U.S. NAVAL RESEARCH LABORATORY

PrivCount: A Distributed System for Safely Measuring Tor

Rob Jansen
U.S. Naval Research Laboratory
Center for High Assurance Computer Systems

Invited Talk, October 4th, 2016
University of Oregon
Department of Computer and Information Science

U.S. NAVAL RESEARCH LABORATORY

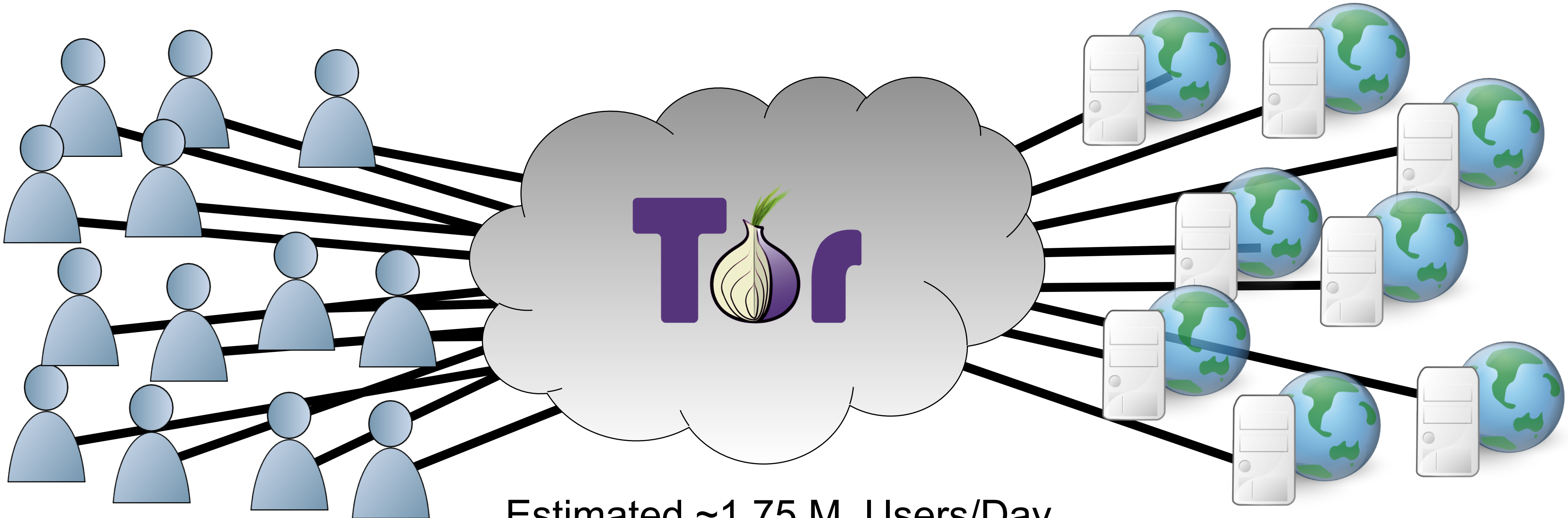
PrivCount: A Distributed System for Safely Measuring Tor

"Safely Measuring Tor", Rob Jansen and Aaron Johnson,
In the *Proceedings of the 23rd ACM Conference on Computer and Communication Security* (CCS 2016).

Rob Jansen
U.S. Naval Research Laboratory
Center for High Assurance Computer Systems

Invited Talk, October 4th, 2016
University of Oregon
Department of Computer and Information Science

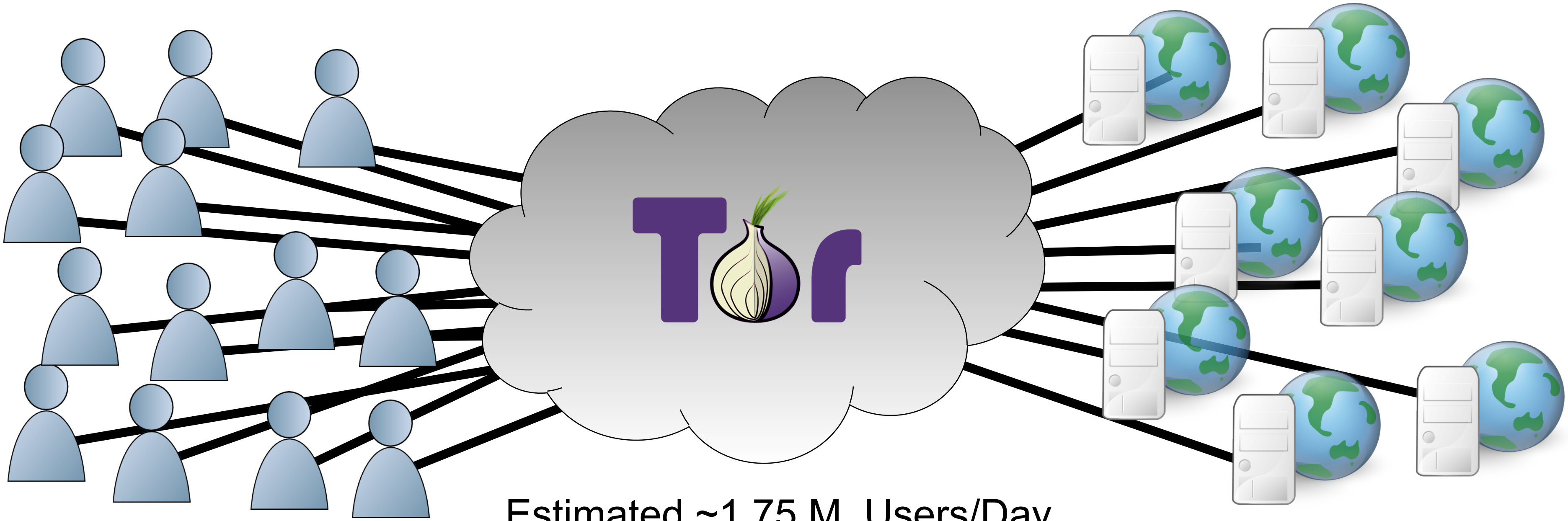
Talk Overview



Estimated ~1.75 M. Users/Day
(metrics.torproject.org)

Tor: an anonymous communication, censorship resistant, privacy-enhancing communication system

Talk Overview

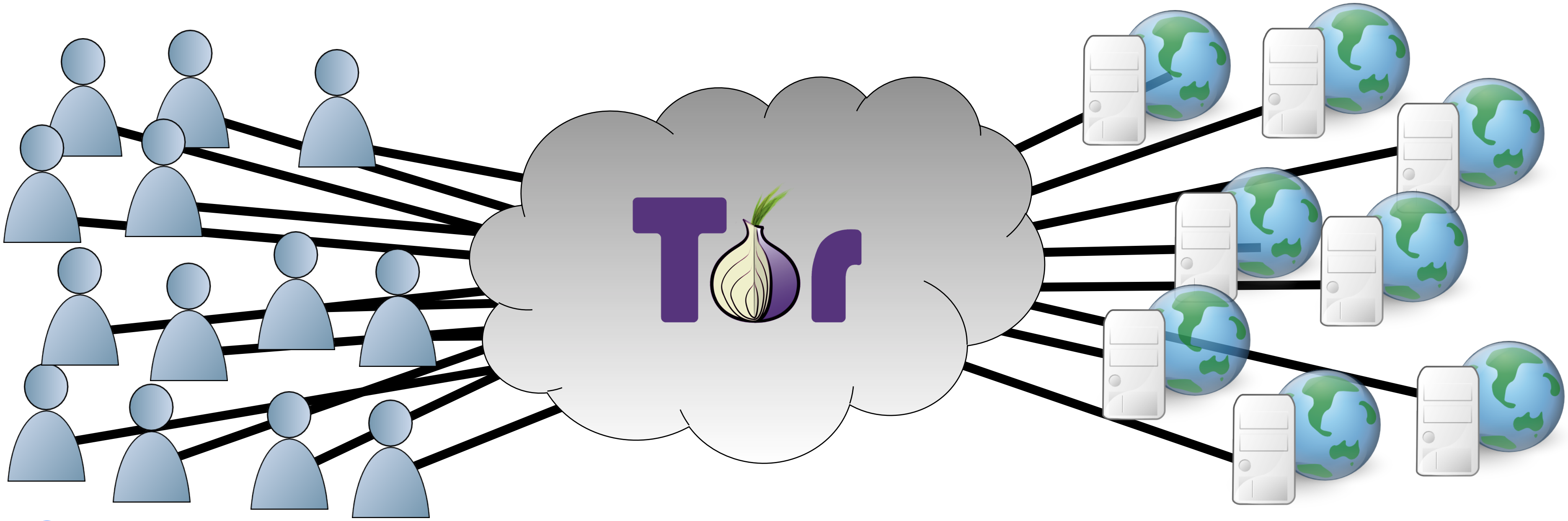


Estimated ~1.75 M. Users/Day
(metrics.torproject.org)

Tor: an anonymous communication, censorship resistant, privacy-enhancing communication system

- How is Tor being used?
- How is Tor being misused?
- How well is Tor performing?

Talk Overview



Objective:

- To gather Tor network usage statistics, safely

Approach:

- Use distributed measurement, secure multiparty computation, and differential privacy

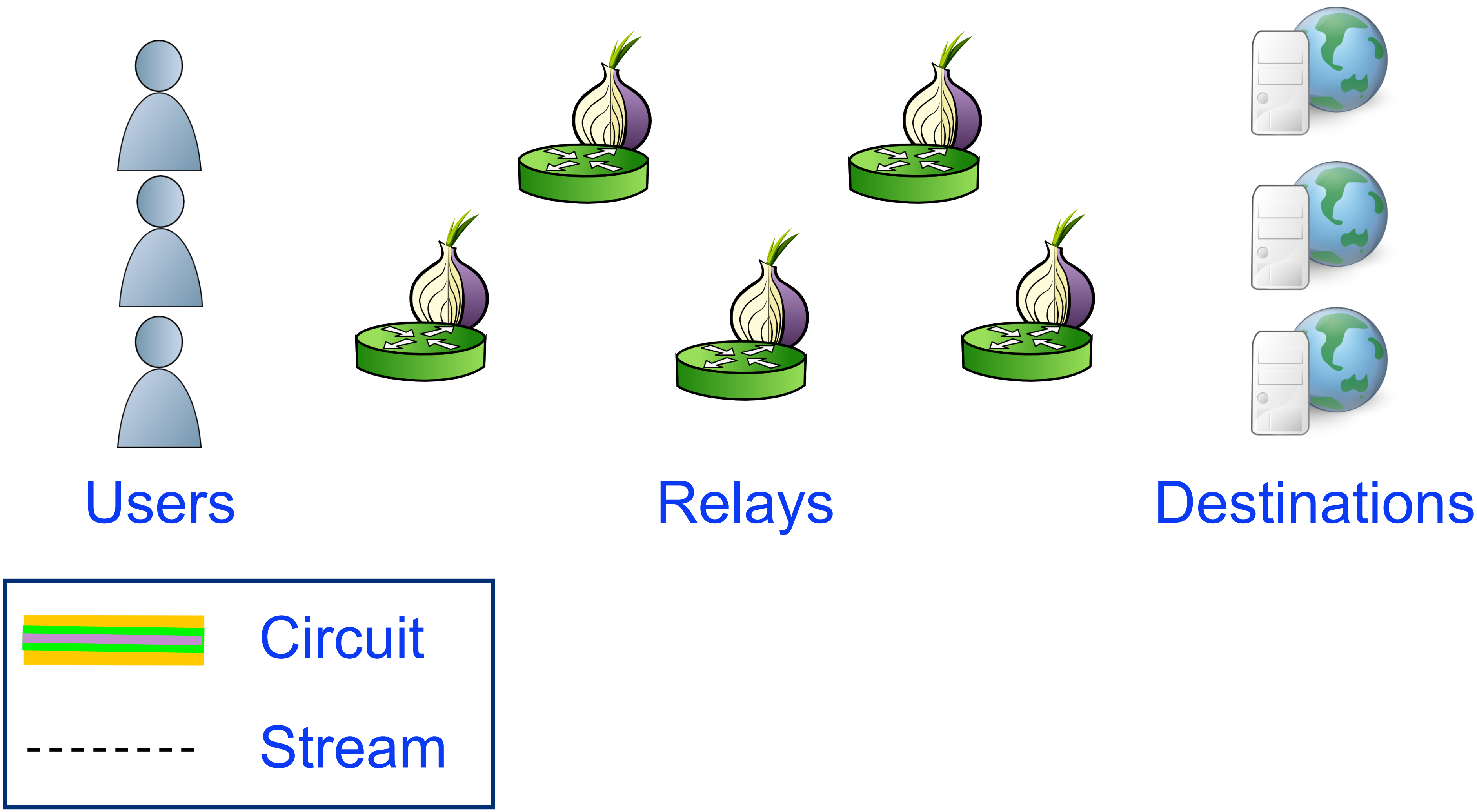
Benefits and Contributions:

- Understand/improve protocols, inform policy discussion
- Improve accuracy, privacy, and collect new statistics

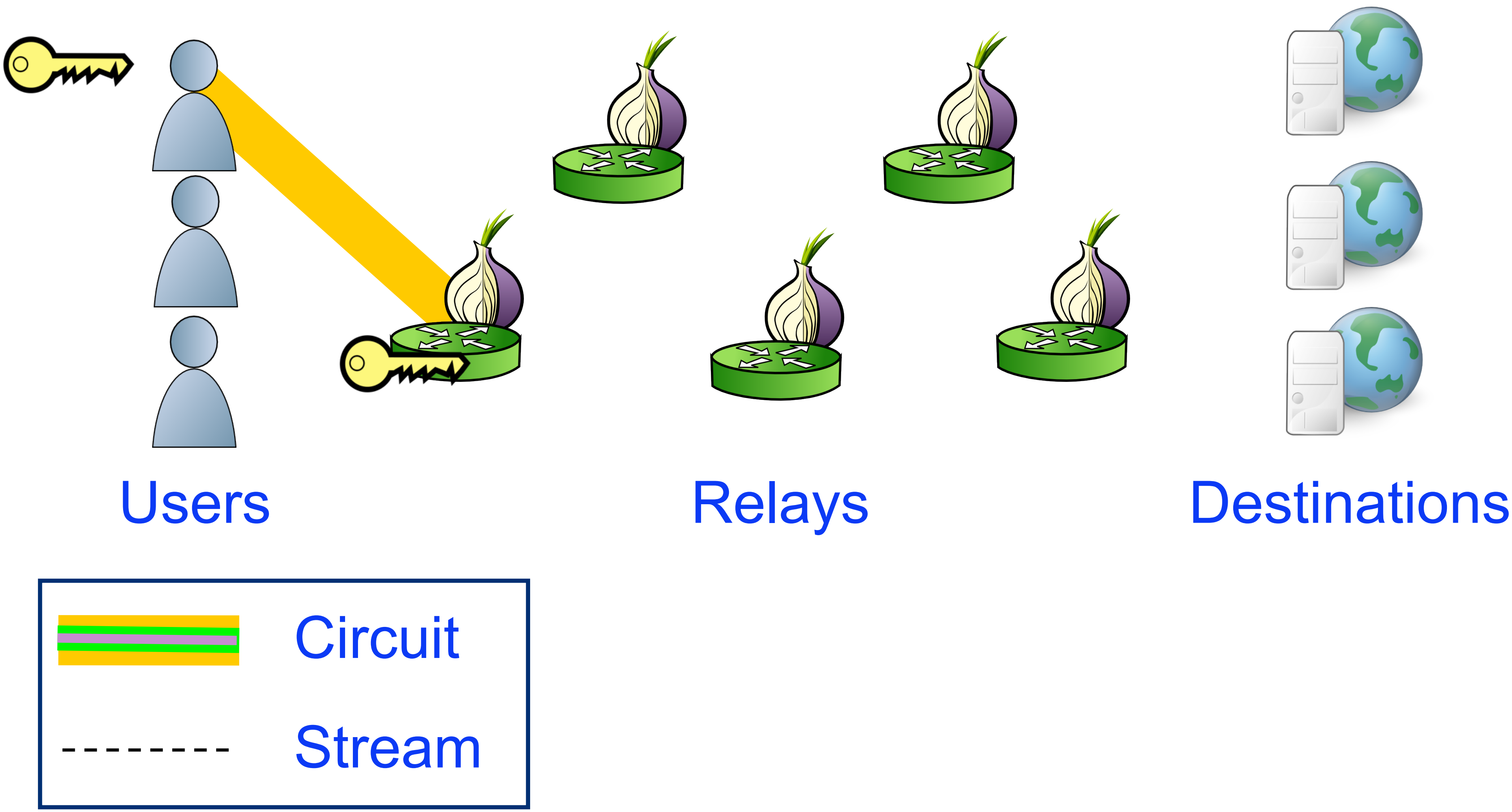
Background and Motivation

- How Tor works
- Why measurements are needed and what to measure
- Measurement challenges

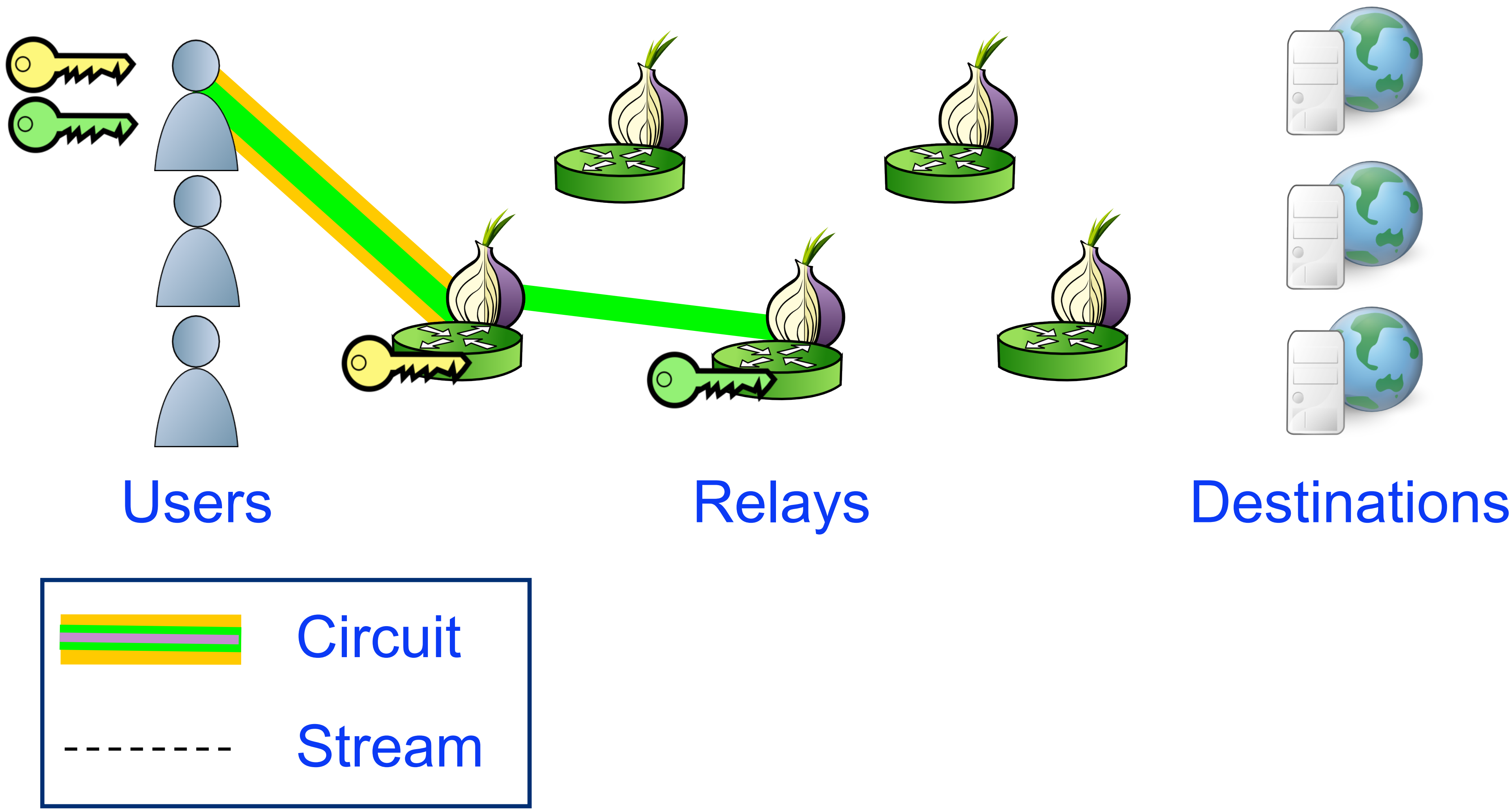
Background: Onion Routing



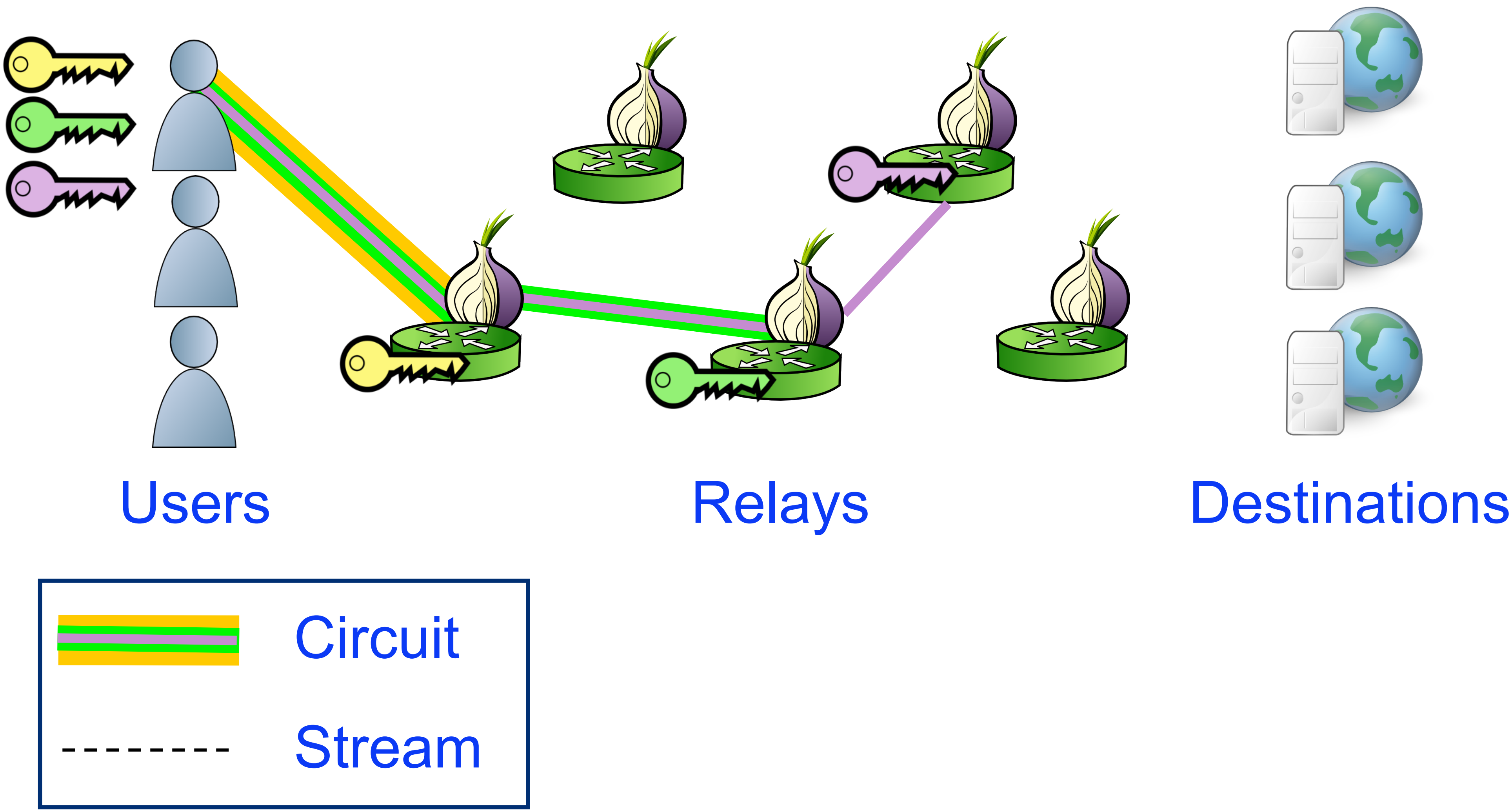
Background: Onion Routing



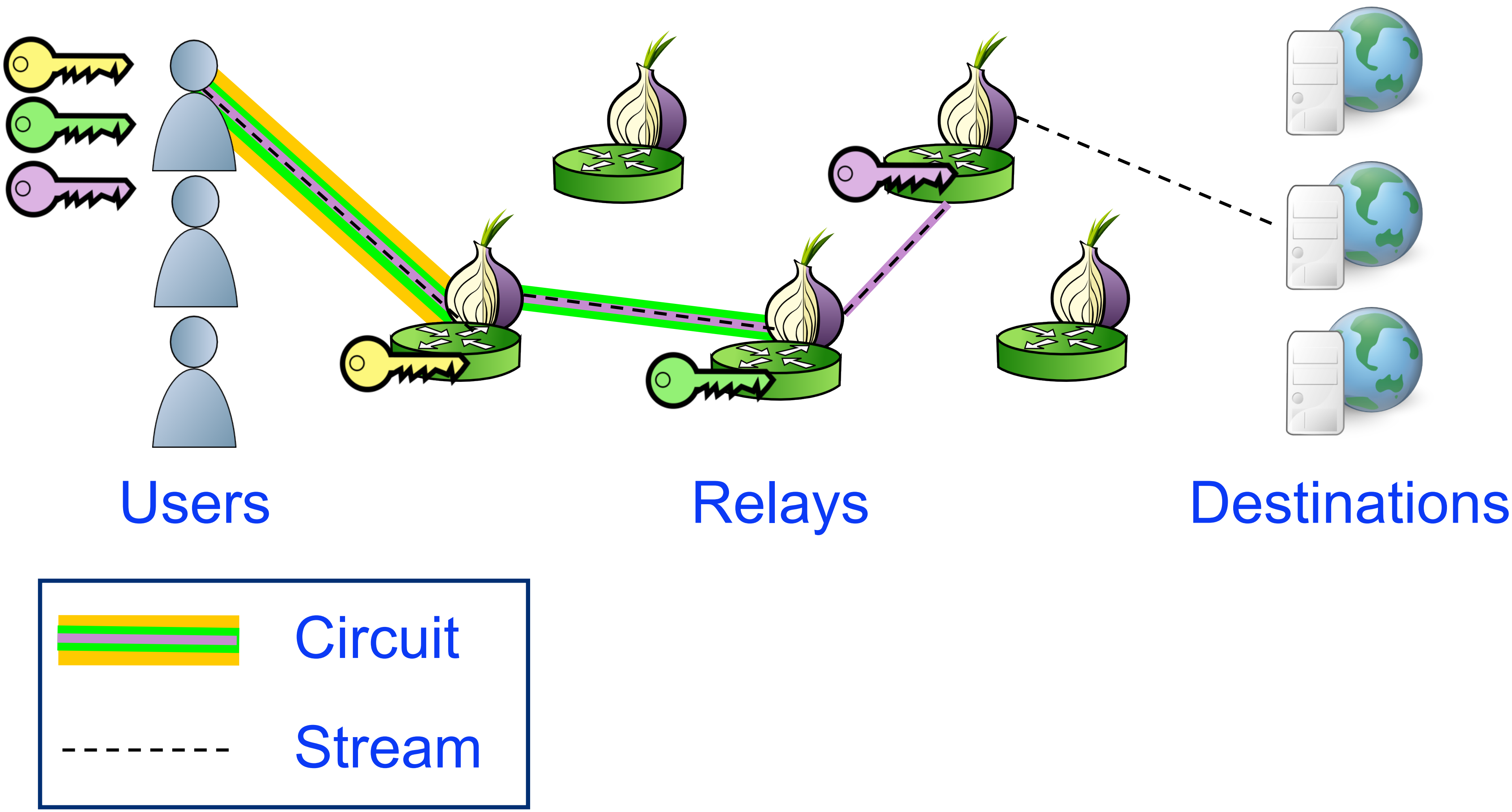
Background: Onion Routing



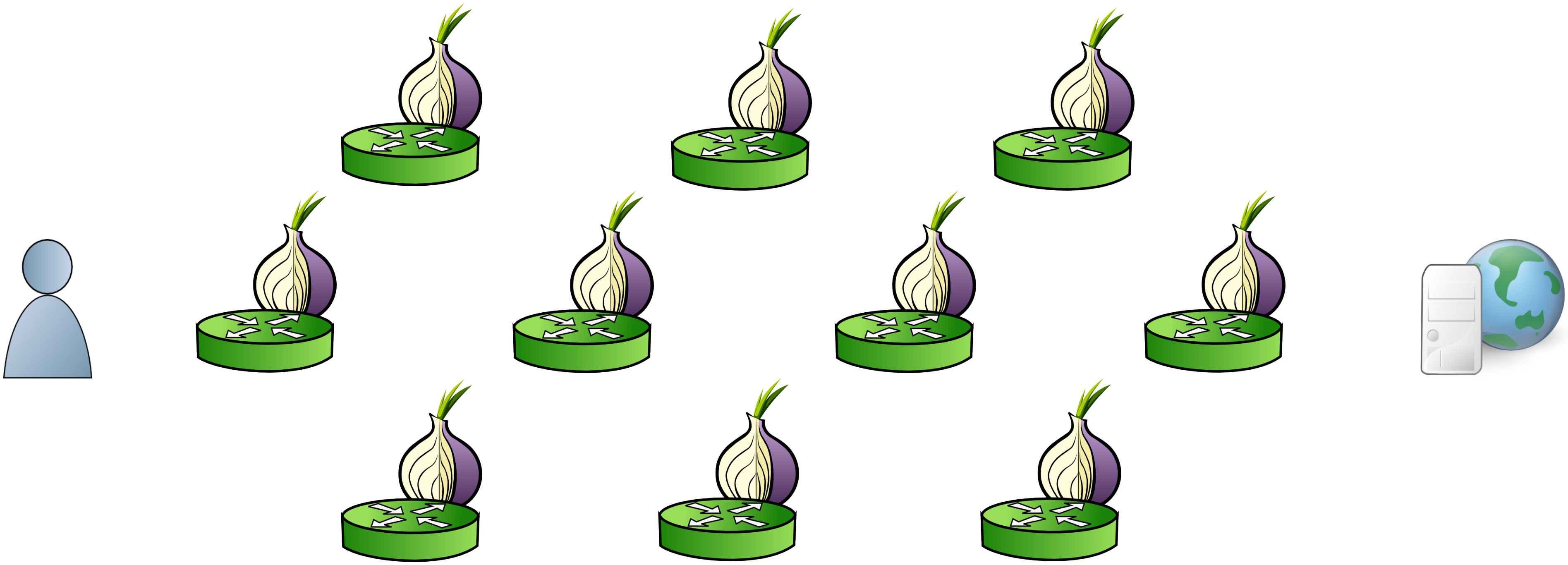
Background: Onion Routing



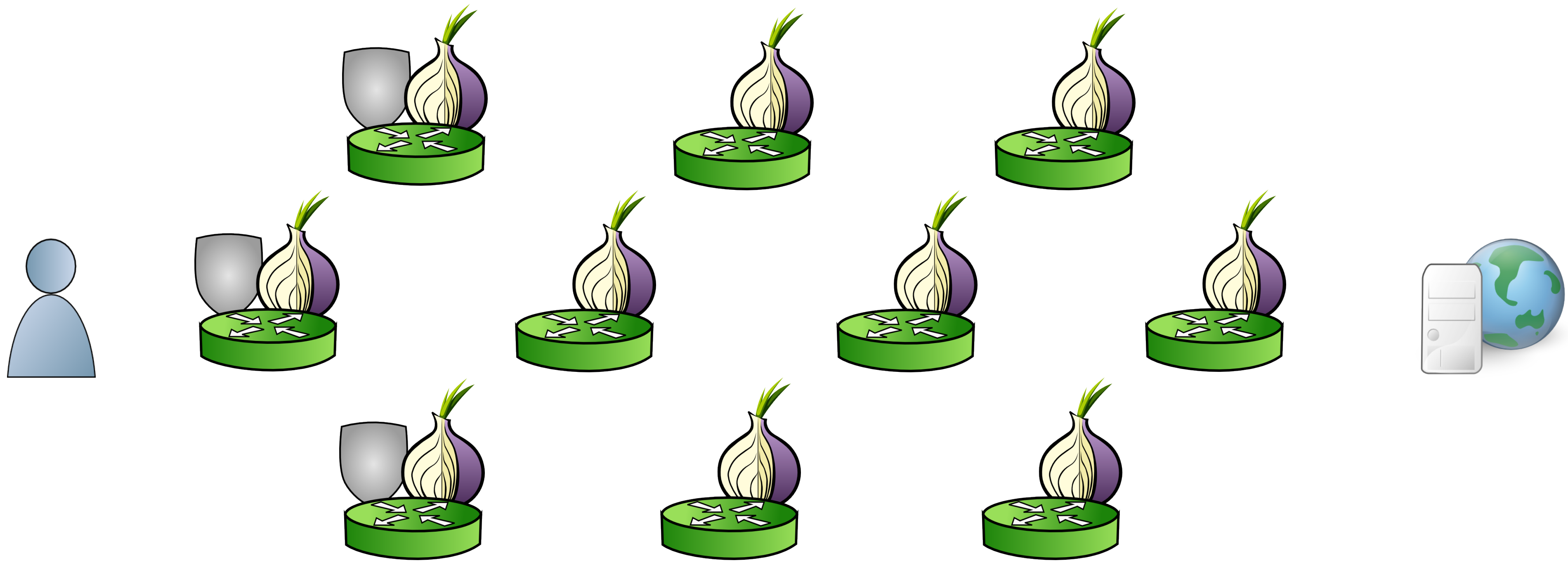
Background: Onion Routing



Background: Using Circuits

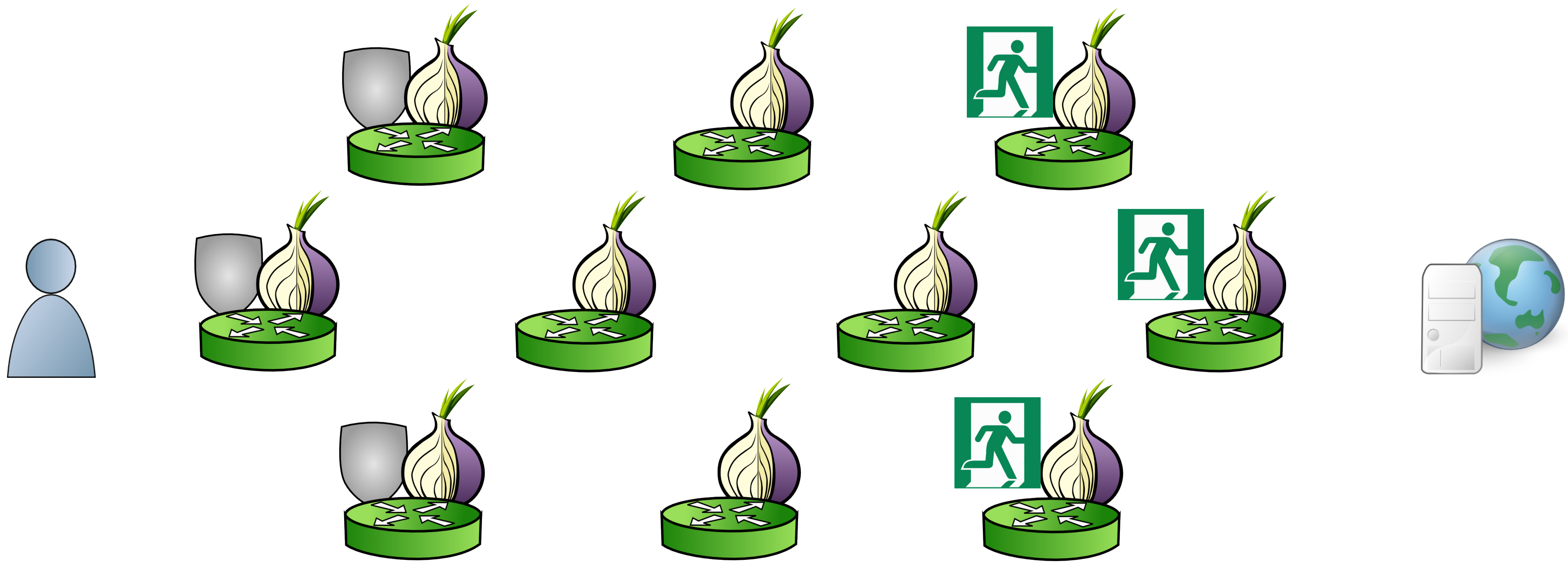


Background: Using Circuits



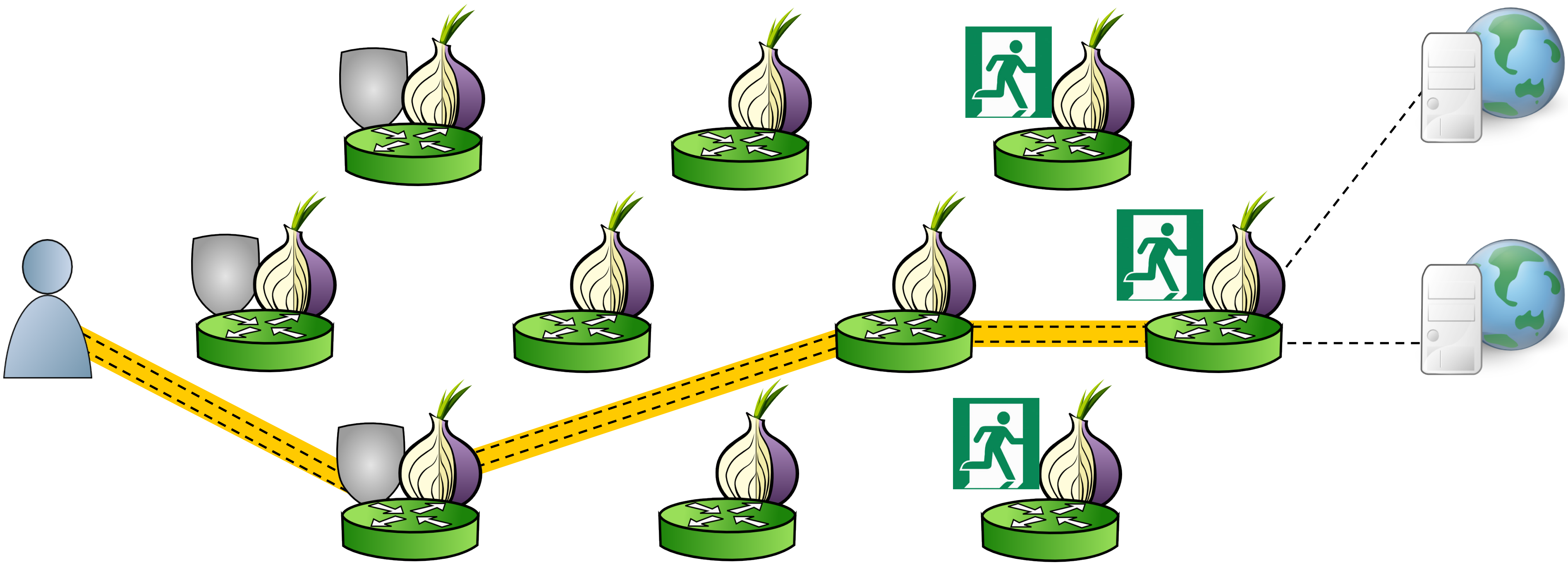
1. Clients begin all circuits with a selected guard

Background: Using Circuits



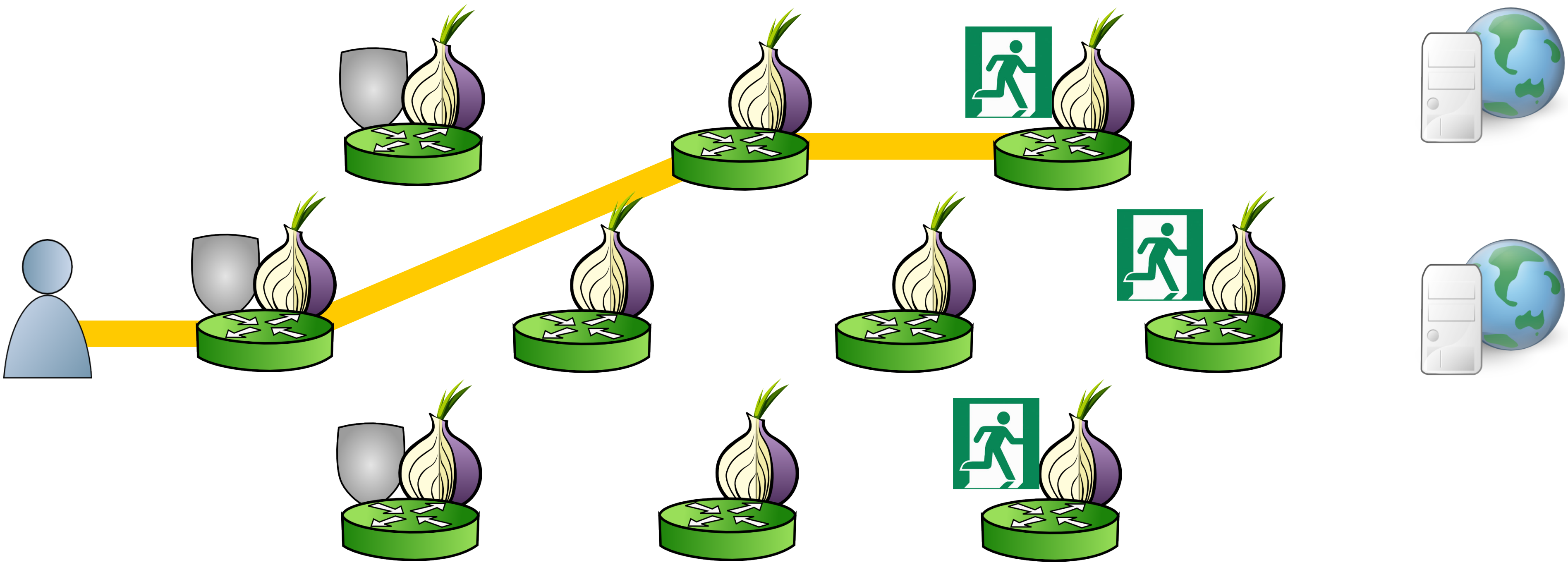
1. Clients begin all circuits with a selected guard
2. Relays define individual exit policies

Background: Using Circuits



1. Clients begin all circuits with a selected guard
2. Relays define individual exit policies
3. Clients multiplex streams over a circuit

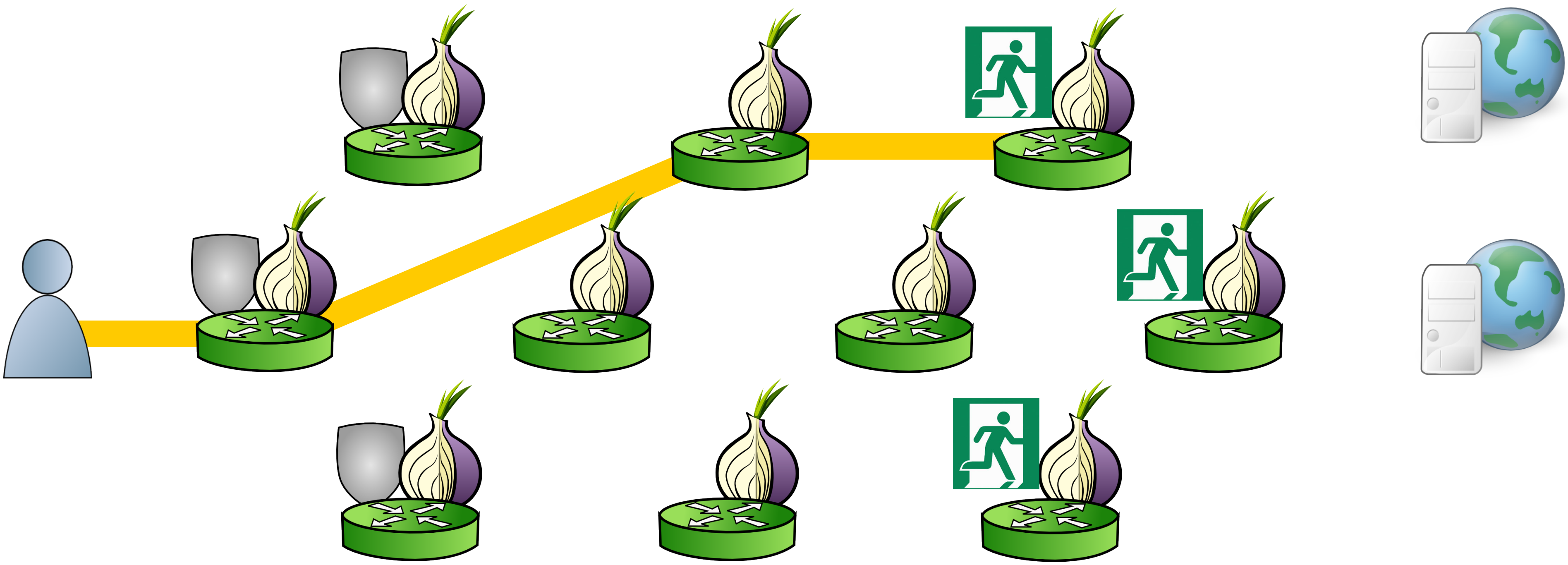
Background: Using Circuits



1. Clients begin all circuits with a selected guard
2. Relays define individual exit policies
3. Clients multiplex streams over a circuit
4. New circuits replace existing ones periodically

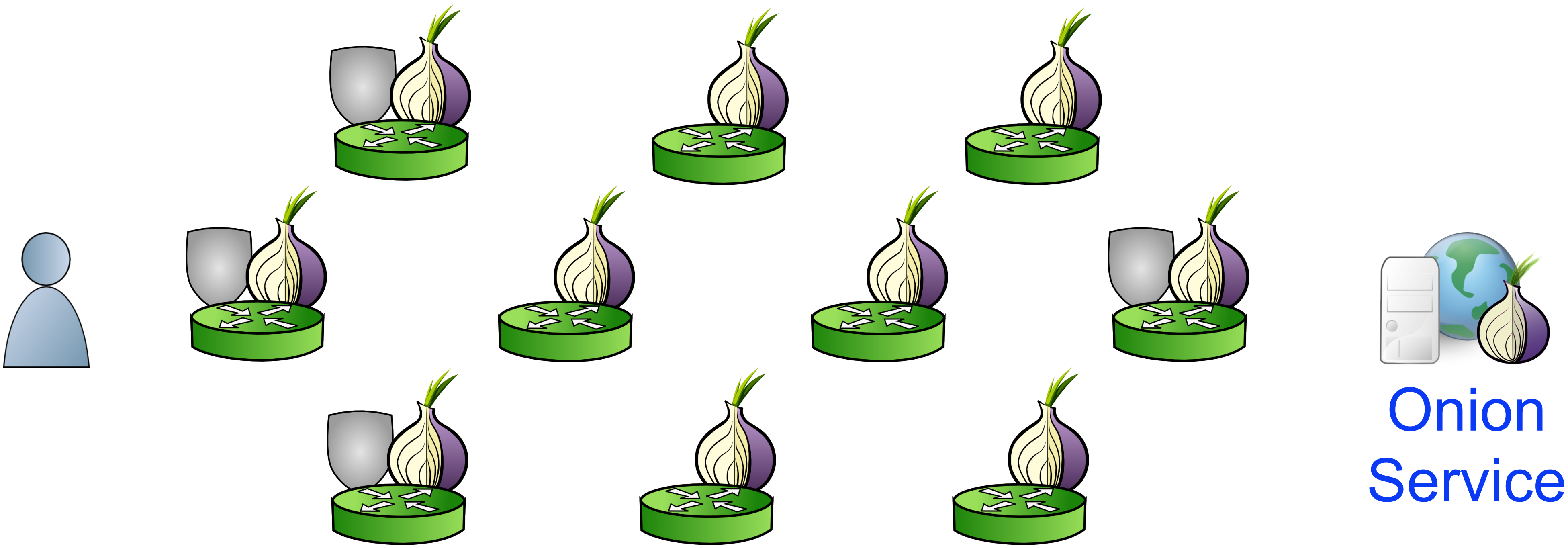
U.S. NAVAL RESEARCH LABORATORY

Background: Using Circuits



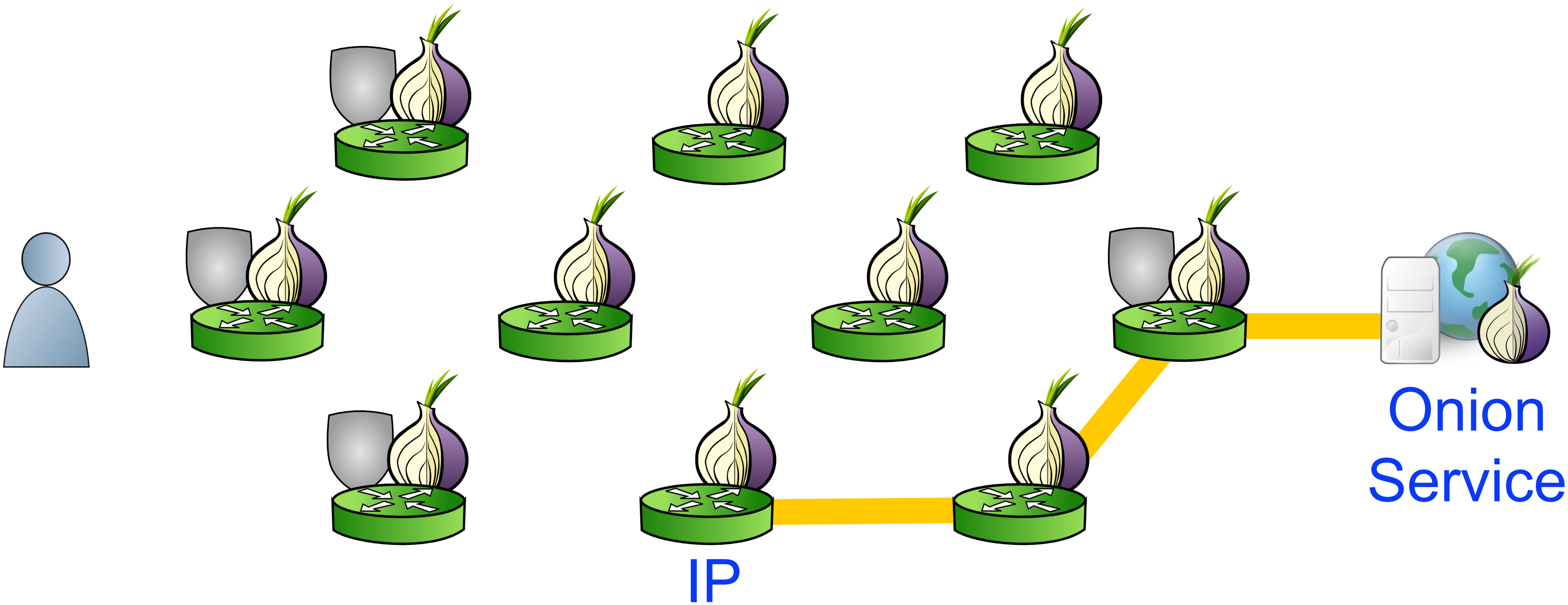
1. Clients begin all circuits with a selected guard
2. Relays define individual exit policies
3. Clients multiplex streams over a circuit
4. New circuits replace existing ones periodically
5. Clients randomly choose relays, weighted by bandwidth

Background: Onion Services



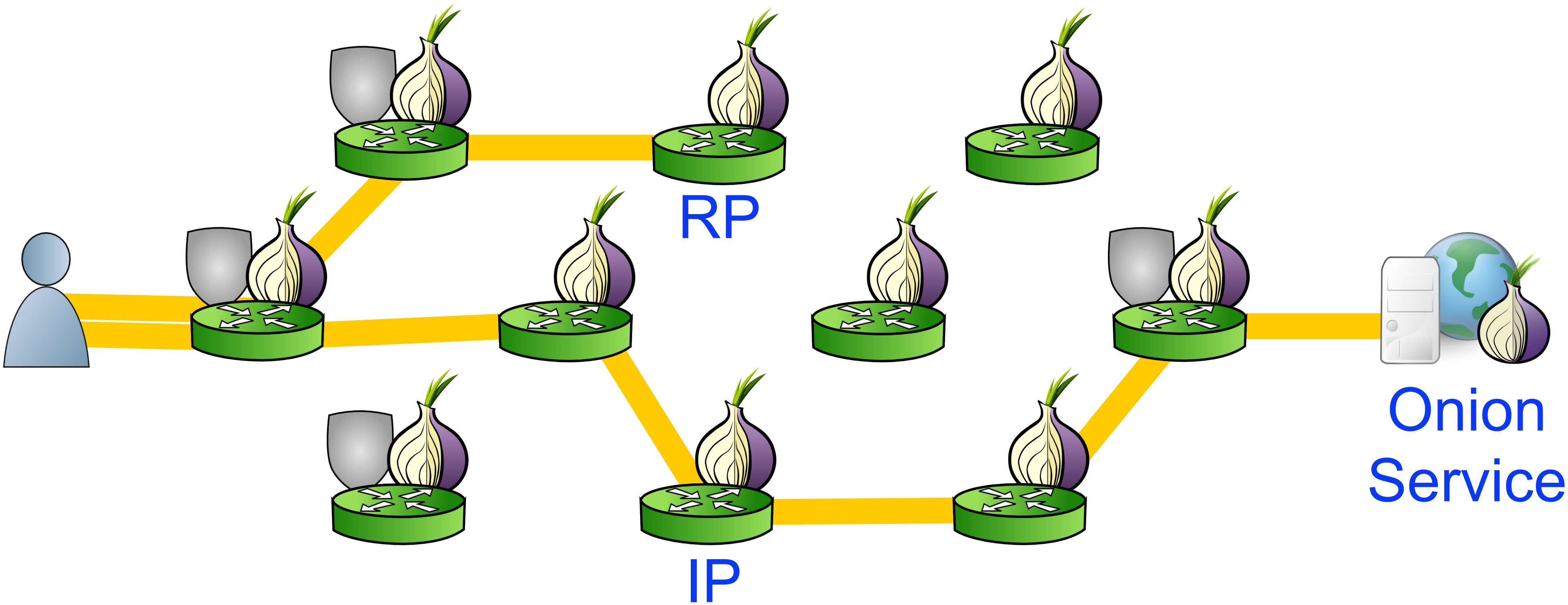
U.S. NAVAL RESEARCH LABORATORY

Background: Onion Services



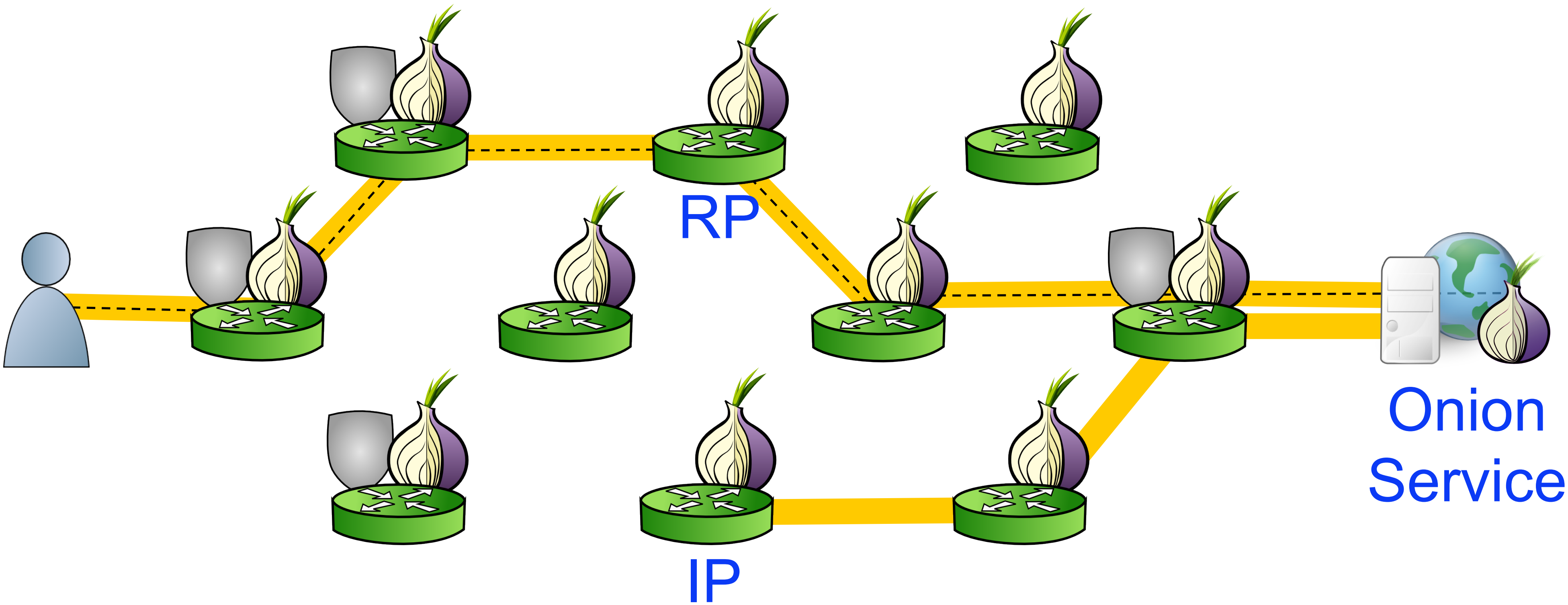
1. Onion services maintain circuits to introduction points (IPs)

Background: Onion Services



1. Onion services maintain circuits to introduction points (IPs)
2. User creates circuit to rendezvous point (RP) and IP and requests connection to RP

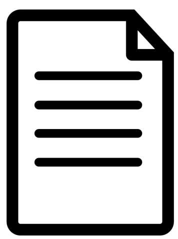
Background: Onion Services



1. Onion services maintain circuits to introduction points (IPs)
2. User creates circuit to rendezvous point (RP) and IP and requests connection to RP
3. Onion service connects to RP

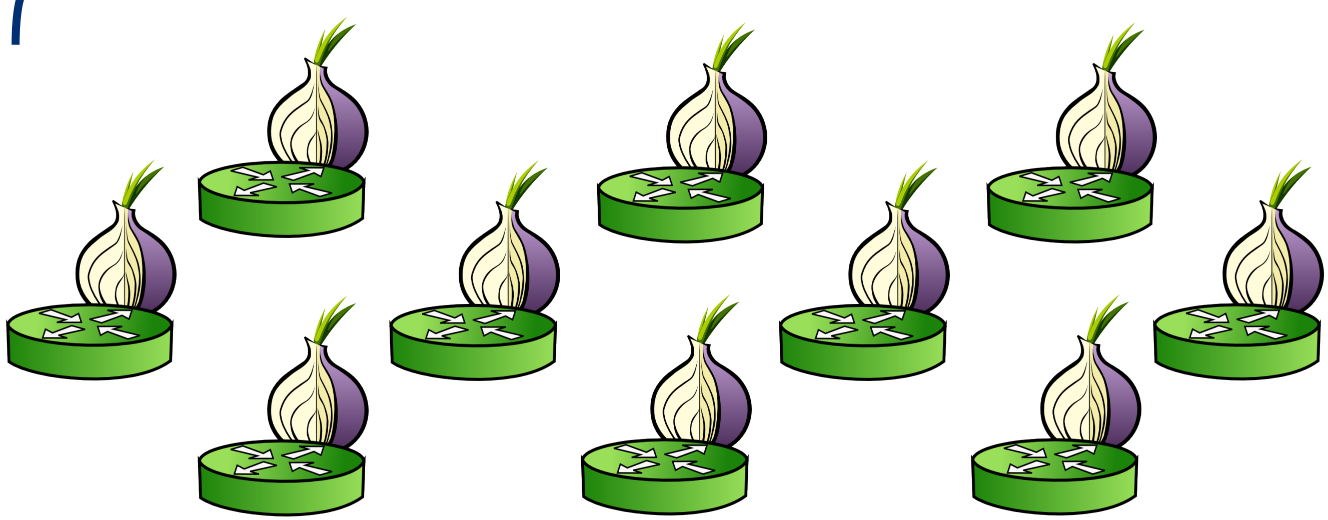
Background: Directory Authorities

Directory Authorities



Hourly network consensus by majority vote

- Relay info (IPs, pub keys, bandwidths, etc.)
- Parameters (performance thresholds, etc.)



Motivation: Why Measure Tor?

Why are Tor network measurements needed?

- To understand usage behaviors to focus effort and resources
- To understand network protocols and calibrate parameters
- To inform policy discussion

Motivation: Why Measure Tor?

Why are Tor network measurements needed?

- To understand usage behaviors to focus effort and resources
- To understand network protocols and calibrate parameters
- To inform policy discussion

"Tor metrics are the ammunition that lets Tor and other security advocates argue for a more private and secure Internet from a position of data, rather than just dogma or perspective."

– Bruce Schneier (2016-06-01)

Motivation: Measurement Challenges

Tor METRICS

https://metrics.torproject.org

Some Existing Measurements

Data Published	Privacy Techniques	Unsafe	Inaccurate
Relay BW available	Test measurements		✖
Relay BW used	Aggregated ~ 4 hours	✖	
Total # daily users	Inferred (consensus fetches)		✖
# users per country	Aggregated ~ 24 hours, rounded, opt-in	✖	
Exit traffic per port	Aggregated ~ 24 hours, opt-in	✖	

U.S. NAVAL RESEARCH LABORATORY

Motivation: Measurement Challenges

Tor METRICS

https://metrics.torproject.org

Safety concerns:

- Per-relay outputs
- Data stored locally
- No privacy proofs

Some Existing Measurements

Data Published	Privacy Techniques	Unsafe	Inaccurate
Relay BW available	Test measurements		✖
Relay BW used	Aggregated ~ 4 hours	✖	
Total # daily users	Inferred (consensus fetches)		✖
# users per country	Aggregated ~ 24 hours, rounded, opt-in	✖	
Exit traffic per port	Aggregated ~ 24 hours, opt-in	✖	

Motivation: Measurement Challenges

Tor METRICS

https://metrics.torproject.org

Accuracy concerns:
- Per-relay noise
- Opt-in and inconsistent sampling

Some Existing Measurements

Data Published	Privacy Techniques	Unsafe	Inaccurate
Relay BW available	Test measurements		✖
Relay BW used	Aggregated ~ 4 hours	✖	
Total # daily users	Inferred (consensus fetches)		✖
# users per country	Aggregated ~ 24 hours, rounded, opt-in	✖	
Exit traffic per port	Aggregated ~ 24 hours, opt-in	✖	

Motivation: Missing Measurements

Many useful statistics are not collected for safety

Users

- Total number of unique users at any time, how long they stay online, how often they join and leave, usage behavior

Relays

- Total bandwidth capacity, congestion and queuing delays, circuit and other failures, denial of service and other attacks

Destinations

- Popular destinations, popular applications, effects of DNS, properties of traffic (bytes and connections per page, etc.)

The PrivCount Measurement System

- PrivCount system architecture
- Distributed measurement and aggregation protocol
- Secure computation and private output

PrivCount: Overview

Distributed measurement system

- “Privacy-preserving counting” system
  - Tracks various types of Tor events, computes statistics from those events
  - Based on PrivEx-S2 by Elahi et al. (CCS 2014)

- Distributes trust using secret sharing across many operators

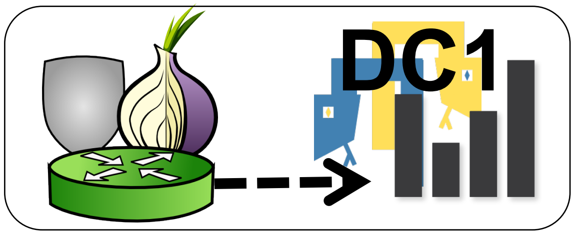
- Achieves forward privacy during measurement
  - the adversary cannot learn the state of the measurement before time of compromise

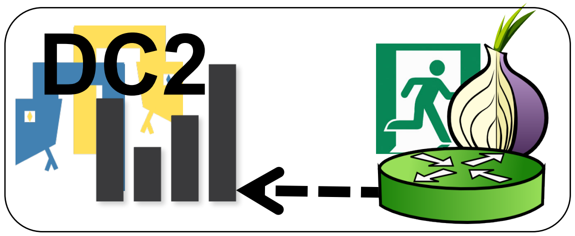
- Provides differential privacy of the results
  - prevents confirmation of the actions of a specific user given the output

PrivCount: Architecture

Data Collectors (DCs)

- Collect events
- Increment counters

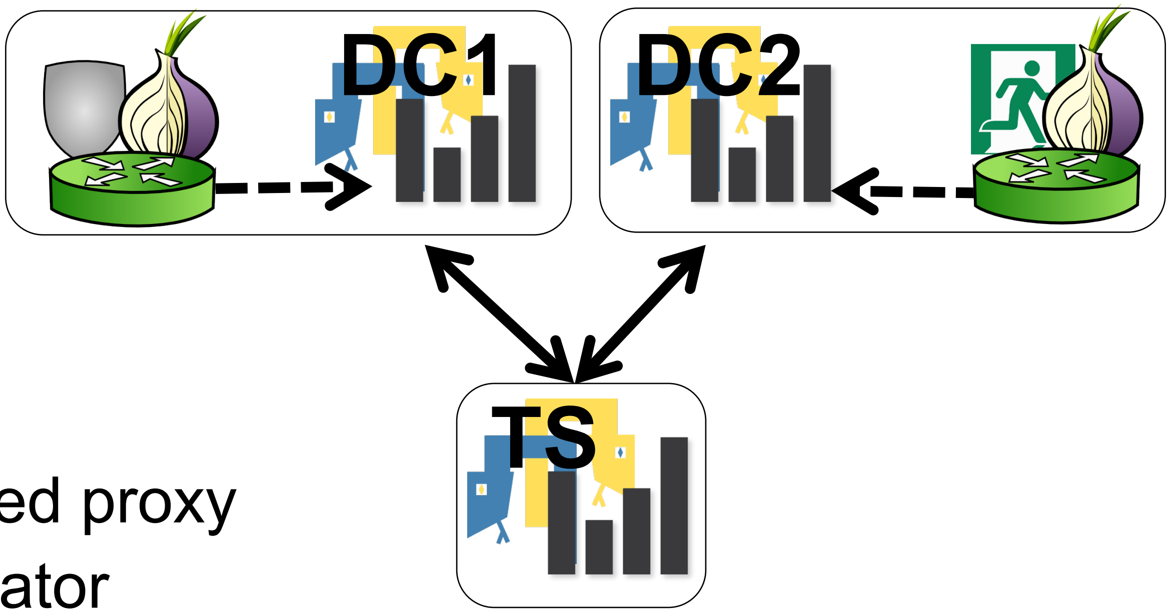




PrivCount: Architecture

Data Collectors (DCs)

- Collect events
- Increment counters



Tally Server (TS)

- Central, untrusted proxy
- Collection facilitator

U.S. NAVAL RESEARCH LABORATORY

PrivCount: Architecture

Data Collectors (DCs)

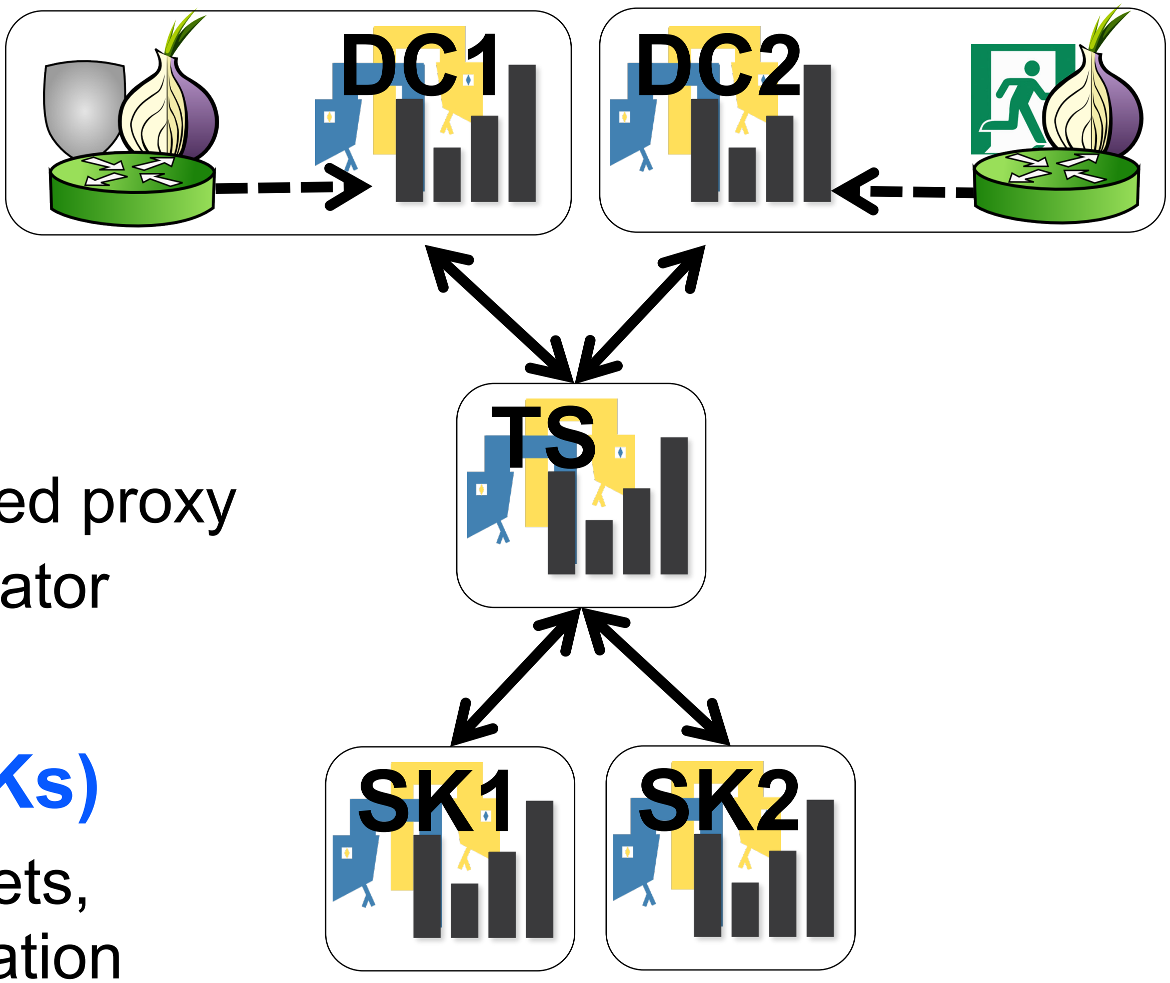
- Collect events
- Increment counters

Tally Server (TS)

- Central, untrusted proxy
- Collection facilitator

Share Keepers (SKs)

- Stores DC secrets, sum for aggregation



U.S. NAVAL RESEARCH LABORATORY

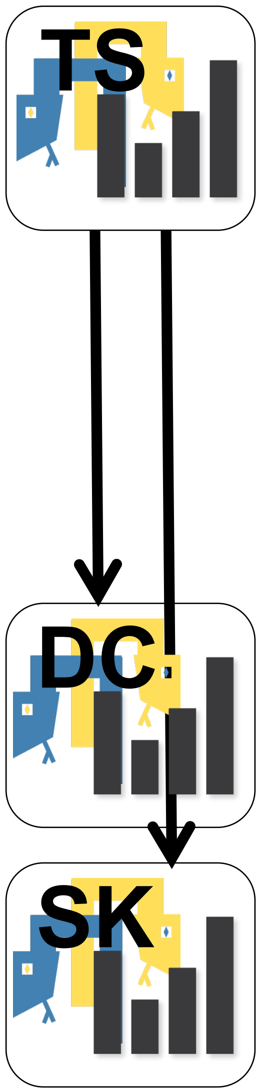
PrivCount: Initialization

TS prepares a deployment document

- DC and SK public keys (assume PKI)
- Noise parameters
  - Differential privacy parameters ε and δ
  - Sensitivity for each statistic (max change due to single client)
  - Reconfiguration time between collection periods
  - Noise weight (relative noise added by each DC)
- Minimum allowed DC subset

TS sends to all DCs and SKs for consent

- DCs and SKs accept only on unanimous consensus



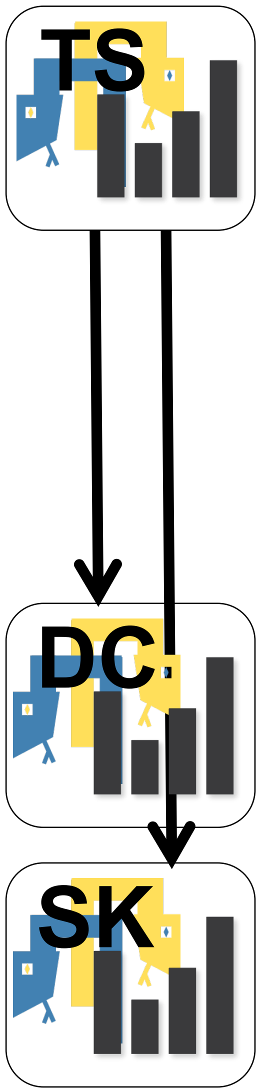
PrivCount: Configuration

TS prepares a configuration document

- Collection start and end time
- Statistics to collect
- Number of counters per statistic
- Range of each bin per statistic
- Estimated value for each statistic
  - maximize relative per-statistic accuracy while providing (ε, δ)-differential privacy

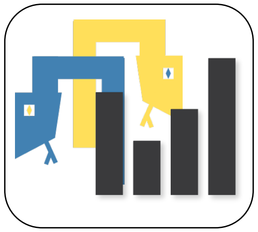
TS sends to all DCs and SKs for consistency

- DCs and SKs accept if consistency check passes



PrivCount: Counting

Counts single numbers and histograms



- Given a value to count:
  - Find bin that contains value
  - Increment counter for that bin

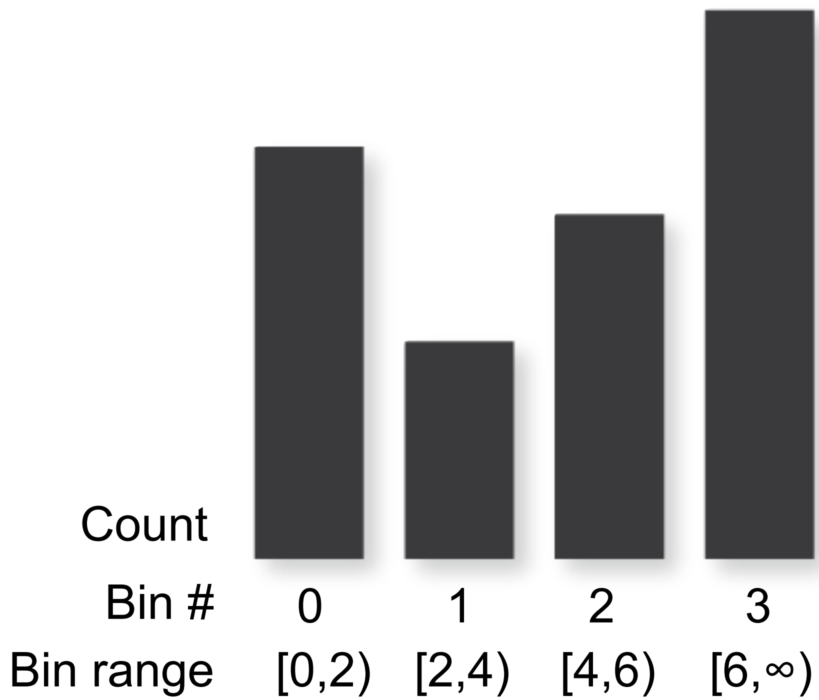
PrivCount: Counting

Counts single numbers and histograms

- Given a value to count:
  - Find bin that contains value
  - Increment counter for that bin

Example

- Counting streams per circuit
- Found value 5
- Increment bin 2



PrivCount: Execution - Setup

1. Generate noise for each counter
   - N ~ Normal(0,ωσ) mod q

Computed from noise parameters in deployment and configuration documents

PrivCount: Execution - Setup

1. Generate noise for each counter
   - N ~ Normal(0,ωσ) mod q

2. Generate random number "share" for each SK
   - S1 ~ Uniform({0, …, q-1})
   - S2 ~ Uniform({0, …, q-1})

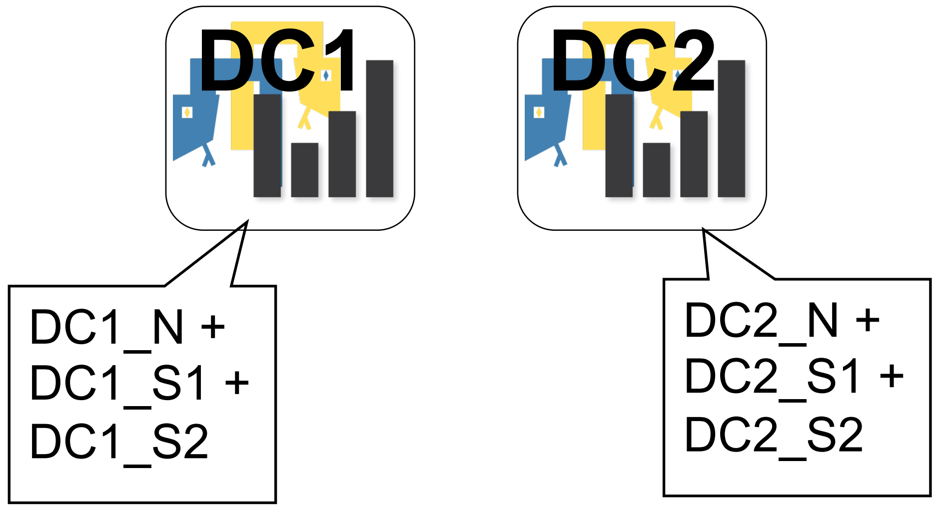
Serve to "blind" the actual count at the DC machine

PrivCount: Execution - Setup

1. Generate noise for each counter
   - N ~ Normal(0,ωσ) mod q

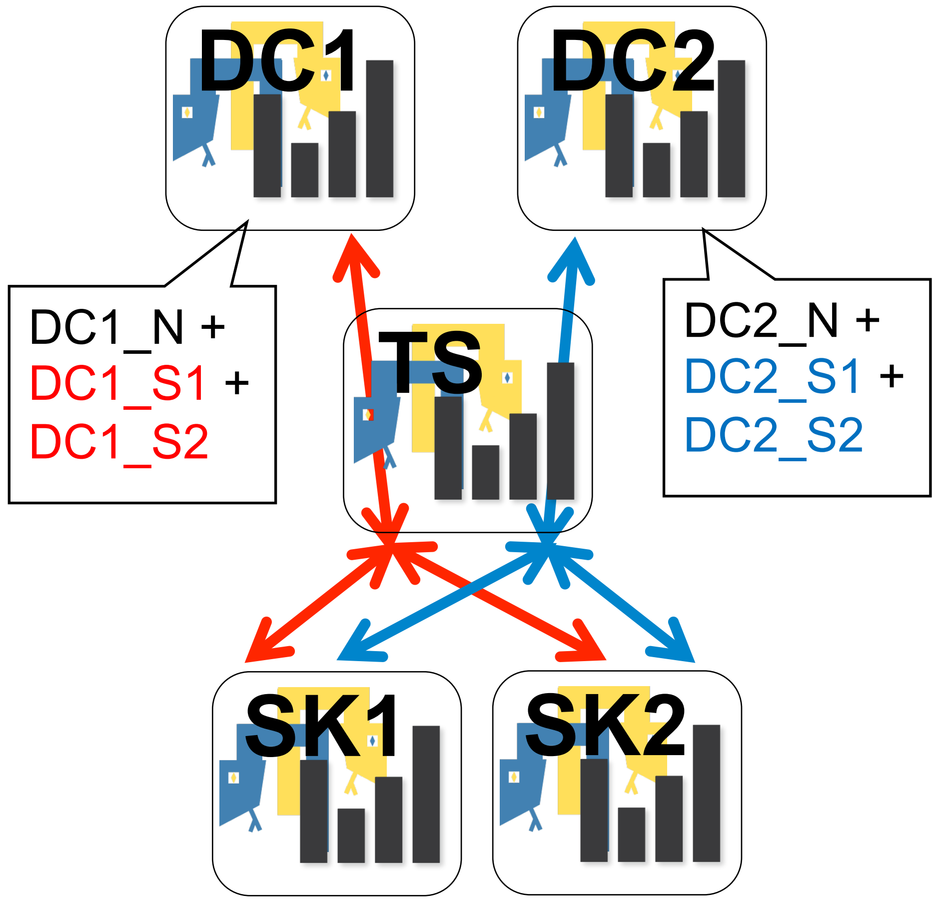
2. Generate random number "share" for each SK
   - S1 ~ Uniform({0, …, q-1})
   - S2 ~ Uniform({0, …, q-1})

3. Initialize counters



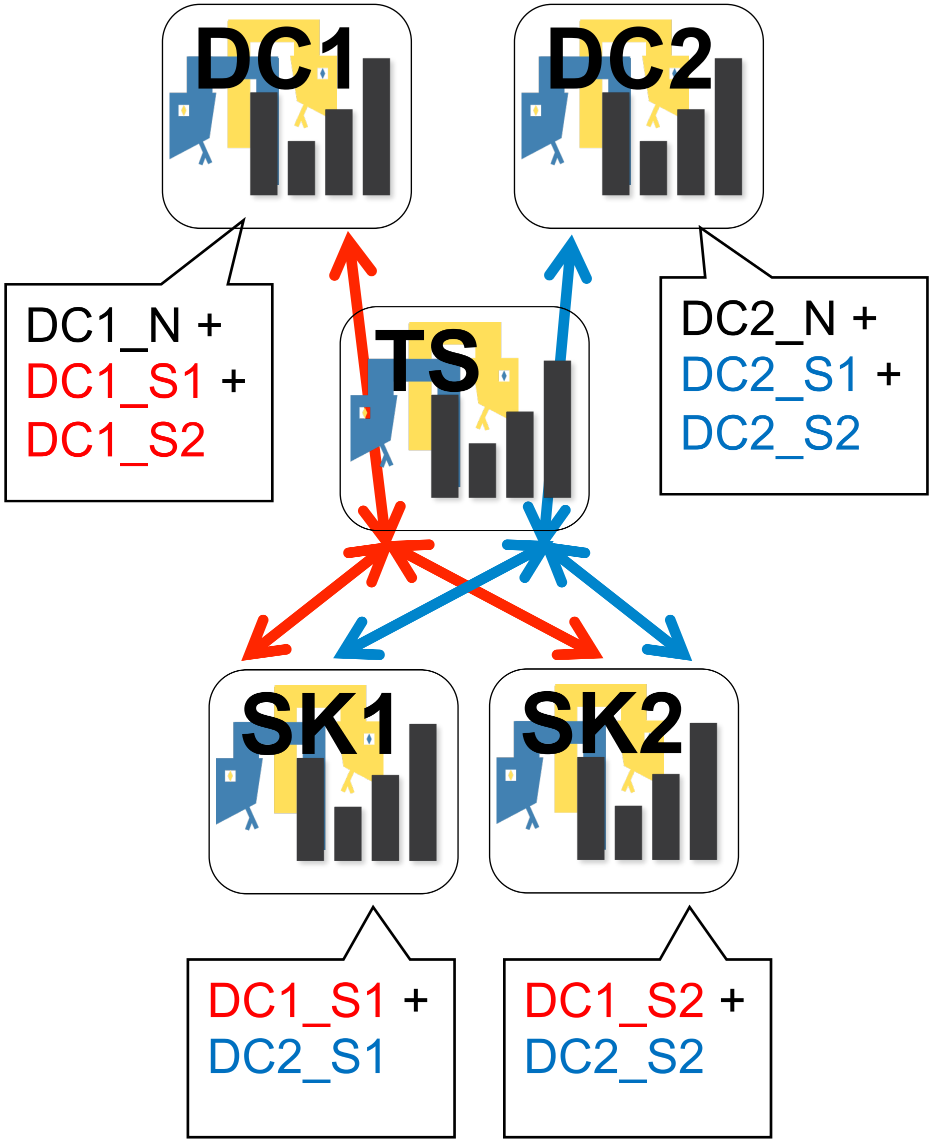
PrivCount: Execution - Setup

1. Generate noise for each counter
   - N ~ Normal(0,ωσ) mod q
2. Generate random number "share" for each SK
   - S1 ~ Uniform({0, …, q-1})
   - S2 ~ Uniform({0, …, q-1})
3. Initialize counters
4. Send shares to SKs, erase

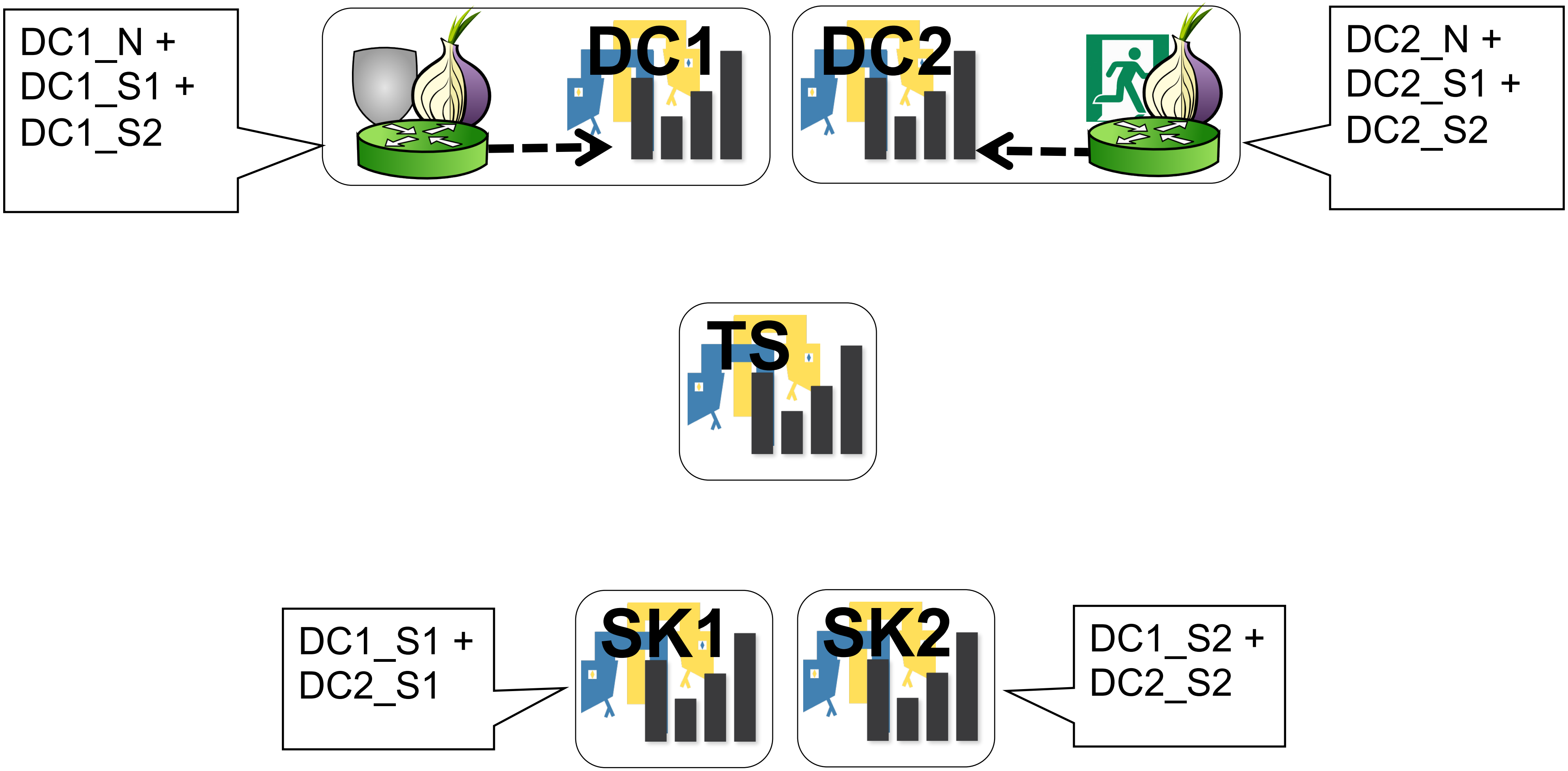


PrivCount: Execution - Setup

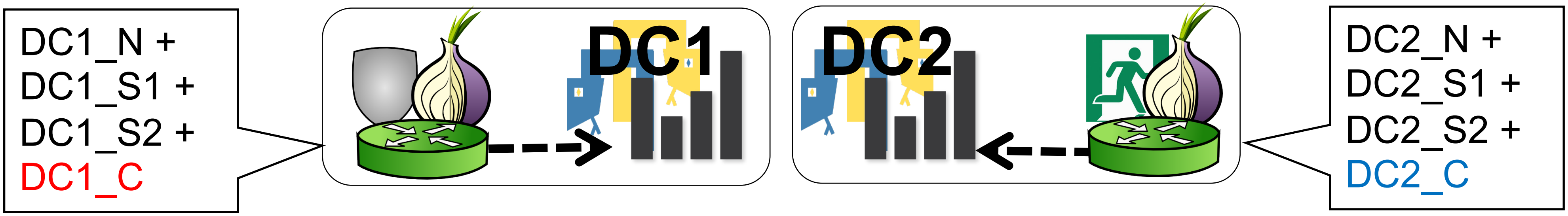
1. Generate noise for each counter
   - N ~ Normal(0,ωσ) mod q
2. Generate random number "share" for each SK
   - S1 ~ Uniform({0, …, q-1})
   - S2 ~ Uniform({0, …, q-1})
3. Initialize counters
4. Send shares to SKs, erase



PrivCount: Execution - Collection

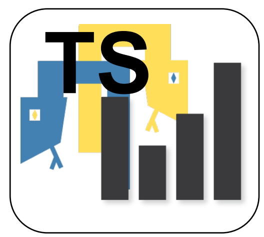


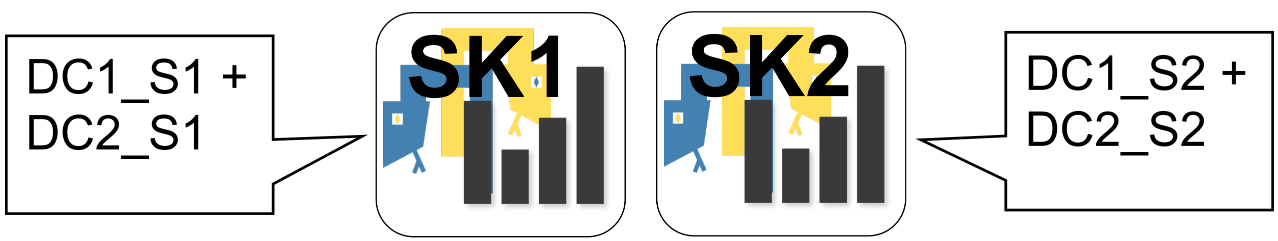
PrivCount: Execution - Collection



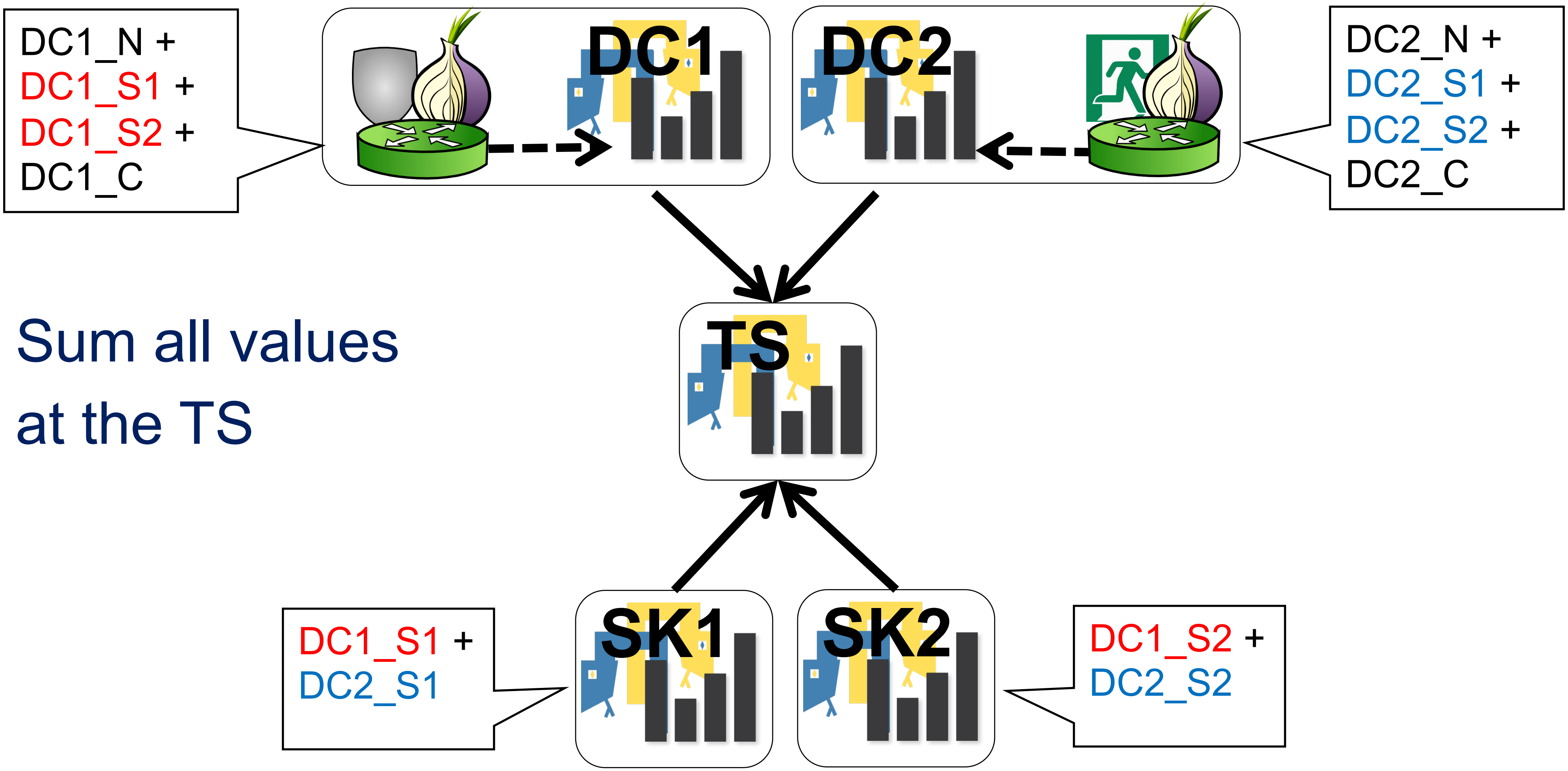
Data collectors

- Collect events
- Increment counters



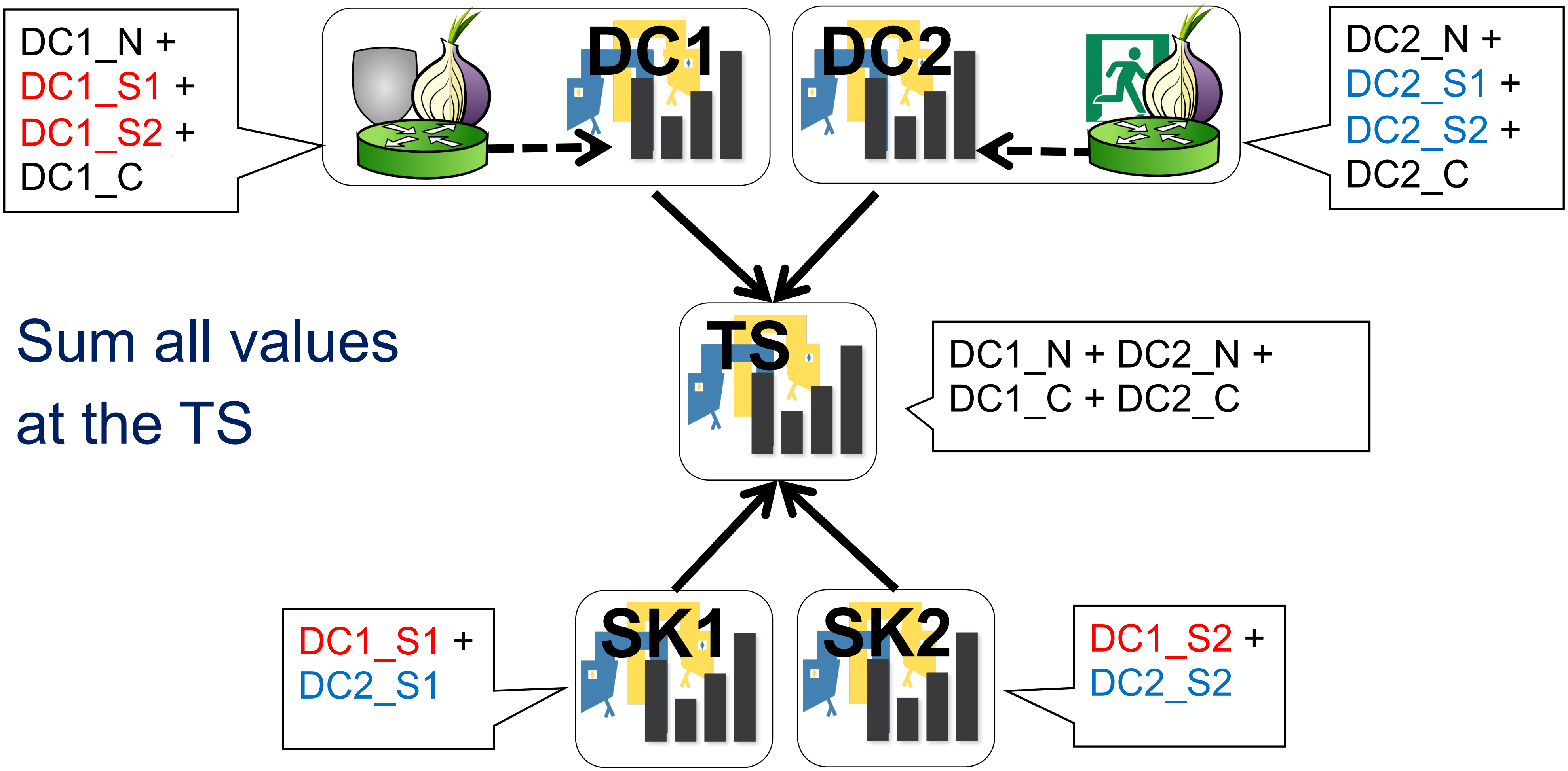


PrivCount: Execution - Aggregation



Sum all values at the TS

PrivCount: Execution - Aggregation

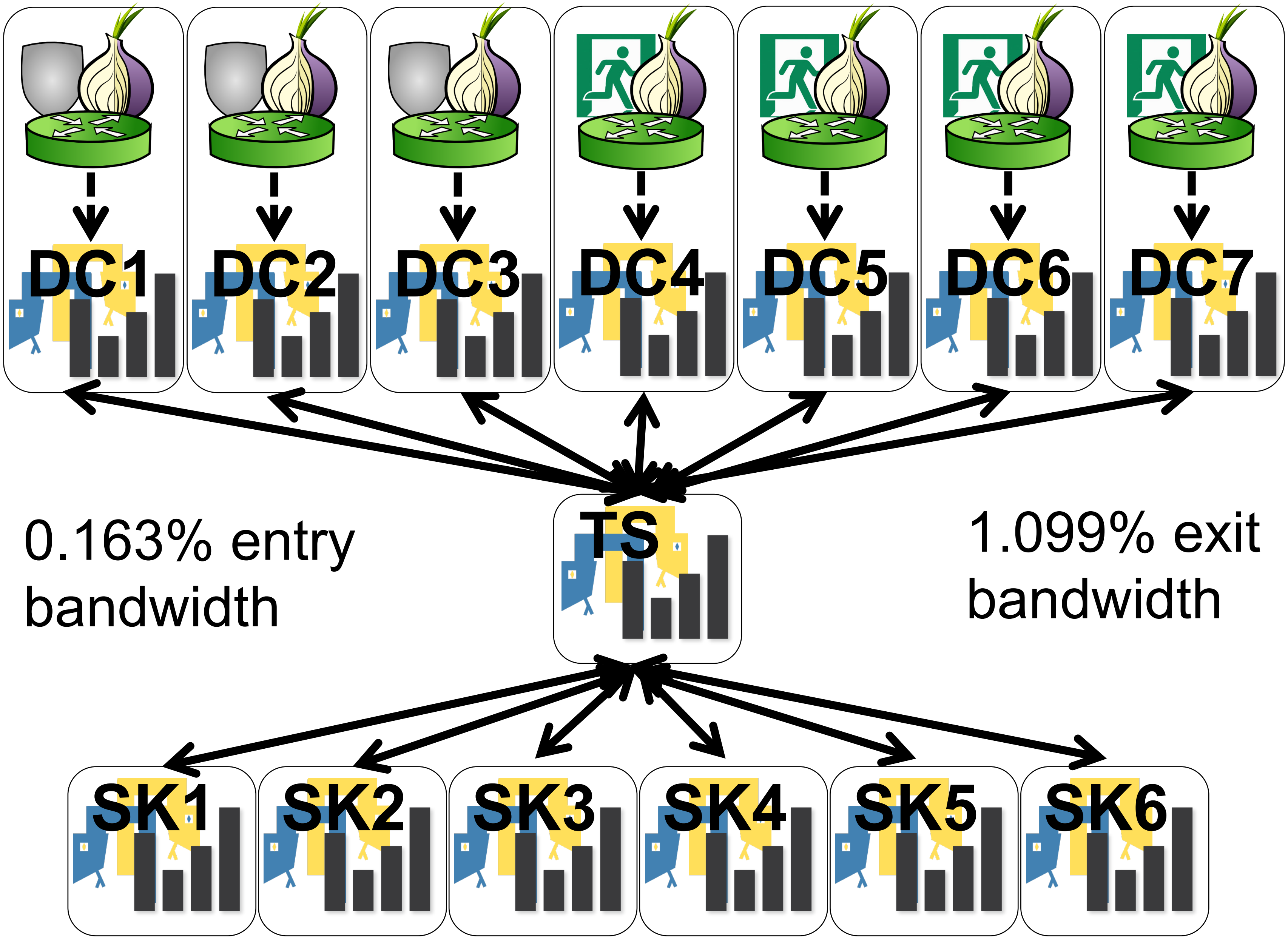


Sum all values at the TS

Deployment and Measurement Results

- Configuring and running Tor relays
- “Exploratory” measurements using various exit policies
- “In-depth” measurements of most popular usage
- Network-wide measurement inference

Deploying PrivCount



0.163% entry bandwidth

1.099% exit bandwidth

U.S. NAVAL RESEARCH LABORATORY

Collection Phases

Exploratory phases

- Explore various exit policies (strict, default, open)
- Explore various applications (web, interactive, other)
- Gather only totals (circuits, streams, bytes)
- Use Tor metrics to estimate input parameters
- Run for 1 day, iterate

Collection Phases

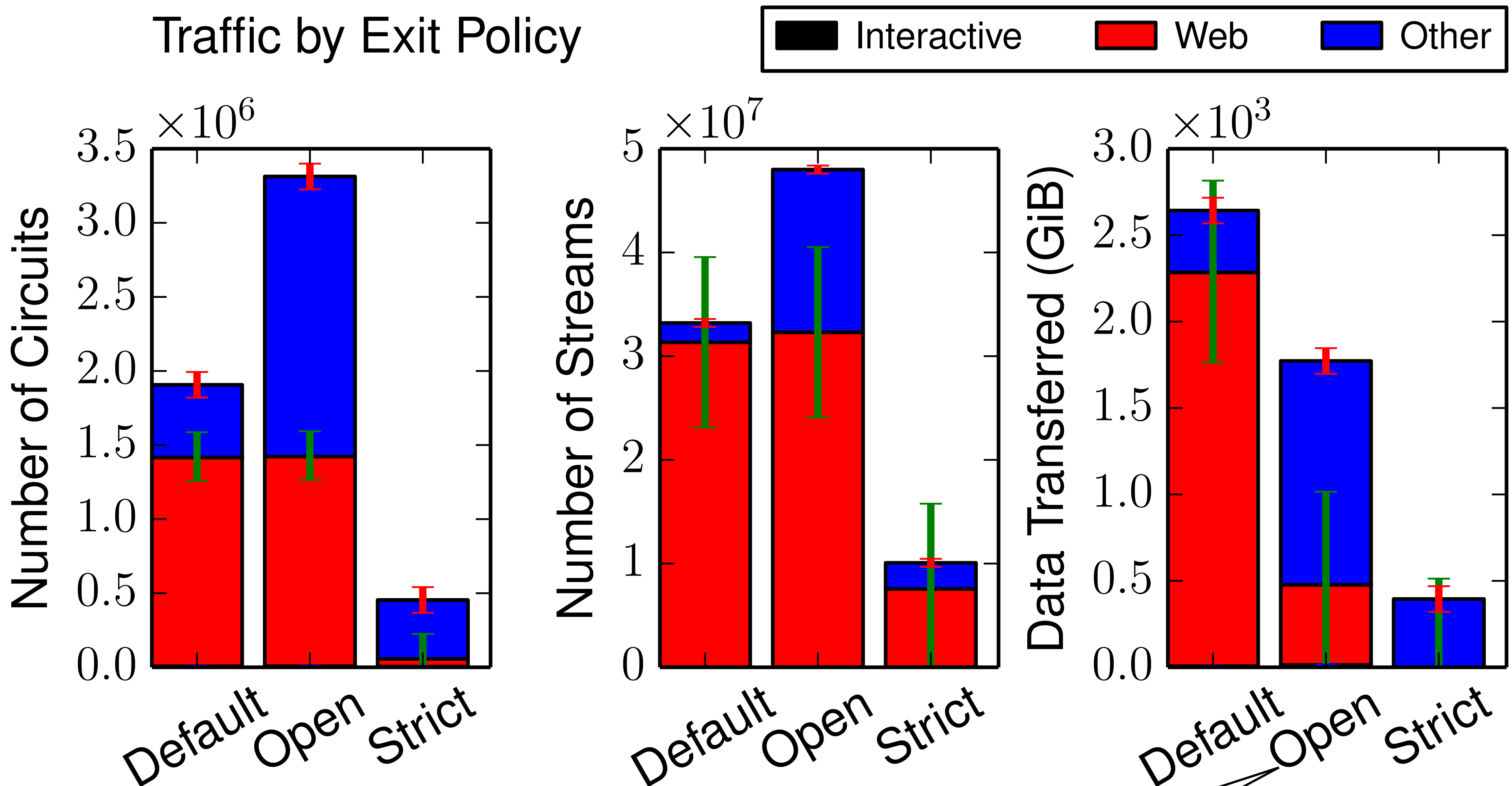
Exploratory phases

- Explore various exit policies (strict, default, open)
- Explore various applications (web, interactive, other)
- Gather only totals (circuits, streams, bytes)
- Use Tor metrics to estimate input parameters
- Run for 1 day, iterate

In-depth phases

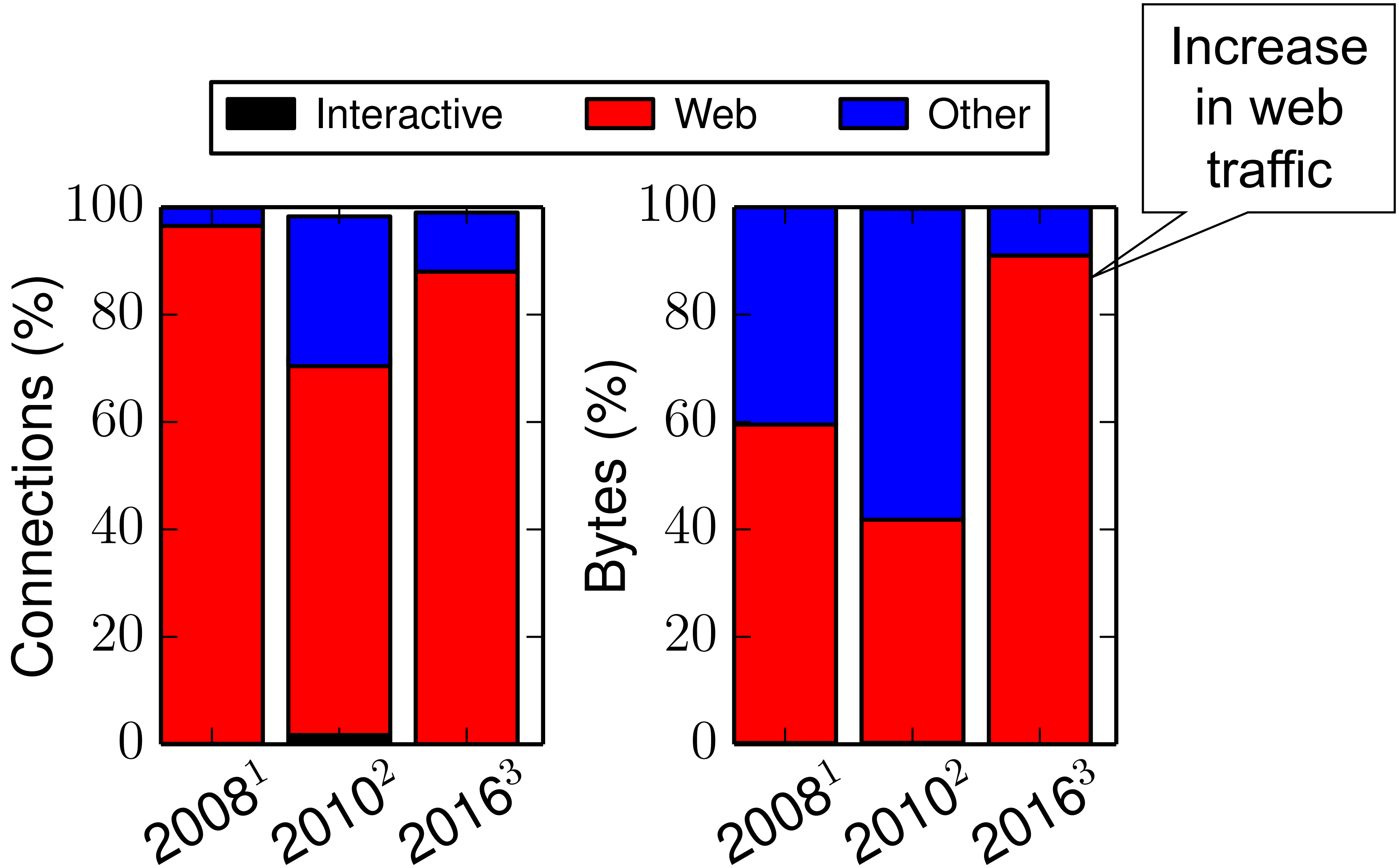
- Focus on most popular exit policy and applications
- Gather totals and histograms
- Use exploratory results to estimate input parameters
- Run for 4 days for client stats, 21 days for exit stats

Results: Exit Policies



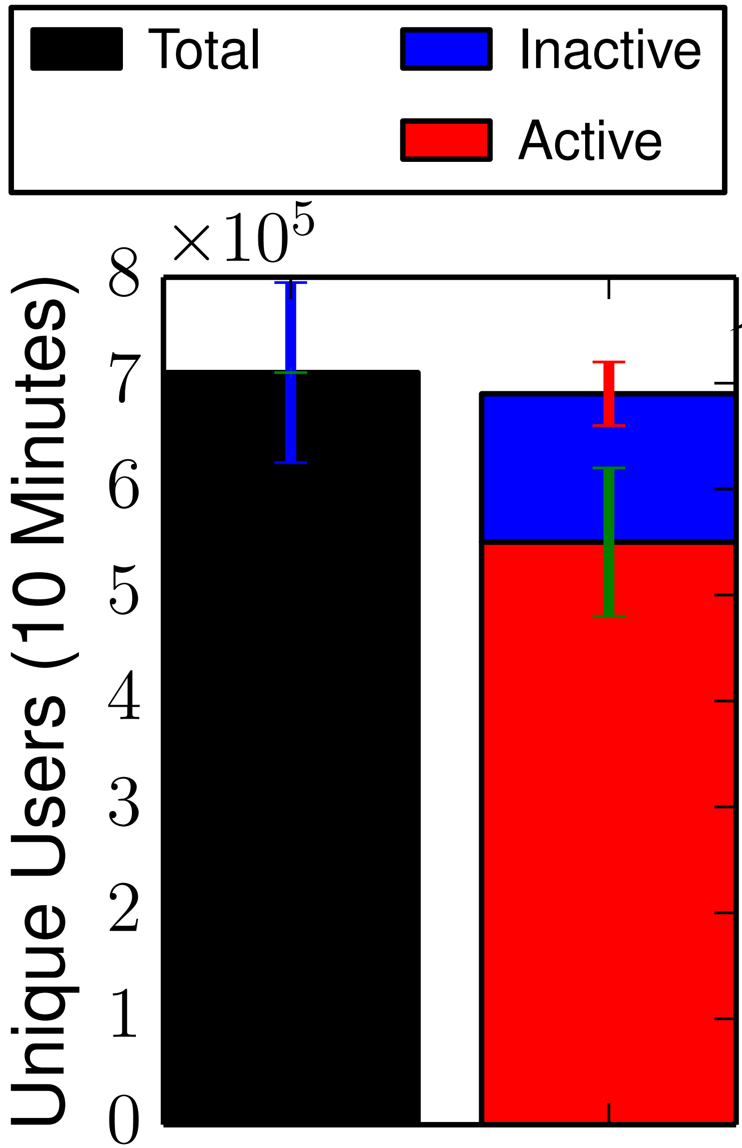
Open file sharing ports reduces web data transferred

Results: Amount and Types of Traffic



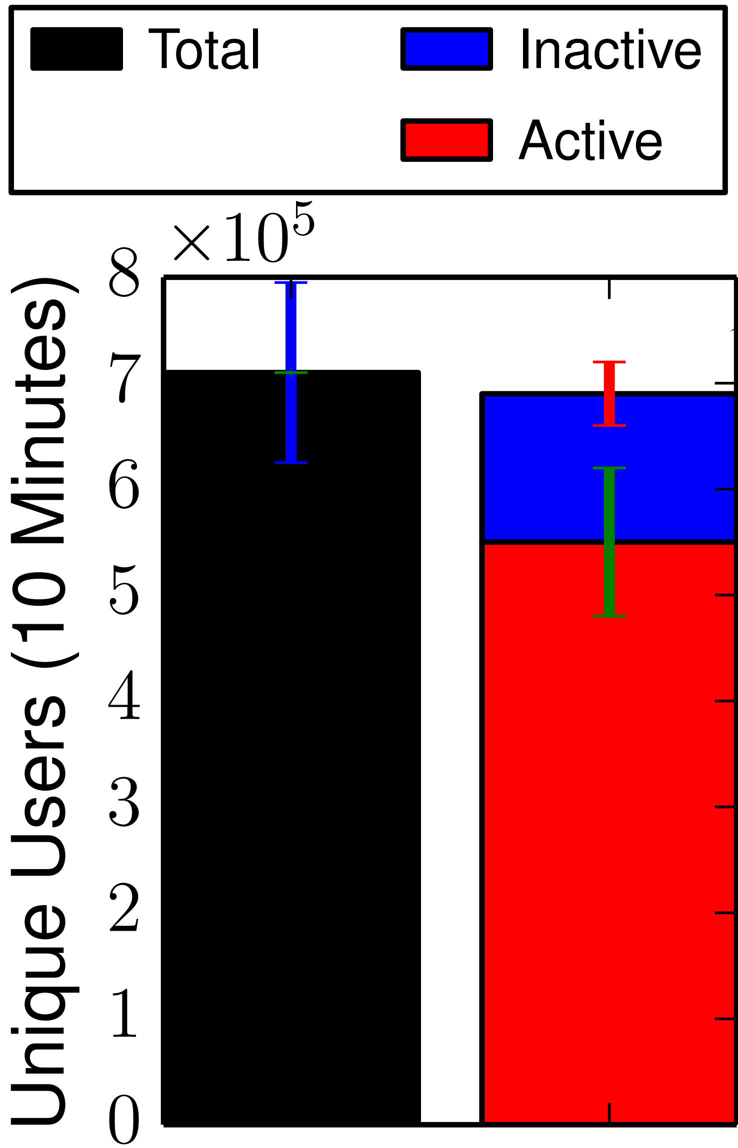
[1] PETS 2008, McCoy... [2] NSS 2010, Chaabane... [3] CCS 2016, Jansen...

Results: Number of Unique Users



710,000 total users
550,000 active users
In an average 10 mins.

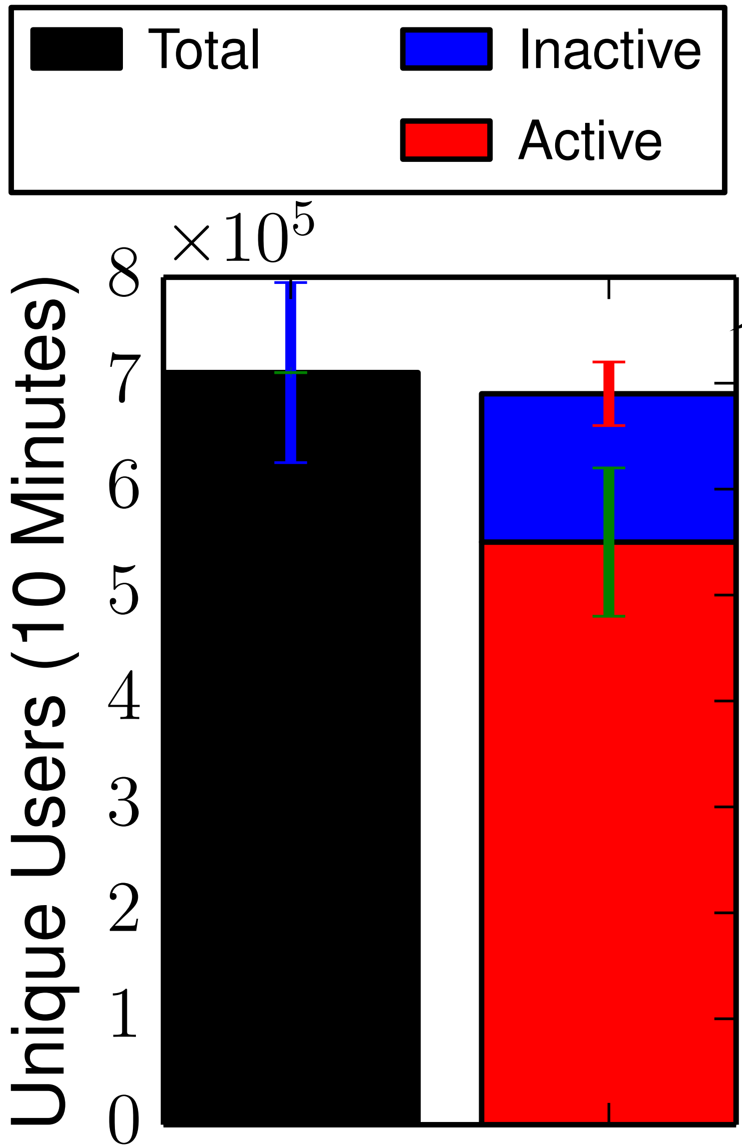
Results: Number of Unique Users



710,000 total users
550,000 active users
In an average 10 mins.

~1,750,000 daily users
(Consensus downloads – https://metrics.torproject.org)

Results: Number of Unique Users



710,000 total users
550,000 active users
In an average 10 mins.

~1,750,000 daily users
(Consensus downloads – https://metrics.torproject.org)

~800,000 – ~1,600,000 average concurrent users
(Tor Browser update pings - https://tor-metrics.shinyapps.io/webstats2/)

Results: Traffic Modeling Statistics

Table 11: Distributions of Tor network activity from histogram-counter in-depth exit statistics

Statistic		Bin Ranges and Count Distribution (with ± 95% CI)							
Active Circuit Life Time (s)		[1, 480):	57%±44%	[480, 720):	45%±42%	[720, 1200):	0%±33%	[1200, ∞):	0%±35%
Streams Per Circuit	Total	[1, 3):	46%±43%	[3, 7):	38%±41%	[7, 15):	31%±40%	[15, ∞):	9%±37%
	Web	[1, 3):	36%±37%	[3, 7):	22%±33%	[7, 15):	13%±31%	[15, ∞):	3%±28%
	Other	[1, 3):	78%±15%	[3, 7):	10%±9%	[7, 15):	0%±8%	[15, ∞):	2%±8%
Client-bound Bytes Per Stream	Total	[1, 2048):	60%±40%	[2048, 16384):	38%±35%	[16384, 65536):	32%±33%	[65536, ∞):	6%±26%
	Web	[1, 2048):	33%±33%	[2048, 16384):	37%±34%	[16384, 65536):	5%±26%	[65536, ∞):	0%±24%
	Other	[1, 2048):	56%±21%	[2048, 16384):	9%±15%	[16384, 65536):	8%±15%	[65536, ∞):	11%±15%
Server-bound Bytes Per Stream	Total	[1, 512):	57%±39%	[512, 1024):	25%±31%	[1024, 4096):	38%±34%	[4096, ∞):	0%±24%
	Web	[1, 512):	41%±35%	[512, 1024):	36%±34%	[1024, 4096):	23%±30%	[4096, ∞):	2%±25%
	Other	[1, 512):	40%±19%	[512, 1024):	6%±14%	[1024, 4096):	15%±16%	[4096, ∞):	1%±14%
Bytes Per Stream Ratio	Total	(-∞, -1):	80%±45%	[-1, 1):	25%±31%	[1, ∞):	0%±21%		
	Web	(-∞, -1):	70%±42%	[-1, 1):	15%±28%	[1, ∞):	0%±21%		
	Other	(-∞, -1):	45%±20%	[-1, 1):	14%±16%	[1, ∞):	12%±15%		
Inter-stream Creation Time (s)	Total	[0, 1):	87%±47%	[1, 5):	16%±29%	[5, 10):	1%±25%	[10, ∞):	0%±23%
	Web	[0, 1):	68%±41%	[1, 5):	8%±27%	[5, 10):	13%±28%	[10, ∞):	14%±28%
	Other	[0, 1):	16%±16%	[1, 5):	10%±15%	[5, 10):	3%±14%	[10, ∞):	12%±15%

Conclusion

Distributed measurement for Tor

- Improve accuracy, safety, security
- Allow us to collect more statistics
- Open source: https://github.com/privcount

Future measurement plans

- Network traffic to produce models that can be used to generate realistic traffic
- Onion services to improve reliability and scalability
- Better techniques for cardinality (e.g., # unique users)
- Detecting denial of service attacks and other misbehavior

Contact

- rob.g.jansen@nrl.navy.mil, robgjansen.com, @robgjansen

Questions