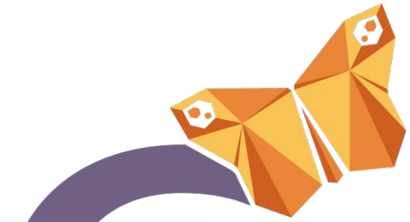


Gatekeeper

First Open Source DDoS Protection System

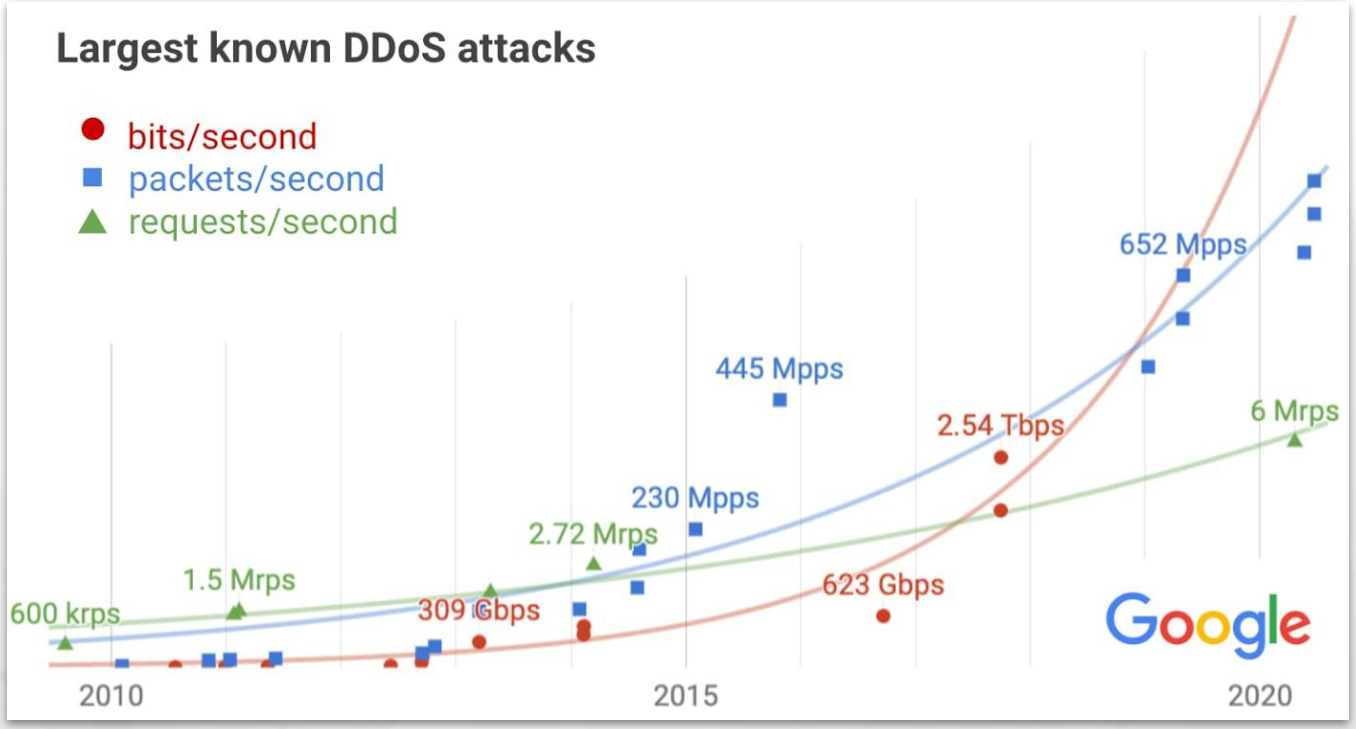
Michel Machado Cody Doucette Qiaobin Fu John W. Byers

hostnet Raytheon BBN Google BOSTON UNIVERSITY

NANOG 82 -- June 14th, 2021

Gatekeeper

Motivation -- Relevance of DDoS attacks



Gatekeeper

Motivation -- Largest DDoS attacks of 2020

Who	Peak	When
AWS	2.3 Tbps	February
Akamai	809 Mpps	June
Cloudflare	754 Mpps	June

809 Mpps is the newest packet-rate record

2.3 Tbps is close to the bandwidth record: 2.54 Tbps in Sep 2017

Gatekeeper

Motivation -- Why Gatekeeper?

Unparalleled multi-vector protection

⇒ All flows are monitored and all filters are active;
alternative solutions have limited filtering capacity;
See paper "The Catch-22 Attack" for details

Scalable

⇒ 1 Tbps deployment underway at Mail.ru

Mitigation in seconds

⇒ More than 80% of attacks last ≤ 4 min according to Kaspersky;
There is not much time for human intervention

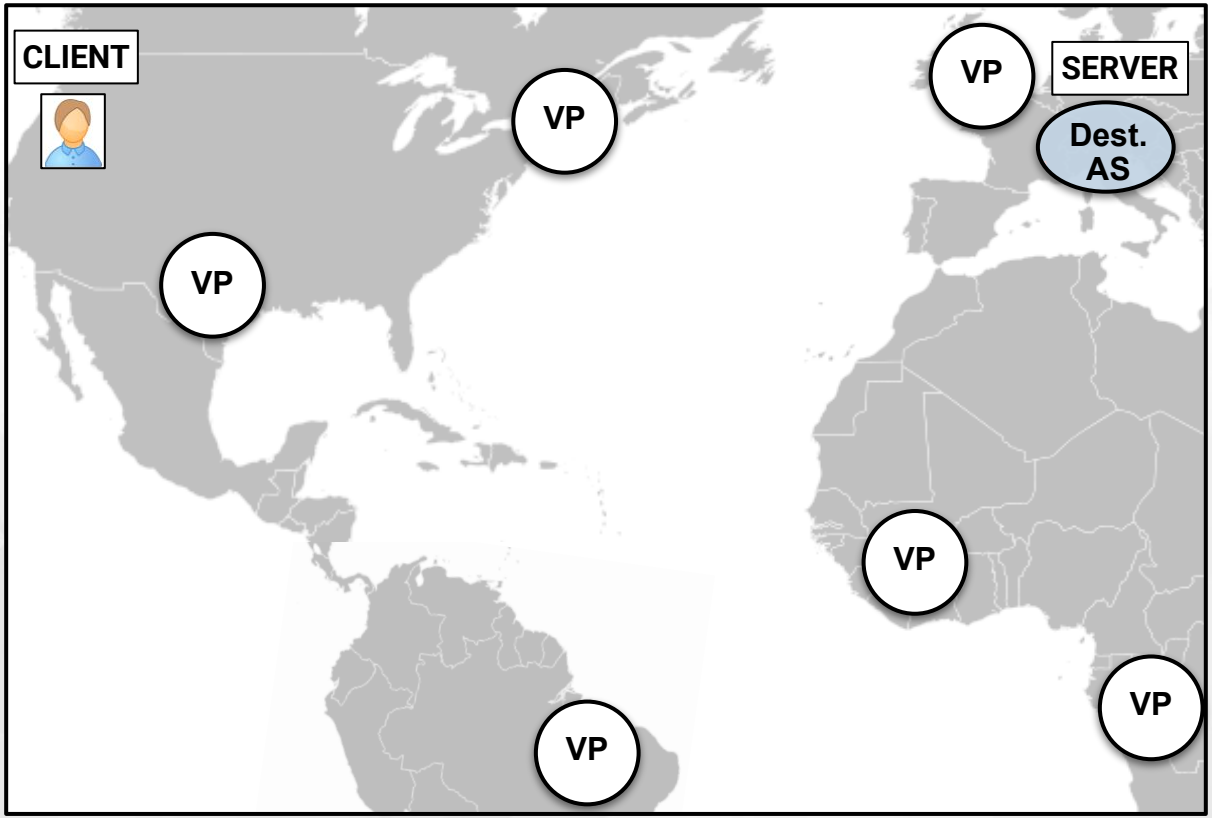
Gatekeeper

Outline

- ✓ Motivation
- How Gatekeeper works
- How to write a destination policy
- Mitigating a SYN flood
- Conclusion

Gatekeeper

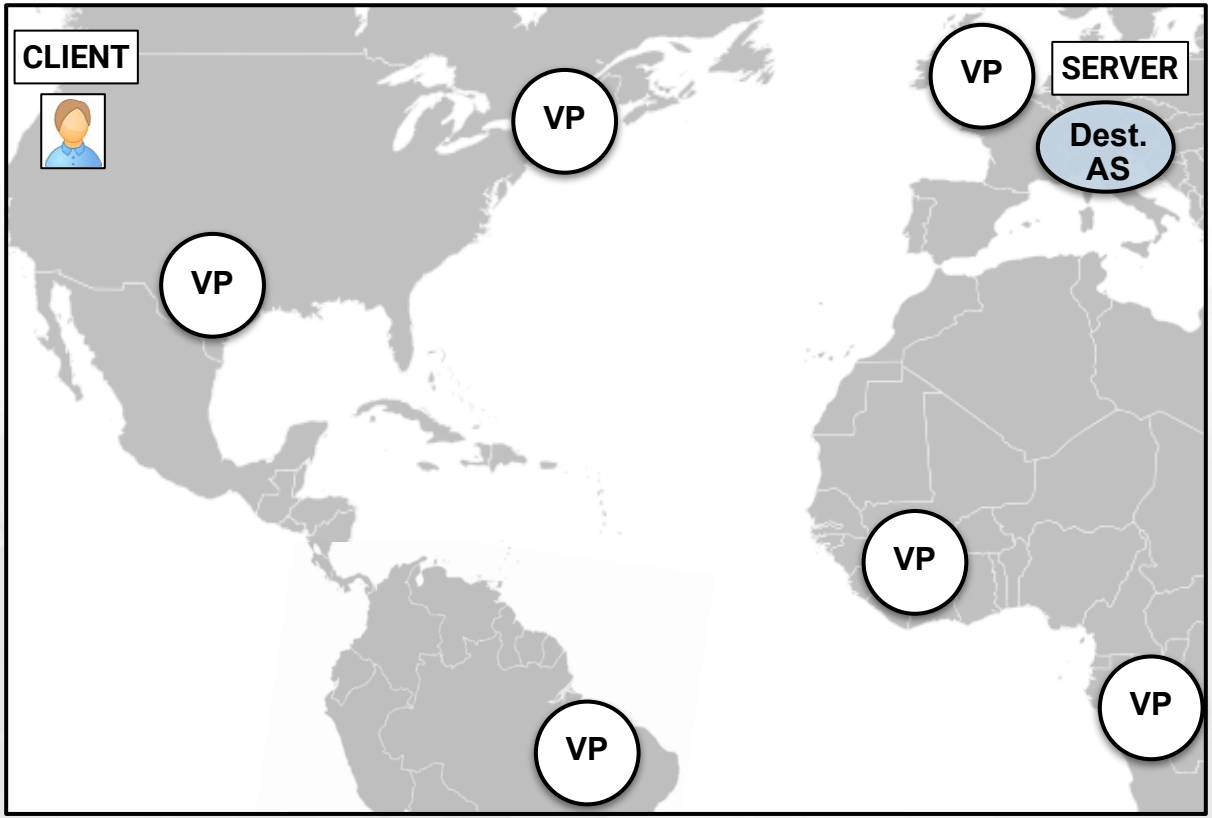
Gatekeeper's components



Vantage points: well-provisioned and geographically distributed locations

Gatekeeper

Gatekeeper's components



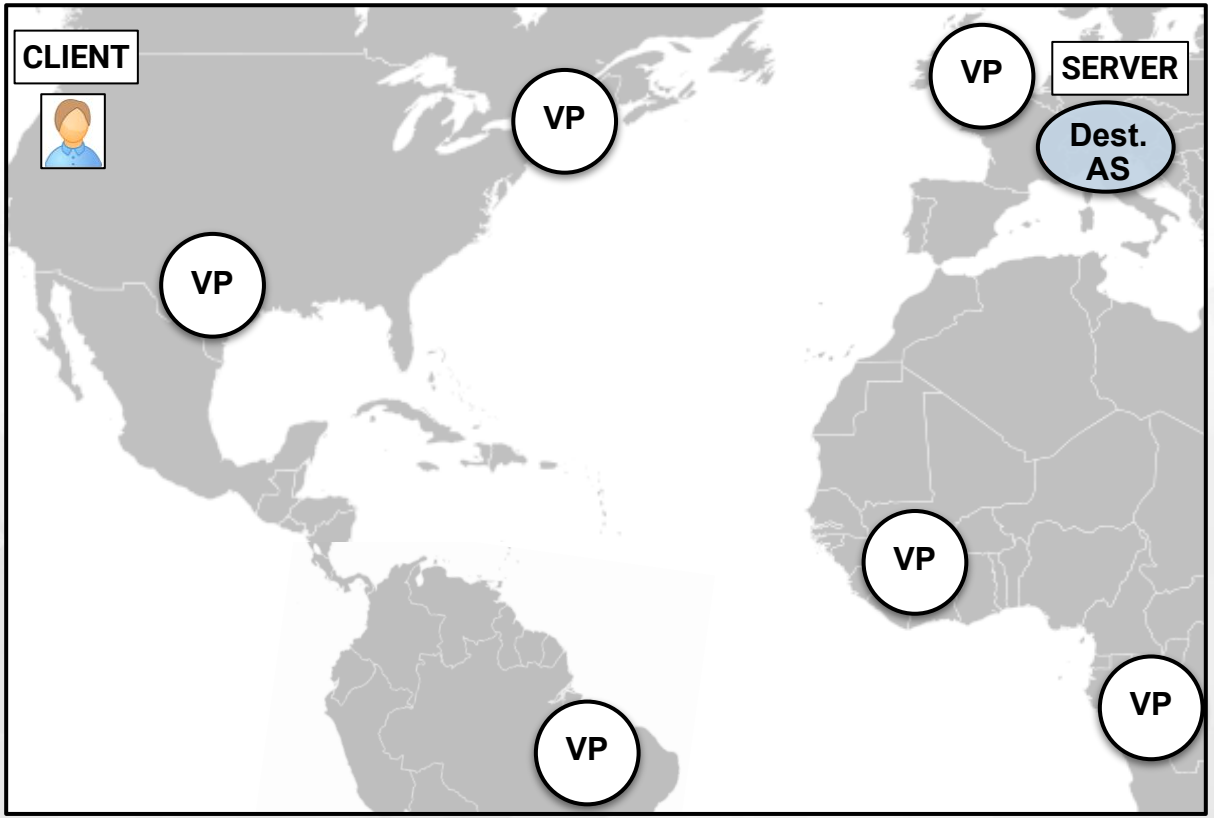
Vantage points: well-provisioned and geographically distributed locations

Requirements:

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

Gatekeeper

Gatekeeper's components



Vantage points: well-provisioned and geographically distributed locations

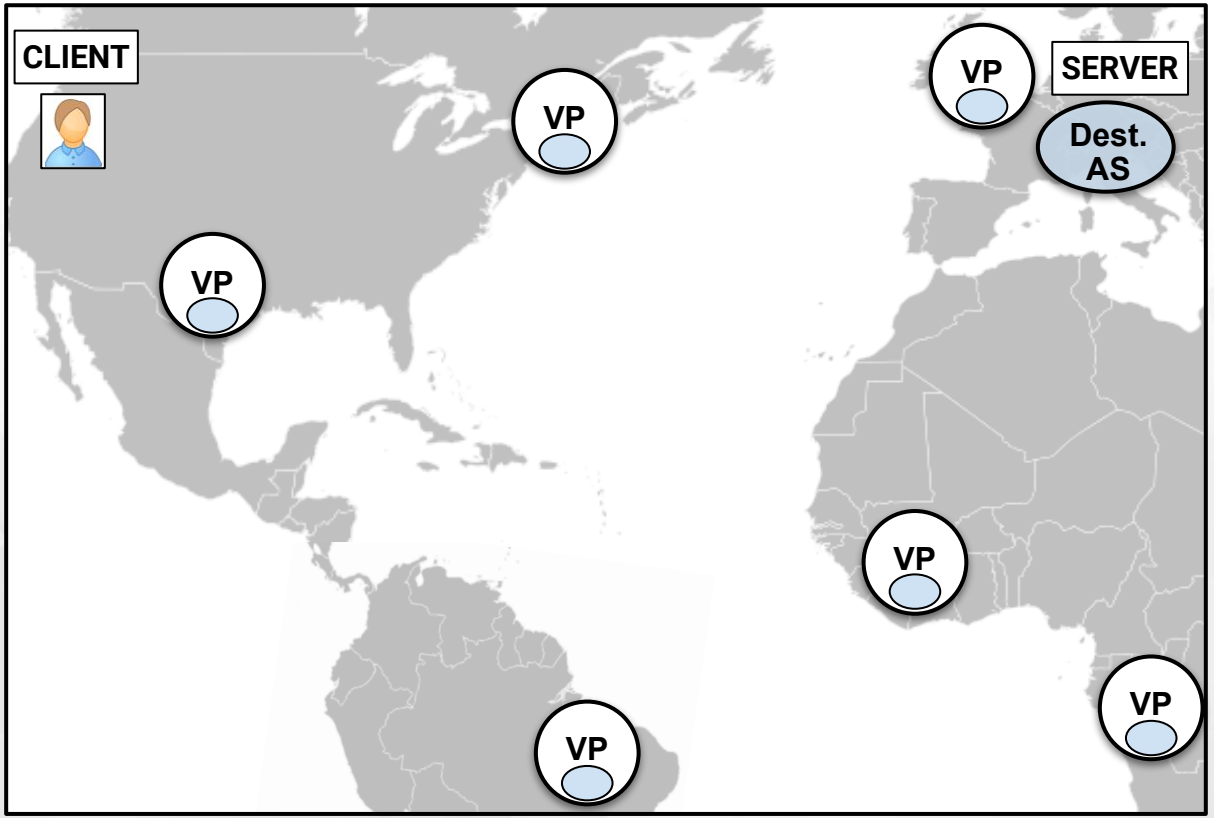
Requirements:

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

Examples:

- Internet exchanges
- Peering link
- Some cloud providers

Gatekeeper's components



Vantage points: well-provisioned and geographically distributed locations

Requirements:

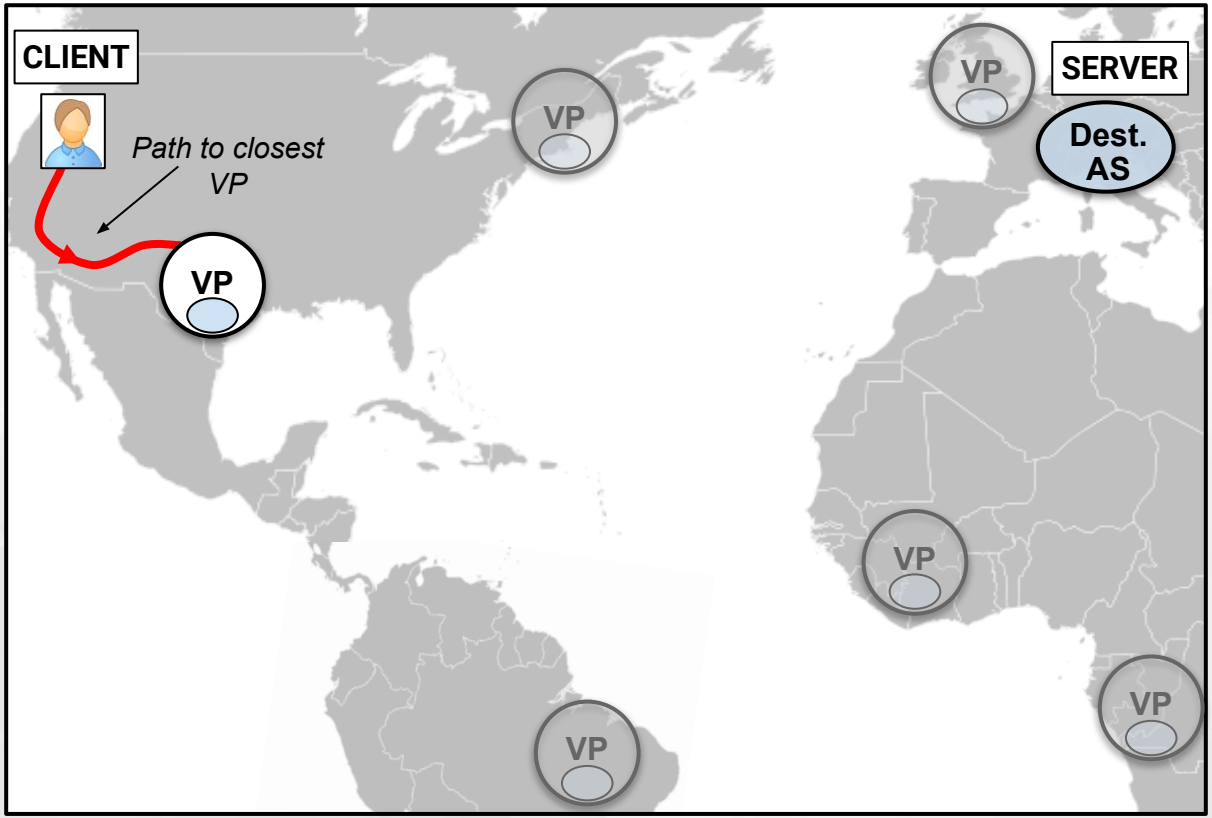
- computing capacity
- cheap ingress bandwidth
- BGP peering
- private links to the protected AS

Examples:

- Internet exchanges
- Peering link
- Some cloud providers

Gatekeeper

Gatekeeper's components



Vantage points: well-provisioned and geographically distributed locations

Requirements:

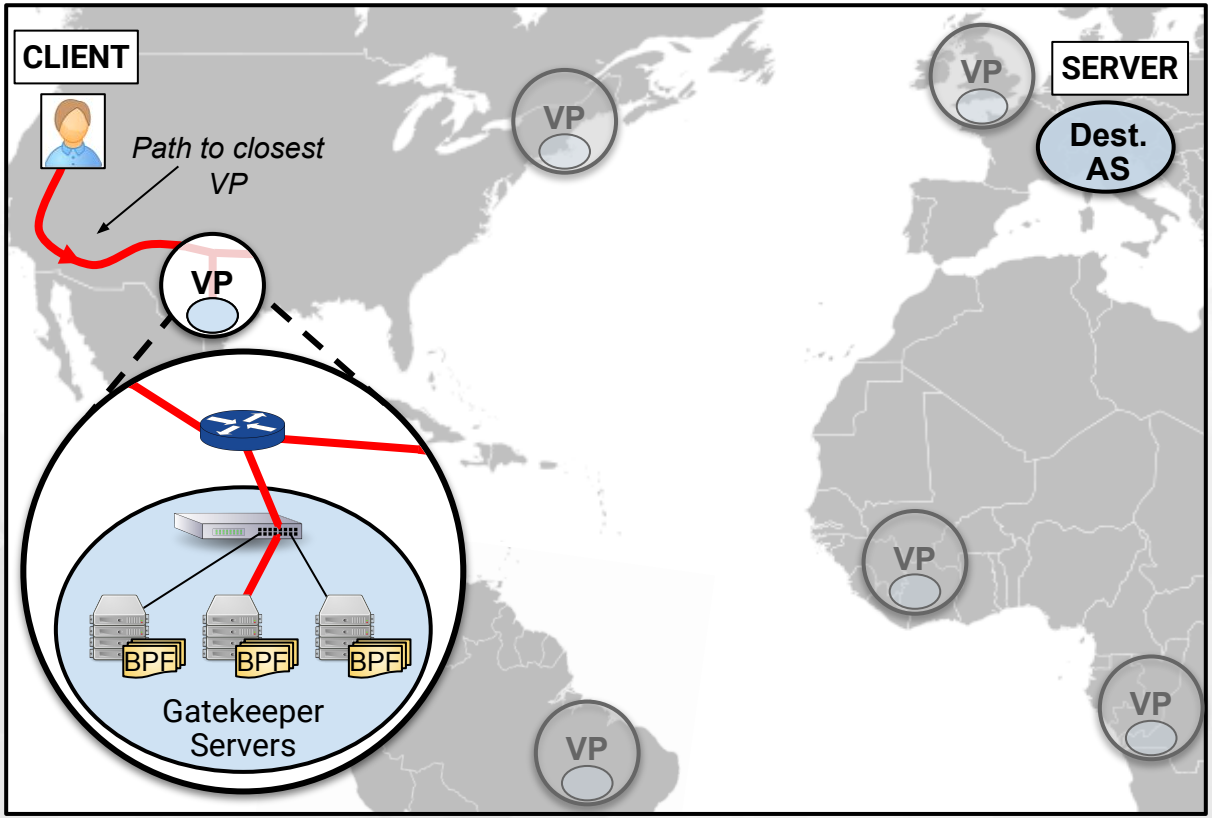
- computing capacity
- cheap ingress bandwidth
- BGP peering
- private links to the protected AS

Examples:

- Internet exchanges
- Peering link
- Some cloud providers

Gatekeeper

Gatekeeper's components



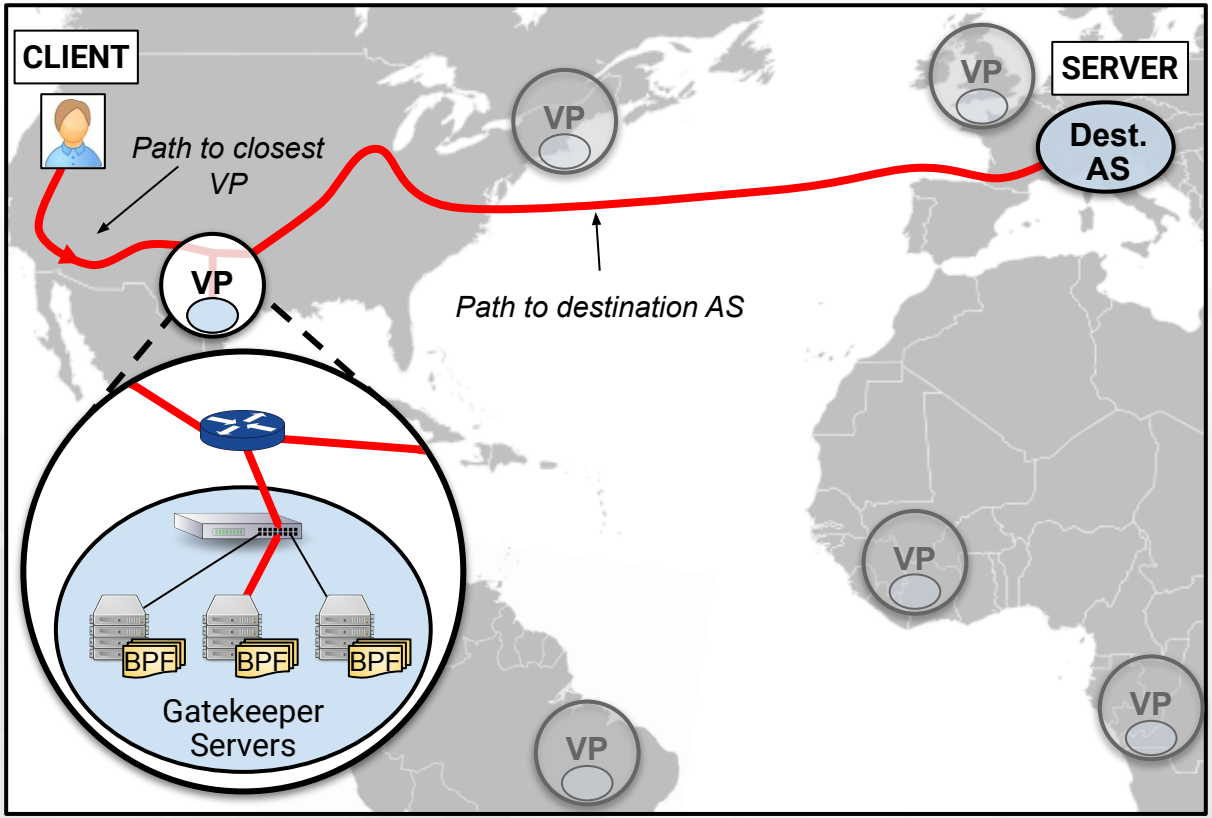
Gatekeeper servers: upstream policy enforcement

Responsibilities:

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

Gatekeeper

Gatekeeper's components

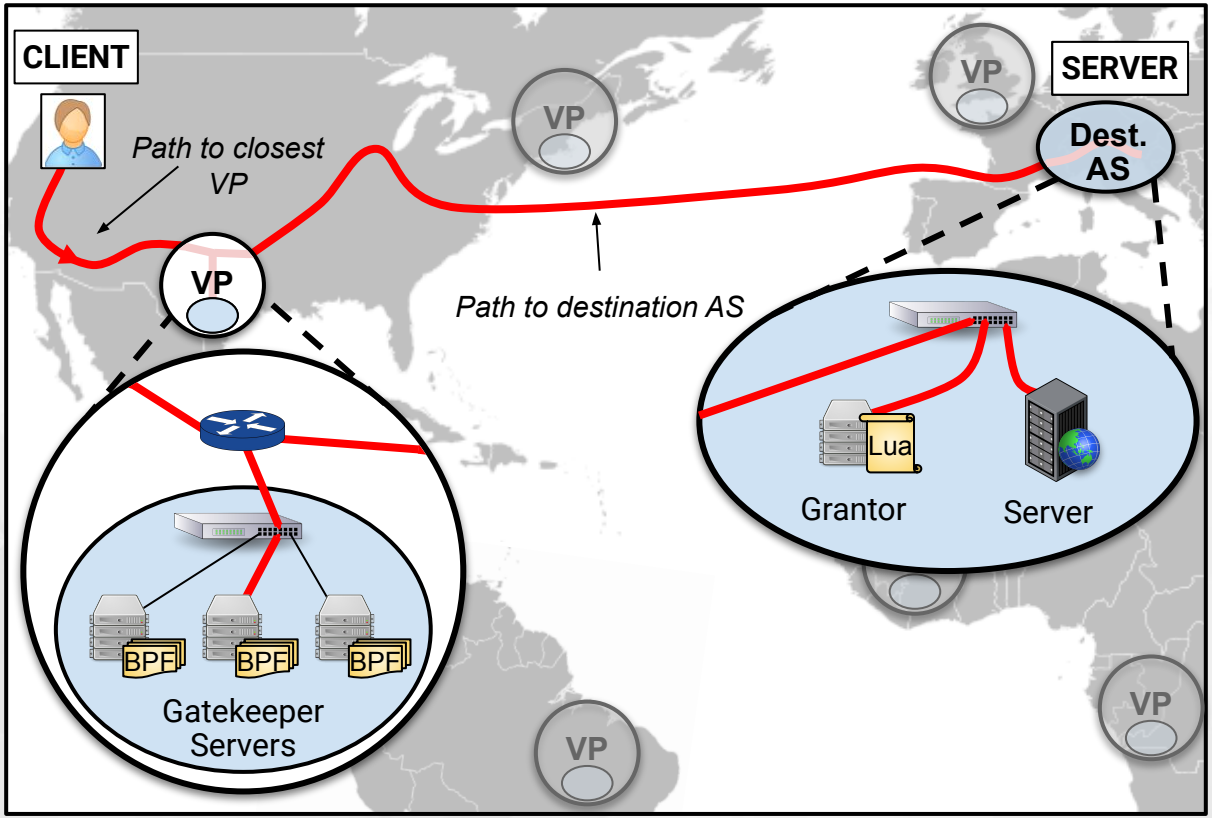


Gatekeeper servers: upstream policy enforcement

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

Gatekeeper

Quick summary

1. Packets from clients are forwarded to the closest VPs
2. Gatekeeper servers forward packets of new flows to Grantor servers, or run BPF programs to decide what to do
3. Grantor servers run a policy to map flows to BPF programs, and forward granted packets to destinations
4. Grantor servers notify Gatekeeper servers of all policy decisions
5. Gatekeeper servers enforce the police decisions

Gatekeeper

Quick summary

1. Packets from clients are forwarded to the closest VPs
2. Gatekeeper servers forward packets of new flows to Grantor servers, or run BPF programs to decide what to do
3. Grantor servers run a policy to map flows to BPF programs, and forward granted packets to destinations
4. Grantor servers notify Gatekeeper servers of all policy decisions
5. Gatekeeper servers enforce the police decisions

Gatekeeper

Outline

- ✓ Motivation
- ✓ How Gatekeeper works
- How to write a destination policy
- Mitigating a SYN flood
- Conclusion

Gatekeeper

Step 1: identify ALL your network profiles

A profile may apply:

to a single server, a group of servers, or

to blocks of IP addresses

Example of a profile: outgoing email servers

- No listening sockets
- Very small ingress traffic footprints

Sources: config files, production servers, docs

Gatekeeper

Step 2: write an BPF program for each profile

Classify packets into one of these bins:

- **Primary:** main purpose of the service
- **Secondary:** needed packets (e.g. TCP SYN, ICMP)
- **Unwanted:** please guess :-)

Enforce primary bandwidth limit before classification

Enforce secondary bandwidth limit after classification on secondary packets

Step 1: Network profiles ➔ Step 2: BPF programs ➔ Step 3: Lua Policy

Gatekeeper

Step 3: map flows to your BPF programs

Just classify flows using the destination IP address

Example: 10.99.99.128/25 are outgoing email servers

This information is a byproduct of Step 1

Grantor servers run this part of the policy (Lua policy)

Gatekeeper

Step 3: map flows to your BPF programs (bonus)

Classify source IP addresses too!

- Reject bogons, abusers, malware
- Tune bandwidth to partners, countries, end users
- Return different profiles to CDNs, crawlers, offices

Manage all your IP ranges with Drib:
https://github.com/andrenth/drib

Gatekeeper

Outline

- ✓ Motivation
- ✓ How Gatekeeper works
- ✓ How to write a destination policy
- Mitigating a SYN flood
- Conclusion

Gatekeeper

A modest testbed on AWS

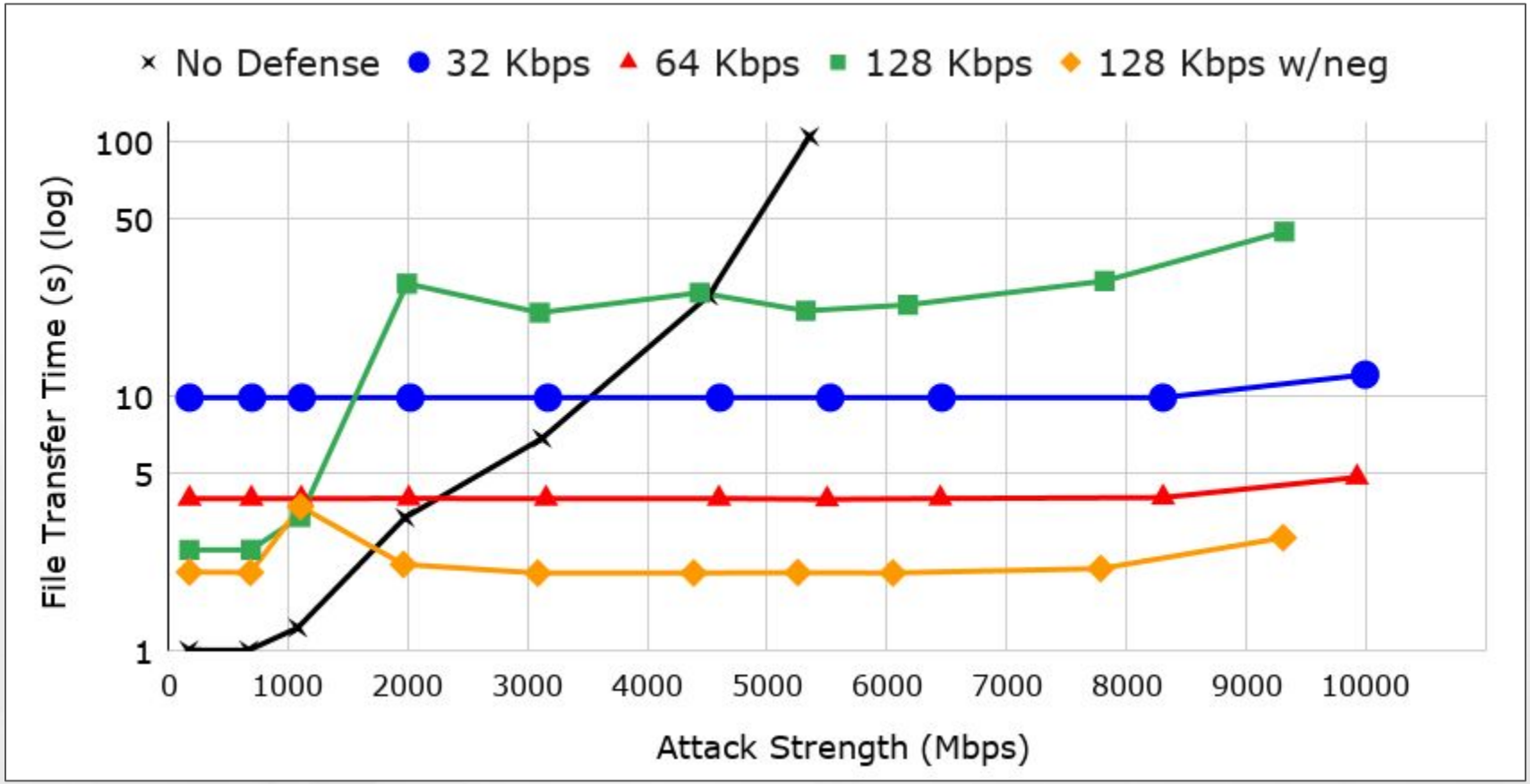
2x Packet generators forging 16K source IP addresses
1x Legit client uploading a 20KB file 50 times

1x Gatekeeper server
1x Router

1x Grantor server
1x Destination web server

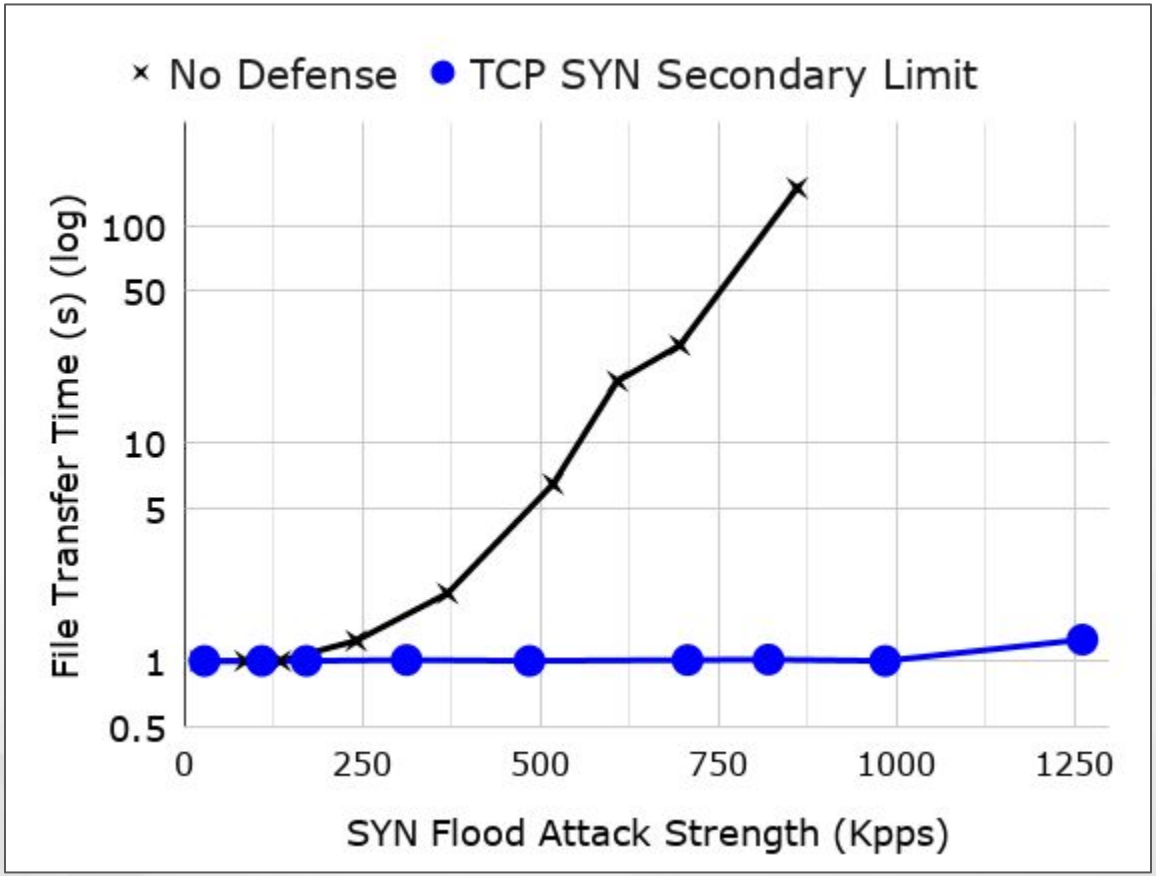
Gatekeeper

Blind limits per flow are effective



Gatekeeper

Secondary limits thwart SYN floods



Gatekeeper

Outline

- ✓ Motivation
- ✓ How Gatekeeper works
- ✓ How to write a destination policy
- ✓ Mitigating a SYN flood
- Conclusion

Gatekeeper

Some future work

Supporting 100 Gbps NICs at line speed

⇒ Cheaper deployments

Supporting load balancing in policies

⇒ Better return on investment

Flow orchestration

⇒ Insurance for endgame

Gatekeeper

Wrap-up

Unparalleled multi-vector protection

Mitigation in seconds

Scalable, open source, and ready for deployment

Impactful features in store for the future

Gatekeeper

https://github.com/AltraMayor/gatekeeper