An Architectural Approach for Mitigating Next-Generation Denial of Service Attacks

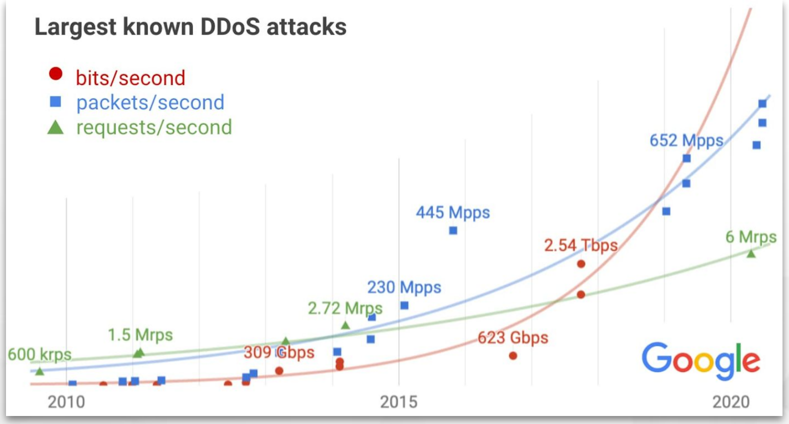
Cody Doucette

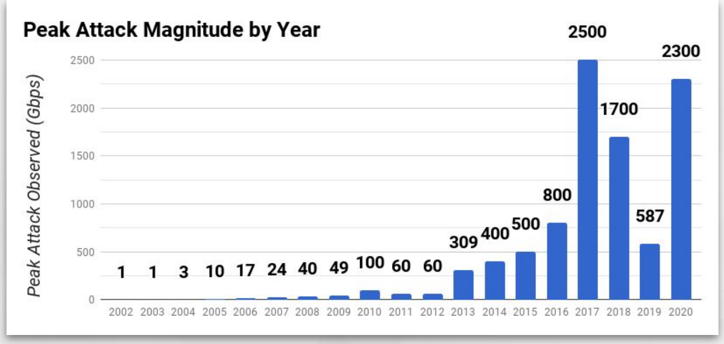
Final Oral Examination
Boston University

December 3, 2020

Relevance of DDoS attacks

By the numbers...





By the headlines...

DDoS attacks increase 542% quarter-over-quarter amid pandemic

Amazon reported sustaining a 2.3 Tbps DDoS attack in 2020

Telegram blames China for 'powerful DDoS attack' during Hong Kong protests

But it could get worse...

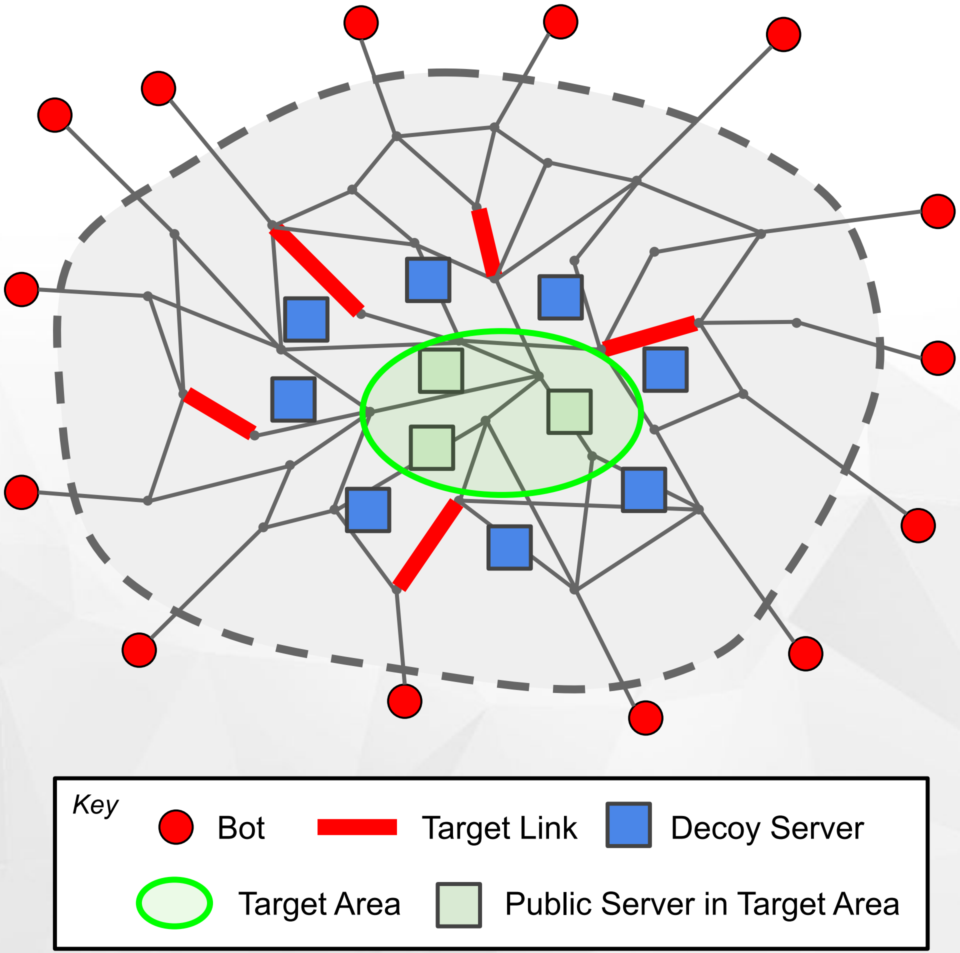
What if I told you an attack could:

- Occur without any attack traffic reaching your servers and services
  ⇒ You don't know it's happening

- Be achieved using low-intensity, legitimate-looking traffic
  ⇒ You can’t figure out who it’s coming from

- Require collaboration between networks to protect and stop
  ⇒ You can’t stop it (by yourself)

Large-scale link attacks



These are properties of large-scale link attacks, (*Crossfire* [S&P '13])

Could it happen?

- It's has happened in the wild!
  ⇒ SpamHaus attack, 2013
  ⇒ ProtonMail attack, 2015

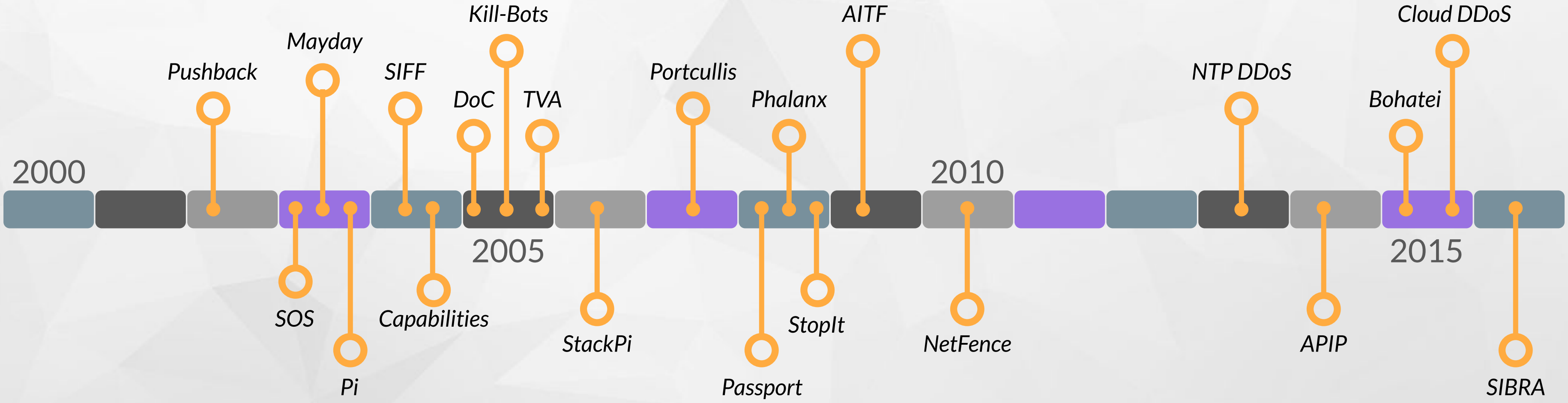
- There are three new developments in the Internet ecosystem which might make large-scale link attacks commonplace:
  - Increasing botnet scale, due to the proliferation of IoT devices
  - Increasing per-bot attack capacity, due to rollout of 5G devices/networks
  - Increasing infrastructure vulnerability, due to the transition to IPv6

- Is there a perfect storm of conditions for a *next generation* of attacks?
  ⇒ Maybe, but let's start from the beginning

Where it started (for us)

We started investigating the literature around DDoS attack defense in 2015

Our goal: figure out why ~~15~~ 20 years of research into mitigation solutions have largely failed to gain traction



Findings

Two main findings:

- DDoS is a fundamentally architectural problem to solve
- Full deployability (not incremental!) must be a top priority

DDoS is Architectural

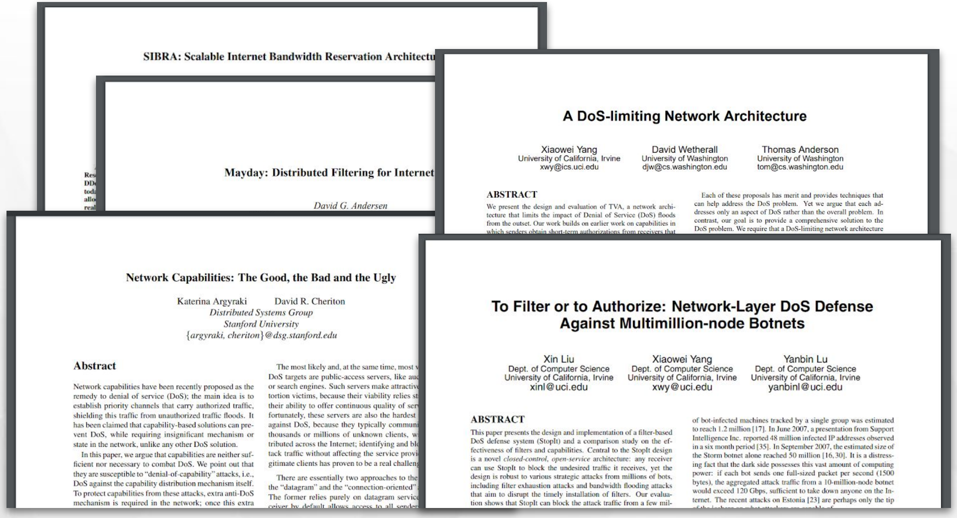
DDoS is a fundamentally architectural problem

- Difficult to forge cooperation between networks (decentralized design)
- Difficult to defend a network against Internet-scale (network of networks)
- Difficult to classify unwanted traffic (open, connectionless network layer)
- Difficult to verify identity of sender (lack of source address verification)

Solution Space

Bridging this gap
requires deployability

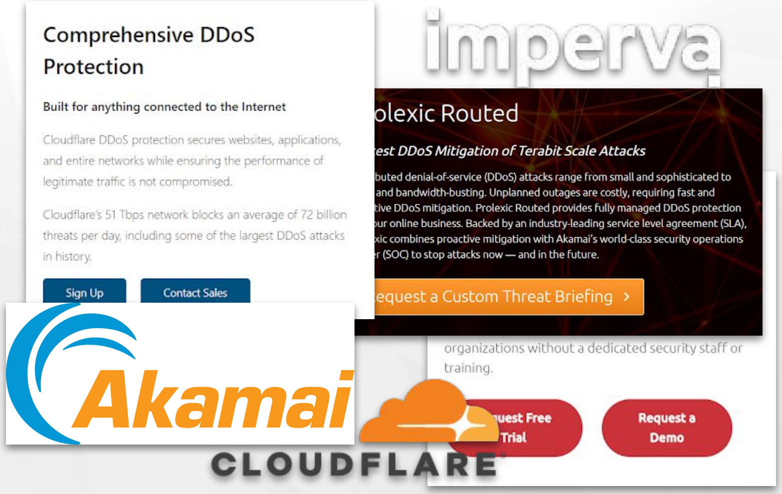
The Research



20 years worth of elegant designs and evaluations that show DDoS is a solvable problem!

… if only the Internet architecture were amenable

The Reality



A DDoS protection market that mostly benefits the entities that control the infrastructure

ISPs, universities, governments have to pay up

Solution: Gatekeeper

We designed a DDoS mitigation system, *Gatekeeper*, to bridge the gap between research and reality

⇒ Incorporates the major lessons learned from decades of research

⇒ Prioritizes deployability as the most important aspect

⇒ Keeps costs low, but enables scaling up as needed

Thesis

Gatekeeper is a mitigation system that neutralizes the architectural issues that make DDoS attacks possible and potent

Even in the case of large-scale link attacks such as Crossfire, which takes advantage of these architectural issues to the extreme, Gatekeeper can break Crossfire's assumptions and provide mitigating maneuvers to hinder it

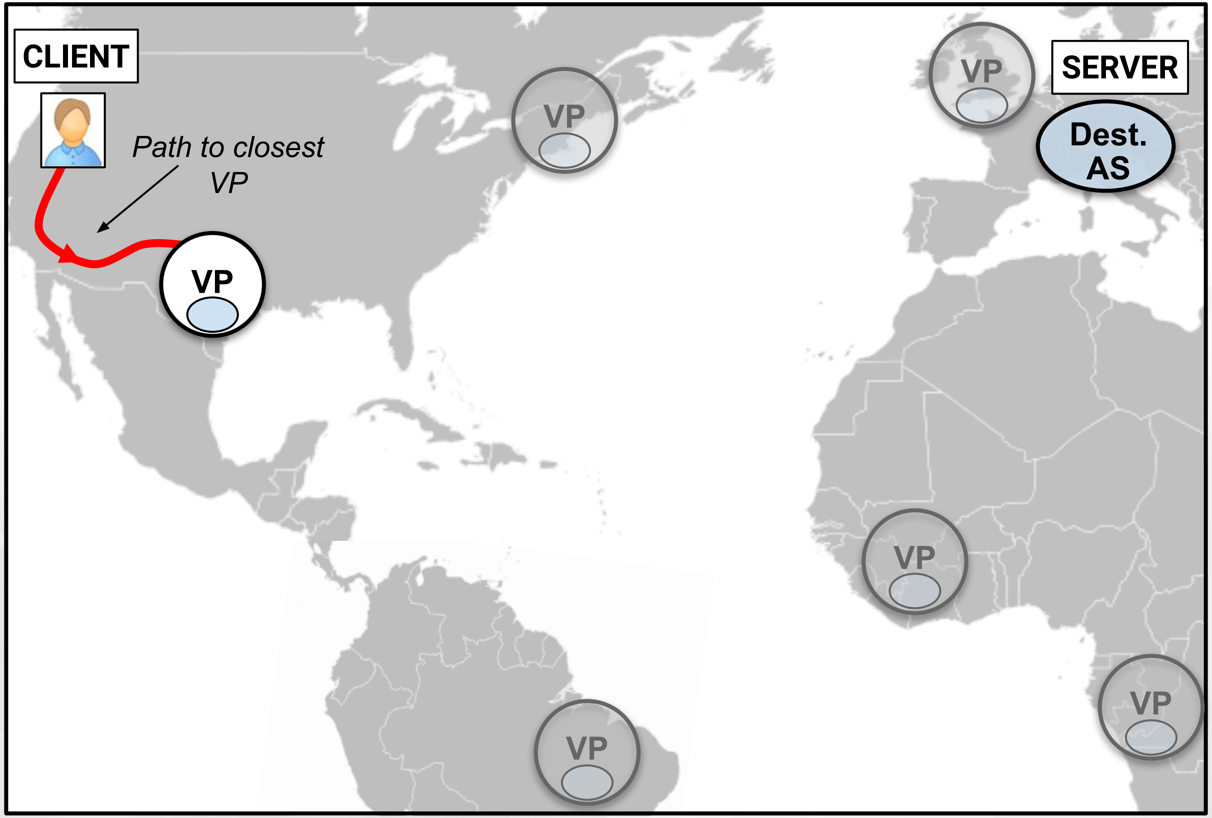
Contributions

- The design, implementation, and evaluation of Gatekeeper, the first open source and fully deployable architectural approach to DDoS mitigation
- A Gatekeeper policy toolkit for network operators, describing basic and advanced techniques that showcase the richness of policy programs
- A cloud and Internet path measurement study that shows Gatekeeper and certain policy techniques may be able to combat large-scale link attacks, an as-of-yet unsolved problem

Agenda

- Background
  - Next-generation attacks
  - Architectural issues and deployability
- Thesis
- Gatekeeper Overview
  - Design
  - Implementation
  - Evaluation
- Gatekeeper Policy Toolkit
- Mitigating Next-Generation Attacks

Gatekeeper's Components



Vantage points:
well-provisioned and geographically distributed locations

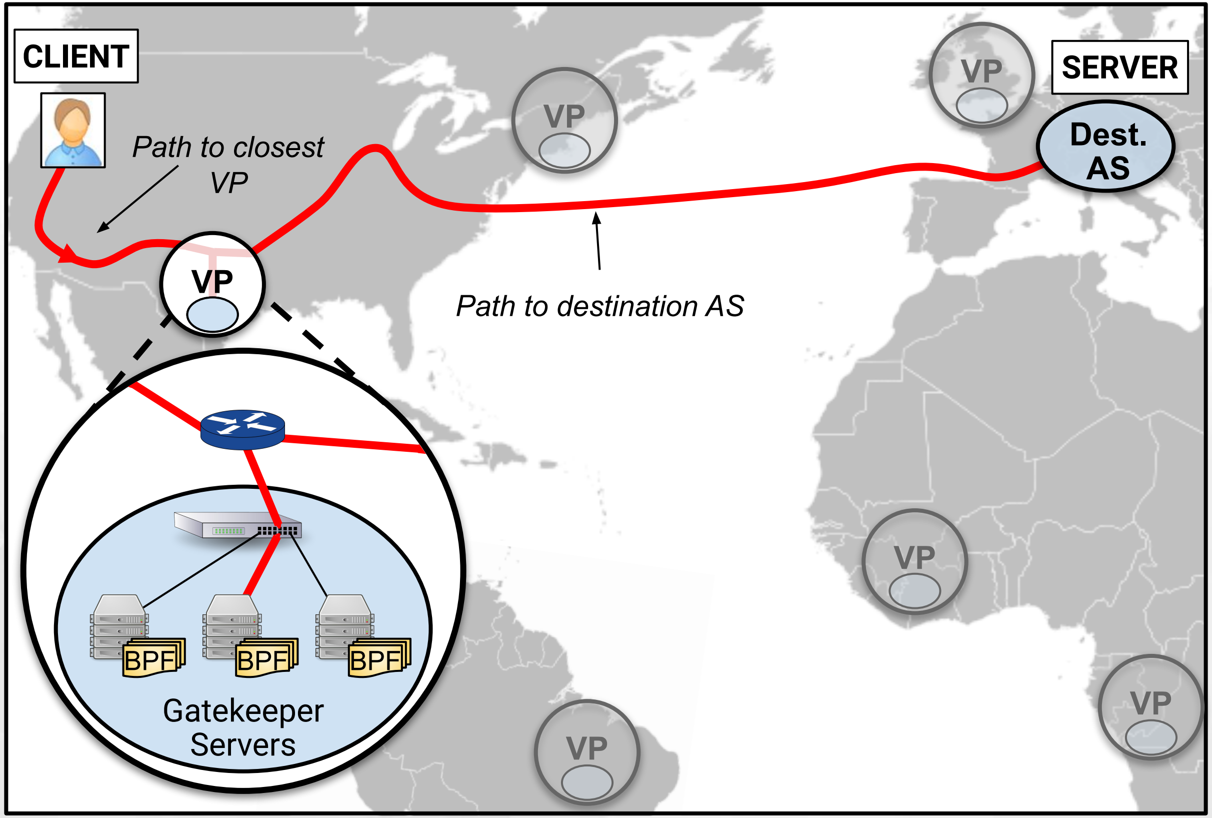
Requirements:

- computing capacity
- cheap ingress bandwidth
- BGP peering
- private links to the protected AS

Examples:

- Internet exchanges
- Peering link
- Some cloud providers

Gatekeeper's Components

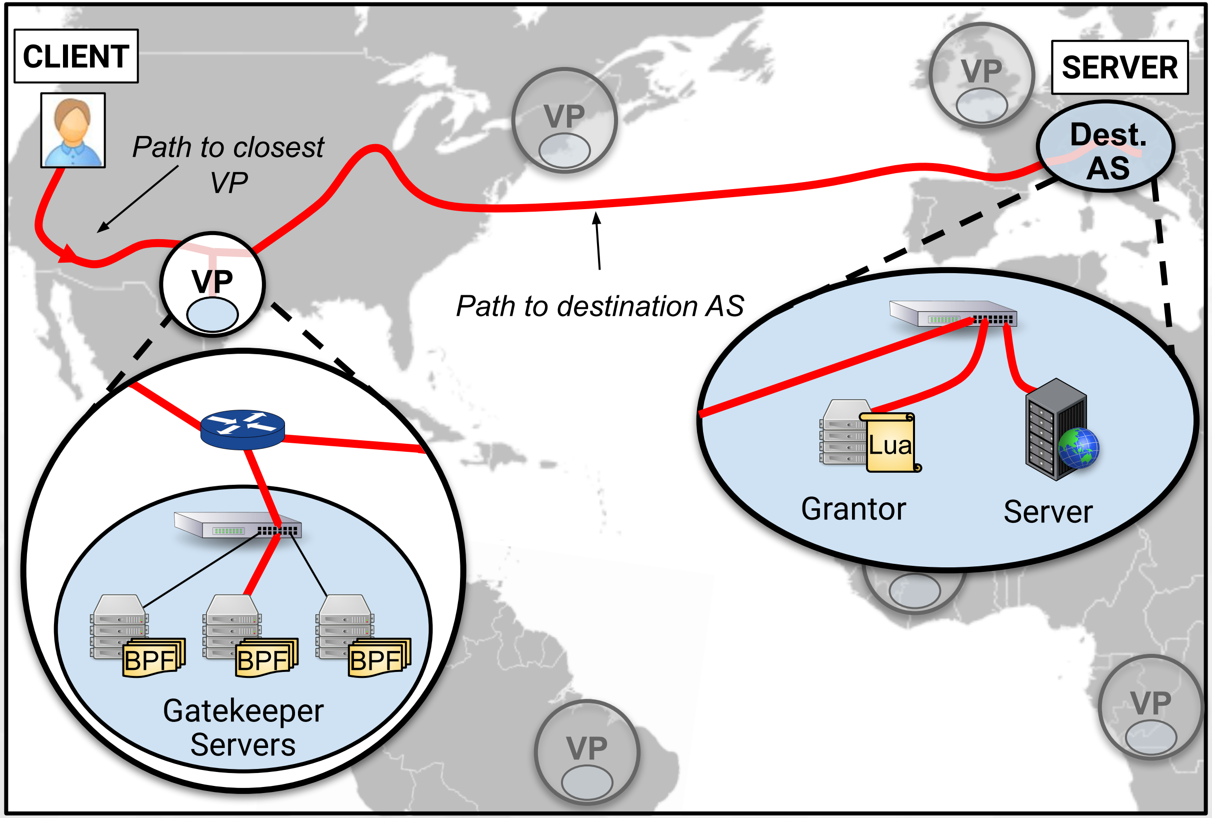


Gatekeeper servers: upstream policy enforcement

Responsibilities:

- Forwarding requests (new flows)
- Dropping or rate-limiting according to per-flow policy enforcement program
- Encapsulating

Gatekeeper's Components



Grantor servers: centralized policy decision making

Responsibilities:

- Making policy decisions about requests and installing those decisions at Gatekeeper
- Decapsulating and sending to destination server

Quick Summary

1. Packets from clients are forwarded to the closest VP
2. Gatekeeper servers send request packets to Grantor servers
3. Grantor servers reject or accept requests based on a policy decision program, and forward granted packets to destinations
4. Grantor servers notify Gatekeeper servers of all their policy decisions
5. Gatekeeper servers enforce the policy decisions using programs

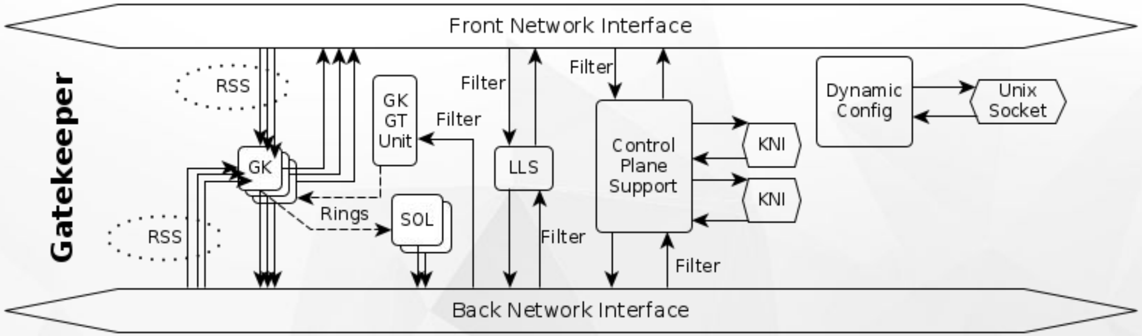
DDoS is Architectural

DDoS is a fundamentally architectural problem

- Difficult to forge cooperation between networks (decentralized design)
  ⇒ Place mitigation system upstream, in strategic vantage points
- Difficult to defend a network against Internet-scale (network of networks)
  ⇒ Make mitigation system distributed and scalable itself
- Difficult to classify unwanted traffic (open, connectionless network layer)
  ⇒ Use network capabilities governed by expressive policies
- Difficult to verify identity of sender (lack of source address verification)
  ⇒ Define policies that leverage vantage point of mitigation system

Implementation Details

Overall goal: implement the system for eventual operational DDoS mitigation use



This thing will be attacked! On purpose!

- Has to be performant, scalable, and fault-tolerant
- Has to support the needs of actual deployment environments

Four-Way Scalability

Gatekeeper can scale in four separate ways:

1. Modular implementation of blocks to scale-up data plane with more threads
2. Support for bonded devices to linearly scale network capacity
3. Gatekeeper and Grantor servers are horizontally scalable
4. Multiple vantage points can be deployed throughout the Internet

Performance Considerations

Gatekeeper leverages many software and hardware techniques for optimizing packet processing

- Kernel bypass (DPDK)
- Batching
- Prefetching
- Branch prediction
- Non-uniform memory access (NUMA)
- EtherType and ntuple filters for mapping control plane packets to blocks
- Receive-side scaling (RSS)

Meeting Operational Requirements

Gatekeeper provides support for features that are required in real-world, operational environments

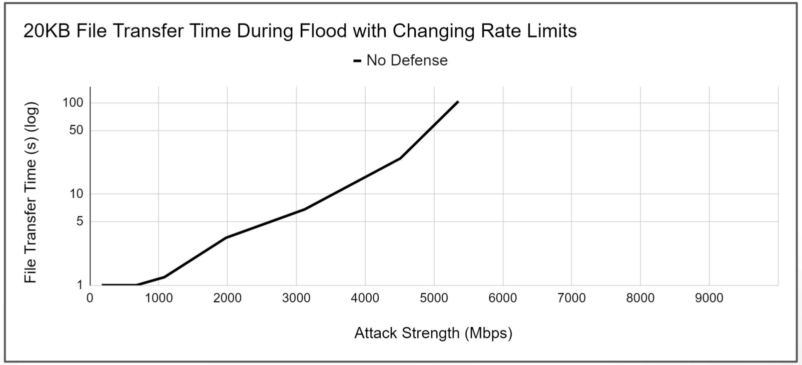
- VLAN tagging
- Rate-limiting logging
- Support for existing control plane tools (e.g. BIRD)
- Runtime configuration client

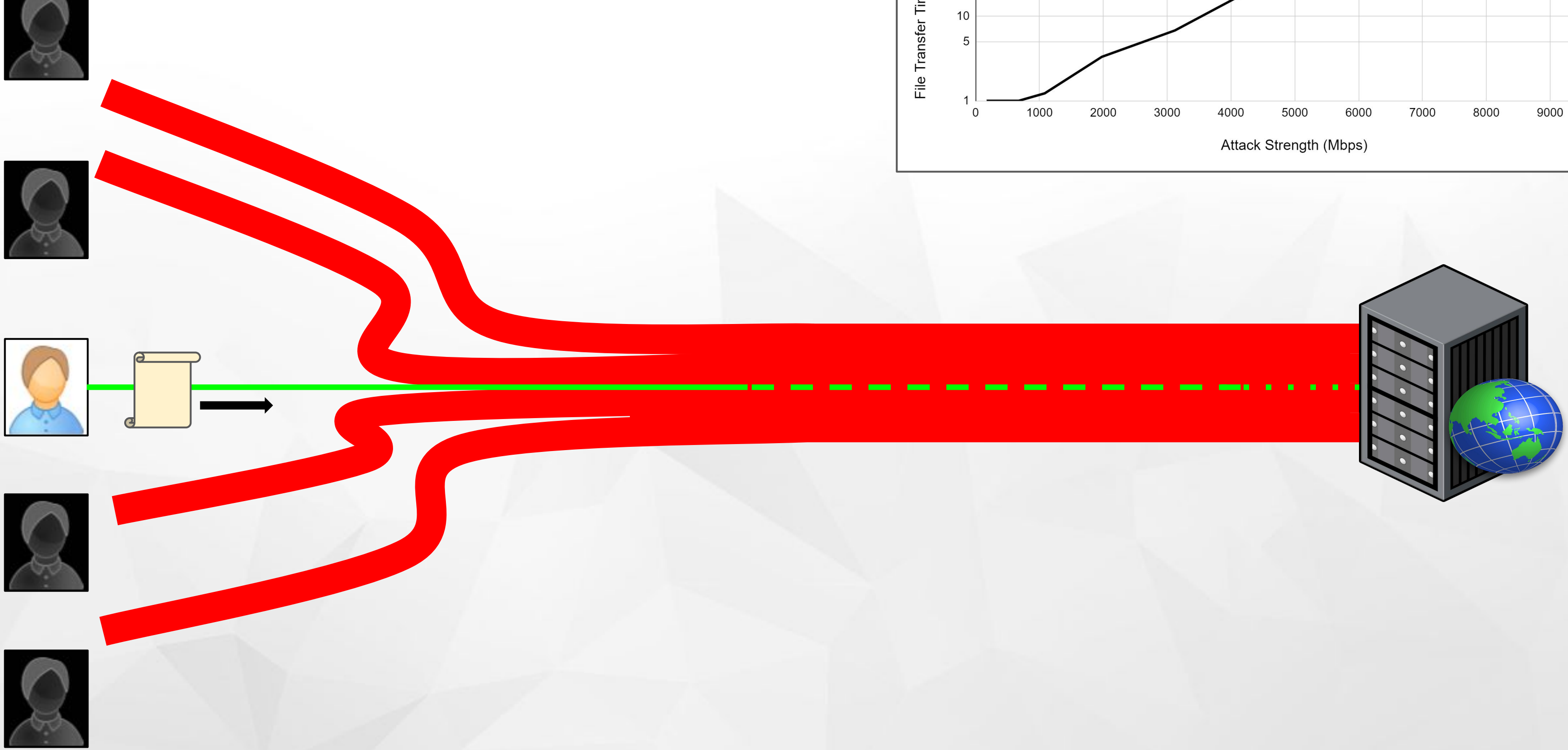
Evaluation

We evaluated Gatekeeper along several axes:

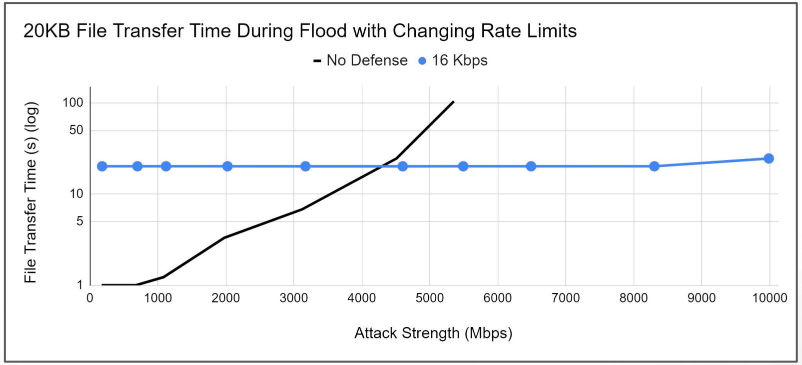
- Basic functionality
  ⇒ Can Gatekeeper mitigate attacks?
- The effect of different policies
  ⇒ How do various policies affect Gatekeeper's ability to mitigate attacks?
- Stress testing
  ⇒ How does Gatekeeper perform under worst-case conditions?
- Cost
  ⇒ How much does Gatekeeper cost, and what do you get for it?

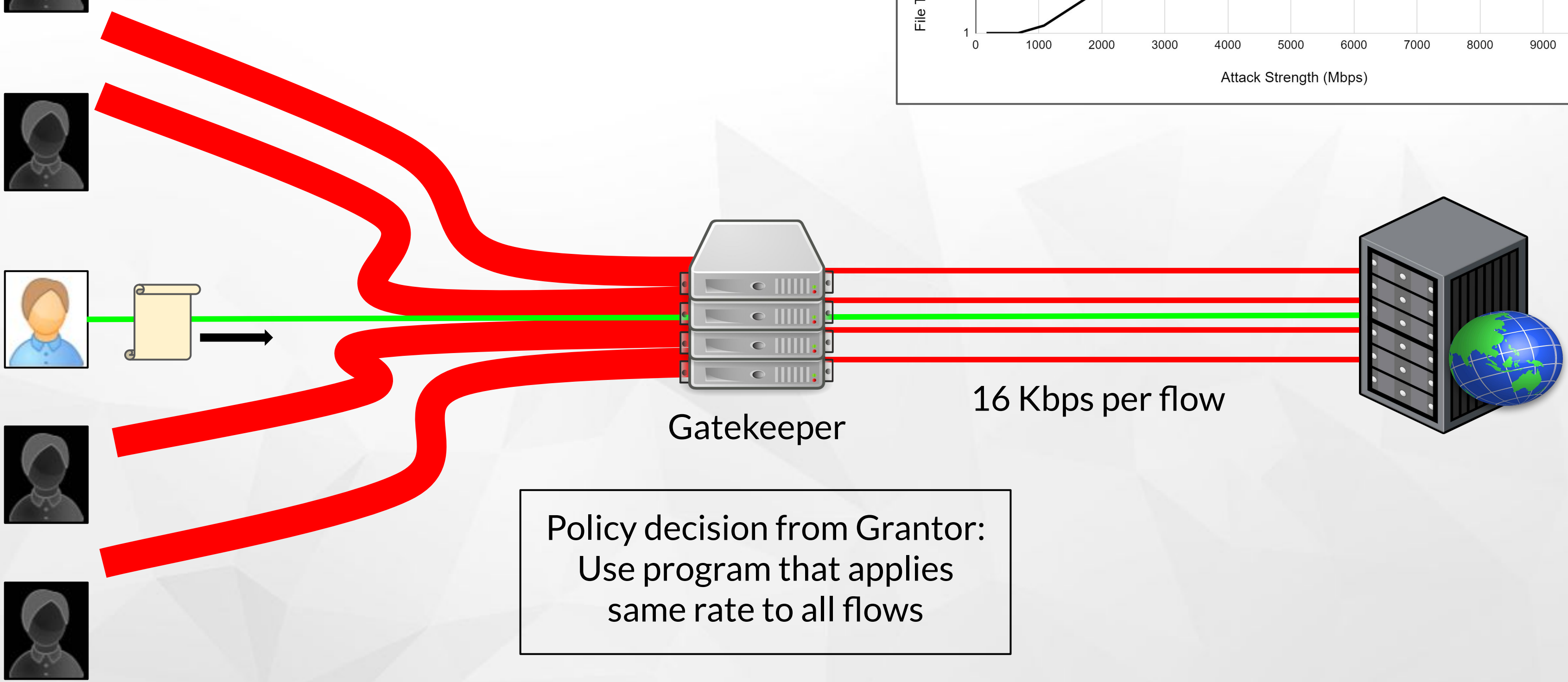
Evaluation Setup





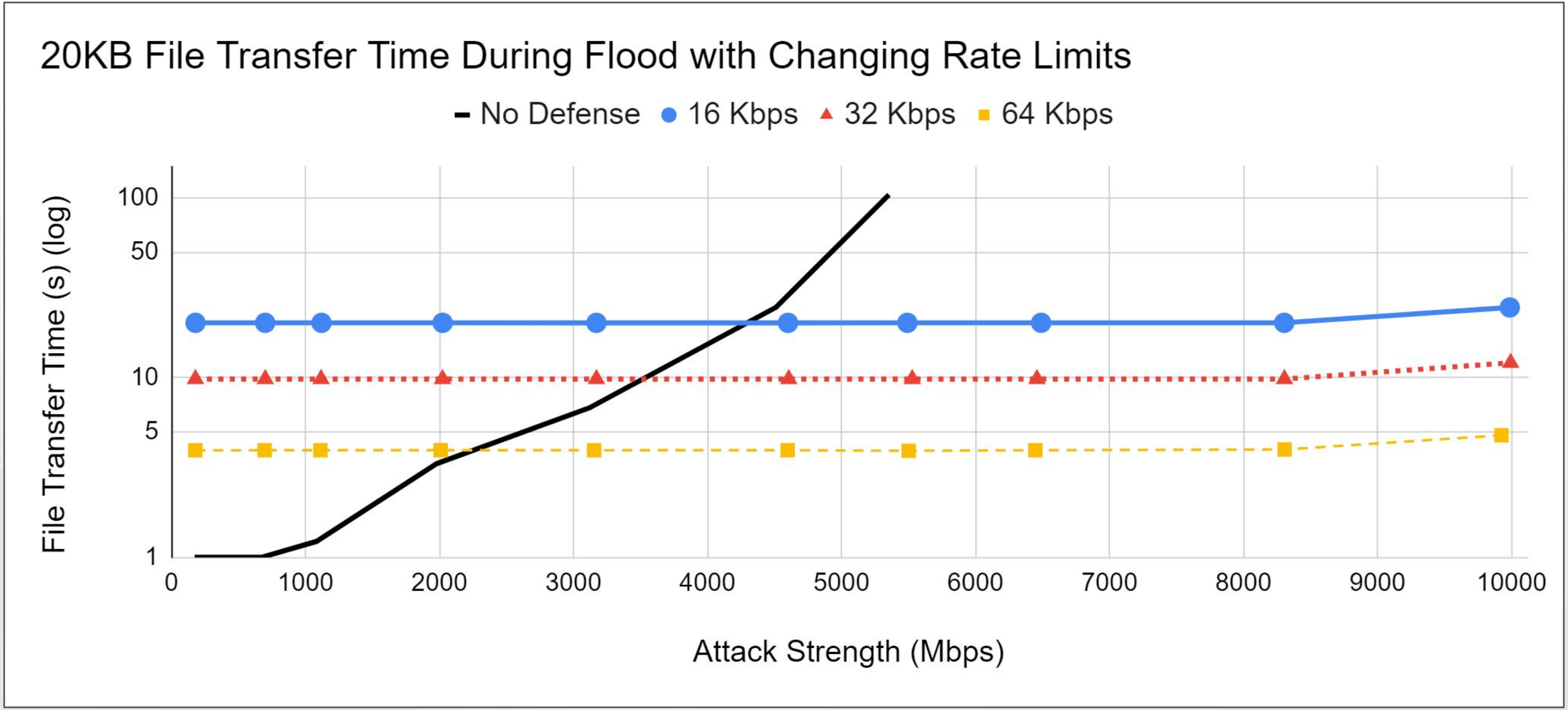
Basic Policies



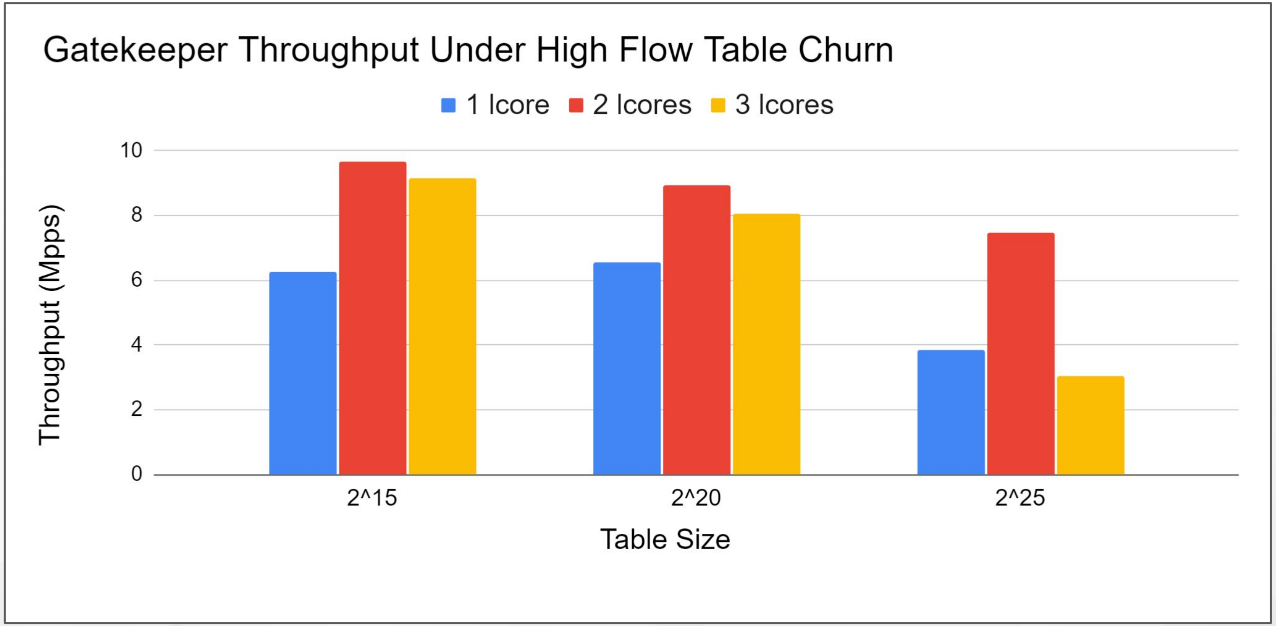


Policy decision from Grantor:
Use program that applies
same rate to all flows

Basic Policy Enforcement



Gatekeeper Packet Throughput w/High Churn



Experimental setup:

- Random source addresses → every packet represents a new flow, flow table is constantly full
- Minimum packet size (64B)
- Run on bare-metal hardware
- Packet generator on same hardware as Gatekeeper

Gatekeeper Cost

- Back-of-the-envelope evaluation using best available estimates from industry partners and quotes from public materials

- Cost of defending against a **2.3 Tbps** attack

Gatekeeper

  - 23 VPs each with a capacity of 100 Gbps
  - Monthly cost per VP: $5k (conservative)
  - Total: $1,380k per year

KrebsonSecurity
In-depth security news and investigation

  - Suffered a 620 Gbps Mirai attack in 2016
  - Was so damaging that Akamai revoked their pro-bono protection
  - "If this kind of thing is sustained, we're definitely talking millions"

- 99% of DDoS attacks are < 20 Gbps
  - Gatekeeper estimate: $12k per year
  - Confidential estimate for service offered to industry partner: $24k

Agenda

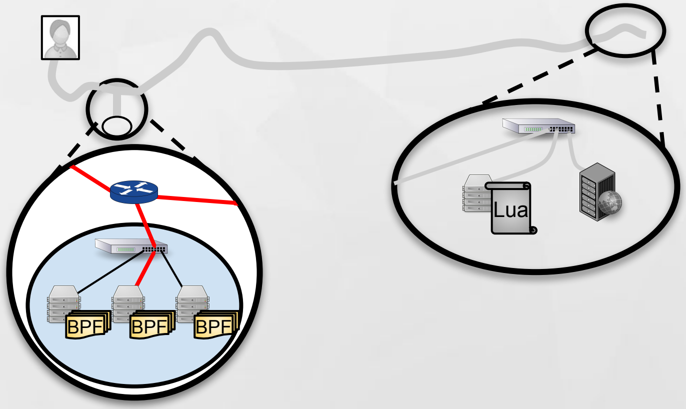
- Background
  - Next-generation attacks
  - Architectural issues and deployability
- Thesis
- Gatekeeper Overview
  - Design
  - Implementation
  - Evaluation
- Gatekeeper Policy Toolkit
- Mitigating Next-Generation Attacks

Policy Toolkit

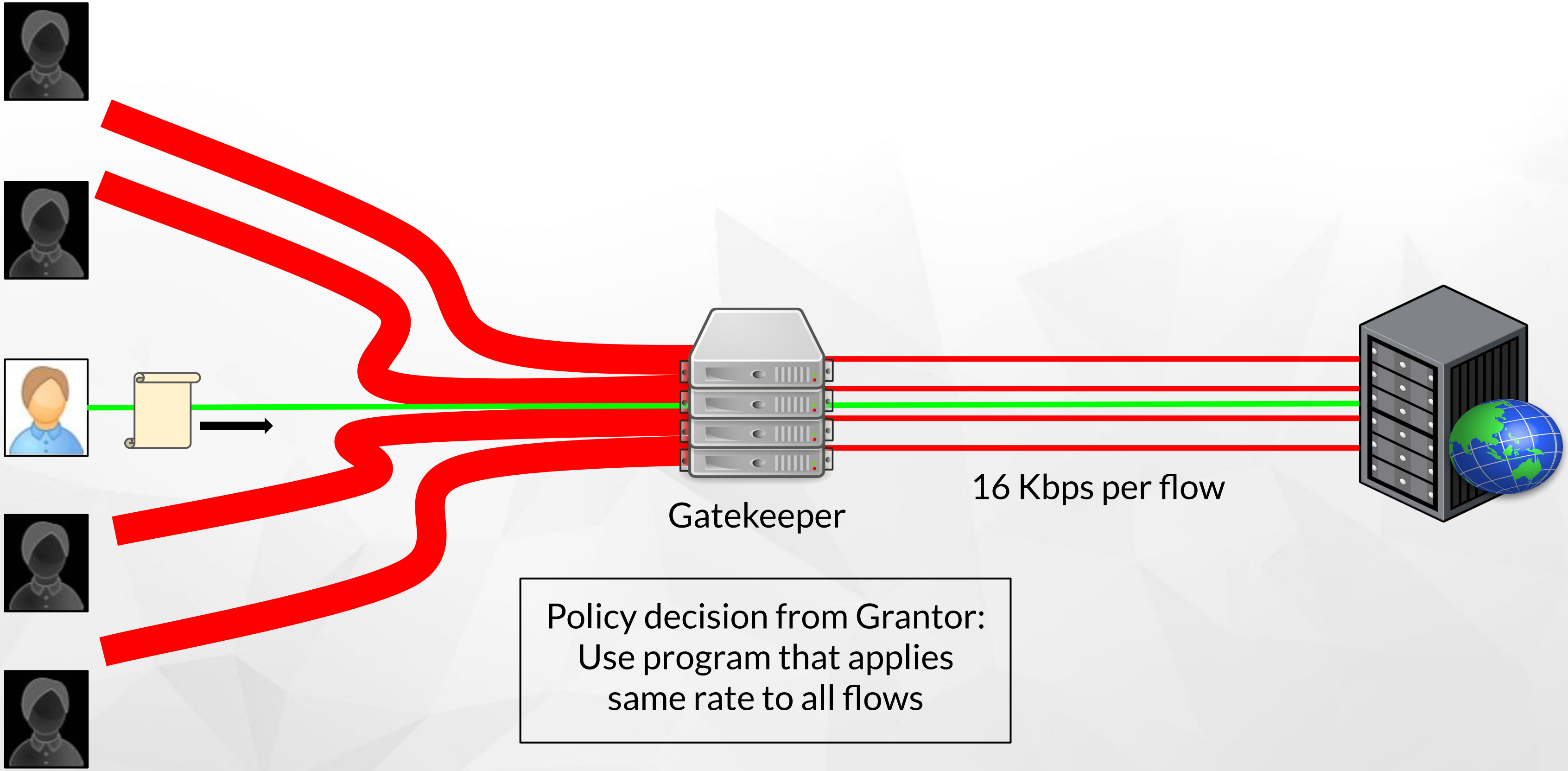
Gatekeeper only works as well as the destination policies that govern it

There are two sides to the policy:

- Policy *decision* programs at Grantor (Lua)
  ⇒ Map flows (source IP, destination IP) pairs to policy decisions
  ⇒ Only sees the first packet of a flow
- Policy *enforcement* programs at Gatekeeper (BPF)
  ⇒ In the simplest case, just drops or rate limits
  ⇒ But can also inspect headers of *every* packet
  ⇒ Each flow is given 64B of program state

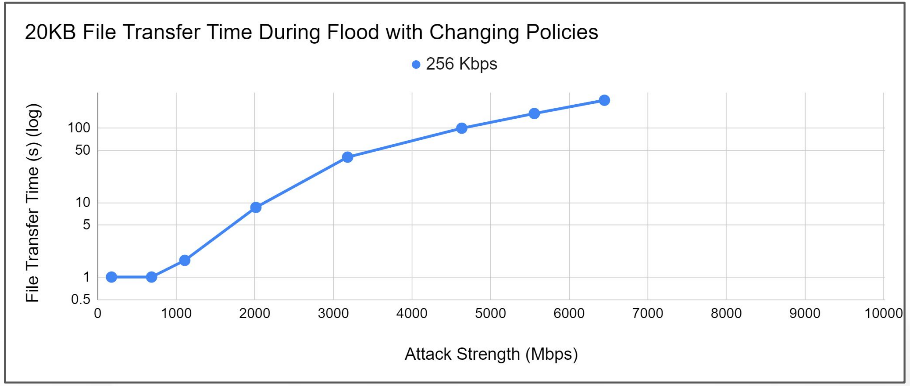


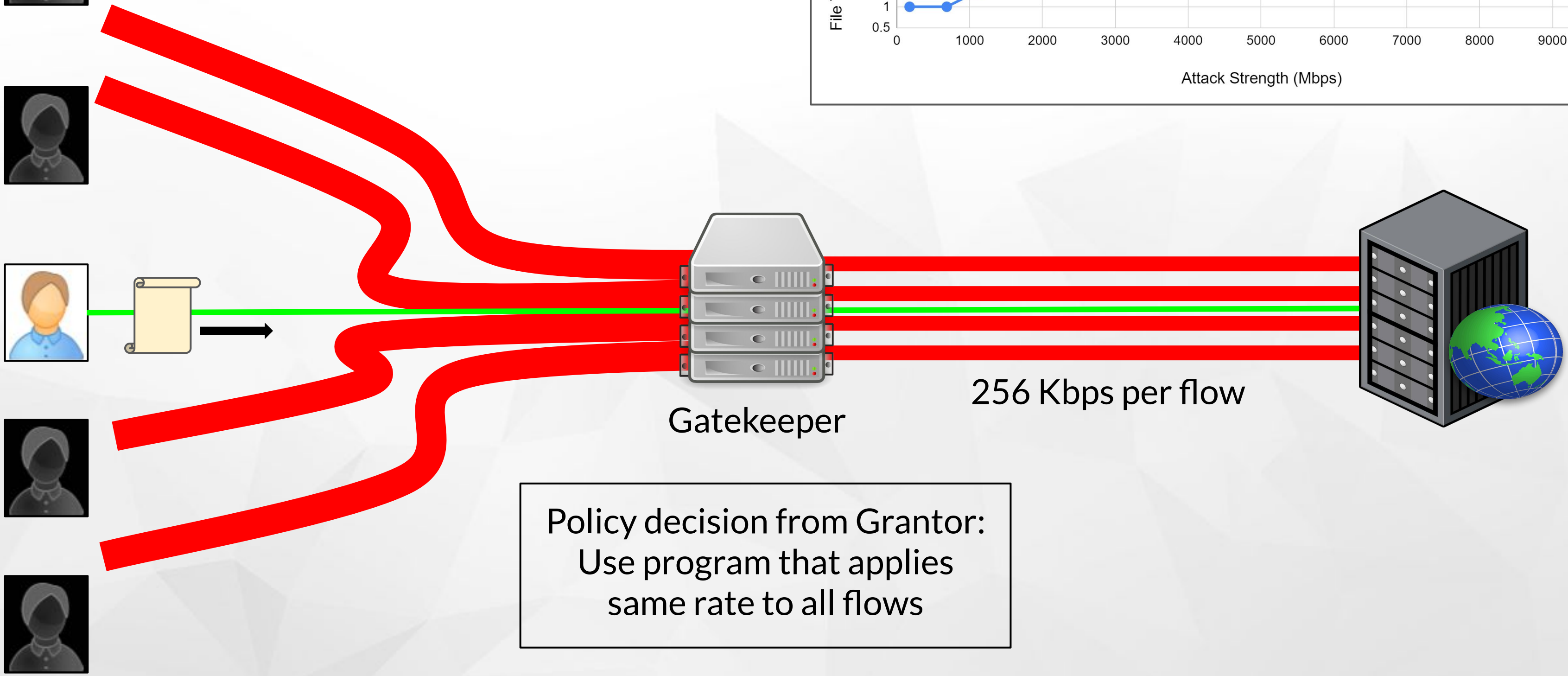
Basic Policies



Policy decision from Grantor:
Use program that applies
same rate to all flows

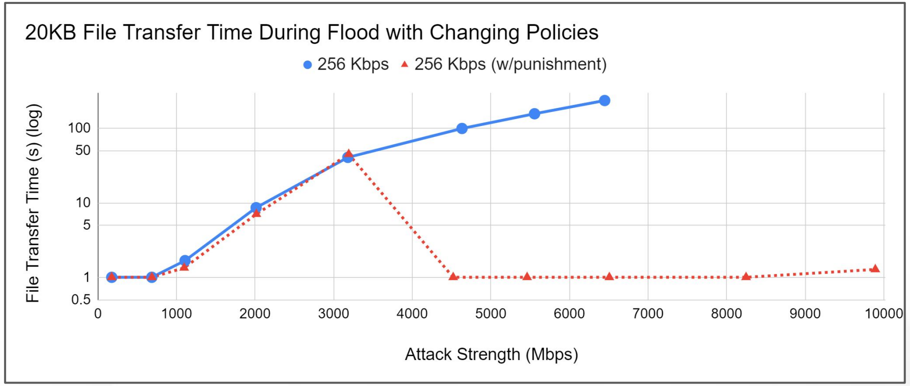
Basic Policies

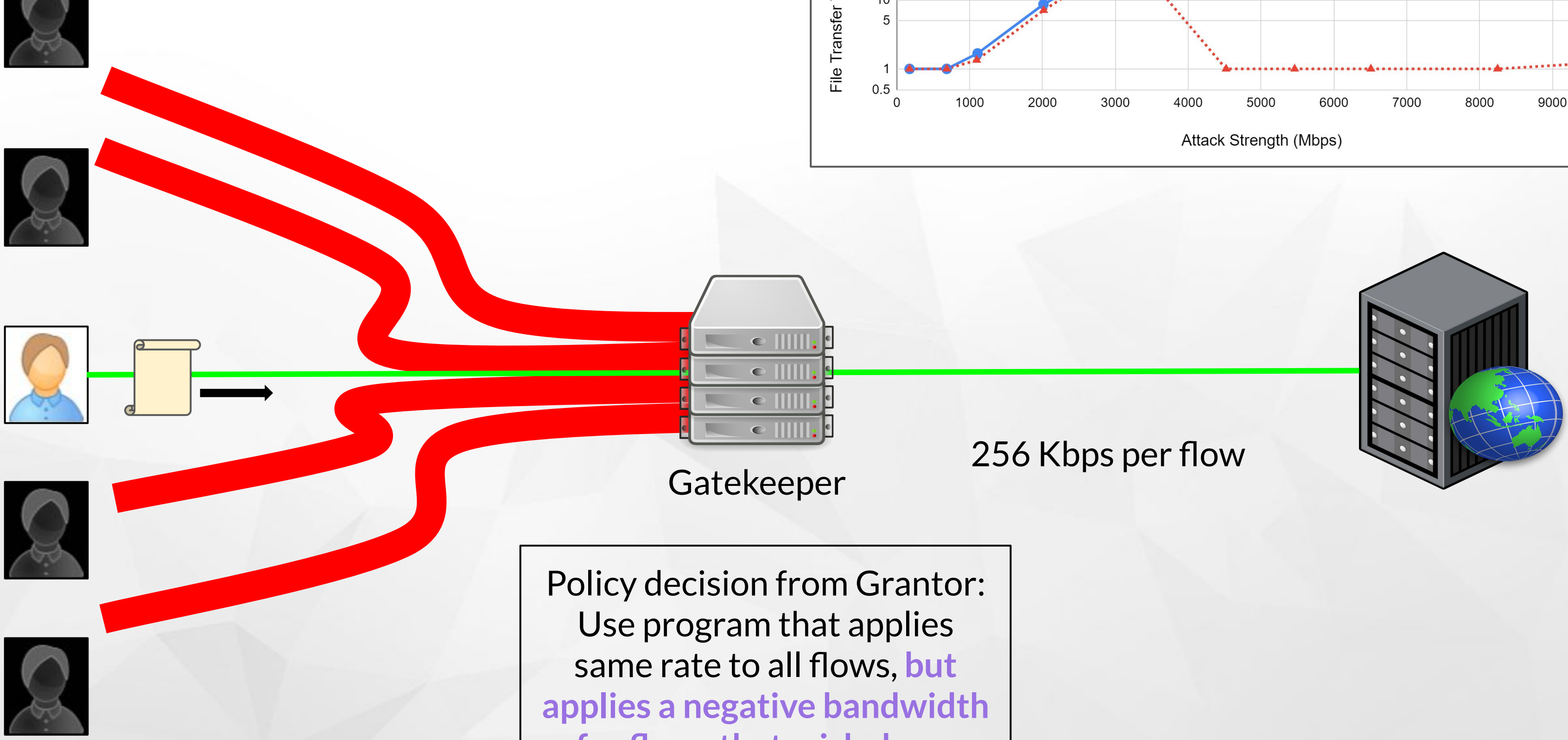




Policy decision from Grantor:
Use program that applies
same rate to all flows

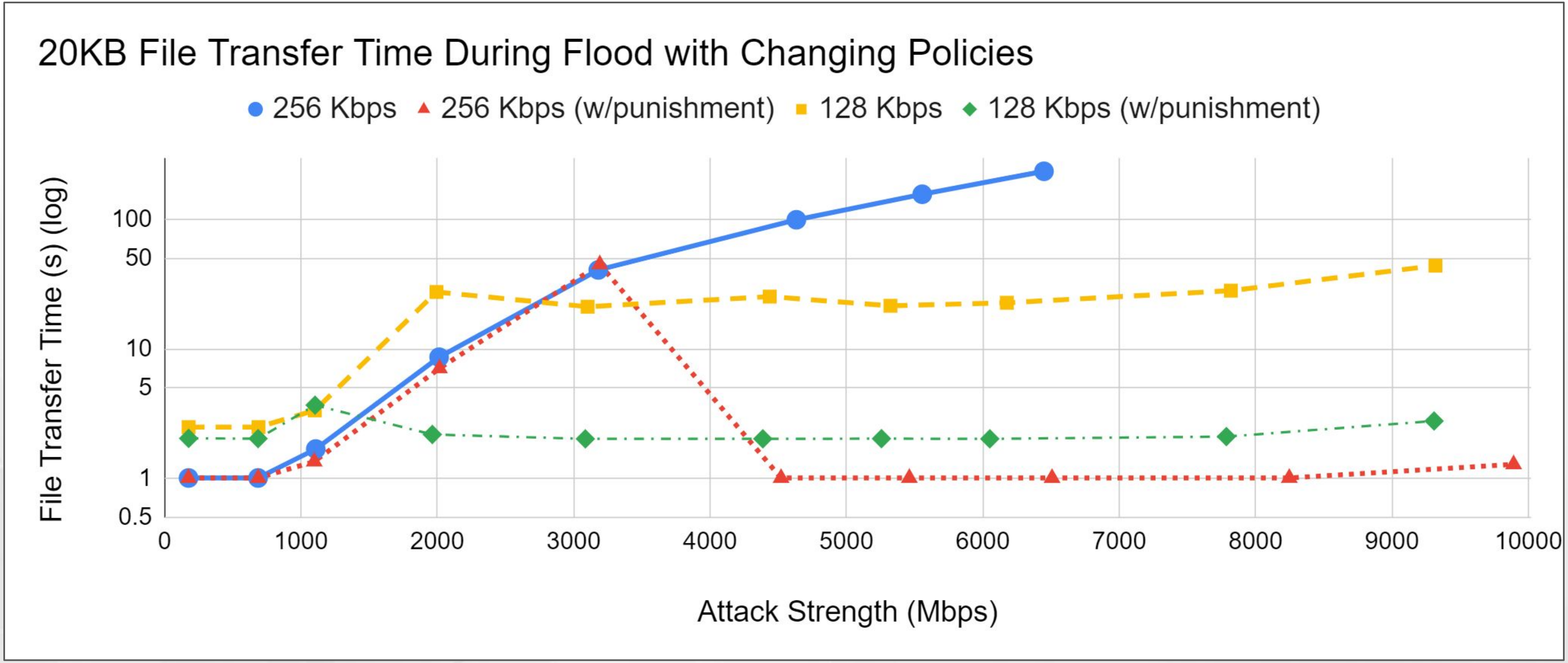
Negative Bandwidth



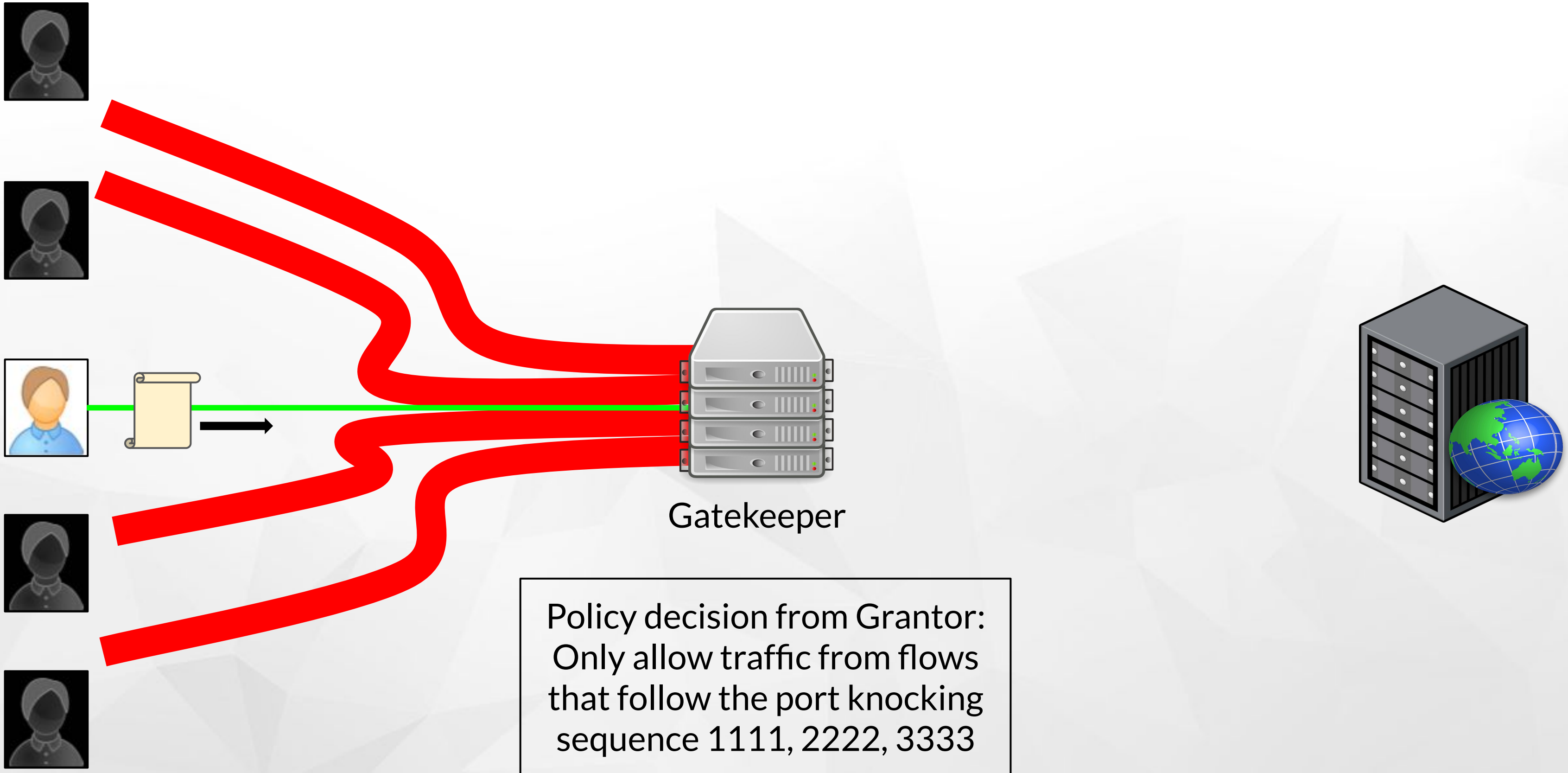


Policy decision from Grantor: Use program that applies same rate to all flows, but applies a negative bandwidth for flows that misbehave

Effect of Negative Bandwidth

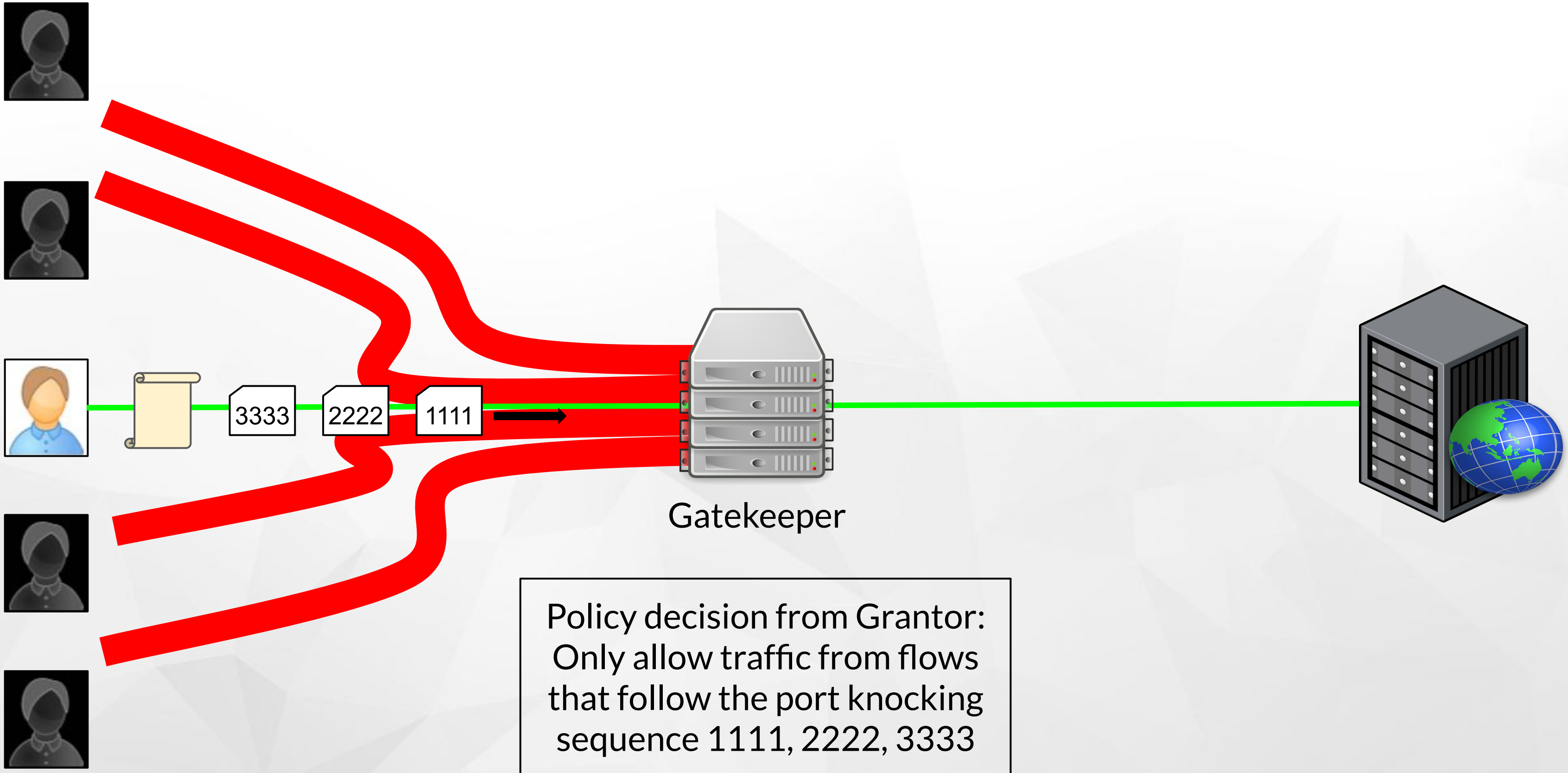


Port Knocking



Policy decision from Grantor:
Only allow traffic from flows
that follow the port knocking
sequence 1111, 2222, 3333

Port Knocking



Policy decision from Grantor:
Only allow traffic from flows
that follow the port knocking
sequence 1111, 2222, 3333

Richness of Policy Enforcement Programs

With per-flow programs and state, you can do things like:

- Deny admission for certain types of packets
  ⇒ Unused ports, amplification attacks, traceroute
- Multiple bandwidth limits
  ⇒ Rate limit TCP SYNs, UDP, ICMP, etc. at a lower rate than normal traffic
- Negative bandwidth
  ⇒ Punish flows that abuse their capability by dropping packets while negative
- Port knocking
  ⇒ Lightweight authentication by probing using a certain sequence of ports

Agenda

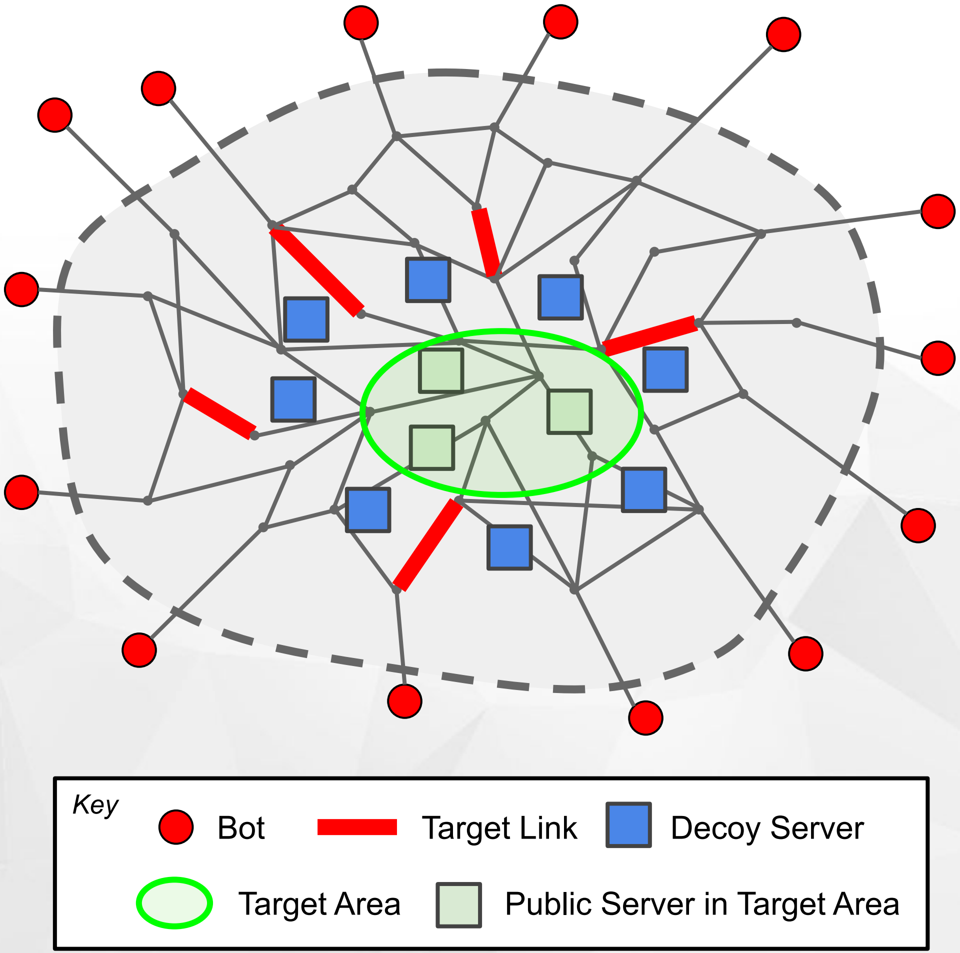
- Background
  - Next-generation attacks
  - Architectural issues and deployability
- Thesis
- Gatekeeper Overview
  - Design
  - Implementation
  - Evaluation
- Gatekeeper Policy Toolkit
- Mitigating Next-Generation Attacks

Next-Generation Attacks

There are three major shifts occurring in the Internet ecosystem: IoT, 5G, IPv6

⇒ Attackers will be more powerful than ever, just as the Internet architecture and infrastructure undergo a major transition

⇒ These trends favor large-scale link attacks like Crossfire



Crossfire Attack Setup

1. Send `traceroute` probes from botnet to decoy servers and public servers to build map of persistent links

2. Pick *target links* -- those that carry densest share of flows

3. Rotate attack between disjoint sets of target links to maintain attack persistence

What Can We Do?

All previous solutions in this space either:

- Are point solutions that make simplifying assumptions
- Require a complete restructuring of the Internet

But Gatekeeper neutralizes the architectural advantages that Crossfire enjoys

- Dilutes the link map construction
- Provides path diversity that circumvents target links
- Enables a moving target defense

Measurement Study

We conducted a measurement study to actually build a Crossfire link map

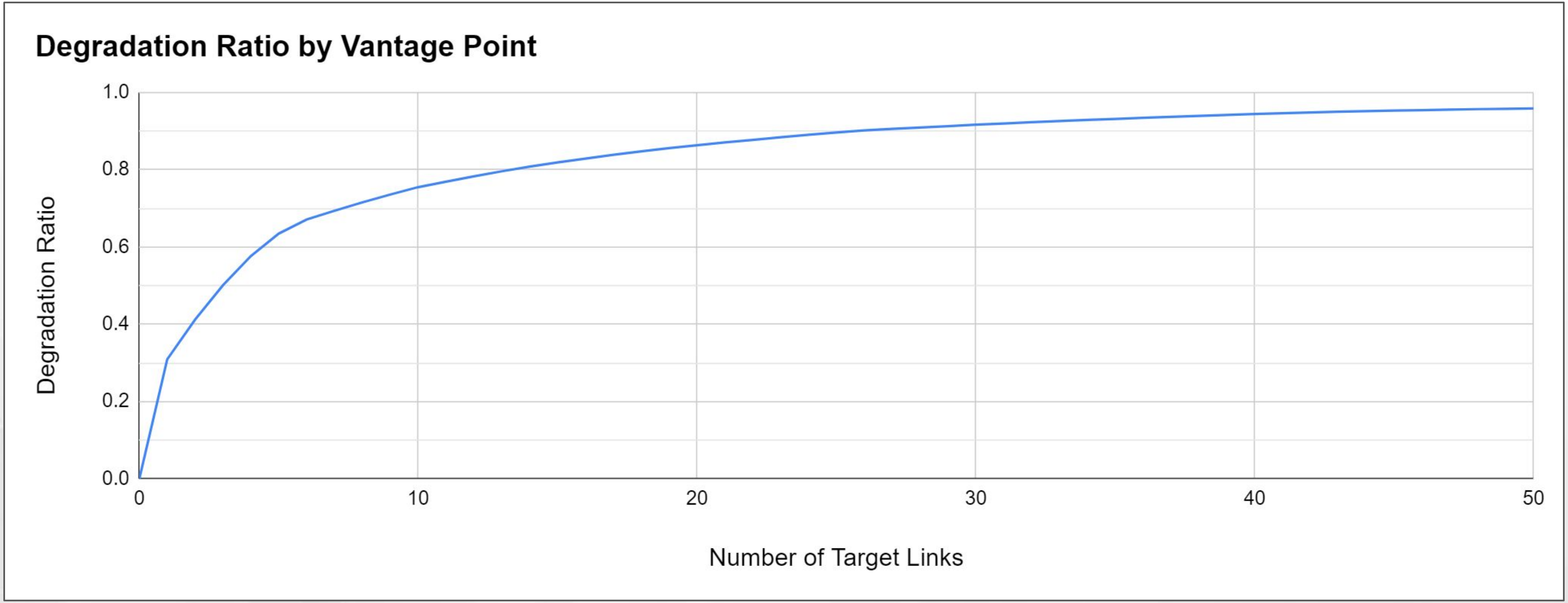
⇒ Bots: `traceroute` servers distributed throughout the Internet

⇒ Target Area: universities in the Boston area

Key metric of success of Crossfire attack: *degradation ratio*

⇒ The fraction of paths to the target area that cross a target link

Degradation Ratio



Measurement Study

We conducted a measurement study to actually build a Crossfire link map

⇒ Bots: `traceroute` servers distributed throughout the Internet

⇒ Target Area: universities in the Boston area

Key metric of success of Crossfire attack: *degradation ratio*

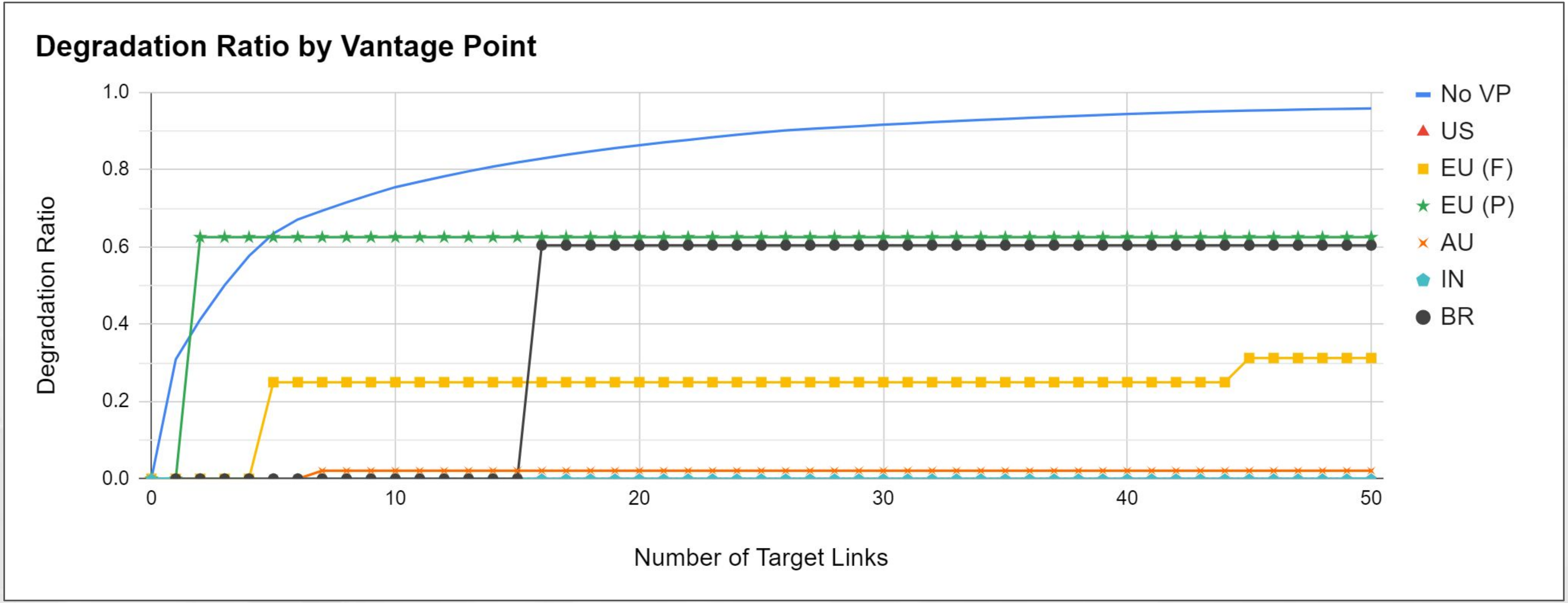
⇒ The fraction of paths to the target area that cross a target link

But in Gatekeeper, all traffic is forwarded through a set of VPs

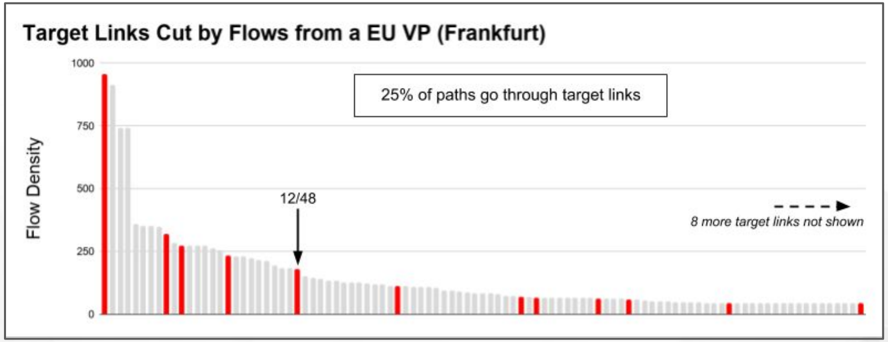
⇒ Do the paths from VPs to the target area cross target links?

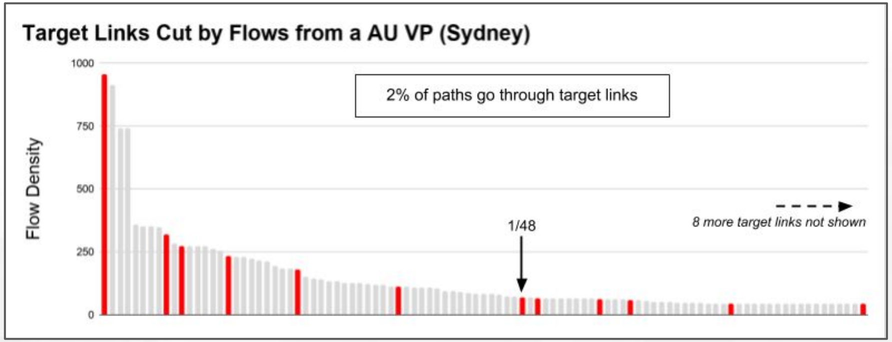
⇒ Use six Amazon cloud nodes in different world regions to see

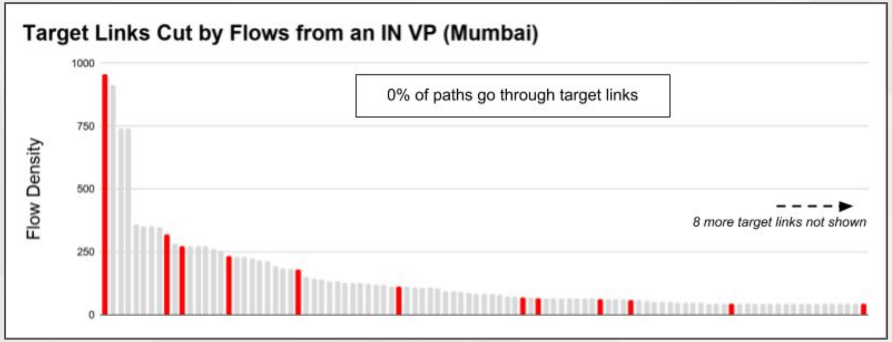
Degradation Ratio

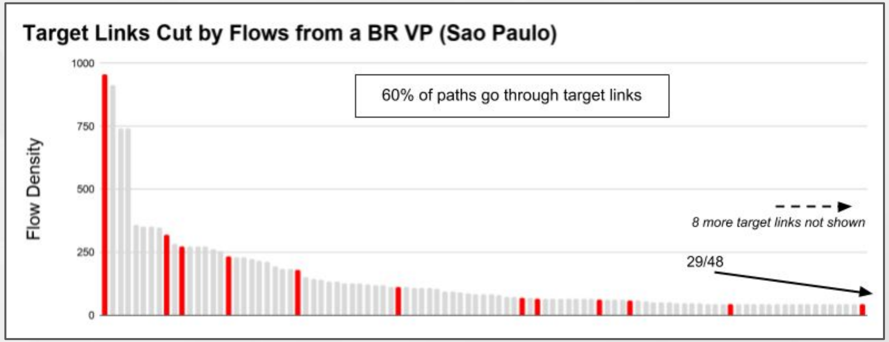


Cloud Paths Crossing Target Links

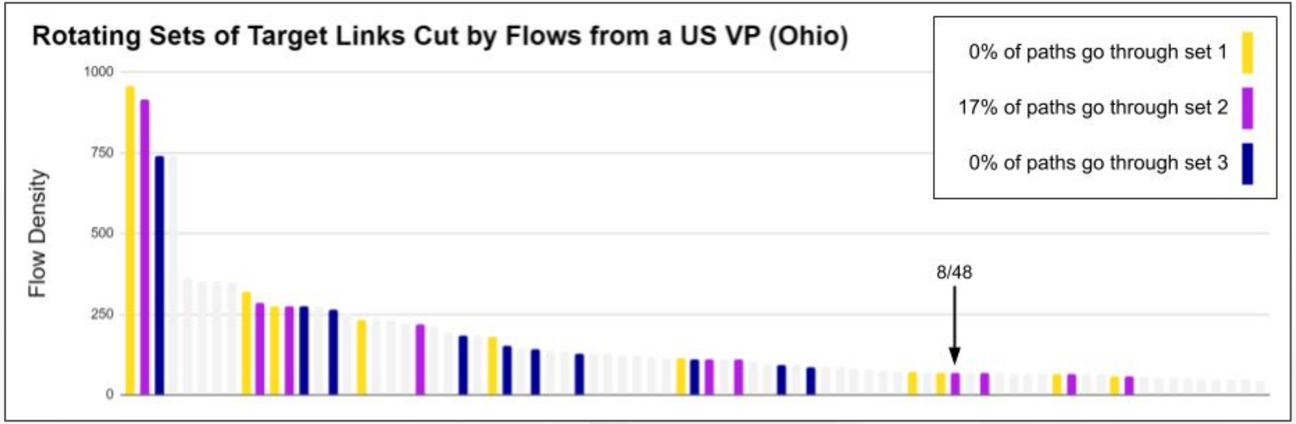


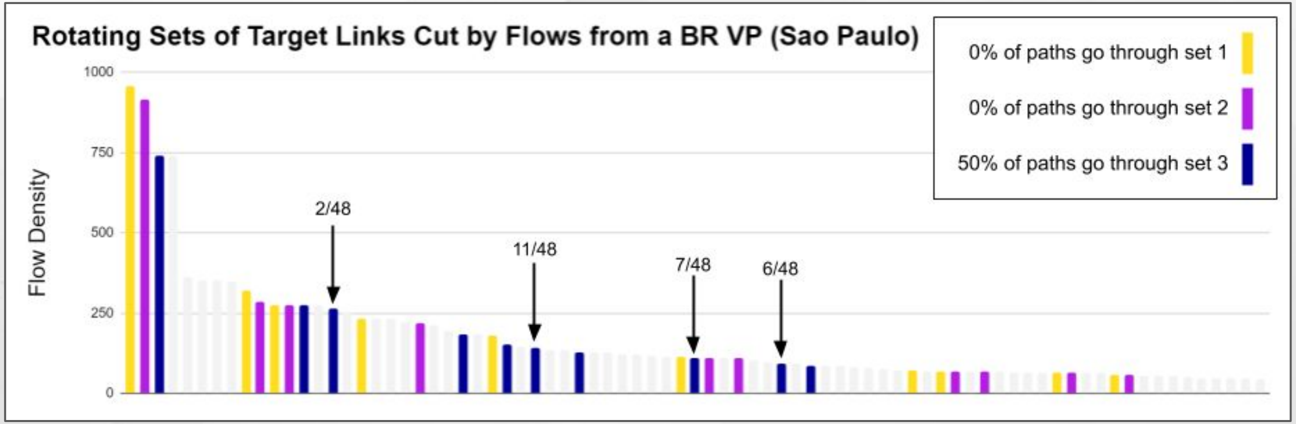






Cloud Paths Crossing Rotating Target Links





Key Takeaway

When Gatekeeper is deployed in cloud environments, it can leverage the independence of cloud paths to circumvent Crossfire target links

Summary

- ⇒ Deployable realization of a network capability system using IXPs and clouds
  - Putting a connection-oriented network layer into practice at last
- ⇒ Enforcement of expressive policies using programs instead of declarative rules
  - Enabling a rich set of algorithms and actions to choose and apply per-flow
- ⇒ Provides opportunities to mitigate next-generation attacks
  - Leverages architectural and topological advantages over link attacks

Tale of Two Deployments

Gatekeeper has achieved the escape velocity needed to go from academia to the real world

DIGIRATI

- Fairly small ISP in Brazil
- Looking for *affordable* yet comprehensive DDoS solution
- Deploying Gatekeeper for 10 Gbps protection

@mail

- Russian social media and ISP giant
- Looking for *scalable* and comprehensive DDoS solution
- Deploying Gatekeeper for 1 Tbps protection

Gatekeeper's value: comprehensive and affordable, yet scalable → suitable for a range of needs and providers

Thank you!

Questions?