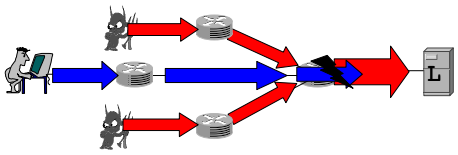
# TVA: A DoS-limiting Network Architecture

Xiaowei Yang (UC Irvine)
David Wetherall (Univ. of Washington)
Thomas Anderson (Univ. of Washington)



## DoS is not even close to be solved



- Address validation is insufficient (botnets)
- Traceback is too little too late (detection only)
- Pushback lacks discrimination (imprecise)
- Secure overlay filtering requires offline authenticators (public servers)

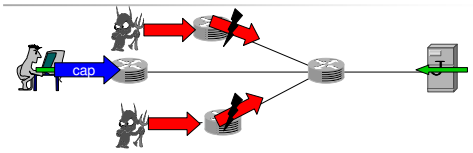


## Capabilities are a promising approach

- Destination control
  - The destinations know better.
- Network filtering based on explicit and unforgeable packet state, i.e., capabilities
  - Only the network can shed load before the damage has been made.
- Anderson et al. [Anderson03], Yarr et al. [Yarr04]



## Sketch of the capability approach




1. Source requests permission to send.
2. Destination authorizes source for limited transfer, e.g, 32KB in 10 secs
   - A capability is the proof of a destination's authorization.
3. Source places capabilities on packets and sends them.
4. Network filters packets based on capabilities.



## Capabilities alone do not effectively limit DoS

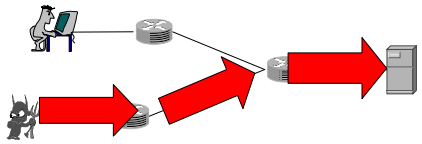
- Goal: minimize the damage of the arbitrary behavior of k attacking hosts.
  - Non-goal: make DoS impossible
- Problems
  1. Request or authorized packet floods
  2. Added functionality in a router's forwarding path
  3. Authorization policies
  4. Deployment
- TVA addresses all of the above.



## Challenges

1. Counter a broad range of attacks, including request and authorized packet floods
2. Router processing with bounded state and computation
3. Effective authorization policies
4. Incrementally deployable

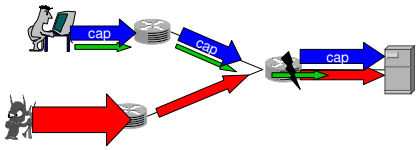
## Request packet floods



- Request packets do not carry capabilities.

7

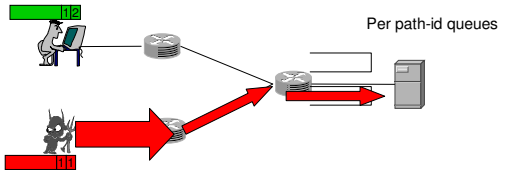
## Counter request packet floods (I)




- Rate-limit request packets

8

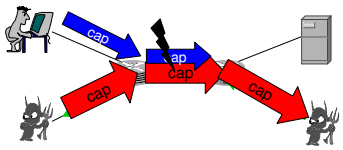
## Counter request packet floods (II)




- Rate-limit request packets
- Routers insert path identifier tags [Yarr03].
- Fair queue requests using the most recent tags.

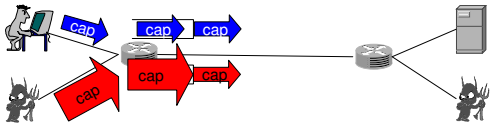
9

## Authorized packet floods




10

## Counter authorized packet floods




- Per-destination queues
- TVA bounds the number of queues.

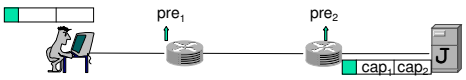
11

## Challenges

1. Counter a broad range of attacks, including request packet floods and authorized packet floods
2. Router processing with bounded state and computation
3. Effective authorization policies

12

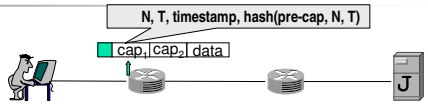
## TVA's implementation of capabilities




- Routers stamp pre-capabilities on request packets
  - **(timestamp, hash(src, dst, key, timestamp)**
- Destinations return fine-grained capabilities
  - **(N, T, timestamp, hash(pre-cap, N, T))**
  - send N bytes in the next T seconds, e.g. 32KB in 10 seconds

13

## Validating fine-grained capabilities




1. A router verifies that the hash value is correct.
2. Checks for expiration: *timestamp + T ≤ now*
3. Checks for byte bound: *sent + pkt_len ≤ N*

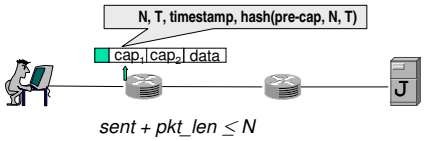
14

## Bounded computation

- The main computation overhead is hash validation.
- On a Pentium Xeon 3.2GHz PC
  - Stamping pre-capabilities takes 460ns
  - Validating capabilities takes 1486ns

15

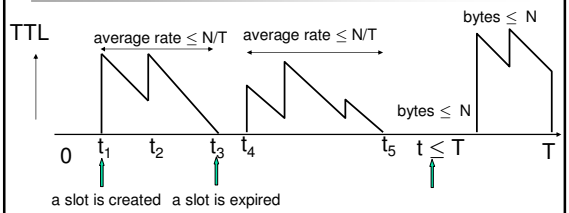
## Bounded state




- Create a slot if a capability sends faster than N/T.
- For a link with a fixed capacity C, there are at most C/(N/T) flows
- Number of slots is bounded by C / (N/T)

16

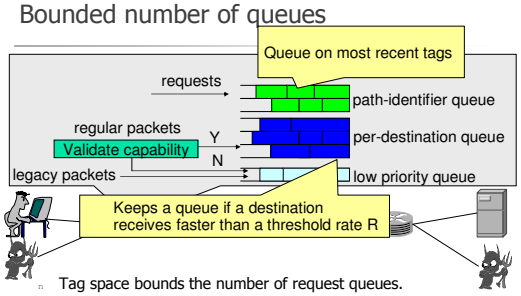
## Worst case byte bound is 2N in T seconds




- If a slot expires, it indicates that a capability sends slower than N/T.

17

## Bounded number of queues




- Tag space bounds the number of request queues.
- Number of destination queues is bounded by C/R

18

## Challenges

1. Counter a broad range of attacks, including request packet floods and authorized packet floods
2. Router processing with bounded state and computation
3. Effective authorization policies



## Simple policies can be effective

- Fine-grained capabilities tolerate authorization mistakes.
- Client policy
  - Authorize requests that match outgoing ones
- Public server policy
  - Authorize all initial requests
  - Stop misbehaving senders
  - A server has control over its incoming traffic when overload occurs.



## Evaluation

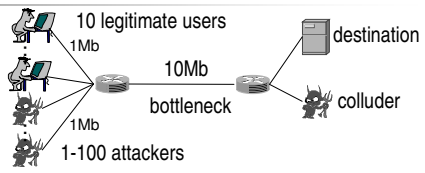


## Overview of different schemes

- SIFF [Yarr04]
  - request and legacy traffic have the same priority
  - authorized traffic has a higher priority
  - time-limited capabilities
- Pushback [Mahajan01, Ioannidis02]
  - Network controlled filtering
- Legacy Internet
  - best-effort



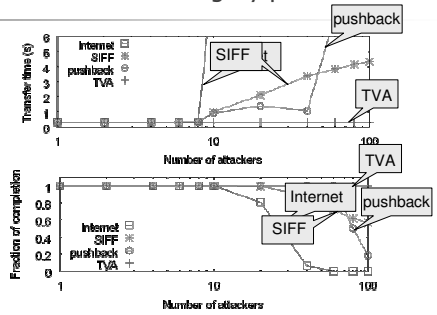
## Ns-2 Simulation Setup



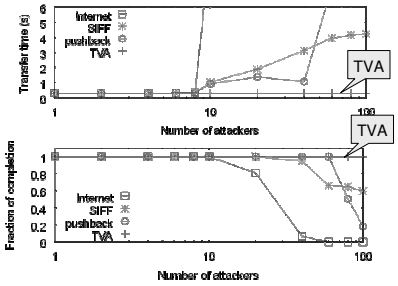

- Scale down topology to speed up simulations
- Two metrics:
  - The transfer time of a fixed-length file (20KB)
  - Fraction of completed transfers



## TVA is able to limit legacy packet floods

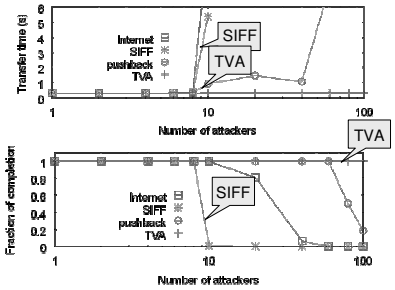


## TVA is able to limit request packet floods



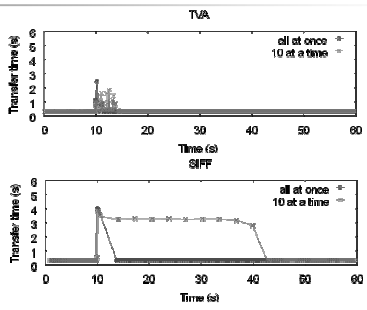

25

## TVA is able to limit authorized packet floods




26

## Simple policies can be effective



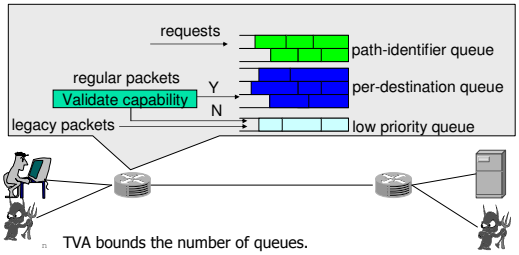

27

## Conclusion

- Key contribution
  - a comprehensive and practical capability system for the first time.
- We made TVA practical in three aspects
  - Counter a broad range of attacks
  - Bounded state and computation
  - Simple and effective authorization policies
- Coming next
  - Testbed implementation
    - Request rate limit, queuing scheme
  - Robust service differentiation
    - Traffic with different priority

28

## Types of Queues inside a TVA-router




- TVA bounds the number of queues.

29

## TVA's implementation of capabilities


- Routers stamp pre-capabilities on request packets
  - **(timestamp, hash(src, dst, key, timestamp)**
- Destinations return fine-grained capabilities
  - **(N, T, timestamp, hash(pre-cap, N, T))**
  - send N bytes in the next T seconds, e.g. 32KB in 10 seconds

30

ERROR: undefined
OFFENDING COMMAND:

STACK: