NetHide: Secure and Practical Network Topology Obfuscation

Roland Meier[1], Petar Tsankov[1], Vincent Lenders[2], Laurent Vanbever[1], Martin Vechev[1]

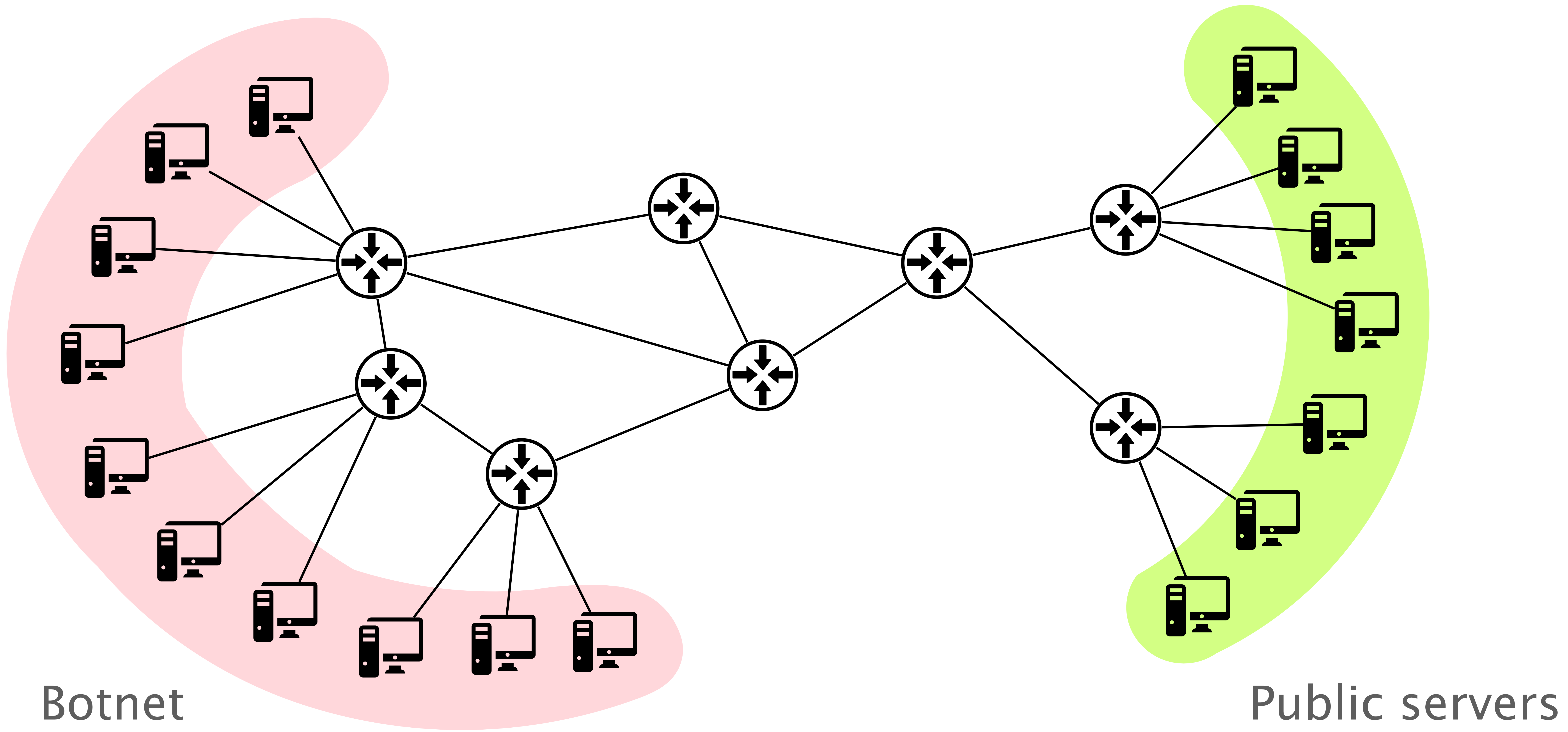
nethide.ethz.ch

USENIX Security 2018

(1) ETH zürich

(2) Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

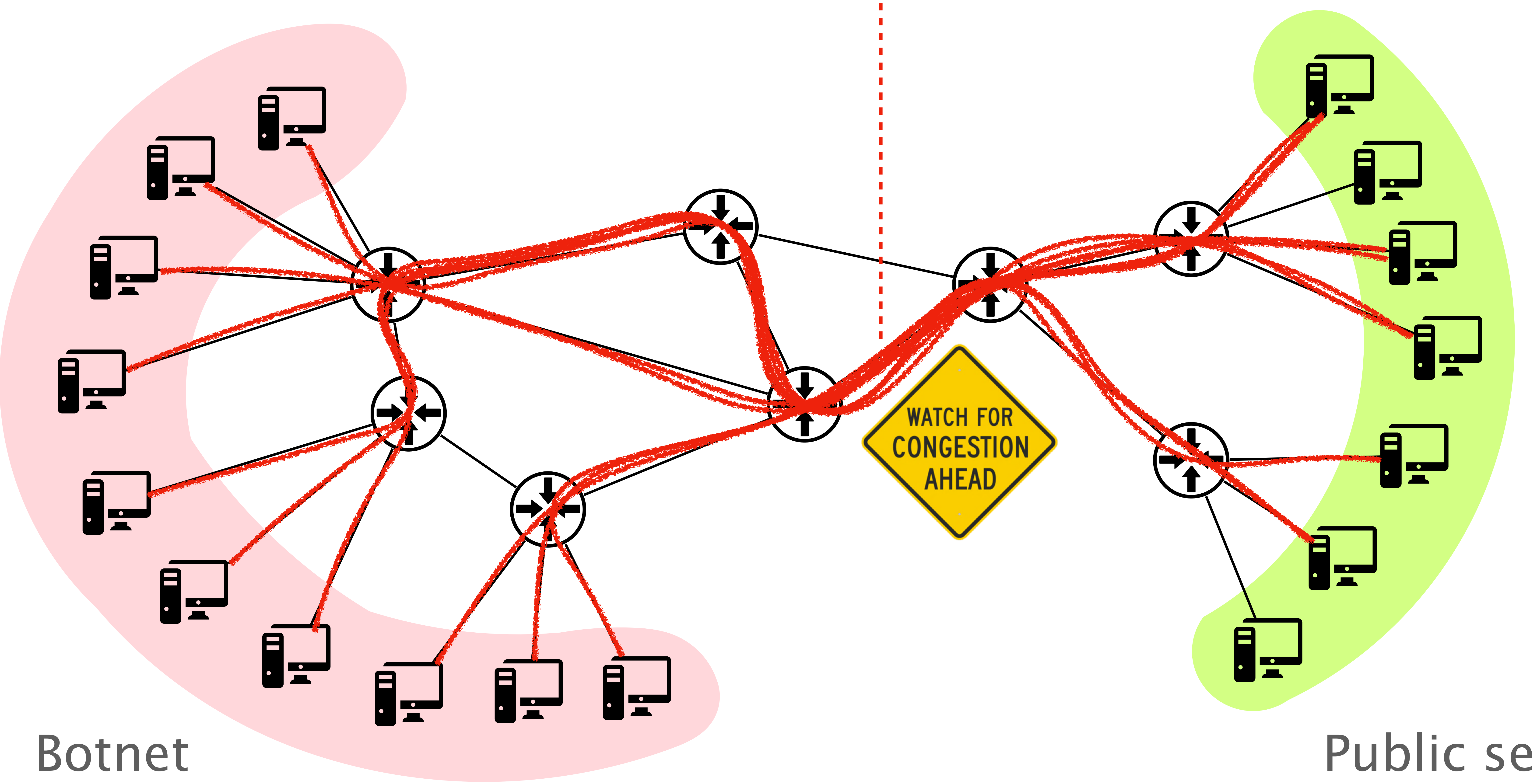
armasuisse


Botnet
Public servers

Link-flooding attacks (LFAs)

target the infrastructure

Low-rate, legitimate flows
spread over many endpoints



Botnet

Public servers

Startups
Apps
Gadgets
Events
Videos
—
Crunchbase
More

Search

Facebook
Samsung
Gaming
Blockchain

ProtonMail On Battling A Sustained DDoS Attack

Natasha Lomas @riptari / Nov 7, 2015 Comment

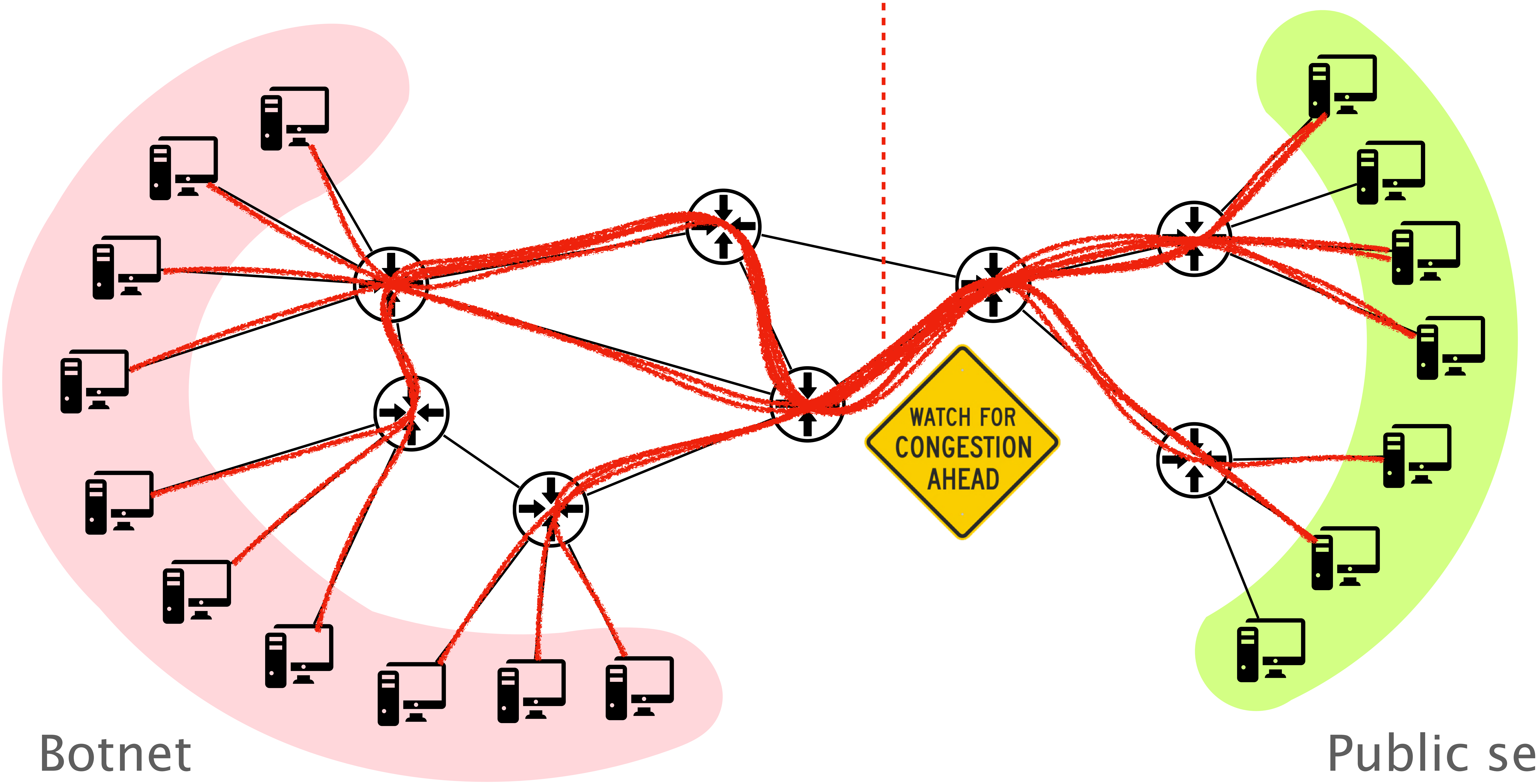
Encrypted webmail provider, ProtonMail, has been fighting a wave of DDoS attacks since November 3 that, by last Friday, had taken its service offline for more than 24 hours. At the time of writing the attacks are still coming.

They have included what ProtonMail co-founder Andy Yen described as a “co-ordinated assault” on its ISP that exceeded 100Gbps and attacked not only the Swiss datacenter but routers in various locations where the ISP has nodes — taking multiple services offline, not just ProtonMail’s email.

Link-flooding attacks (LFAs)

target the infrastructure

Low-rate, legitimate flows
spread over many endpoints

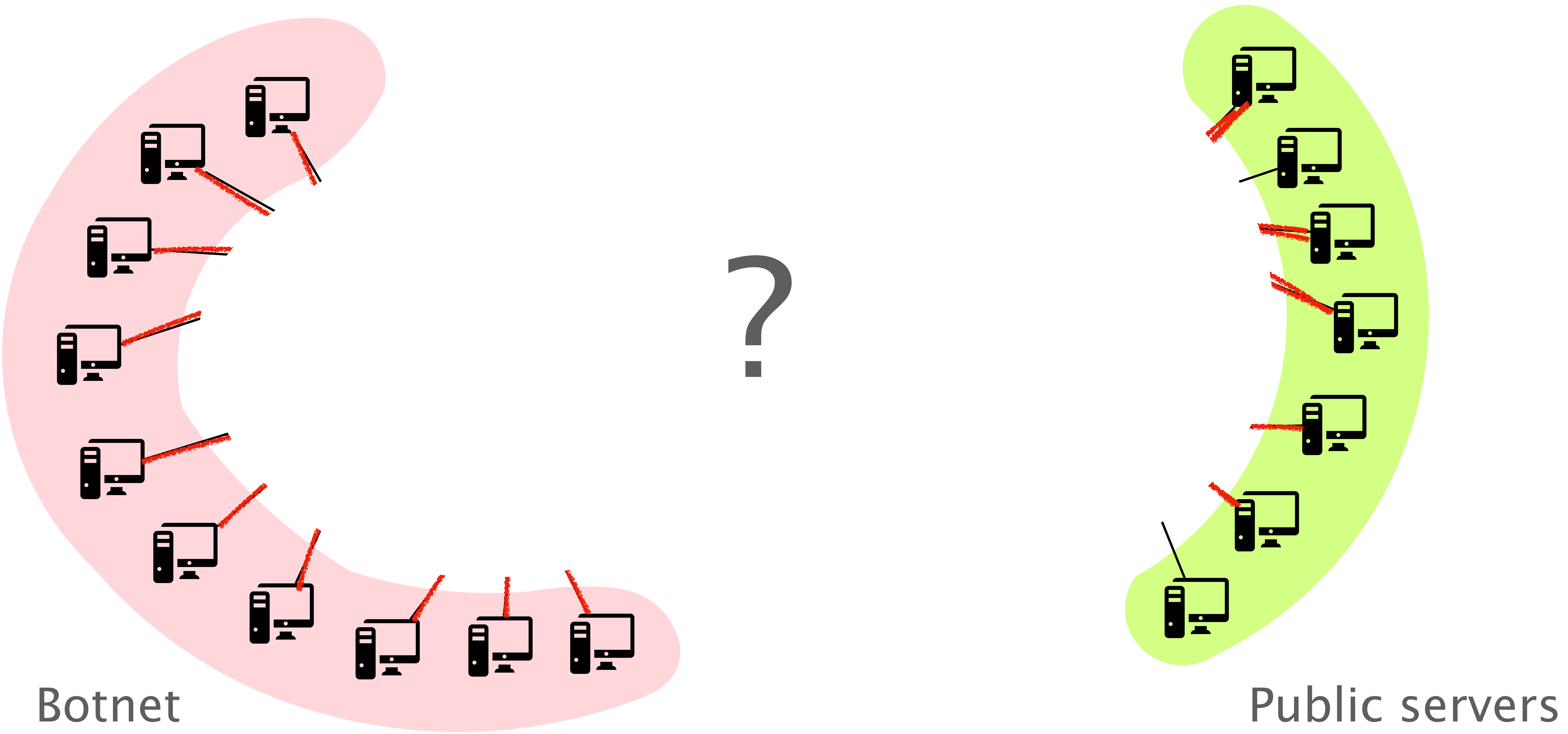


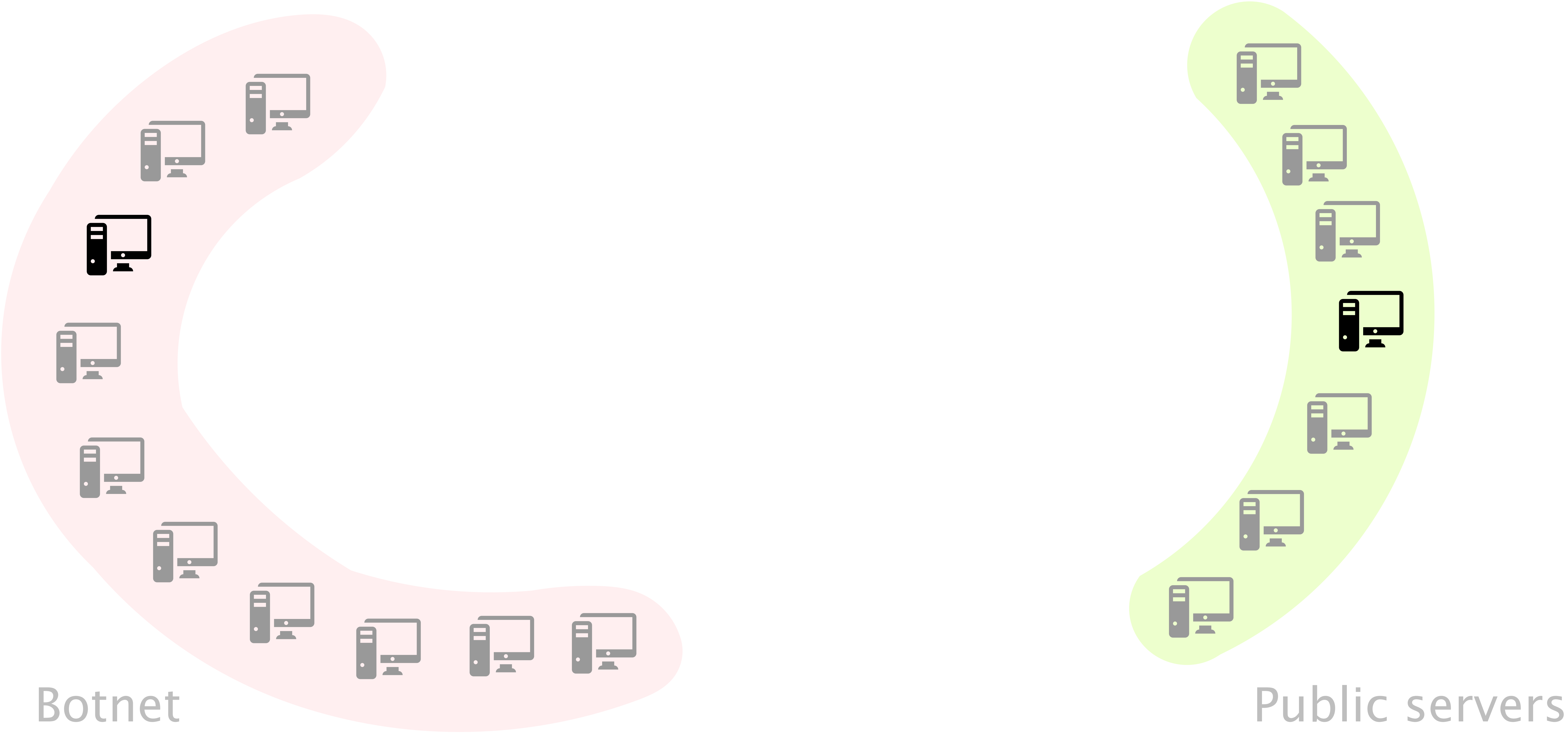
Botnet

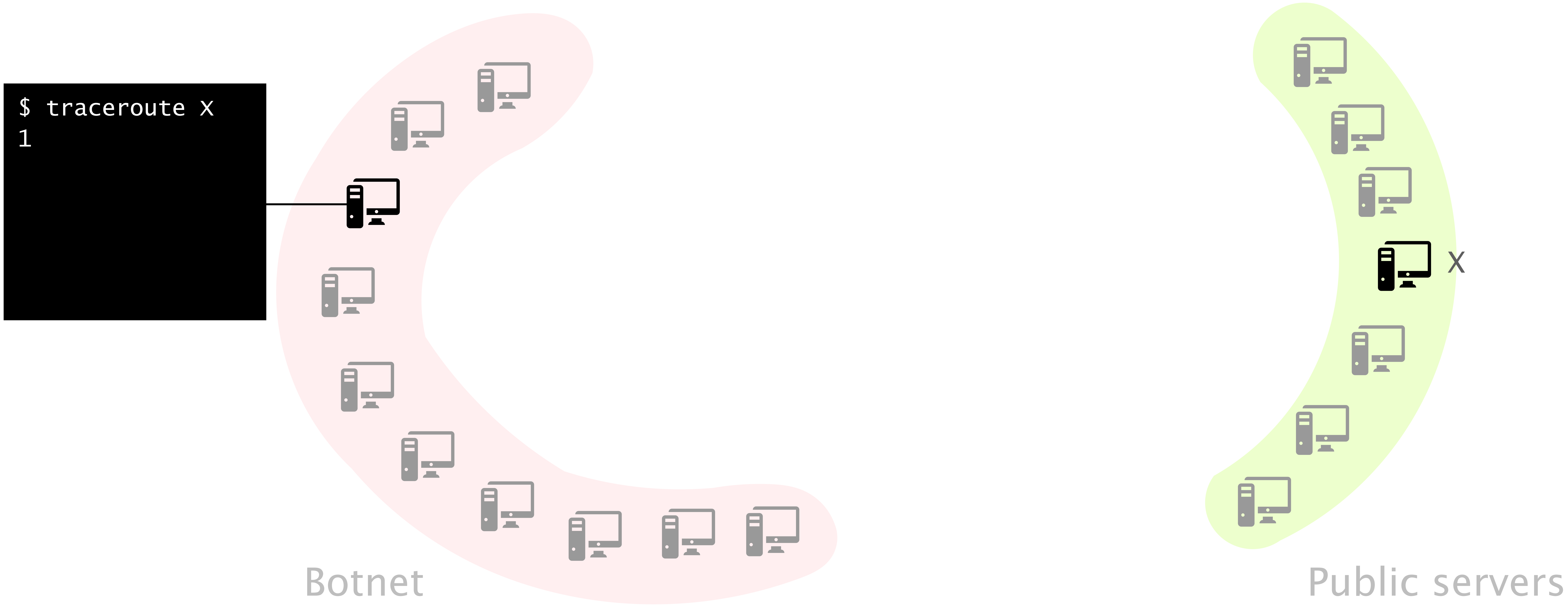
Public servers

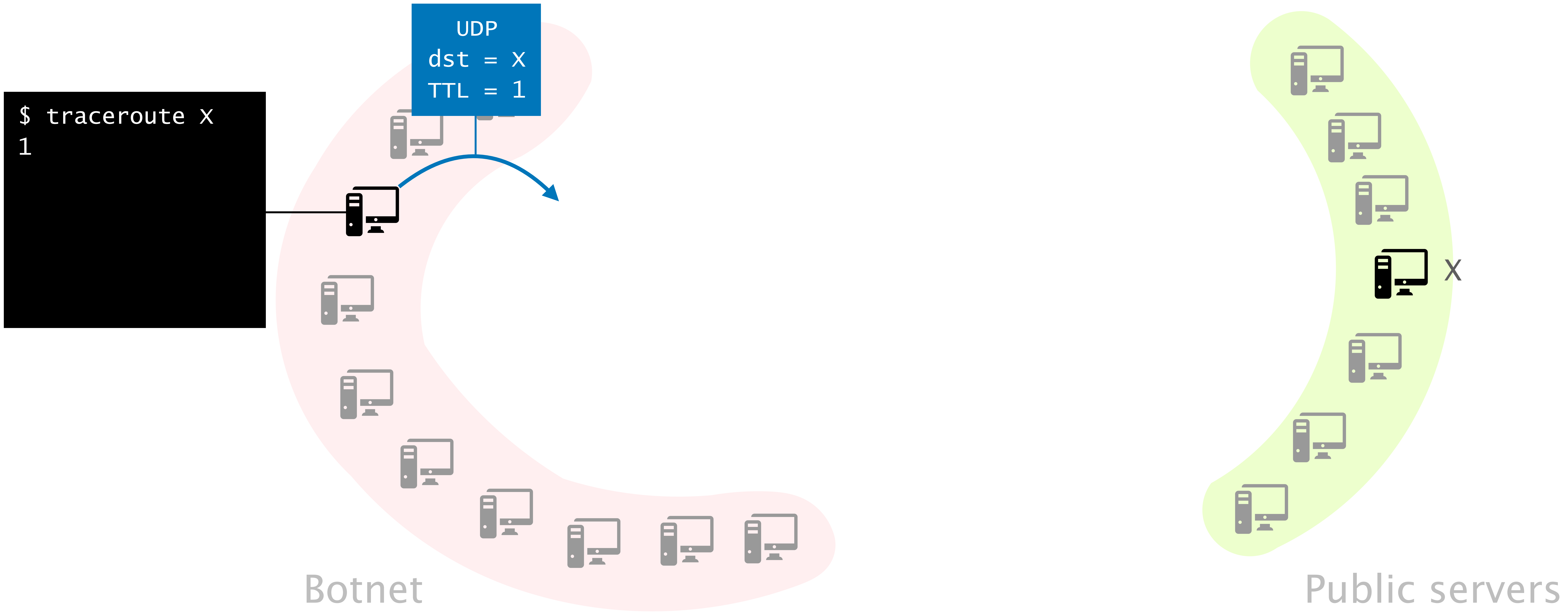
Link-flooding attacks (LFAs)

require knowing the topology

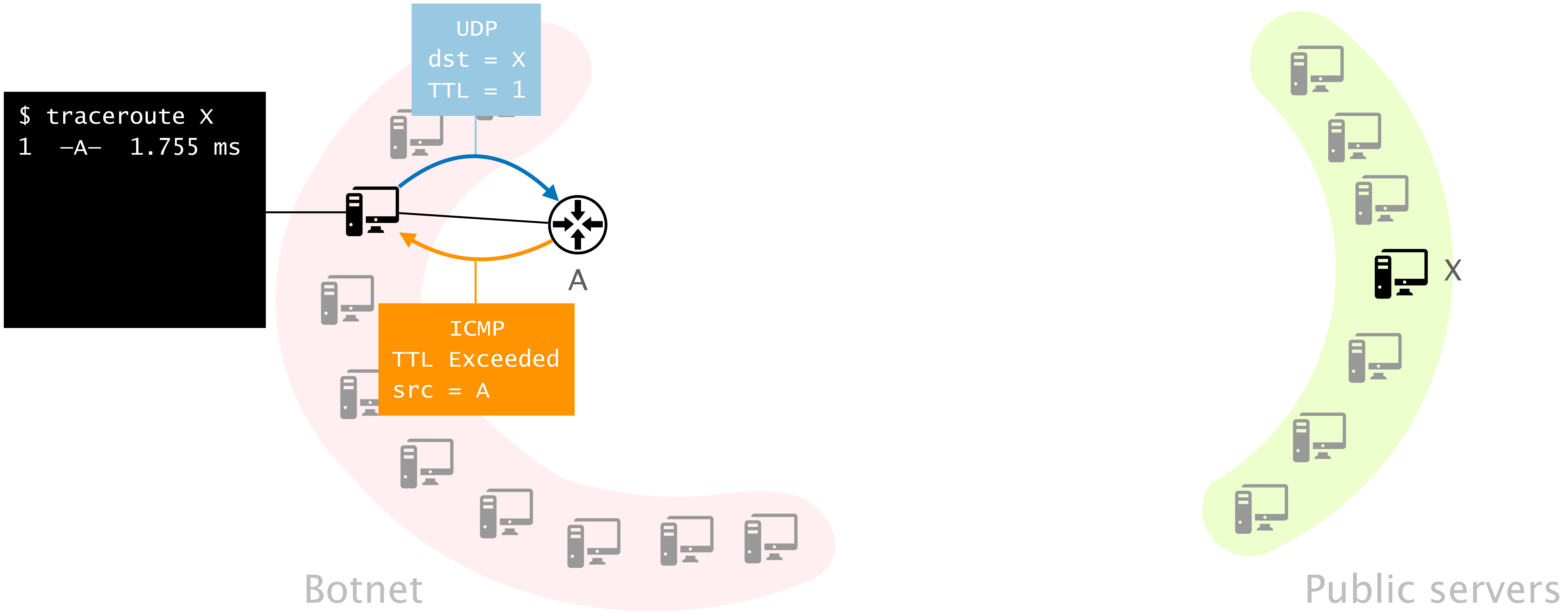


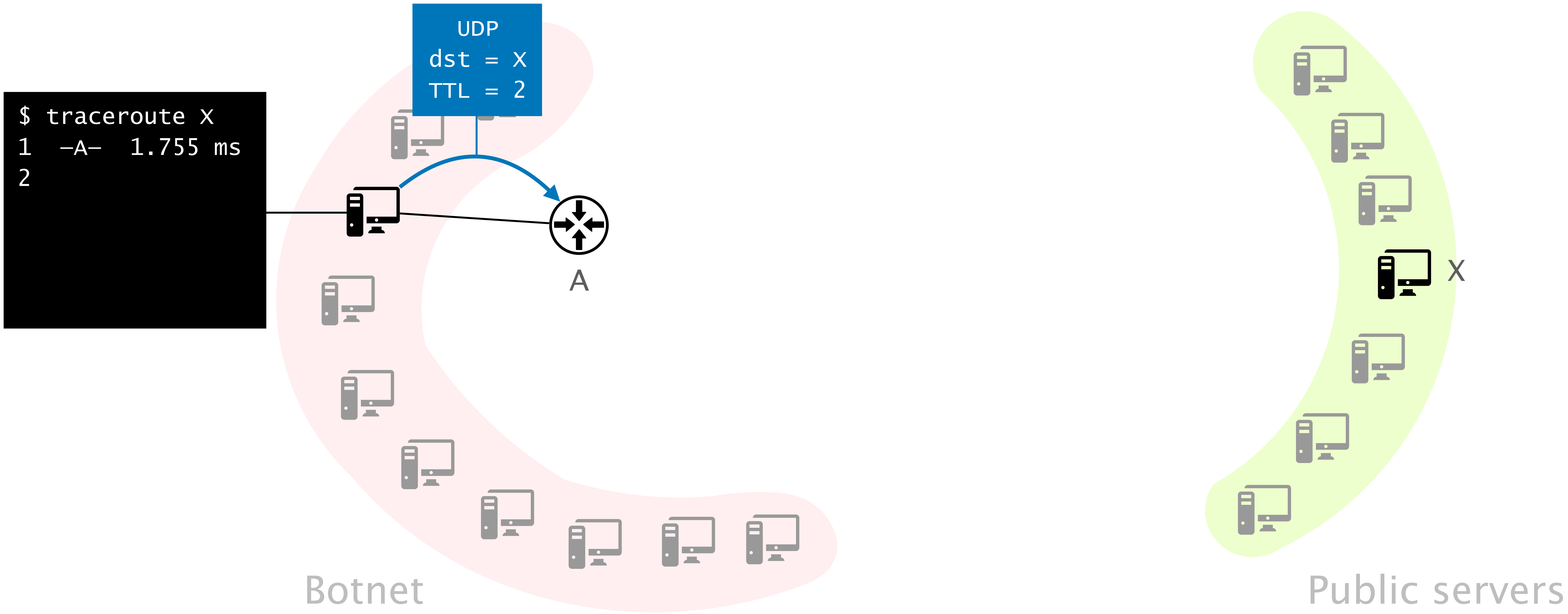

Botnet
Public servers

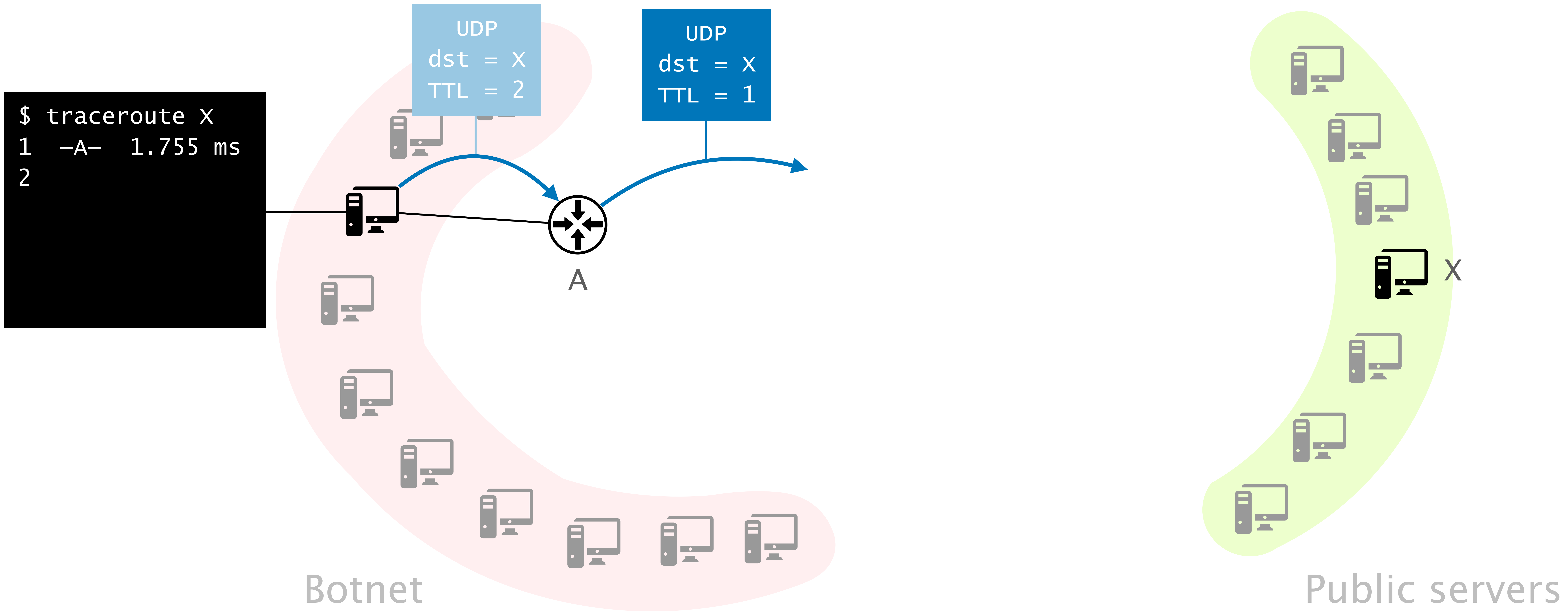

$ traceroute X
1
X
Botnet
Public servers

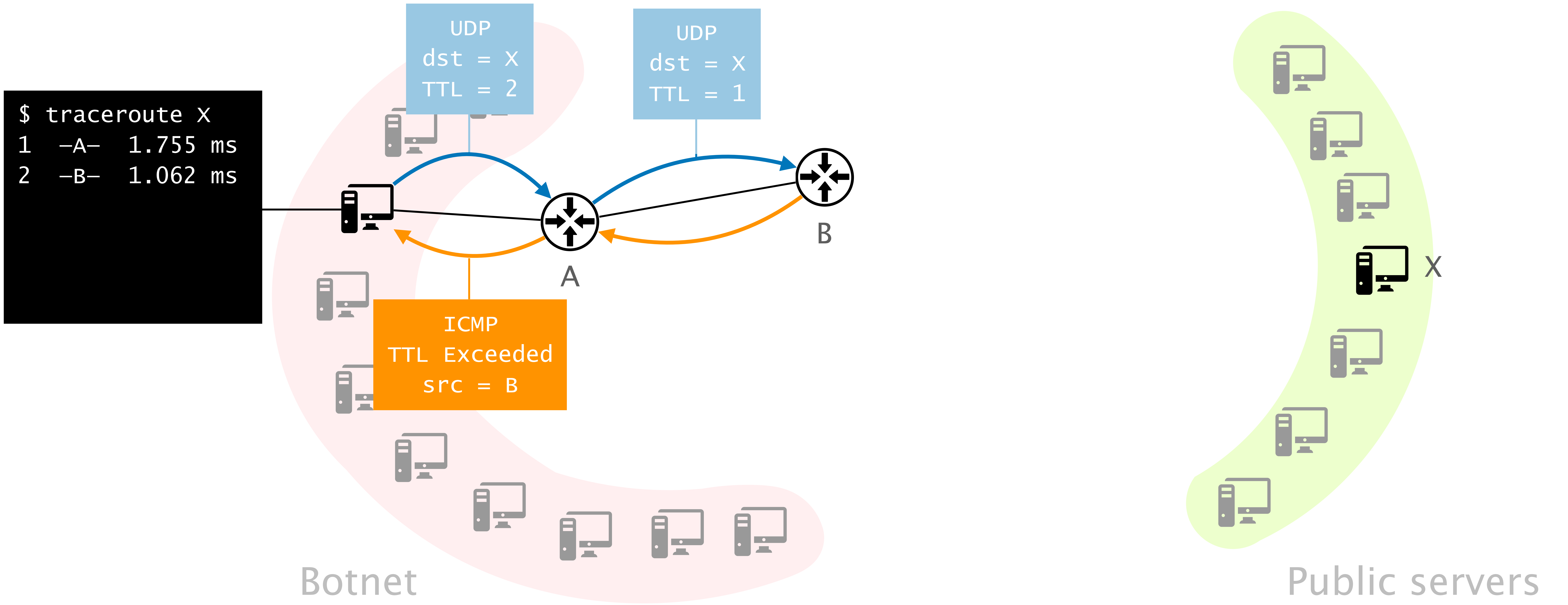


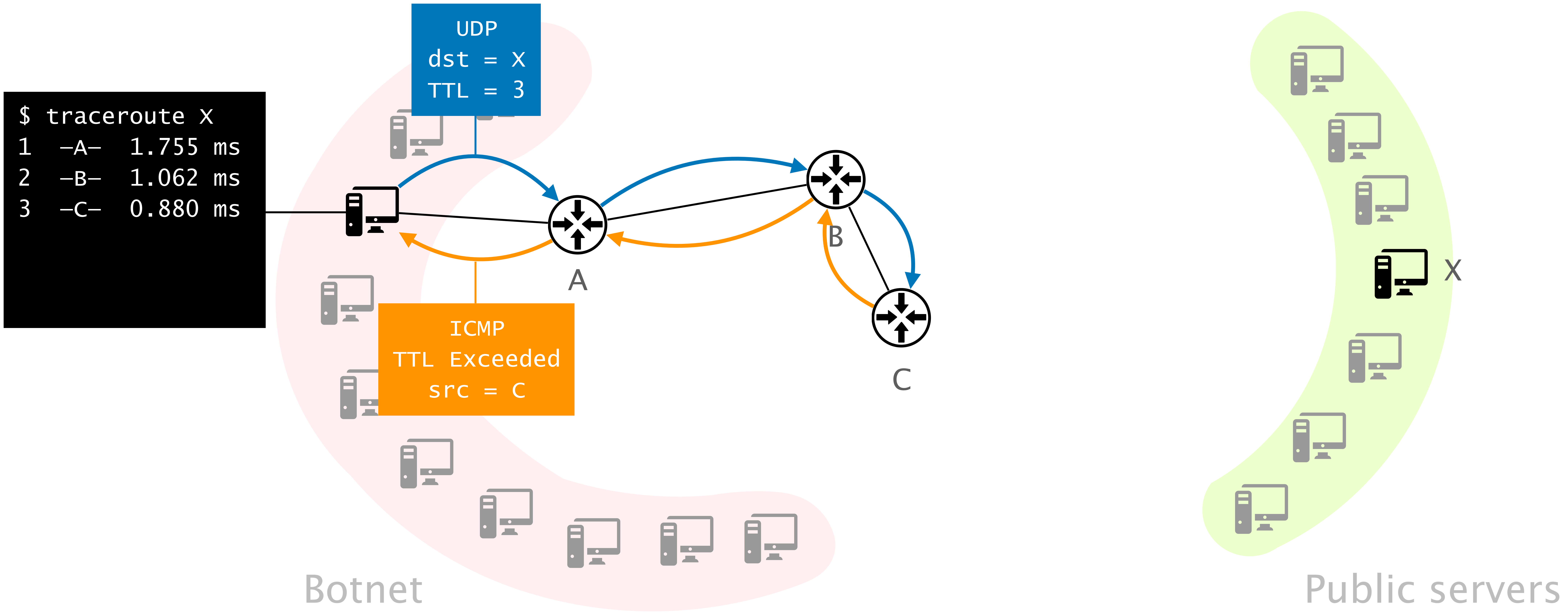
$ traceroute X
1
UDP
dst = X
TTL = 1
X
Botnet
Public servers

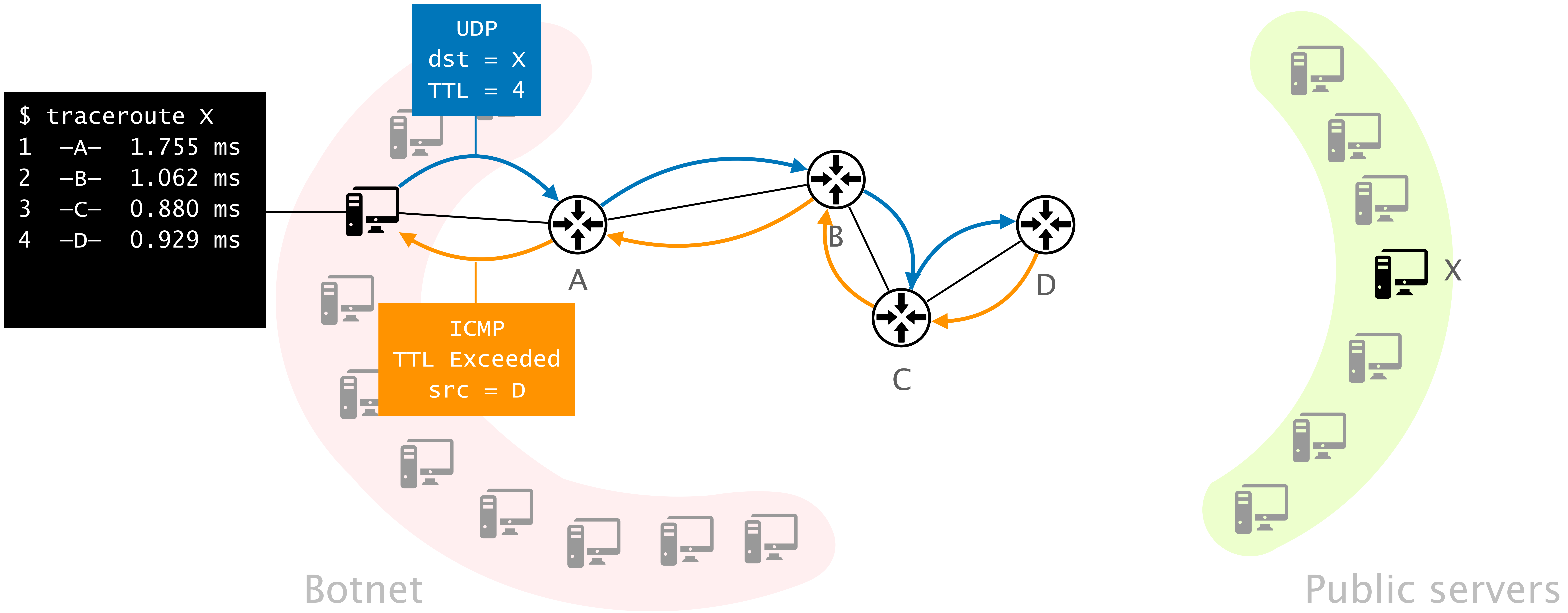

$ traceroute X
1 -A- 1.755 ms
UDP
dst = X
TTL = 1
A
ICMP
TTL Exceeded
src = A
X
Botnet
Public servers

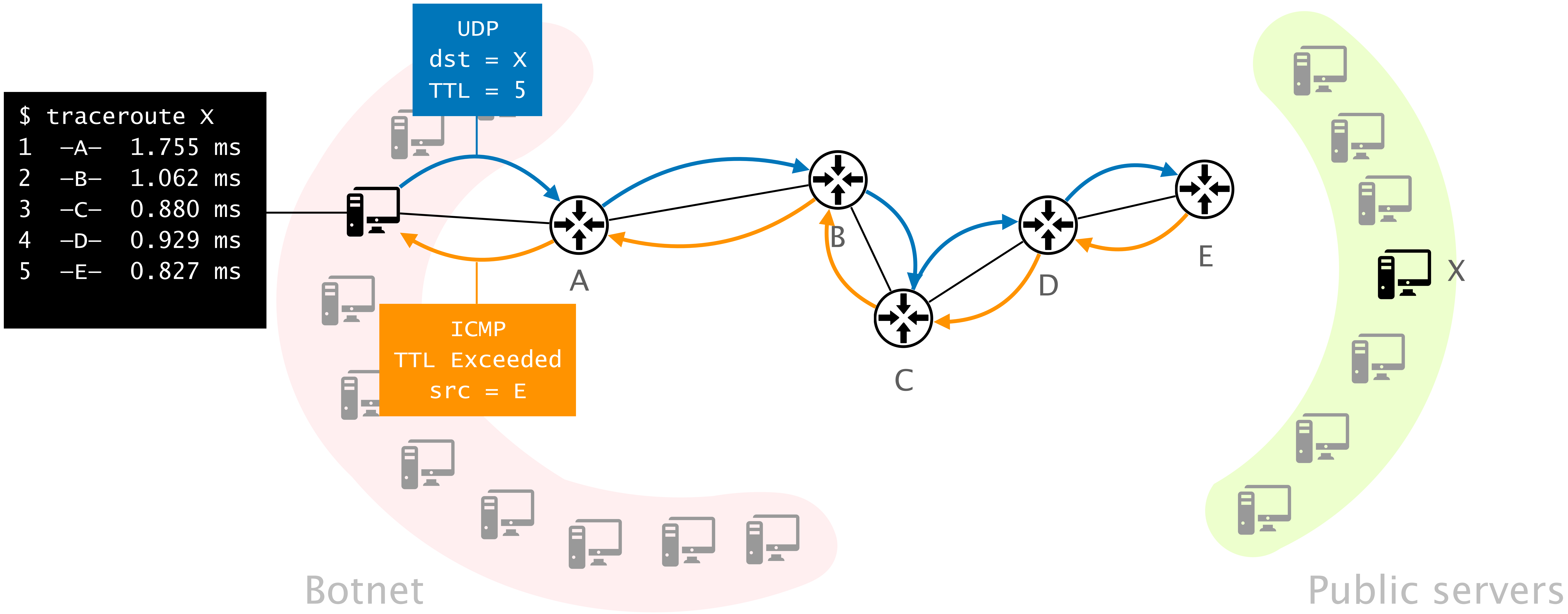

$ traceroute X
1 -A- 1.755 ms
2
UDP
dst = X
TTL = 2
A
X
Botnet
Public servers

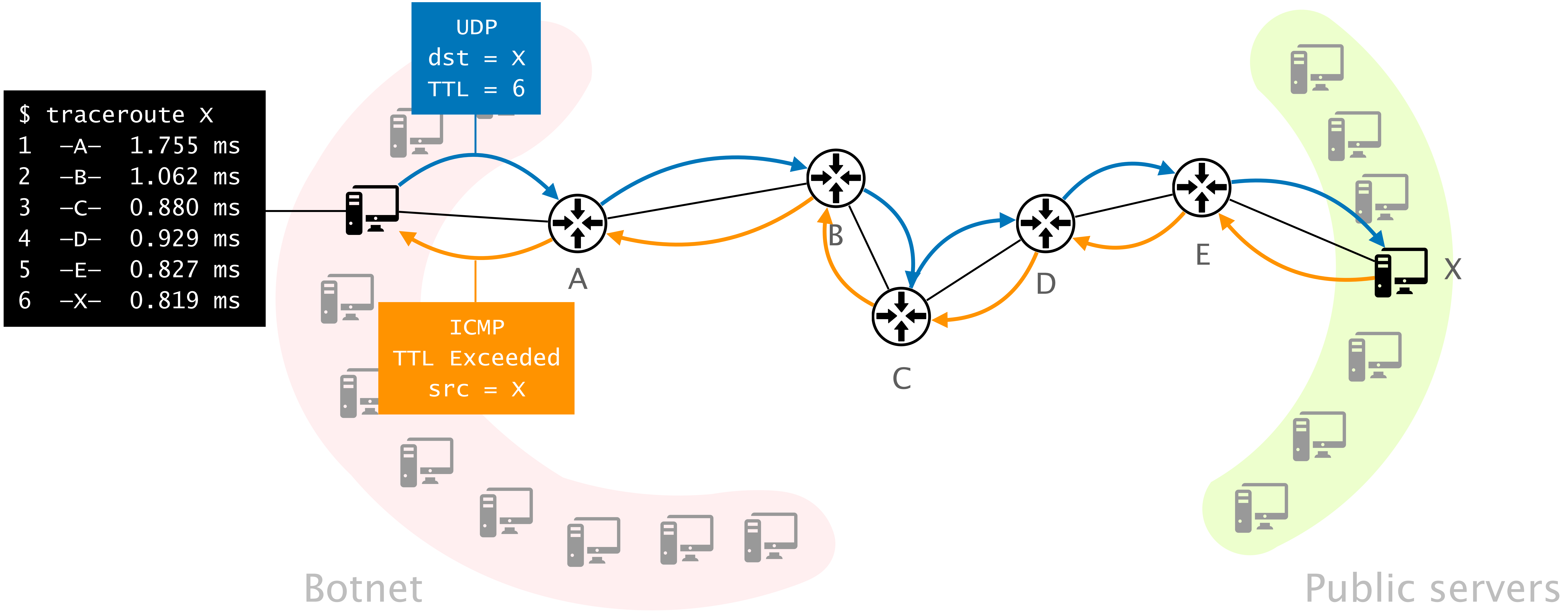

$ traceroute X
1 -A- 1.755 ms
2
UDP
dst = X
TTL = 2
UDP
dst = X
TTL = 1
A
X
Botnet
Public servers


$ traceroute X
1 -A- 1.755 ms
2 -B- 1.062 ms
UDP
dst = X
TTL = 2
UDP
dst = X
TTL = 1
ICMP
TTL Exceeded
src = B
A
B
X
Botnet
Public servers

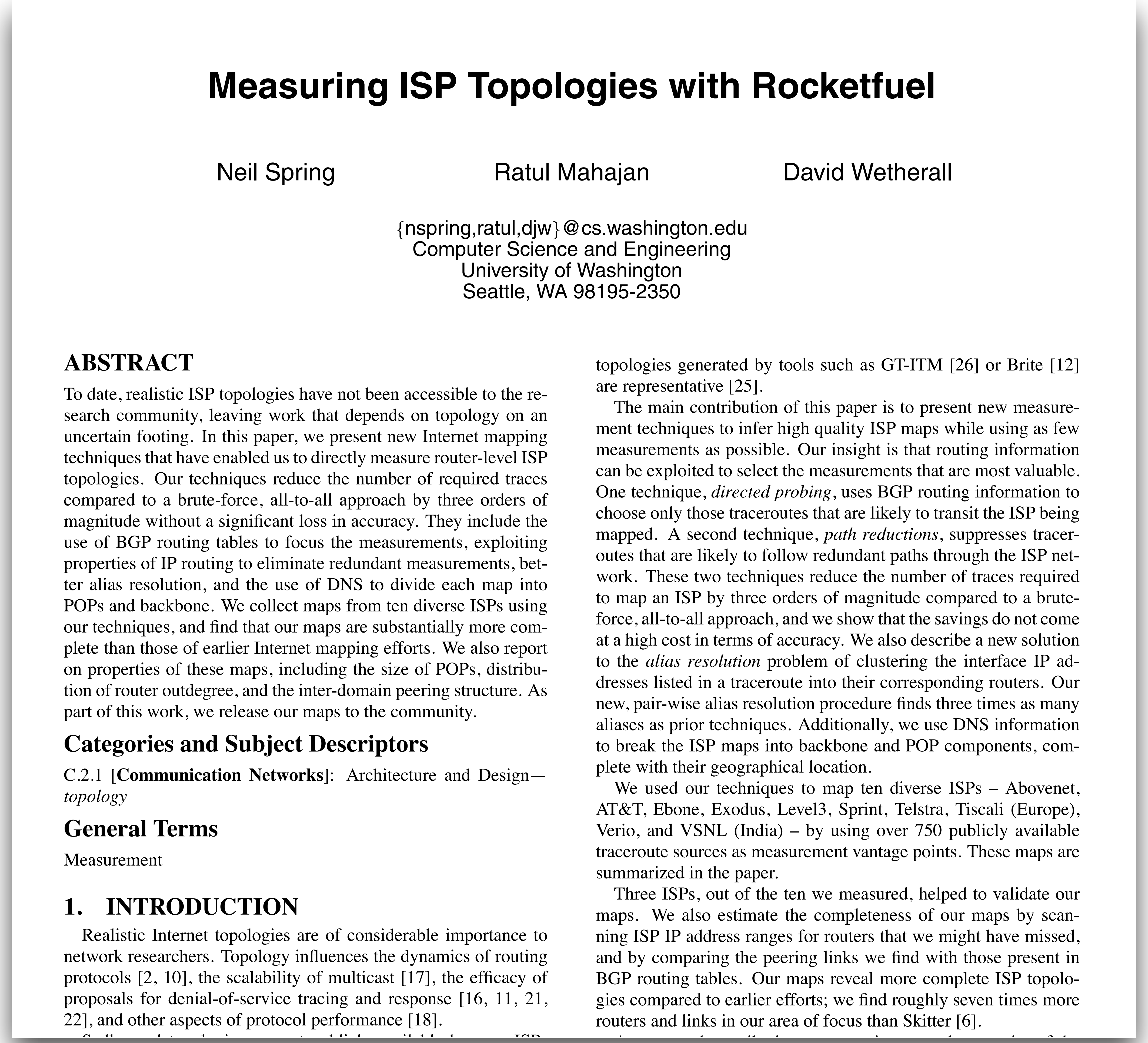

$ traceroute X
1 -A- 1.755 ms
2 -B- 1.062 ms
3 -C- 0.880 ms
UDP
dst = X
TTL = 3
ICMP
TTL Exceeded
src = C
A
B
C
X
Botnet
Public servers


UDP
dst = X
TTL = 4
$ traceroute X
1 –A– 1.755 ms
2 –B– 1.062 ms
3 –C– 0.880 ms
4 –D– 0.929 ms
ICMP
TTL Exceeded
src = D
A
B
C
D
X
Botnet
Public servers


$ traceroute X
1 -A- 1.755 ms
2 -B- 1.062 ms
3 -C- 0.880 ms
4 -D- 0.929 ms
5 -E- 0.827 ms
UDP
dst = X
TTL = 5
ICMP
TTL Exceeded
src = E
A
B
C
D
E
X
Botnet
Public servers


$ traceroute X
1 –A– 1.755 ms
2 –B– 1.062 ms
3 –C– 0.880 ms
4 –D– 0.929 ms
5 –E– 0.827 ms
6 –X– 0.819 ms
UDP
dst = X
TTL = 6
ICMP
TTL Exceeded
src = X
A
B
C
D
E
X
Botnet
Public servers

Learning large topologies by combining many path measurements



Measuring ISP Topologies with Rocketfuel

Neil Spring Ratul Mahajan David Wetherall

{nspring,ratul,djw}@cs.washington.edu
Computer Science and Engineering
University of Washington
Seattle, WA 98195-2350

ABSTRACT

To date, realistic ISP topologies have not been accessible to the research community, leaving work that depends on topology on an uncertain footing. In this paper, we present new Internet mapping techniques that have enabled us to directly measure router-level ISP topologies. Our techniques reduce the number of required traces compared to a brute-force, all-to-all approach by three orders of magnitude without a significant loss in accuracy. They include the use of BGP routing tables to focus the measurements, exploiting properties of IP routing to eliminate redundant measurements, better alias resolution, and the use of DNS to divide each map into POPs and backbone. We collect maps from ten diverse ISPs using our techniques, and find that our maps are substantially more complete than those of earlier Internet mapping efforts. We also report on properties of these maps, including the size of POPs, distribution of router outdegree, and the inter-domain peering structure. As part of this work, we release our maps to the community.

Categories and Subject Descriptors

C.2.1 [**Communication Networks**]: Architecture and Design—*topology*

General Terms

Measurement

1. INTRODUCTION

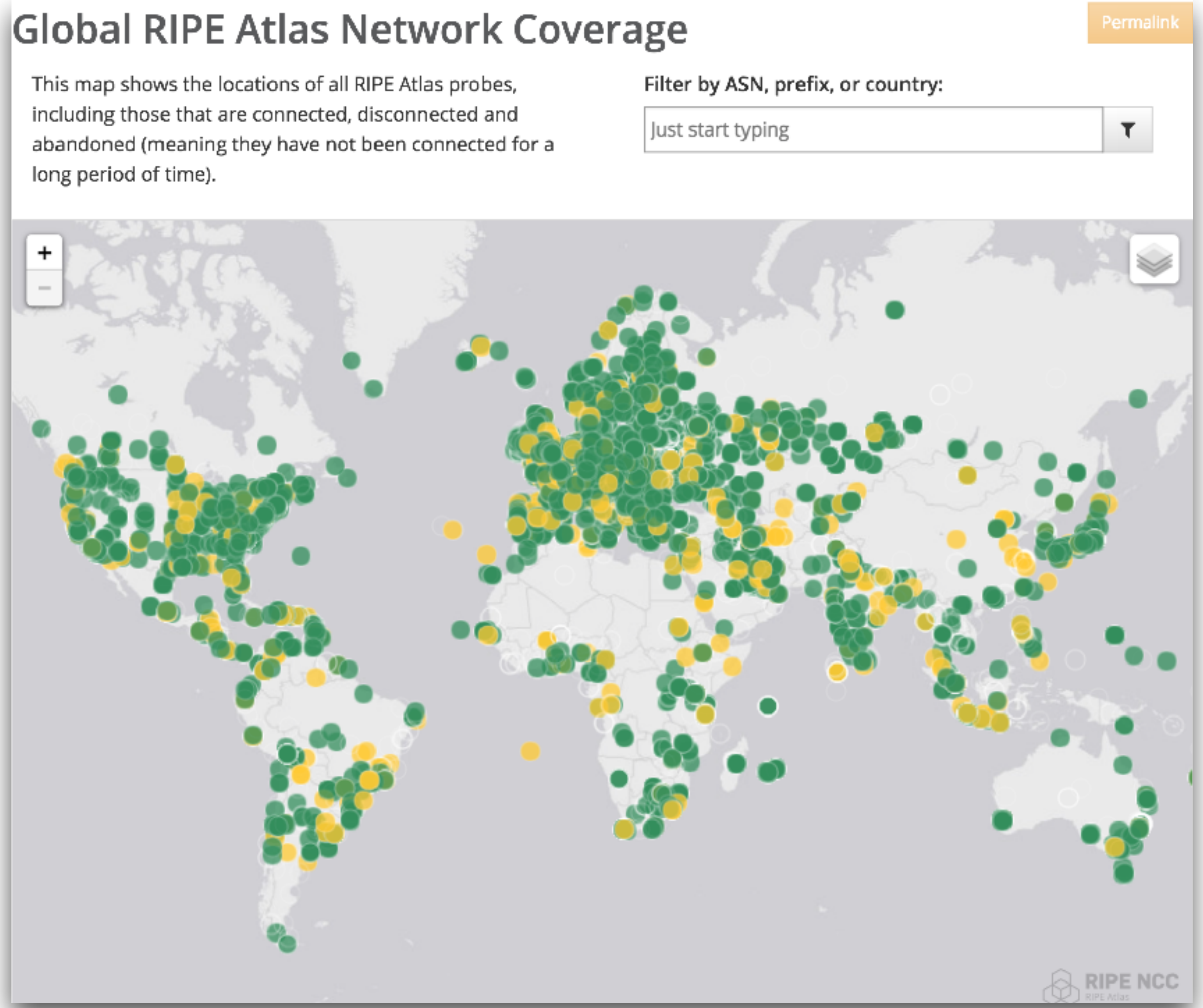
Realistic Internet topologies are of considerable importance to network researchers. Topology influences the dynamics of routing protocols [2, 10], the scalability of multicast [17], the efficacy of proposals for denial-of-service tracing and response [16, 11, 21, 22], and other aspects of protocol performance [18].

topologies generated by tools such as GT-ITM [26] or Brite [12] are representative [25].

The main contribution of this paper is to present new measurement techniques to infer high quality ISP maps while using as few measurements as possible. Our insight is that routing information can be exploited to select the measurements that are most valuable. One technique, *directed probing*, uses BGP routing information to choose only those traceroutes that are likely to transit the ISP being mapped. A second technique, *path reductions*, suppresses traceroutes that are likely to follow redundant paths through the ISP network. These two techniques reduce the number of traces required to map an ISP by three orders of magnitude compared to a brute-force, all-to-all approach, and we show that the savings do not come at a high cost in terms of accuracy. We also describe a new solution to the *alias resolution* problem of clustering the interface IP addresses listed in a traceroute into their corresponding routers. Our new, pair-wise alias resolution procedure finds three times as many aliases as prior techniques. Additionally, we use DNS information to break the ISP maps into backbone and POP components, complete with their geographical location.

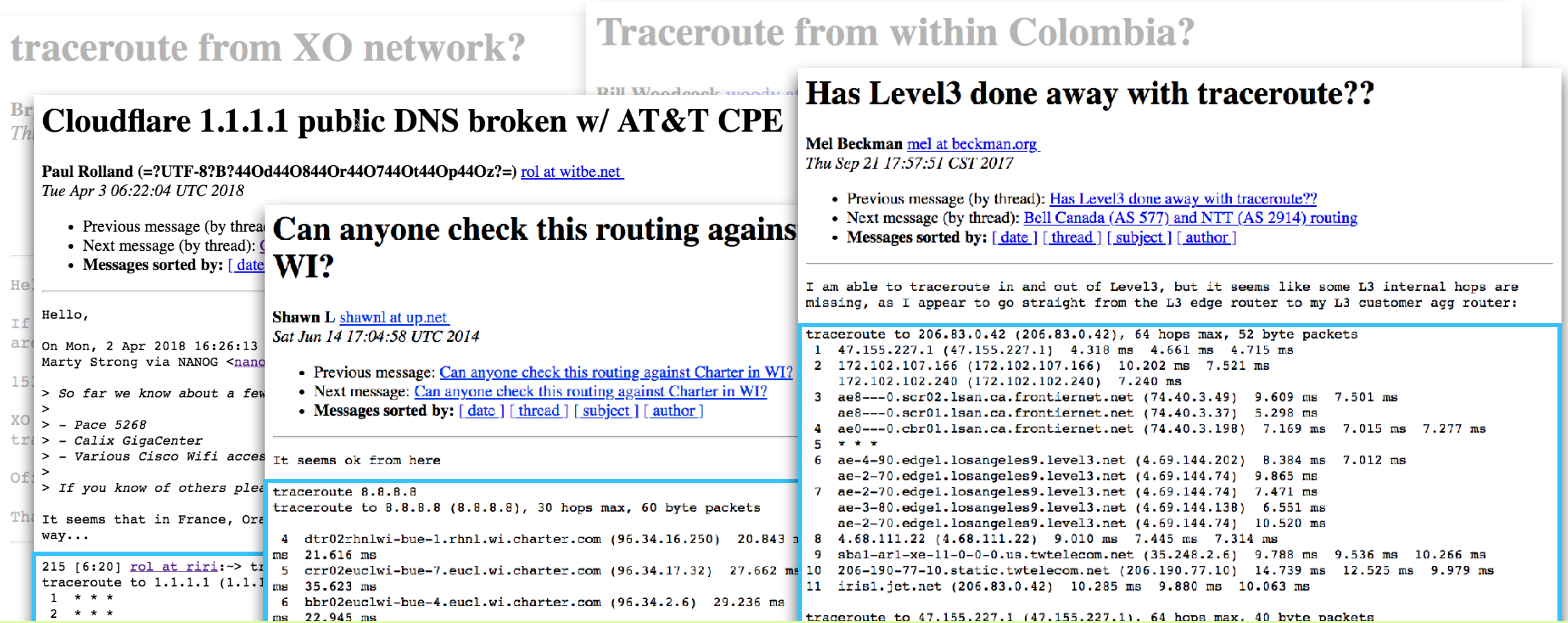
We used our techniques to map ten diverse ISPs – Abovenet, AT&T, Ebone, Exodus, Level3, Sprint, Telstra, Tiscali (Europe), Verio, and VSNL (India) – by using over 750 publicly available traceroute sources as measurement vantage points. These maps are summarized in the paper.

Three ISPs, out of the ten we measured, helped to validate our maps. We also estimate the completeness of our maps by scanning ISP IP address ranges for routers that we might have missed, and by comparing the peering links we find with those present in BGP routing tables. Our maps reveal more complete ISP topologies compared to earlier efforts; we find roughly seven times more routers and links in our area of focus than Skitter [6].



So the solution is to hide the topology?

yes, but…



traceroute is an essential debugging tool parts of

So the solution is to hide the topology?

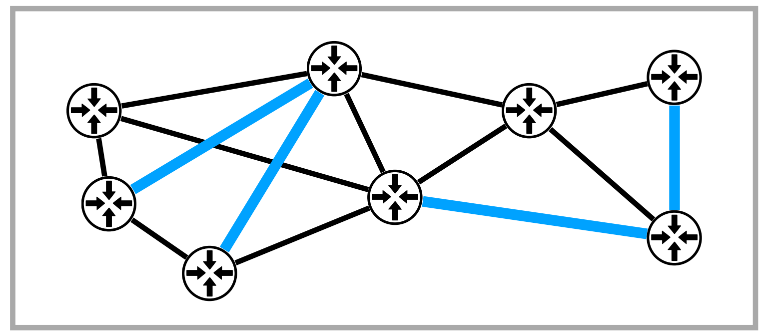
which parts?

parts of

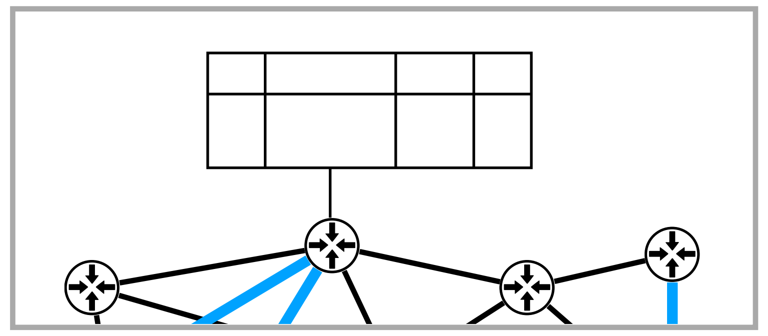
So the solution is to hide the topology?

how?

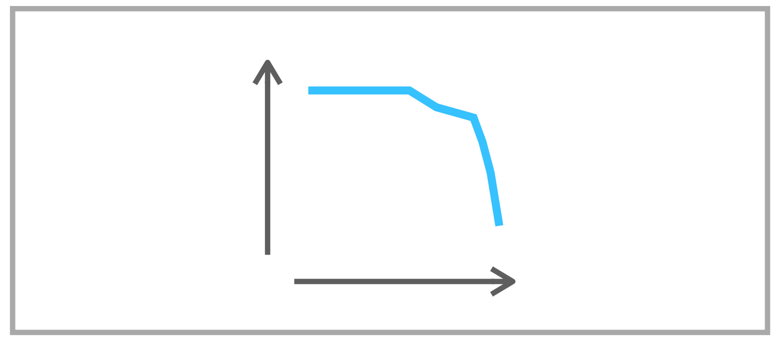
NetHide: Secure and Practical Network Topology Obfuscation



NetHide computes a secure virtual topology
that is similar to the physical topology



NetHide deploys the virtual topology
using programmable networks



NetHide works for realistic topologies
and maintains the utility of debugging tools

Reactive and proactive strategies to mitigate link-flooding attacks

- Reactive

- Proactive

Reactive and proactive strategies to mitigate link-flooding attacks

- Reactive act upon detecting a LFA
  [CoDef, Liaskos, SPIFFY]

  - cannot prevent LFAs
  - impact on production traffic

- Proactive

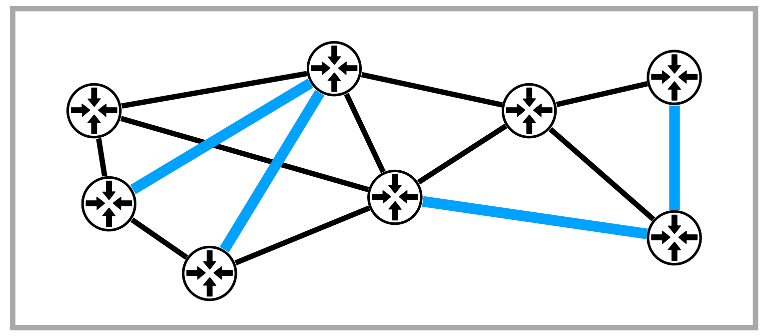
Reactive and proactive strategies to mitigate link-flooding attacks

- Reactive — act upon detecting a LFA
  [CoDef, Liaskos, SPIFFY]
  - cannot prevent LFAs
  - impact on production traffic
- Proactive — Aim at preventing LFAs
  [HoneyNet, Linkbait, Trassare]
  - make debugging tools unusable

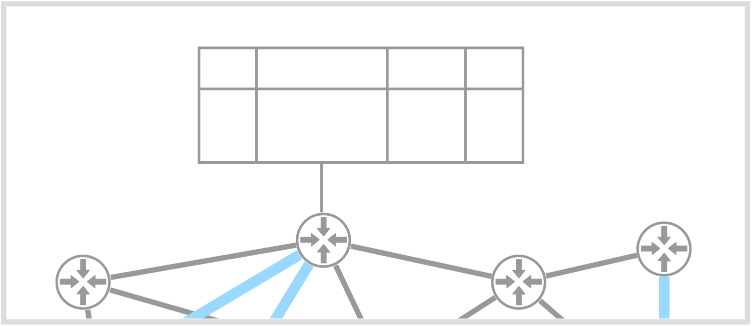
Reactive and proactive strategies to mitigate link-flooding attacks

- Reactive — act upon detecting a LFA
  [CoDef, Liaskos, SPIFFY]
  - cannot prevent LFAs
  - impact on production traffic
- Proactive — Aim at preventing LFAs
  [HoneyNet, Linkbait, Trassare]
  - ~~make debugging tools unusable~~ [NetHide]

NetHide: Secure and Practical Network Topology Obfuscation



NetHide computes a secure virtual topology
that is similar to the physical topology



NetHide deploys the virtual topology
using programmable networks

NetHide works for realistic topologies
and maintains the utility of debugging tools

Topology obfuscation
as an optimization problem

Given the **physical topology P**,

compute a **virtual topology V**, such that

- **V** is robust against link-flooding attacks
- **V** has maximal practicality

Topology obfuscation
as an optimization problem

Given the **physical topology P**,

compute a **virtual topology V**, such that

- **V** is robust against link-flooding attacks
- **V** has maximal practicality

Attacker can run flows between pairs of routers

Attacker

- controls a set of hosts
  i.e. a botnet
- has a budget of flows to run
  flows between nodes (routers)
- has no prior knowledge about topology
  learns topology e.g. through traceroute

A topology is robust against LFAs,
if the flow density of each link does not exceed its capacity

$$\forall l \in L' :$$

Links in **V**

A topology is robust against LFAs, if the flow density of each link does not exceed its capacity

Flow density of the link
(# of flows using it)

$$\forall l \in L' : \mathrm{fd}(l)$$

Links in **V**

A topology is robust against LFAs, if the flow density of each link does not exceed its capacity

Flow density of the link
(# of flows using it)

$$\forall l \in L' : \mathrm{fd}(l) \leq \mathrm{c}(l)$$

Links in **V**

Capacity of the link
(max # of flows)

Two basic strategies for attacking the virtual topology despite obfuscation

- Invert obfuscation
  and attack based on physical topology
  - Infeasible (more later)

- “guess” a promising attack
  based on the virtual topology
  - Incurs high overhead for attacker (see paper)

Topology obfuscation
as an optimization problem

Given the **physical topology P**,

compute a **virtual topology V**, such that

- **V** is robust against link-flooding attacks
- **V** has maximal practicality

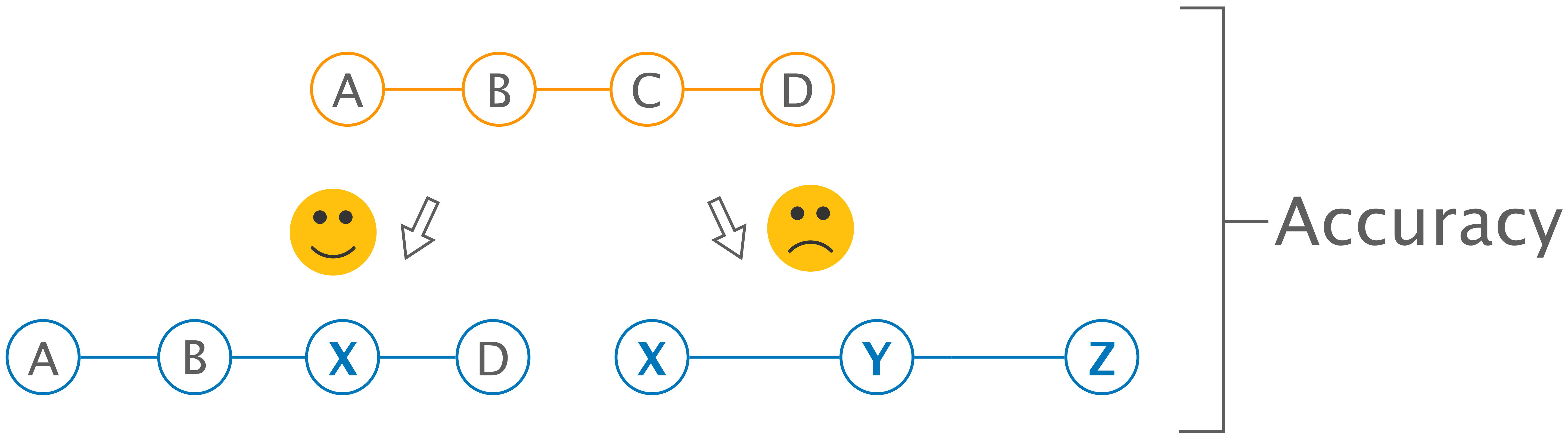
Accuracy and utility measure the closeness of P and V

- Virtual paths are similar to physical paths

- Failures in P are reflected in V

Accuracy and utility measure the closeness of **P** and **V**

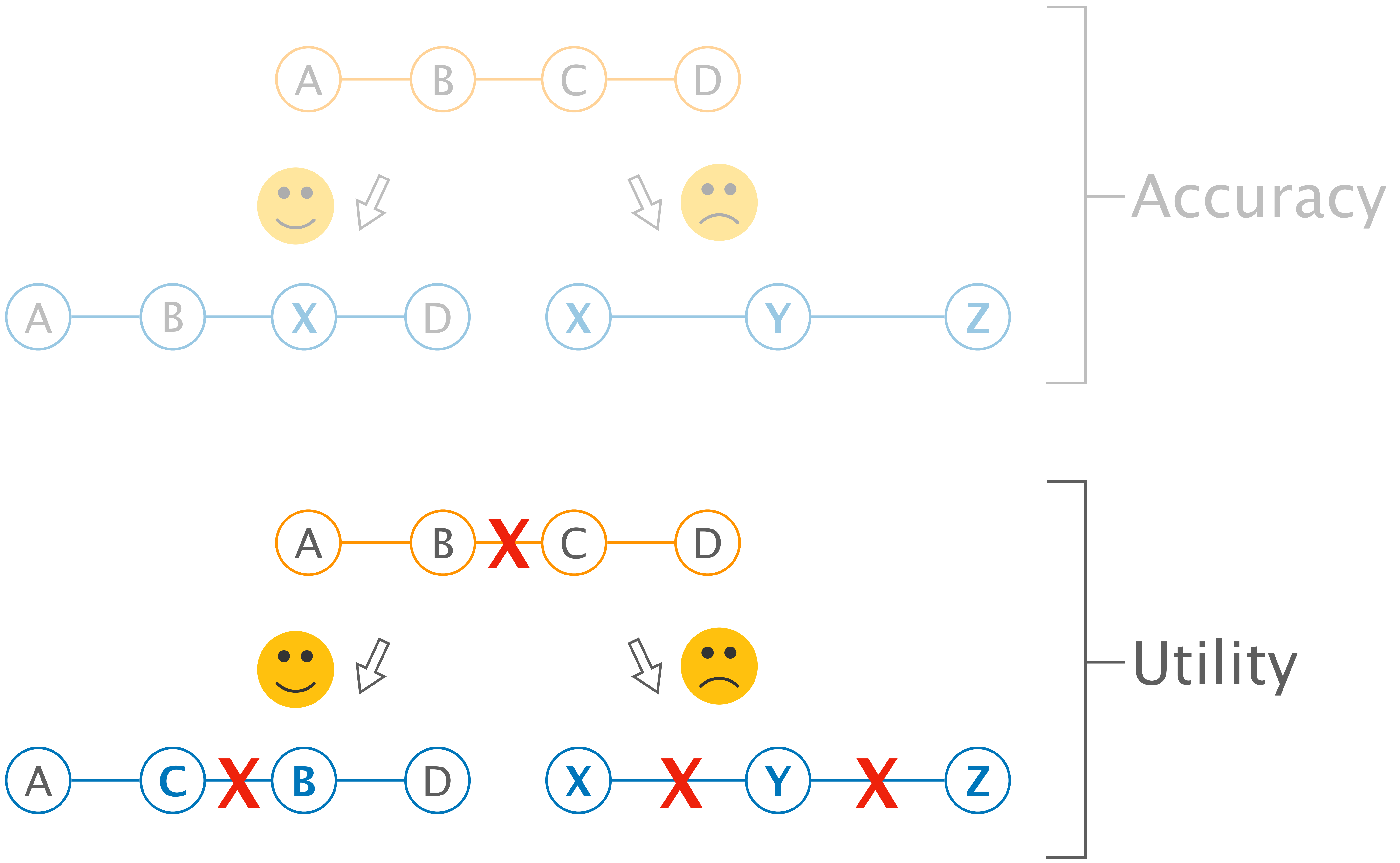
- Virtual paths are similar to physical paths



- Failures in **P** are reflected in **V**

Accuracy and utility measure the closeness of P and V

- Virtual paths are similar to physical paths
- Failures in **P** are reflected in **V**



Topology obfuscation as an optimization problem

Given the **physical topology P**,

compute a **virtual topology V**, such that

- **V** is robust against link-flooding attacks
- **V** has maximal practicality

NetHide optimizes over a random sample of solutions to improve performance and security

all possible solutions

$$\mathcal{O}(N^N)$$

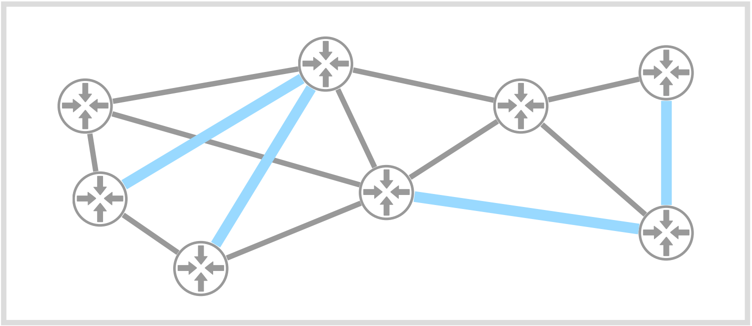
topology size
(# of routers)

random sample

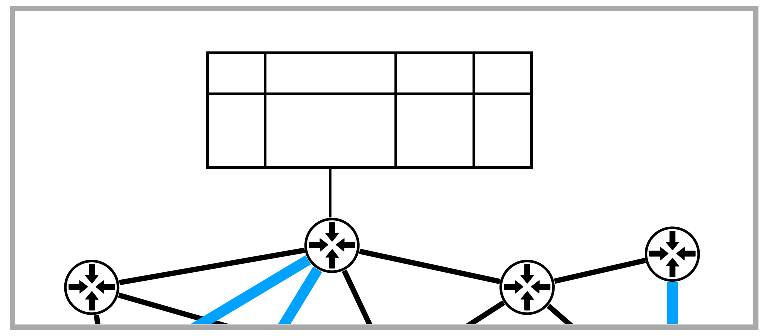
$$\mathcal{O}(N)$$

- better performance
- harder to invert obfuscation
- still high accuracy and utility

NetHide: Secure and Practical Network Topology Obfuscation



NetHide computes a secure virtual topology that is similar to the physical topology



NetHide deploys the virtual topology using programmable networks

NetHide works for realistic topologies and maintains the utility of debugging tools

Utility-preserving topology deployment

Deploy the **virtual topology V**, such that

- debugging tools still work
- network performance is not impacted
- it scales to large networks

Utility-preserving topology deployment

Deploy the **virtual topology V**, such that

- debugging tools still work
- network performance is not impacted
- it scales to large networks

Maintaining the utility of debugging tools requires sending packets through the actual network

- ⛔ Answer from a central controller
- ⛔ Answer at the edge
- ⛔ Answer in a virtual clone of the network
- ➡ Answer from the correct device that appears on the path

Utility-preserving topology deployment

Deploy the **virtual topology V**, such that

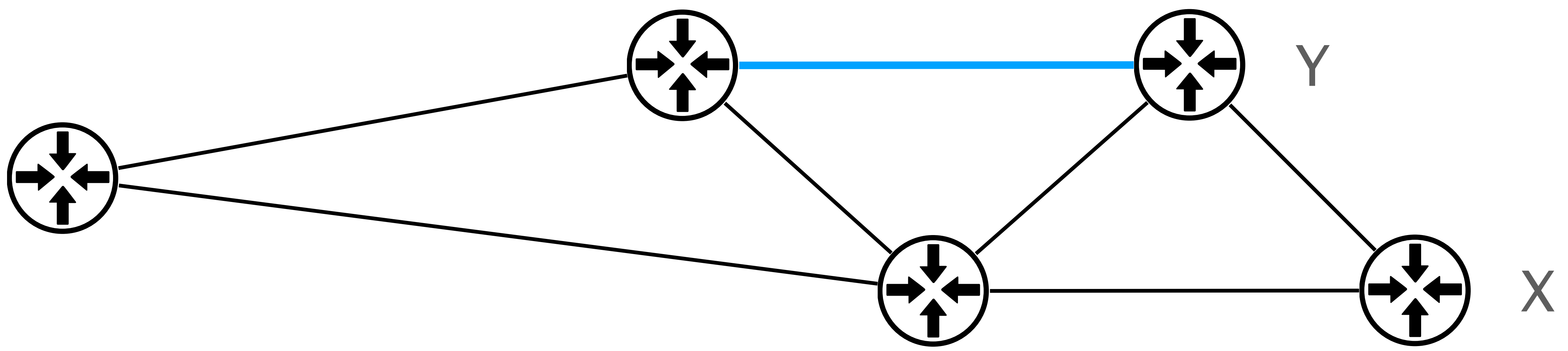
- debugging tools still work
- network performance is not impacted
- it scales to large networks

Programmable network devices allow modifying tracing packets at line rate

- Read & modify packet headers
  e.g. the TTL value
- Basic operations
  e.g. hash functions and checksums
- Add or remove custom headers
  to store information

Programmable network devices
are configured through match+action tables

If I receive a packet to X with TTL = i,
I should send it to Y with TTL = j

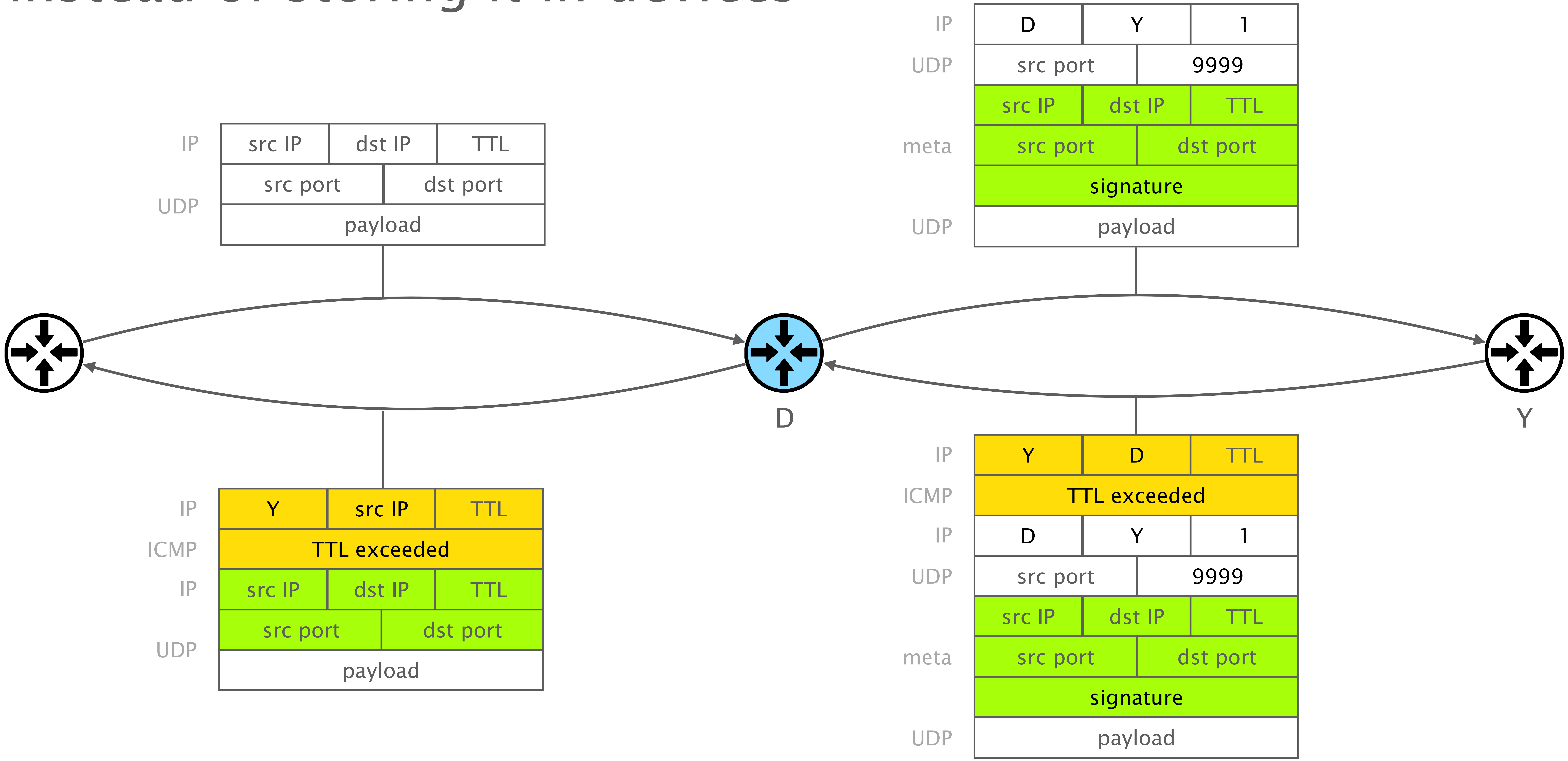


Utility-preserving topology deployment

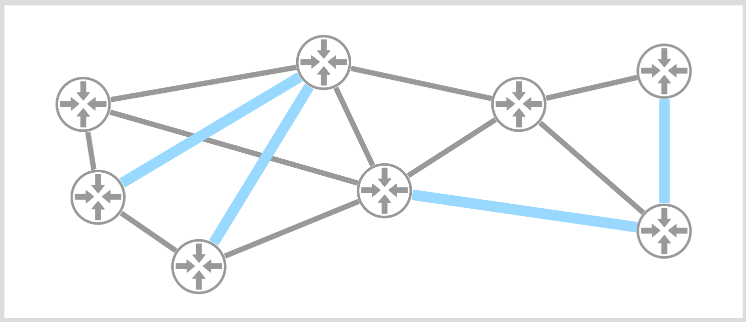
Deploy the **virtual topology V**, such that

- debugging tools still work
- network performance is not impacted
- it scales to large networks

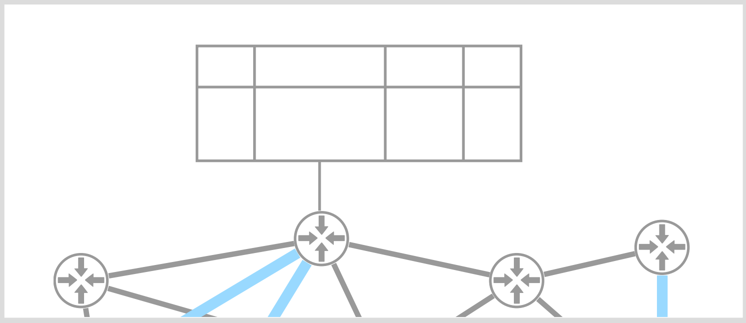
Encoding state in packets instead of storing it in devices



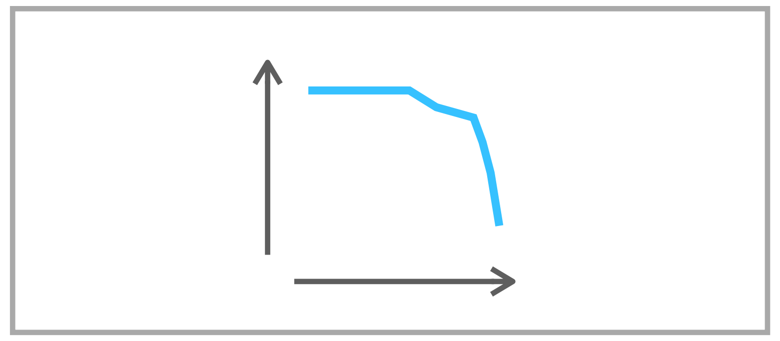
NetHide: Secure and Practical Network Topology Obfuscation



NetHide computes a secure virtual topology
that is similar to the physical topology



NetHide deploys the virtual topology
using programmable networks

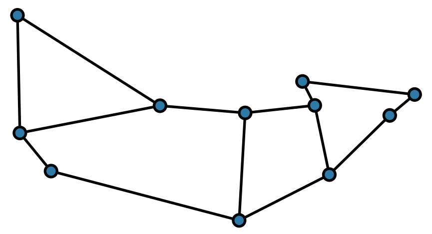


NetHide works for realistic topologies
and maintains the utility of debugging tools

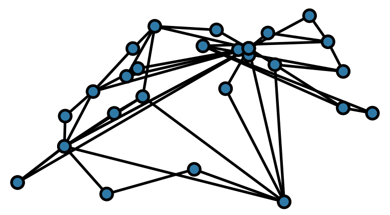
We evaluated various aspects of NetHide based on 3 real topologies

- Accuracy and utility
- Performance
- Timing
- Partial deployment
- Security

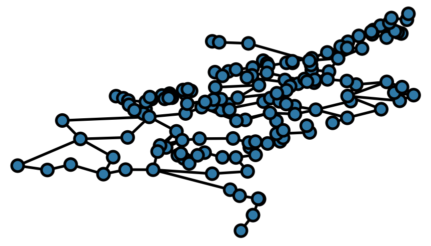
×



Abilene



Switch

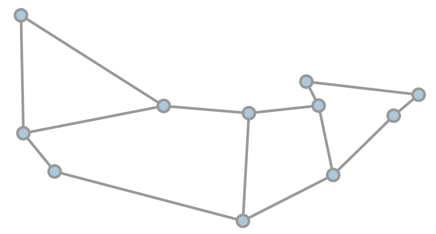


US Carrier

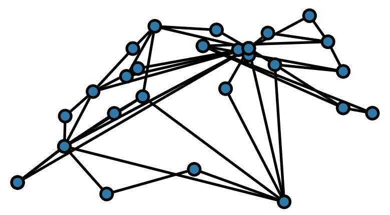
We evaluated various aspects of NetHide based on 3 real topologies

- Accuracy and utility
- Performance
- Timing
- Partial deployment
- Security

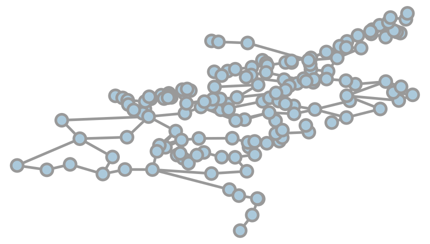
×



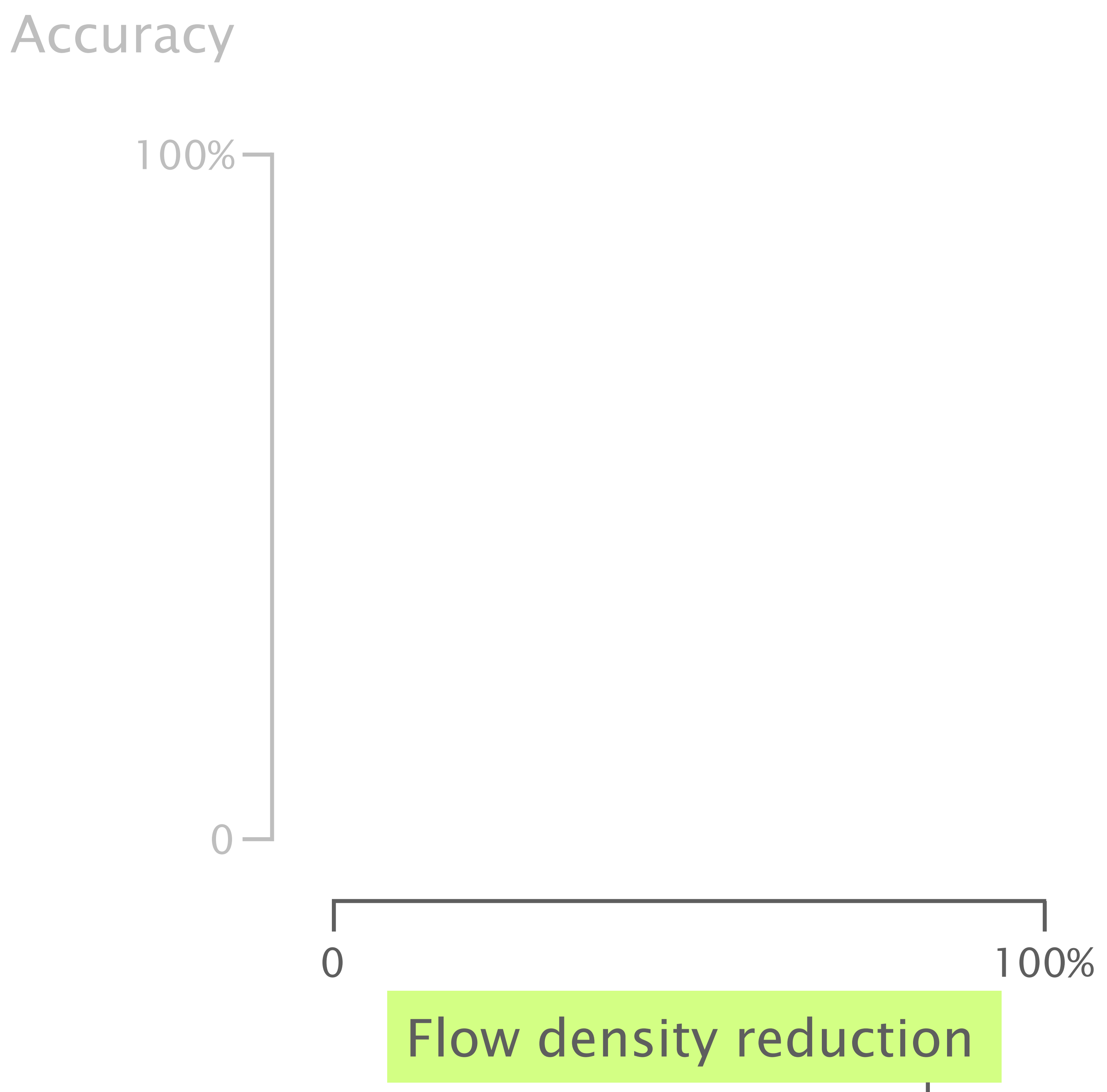
Abilene

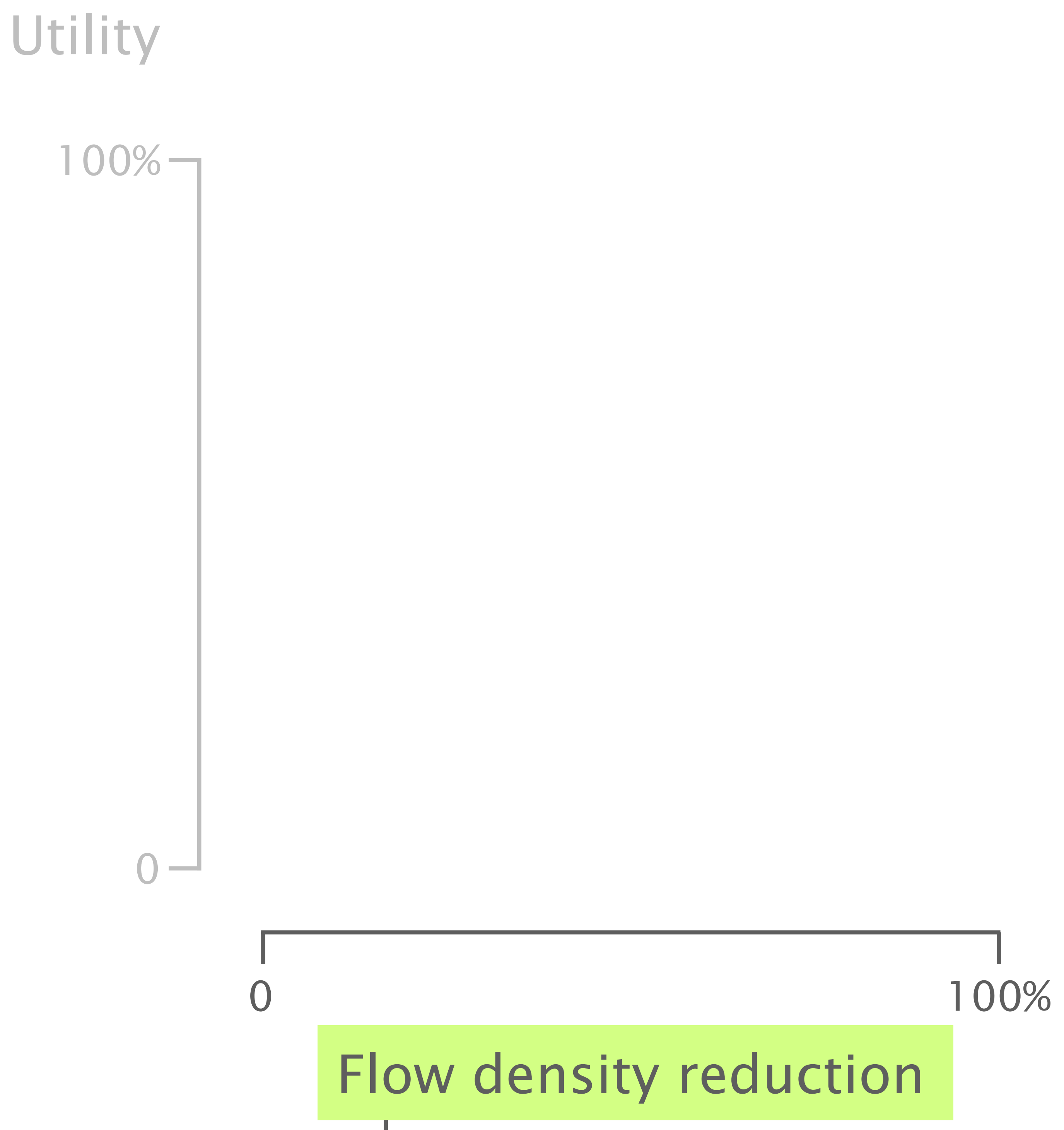


Switch



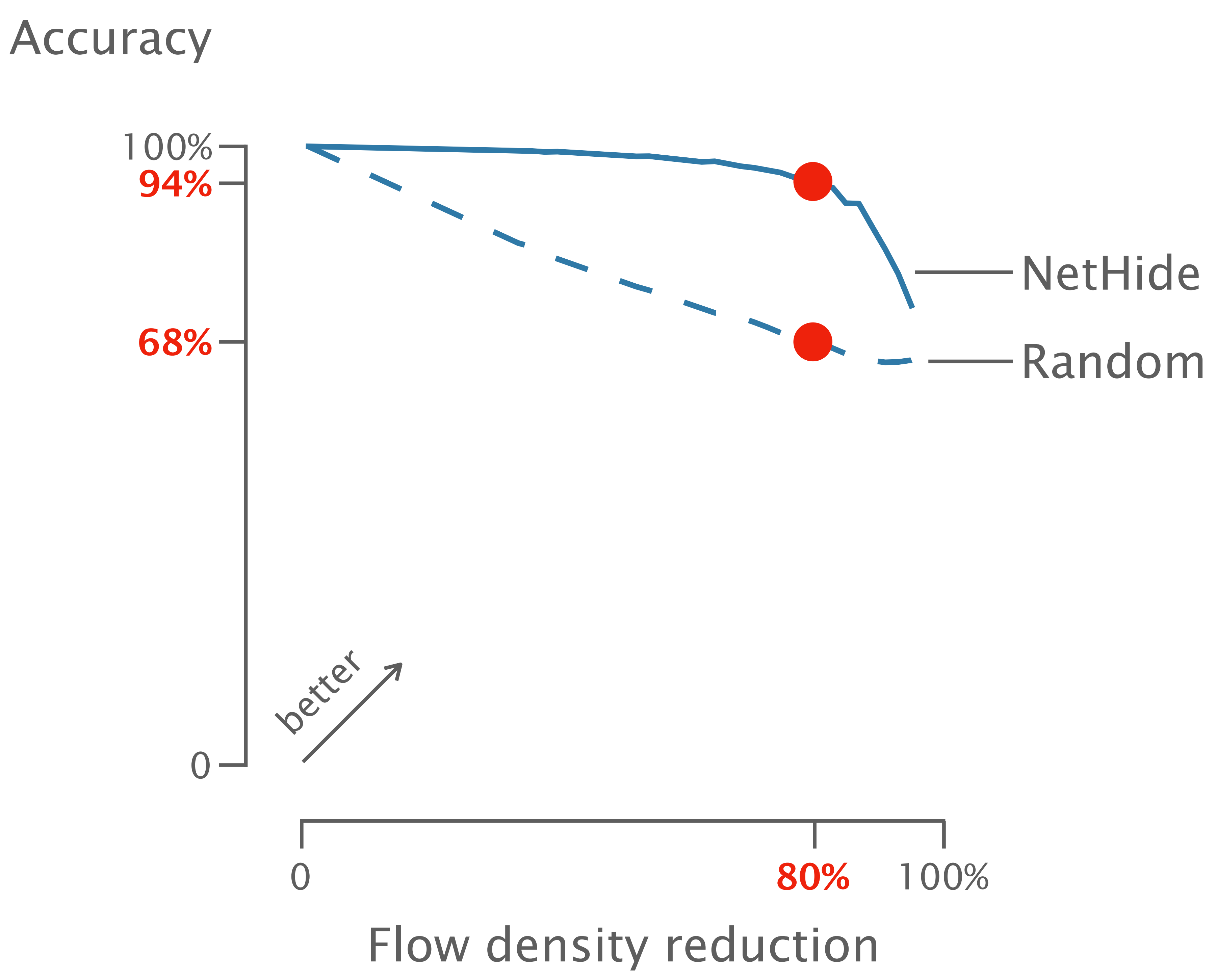
US Carrier

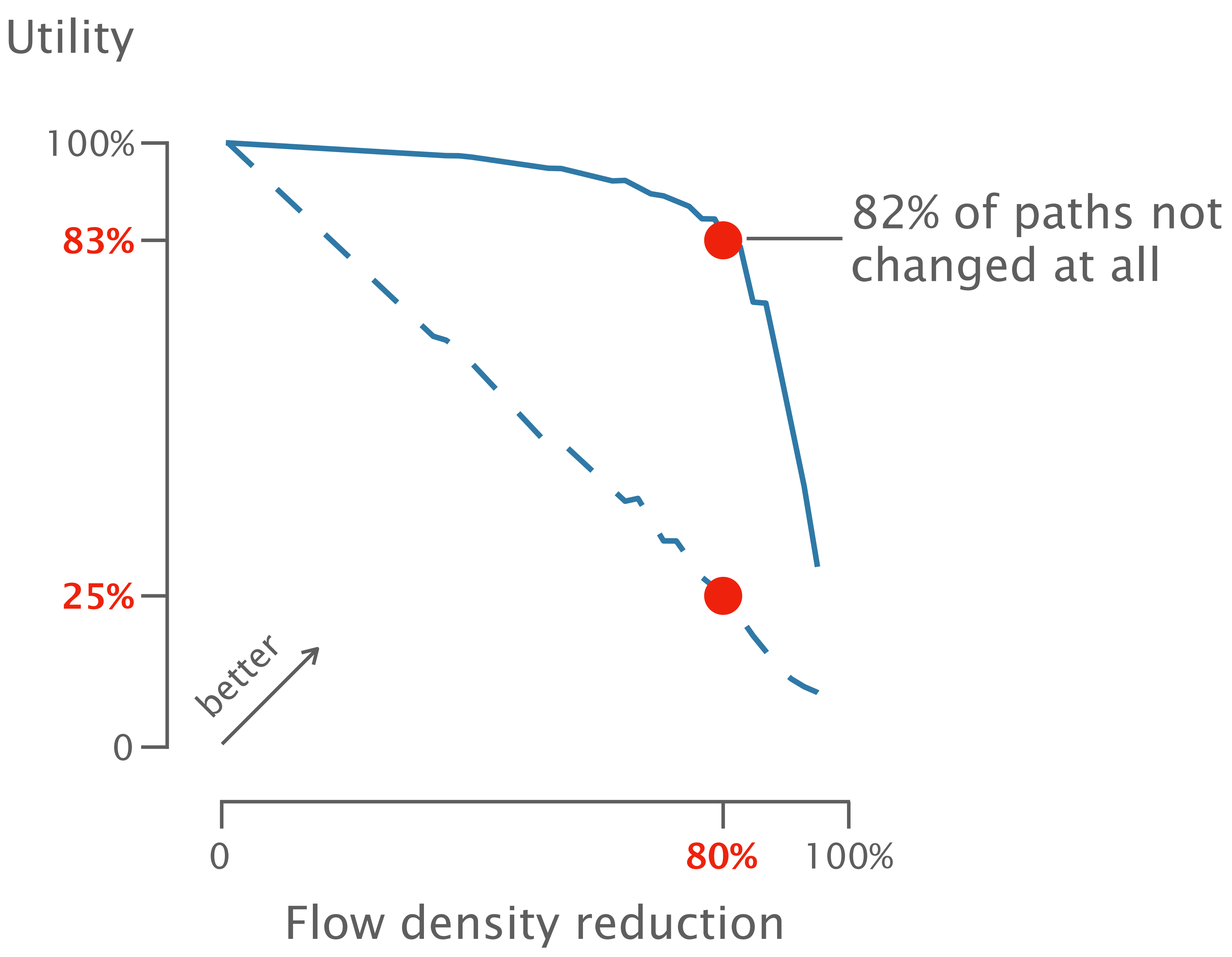




Ratio between the flow density
in the physical and the virtual topology

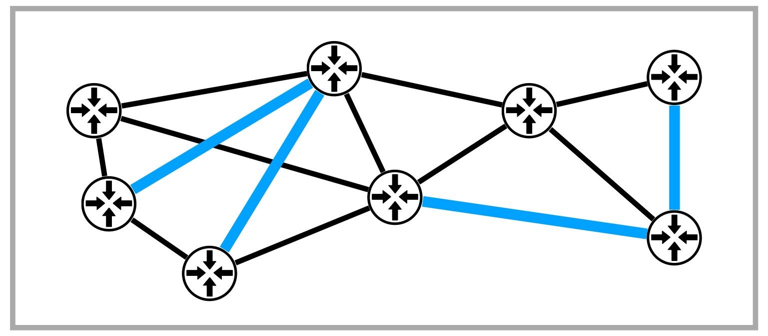
High protection
with small impact on accuracy and utility



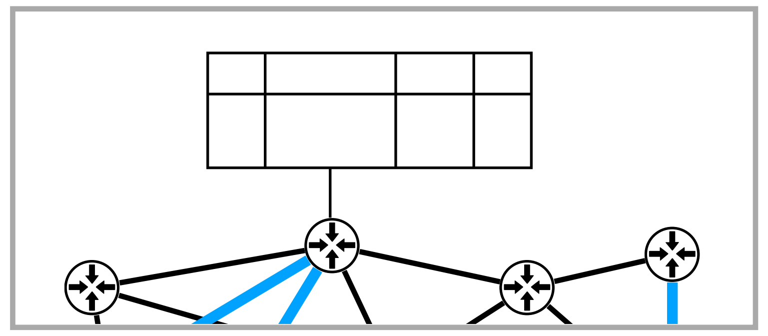


NetHide: Secure and Practical Network Topology Obfuscation

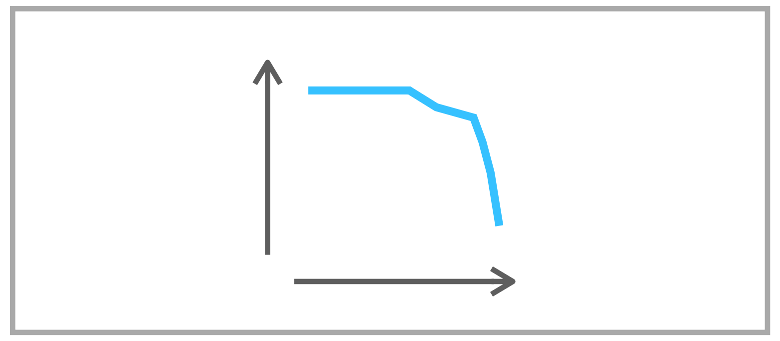
nethide.ethz.ch



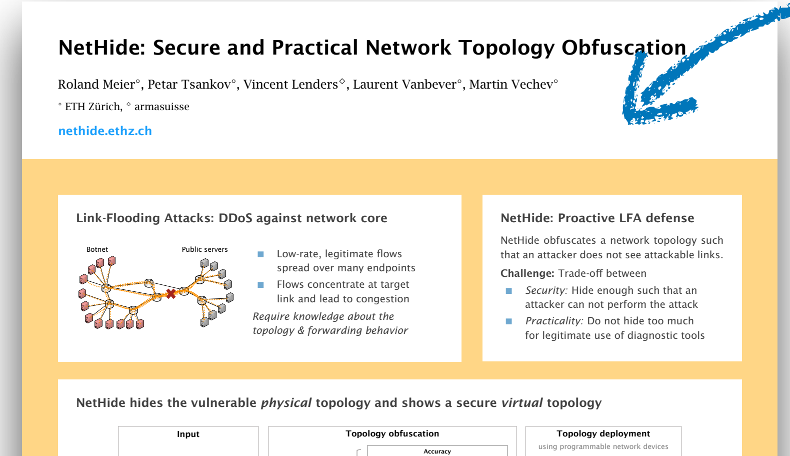
NetHide computes a secure virtual topology that is similar to the physical topology



NetHide deploys the virtual topology using programmable networks



NetHide works for realistic topologies and maintains the utility of debugging tools



Join me at the poster session

ETH zürich

Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

armasuisse