A Formal Analysis of Onion Routing

10/26/2007

Aaron Johnson (Yale)

with

Joan Feigenbaum (Yale)

Paul Syverson (NRL)

Papers

1. ***A Model of Onion Routing with Provable Anonymity***
   Financial Cryptography and Data Security 2007
2. ***A Probabilistic Analysis of Onion Routing in a Black-box Model***
   Workshop on Privacy in the Electronic Society 2007

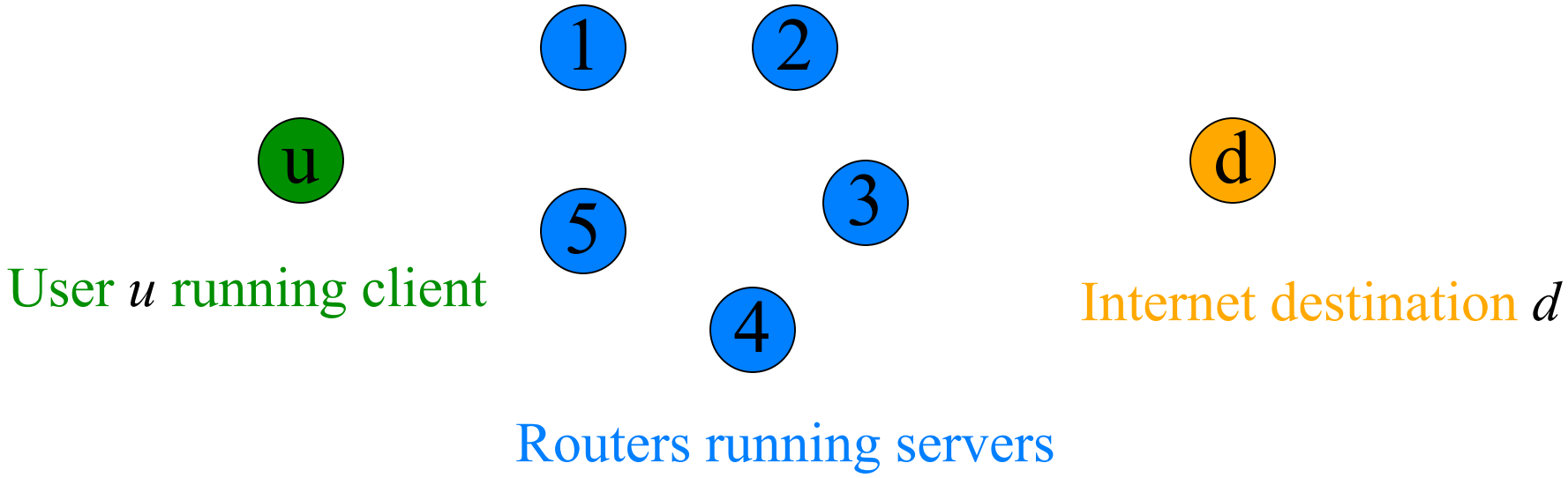
Anonymous Communication

- *Sender anonymity*: Adversary can't determine the sender of a given message
- *Receiver anonymity*: Adversary can't determine the receiver of a given message
- *Relationship anonymity*: Adversary can't determine who talks to whom

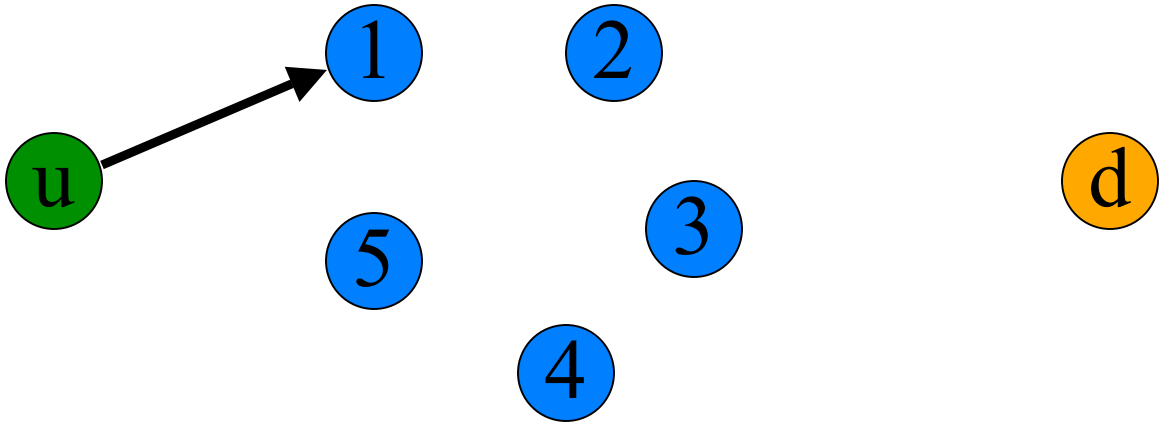
Anonymous Communication

- *Sender anonymity*: Adversary can't determine the sender of a given message
- *Receiver anonymity*: Adversary can't determine the receiver of a given message
- *Relationship anonymity*: Adversary can't determine who talks to whom

How Onion Routing Works

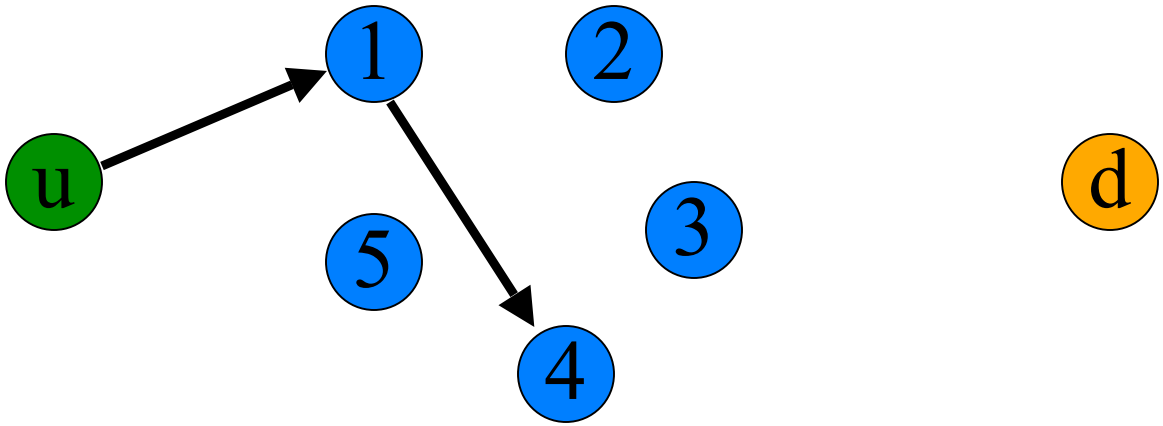


How Onion Routing Works



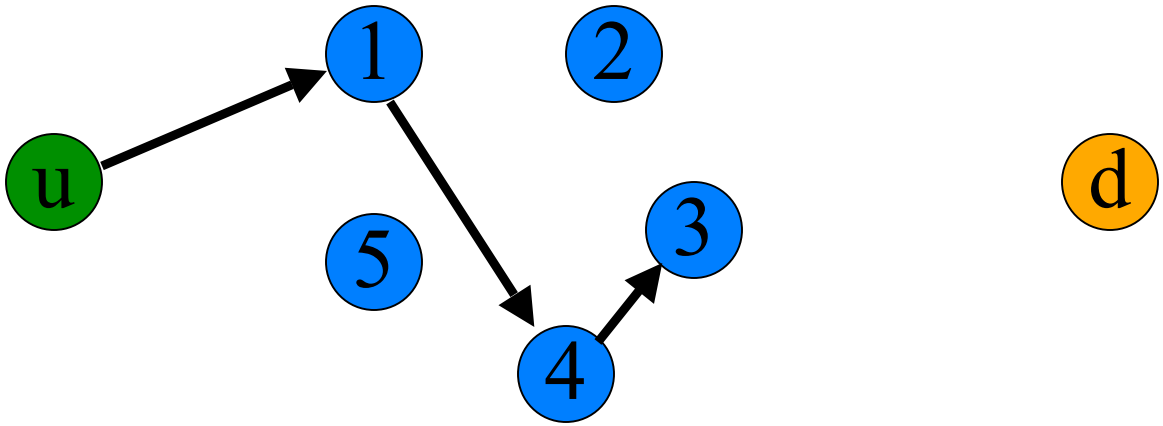
1. *u* creates 3-hop circuit through routers

How Onion Routing Works



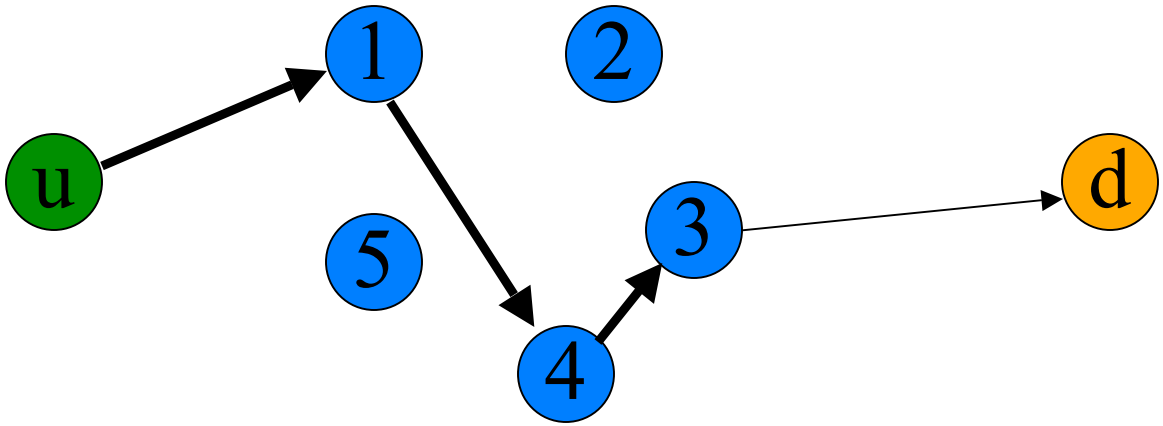
1. *u* creates 3-hop circuit through routers

How Onion Routing Works



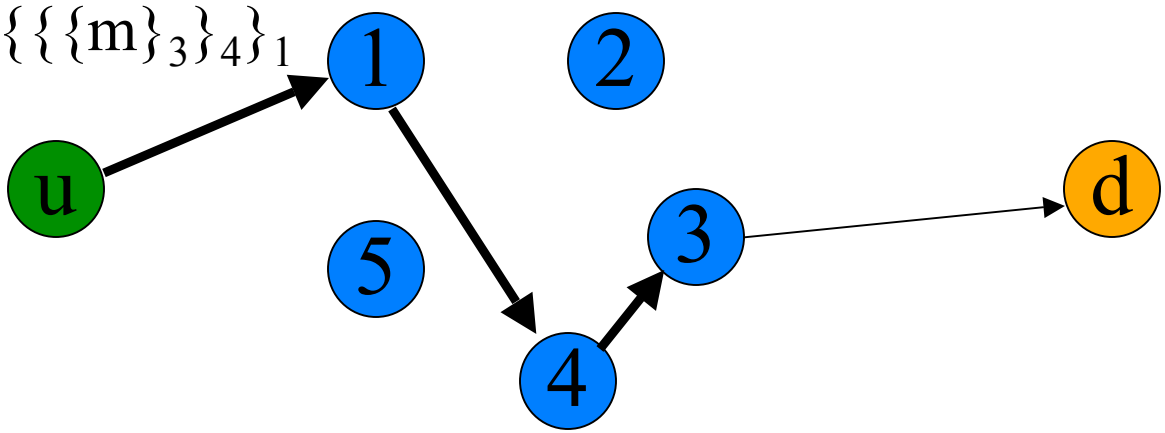
1. *u* creates 3-hop circuit through routers

How Onion Routing Works



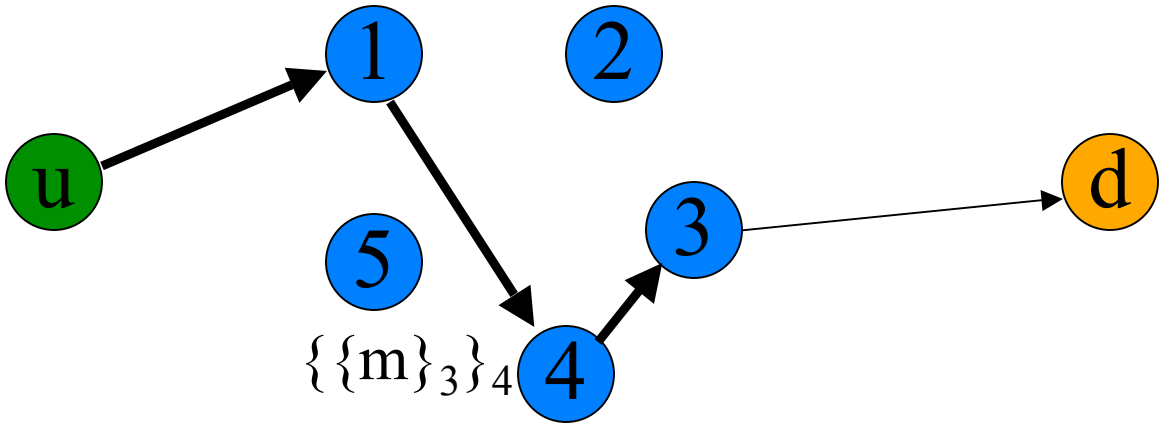
1. *u* creates 3-hop circuit through routers
2. *u* opens a stream in the circuit to *d*

How Onion Routing Works



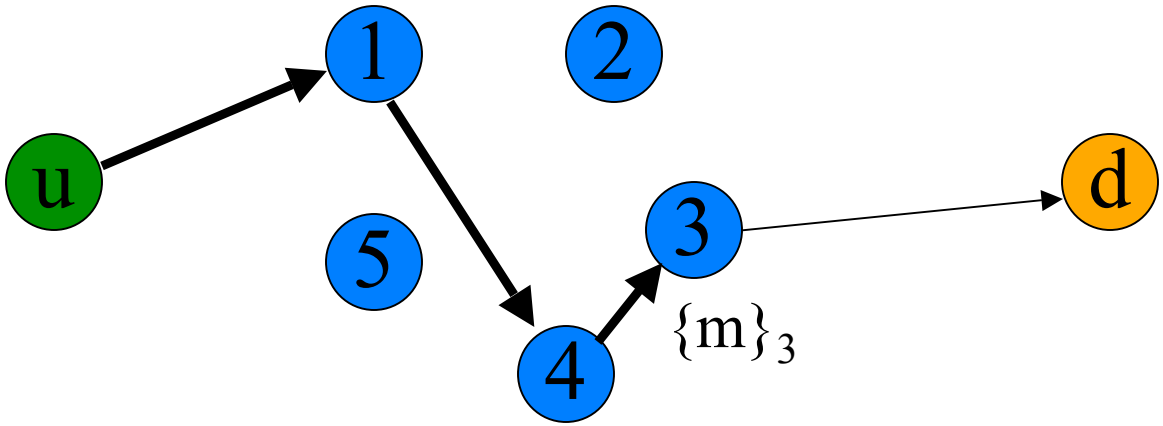
1. *u* creates 3-hop circuit through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

How Onion Routing Works



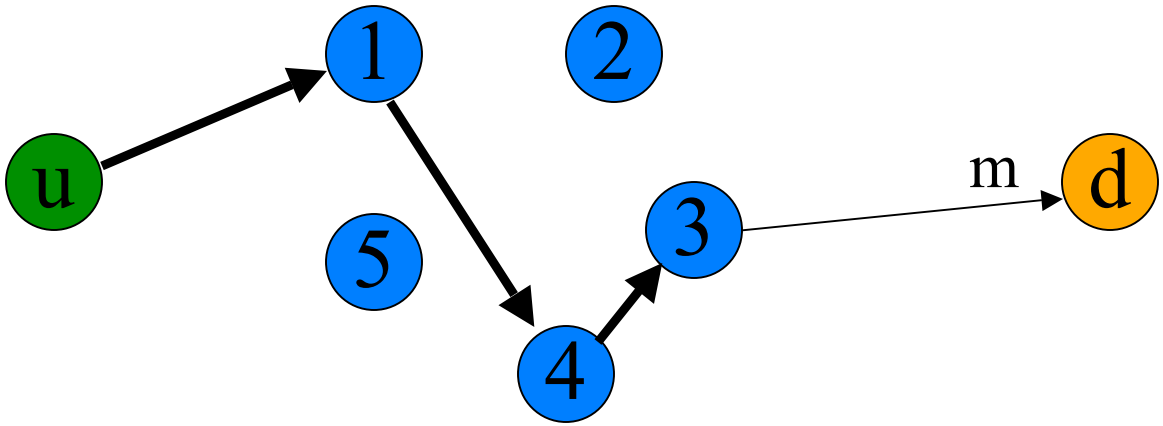
1. *u* creates 3-hop circuit through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

How Onion Routing Works



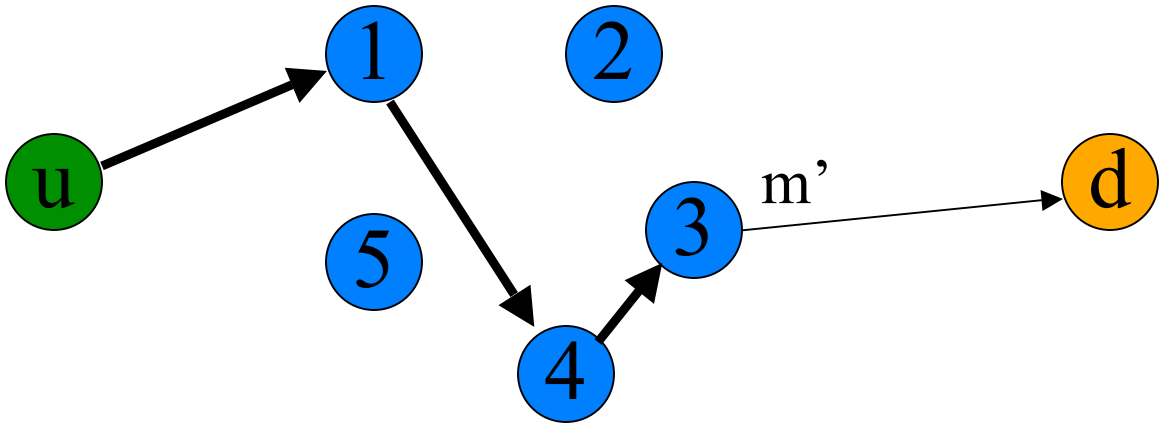
1. *u* creates 3-hop circuit through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

How Onion Routing Works



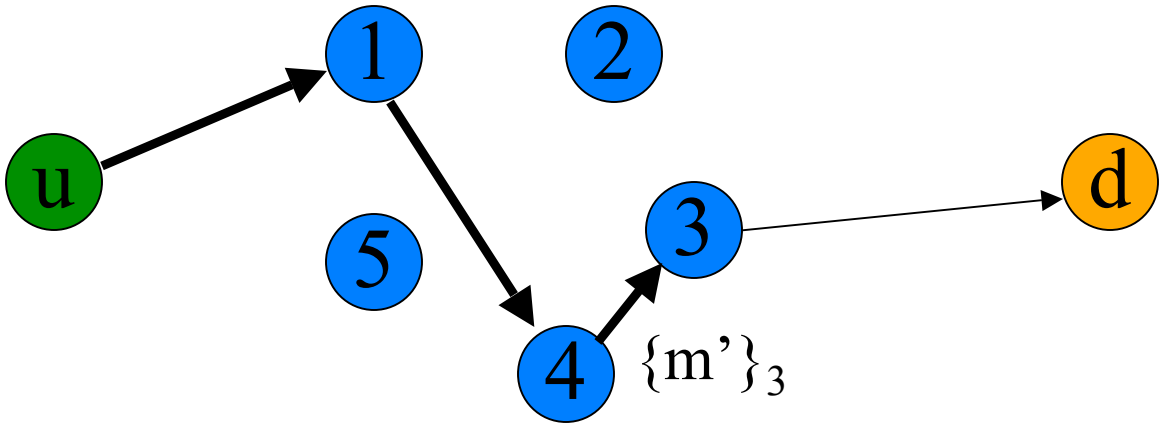
1. *u* creates 3-hop circuit through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

How Onion Routing Works



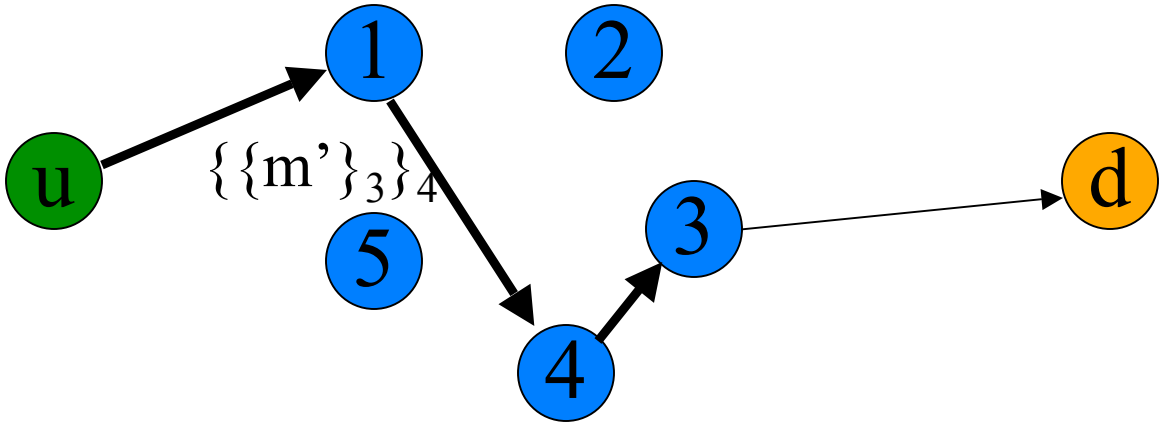
1. *u* creates 3-hop circuit through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

How Onion Routing Works



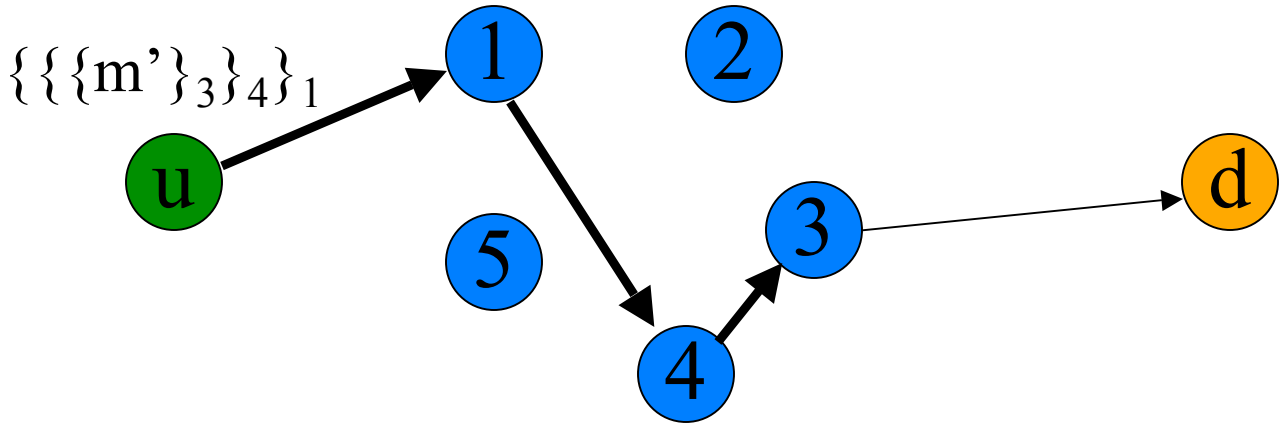
1. *u* creates 3-hop circuit through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

How Onion Routing Works



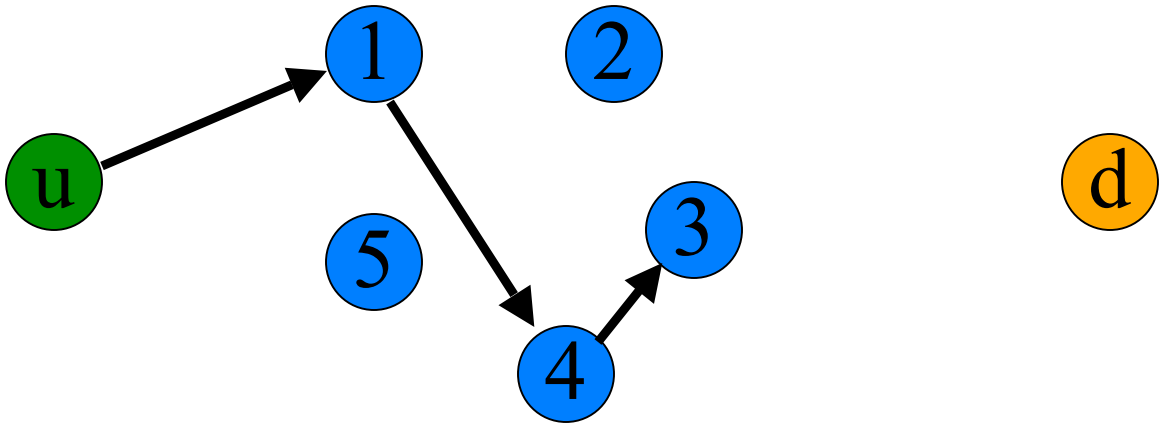
1. *u* creates 3-hop circuit through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

How Onion Routing Works



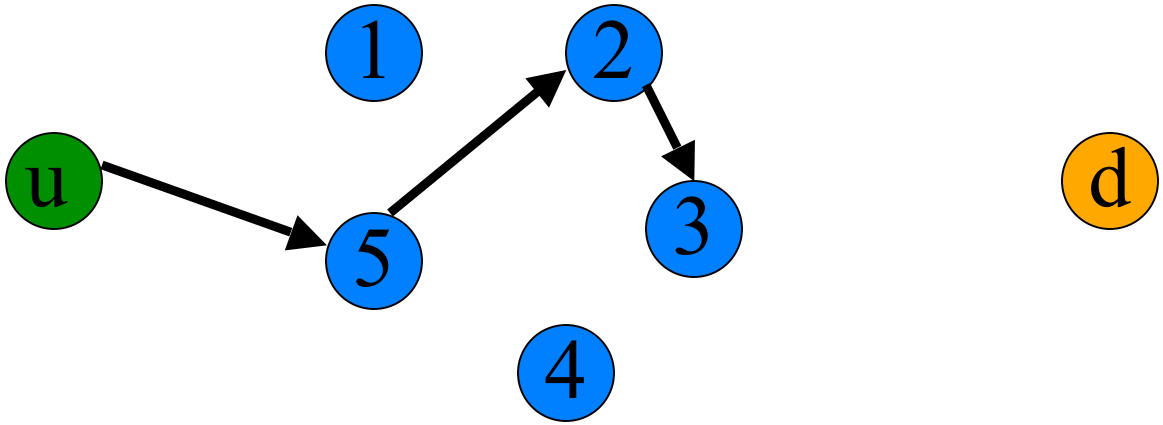
1. *u* creates 3-hop circuit through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged

How Onion Routing Works



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

How Onion Routing Works



1. *u* creates 3-hop circuit through routers
2. *u* opens a stream in the circuit to *d*
3. Data is exchanged.
4. Stream is closed.
5. Circuit is changed every few minutes.

Results

Results

1. Formally model onion routing using input/output automata

Results

1. Formally model onion routing using input/output automata
2. Analyze relationship anonymity
   a. Characterize situations with *possibilistic* anonymity
   b. Bound *probabilistic* anonymity in worst-case and typical situations

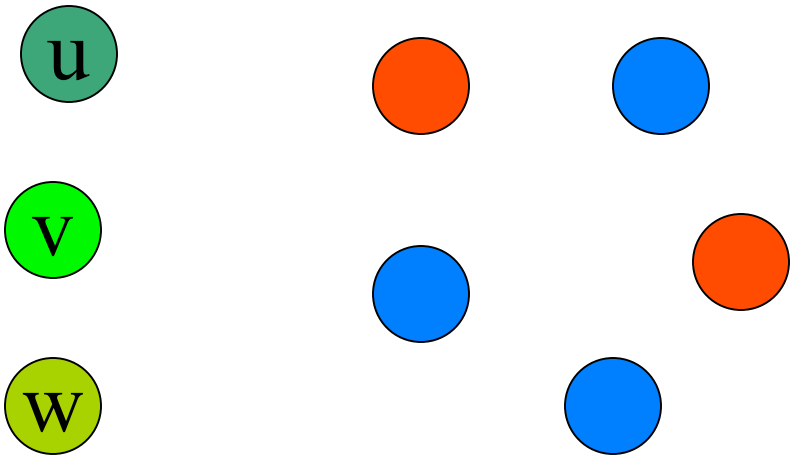
Related Work

- *A Formal Treatment of Onion Routing*
  Jan Camenisch and Anna Lysyanskaya
  CRYPTO 2005
- *A formalization of anonymity and onion routing*
  S. Mauw, J. Verschuren, and E.P. de Vink
  ESORICS 2004
- *Towards an Analysis of Onion Routing Security*
  P. Syverson, G. Tsudik, M. Reed, and C. Landwehr
  PET 2000

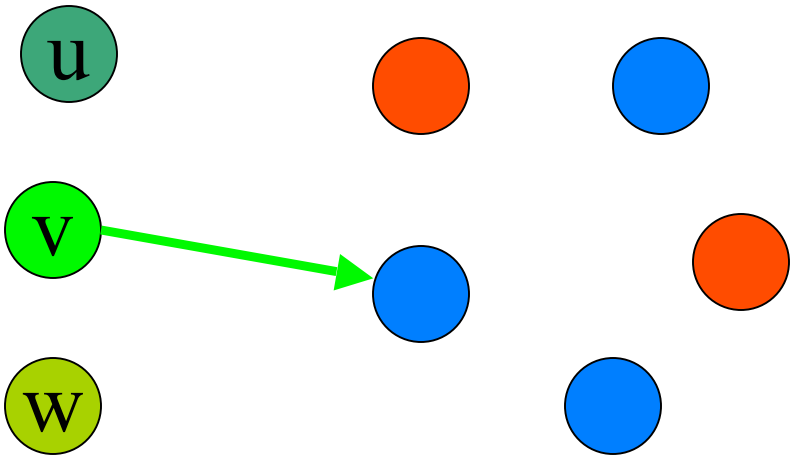
Model

- Constructed with I/O automata
  - Models asynchrony
  - Relies on abstract properties of cryptosystem
- Simplified onion-routing protocol
  - No key distribution
  - No circuit teardowns
  - No separate destinations
  - Each user constructs a circuit to one destination
  - Circuit identifiers

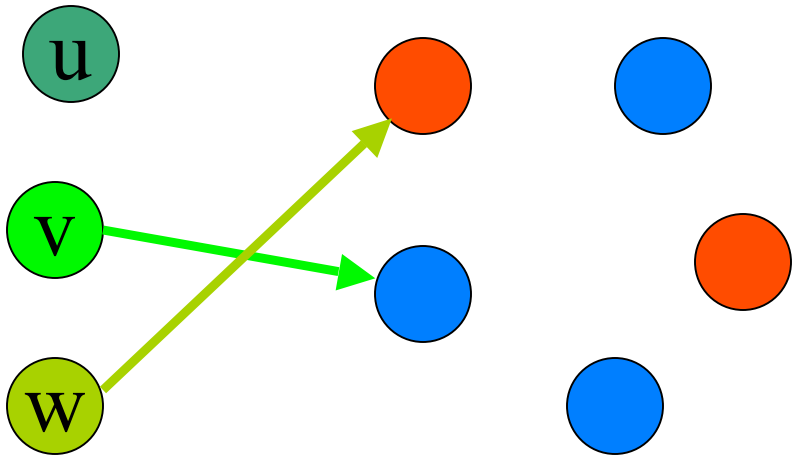
Automata Protocol



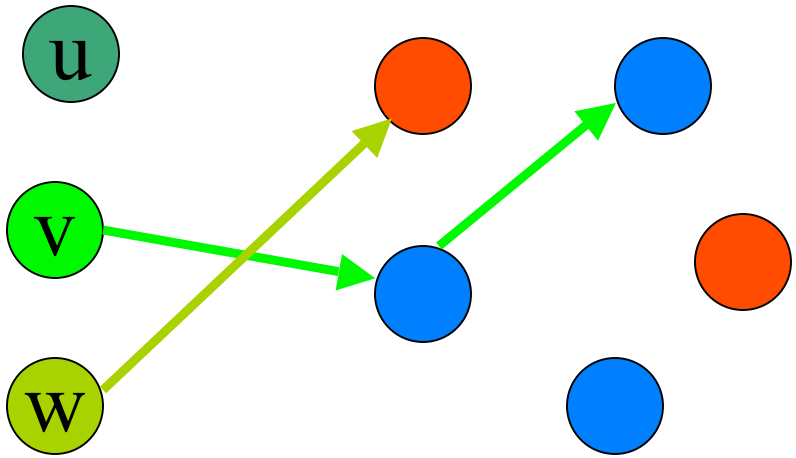
Automata Protocol



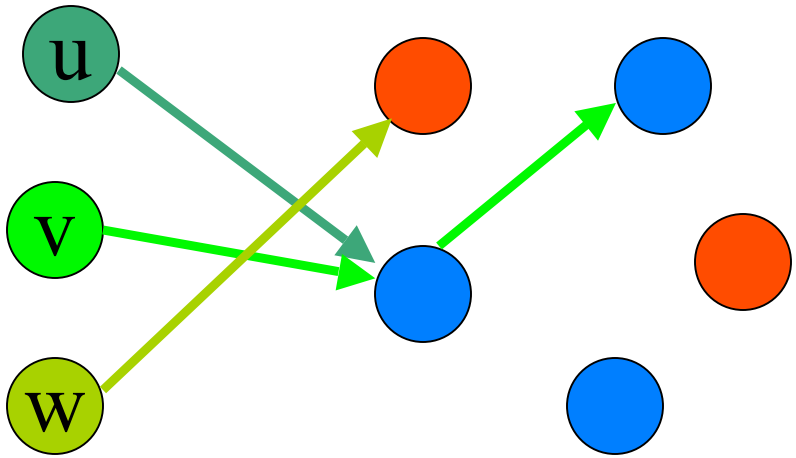
Automata Protocol



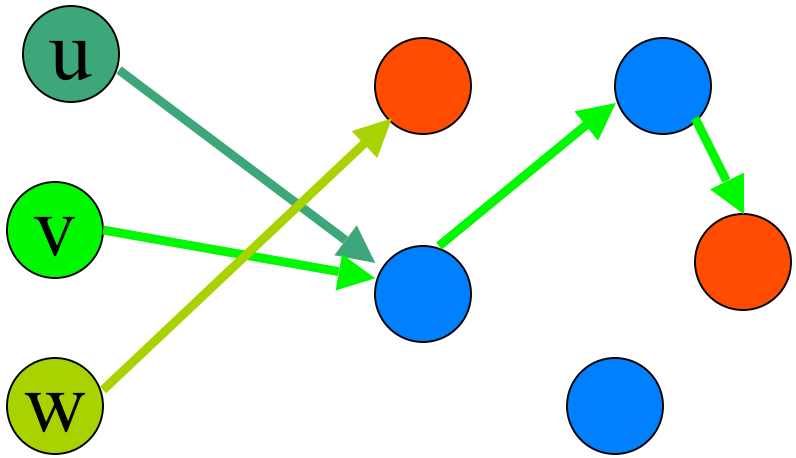
Automata Protocol



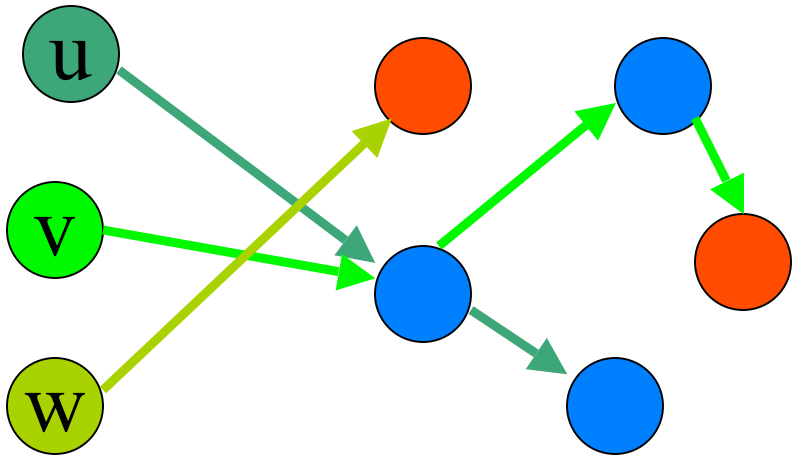
Automata Protocol



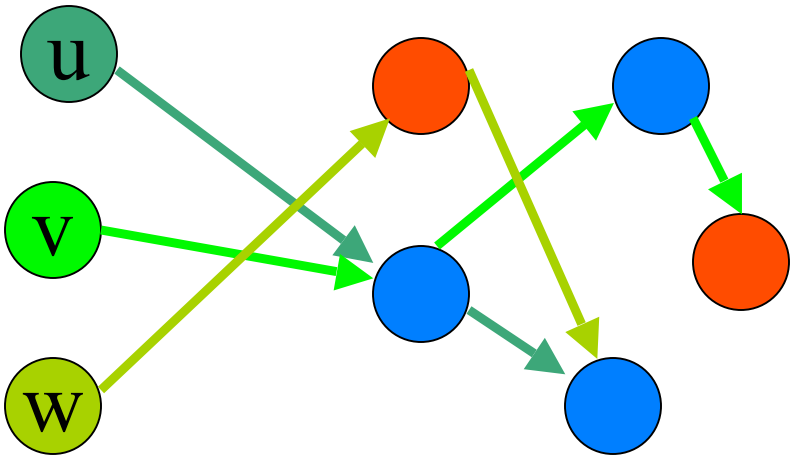
Automata Protocol



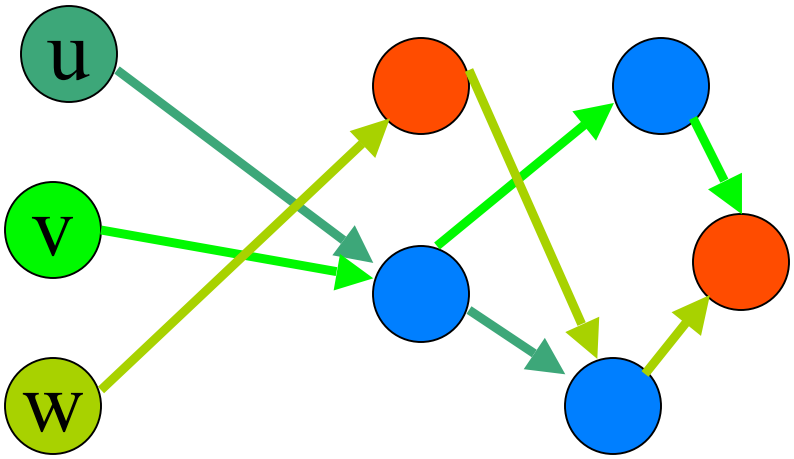
Automata Protocol



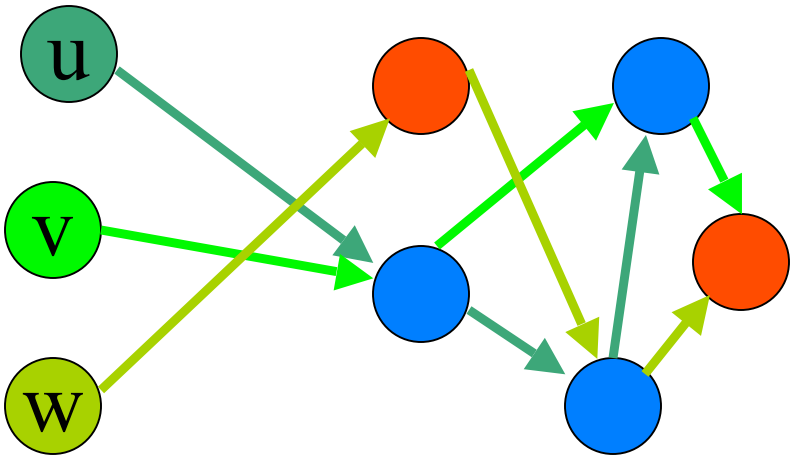
Automata Protocol



Automata Protocol



Automata Protocol



Creating a Circuit

Creating a Circuit

1. CREATE/CREATED

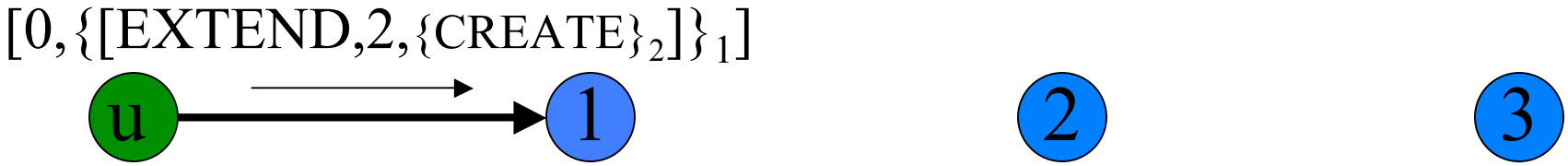
Creating a Circuit

1. CREATE/CREATED

Creating a Circuit

1. CREATE/CREATED

Creating a Circuit



1. CREATE/CREATED
2. EXTEND/EXTENDED

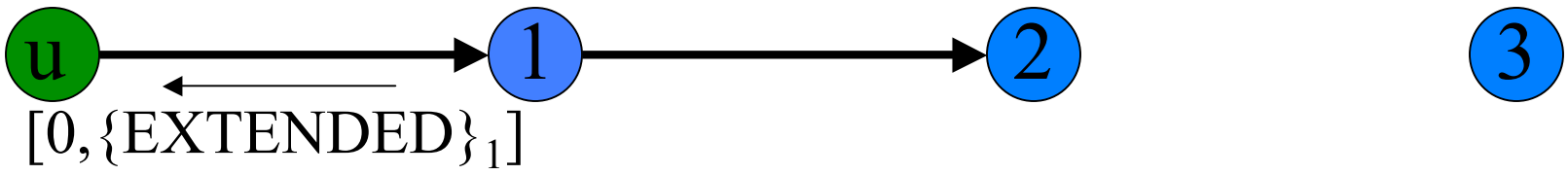
Creating a Circuit

1. CREATE/CREATED
2. EXTEND/EXTENDED

Creating a Circuit

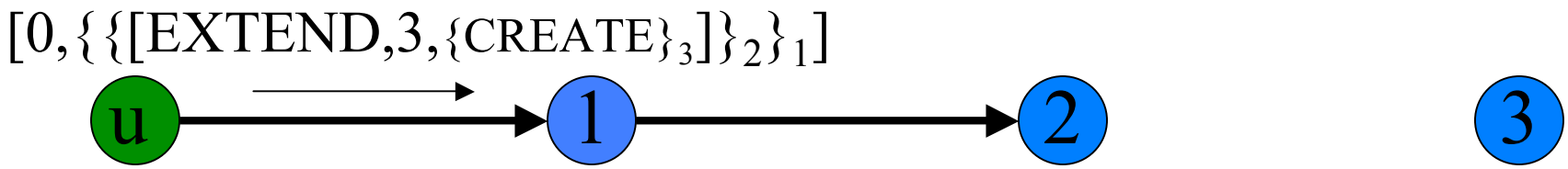
1. CREATE/CREATED
2. EXTEND/EXTENDED

Creating a Circuit



1. CREATE/CREATED
2. EXTEND/EXTENDED

Creating a Circuit



1. CREATE/CREATED
2. EXTEND/EXTENDED
3. [Repeat with layer of encryption]

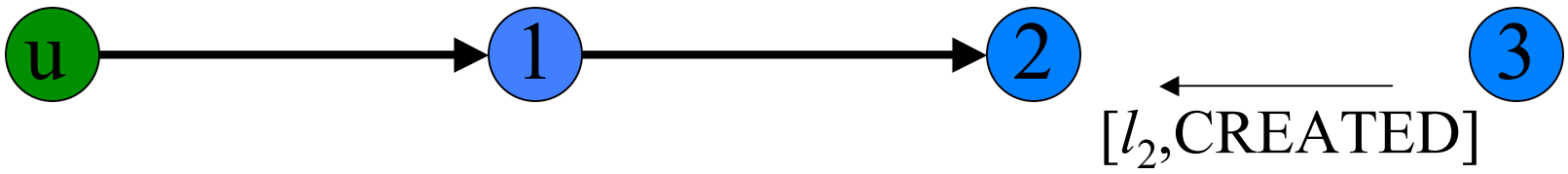
Creating a Circuit

1. CREATE/CREATED
2. EXTEND/EXTENDED
3. [Repeat with layer of encryption]

Creating a Circuit

1. CREATE/CREATED
2. EXTEND/EXTENDED
3. [Repeat with layer of encryption]

Creating a Circuit

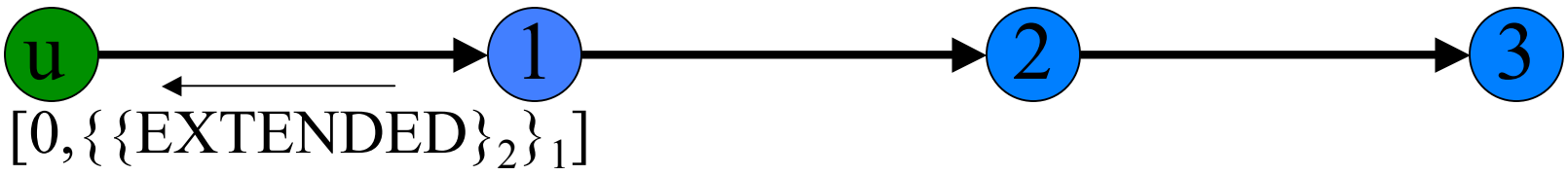


1. CREATE/CREATED
2. EXTEND/EXTENDED
3. [Repeat with layer of encryption]

Creating a Circuit

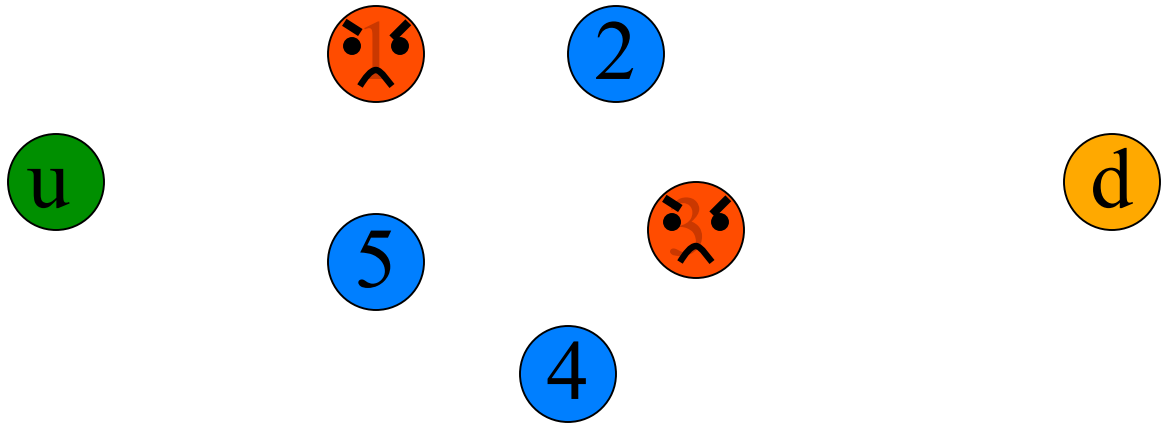
1. CREATE/CREATED
2. EXTEND/EXTENDED
3. [Repeat with layer of encryption]

Creating a Circuit



1. CREATE/CREATED
2. EXTEND/EXTENDED
3. [Repeat with layer of encryption]

Adversary



Active & Local

Possibilistic Anonymity

Anonymity

Let U be the set of users.
Let R be the set of routers.
Let l be the path length.

Anonymity

Let U be the set of users.

Let R be the set of routers.

Let l be the path length.

Definition (configuration):

A ***configuration*** is a function $U \rightarrow R^l$ mapping each user to his circuit.

Anonymity

Let U be the set of users.

Let R be the set of routers.

Let l be the path length.

Definition (configuration):

A ***configuration*** is a function $U \rightarrow R^l$ mapping each user to his circuit.

Definition (indistinguishability):

Executions α and β are ***indistinguishable*** to adversary A when his actions in β are the same as in α after possibly applying the following:

ξ: A permutation on the keys not held by A.

π: A permutation on the messages encrypted by a key not held by A.

Anonymity

Definition (fair): In *fair* executions actions enabled infinitely often occur infinitely often

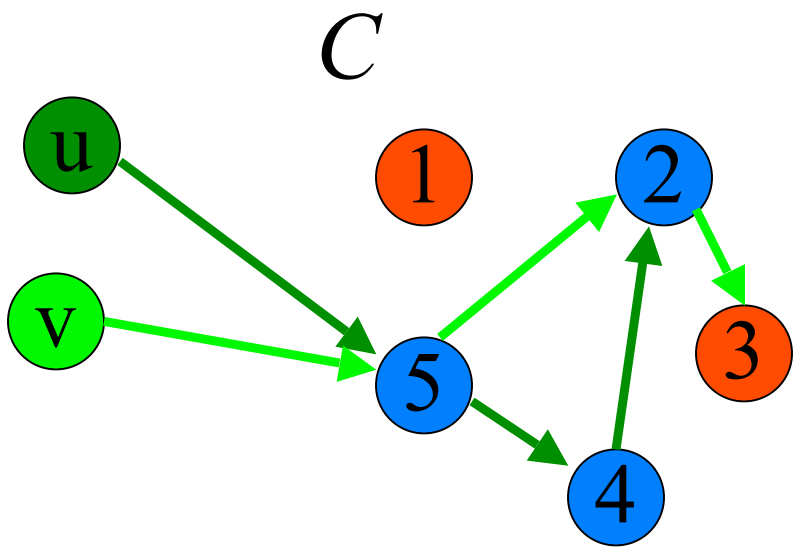
Definition (cryptographic): In *cryptographic* executions no encrypted control messages are sent before they are received unless the sender possesses the key

Definition (relationship anonymity):

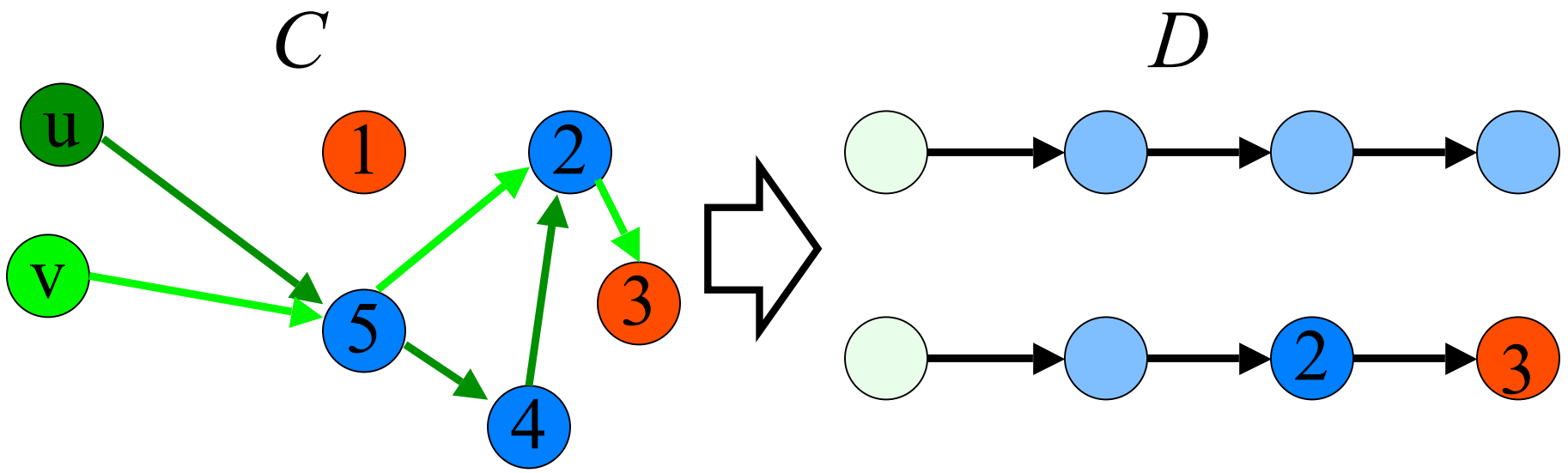
u and *d* have *relationship anonymity* in configuration *C* with respect to adversary *A* if, for every fair, cryptographic execution of *C* in which *u* talks to *d*, there exists a fair, cryptographic execution that is indistinguishable to *A* in which *u* does not talk to *d*.

Theorem 1: Let C and D be configurations for which there exists a permutation ρ: $U \rightarrow U$ such that $C_i(u) = D_i(\rho(u))$ if $C_i(u)$ or $D_i(\rho(u))$ is compromised or is adjacent to a compromised router. Then for every fair, cryptographic execution α of C there exists an indistinguishable, fair, cryptographic execution β of D. The converse also holds.

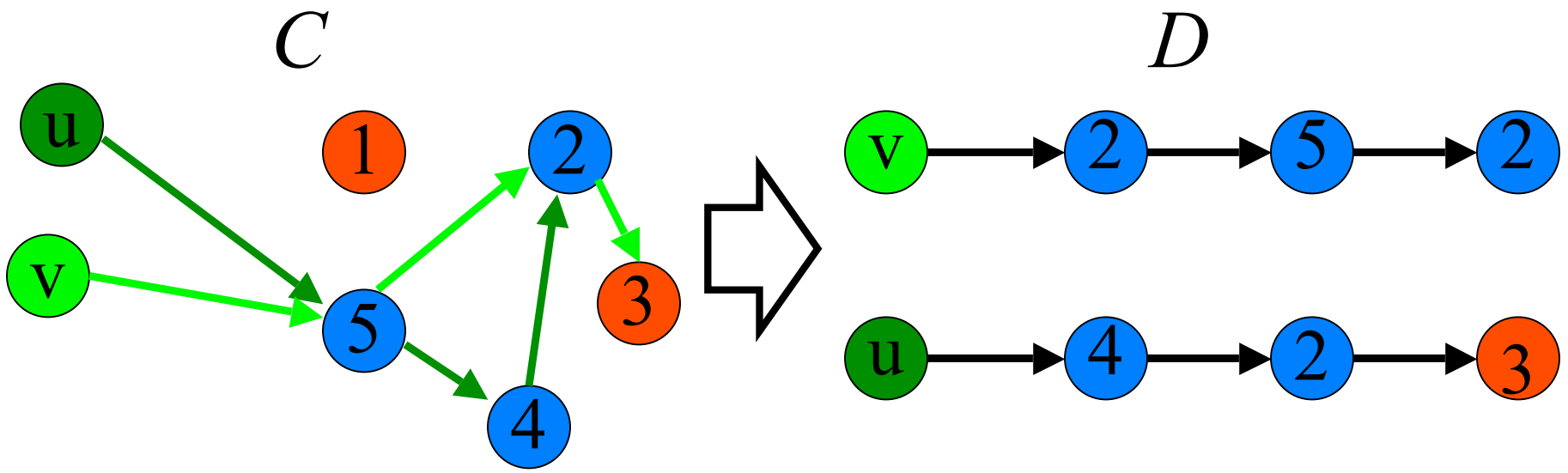
Theorem 1: Let C and D be configurations for which there exists a permutation ρ: $U \rightarrow U$ such that $C_i(u) = D_i(\rho(u))$ if $C_i(u)$ or $D_i(\rho(u))$ is compromised or is adjacent to a compromised router. Then for every fair, cryptographic execution α of C there exists an indistinguishable, fair, cryptographic execution β of D. The converse also holds.



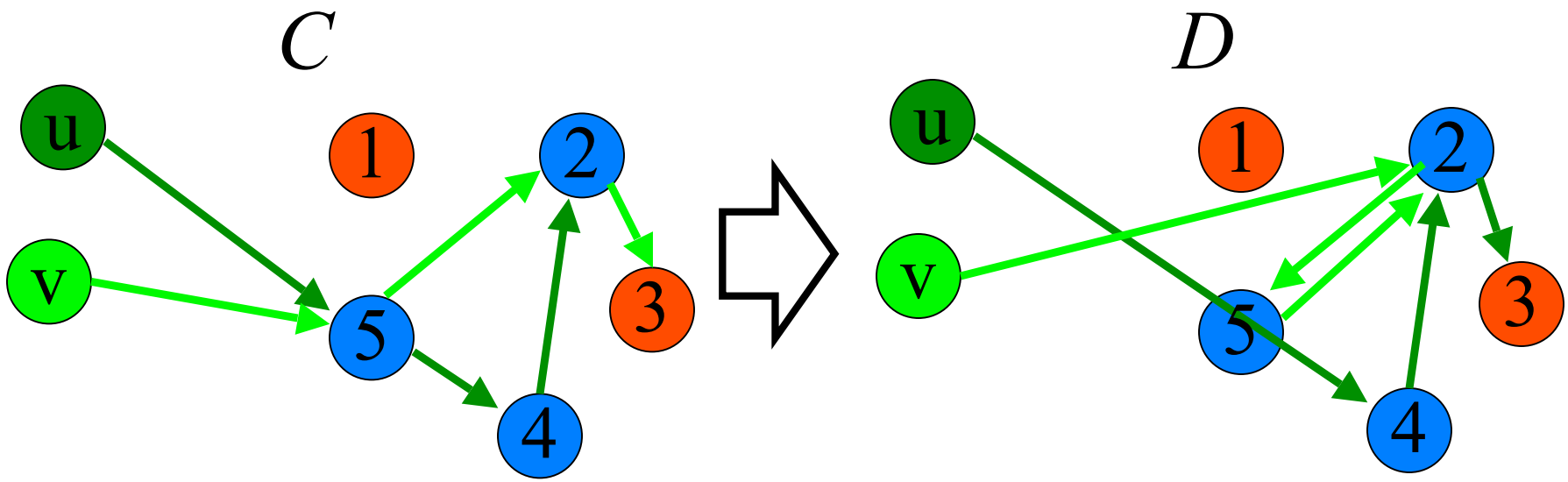
Theorem 1: Let C and D be configurations for which there exists a permutation ρ: $U \rightarrow U$ such that $C_i(u) = D_i(\rho(u))$ if $C_i(u)$ or $D_i(\rho(u))$ is compromised or is adjacent to a compromised router. Then for every fair, cryptographic execution α of C there exists an indistinguishable, fair, cryptographic execution β of D. The converse also holds.



Theorem 1: Let C and D be configurations for which there exists a permutation ρ: $U \rightarrow U$ such that $C_i(u) = D_i(\rho(u))$ if $C_i(u)$ or $D_i(\rho(u))$ is compromised or is adjacent to a compromised router. Then for every fair, cryptographic execution α of C there exists an indistinguishable, fair, cryptographic execution β of D. The converse also holds.



Theorem 1: Let C and D be configurations for which there exists a permutation ρ: $U \rightarrow U$ such that $C_i(u) = D_i(\rho(u))$ if $C_i(u)$ or $D_i(\rho(u))$ is compromised or is adjacent to a compromised router. Then for every fair, cryptographic execution α of C there exists an indistinguishable fair, cryptographic execution β of D. The converse also holds.

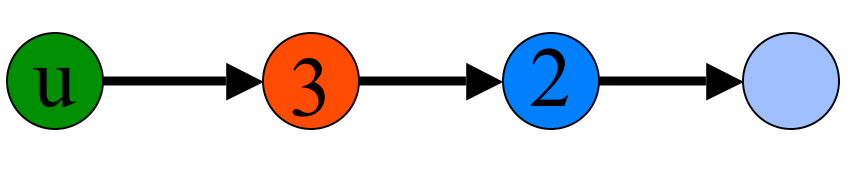


Relationship anonymity

Corollary : A user and destination have rel. anonymity iff:

Relationship anonymity

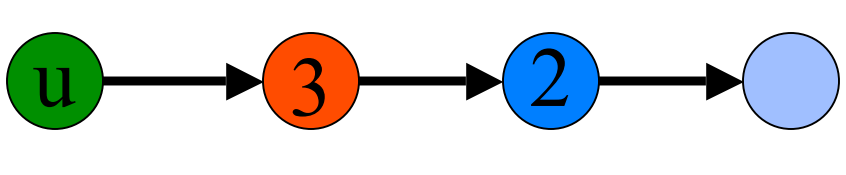
Corollary : A user and destination have rel. anonymity iff:



The last router is unknown.

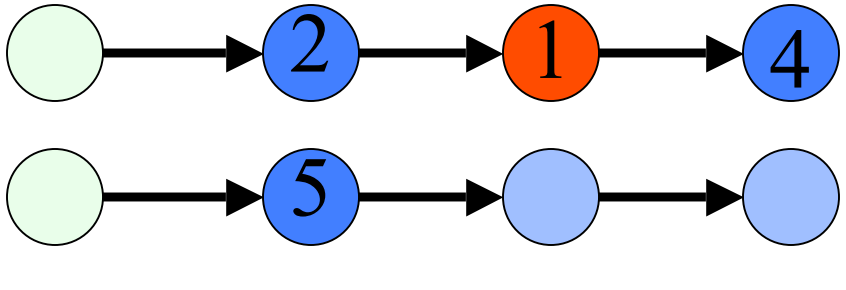
Relationship anonymity

Corollary : A user and destination have rel. anonymity iff:



The last router is unknown.

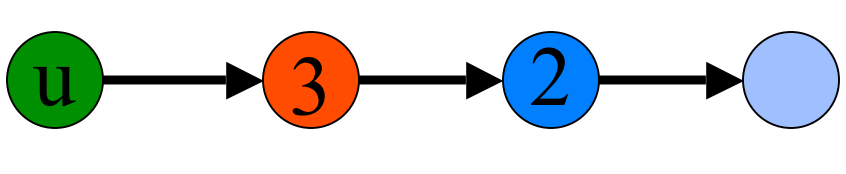
OR



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

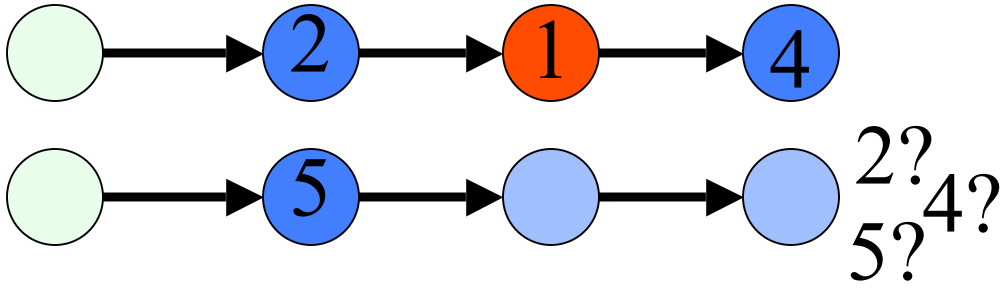
Relationship anonymity

Corollary : A user and destination have rel. anonymity iff:



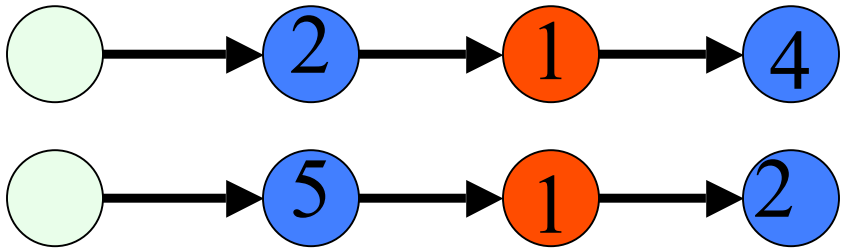
The last router is unknown.

OR



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

OR



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

Probabilistic Anonymity

Adding probability

1. To configurations
   a. User u selects destination d with probability p^u_d.
   b. Users select routers randomly.
2. To executions
   a. Each execution in a configuration is equally likely.

Definition (relationship anonymity):

u and *d* have relationship anonymity in configuration *C* with respect to adversary *A* if, for every fair, cryptographic execution of *C* in which *u* talks to *d*, there exists a fair, cryptographic execution that is indistinguishable to *A* in which *u* does not talk to *d*.

Definition (relationship anonymity):

u and *d* have relationship anonymity in configuration *C* with respect to adversary *A* if, for every fair, cryptographic execution of *C* in which *u* talks to *d*, there exists a fair, cryptographic execution that is indistinguishable to *A* in which *u* does not talk to *d*.

Probabilistic relationship anonymity metric:

Let X be a random configuration.

Let X_D: $U \rightarrow D$ be the destinations in X.

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

- Other measures (e.g. entropy, min entropy)
- Measures effect on the individual user
- Exact Bayesian inference
  - Adversary after long-term intersection attack
  - Worst-case adversary

Probabilistic Anonymity

1. Fixing u and d doesn't determine C.
2. $Y(C)$ thus has a distribution.
3. This distribution depends on each p^v.
4. $\mathbf{E}[Y \mid X_D(u)=d]$
   a. Worst case
   b. Typical case

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 2: 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 Estimates

Let n be the number of nodes.
Let b be the fraction of compromised nodes.

Worst-case Estimates

Let n be the number of nodes.

Let b be the fraction of compromised nodes.

Theorem 3: When $p^v_\delta = 1$ for all $v \neq u$:

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

Worst-case Estimates

Let n be the number of nodes.
Let b be the fraction of compromised nodes.

Theorem 3: When $p^v_\delta = 1$ for all $v \neq u$:

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

Theorem 4: When $p^v_d = 1$ for all $v \neq u$:

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

Worst-case Estimates

Let n be the number of nodes.

Let b be the fraction of compromised nodes.

Theorem 3: When $p^v_\delta = 1$ for all $v \neq u$:

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

Worst-case Estimates

Let n be the number of nodes.
Let b be the fraction of compromised nodes.

Theorem 3: When $p^v_\delta = 1$ for all $v \neq u$:

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

Worst-case Estimates

Let n be the number of nodes.
Let b be the fraction of compromised nodes.

Theorem 3: When $p^v{}_\delta = 1$ for all $v \neq u$:

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

Typical Case

Let each user select from the Zipfian distribution:

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

Theorem 5:

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

Typical Case

Let each user select from the Zipfian distribution:

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

Theorem 5:

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

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

Results

1. Formally model onion routing using input/output automata
2. Analyze relationship anonymity
   a. Characterize situations with *possibilistic* anonymity
   b. Bound *probabilistic* anonymity in worst-case and typical situations

Future Work

1. Extend analysis to other types of anonymity and to other systems.
2. Examine how quickly users distribution are learned.
3. Analyze timing attacks.