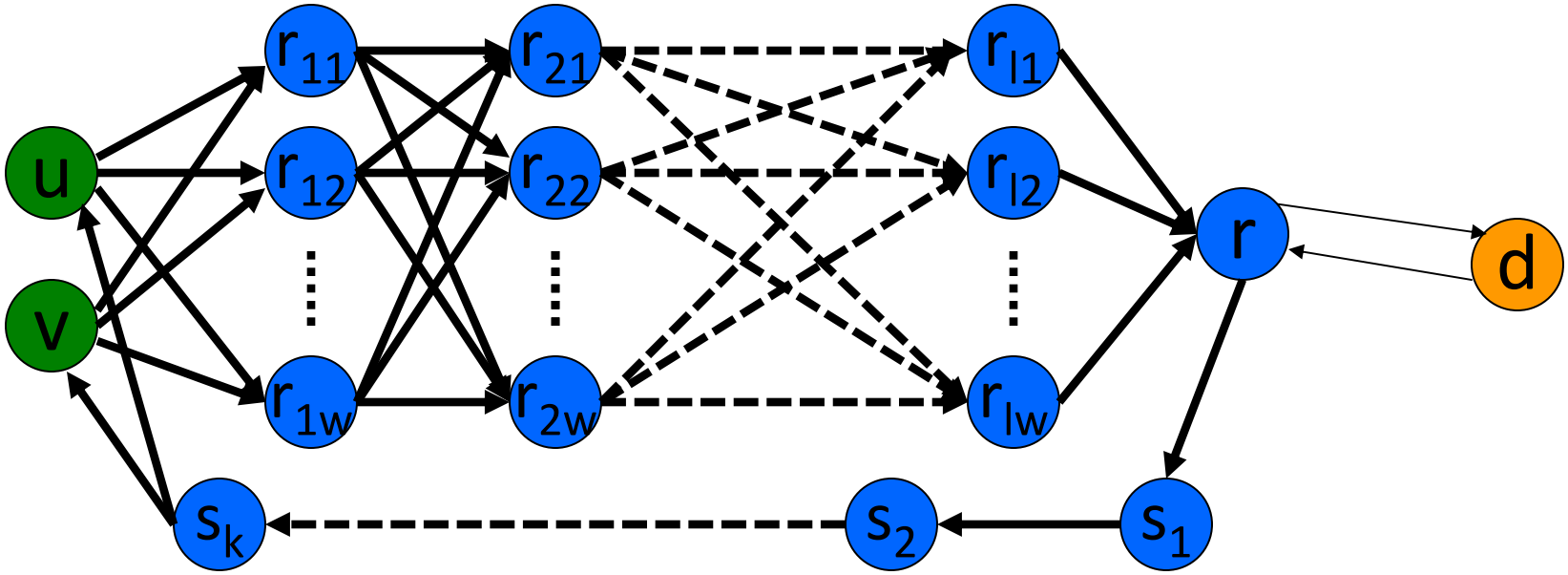
- Users send copies of each onion through a common random *layered mesh* ($w = \log l$).

Preventing Timing Attacks



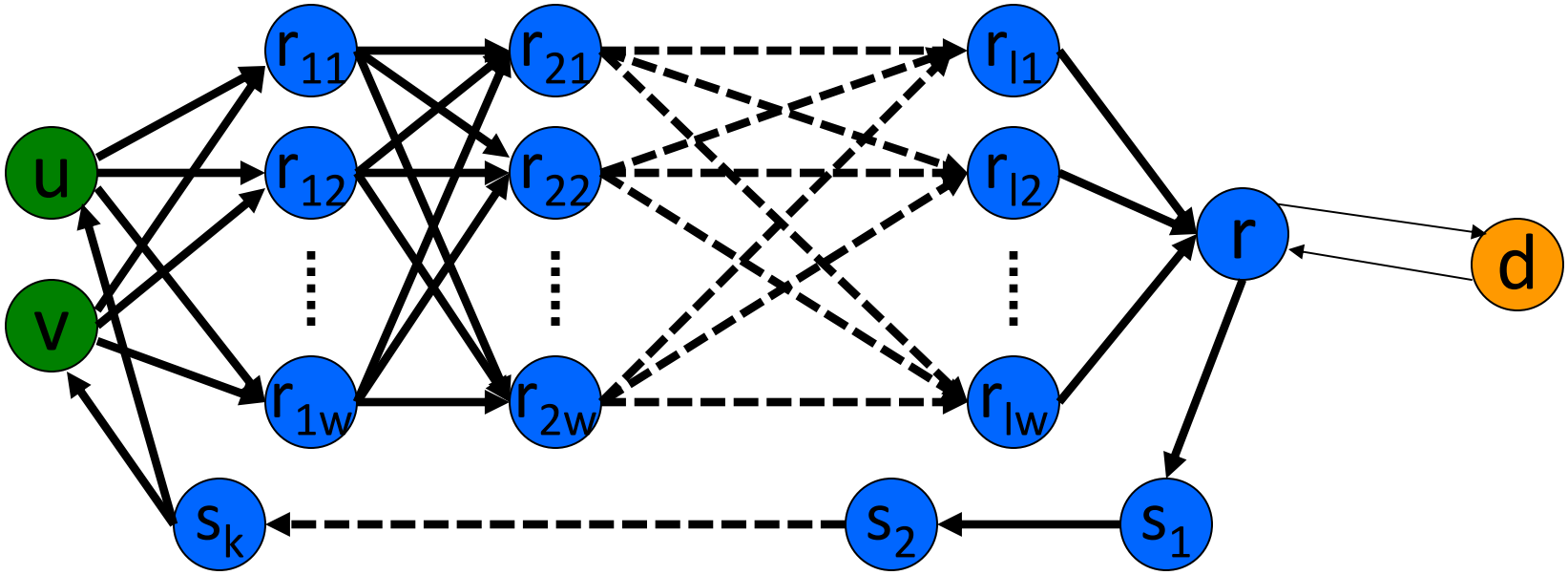
- Users send copies of each onion through a common random *layered mesh* ($w = \log l$).

Preventing Timing Attacks



- Users send copies of each onion through a common random *layered mesh* ($w = \log l$).
- Response or dummy onions are sent back on a path after a predefined delay.

Preventing Timing Attacks



Theorem: The probability of compromise p_c that the adversary can block packets is

$$p_c(b) = b^{k+1} + b(1 - b^k)\,\Pr[\mathrm{B}]$$

where B is that A first controls a layer.

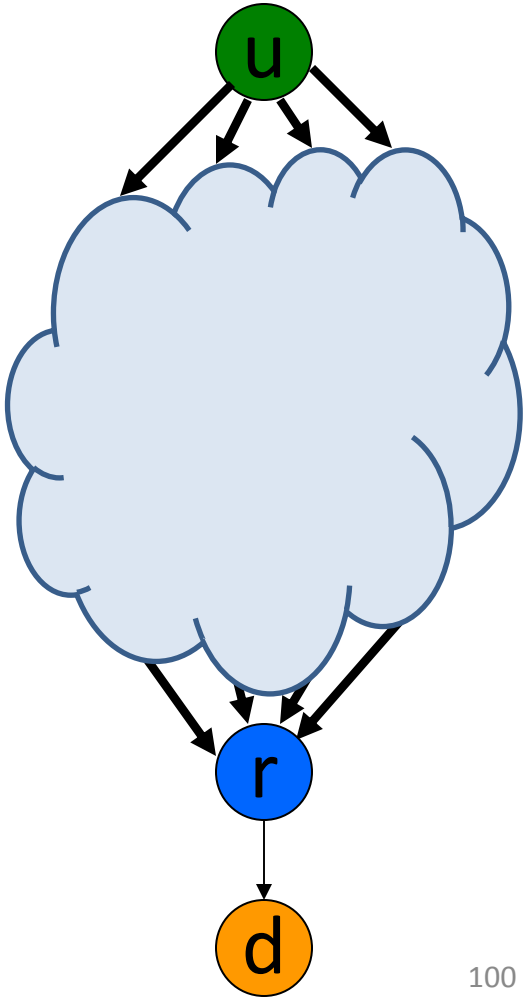
And $\lim_{l \to \infty} \Pr[\mathrm{B}] = \begin{cases} 0 & \text{for } b < ½ \\ 1 & \text{for } b > ½. \end{cases}$

Preventing Timing Attacks

Theorem: Among such forwarding schemes, lim $p_c(b)=0$ for largest b possible.

Preventing Timing Attacks

Theorem: Among such forwarding schemes, lim $p_c(b)$=0 for largest b possible.

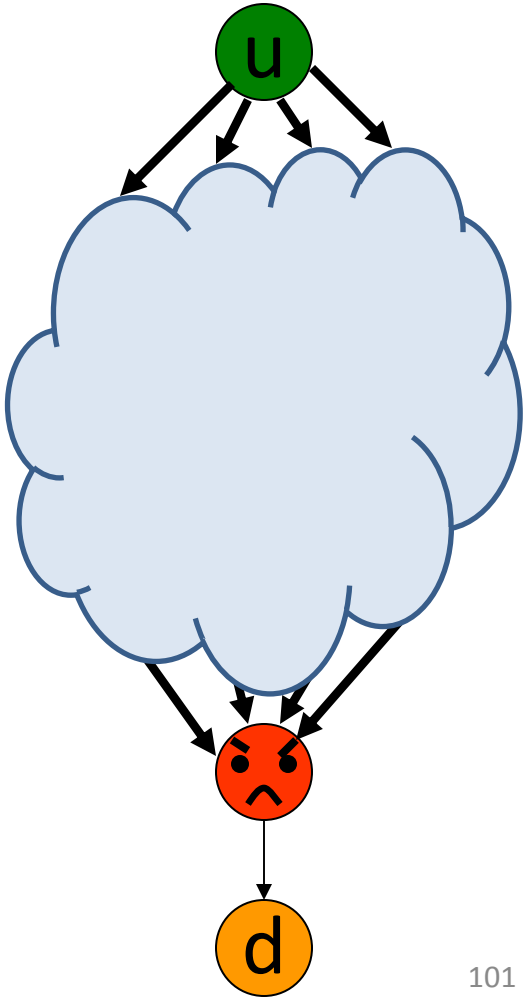


Preventing Timing Attacks

Theorem: Among such forwarding schemes, lim $p_c(b)=0$ for largest b possible.

Proof:

1.Assume the last router r is compromised.



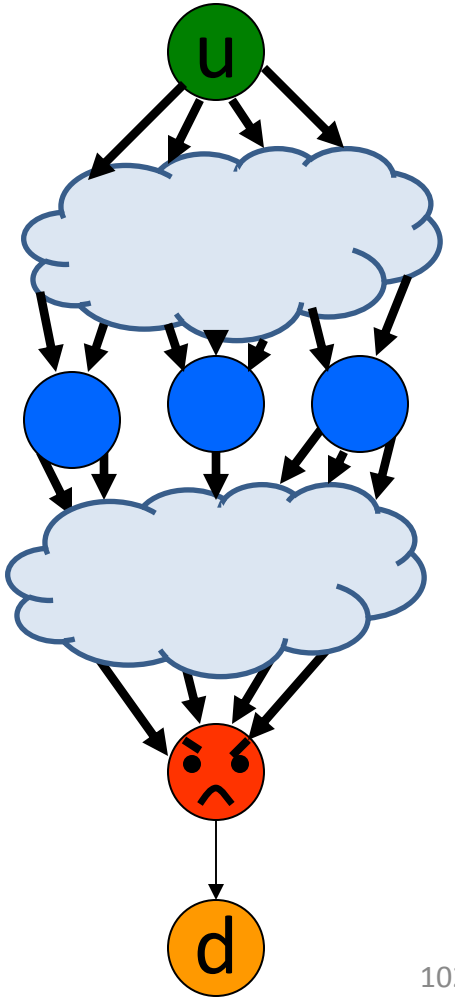
Preventing Timing Attacks

Theorem: Among such forwarding schemes, lim $p_c(b)=0$ for largest b possible.

Proof:

1.Assume the last router *r* is compromised.

2.There must be a cut of uncompromised routers in the graph between *u* and *r*.

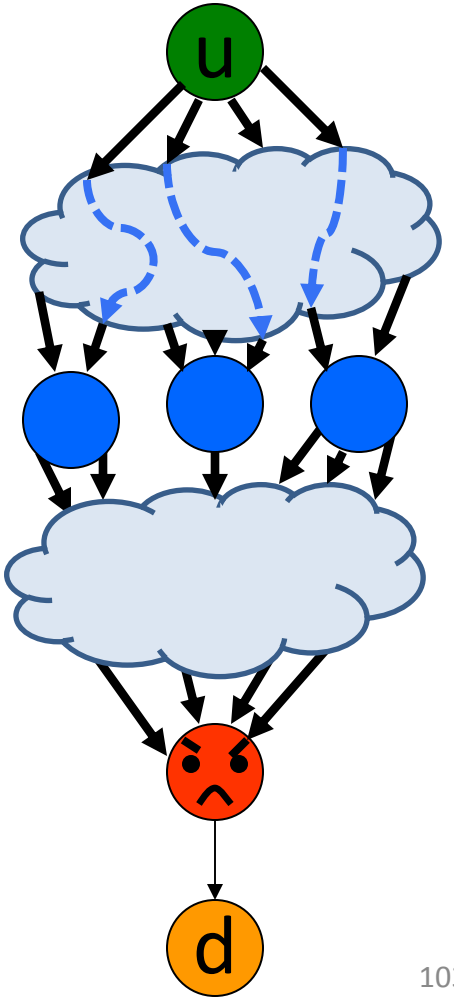


Preventing Timing Attacks

Theorem: Among such forwarding schemes, lim $p_c(b)=0$ for largest b possible.

Proof:

1. Assume the last router r is compromised.
2. There must be a cut of uncompromised routers in the graph between u and r.
3. Each router in the cut must have a path of uncompromised routers from u.



Preventing Timing Attacks

Theorem: Among such forwarding schemes, lim $p_c(b)=0$ for largest b possible.

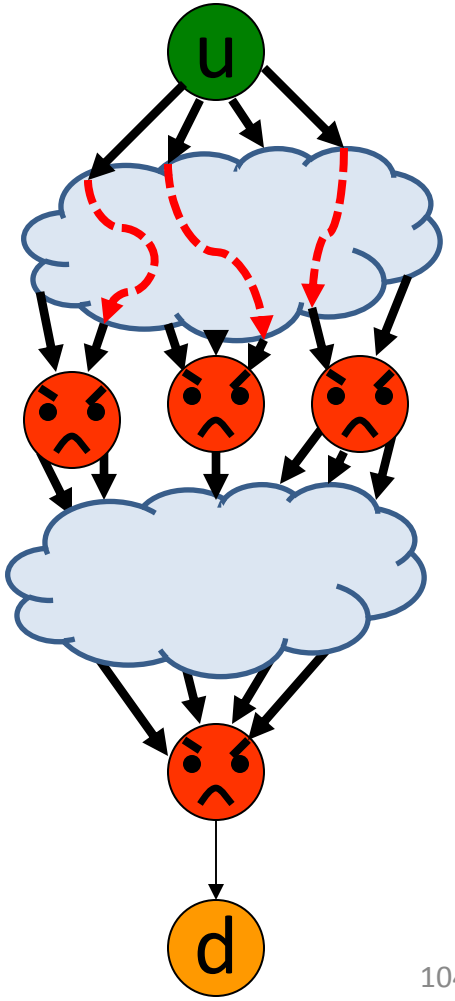
Proof:

1.Assume the last router *r* is compromised.

2.There must be a cut of uncompromised routers in the graph between *u* and *r*.

3.Each router in the cut must have a path of uncompromised routers from *u*.

4.The reverse of this situation happens with the same probability when *b*' = 1-*b*. This case does not provide anonymity.

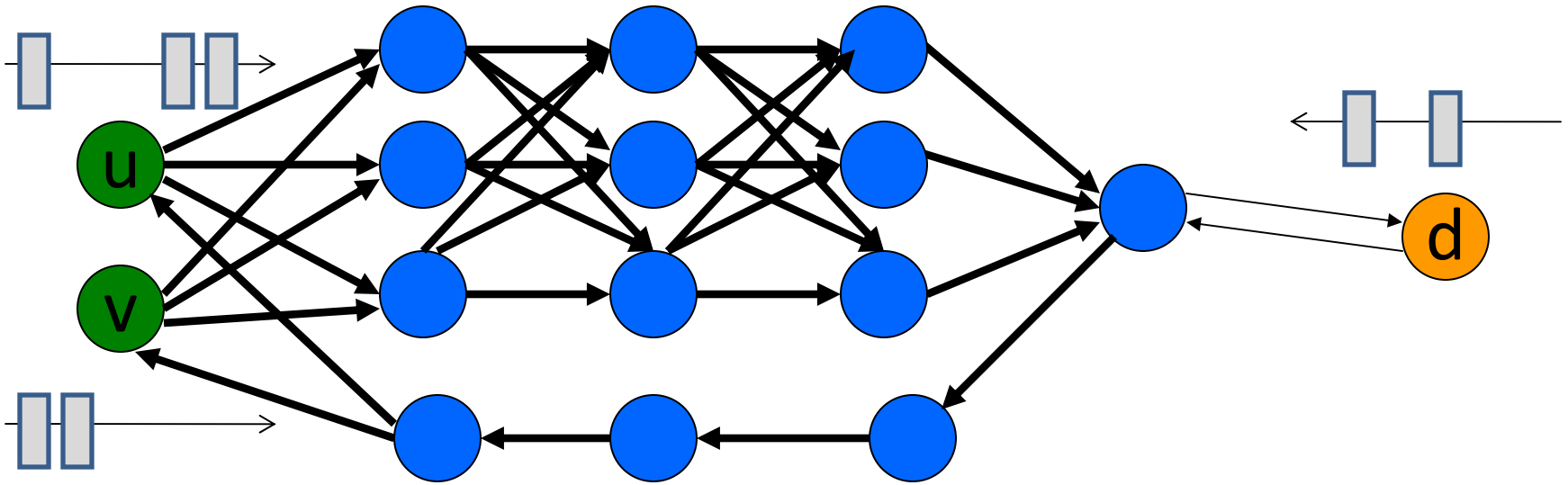


Protocol Efficiency

- Onion routing
  - Latency: $l+1$
  - Message complexity: l+1
- Onion routing with trust
  - Latency: $l+1$
  - Message complexity: l+1
- Layered mesh
  - Latency: $l+2$
  - Message complexity: $(l-1)\log^2 l + 2\log l + 1$

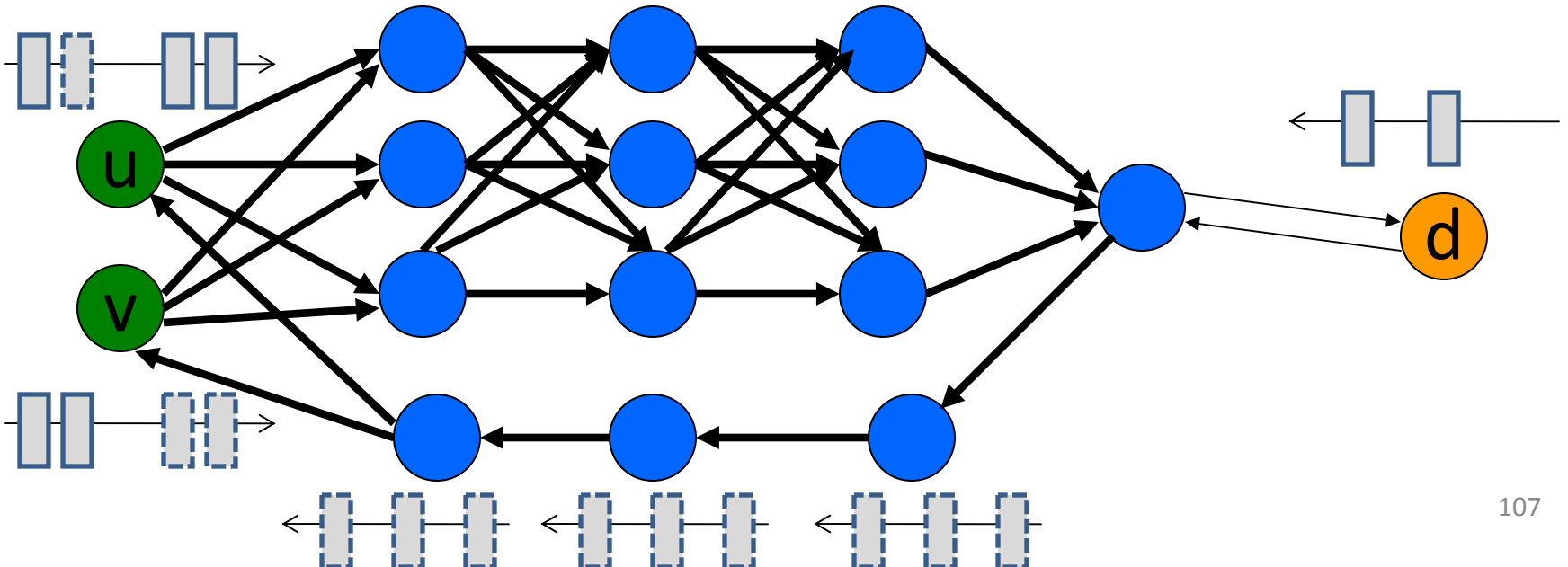
Practical Protocol

- Stream communication
  - User opens/closes stream
  - User and destination send messages on stream



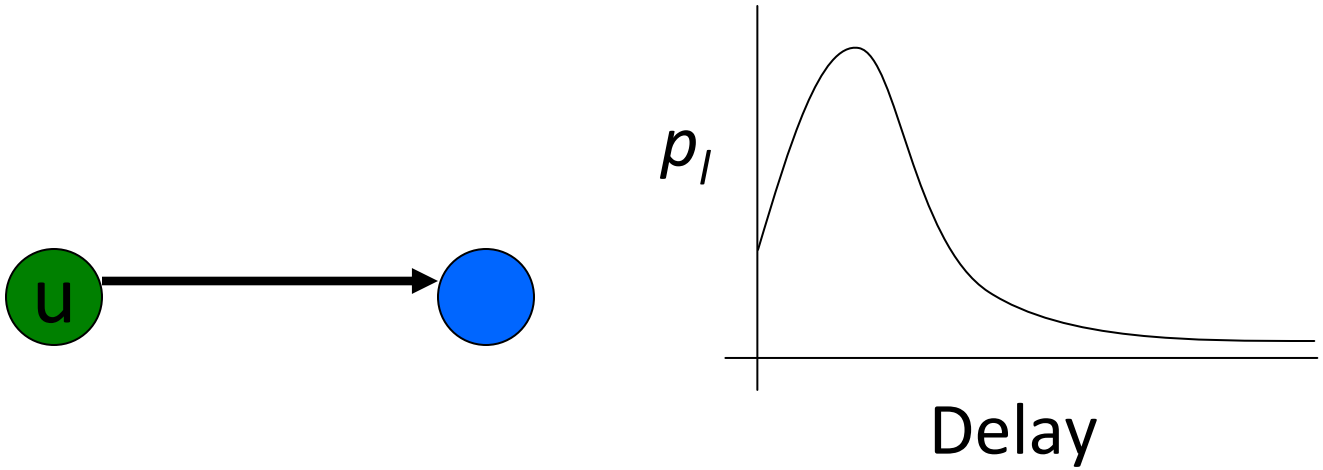
Practical Protocol

- Stream communication
  - User opens/closes stream
  - User and destination send messages on stream
- *Protocol changes*
  - Stream open/close
  - Users apply padding scheme
  - Specify padding scheme from destination



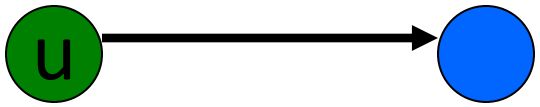
Practical Protocol

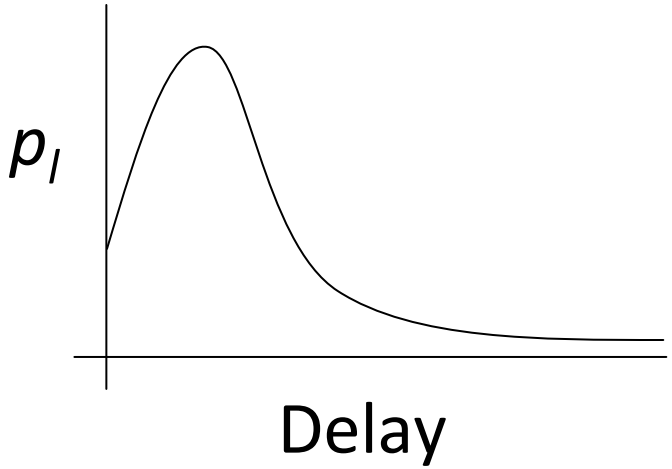
- Improved timing model
  - Each link *l* has a delay distribution p_l.



Practical Protocol

- Improved timing model
  - Each link *l* has a delay distribution p_l.
- *Protocol changes*
  - Onions include times instructing routers when to send
  - Delay $d(l)$ guarantees an arrival probability of
    $$p = \int_0^{d(l)} p_l(x)\, dx$$
- Send times in layer i are all t_i:
  $$t_i = t_{i-1} + \max d(r_{(i-1)j}, r_{i,k})$$

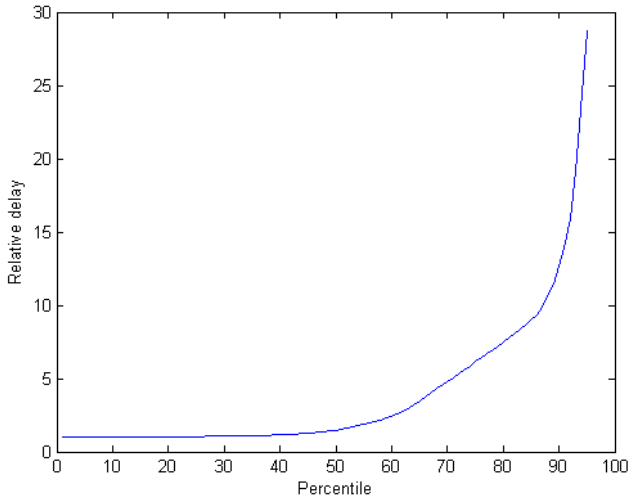




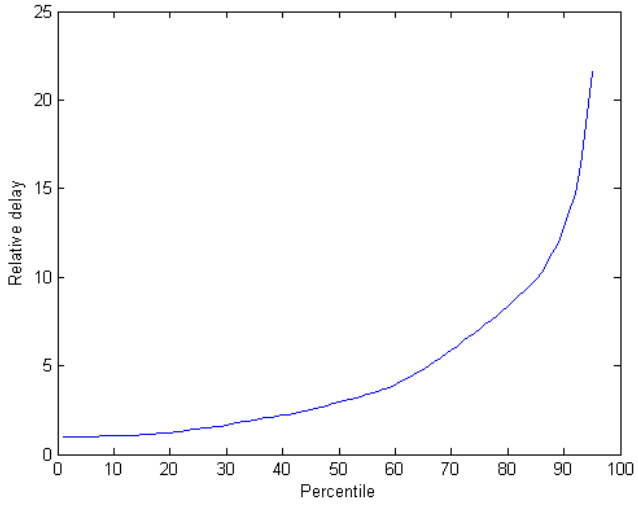
Network Measurements

- Measured delays in Tor
  - 2/22 – 3/21/09
  - 745 appropriate exit routers
  - Circuit connection times
  - Data forwarding times

Added connection delays



Added forwarding delays



Future Work

- Model additions
  - User connections over time
  - Stream functionality
  - Modeling congestion
- Show that padding schemes are feasible in practice.
- Understand using feasible anonymity primitives as building blocks.