NETWORK CAPABILITIES

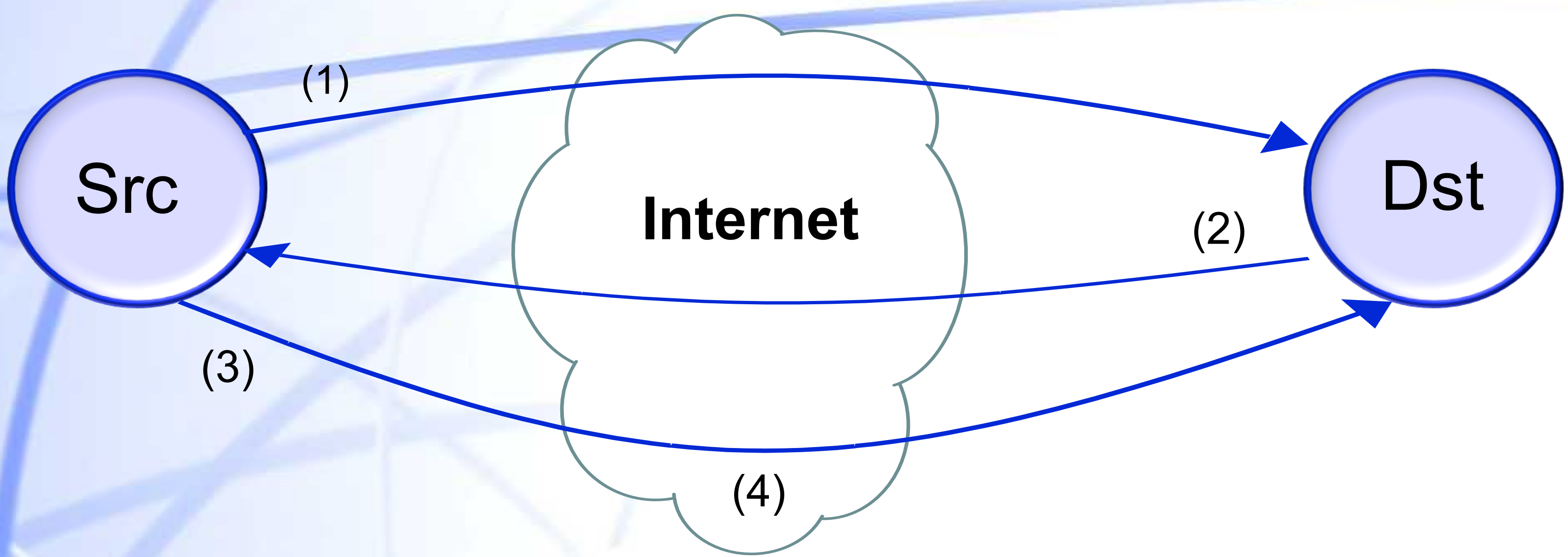
Basic solution

Introduction

"If we could start over, how would we re-design the Internet to be resistant to DoS attacks?"

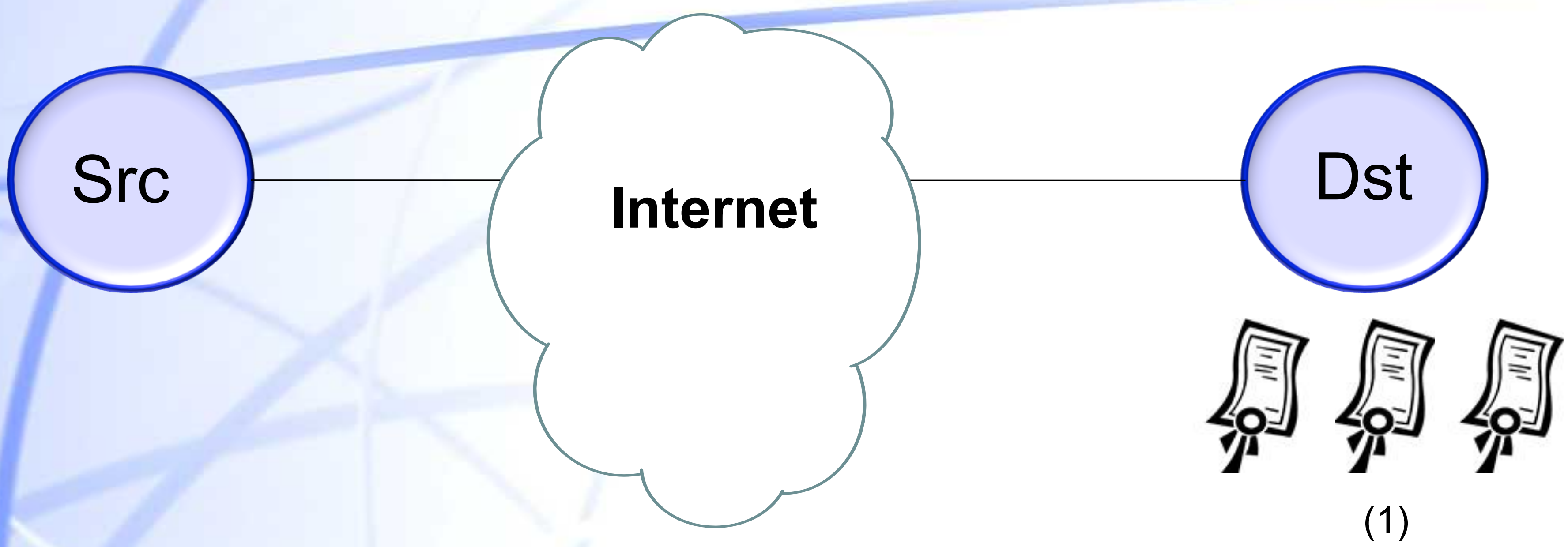
- **Defined in:**
  - **T. Anderson, T. Roscoe, and D. Wetherall. Preventing Internet Denial of Service with Capabilities. In *ACM HotNets-II, 2003.***
- **Objectives:**
  - **Complete**
  - **Open to new applications**
  - **Secure**

Simplified scheme



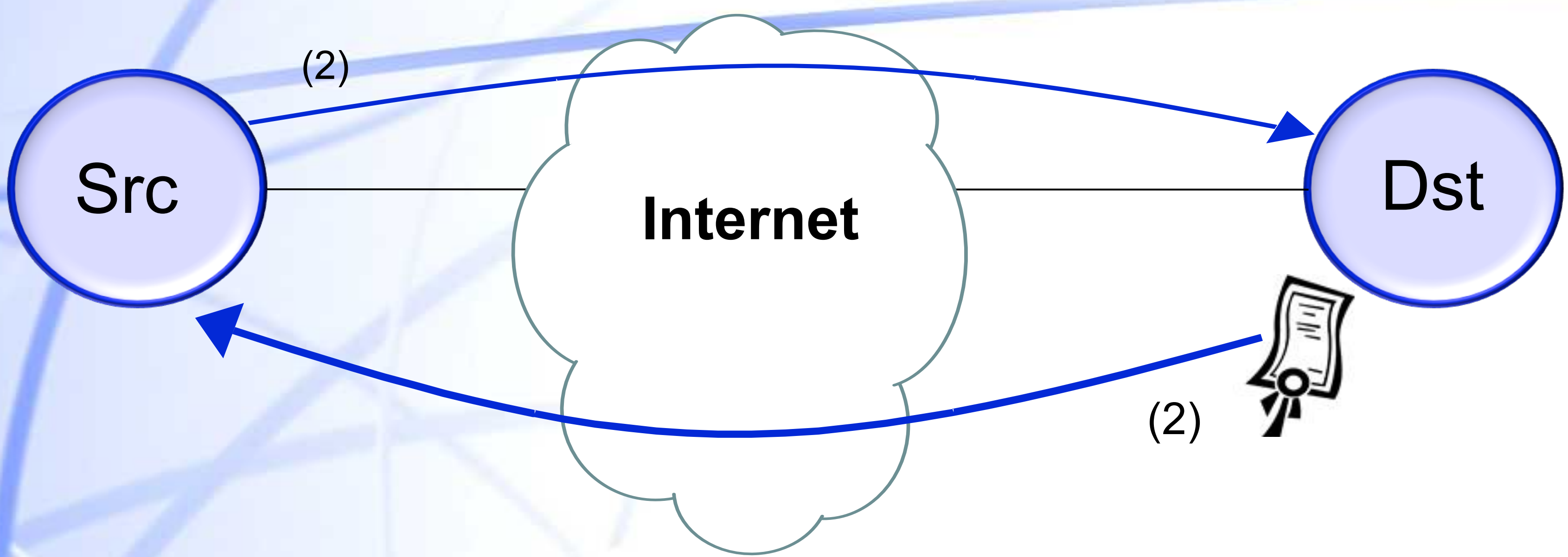
1. **Source requests permission to send**
2. **Destination grants permission**
3. **Source sends authorized traffic**
4. **Network enforces destination decision**

Conceptual scheme



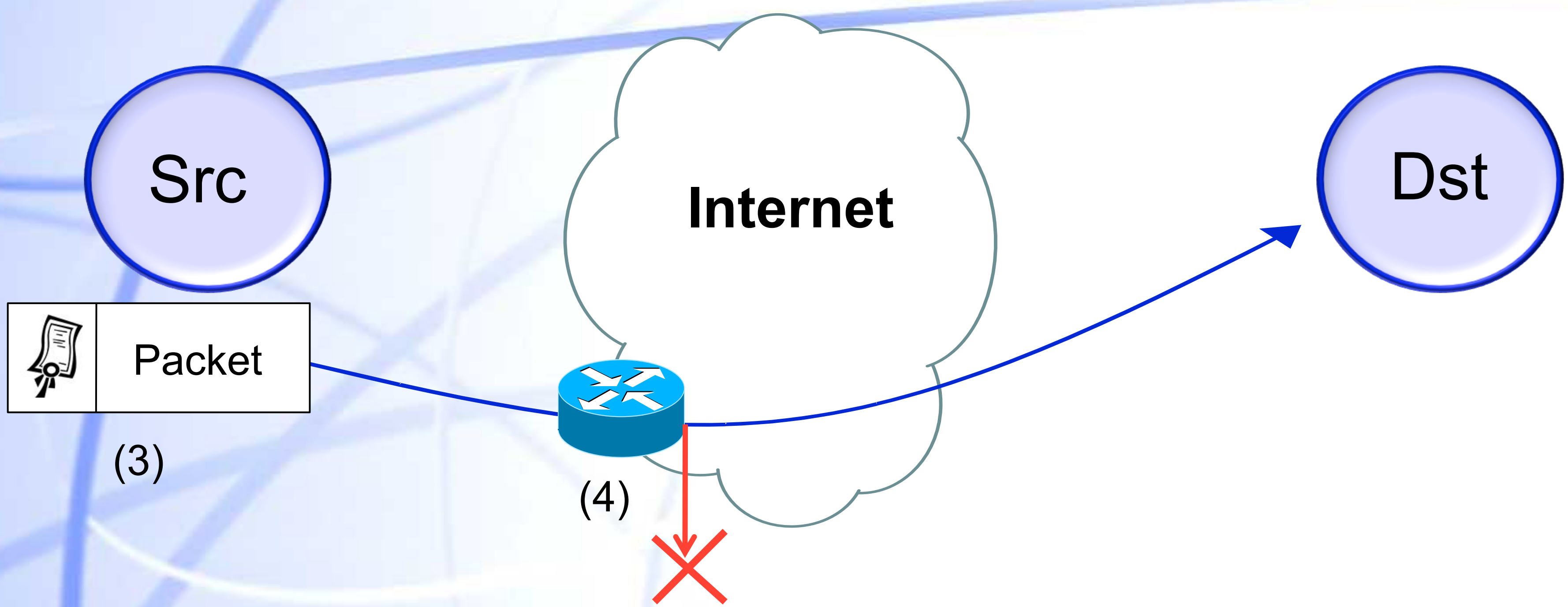
1. **Each destination generates certificates (i.e. tokens representing permission to send):**
   - **Each certificate contains a timestamp**
   - **The certificate is signed with the private key of the destination**

Conceptual scheme



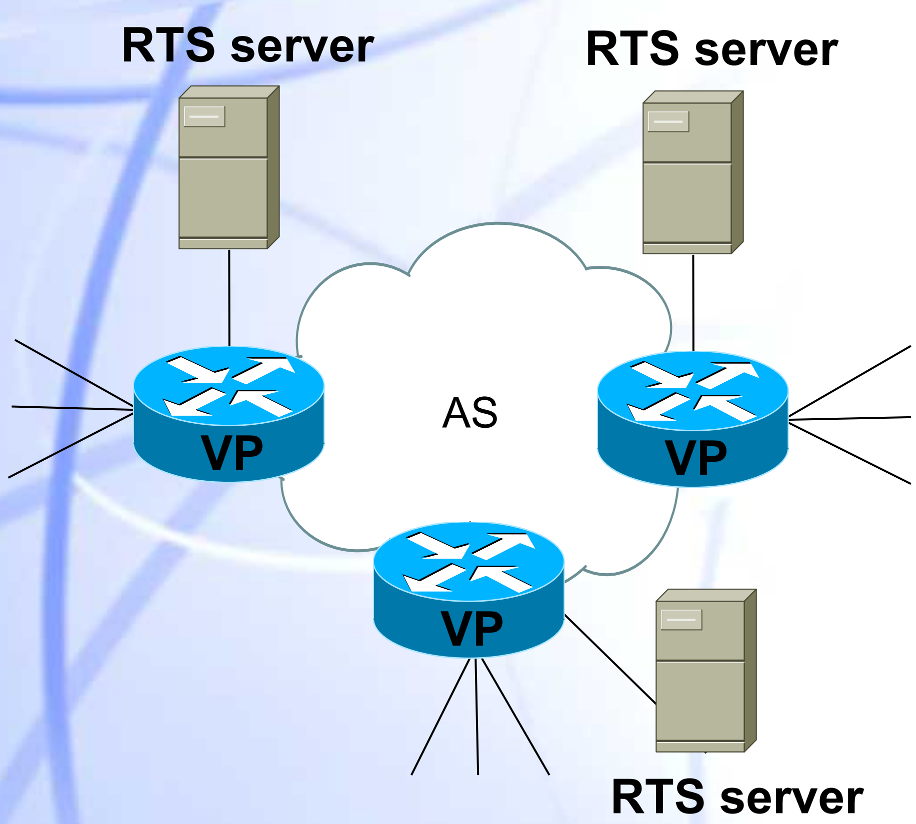
2. **Certificates are requested from the source and granted by the destination, using a protected setup channel**

Conceptual scheme



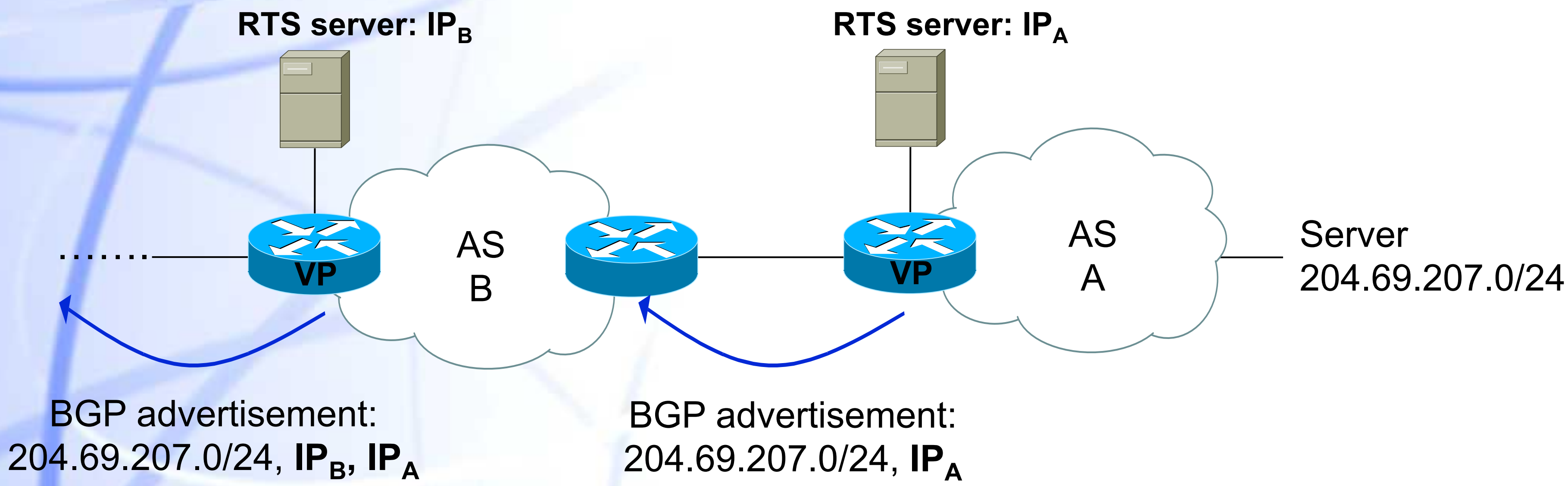
3. **Each packet includes a certificate**
4. **Routers can discard packets not requested by the destination**

Strawman design: architecture



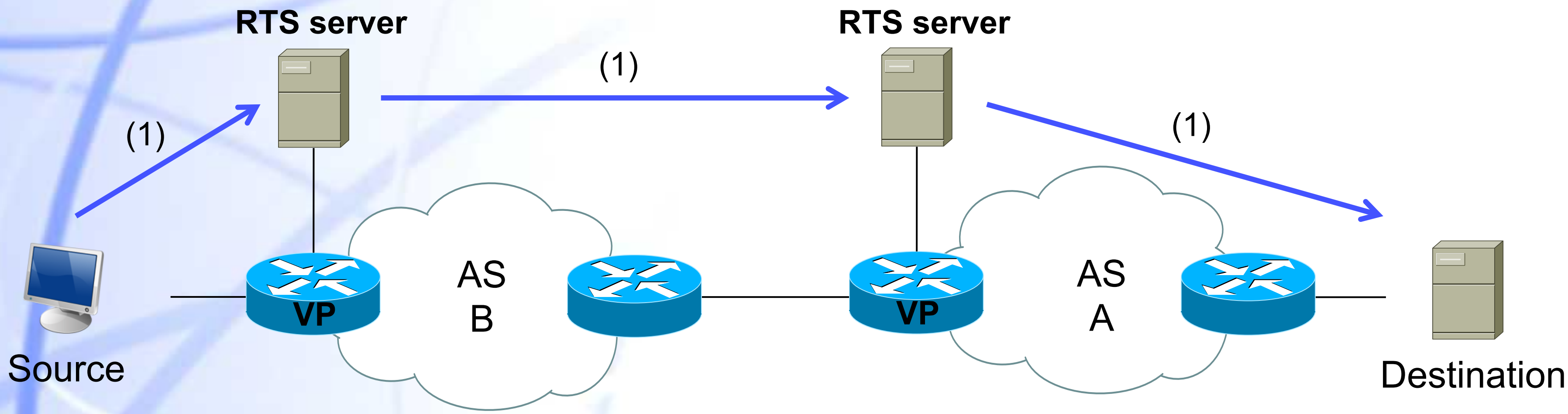
- **Request to Send (RTS) servers**
  - Provide the means to obtain tokens
  - Co-located with BGP speakers
- **Verification points (VPs)**
  - Perform access control
  - Part of the router line cards

Strawman design: Discovering RTS servers



- **Destination AS annotates their BGP advertisements with the IP address of the RTS server**
- **AS on the path adds its RTS server to the BGP advertisement**
- **Sources can get a series of RTS servers to send requests to destinations**

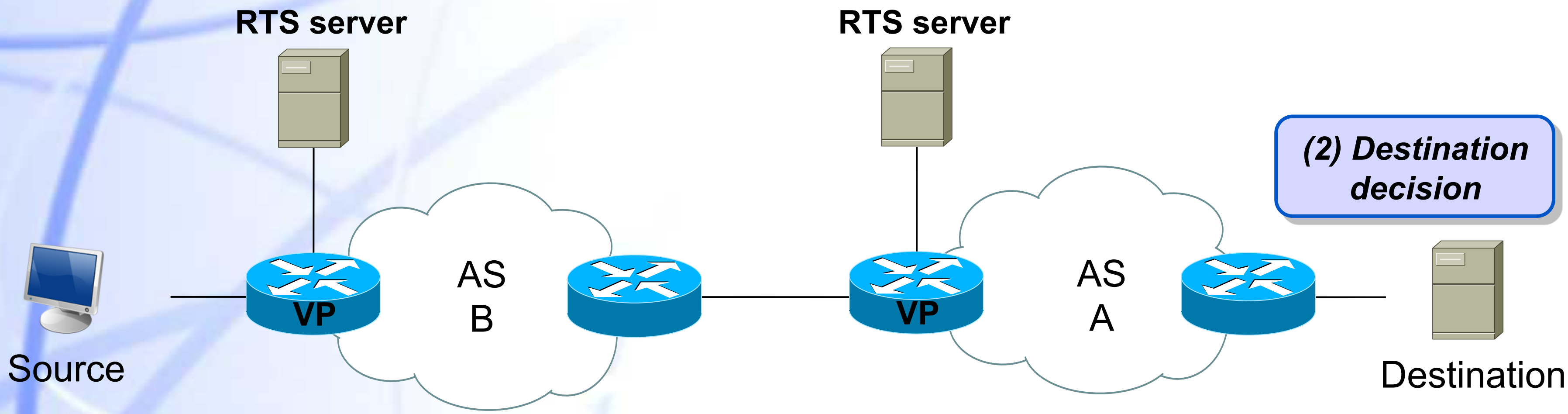
Strawman design: Obtaining capabilities



① **A source must obtain capabilities before sending traffic to a given destination:**

- **An RTS packet is sent to the first RTS server on the path to the destination**
- **The RTS request is relayed along the chain of RTS servers**

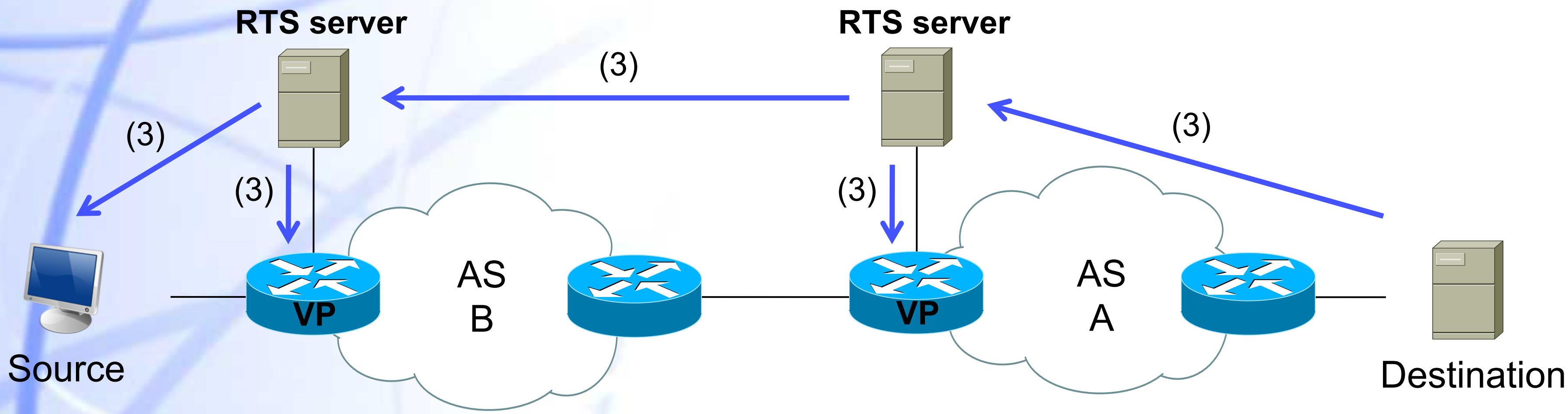
Strawman design: Obtaining capabilities



② **Destination decides whether to allow the source to send further packets**

- **Simple policies: to allow incoming traffic…**
  - ✓ **from well known remote locations,**
  - ✓ **in response to outgoing traffic,**
  - ✓ **etc.**

Strawman design: Obtaining capabilities



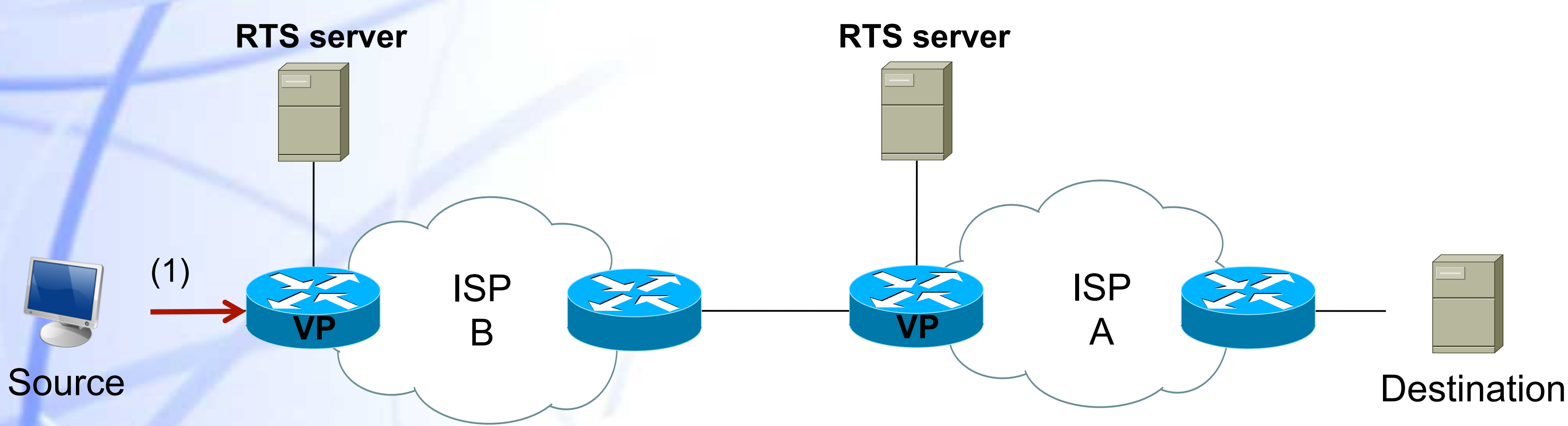
③ **If the destination authorizes the source, then:**

- **It mints a set of capabilities**
- **It sends a capability and an initial sequence value (s_0) to the source via the RTS servers**
- **The pair {capability, s_0} is associated with the authorized flow in the VPs**

Strawman design: Generating capabilities

- **If the destination allows the source to send it packets:**
  - **It generates a chain of K 64-bit one-way hash values: $h_1, h_2, \dots h_K$**
  - **Each h_i is a capability**
    - ✓ **It allows the source to send "*n*" packets in the next "*t*" seconds**
  - **The last hash value (h_K) is the first capability granted to the source**

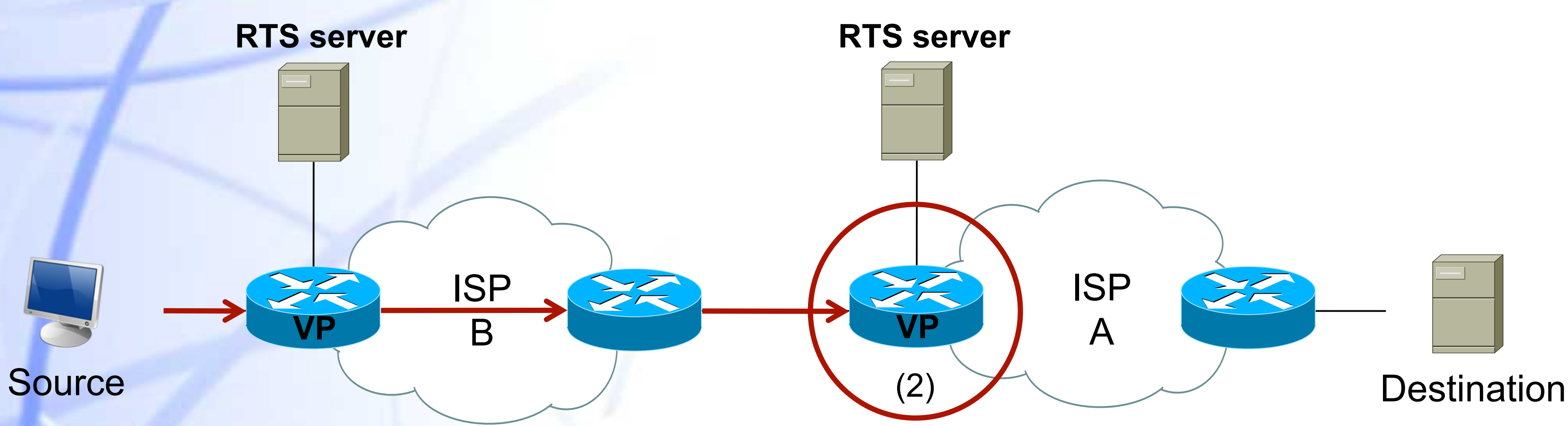
Strawman design: Sending with capabilities



① **After receiving the capability, the source:**

- **Starts the transmission**
- **Labels each packet with {capability, s_0}**

Strawman design: Sending with capabilities



② **When each VP receives a packet:**

- **IF {capability, s_0, flow identifiers} is stored THEN:**
  - ✓ **The packet is forwarded**
  - ✓ **The count of times the capability has been used is increased**
- **ELSE the packet is discarded**

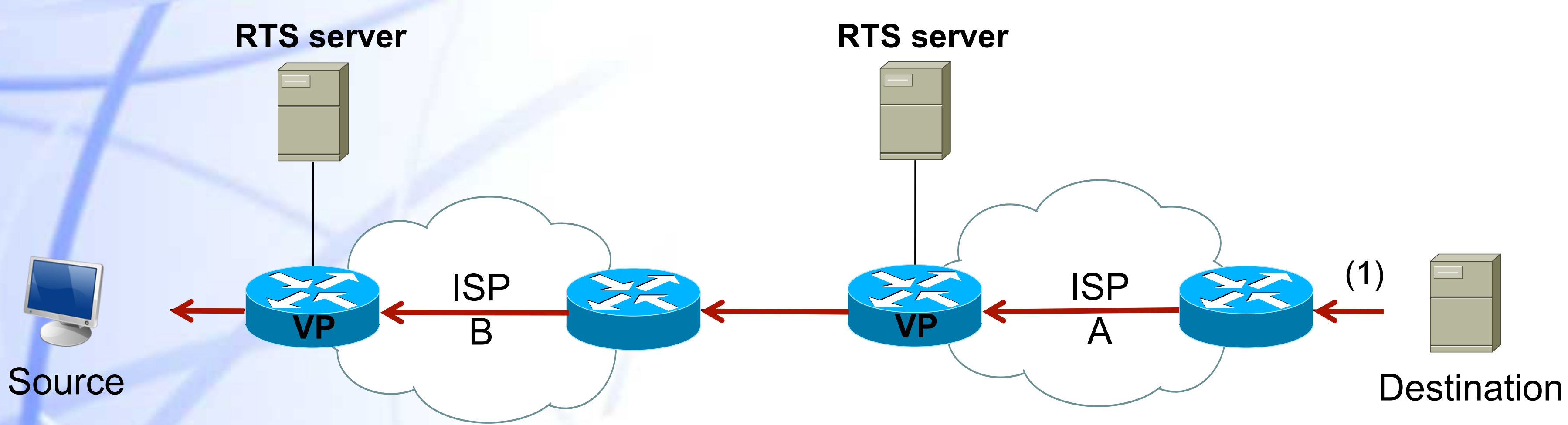
When the count reaches *n* or *t* seconds have passed:

- **The capability is flushed from the VP**

Strawman design: considerations

- The possession of capabilities provides authorization to use a network path
- The design assumes that:
  - Attackers cannot guess the capability values
  - Attackers cannot snoop links along the path
- However, the mechanism prevents attackers from using stolen capabilities to disrupt other unrelated paths

Strawman design: Acquiring new capabilities



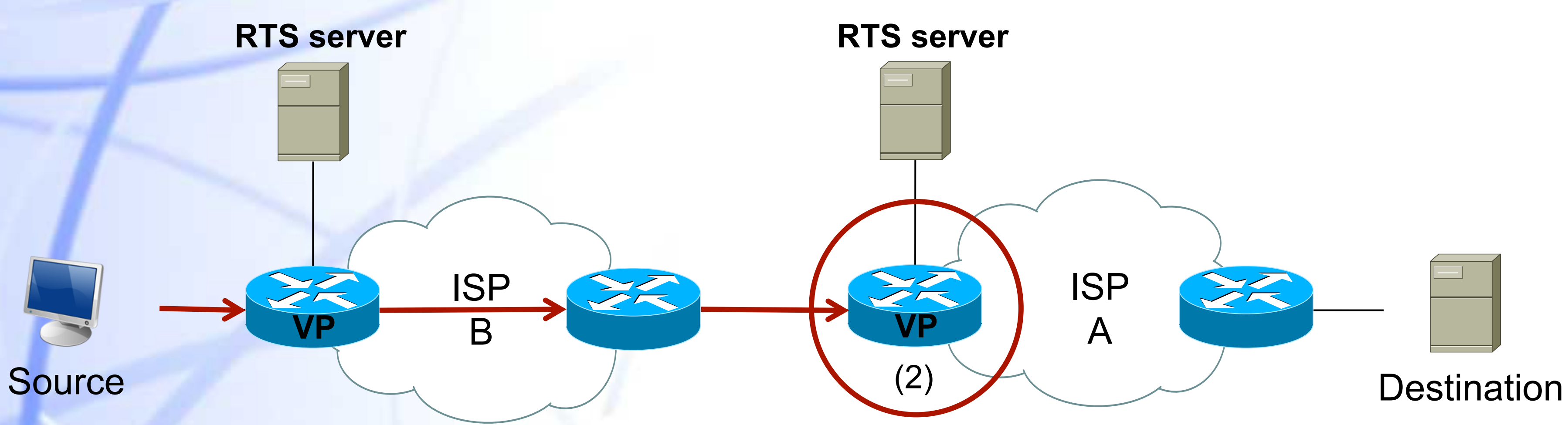
① **When almost *n* packets have been received:**

- **The destination sends h_{K-1} to the source**

After sending *n* packets with h_K, the source:

- **Switches to h_{K-1}**
- **Increments the sequence number to s_1**
- **Continues sending to the destination**

Strawman design: Acquiring new capabilities



② **When a VP receives a packet:**

- **IF the packet belongs to a known flow, with an incremented number of sequence and a new capability, THEN:**
  - ✓ **The VP verifies if $HASH\ (h_{k-1}) = h_K$**
  - ✓ **If so, it updates {capability, s_1} for the flow**

Strawman design: considerations

- The renewal procedure avoids considerable load on RTS servers
- VP must be provisioned to check and update a new capability at line rate
- Packet reordering can be handled by having VPs retain the previous capability

- The strawman design allows the destination to authorize communications from the source
  - Yet, it can selectively shut off any flow, by not revealing the previous hash value

Strawman design:
Protecting RTS servers from DoS

- **RTS servers should only receive requests from:**
  - **Local clients**
  - **Adjacent RTS servers**
- **Network filtering can discard all other traffic to RTS servers**
  - **Attackers cannot block the RTS channel except in their immediate vicinity**
  - **Even if hosts, routers, VPs and RTS servers are compromised, the damage is limited**

Strawman design: conclusion

- The strawman design is a proof-of-feasibility of the capability-based approach
- There is much room for improvement:
  - A destination can vary the granularity of authorization
    - ✓ Highly trusted sources can be granted large transmit windows
    - ✓ Suspicious sources can be treated cautiously
  - The scheme can be extended to protect links inside the network

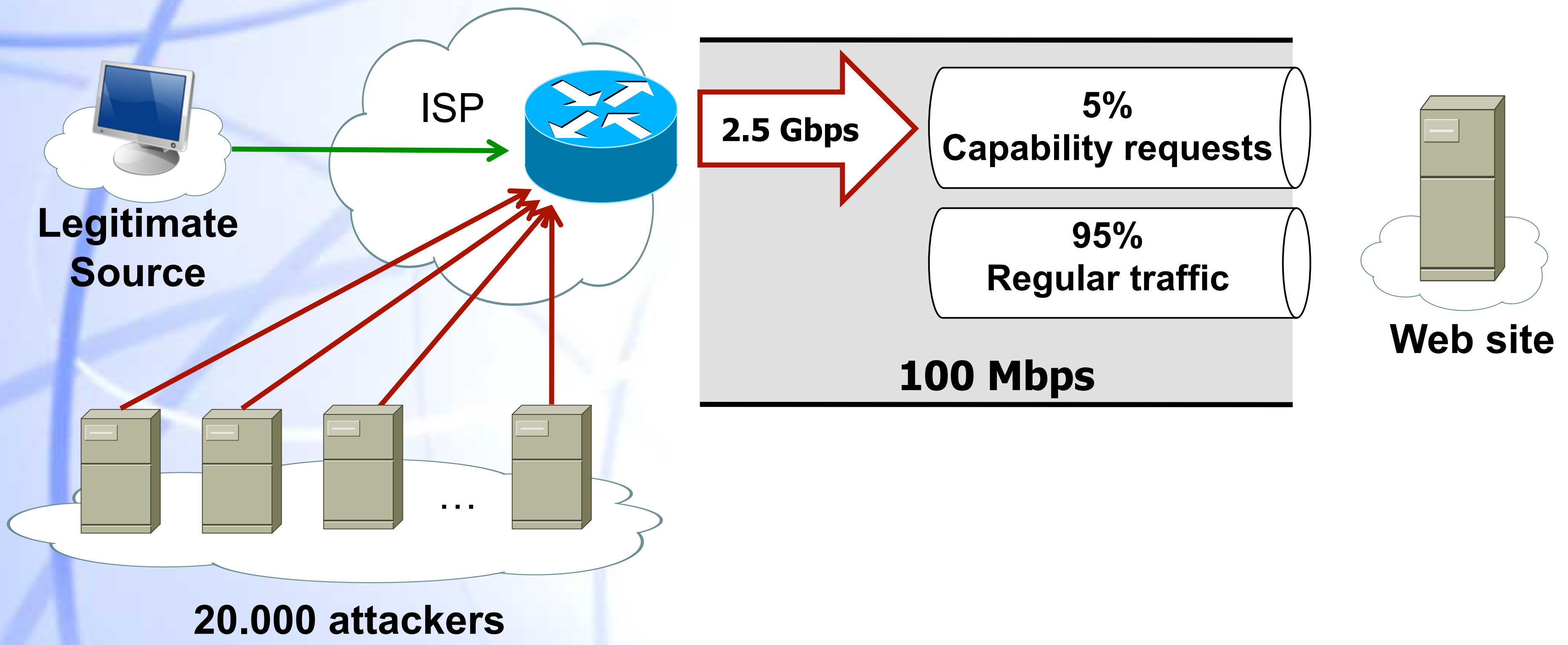
NETWORK CAPABILITIES

Challenges when developing a capability-based solution

Denial of Capability (DoC)

- **Described in:**
  - **K. Argyraki and D. Cheriton. "Network capabilities: The good, the bad and the ugly". In *Proceedings of Workshop on Hot Topics in Networks (HotNets-IV), November 2005***
- **Network capabilities are susceptible to Denial of Capability attacks:**
  - ***DoS against the capability distribution mechanism***

Denial of Capability (II)



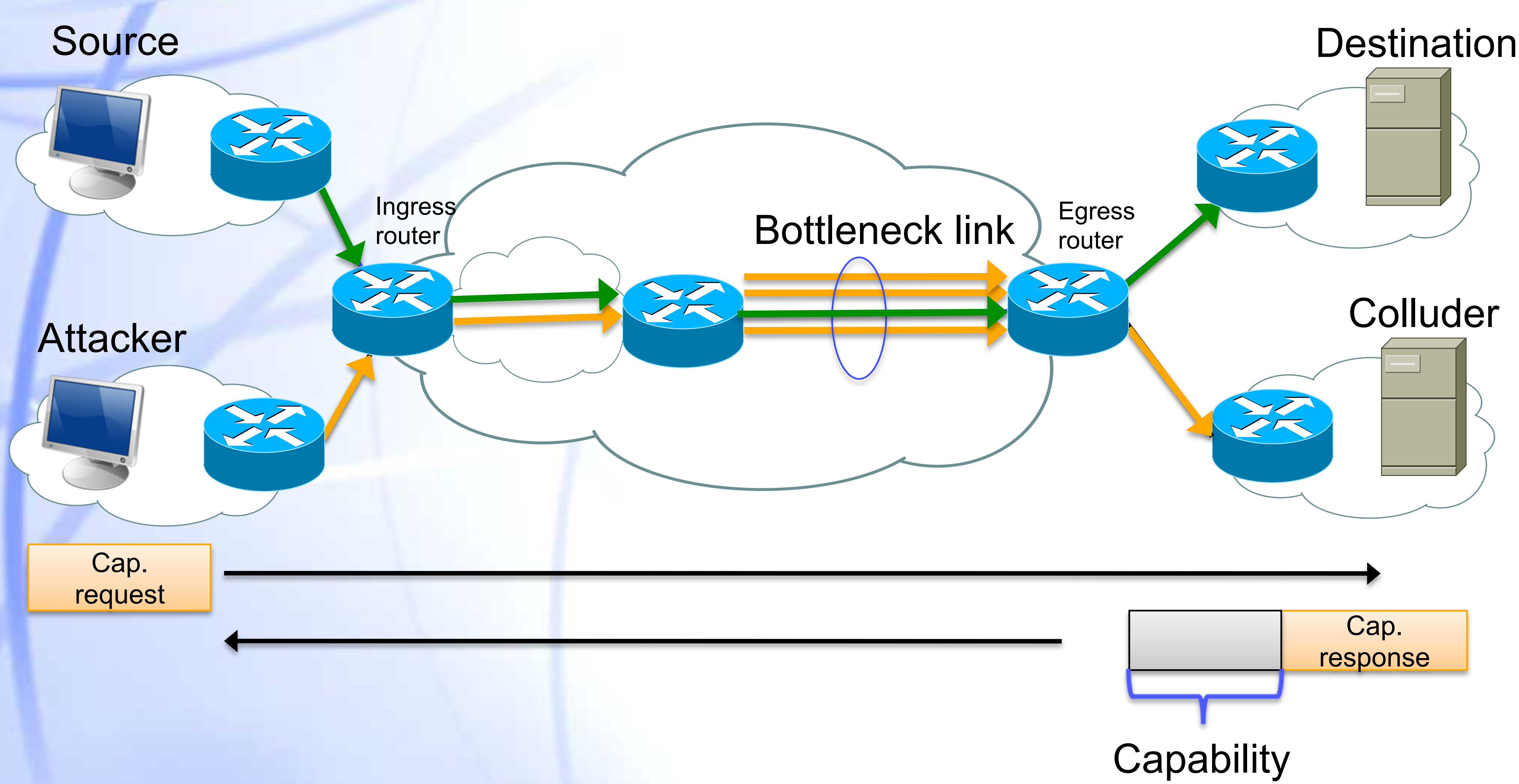
Denial of Capability (III)

- **Example:**
  - **Capability requests: 64 bytes long**
    - ✓ **The web site can accept 10000 capability requests per second**
    - ✓ **Attack sources generate 5 million capability requests per second!!**
  - **If a legitimate client retransmits the capability request every second:**
    - ✓ **The probability of the client accessing the web site within 20 seconds is ≈ 0.04**
    - ✓ **Average time of connection establishment greater than 8 min**

Denial of Capability (IV)

- To protect against DoC, extra anti-DoS mechanism is necessary
- Authors claim that:
  - Once that this anti-DoS mechanism has been deployed, it can be used to protect all the traffic
    - ✓ Capabilities are not necessary/sufficient to defend DoS
  - Can you find an argument against this reasoning?

Authorized traffic flood



Other challenges

- **Appropriate setup of destination policies**
  - **To discriminate between authorized and unauthorized requests**
- **Unforgeable capabilities**
  - **An attacker should not be able to forge a capability**
  - **It should not be able to use a capability generated for another party**
- **Resource constraints**
  - **Ex: memory and computing time at routers**
  - **Capabilities should work with bounded router state**
- **Accommodate route changes and failures**

Capability-based approach

- T. Anderson, T. Roscoe, and D. Wetherall. Preventing Internet Denial of Service with Capabilities. In *ACM HotNets-II, 2003.*
- *SIFF: A Stateless Internet Flow Filter to Mitigate DDoS Flooding Attacks. In IEEE Symposium on S&P, 2004. [37] X. Yang, D. Wetherall, and T. Anderson.*
- X. Yang, D. Wetherall, and T. Anderson. A DoS-limiting network architecture. In *Proceedings of ACM SIGCOMM, August 2005.*
- B. Parno, D. Wendlandt, E. Shi, A. Perrig, B. Maggs, and Y.-C. Hu. Portcullis: Protecting Connection Setup from Denial-of-Capability Attacks. In *ACM SIGCOMM, 2007*
- Maitreya Natu and Jelena Mirkovic. Fine-grained capabilities for flooding DDoS defense using client reputations. In *Proceedings of the 2007 workshop on Large scale attack defense (LSAD '07). ACM, New York, NY, USA, 105-112*
- L. Wang, Q. Wu, and D. D. Luong. Engaging edge networks in preventing and mitigating undesirable network traffic. In Workshop on Secure Network Protocol (NPSEC), October 2007
- Xin Liu, Xiaowei Yang, and Yong Xia. NetFence: preventing internet denial of service from inside out. In *Proceedings of the ACM SIGCOMM 2010). ACM, New York, NY, USA, 255-266.*