Datagrams vs. Connections
Datagrams vs. Connections

- Connection-less network layer
  - flexibility, simplicity
  - best-effort service
Datagrams vs. Connections

• Connection-less network layer
  - flexibility, simplicity
  - best-effort service

• Connection-oriented network layer
  - end-to-end guarantees
  - more mechanism in routers, connection setup
Bandwidth Flooding Attacks
Bandwidth Flooding Attacks
Bandwidth Flooding Attacks

• Victim's link flooded with malicious traffic
Bandwidth Flooding Attacks

- Victim's link flooded with malicious traffic
- Legitimate TCP clients back off
Datagrams vs. Connections
Datagrams vs. Connections

• Datagram approach
  - allow all, explicitly deny bad traffic
  - use filtering to block bad traffic
Datagrams vs. Connections

• Datagram approach
  - allow all, explicitly deny bad traffic
  - use filtering to block bad traffic

• Connection-oriented (capability) approach
  - deny (or limit) all, explicitly allow good traffic
  - use network-layer connections to shield good traffic
What about Connection Setup?
What about Connection Setup?

- Must protect connection setup against DoS
What about Connection Setup?

- Must protect connection setup against DoS
- Necessarily datagram traffic
What about Connection Setup?

- Must protect connection setup against DoS
- Necessarily datagram traffic
- Need datagram DoS solution
What about Connection Setup?

- Must protect connection setup against DoS
- Necessarily datagram traffic
- Need datagram DoS solution
- Can use to protect *all* datagrams
What about Connection Setup?

- Must protect connection setup against DoS
- Necessarily datagram traffic
- Need datagram DoS solution
- Can use to protect *all* datagrams

Once datagram DoS solution is deployed, connections become unnecessary
The Datagram Approach
The Datagram Approach

- Explicitly filter traffic from bad sources
The Datagram Approach

- Explicitly filter traffic from bad sources
- **Securely** move filtering state close to sources
  - Active Internet Traffic Filtering (USENIX '05)
Capabilities: Stateless Connections
Capabilities: Stateless Connections

marking/verification nodes

srv

cli
Capabilities: Stateless Connections

capability request
Capabilities: Stateless Connections

capability request

capability

srv

cli
Capabilities: Stateless Connections

- Ticket to send $n$ bytes within $t$ seconds
Capabilities: Stateless Connections

- Ticket to send $n$ bytes within $t$ seconds
- No filtering state, no special inter-ISP relationships
Capabilities: Stateless Connections

- Ticket to send \( n \) bytes within \( t \) seconds
- No filtering state, no special inter-ISP relationships

Elegant and easy to deploy
DoS with Capability Requests

- Can flood victim with capability requests
DoS with Capability Requests

- Can flood victim with capability requests
DoS with Capability Requests

- Can flood victim with capability requests
- New client has trouble connecting to site
DoS with Capability Requests

- Can flood victim with capability requests
- New client has trouble connecting to site

Denial of Capability
Setup vs. General Traffic
Setup vs. General Traffic

• Are setup requests easier to protect?
  - more resistant to loss
  - more predictable
Setup vs. General Traffic

• Are setup requests easier to protect?
  – more resistant to loss
  – more predictable

• Our position: Setup traffic is not different
  – with respect to vulnerability to DoS
  – and means required to protect it
Is Connection Setup Resistant to Loss?
Is Connection Setup Resistant to Loss?

- Assume victim knows good clients
Is Connection Setup Resistant to Loss?

- Assume victim knows good clients
- A single setup request must get through
Is Connection Setup Resistant to Loss?

• Assume victim knows good clients
• A single setup request must get through
• Can retransmit setup request until connected
Is Connection Setup Resistant to Loss?

- Assume victim knows good clients
- A single setup request must get through
- Can retransmit setup request until connected
- Probability of failure decreases exponentially
Is Connection Setup Resistant to Loss?

2.5 Gbps attack traffic
Is Connection Setup Resistant to Loss?

- Good client retransmits every second
Is Connection Setup Resistant to Loss?

- Good client retransmits every second
- Expected time to connection is over 8 minutes
Is Connection Setup Resistant to Loss?

- Good client retransmits every second
- Expected time to connection is over 8 minutes

Response time suffers
Is Setup Traffic Policeable?
Is Setup Traffic Policeable?

- Attack sources send more than good sources
Is Setup Traffic Policeable?

- Attack sources send more than good sources
- **Fair-queue** setup requests
Is Setup Traffic Policeable?

- Attack sources send more than good sources
- **Fair-queue** setup requests
- Each source gets same share of receiver's bwdth
Is Setup Traffic Policeable?

- Fair-queuing *per incoming interface*
Is Setup Traffic Policeable?

- Fair-queuing per incoming interface
- Ineffective during highly distributed attacks
Is Setup Traffic Policeable?

- Fair-queuing per source
Is Setup Traffic Policeable?

- Fair-queuing per source
- Similar state with per-source filtering
Is Setup Traffic Policeable?

- Fair-queuing per source
- Similar state with per-source filtering

At the cost of simplicity and deployability
The Datagram Approach

- Explicitly filter setup requests from bad sources
The Datagram Approach

- Explicitly filter setup requests from bad sources
- Explicitly filter all traffic from bad sources
The Datagram Approach

- Explicitly filter setup requests from bad sources
- Explicitly filter all traffic from bad sources

Connections become unnecessary
Capabilities as an Optimization
• At least connected clients are unaffected by attack
Unless there Are Lots of Bad Guys
Unless there Are Lots of Bad Guys

• *Undetected* bad sources acquire capabilities
Unless there Are Lots of Bad Guys

- **Undetected** bad sources acquire capabilities
- Victim must decide how to split bandwidth
Unless there Are Lots of Bad Guys

- **Undetected** bad sources acquire capabilities
- Victim must decide how to split bandwidth
- **Randomly** chooses which capabilities to renew
Unless there Are Lots of Bad Guys

- **Undetected** bad sources acquire capabilities
- Victim must decide how to split bandwidth
- **Randomly** chooses which capabilities to renew
- Good clients lose to bad sources
Unless there Are Lots of Bad Guys

• **Undetected** bad sources acquire capabilities
• Victim must decide how to split bandwidth
• **Randomly** chooses which capabilities to renew
• Good clients lose to bad sources

**Undetected bad sources can always harm good traffic**
Capabilities = Reservations
Capabilities = Reservations

• Sender reserves receiver's bandwidth
Capabilities = Reservations

- Sender reserves receiver's bandwidth
- Challenge: make the “right” reservation
Capabilities = Reservations

- Sender reserves receiver's bandwidth
- Challenge: make the “right” reservation
- Large botnets: each attack source sends low rate
Capabilities = Reservations

- Sender reserves receiver's bandwidth
- Challenge: make the “right” reservation
- Large botnets: each attack source sends low rate
- Less relevant to restrict per-sender bandwidth
Capabilities $=\text{Reservations}$

- Sender reserves receiver's bandwidth
- Challenge: make the “right” reservation
- Large botnets: each attack source sends low rate
- Less relevant to restrict per-sender bandwidth
- More relevant to monitor traffic patterns
Conclusions

- Connections can protect good traffic against DoS
- Connection-setup relies on datagrams
  - must protect datagrams against DoS
- Connections become unnecessary
Conclusions

- Connections can protect good traffic against DoS
- Connection-setup relies on datagrams
  - must protect datagrams against DoS
- Connections become unnecessary
- Capabilities may be useful optimization
  - must compute the “right” capability for each source