Save the Internet from SVCs (Stealth Virtual Circuits)

Katerina Argyraki + David R Cheriton

The need for traffic policies

- TX policies for dependable routing
  - route around failures + untrusted domains
  - sender must control path of outgoing traffic
- RX policies for DDoS protection
  - identify + block bad sources + domains
  - receiver must know path of incoming traffic

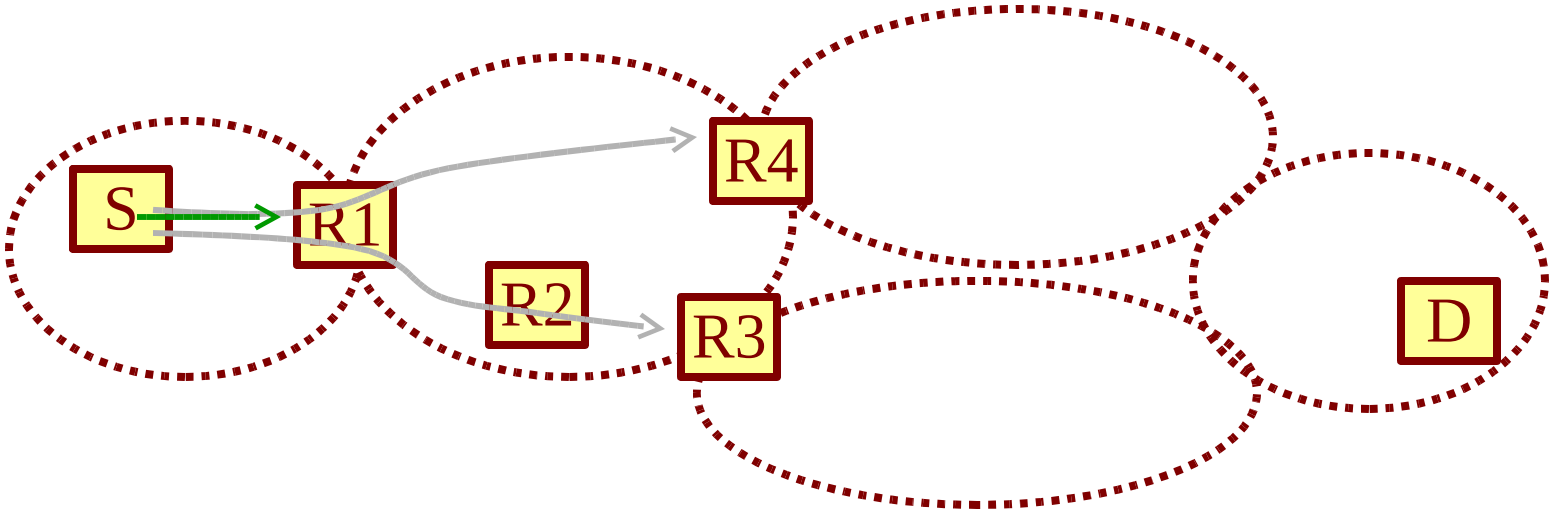
We need outgoing + incoming path control

Proposed solutions

- TX policies with policy labels
  - ToS field in IPv4,
  - DiffServ, IPv6 flow labels
- RX policies with hop-by-hop traceback
  - Push filtering of DDoS traffic hop-by-hop into the network

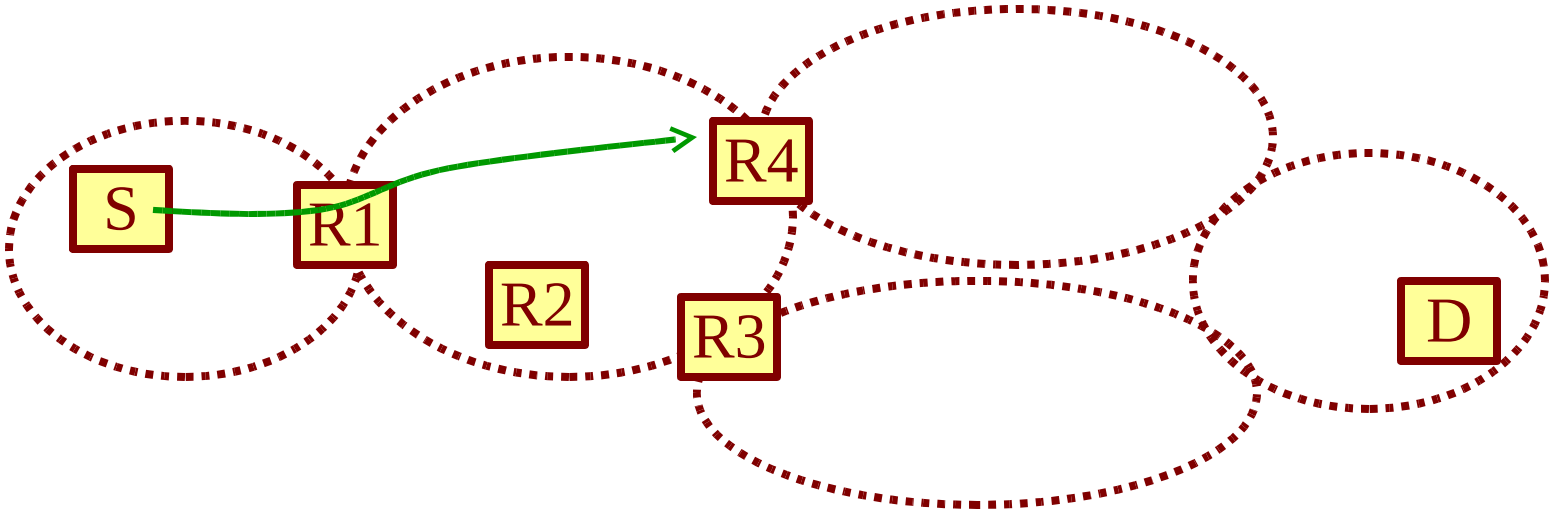
Hidden requirement for network-layer virtual circuits = *stealth virtual circuits*

Policy labels: the theory



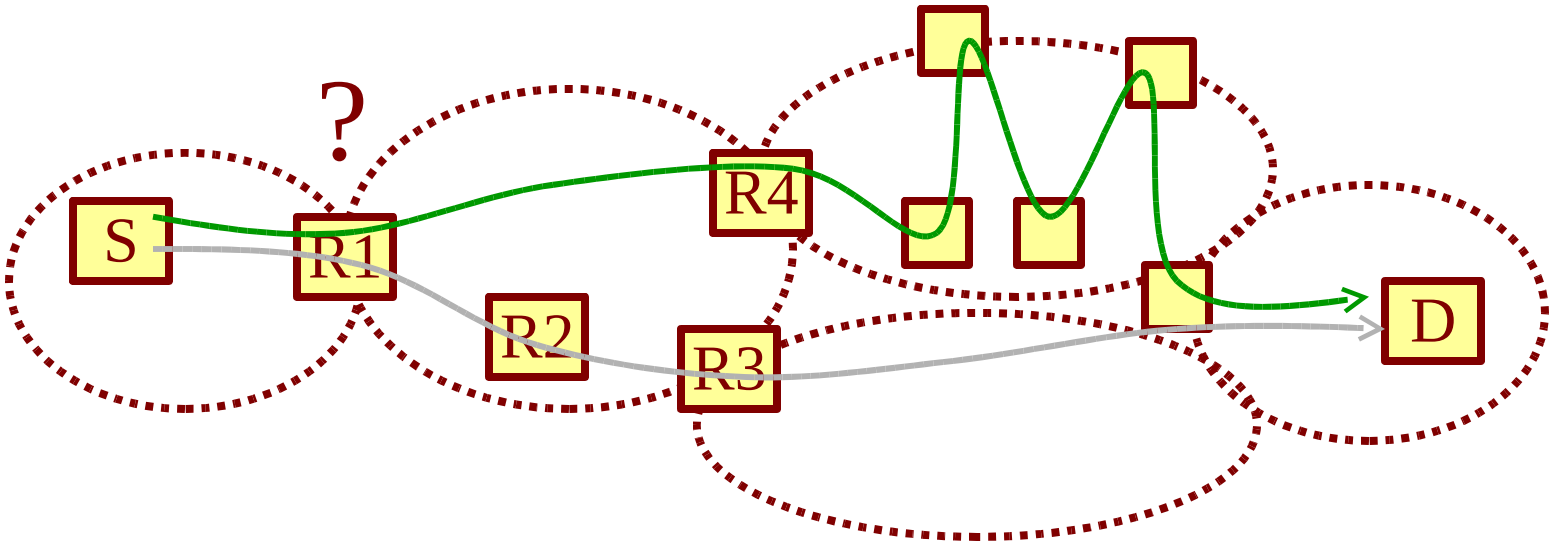
- Sender tags packet with label
  - e.g., “low-delay”
- Each ISP maps label to fwd’ing behavior
  - e.g., chooses lowest delay path

Policy labels: the theory



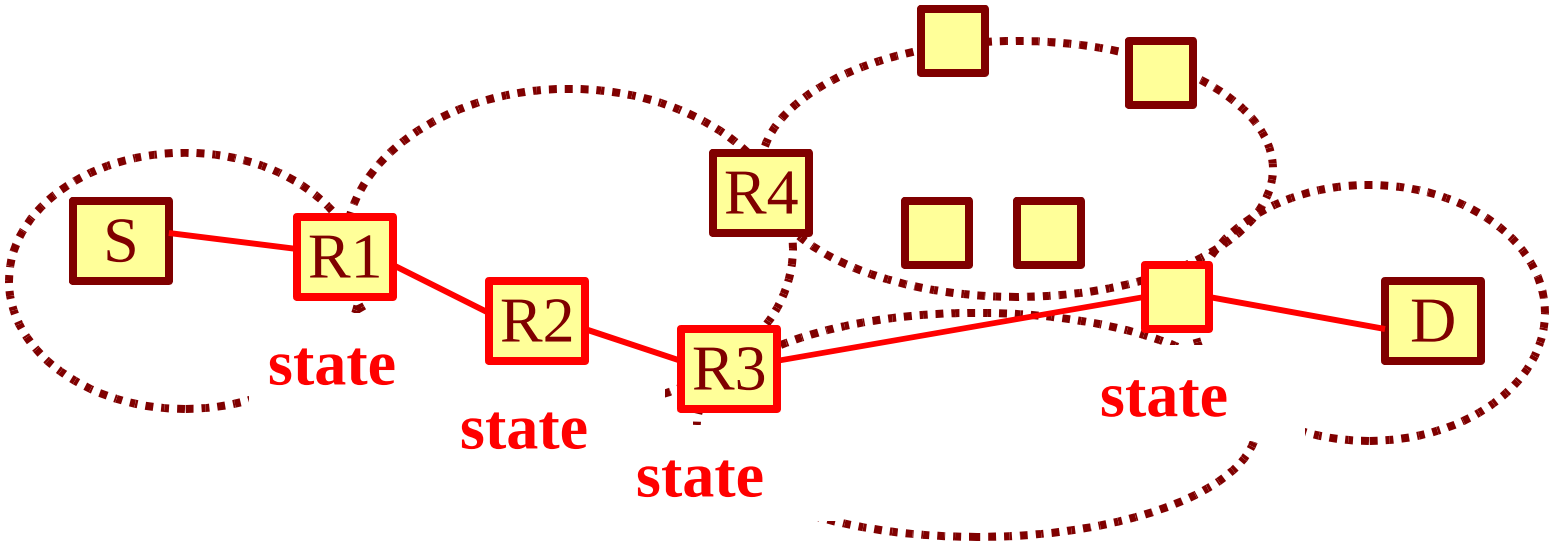
- Sender tags packet with label
  - e.g., “low-delay”
- Each ISP maps label to fwd’ing behavior
  - e.g., chooses lowest delay path

Policy labels: the catch



- ISP has no clue about end-to-end paths
  - only local view of its own network
- Unless...

Policy labels: the truth



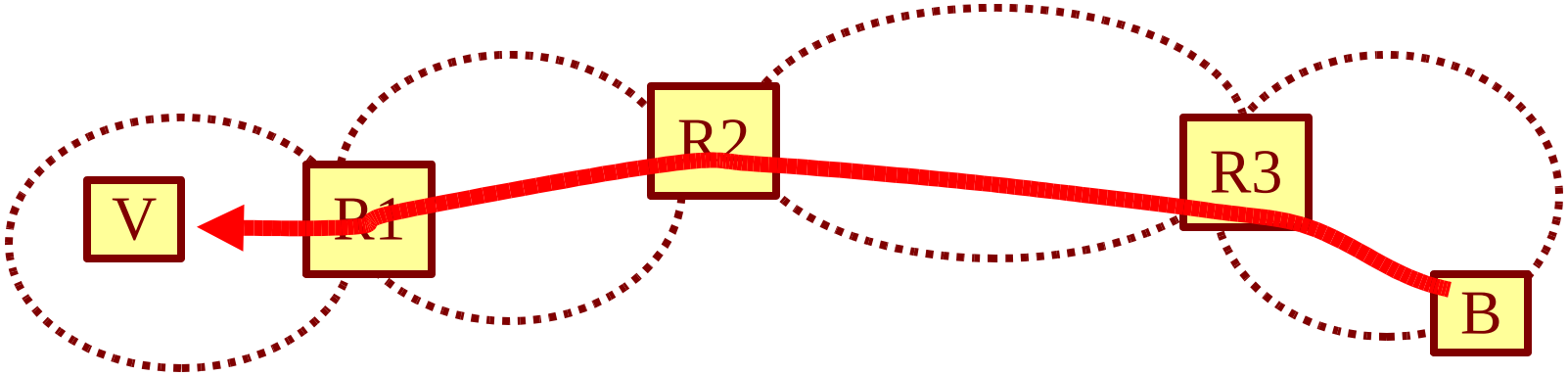
- Routers keep end-to-end policy state
  - map label, src + dst to the right next hop
- How is this state set up?

Policy labels: the truth

- Explicitly: connection setup
  - source sends “reservation” messages
- Implicitly (stealth mode): ISPs monitor end-to-end paths
- Either way, it’s a network-layer connection

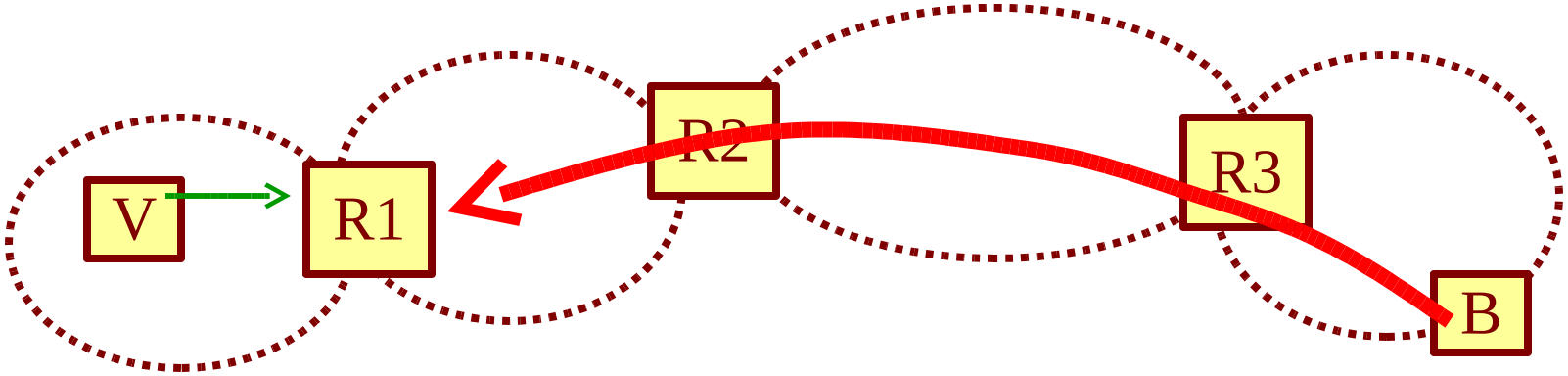
Policy labels -> stealth virtual circuits

Hop-by-hop traceback: the theory



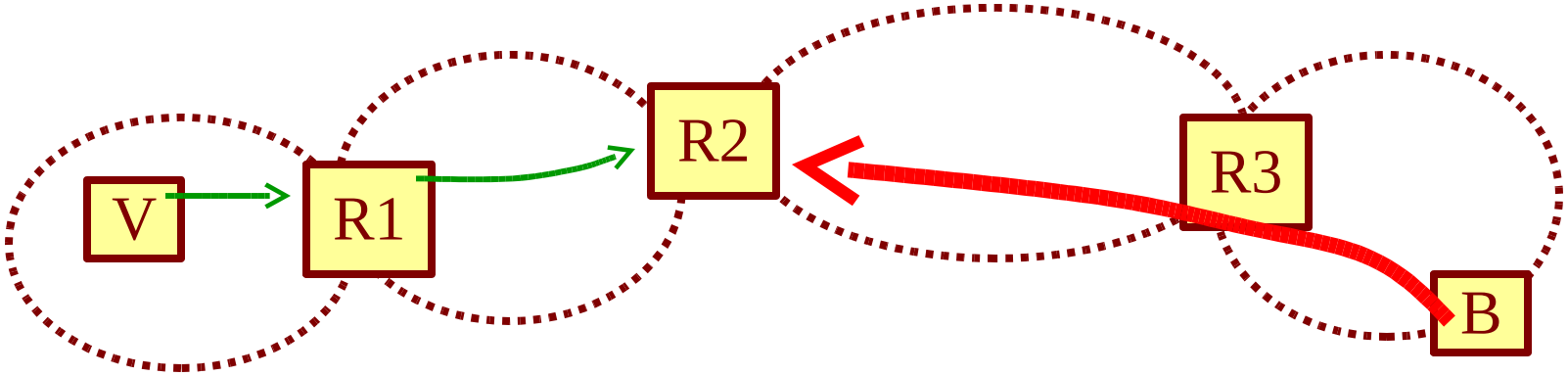
- Victim does not know where bad traffic is coming from

Hop-by-hop traceback: the theory



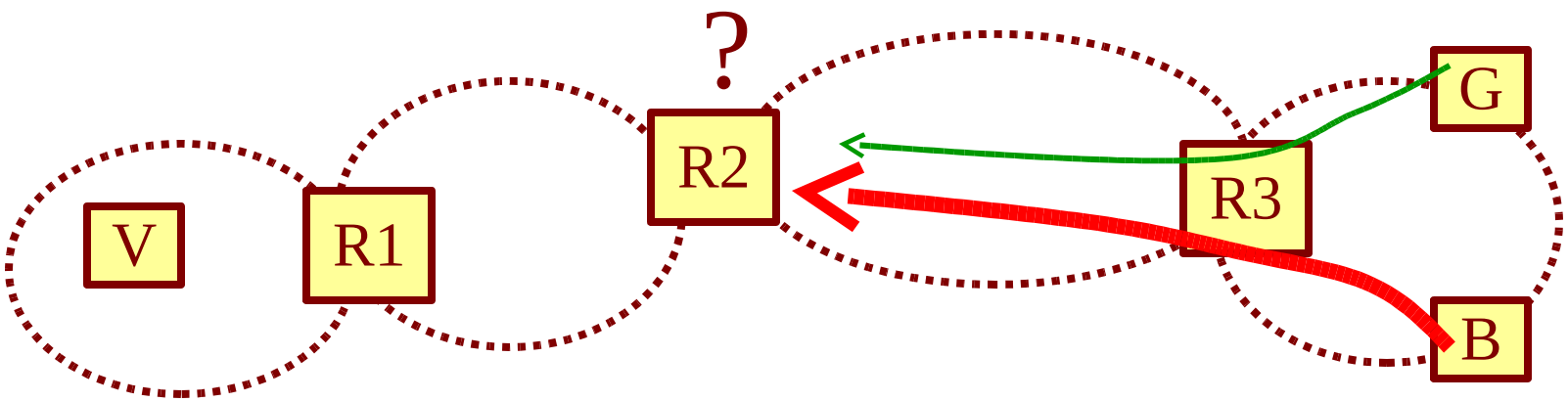
- Victim identifies last hop that fwd's bad traffic + sends filtering request

Hop-by-hop traceback: the theory



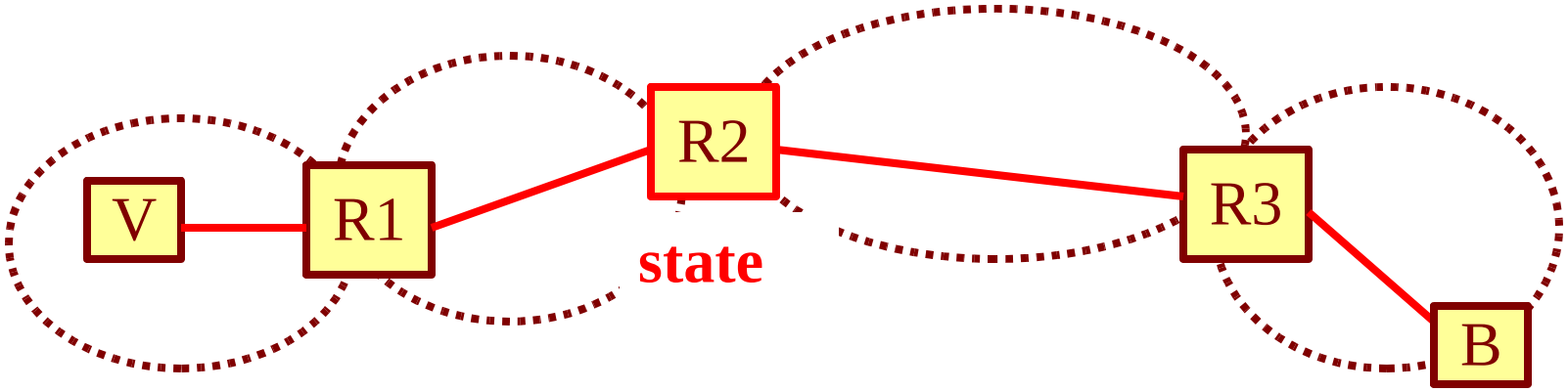
- Victim identifies last hop that fwd's bad traffic + sends filtering request
- Filtering request propagated hop-by-hop through the network

Hop-by-hop traceback: the catch



- ISP has no clue which traffic is bad
  - rate-limits all traffic to victim
- Drops most good traffic to victim
  - bad traffic rate >> good traffic rate
- Unless...

Hop-by-hop traceback: the truth



- Routers keep end-to-end filtering state
  - block traffic from source B to victim V
- That's a network-layer "anti-connection"

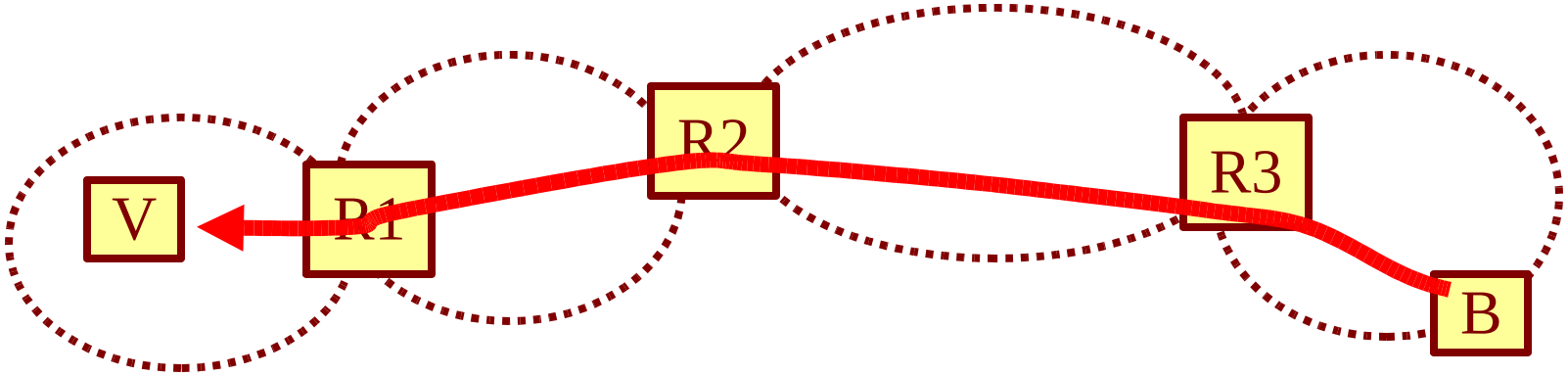
Hop-by-hop traceback -> stealth virtual circuits

SVC alternative: The datagram approach

- Add state to the packet: Loose Source Record Route (LSRR)
  - LS for TX control
  - RR for RX control
- Implement policy + control at the edges
  - sender uses LS to specify route
  - receiver uses RR to do filtering

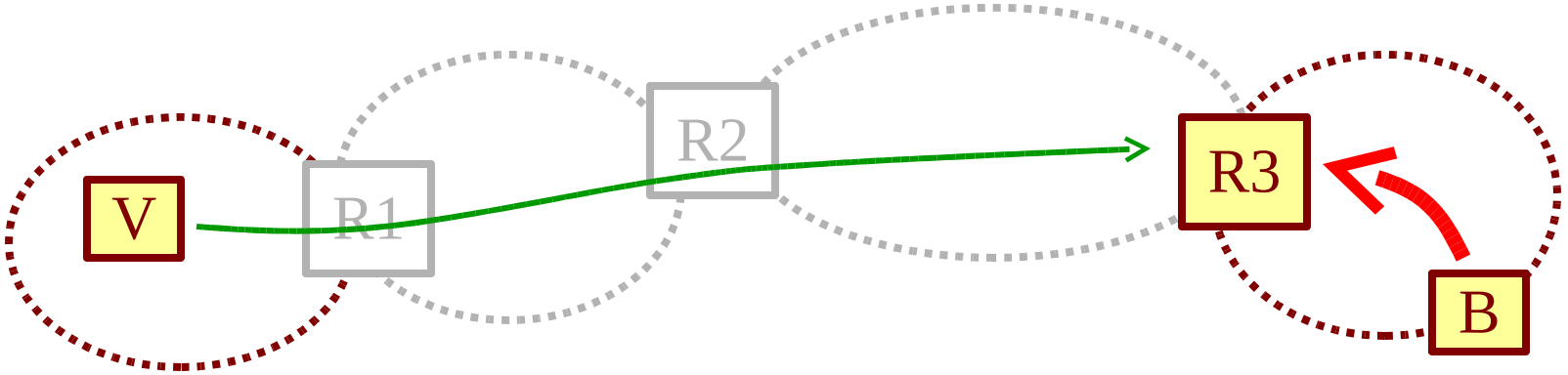
LSRR -> path control, no SVCs

Receive policies with LSRR



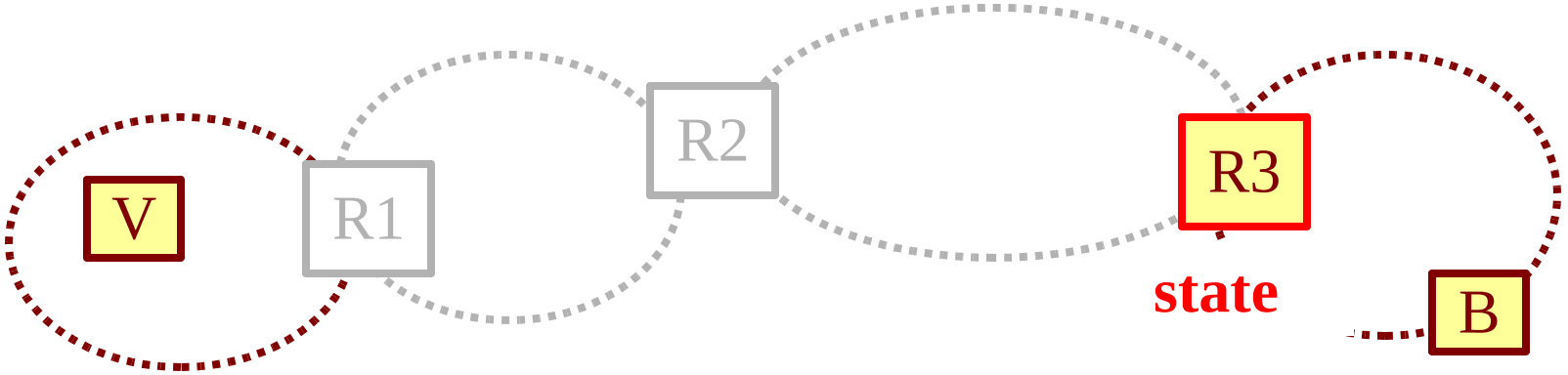
- Victim knows set of border routers on attack path from recorded route
- Victim identifies border router closest to attack source
  - call it “attack gateway”

Receive policies with LSRR



- Contacts attack gateway directly, bypassing Internet core
- Asks to block traffic from attack source to victim

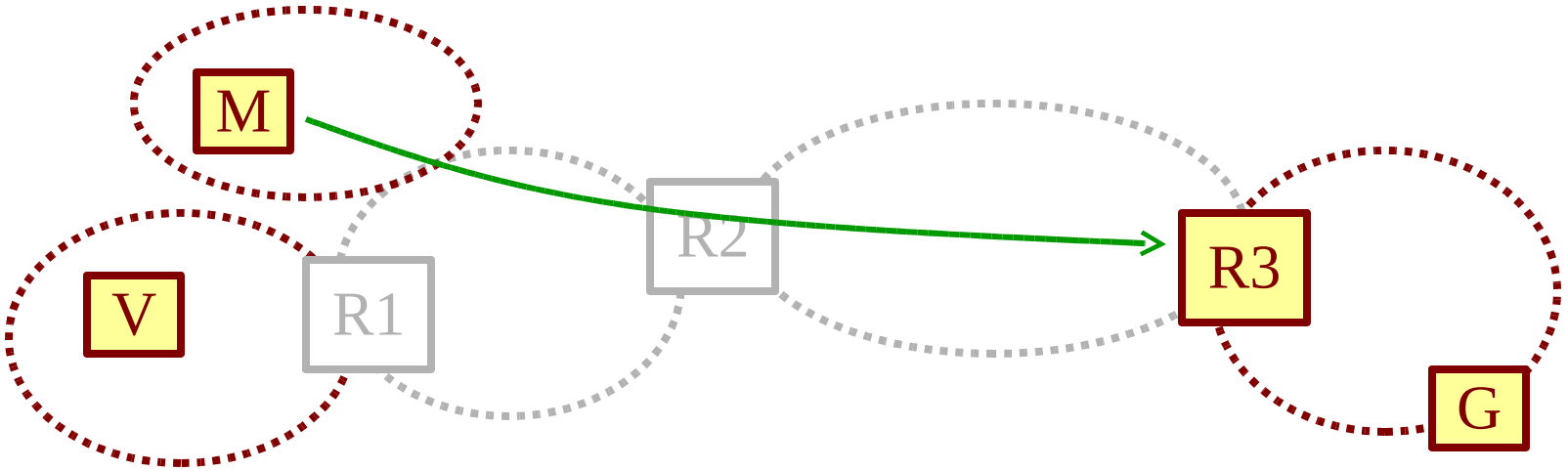
Receive policies with LSRR



- Victim contacts attack gateway directly, bypassing Internet core
- Asks to block traffic from attack source to victim

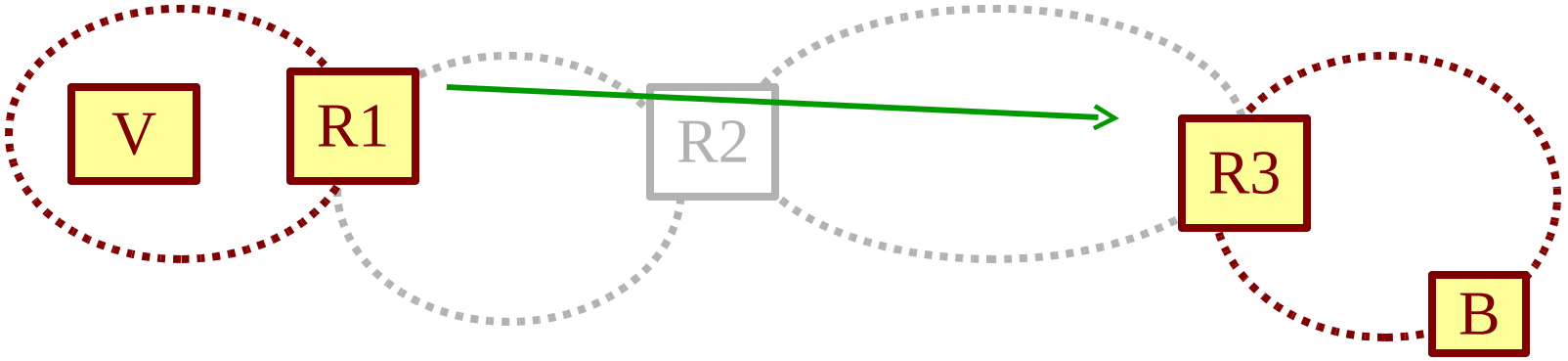
Keeps state at the edges, avoids SVCs

Isn’t this insecure?



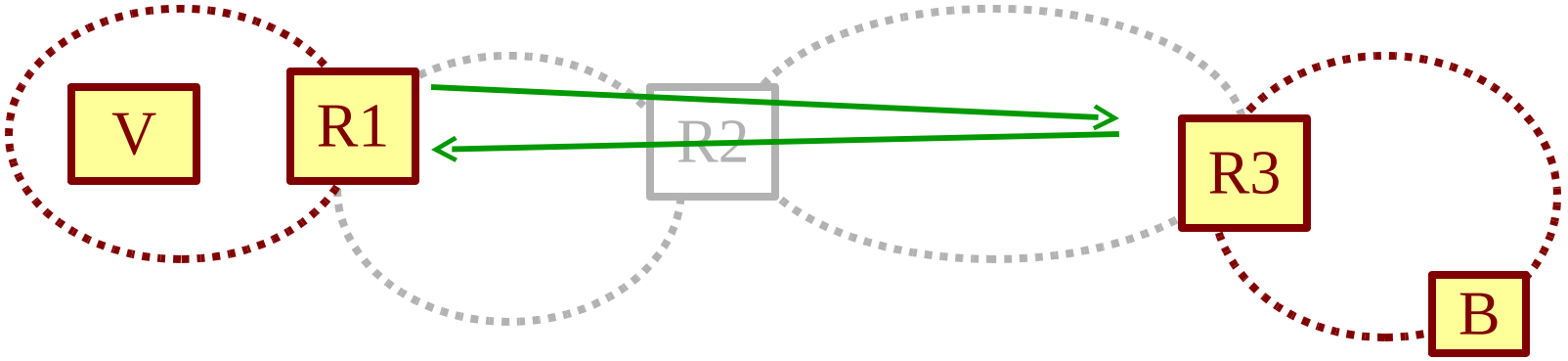
- Malicious node M pretends to be victim V
- Asks from R3 to stop traffic from G to V
- M disrupts G-V communications

3-way handshake



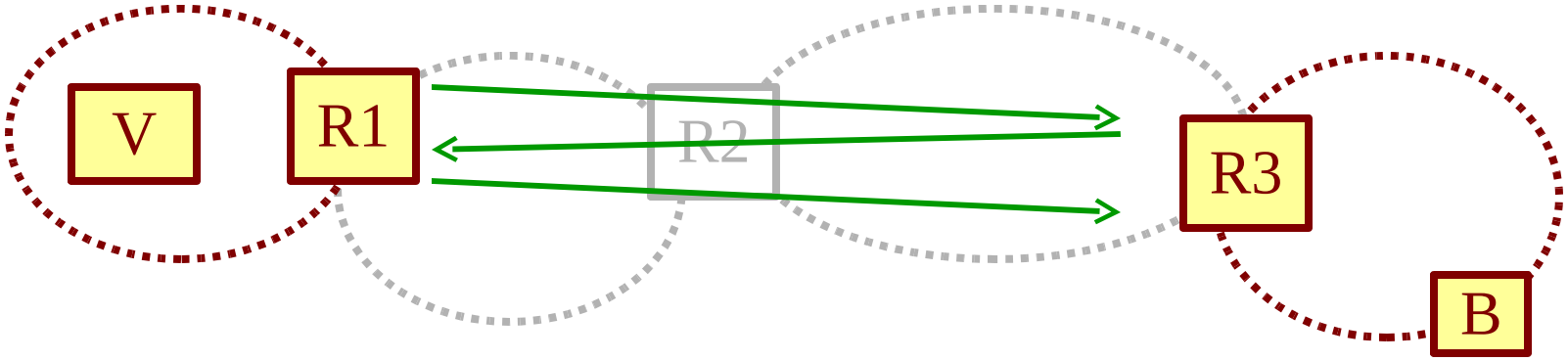
- Victim's net sends filtering request

3-way handshake



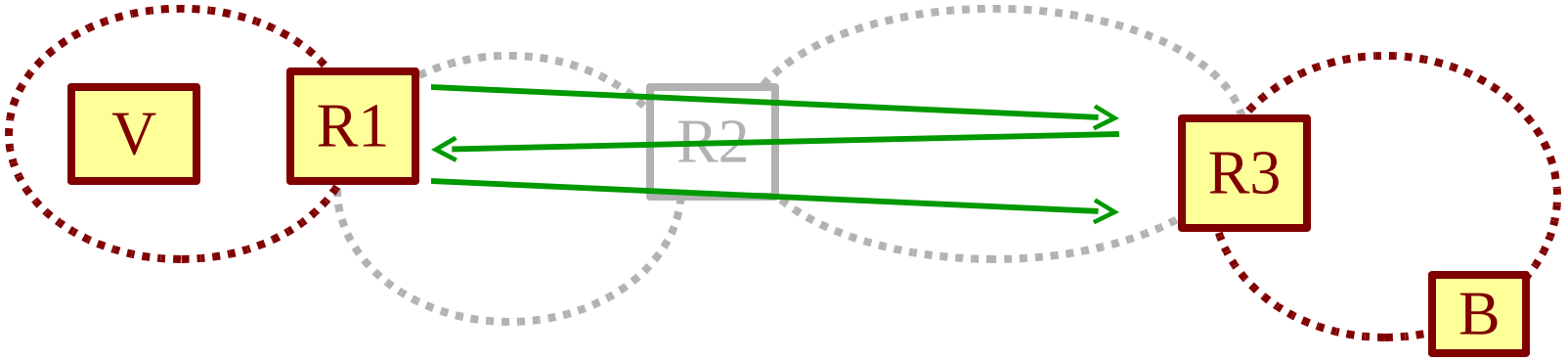
- Victim's net sends filtering request
- Attack gw responds with ACK that includes nonce

3-way handshake



- Victim's net sends filtering request
- Attack gw responds with ACK that includes nonce
- Victim's net redirects ACK with nonce

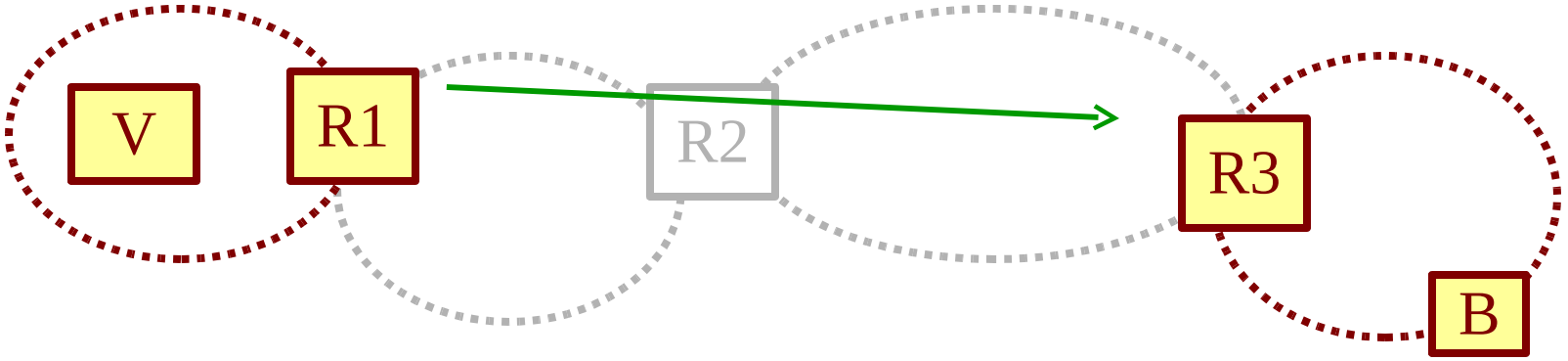
3-way handshake



- Victim's net sends filtering request
- Attack gw responds with ACK that includes nonce
- Victim's net redirects ACK with nonce

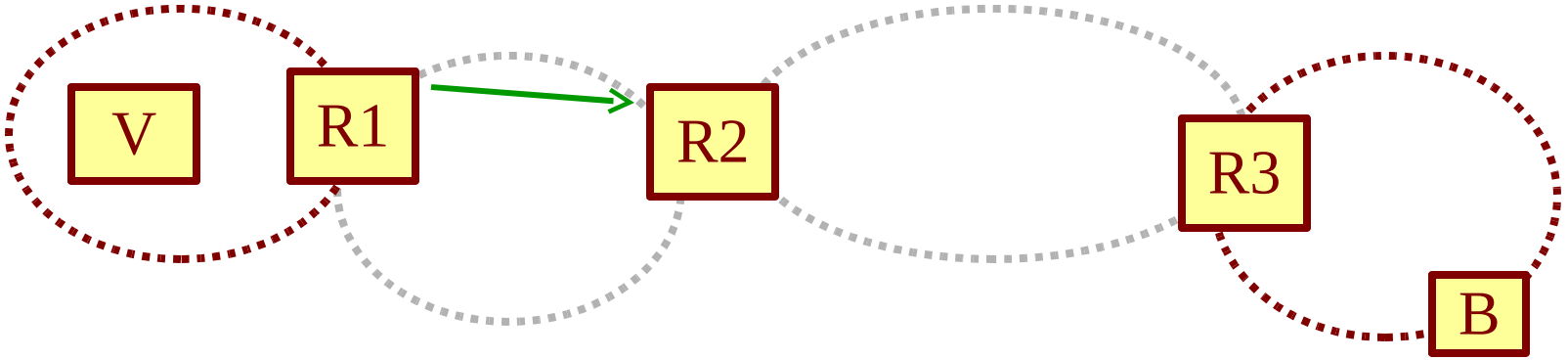
Prevents abuse by off-the-path nodes

Why cooperate?



- Attack gateway ignores request
  - it's compromised
  - it has no motive

Escalation



- Next border router contacted
- Asked to block all traffic from attack gw to victim
  - This is not an SVC, no end-to-end state
- Attack gateway cooperates or looses connectivity to victim

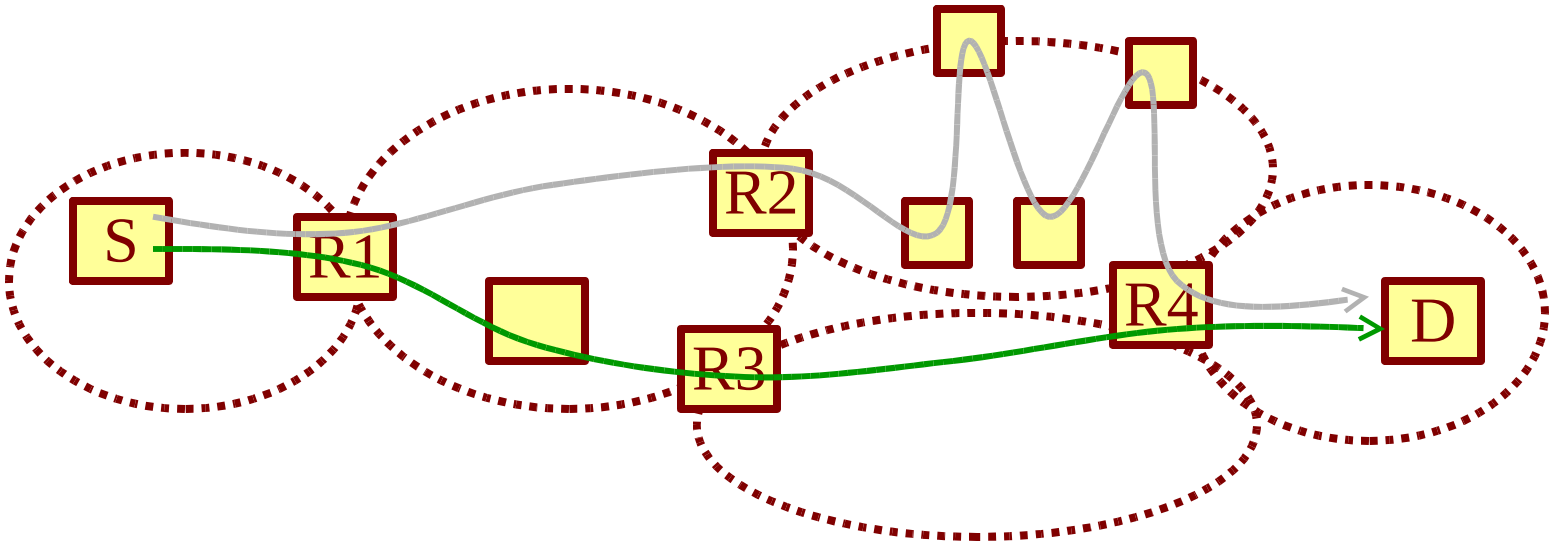
Identify the last point of trust

Receive policies with LSRR

- Expect edge-based filtering
  - routers not massively compromised
- Thousands of filters per router
  - millions of filters to block DDoS at the edges
  - thousands to block it at the core
- For details: Active Internet Traffic Filtering (AITF) [Argyraki+ Cheriton 04]

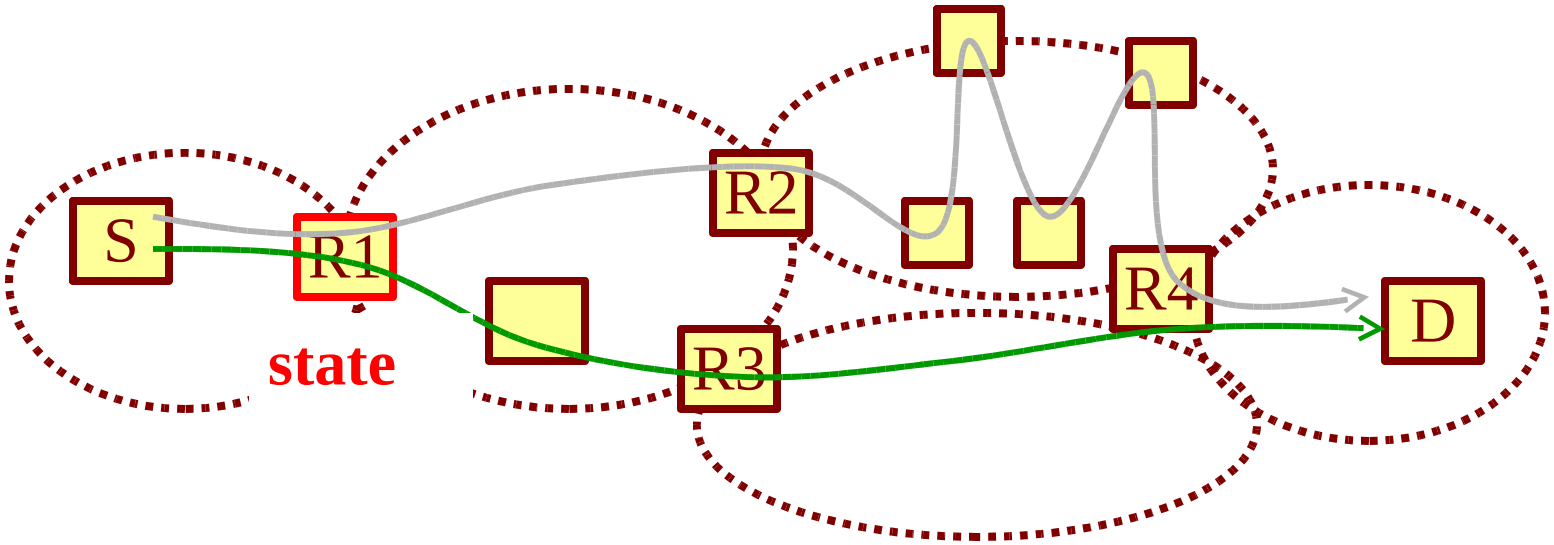
LSRR + AITF = DDoS protection without SVCs

Transmit policies with LSRR



- Edge-router computes + monitors two paths to each destination prefix
  - R1: [R2, R4], [R3, R4]

Transmit policies with LSRR



- Edge-router computes + monitors two paths to each destination
  - R1: [R2, R4], [R3, R4]

Keeps state at the edges, avoids SVCs

Can the edges handle it?

- Feedback Based Routing (FBR) [Zhu+Cheriton 03]: It is feasible to compute and maintain required state at the edges

LSRR+FBR=dependable routing without SVCs

WRAP: LSRR done right

- Shim protocol over IP, not IPv4 option
- Easy to implement in hardware
- Can filter with conventional IP filters
  - IP header: src + dst fields always refer to last + next hop
- Upgrade only border routers
  - incrementally deployable

Path control without IPv4 LSRR problems

Conclusions

- End-systems need path control for dependable routing + DDoS defense
- Many solutions compromise Internet with network-layer connections
- Loose Source Record Route seems the only way out

LSRR: keep the Internet clean of SVCs

Problems with IPv4 LSRR

- Currently implemented as IP option
  - LSRR pkts handled off the fast path
  - not meant for frequent use
- Requires special filtering capabilities
  - IP source address refers to original source, not last hop
  - cannot use conventional IP filters

IP option impl unsuitable for Internet

Current industry practice

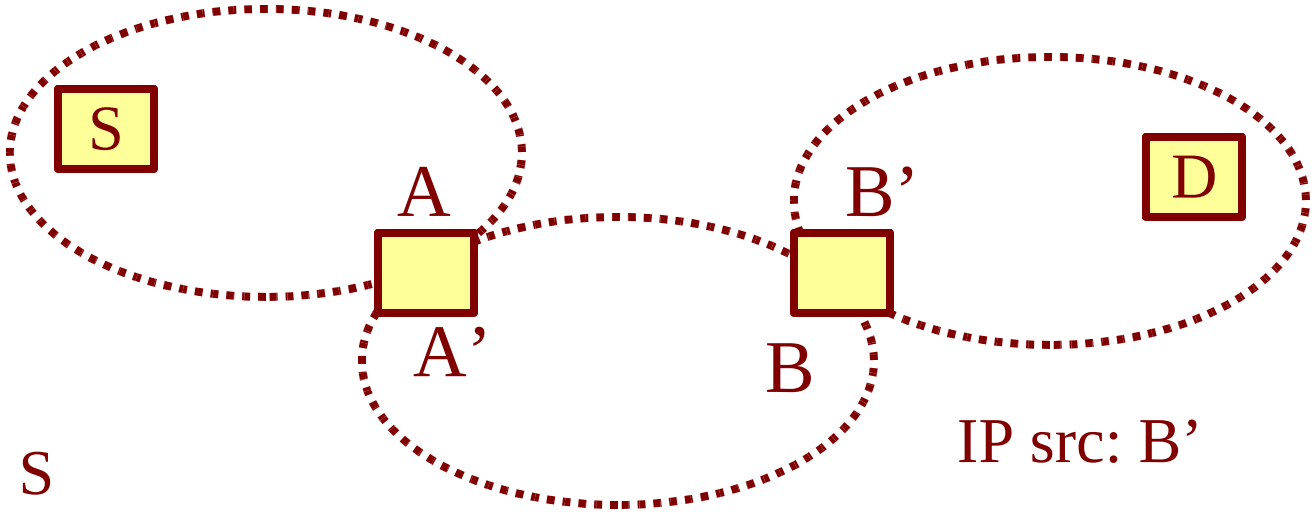
- First-hop control
  - connect to multiple ISPs + choose the one that yields best path
  - fails when ISPs use overlapping paths
- Last-hop control
  - identify last hop that forwards most undesired traffic + block
  - sacrifices good traffic

Single-hop path control is not enough

SVCs are bad for the Internet

- Quadratic growth in state requirements
  - # Internet sources x # Internet destinations
- State setup/teardown + maintenance processing overhead
  - redo on router reboot
  - handled by control plain = asking for DoS
- Edges need control anyway
  - why pay for duplicate functionality?

LSRR example



IP src: S

IP dst: A

forward path: [B,D]

recorded path: []

IP src: A’

IP dst: B

forward path: [D]

recorded path: [S]

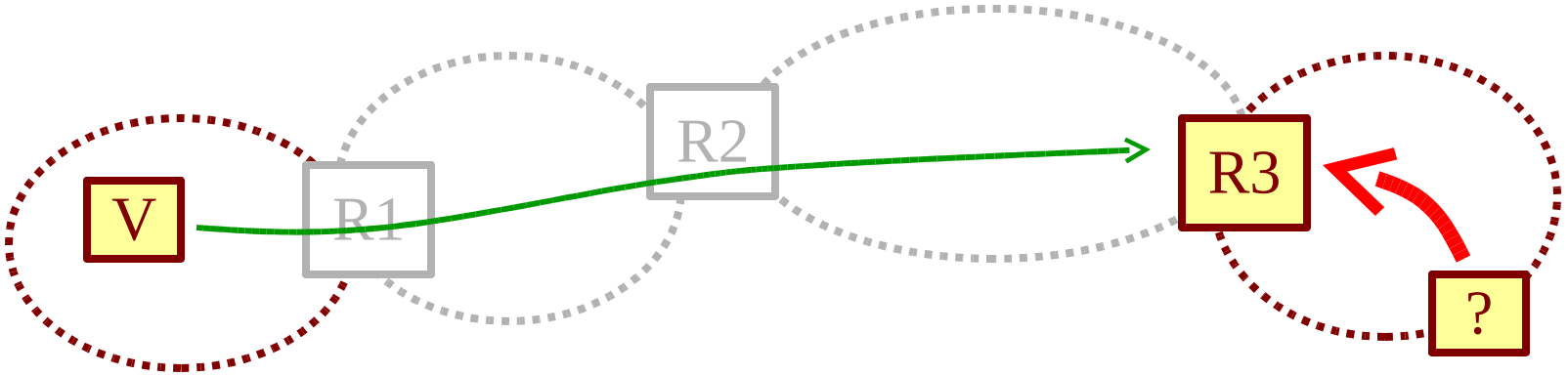
IP src: B’

IP dst: D

forward path: []

recorded path: [S,A’]

Source address spoofing effect



- Attack gateway may allow spoofing
  - Bad source *B* spoofs multiple IP addresses
- Victim can identify + block only aggregate
  - all traffic from R3 to *V*

Spoofing affects filtering granularity