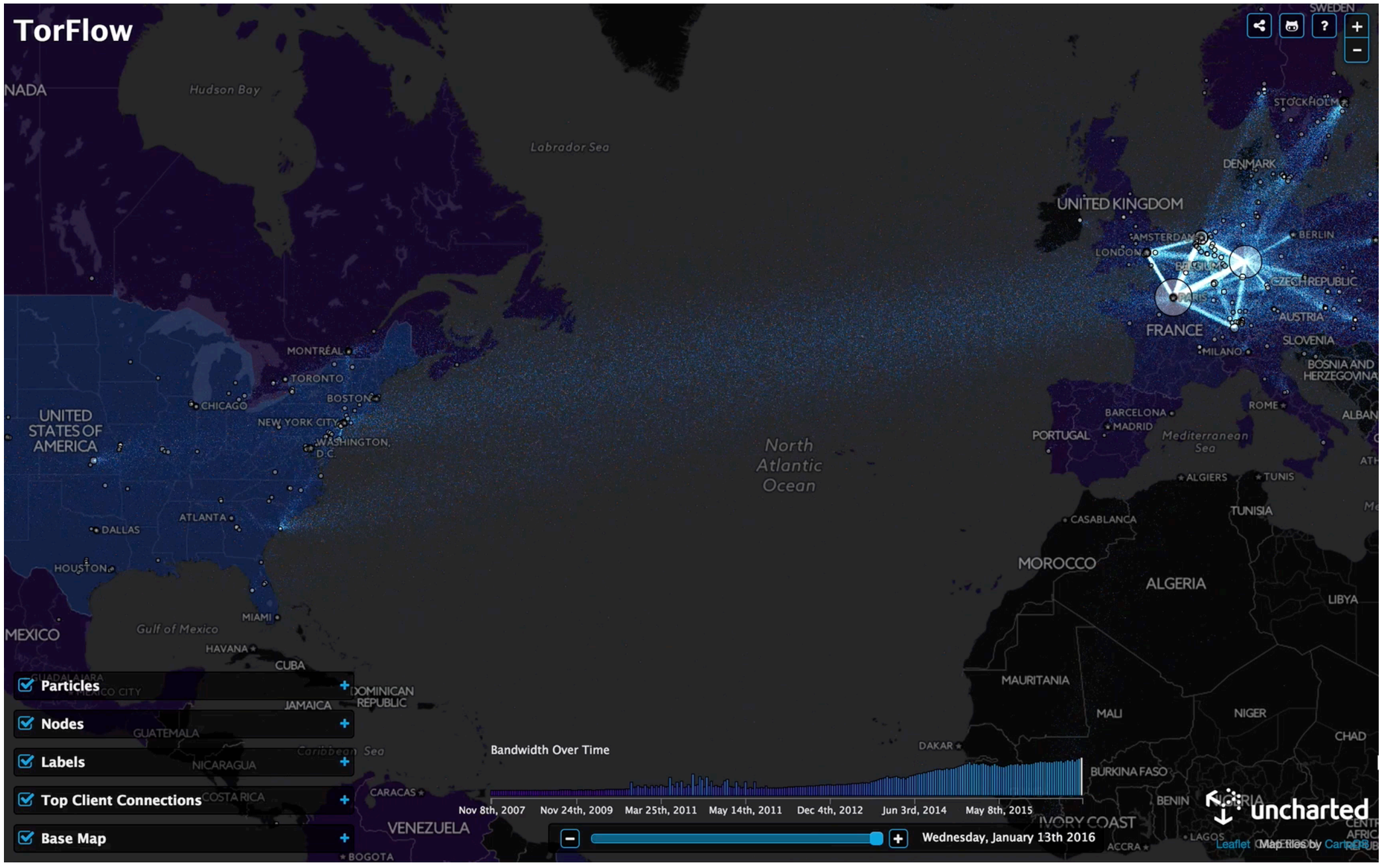
# HisTorε: Differentially Private and Robust Statistics Collection for Tor

Akshaya Mani, Micah Sherr

Georgetown University

Tor is an anonymity service,
used by *N* people to do **?**, **?**, and **?**

# Tor is an anonymity service, used by *N* people to do **?**, **?**, and **?**

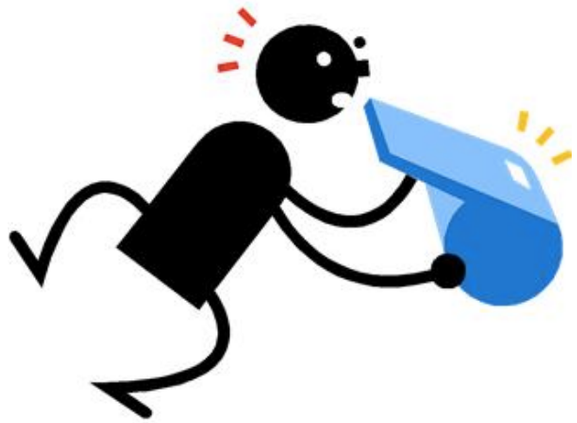


# Who uses Tor, and for what?

## According to the Tor Project:

Ordinary Internet users
for more private browsing


Journalists
to safely report the news



Whistleblowers
to report wrongdoings

# Who uses Tor, and for what?

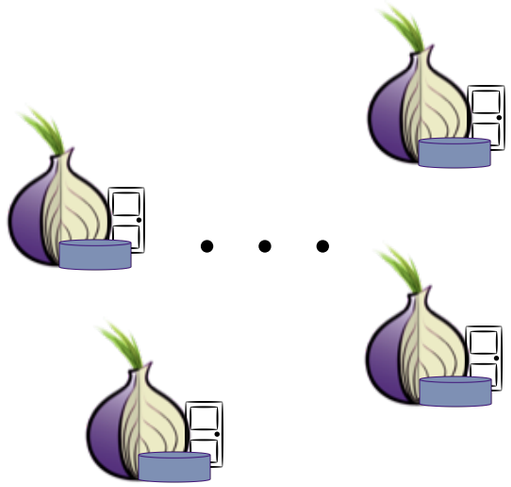
According to others:

# Motivation

**Safely Measure how Tor is used**

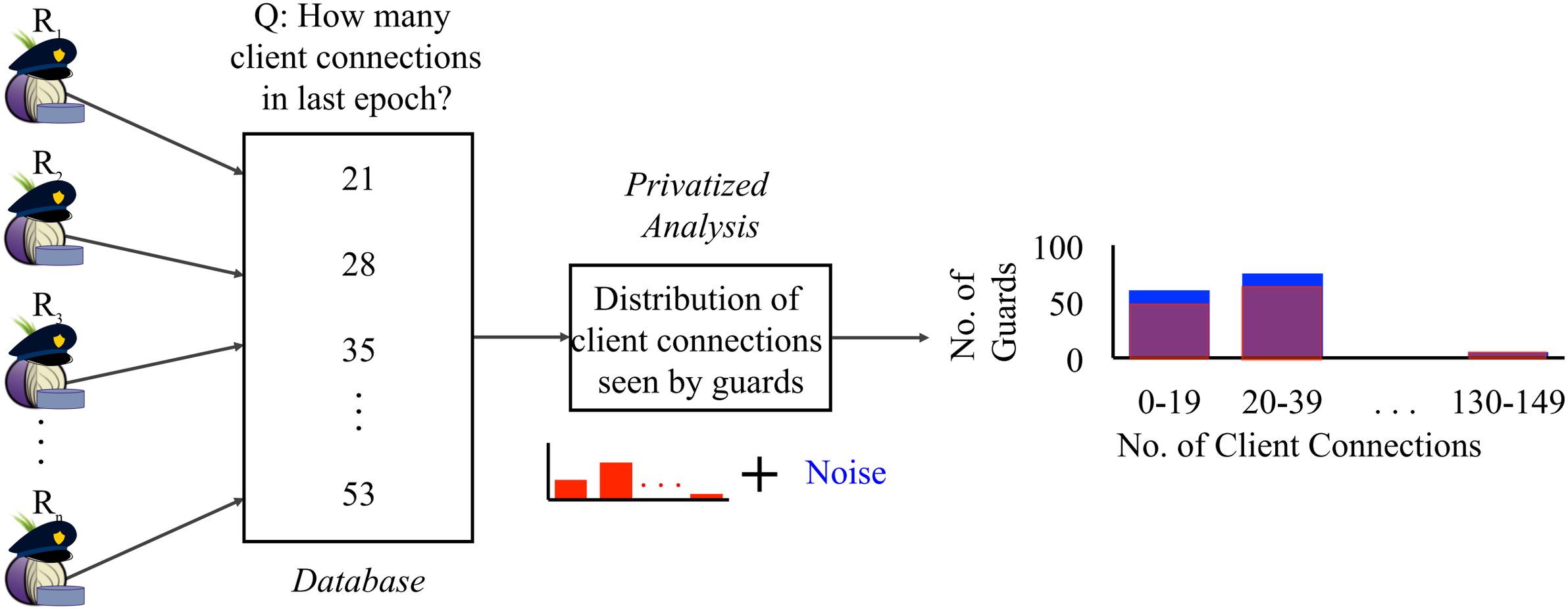
# A Naïve Solution

E.g. *How many exit relays forward traffic to* ***google.com****?*



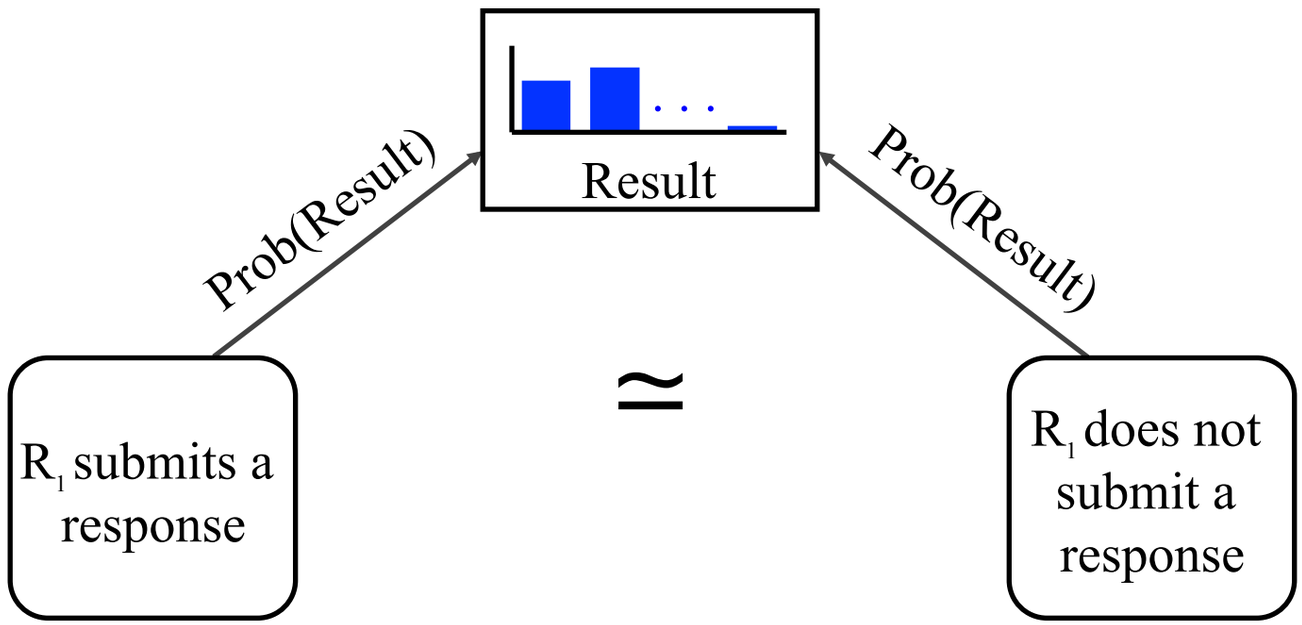
- Naïve solution:  keep log of websites visited
  - Significantly endangers users' anonymity / antithetical to Tor's mission
- **Measurement framework should not risk anonymity**

# Safe Tor Measurements with Differential Privacy



# A 30 Second Primer on Differential Privacy

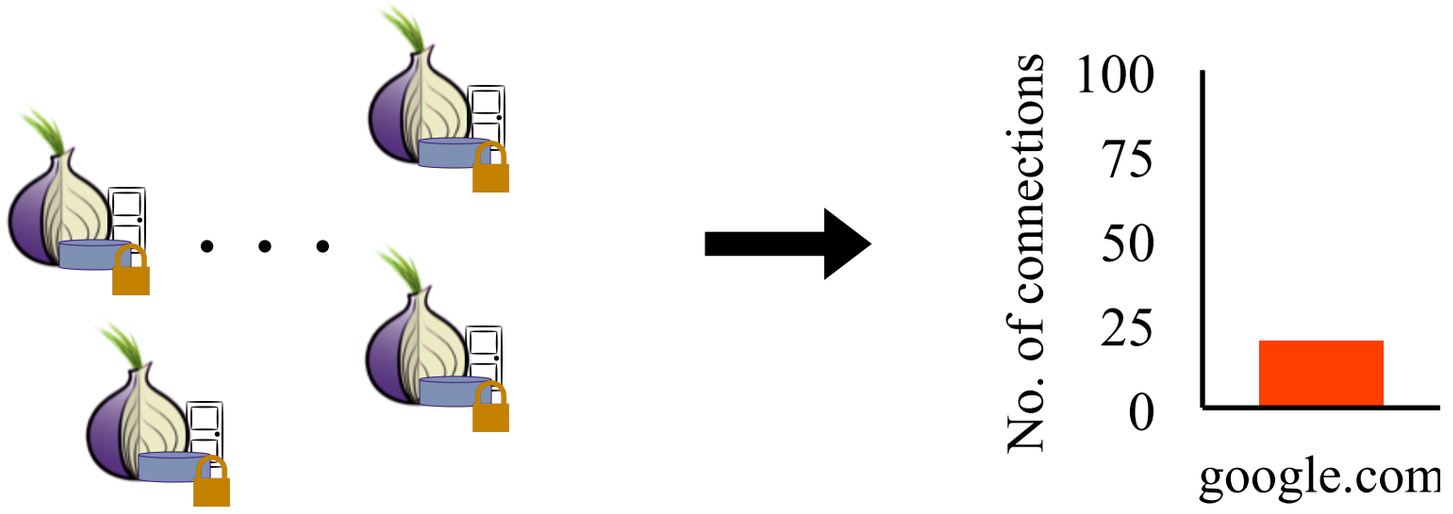
- **Differential Privacy** [Dwork ICALP' 06]




- Minimizes and quantifies the privacy risk
- Provides good accuracy
- Makes no assumptions on knowledge of attacker

# Privacy != Integrity

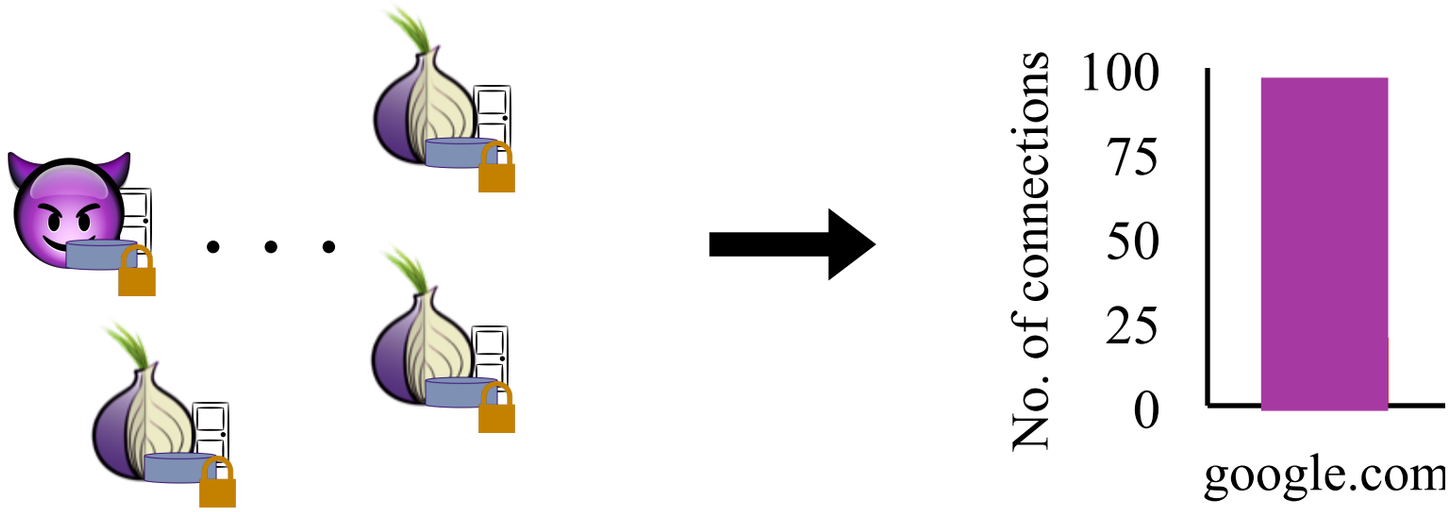
*E.g. How many exit relays forward traffic to **google.com**?*




- PrivEx [Elahi et al. CCS'14], PrivCount [Jansen et al. CCS' 16]
- Raises the bar for safe Tor measurements
- But… a malicious relay can drive aggregate to arbitrary value

# Privacy != Integrity

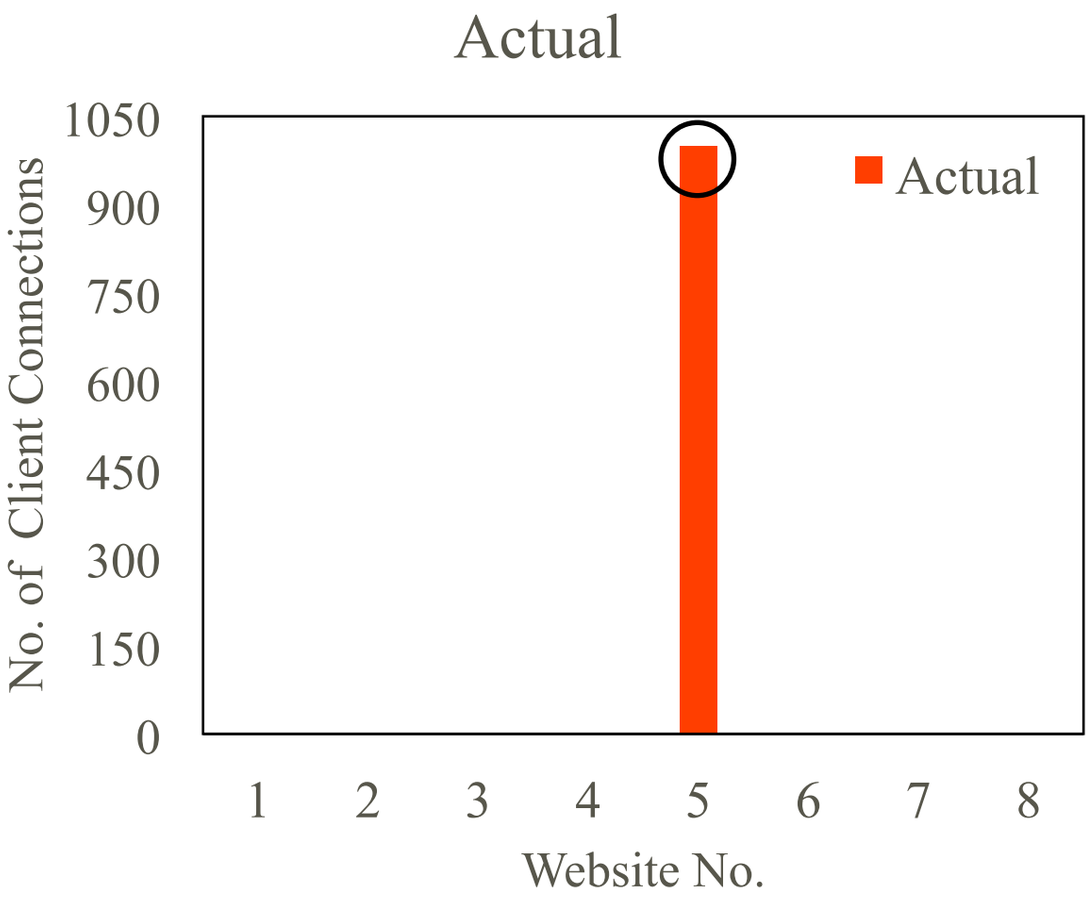
*E.g. How many exit relays forward traffic to **google.com**?*



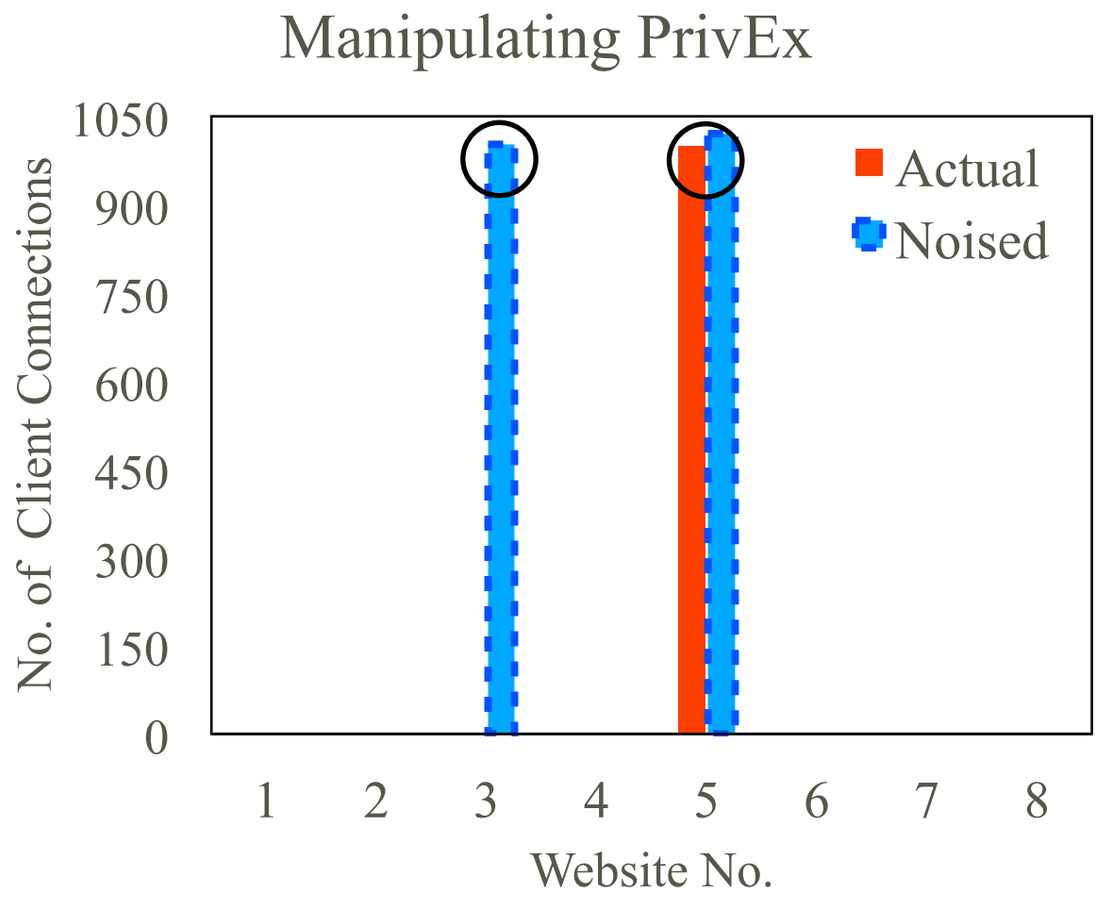

- PrivEx [Elahi et al. CCS'14], PrivCount [Jansen et al. CCS' 16]
- Raises the bar for safe Tor measurements
- But… a malicious relay can drive aggregate to arbitrary value

# Manipulating Privex

- *No. of visits destined to 15 particular websites*




- Website #5 is popular

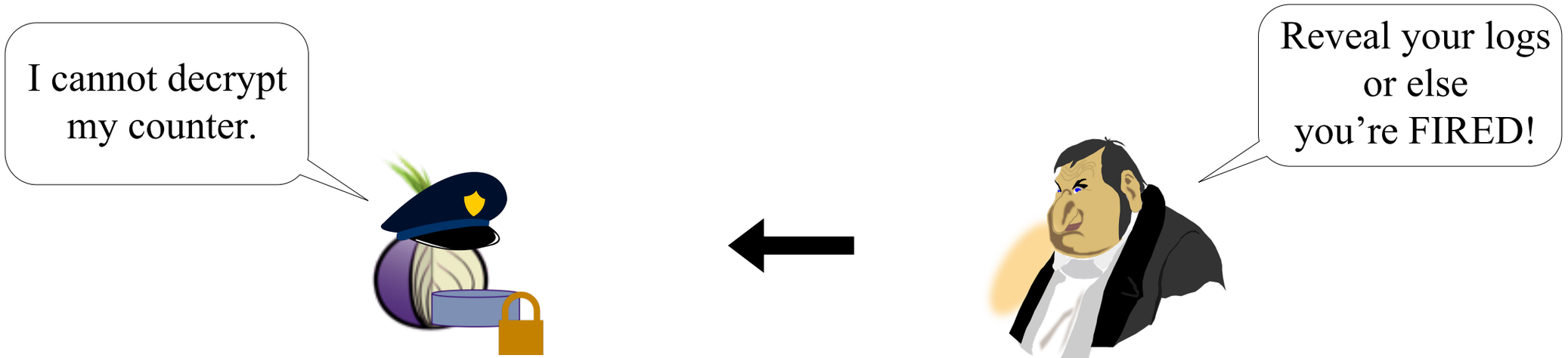



- Website #3 & #5 are popular

# HisTorε

- **His**togram query for **Tor** with (**ε**, $\delta$) differential privacy
- Provides strong integrity guarantees
- Resistant to “compulsion attacks”
- Highly accurate
- Incurs low overheads

# Compulsion Attack

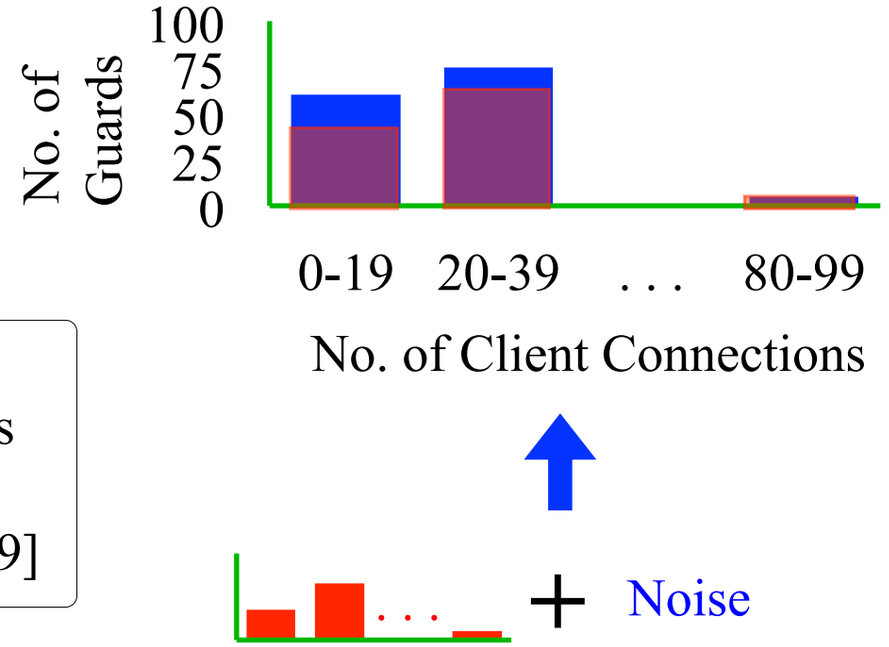


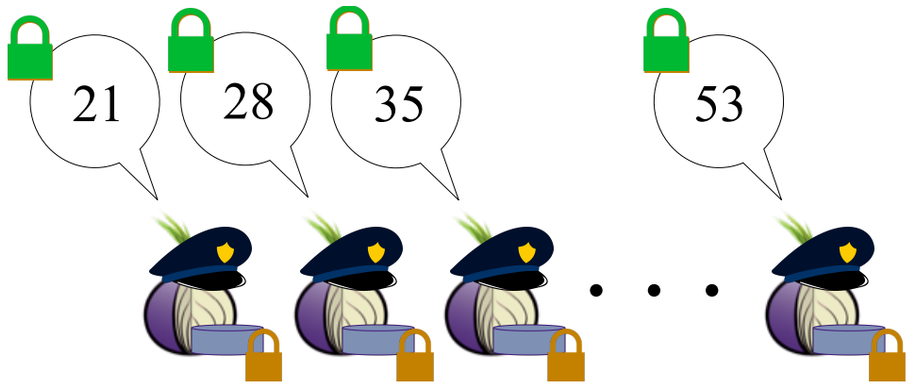

- Gathering statistics poses privacy risk
  - Performing logging at relays inherently increases privacy risks
- If relays keep logs, they can be compelled to reveal them
- Measurement system should resist compulsion attacks by obliviously storing local counters

# HisTorε at 25,000 ft

Analyst

E.g. How many client connections have the guards seen?
[0 - 19] [20 - 39] . . . [80 - 99]



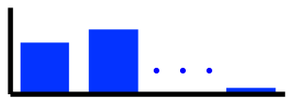



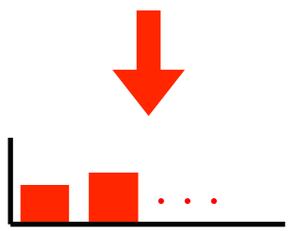

Relays

Mixes

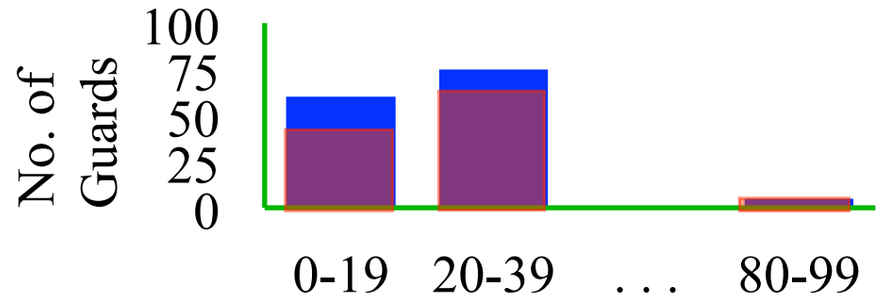
# HisTorε at 25,000 ft

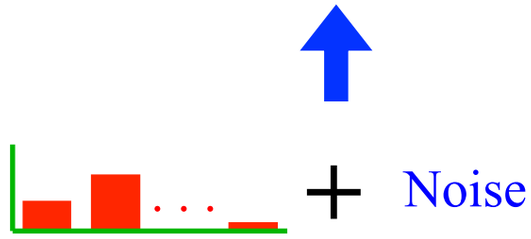
Analyst

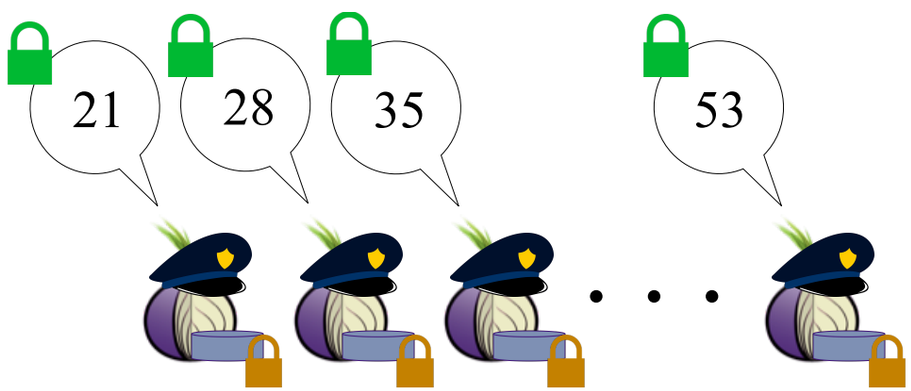




E.g. How many client connections have the guards seen?
[0 - 19] [20 - 39] . . . [80 - 99]




Relays

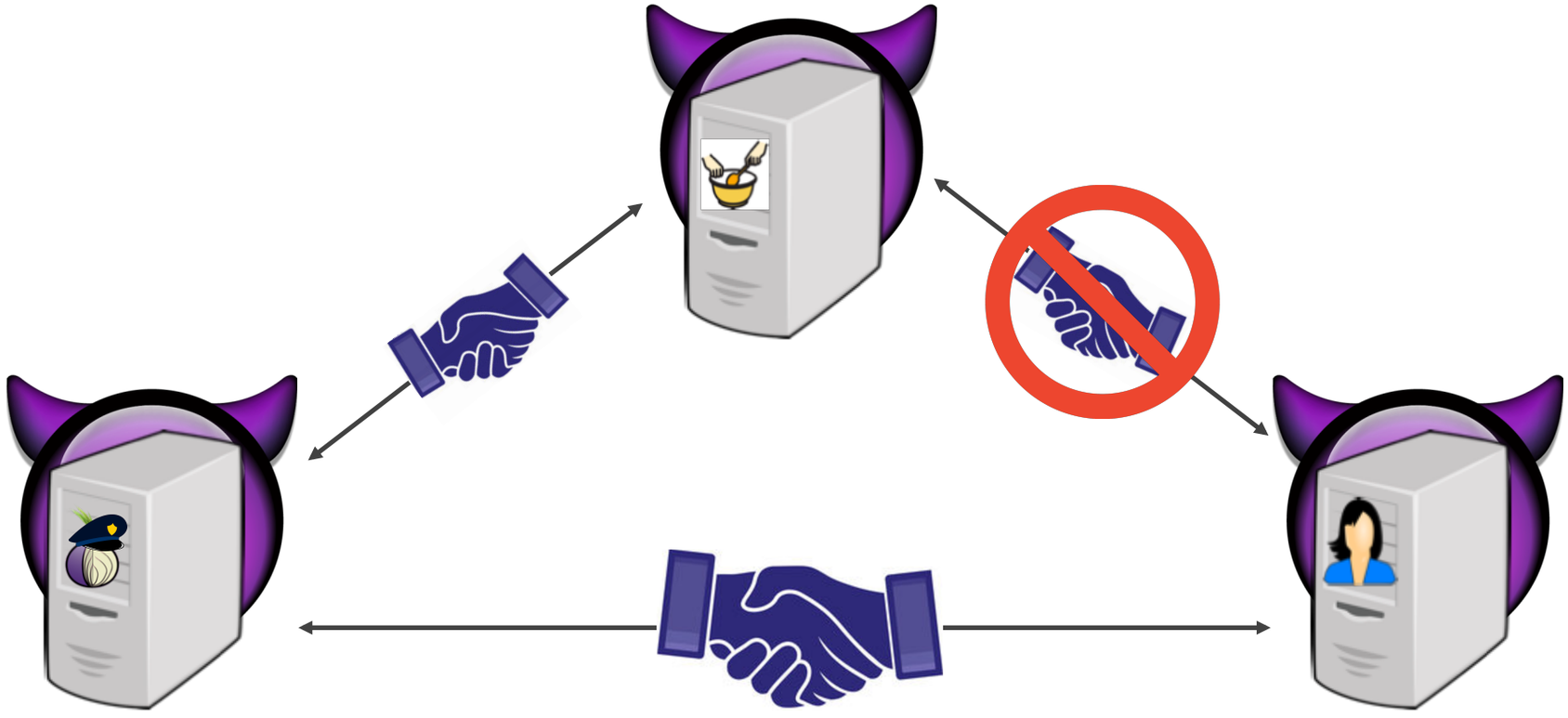
Mixes

# Trust Assumptions

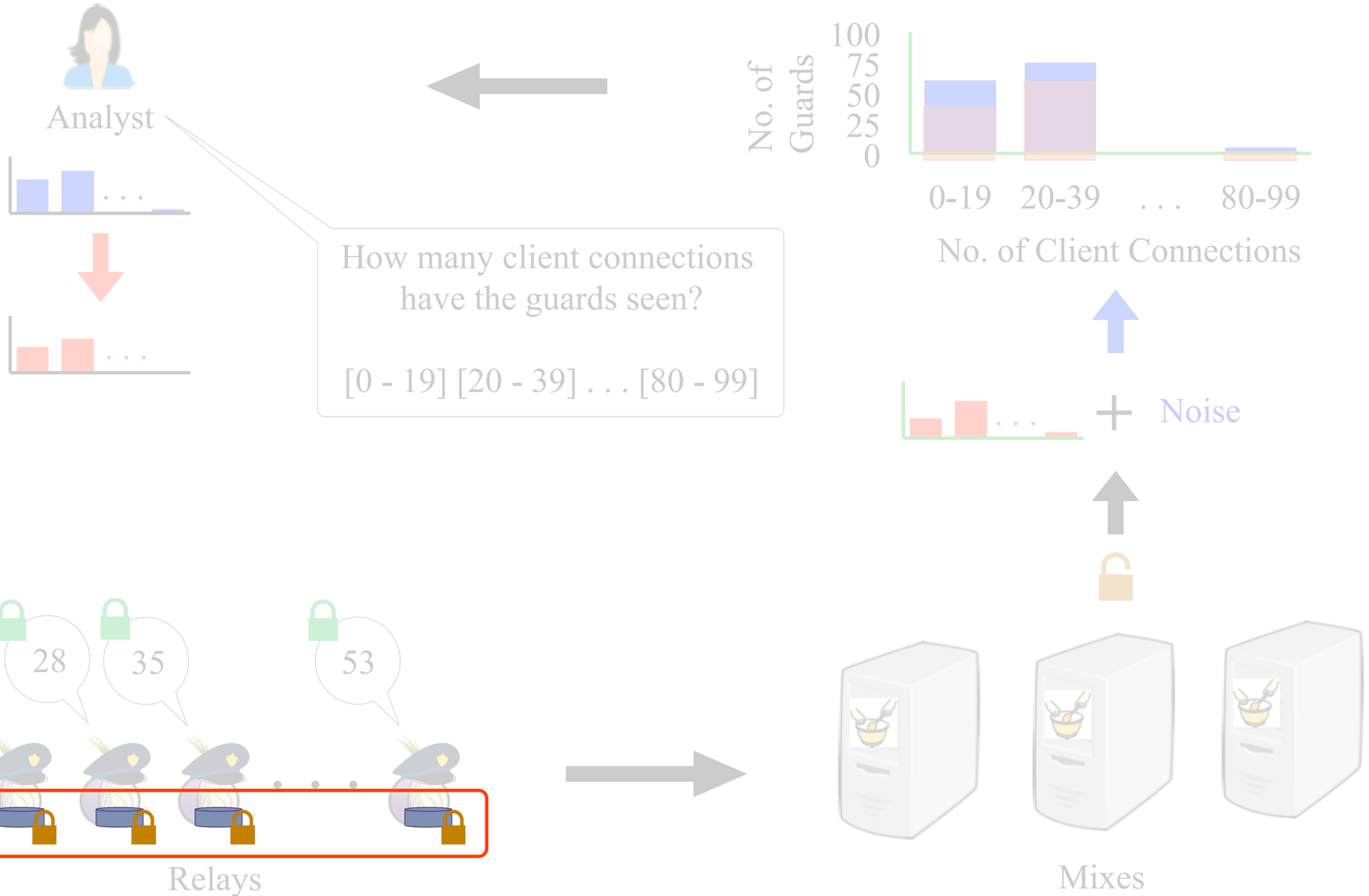
- Malicious participants…
  - …disobey protocols, and/or refuse to participate

# Trust Assumptions

- Malicious participants…
  - …disobey protocols, and/or refuse to participate

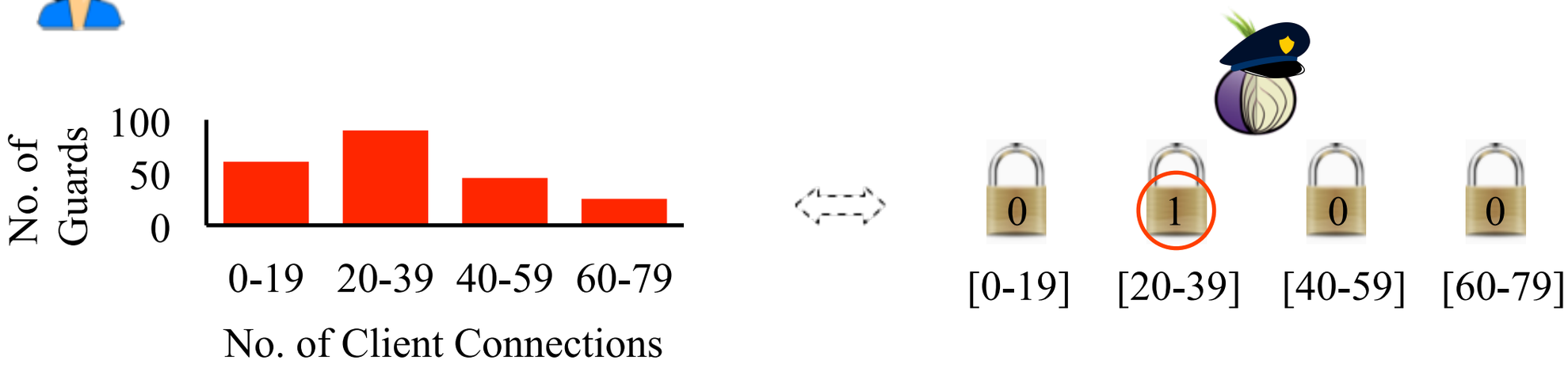


# Maintaining Oblivious Counters



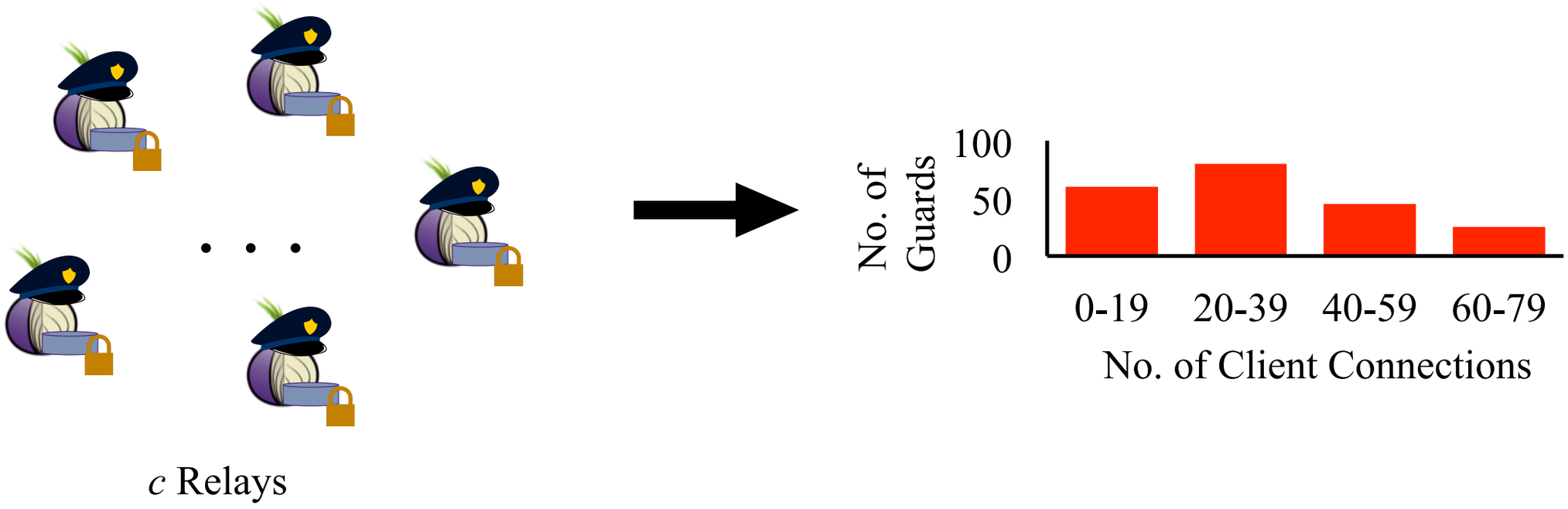
# Maintaining Oblivious Counters

Q: *What is the distribution of client connections seen by guards?*



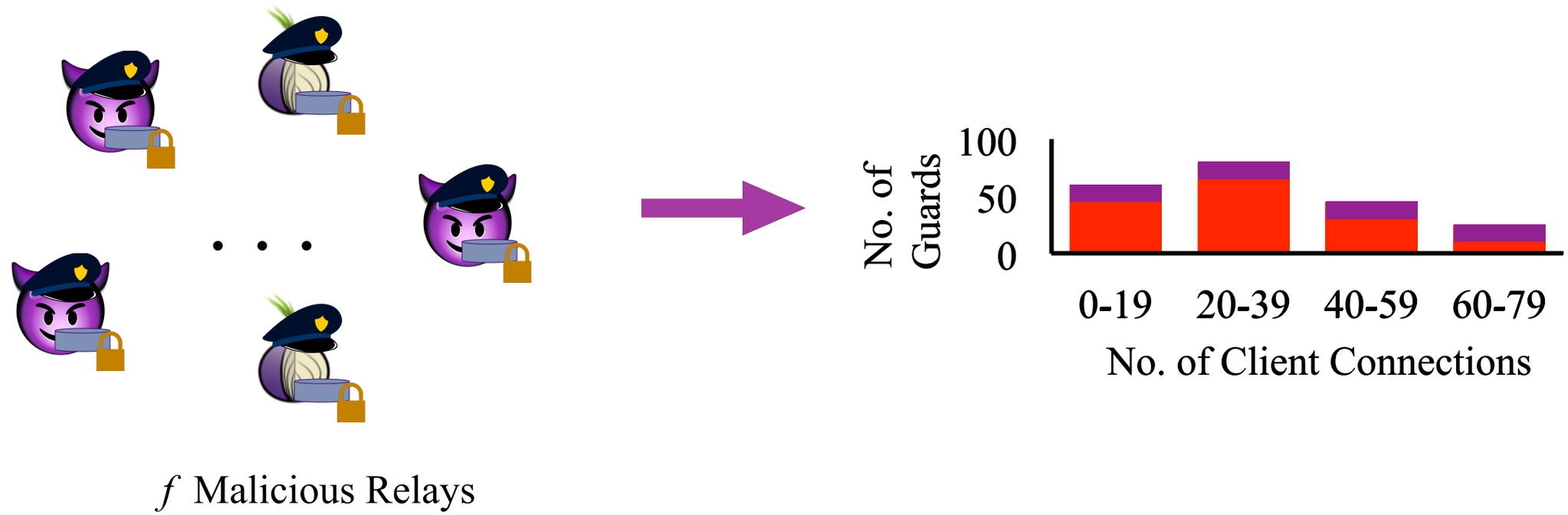

- Each relay maintains an encrypted binary vector
- At most one bin is set to *1*
- Each binary element is GM-encrypted <u>using the public key of a mix</u>
  - Probabilistic public key bit encryption
  - Xor homomorphic

# Robustness



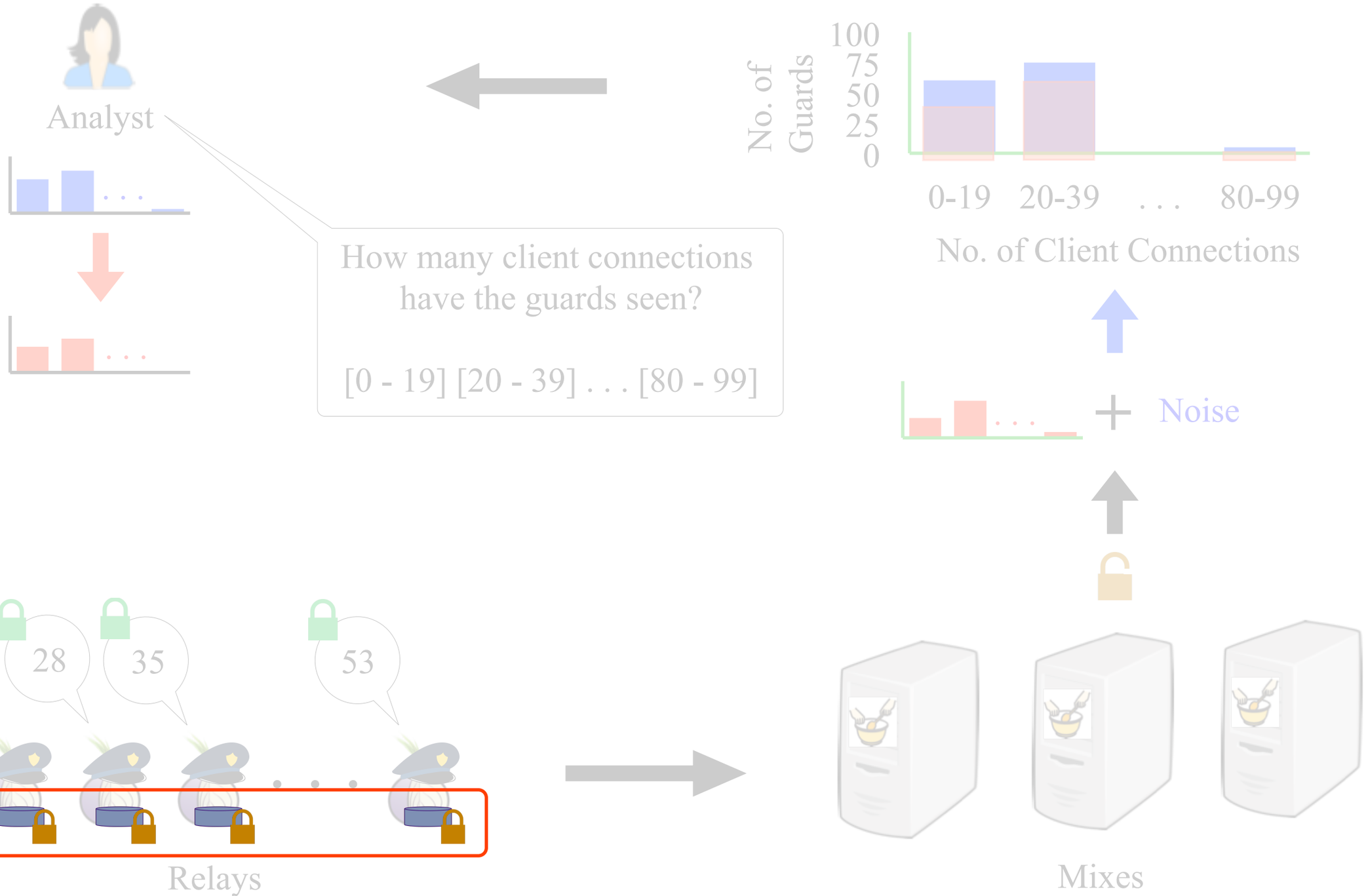

- Malicious relays may report erroneous data
- Historε's integrity guarantees:
  - GM encryption ensures that legitimate values are either *0* or *1*
  - Each relay can contribute at most *1* to each bin in its counter
  - Maximum influence is thus bounded by the number of malicious relays

# Robustness




- Malicious relays may report erroneous data
- Historε's integrity guarantees:
  - GM encryption ensures that legitimate values are either *0* or *1*
  - Each relay can contribute at most *1* to each bin in its counter
  - Maximum influence is thus bounded by the number of malicious relays

# Incrementing Oblivious Counters



# Incrementing Oblivious Counters


New Packet!

t = 0

- Initialize all but the first element to encryptions of *0*
- Initialize *low order* counter *t* to 0
- When a relay observes the "statistic of interest" (e.g. client connections)
  - right shift the encrypted *1* whenever *t* reaches the bin width

# Incrementing Oblivious Counters


New Packet!

t = 1

- Initialize all but the first element to encryptions of *0*
- Initialize *low order* counter *t* to 0
- When a relay observes the "statistic of interest" (e.g. client connections)
  - right shift the encrypted *1* whenever *t* reaches the bin width

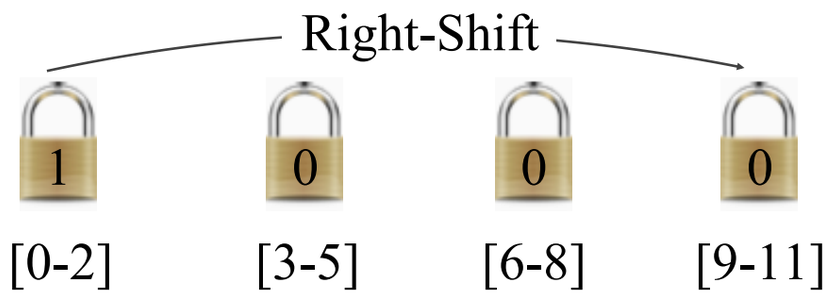
# Incrementing Oblivious Counters


New Packet!

t = 2

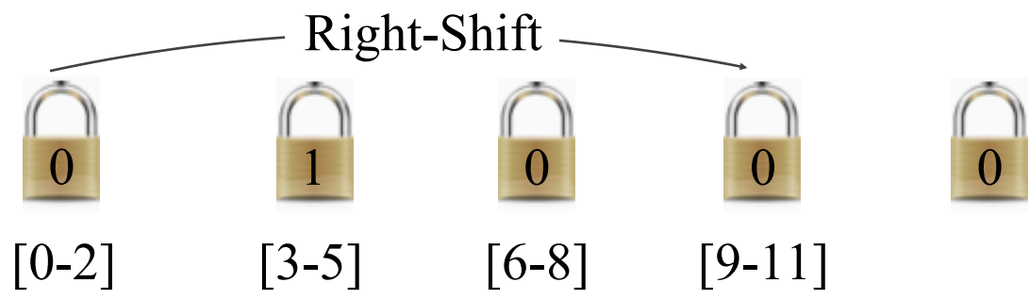
- Initialize all but the first element to encryptions of *0*
- Initialize *low order* counter *t* to 0
- When a relay observes the "statistic of interest" (e.g. client connections)
  - right shift the encrypted *1* whenever *t* reaches the bin width

# Incrementing Oblivious Counters



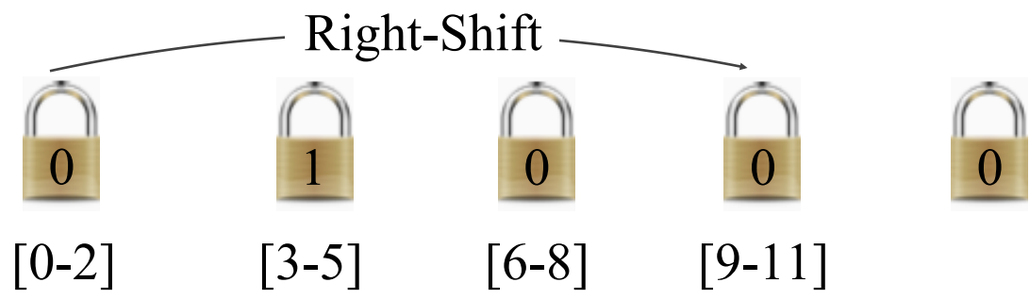

- Initialize all but the first element to encryptions of *0*
- Initialize *low order* counter *t* to 0
- When a relay observes the "statistic of interest" (e.g. client connections)
  - right shift the encrypted *1* whenever *t* reaches the bin width

# Incrementing Oblivious Counters




- Initialize all but the first element to encryptions of *0*
- Initialize *low order* counter *t* to 0
- When a relay observes the "statistic of interest" (e.g. client connections)
  - right shift the encrypted *1* whenever *t* reaches the bin width

# Incrementing Oblivious Counters

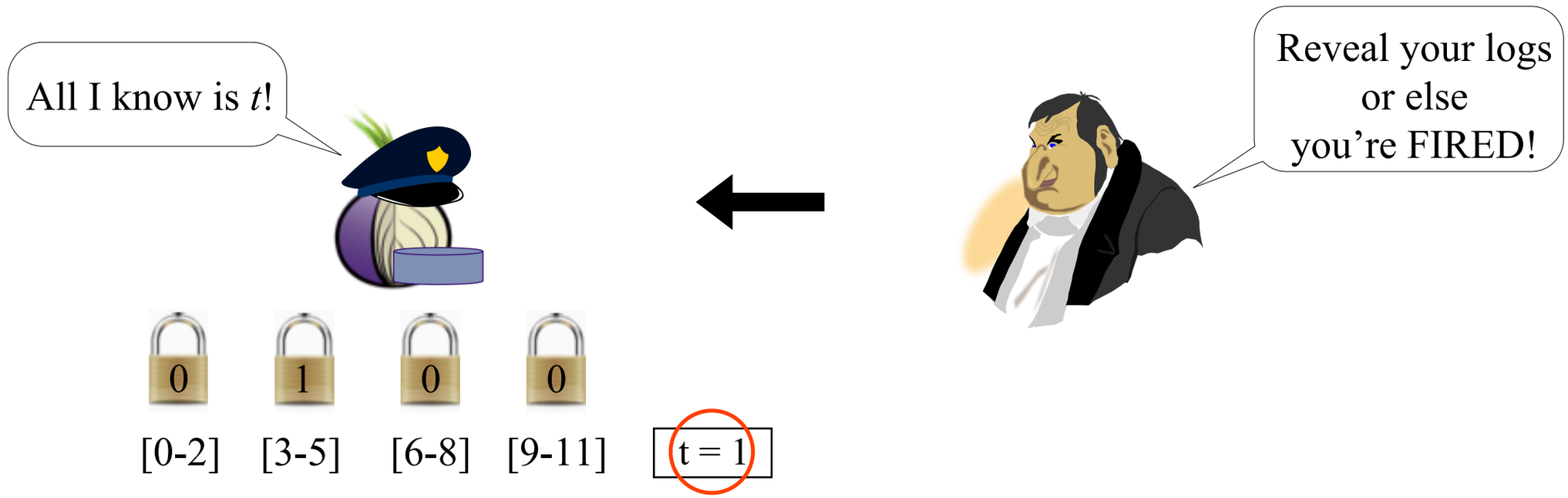



- Initialize all but the first element to encryptions of *0*
- Initialize *low order* counter *t* to 0
- When a relay observes the "statistic of interest" (e.g. client connections)
  - right shift the encrypted *1* whenever *t* reaches the bin width
  - reset counter *t* to 0

# Incrementing Oblivious Counters

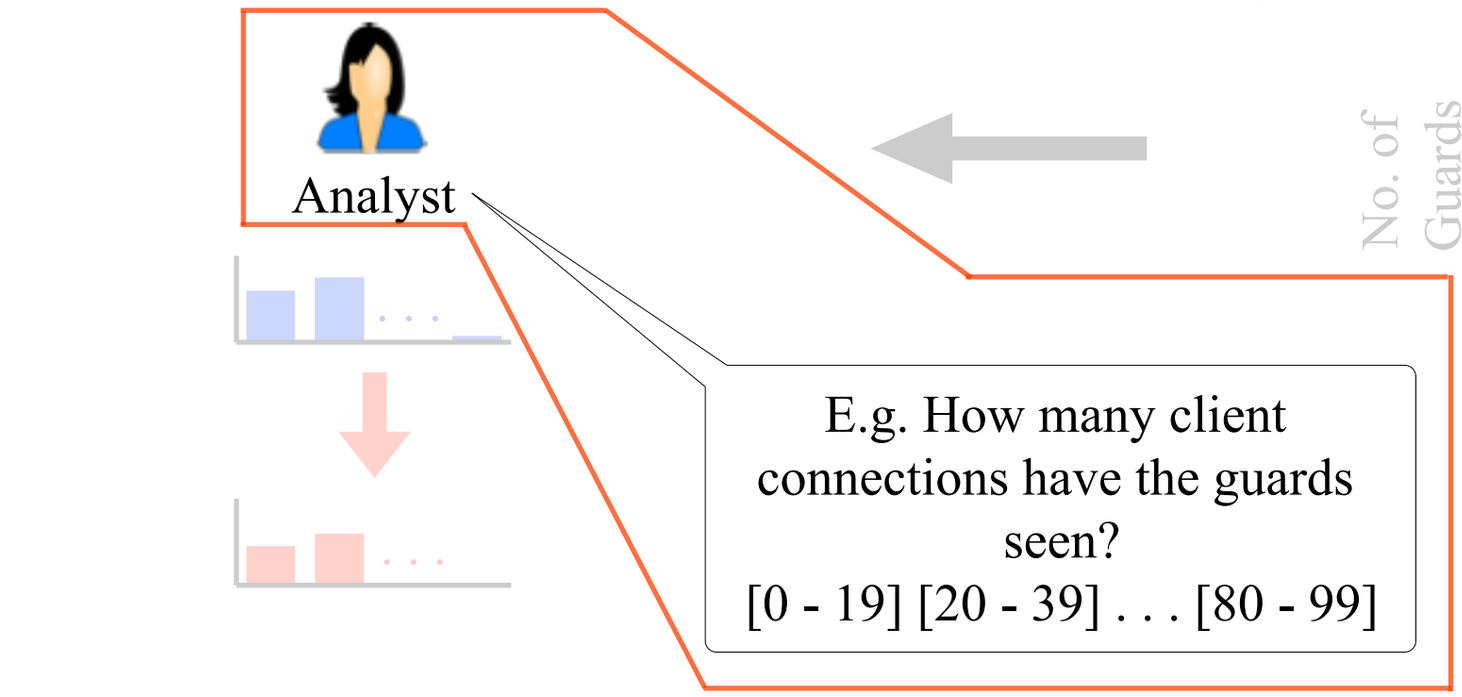

- Initialize all but the first element to encryptions of *0*
- Initialize *low order* counter *t* to 0
- When a relay observes the "statistic of interest" (e.g. client connections)
  - right shift the encrypted *1* whenever *t* reaches the bin width
  - reset counter *t* to 0

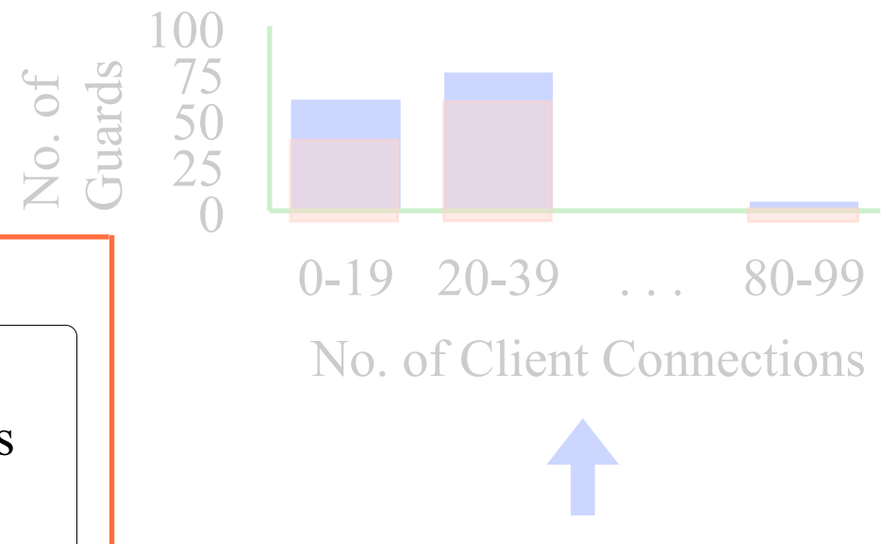
# Compulsion Attack



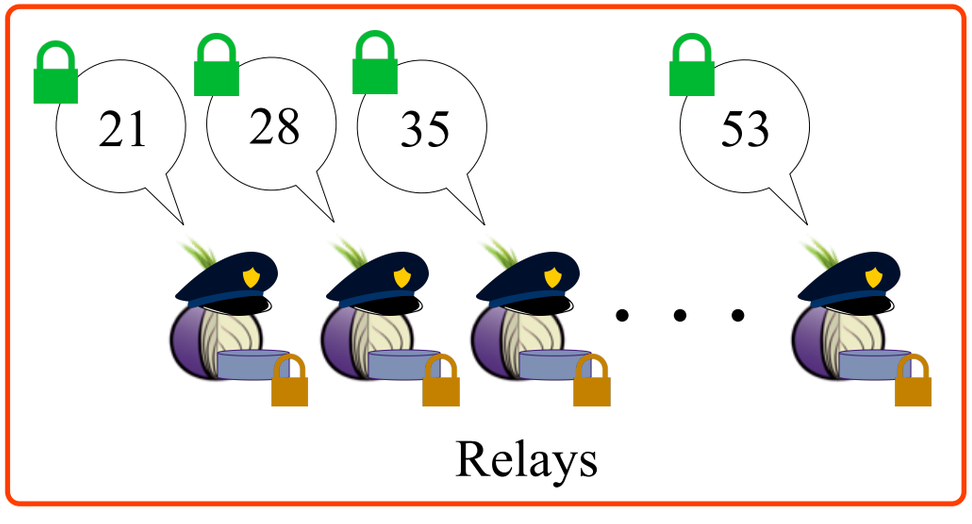

- *Oblivious counters* minimize the amount of information leaked
  - Relays cannot decrypt the counters on their own
  - Leaks counter *t*

# Relays


Analyst
E.g. How many client connections have the guards seen?
[0 - 19] [20 - 39] . . . [80 - 99]


No. of Guards
100
75
50
25
0
0-19
20-39
. . .
80-99
No. of Client Connections


+ Noise

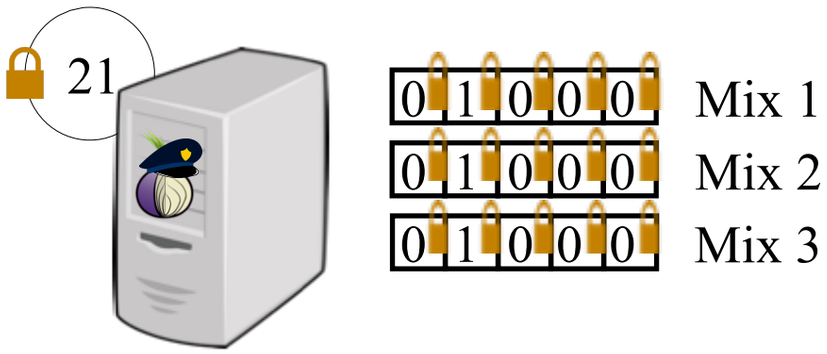

21
28
35
53
Relays


Mixes

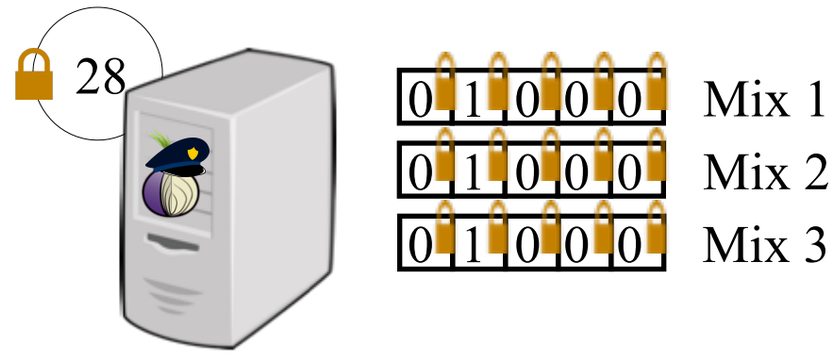
# Relays

Analyst

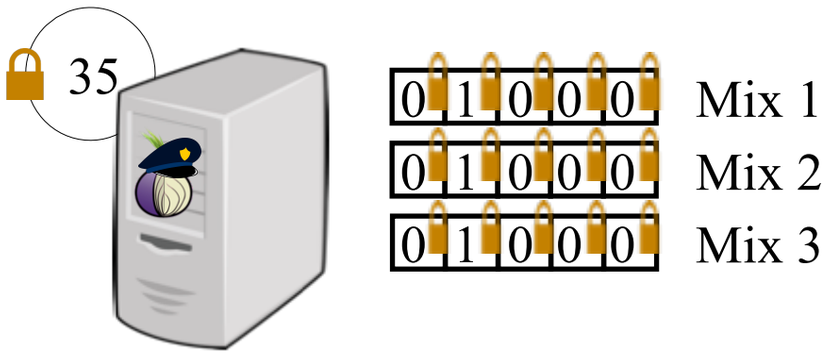
Q: How many client connections have the guards seen?
[0 - 19] [20 - 39] [40 - 59] [60 - 79] [80 - 99]



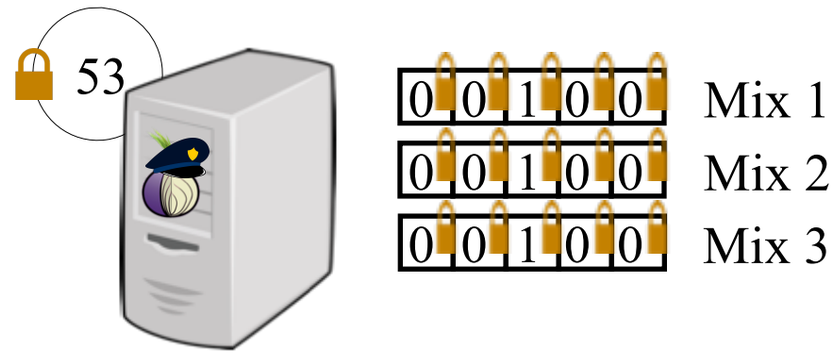

$R_1$




$R_2$




$R_3$

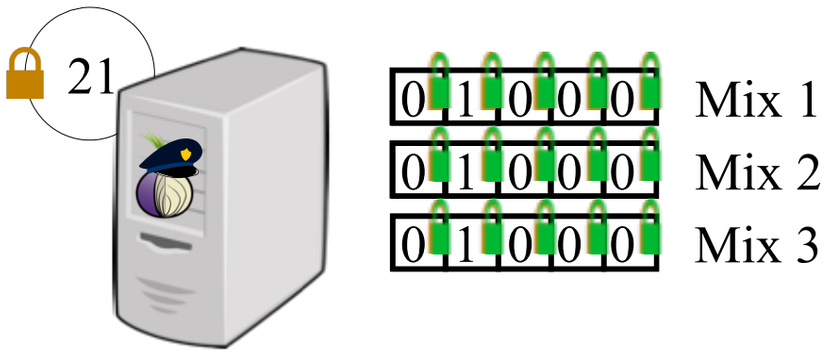



$R_4$

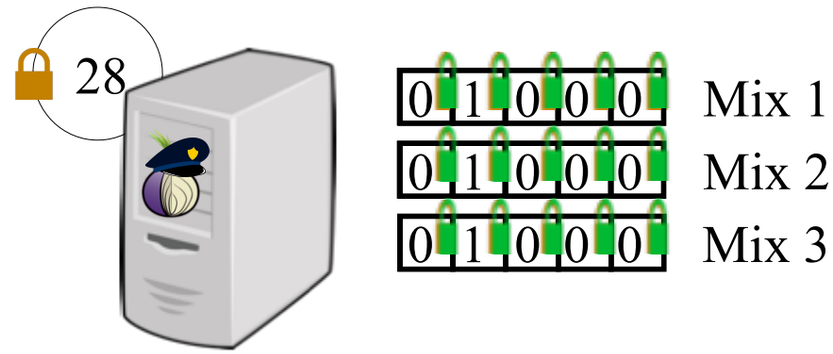
# Relays

Analyst

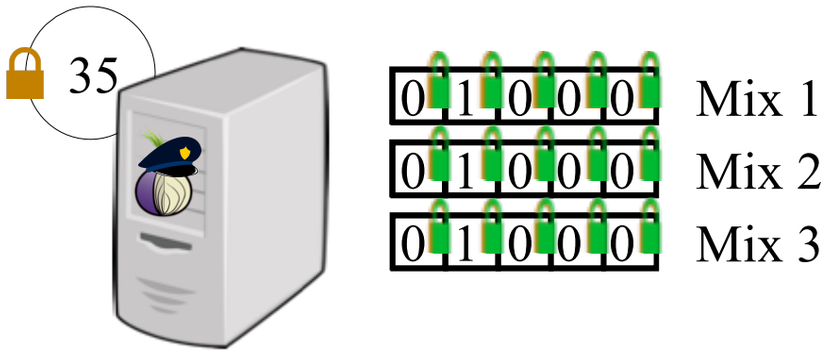
Q: How many client connections have the guards seen?
[0 - 19] [20 - 39] [40 - 59] [60 - 79] [80 - 99]



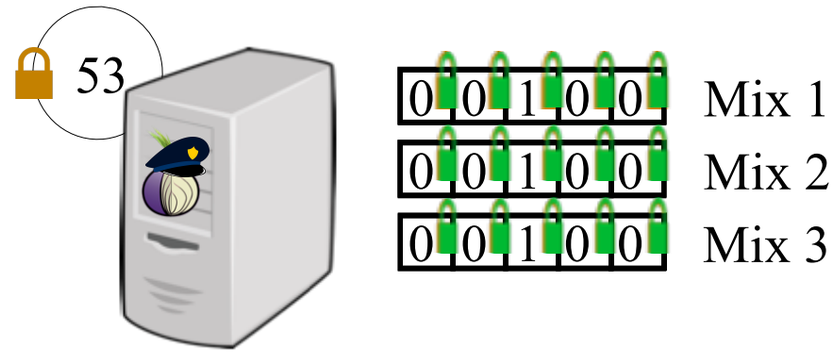

$R_1$




$R_2$



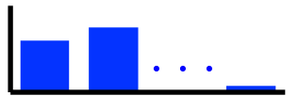

$R_3$




$R_4$

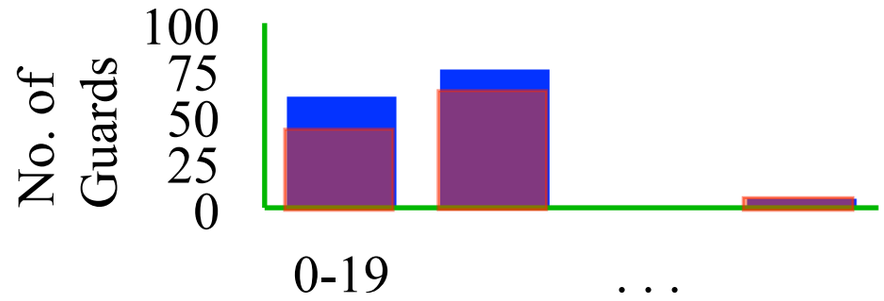
# Mix & Analyst

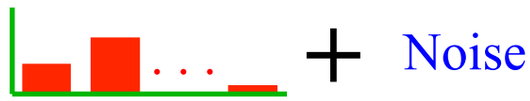
Analyst



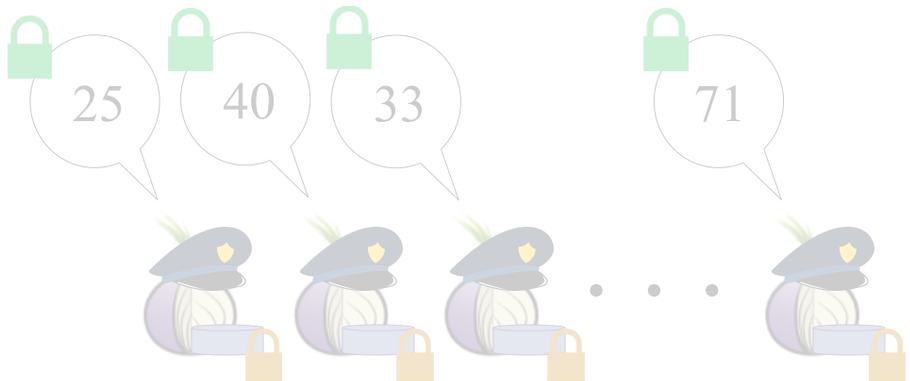
How many client connections have the guards seen?

[0 - 19] [20 - 39] . . . [130 - 149]



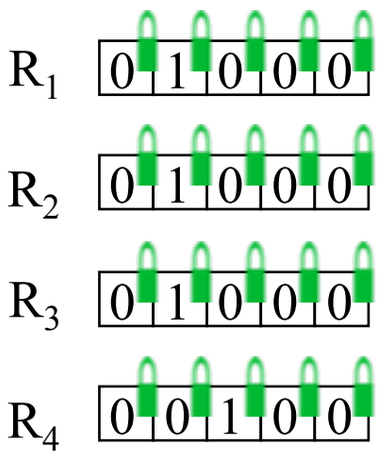




Mixes



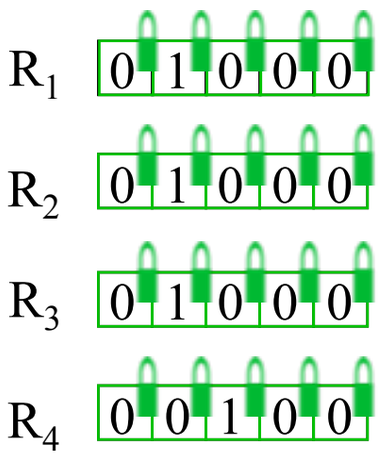

Relays

# Mix

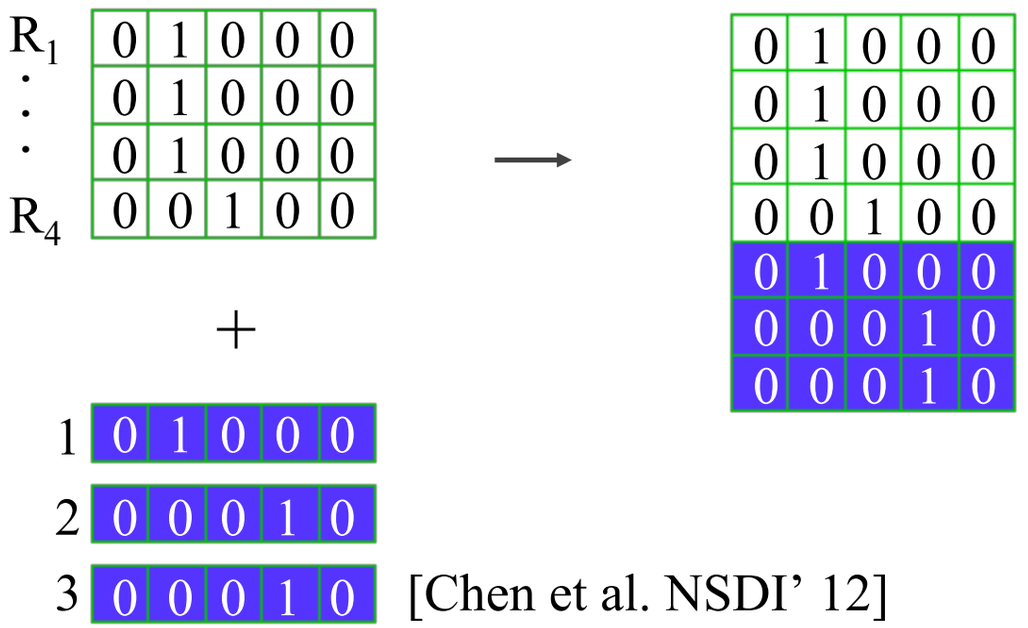


# Mix

## Decrypt




## Add Noise




[Chen et al. NSDI' 12]

# Mix

## Decrypt

| | | | | | |
|---|---|---|---|---|---|
| $R_1$ | 0 | 1 | 0 | 0 | 0 |
| $R_2$ | 0 | 1 | 0 | 0 | 0 |
| $R_3$ | 0 | 1 | 0 | 0 | 0 |
| $R_4$ | 0 | 0 | 1 | 0 | 0 |

→

## Add Noise

| | | | | | |
|---|---|---|---|---|---|
| $R_1$ | 0 | 1 | 0 | 0 | 0 |
| ⋮ | 0 | 1 | 0 | 0 | 0 |
| | 0 | 1 | 0 | 0 | 0 |
| $R_4$ | 0 | 0 | 1 | 0 | 0 |

+

| | | | | | |
|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 0 | 1 | 0 |

→

## Shuffle

Column-wise ↓ ↑ Column-wise

| | | | | |
|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |

[Chen et al. NSDI' 12]

# Analyst

De-obfuscate | Aggregate | Reduce Noise

Mix

| | | | | |
|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |

→

| | | | | |
|---|---|---|---|---|
| 0 | 4 | 1 | 2 | 0 |

→

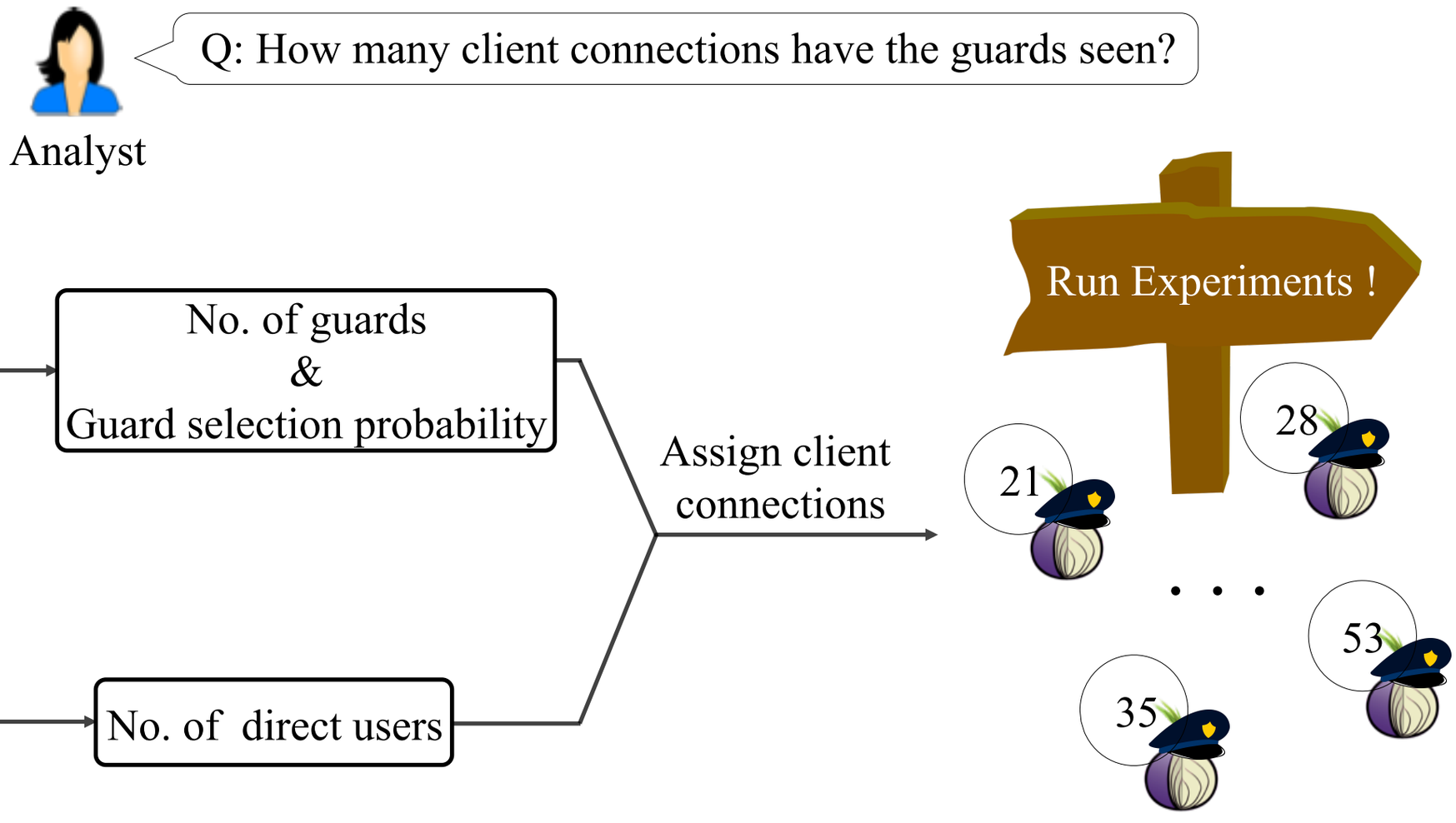
| | | | | |
|---|---|---|---|---|
| 0 | 4 | 1 | 2 | 0 |

$-\frac{3}{2}$

$=$

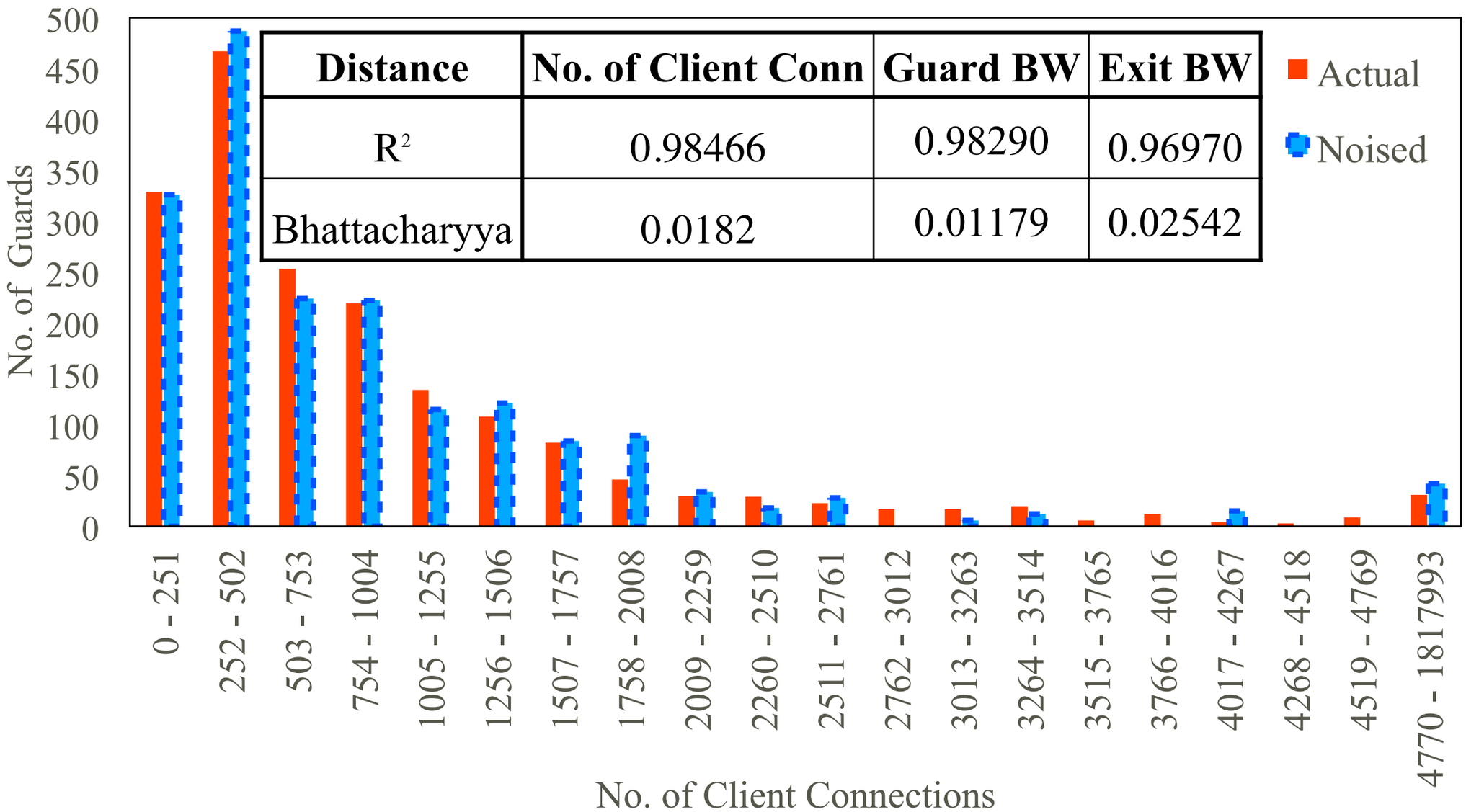
| | | | | |
|---|---|---|---|---|
| 0 | 2 | 0 | 0 | 0 |

# Evaluating HisTorϵ



# Accuracy

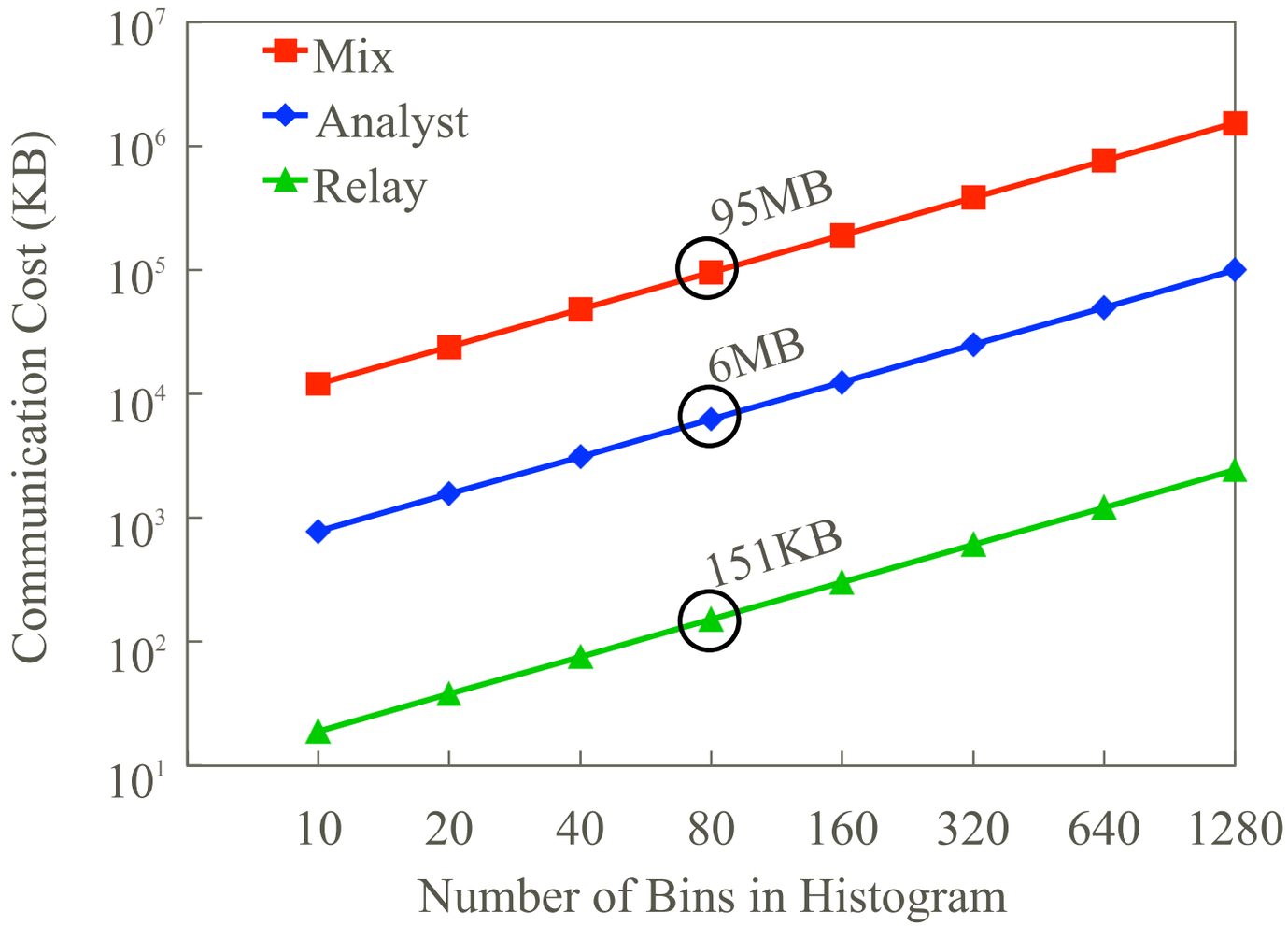
- *No. of client connections as seen by guards*




| Distance | No. of Client Conn | Guard BW | Exit BW |
|---|---|---|---|
| $R^2$ | 0.98466 | 0.98290 | 0.96970 |
| Bhattacharyya | 0.0182 | 0.01179 | 0.02542 |

# Communication Cost

- HisTorε incurs very little bandwidth overhead




Roughly
42 Bps
for each relay

# Summary

- **His**togram query for **Tor** with (**ε**, $\delta$) differential privacy
- Provides strong integrity guarantees
  - Maximum influence is bounded by the number of malicious relays
- Provides strong privacy guarantees
  - Resistant to “compulsion attacks”
- Highly accurate
- Incurs low overheads

HisTorε: Differentially Private and Robust Statistics Collection for Tor

Akshaya Mani, Micah Sherr  Georgetown University

# Summary

- **His**togram query for **Tor** with (**ε**, $\delta$) differential privacy
- Provides strong integrity guarantees
  - Maximum influence is bounded by the number of malicious relays
- Provides strong privacy guarantees
  - Resistant to “compulsion attacks”
- Highly accurate
- Incurs low overheads

HisTorε: Differentially Private and Robust Statistics Collection for Tor

Akshaya Mani, Micah Sherr  Georgetown University