Countering Denial of Service
(and why it's hard)

Katerina Argyraki, EPFL

Network systems

State

where is it stored

how is it managed

how much does it cost

Denial of service

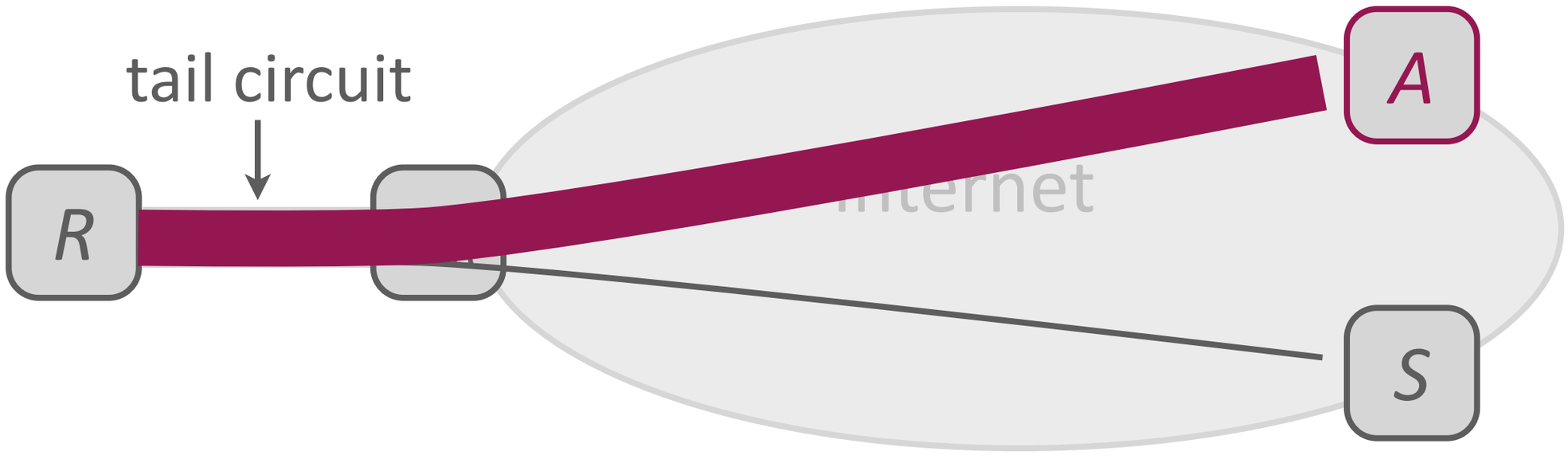
Twitter down for hours (2009)

South Ossetian news portals down for days (2007)

BetCris.com down for months (2003)

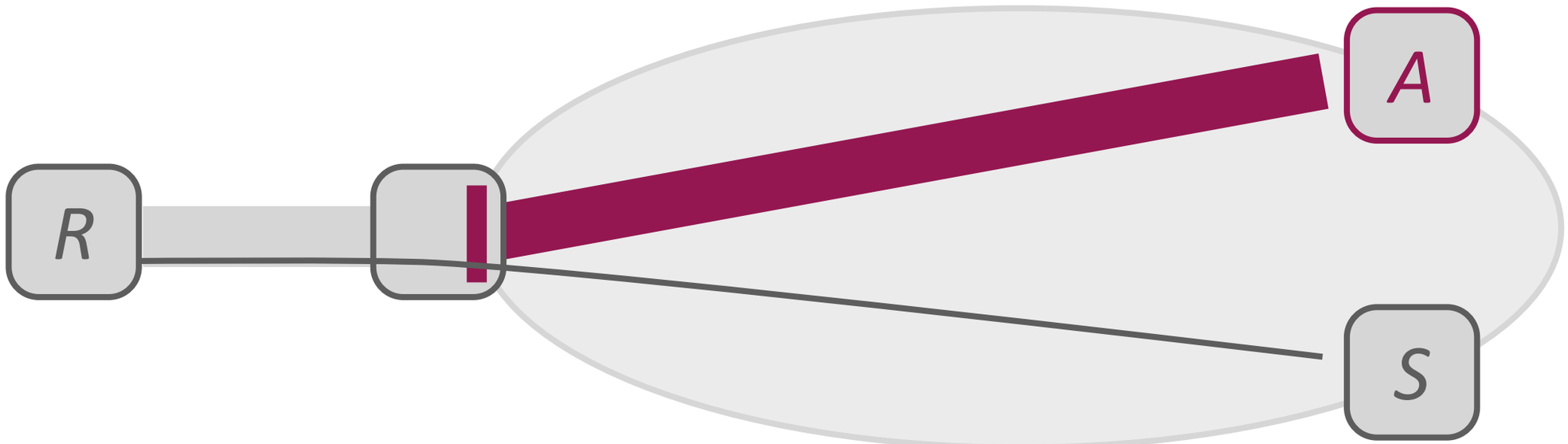
It's a big, unsolved problem

Denial of service



Target: tail-circuit bandwidth

Network filtering



State: {*A, R*}

Code: *if ({packet.src, packet.dst} in State)*
block packet;

Block attackers at the receiver's gateway

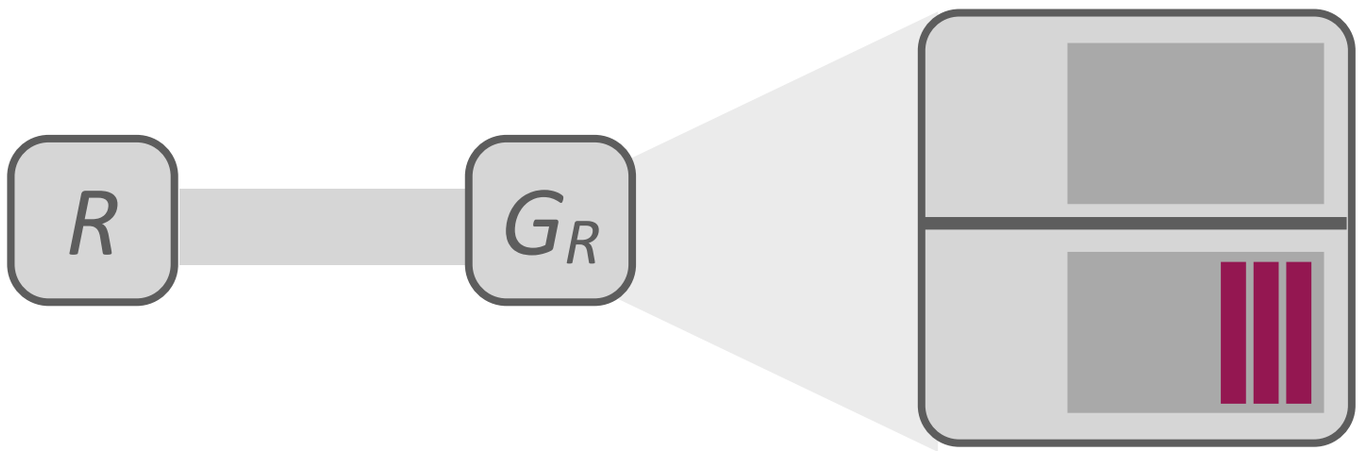
State

State: {attacker, receiver} pairs

Where: receiver's gateway

Managed: locally

Internet routers

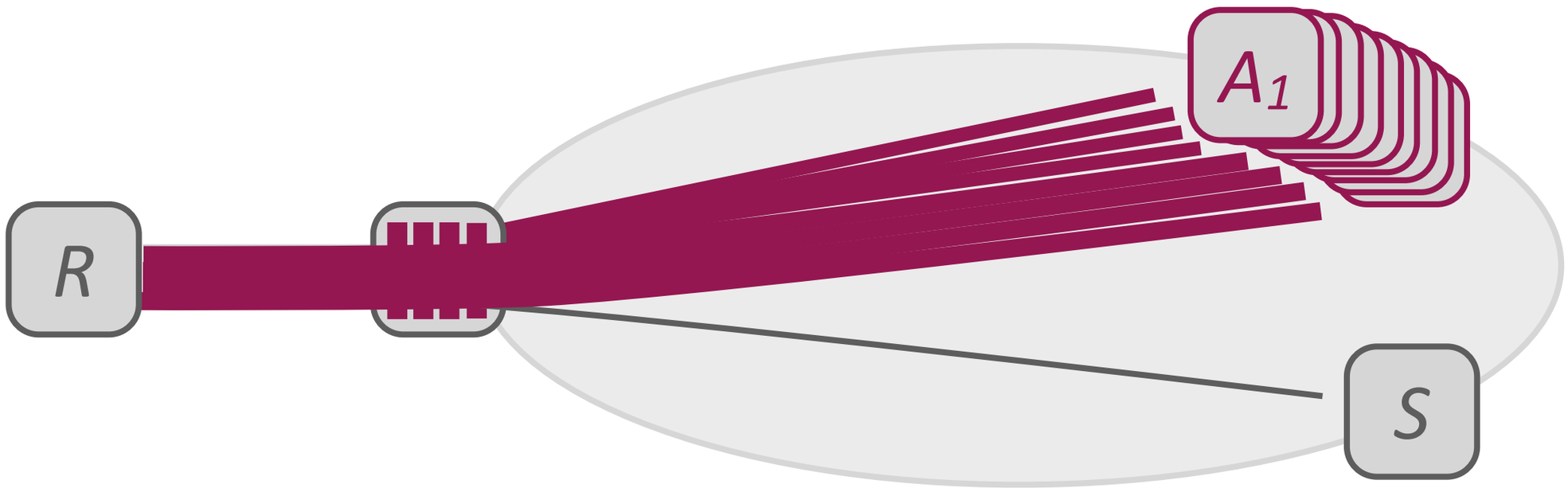


control plane
(cheap) DRAM

data plane
(expensive) SRAM

Network filtering is expensive

Distributed denial of service



Target: filtering resources + tail circuit

Distributed filtering

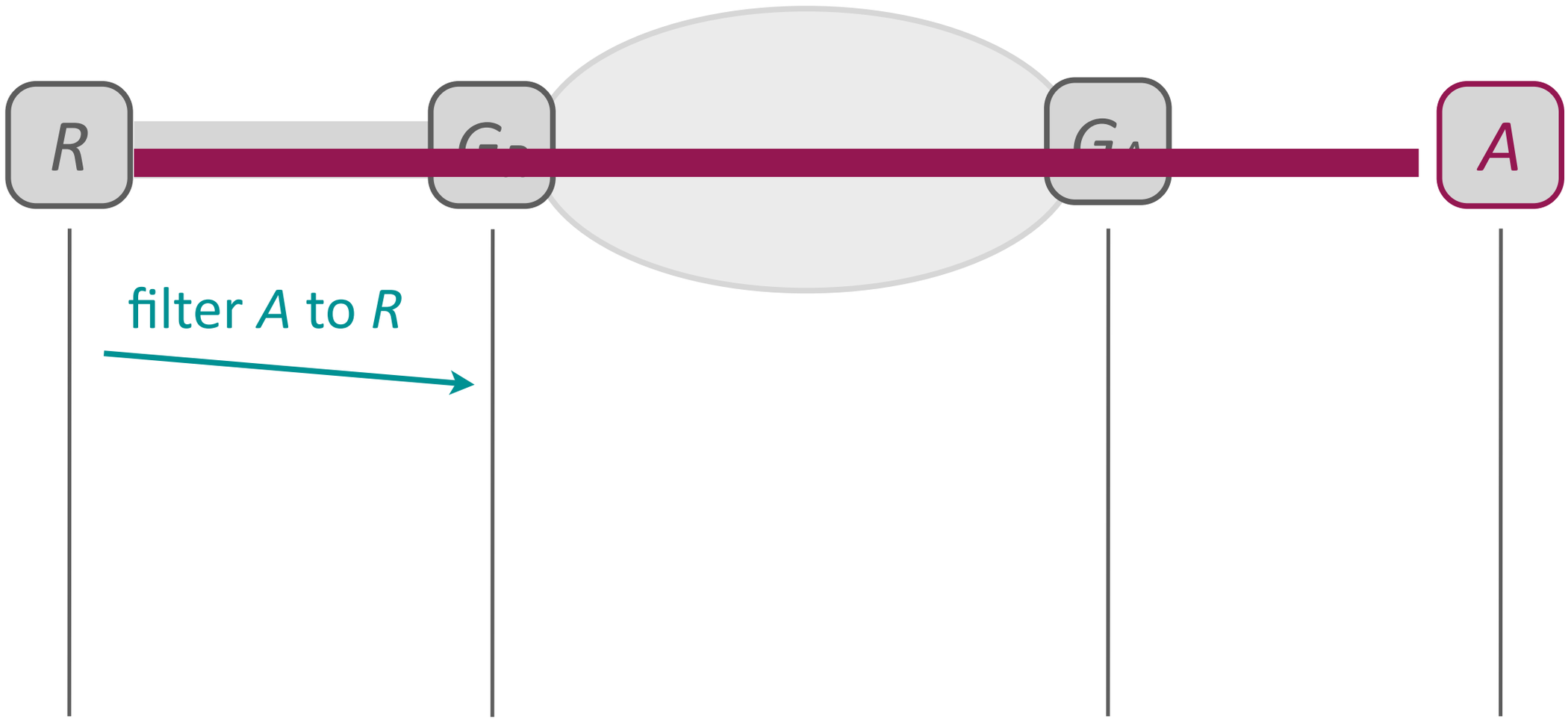
Identify routers close to attack sources

Ask them to block attack traffic

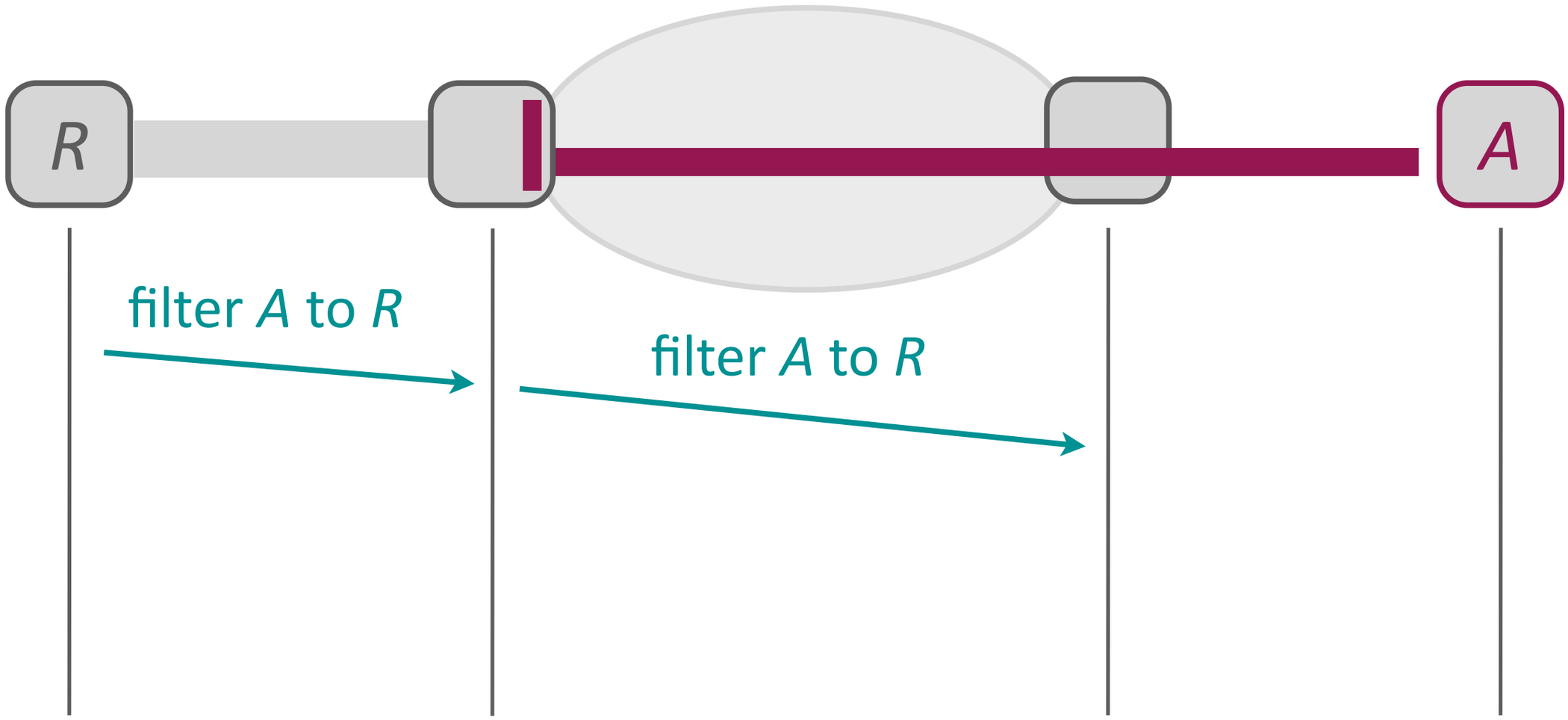
(AITF, Argyraki and Cheriton, 2005)

Need a filter-propagation protocol

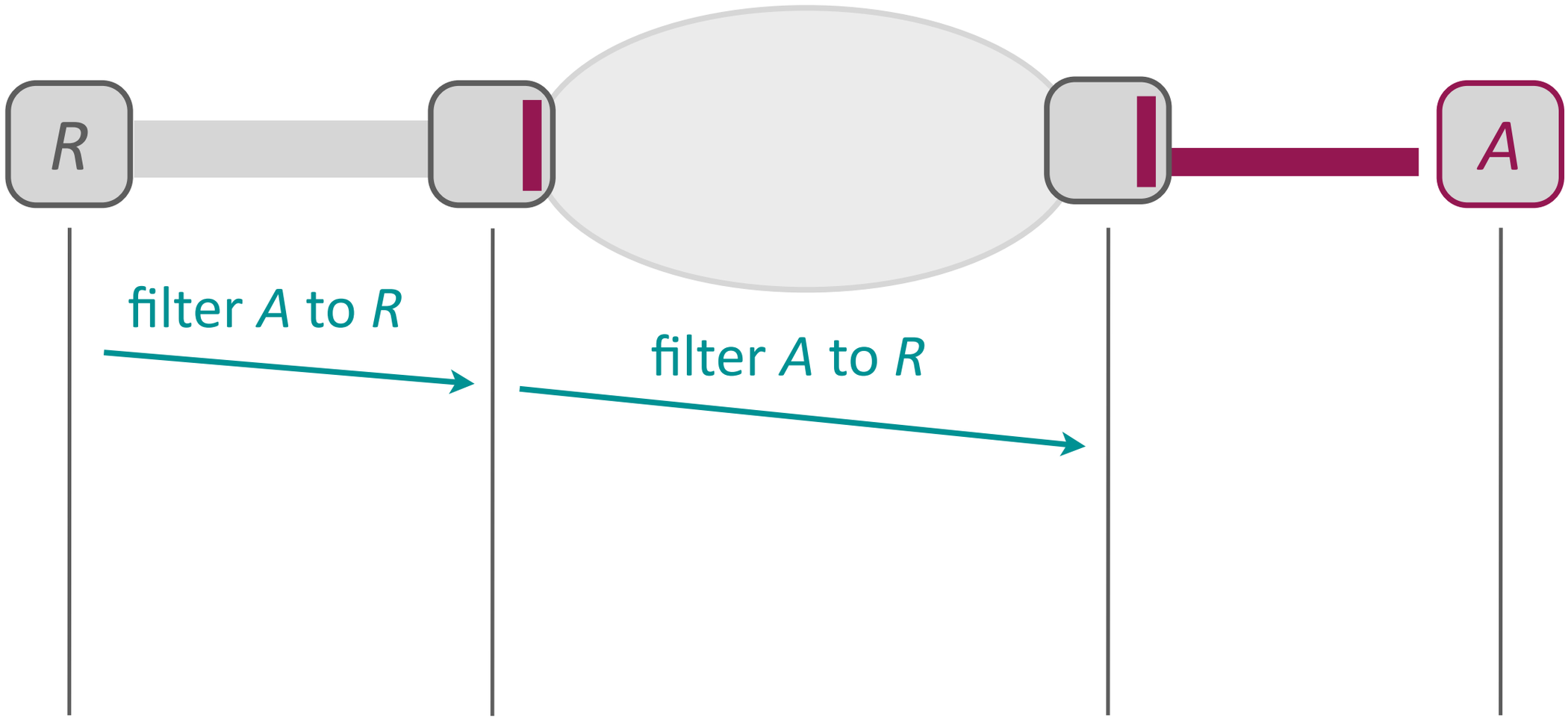
Filter propagation



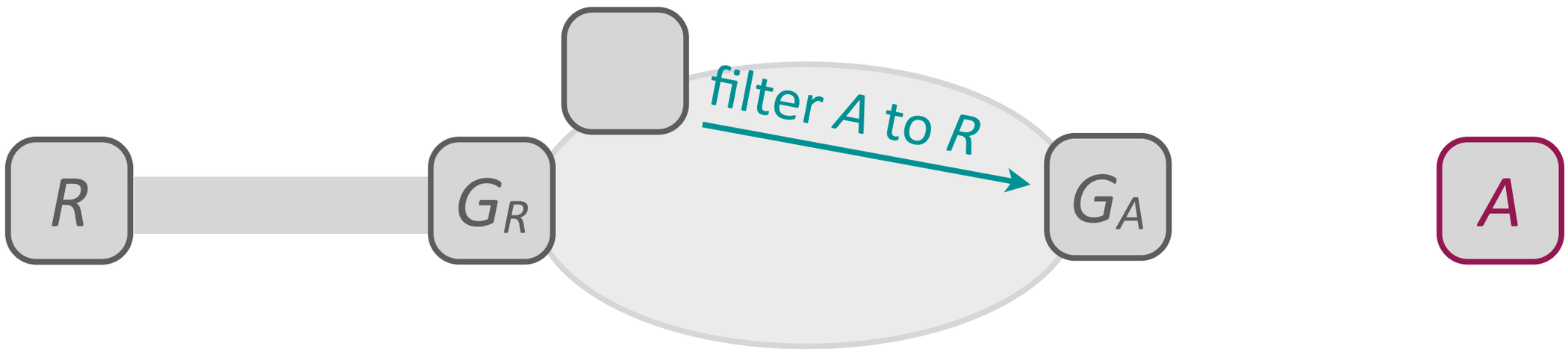
Filter propagation



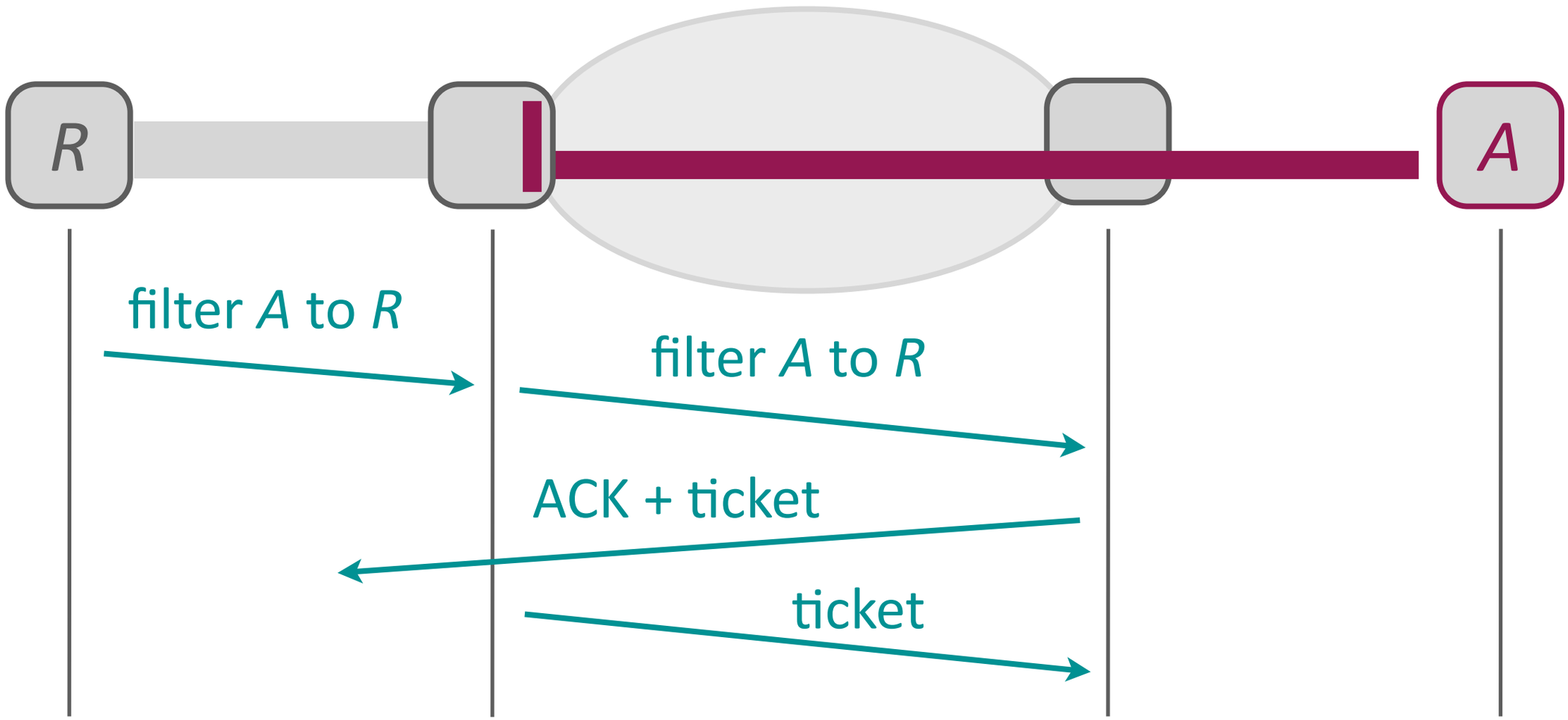
Filter propagation



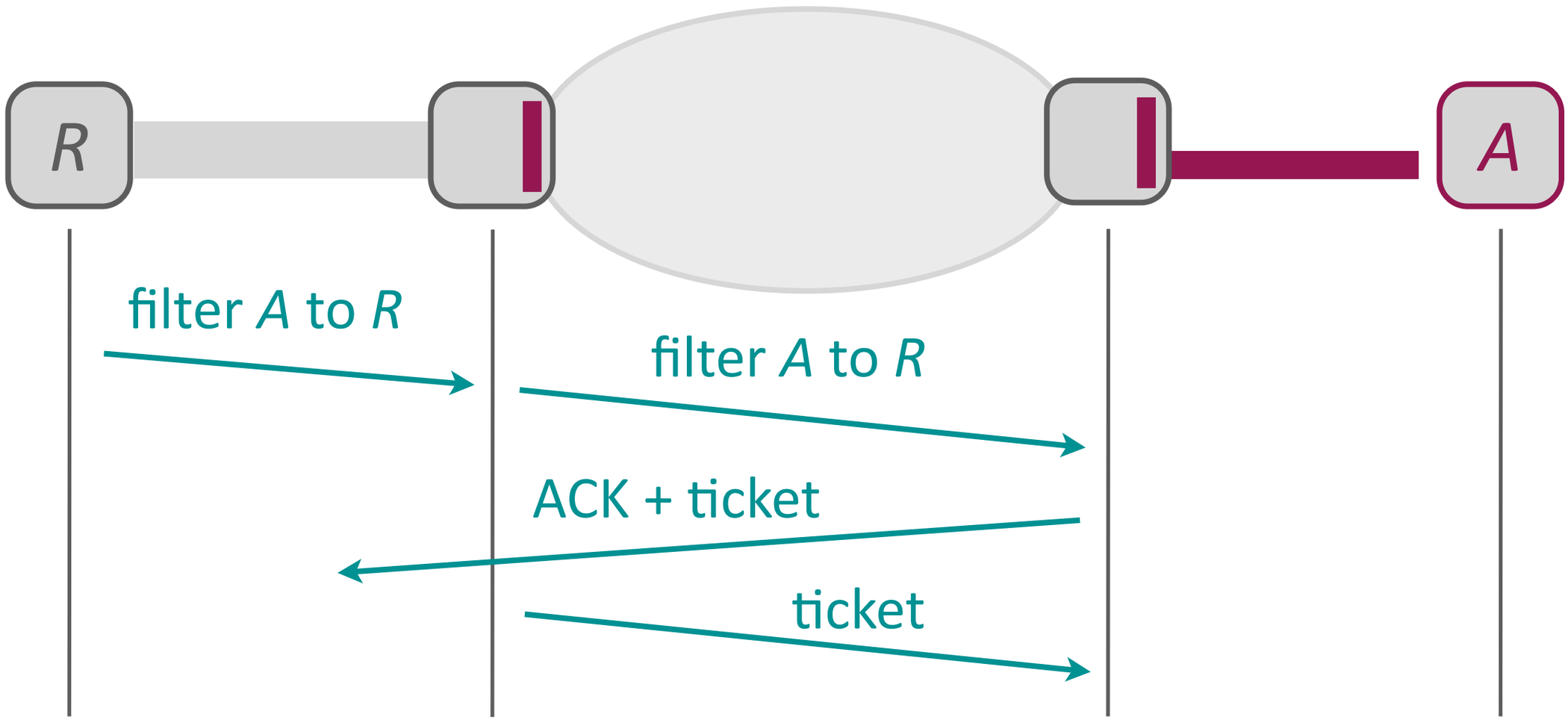
Malicious filtering requests?



Filter propagation continued

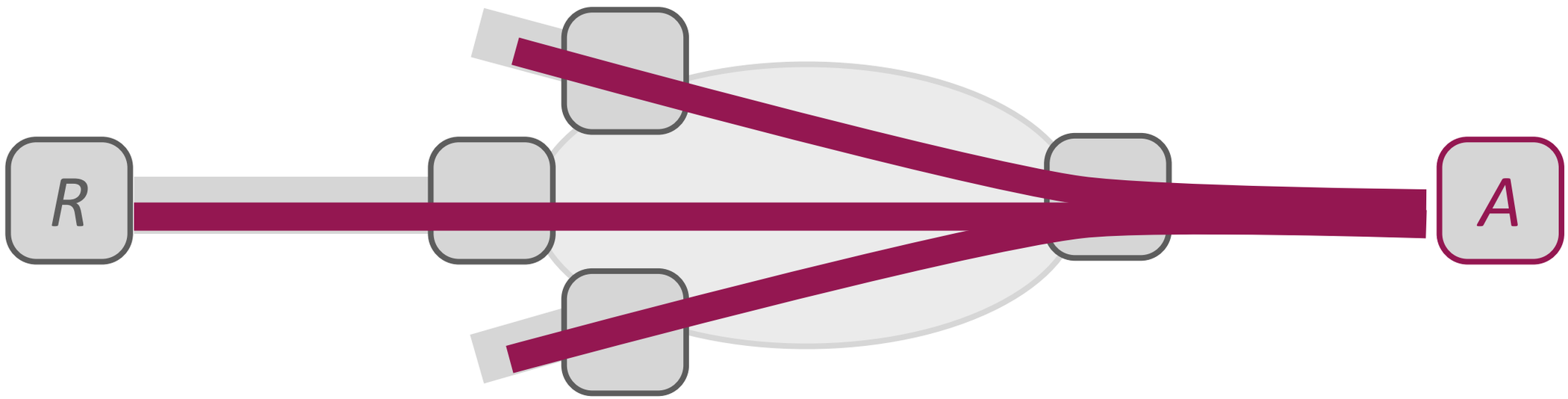


Filter propagation continued

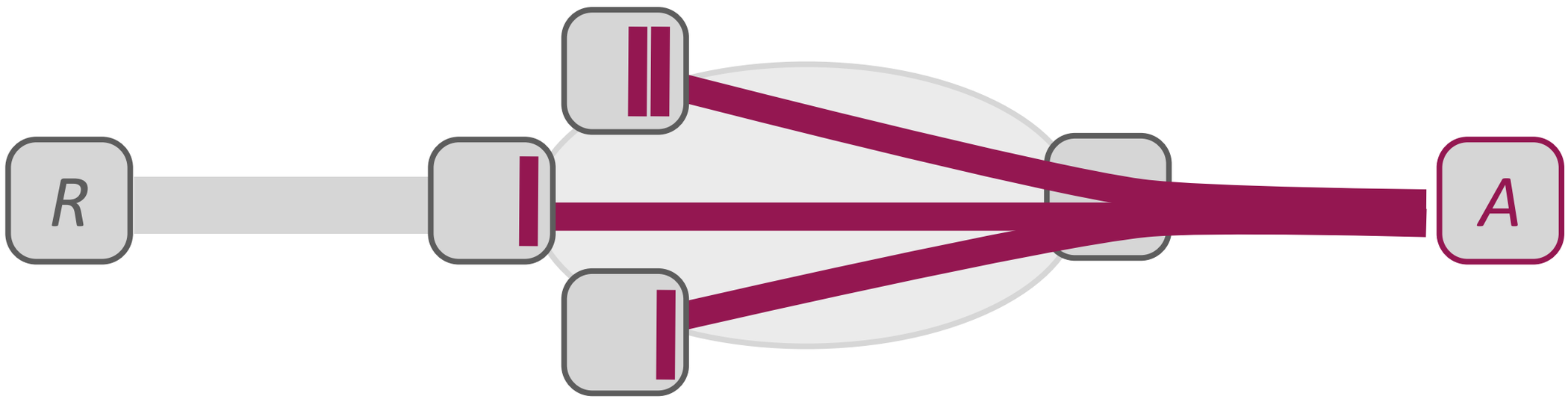


G_R proves it is on the path by 3-way handshake

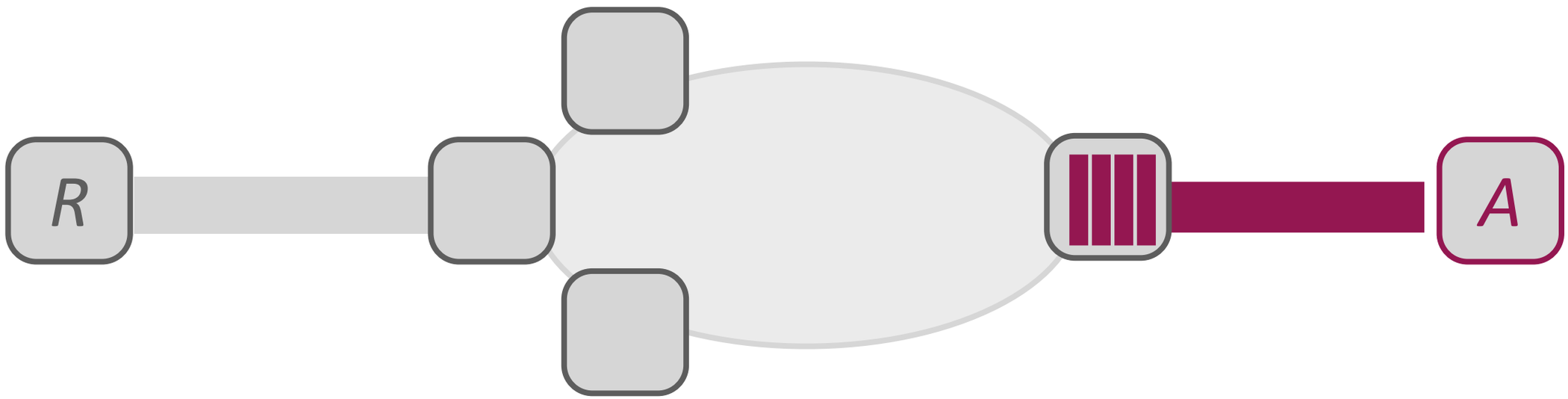
Busy attackers?



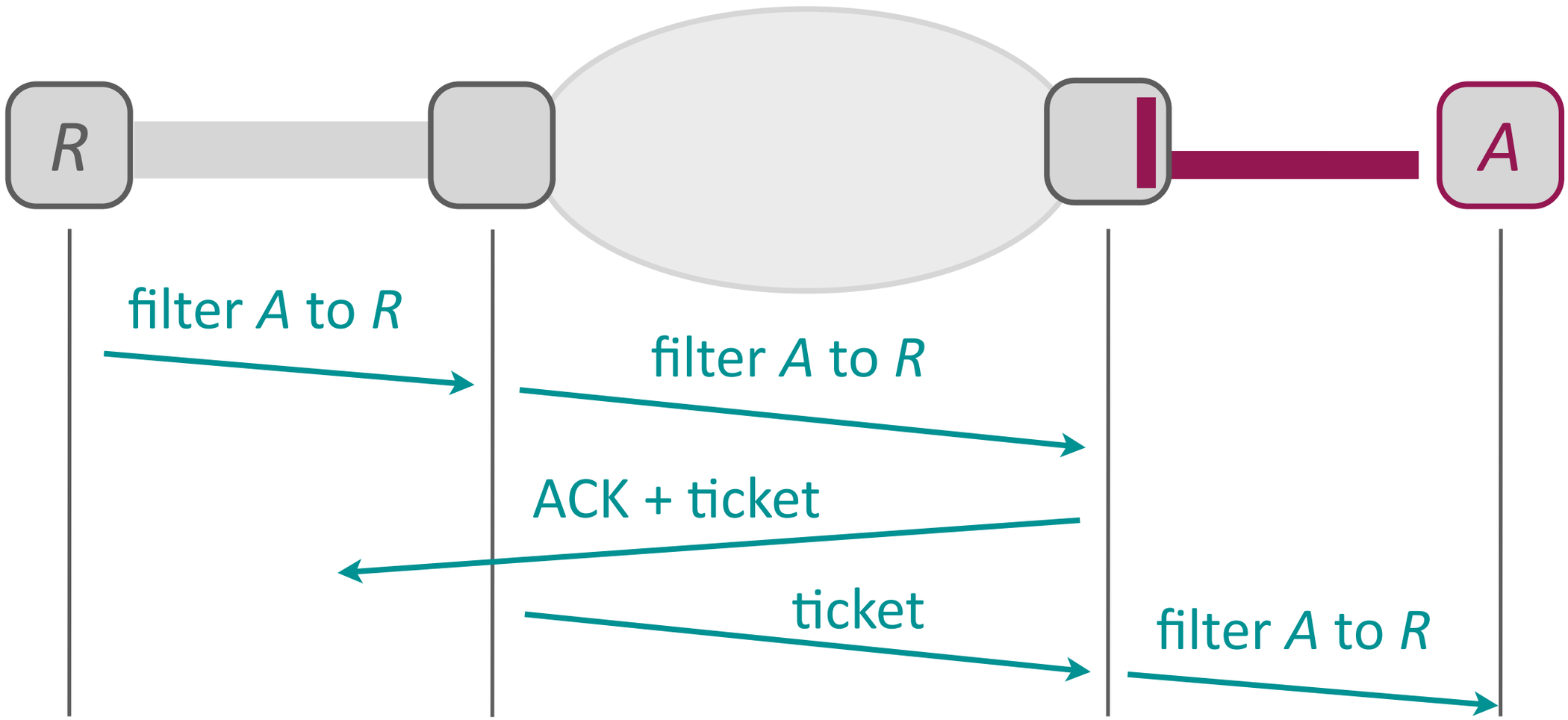
Busy attackers?



Busy attackers?

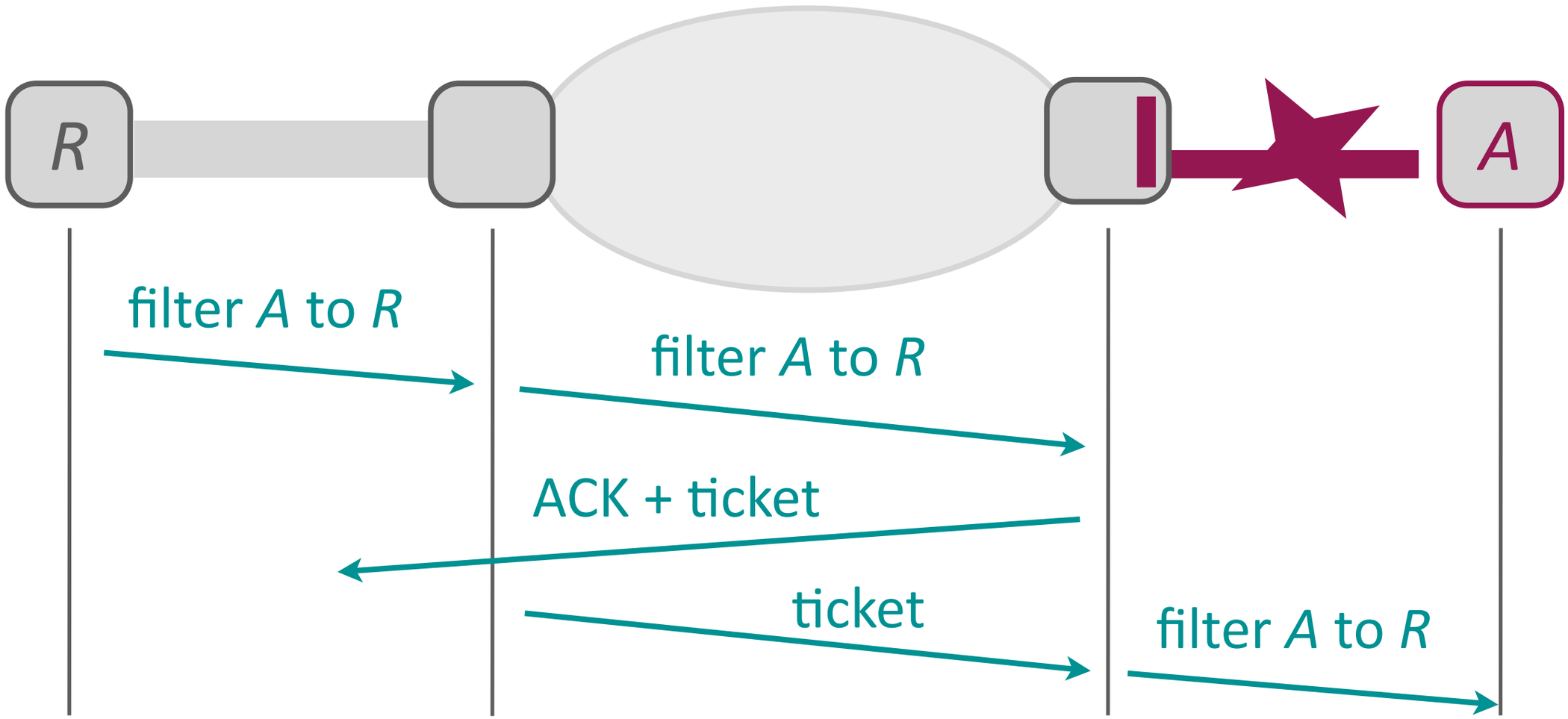


Filter propagation continued



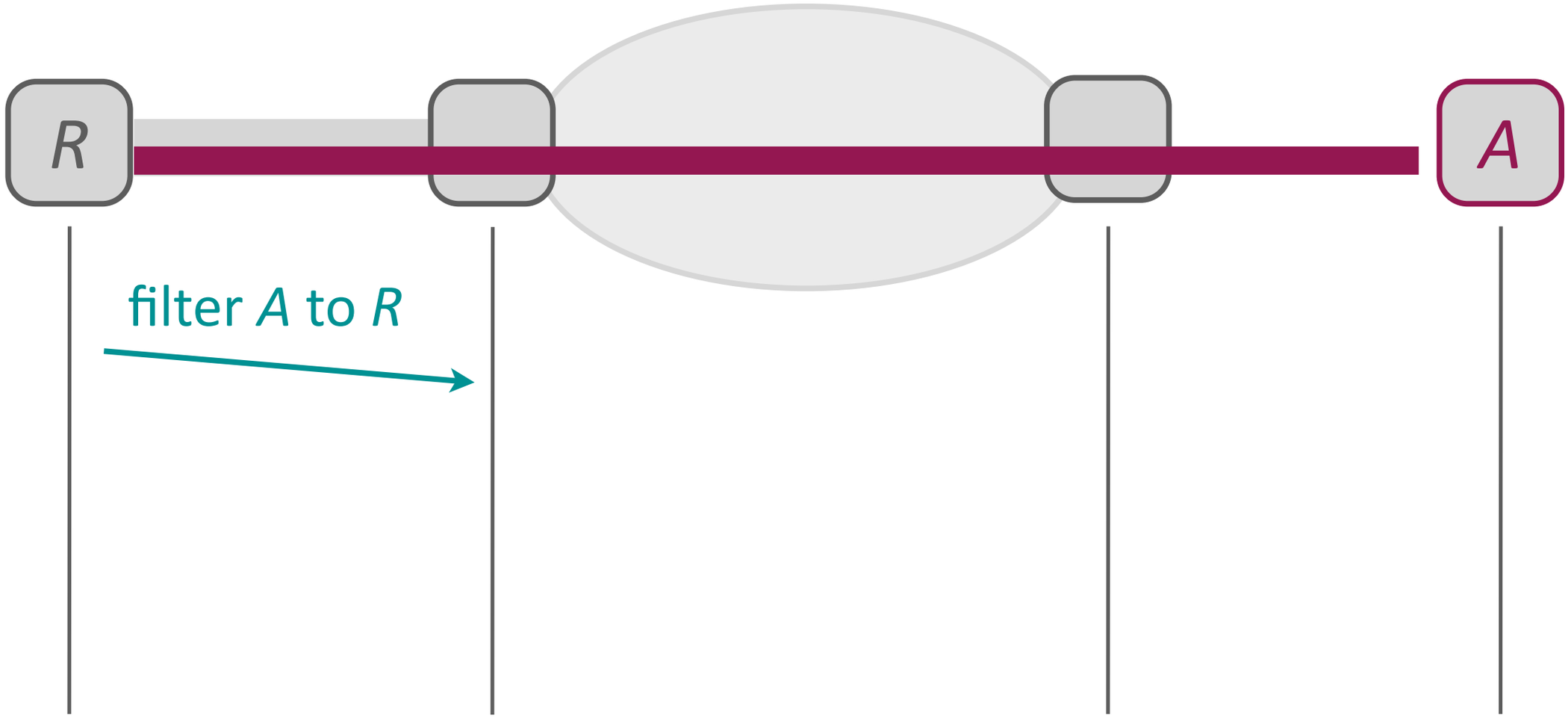
Keep in-network filters temporarily

Filter propagation continued

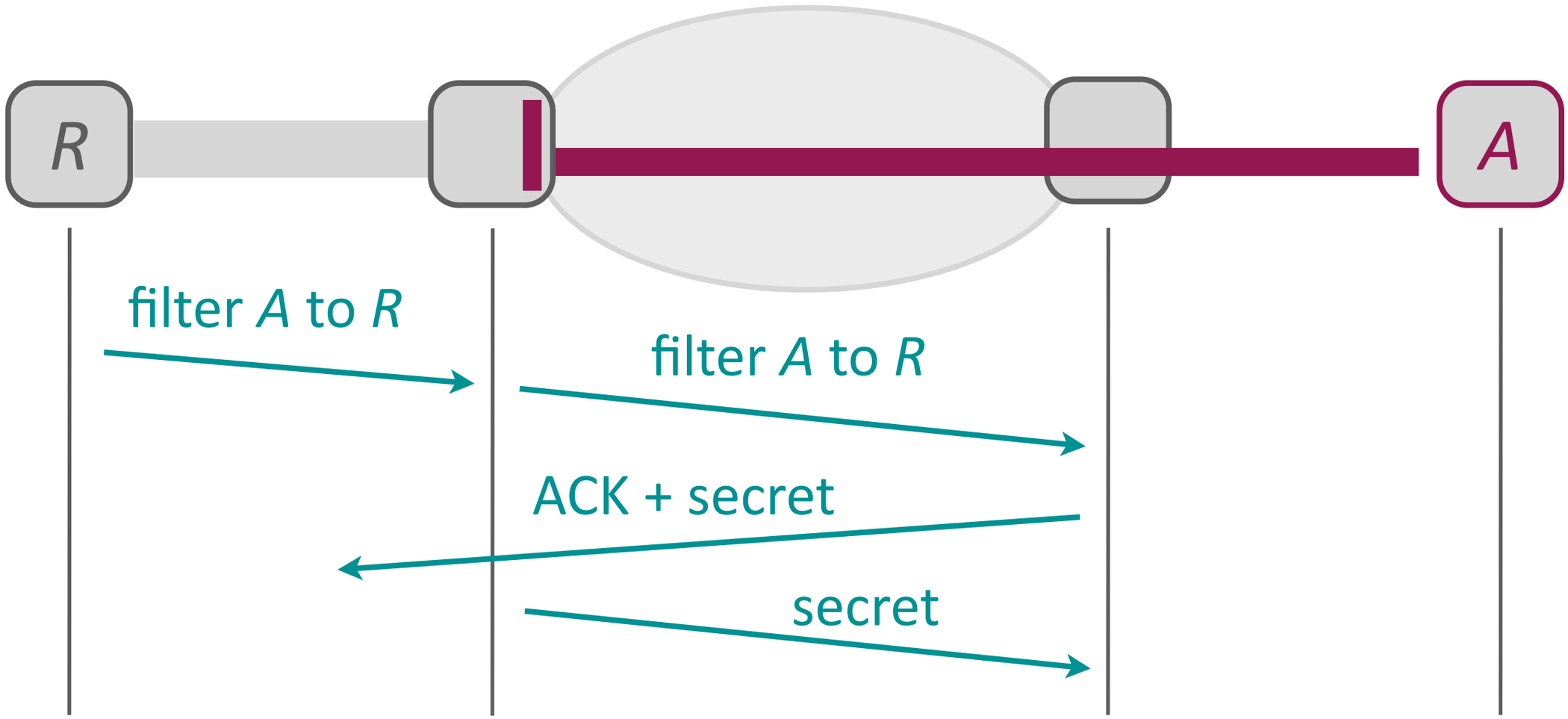


Disconnection = cheap filtering

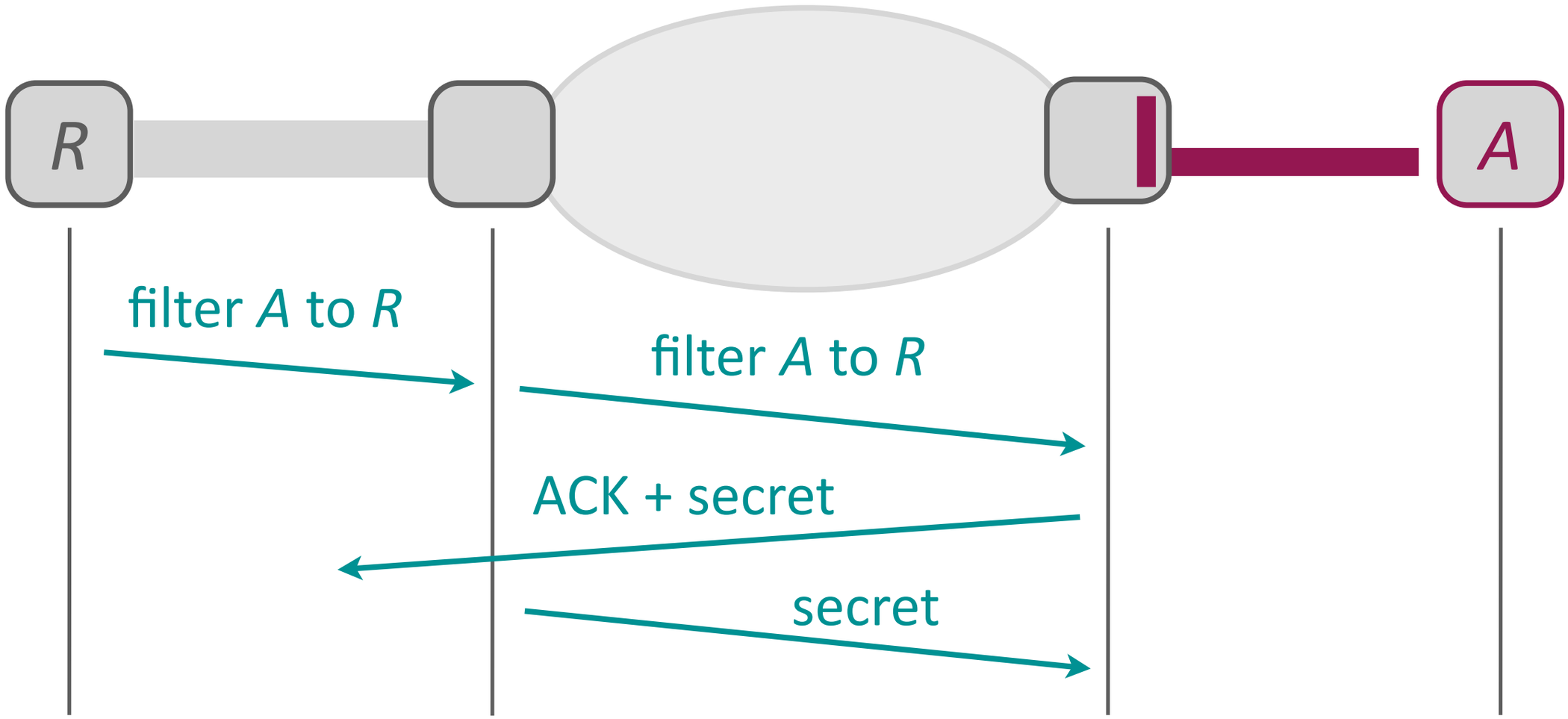
Repeat offenders?



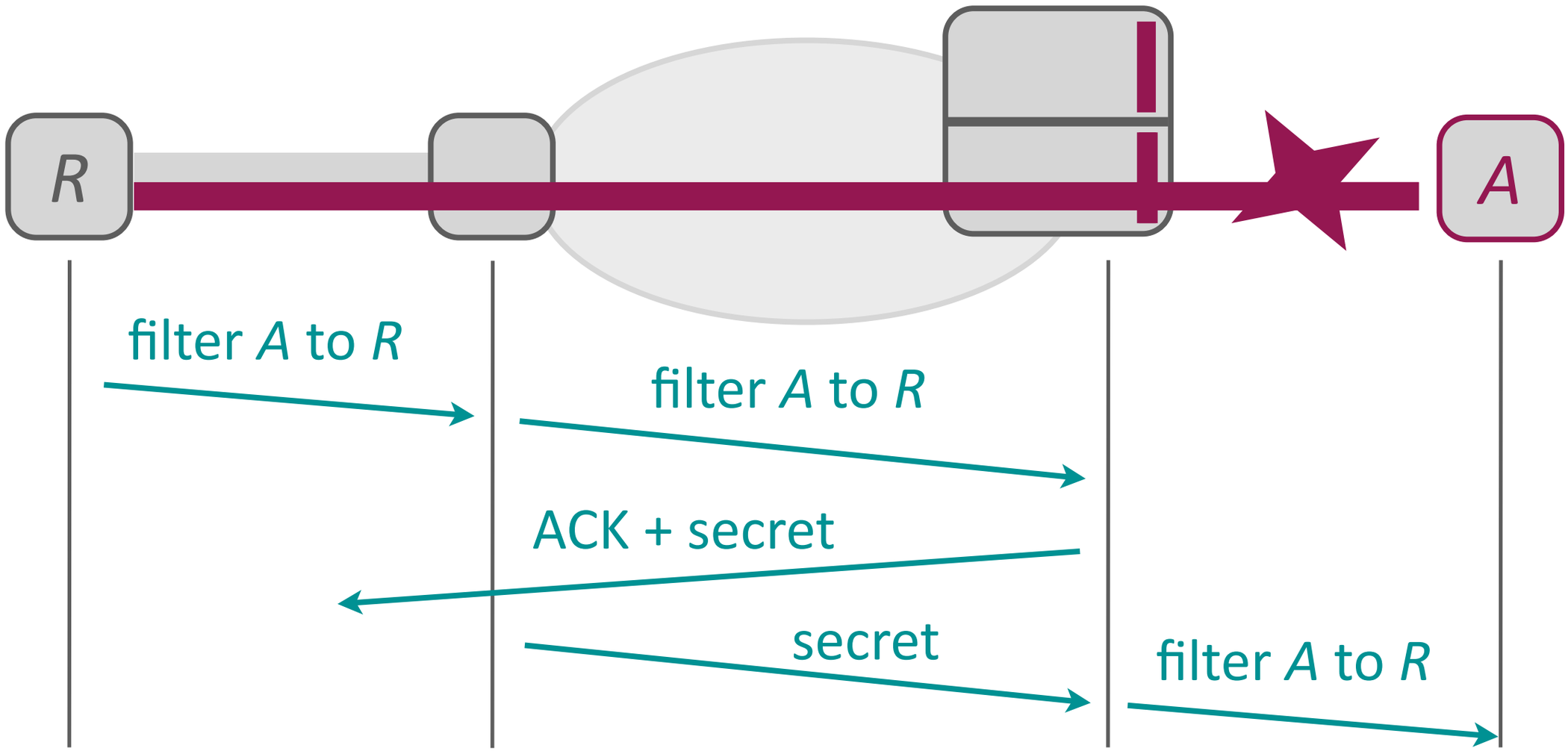
Repeat offenders?



Repeat offenders?

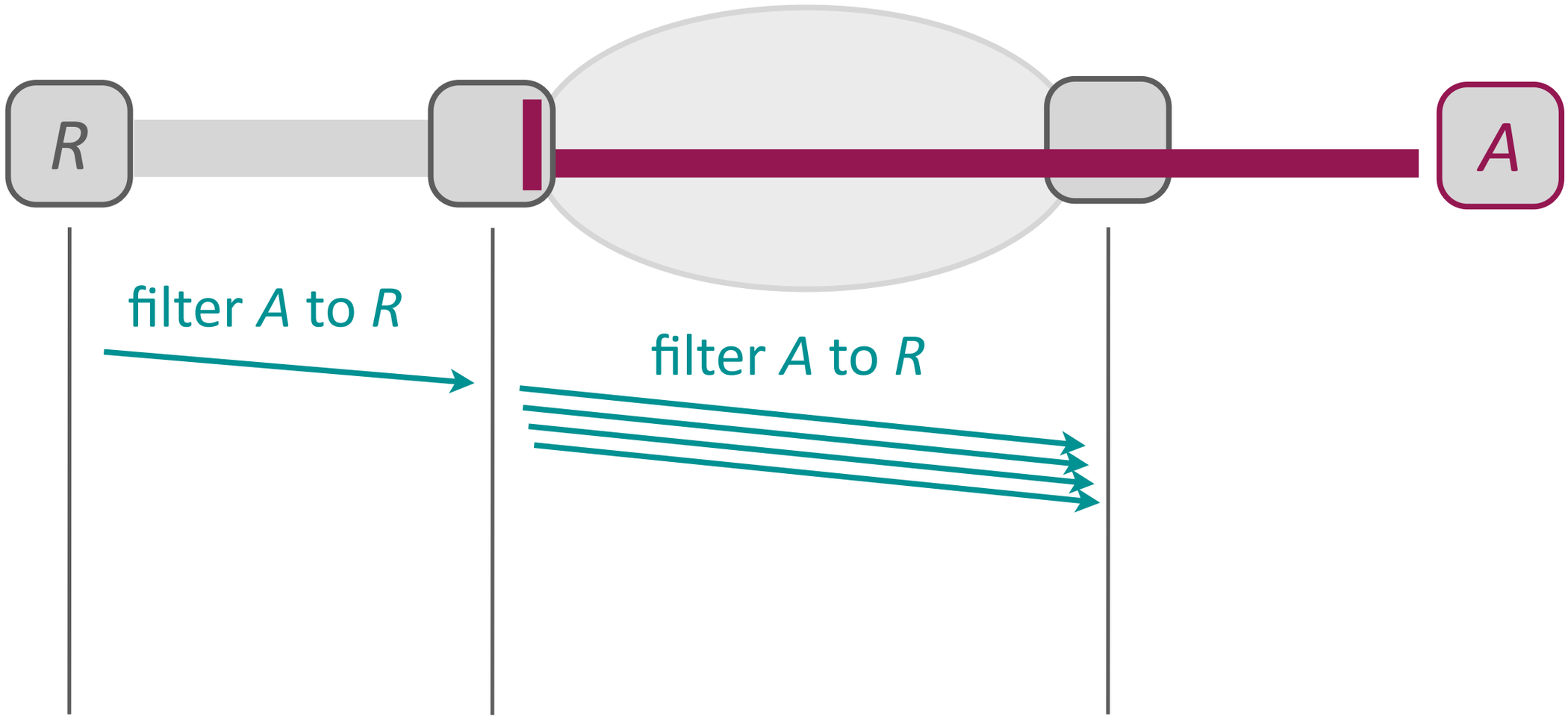


Repeat offenders?

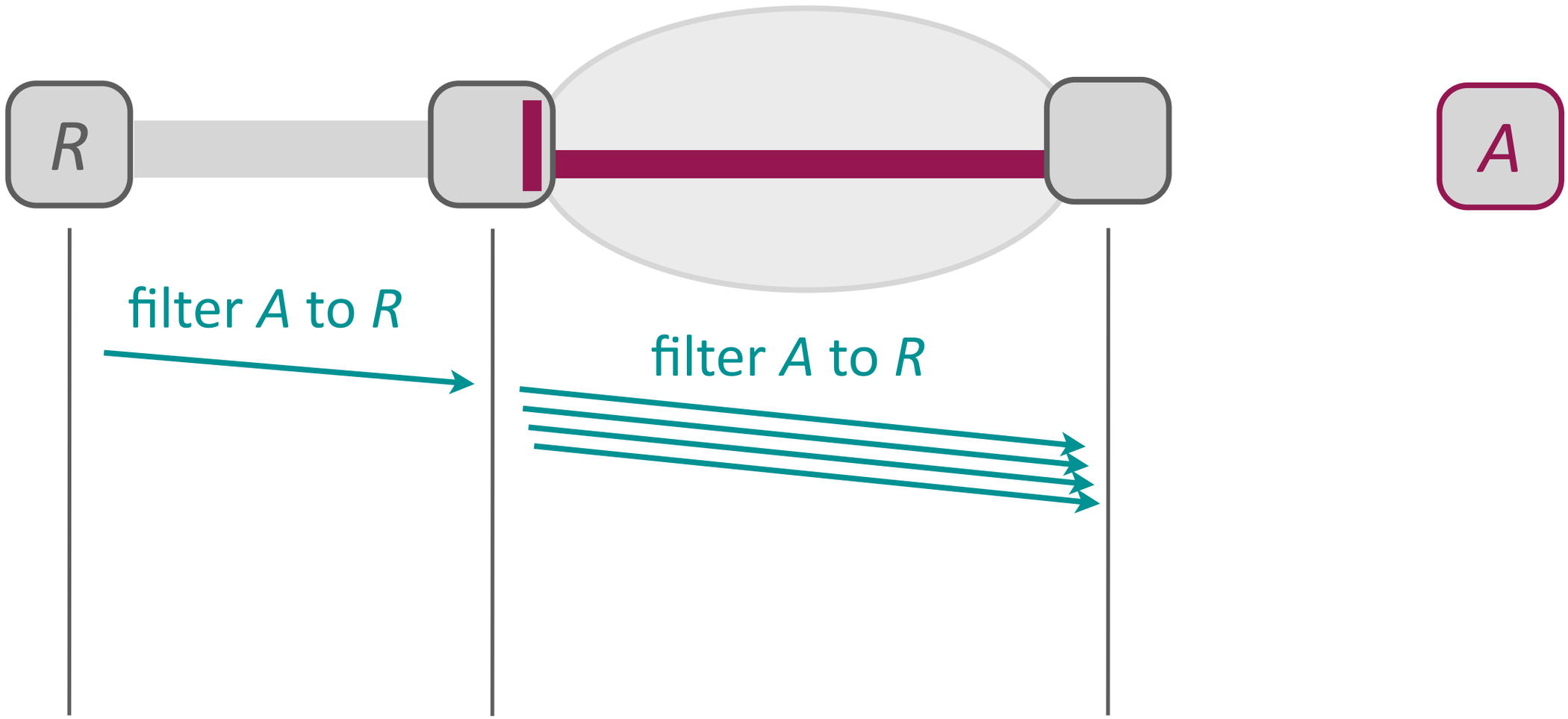


Keep filtering state in the control plane

Non-cooperative networks?

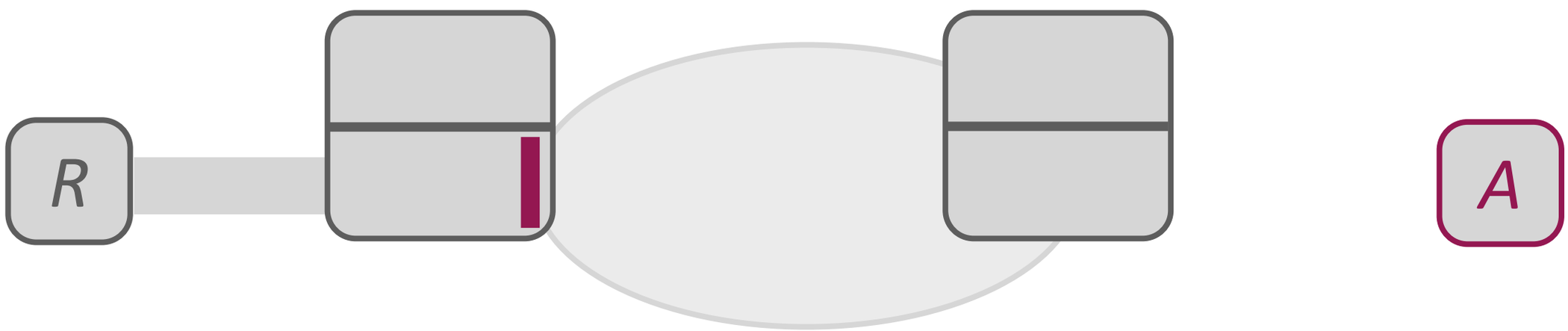


Non-cooperative networks?



... get disconnected from R

State



State: {attacker, receiver} pairs

Where: control plane of attacker' gateway

Managed: filter-propagation protocol

Ticket-based authorization

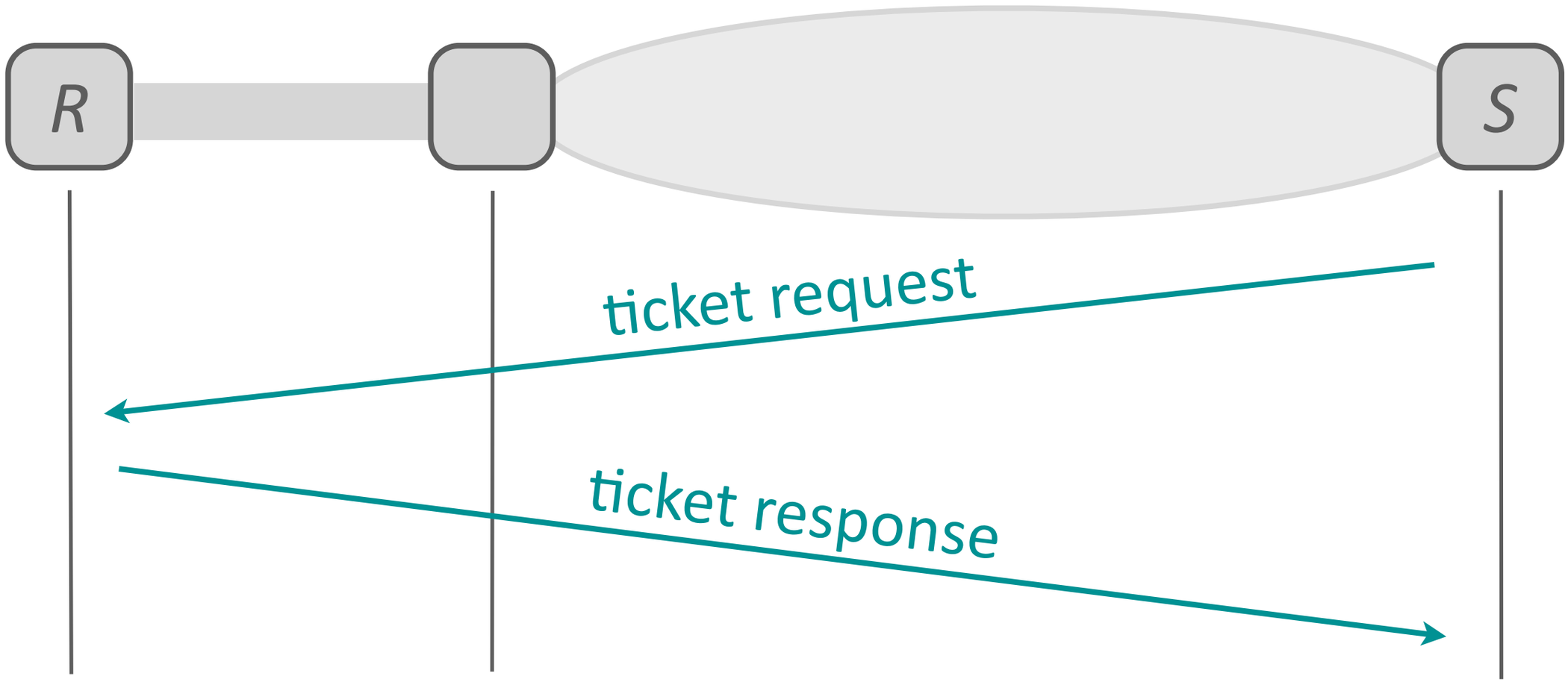
Give tickets to well behaved senders

Verify tickets inside the network

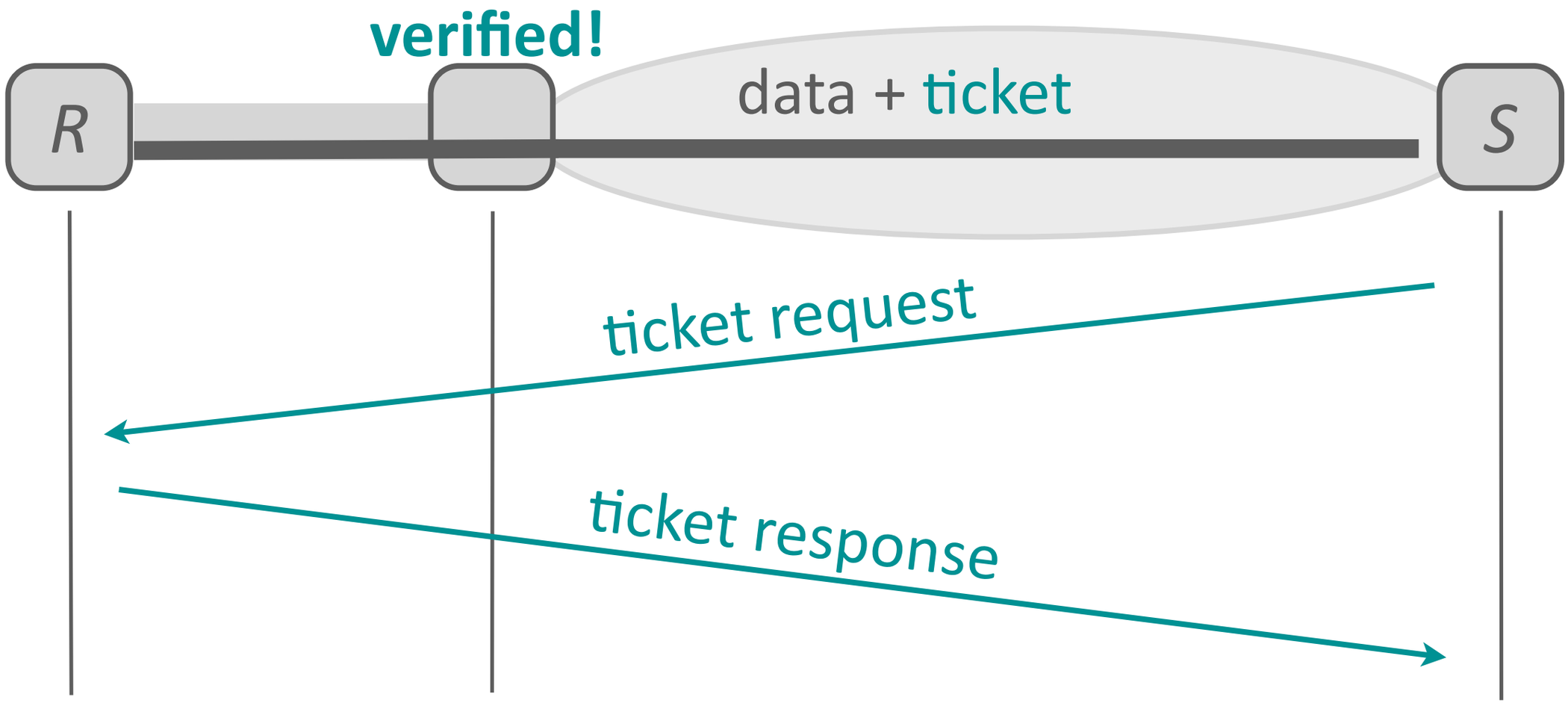
Need to sketch distributing and verifying

(SIFF, Yaar, Perrig and Song, 2004)

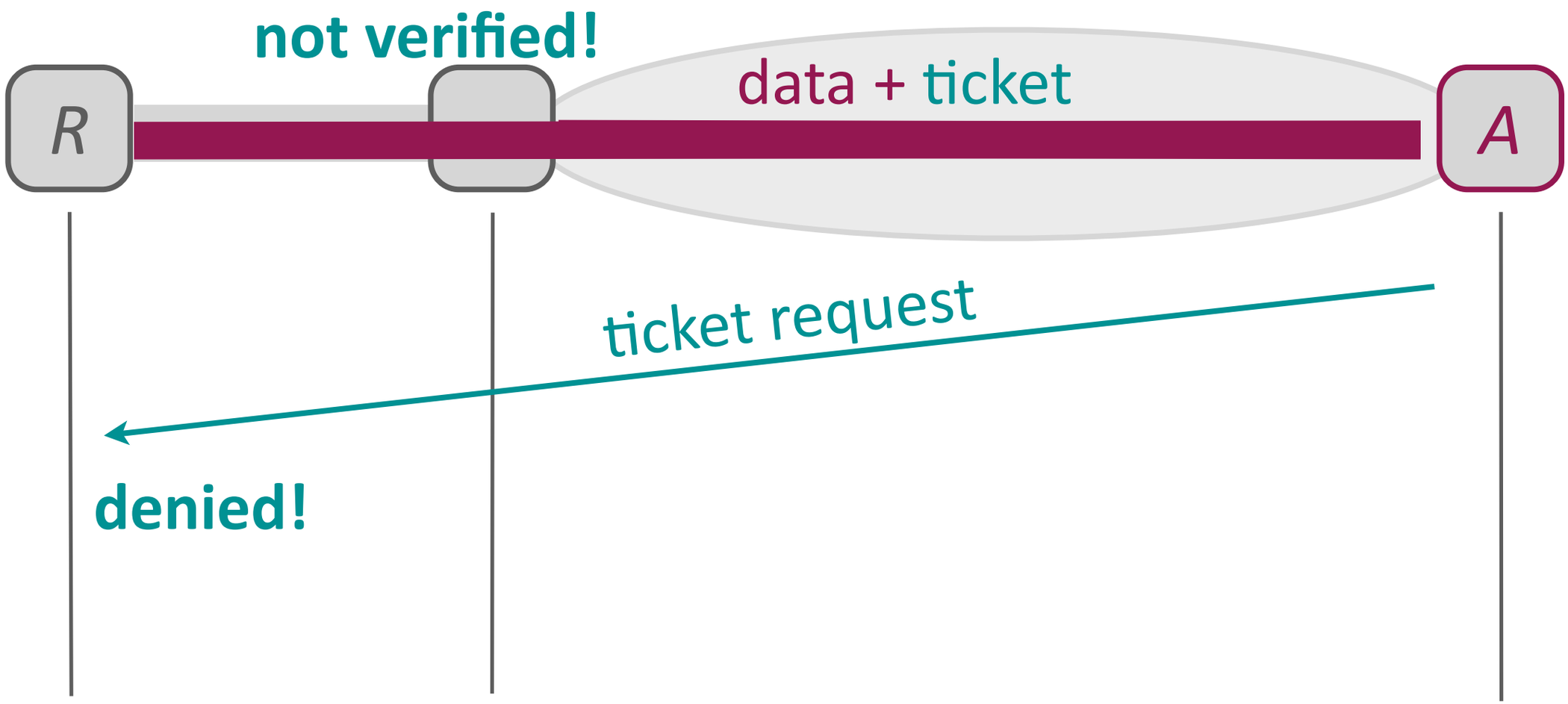
Ticket distribution



Ticket verification



Ticket verification

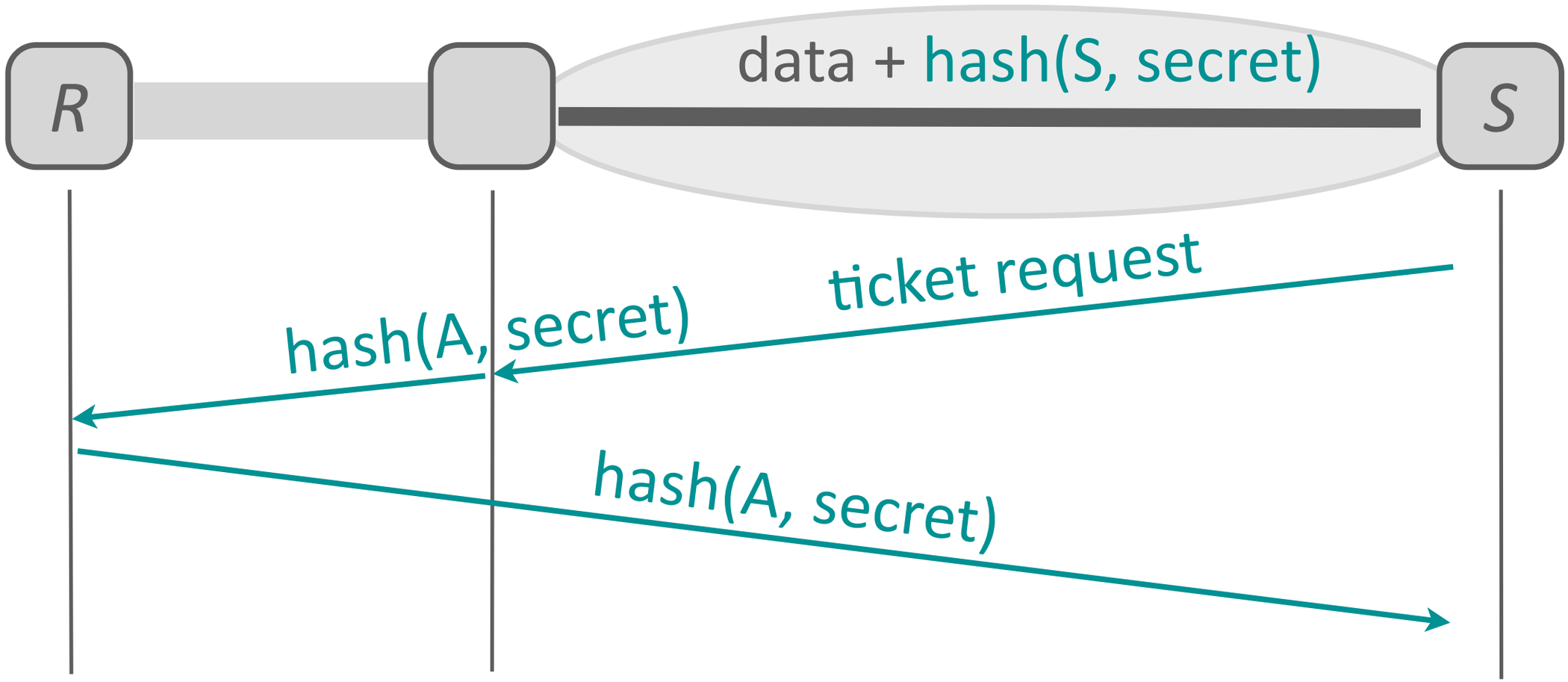


Stateless filtering

State: -

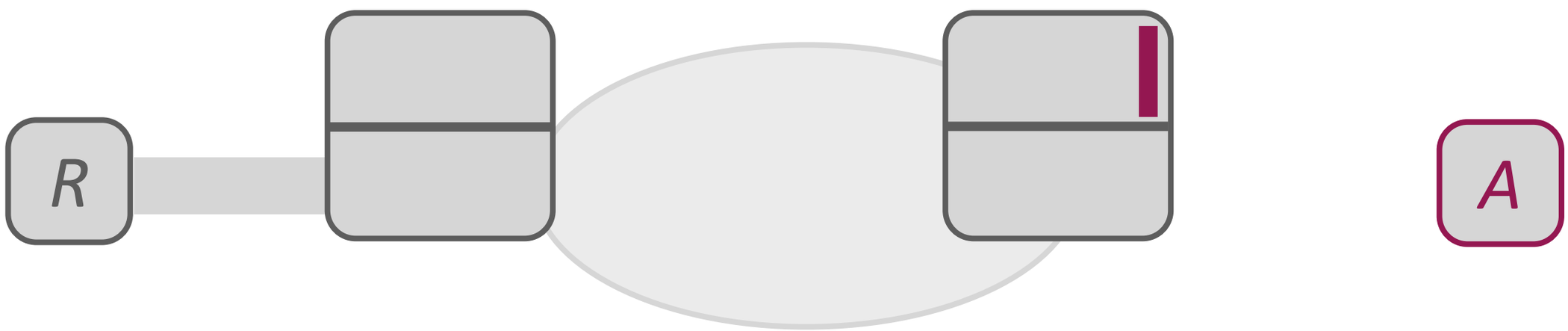
Code: *if (not verify(ticket))*
block packet;

Ticket construction



Remove filtering state from the network

State

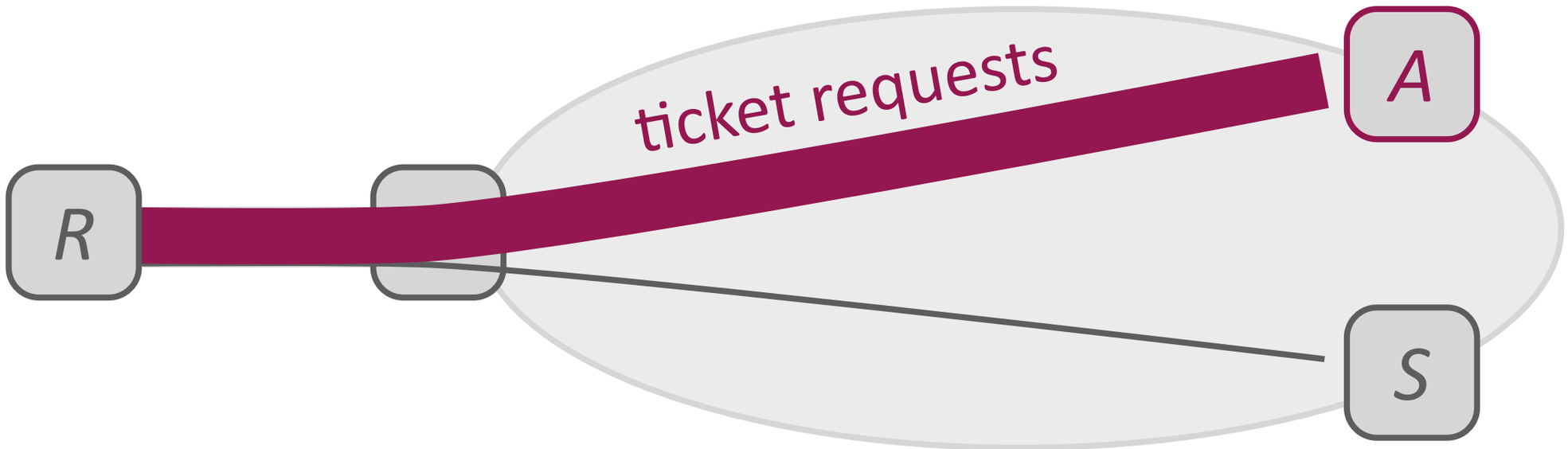


State: {sender, receiver} pairs

Where: senders

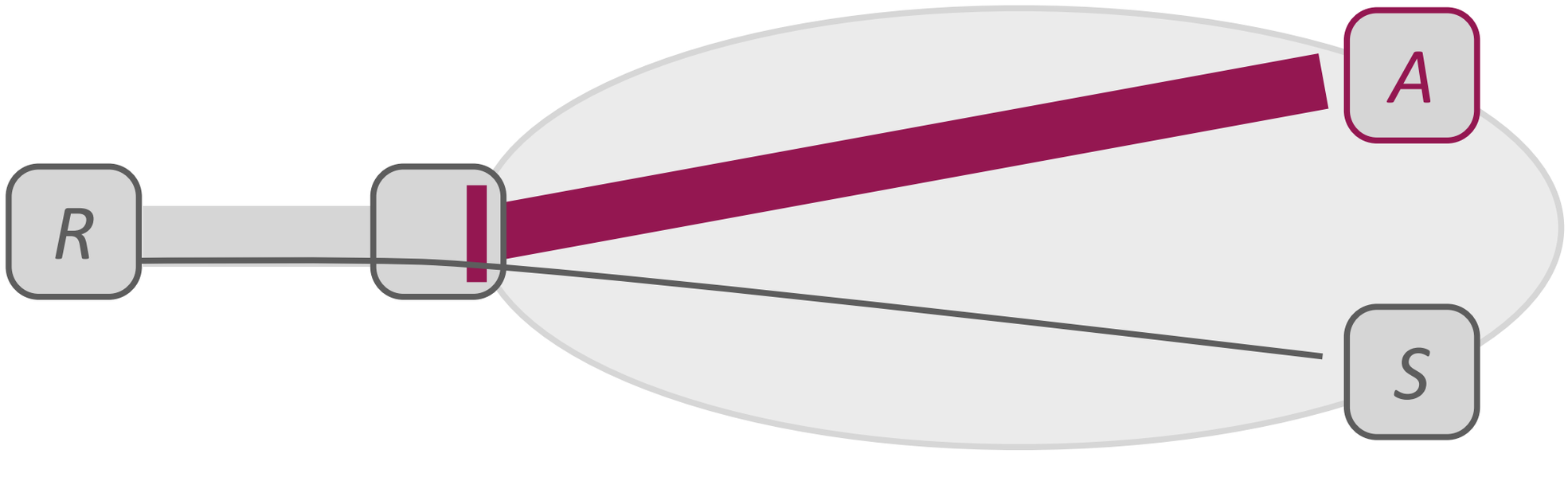
Managed: ticket-distribution protocol

Denial of ticket



Target: tail circuit + ticket distribution

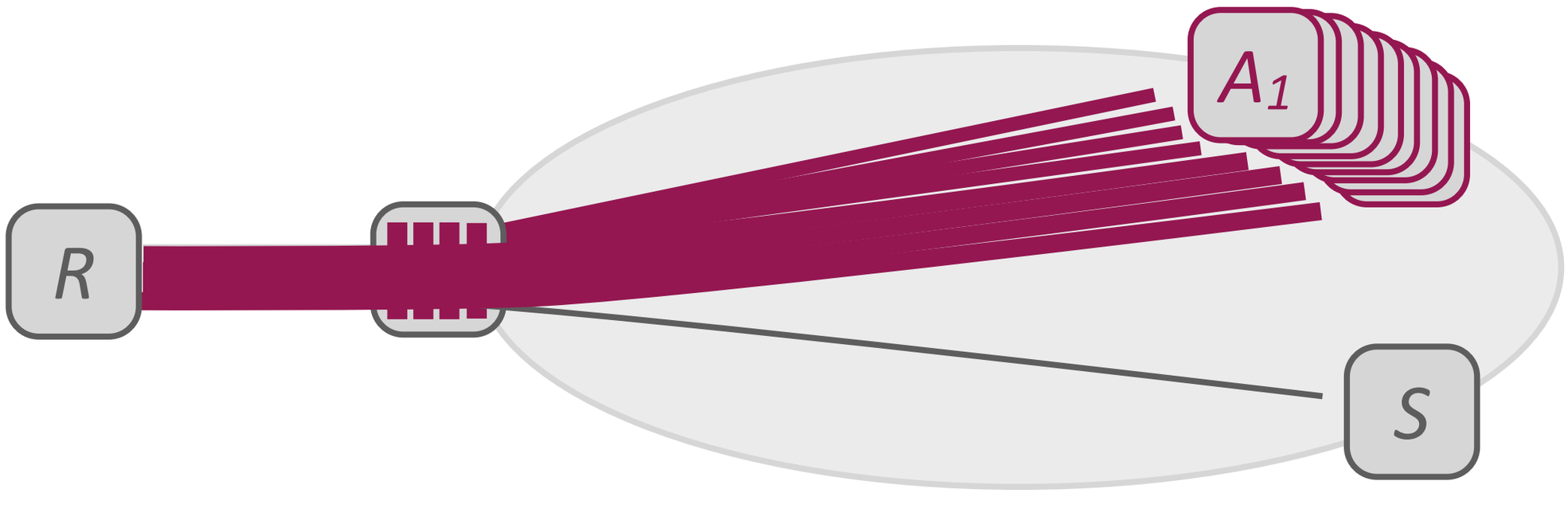
Tickets + network filtering



Block attackers in the network

(TVA, Yang, Wetherall and Anderson, 2005)

Distributed denial of ticket

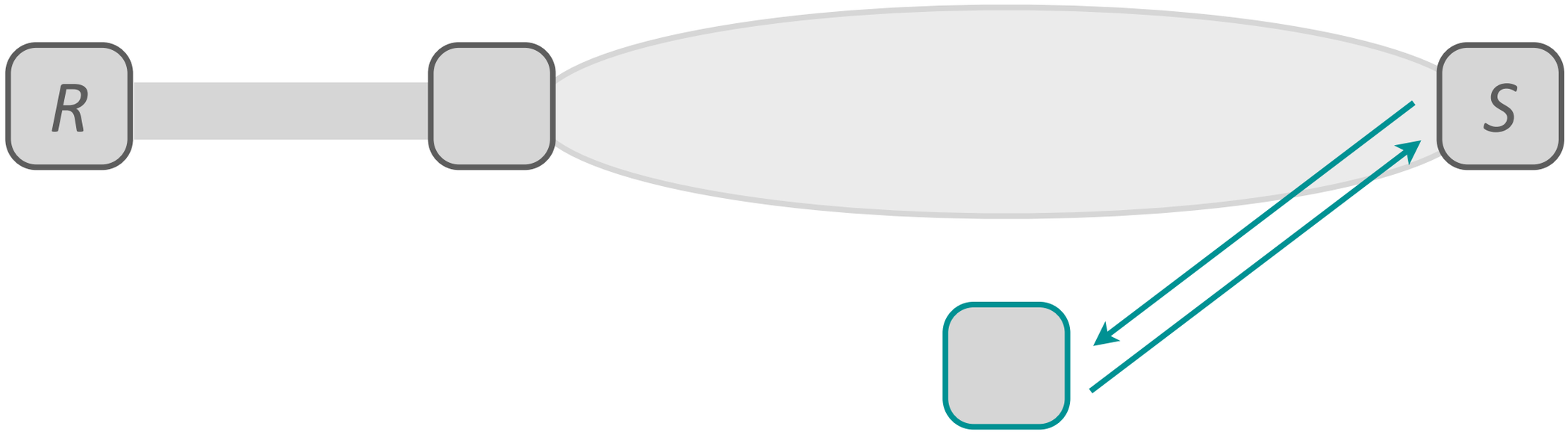


Target: filtering resources + tail circuit + ticket distribution

Tickets + distributed filtering

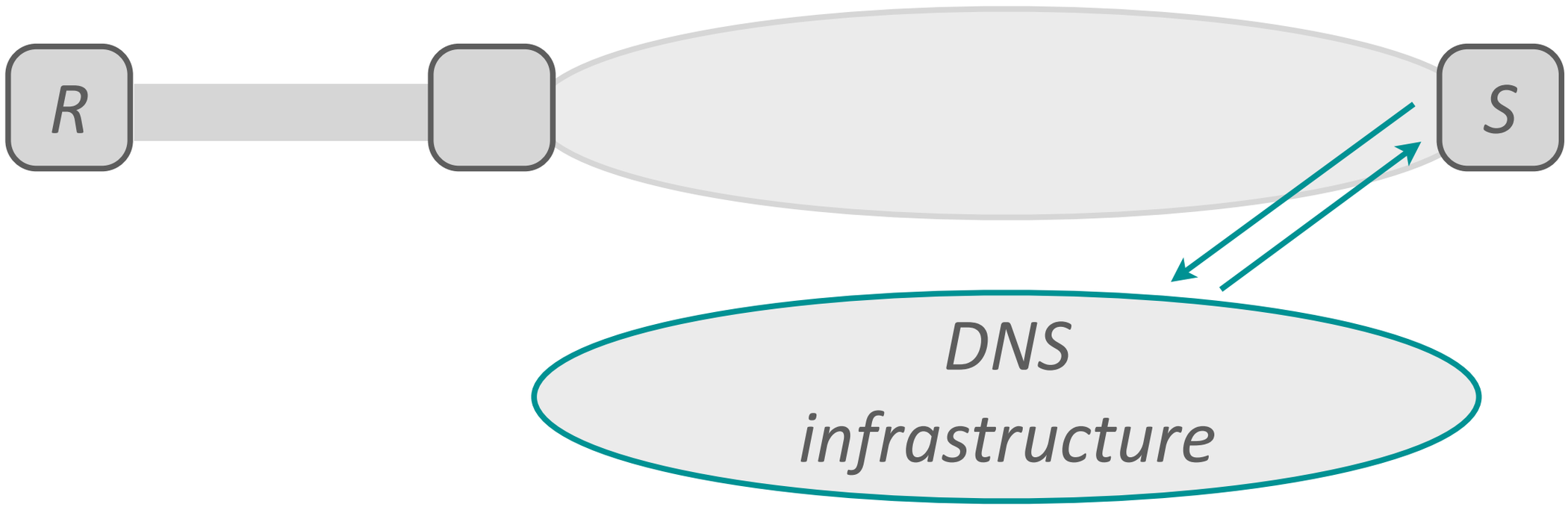
Need a filter-propagation protocol

Outsource ticket distribution

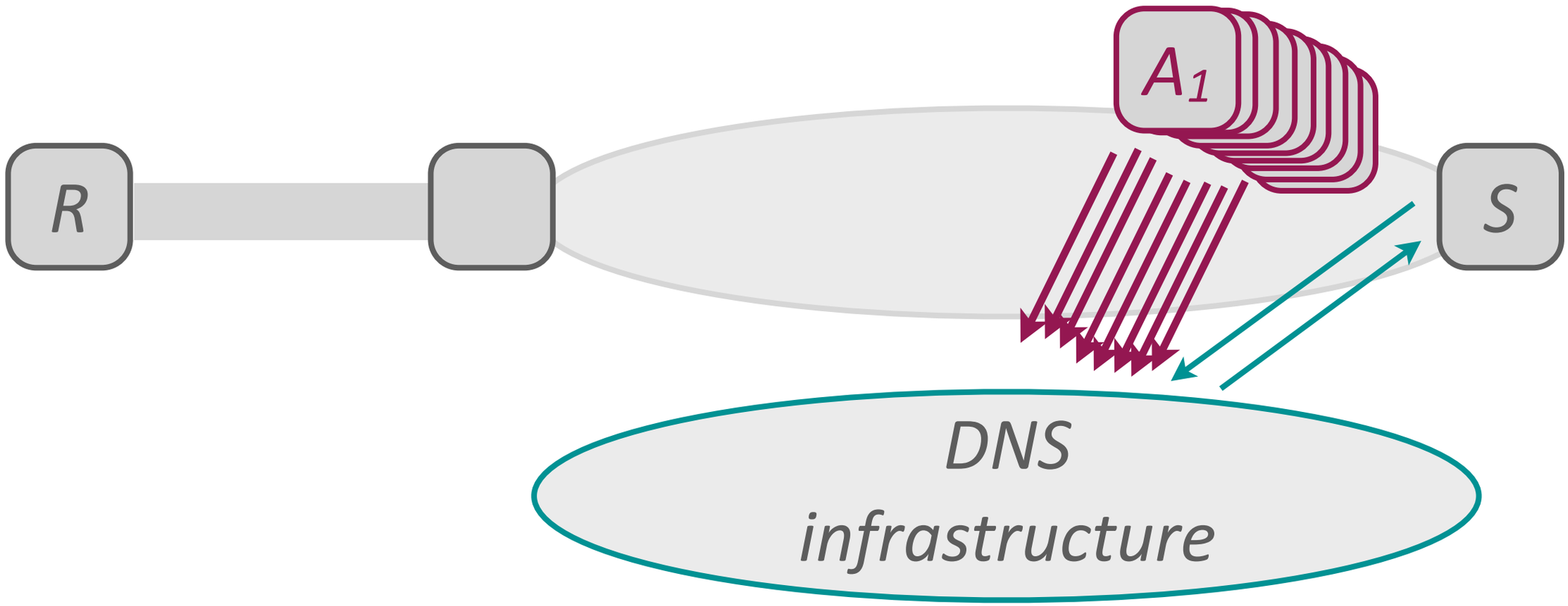


(Portcullis, Parno and Perrig, 2007)

Outsource ticket distribution

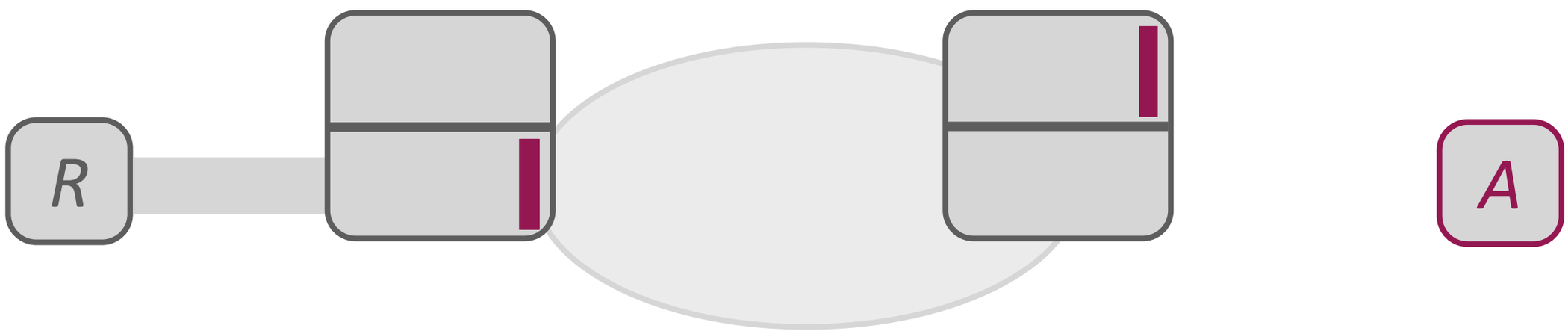


Outsource ticket distribution



Target: the DNS infrastructure

State



State: {sender/attacker, receiver} pairs

Where: senders + network

Managed: ticket distribution + filtering propagation

Fair-share the Internet

Fixed number of connections per sender

Reduces filtering state

Changes the nature of the Internet

Conclusion

Identify minimum state

Move it where it is easy to manage

Beware of hidden state!