

IP Spoofing

- Sending packets with fake source IP address
 - Impersonate someone with special trust
 - A lot of service requests with a specific address
 - Servers will send replies to this address, overwhelming it – reflector DDoS attacks
 - Malicious packets with random addresses
 - Do some damage and avoid being caught
 - A lot of service requests with random addresses
 - Overwhelm the server who thinks that many clients want to access it – DDoS

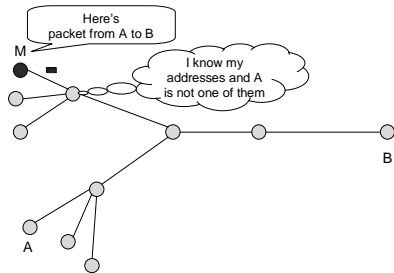
1

IP Spoofing

- How come someone can send packets with fake addresses and no one notices?
 - No one checks!
 - Even if someone would check how would we know that the address is fake?
 - Known fake addresses
 - Check close to the source, whether the address belongs within an assigned network range (Ingress filtering)
 - Check in the core, whether the route is correct for the given source address (Route-based filtering) or whether some other parameter is within range (Feature filtering)

2

Ingress Filtering



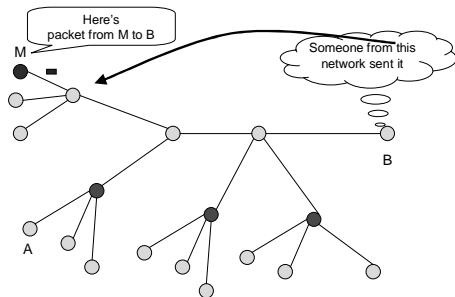
3

Ingress Filtering

- Ingress filtering cannot solve the problem
 - It will never be deployed everywhere
 - If ingress filtering is not deployed everywhere attackers can still spoof any address on the Internet
 - There are ways around ingress filtering

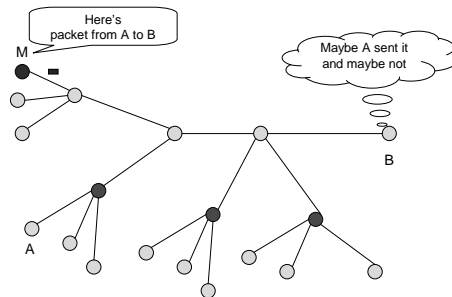
4

Purple Nodes Run Ingress Filtering



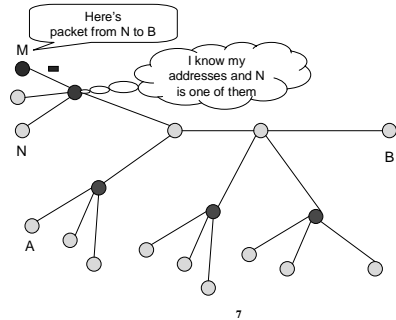
5

Purple Nodes Run Ingress Filtering



6

A Way Around — Subnet Spoofing



7

Why Don't People Run Ingress Filtering?

- It is easy! It improves security! Why not run it?
- Some people run it
- In current routers it is implemented in the slow path — in the software not the hardware
- It needs a very large deployment to improve the situation

8

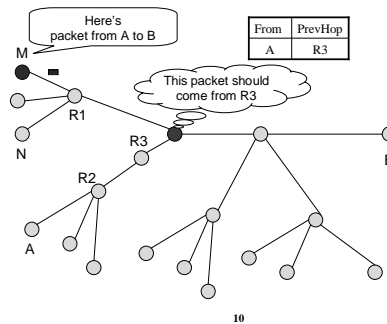
Route-Based Filtering

- Deploy filtering in the core
 - As Internet service to combat spoofing
- Filter based on “expected” path the traffic from a given source address should follow
 - If the traffic comes on another path, drop it
- Intuitively, we would need fewer deployment points

“On the Effectiveness of Route-Based Packet Filtering for Distributed DoS Attack Prevention in Power-Law Internets”, Kihong Park, Heejo Lee, Proceedings of SIGCOMM 2001

9

Route Based Filtering



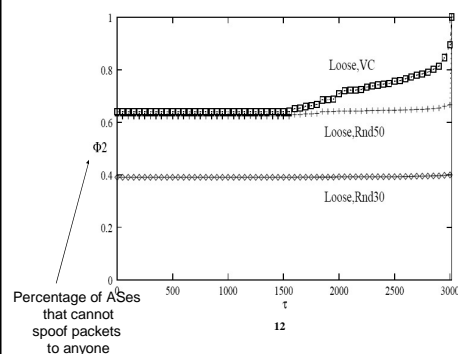
10

Route-Based Filtering

- Now we need a few core nodes to run the filtering
 - Deployment at 18% of core ASes would handle 90% of spoofed packets
 - But not just at any 18% - they have to form a *vertex cover* of core AS topology
 - We are naturally looking for the smallest vertex cover which is NP-hard problem but know heuristics exist
 - Rough approximation of a vertex cover chooses 18% of best-connected nodes

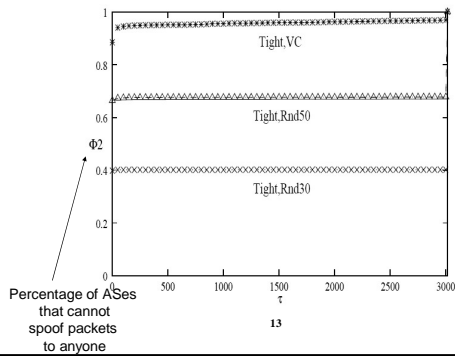
11

Route-Based Filtering



12

Route-Based Filtering



Route-Based Filtering

- Additional performance results:
 - Should we keep per-source or per-source-and-destination tables? (semi-maximal vs. maximal filters)
 - It doesn't matter, performance doesn't change
 - So we can keep only per-source tables – it is cheaper
 - Can we trace back the attack?
 - Yes to 4-5 possible ASes

14

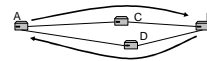
Route-Based Filtering vs Traceback

- Which one is better?
 - More accurate?
 - Less expensive?

15

How to Build Incoming Tables?

- This is challenging because:
 - Asymmetric routing - path from A to B can be different than path from B to A



- Routing policies - each router selects routes based on its policy
- Routing changes - incoming interface information must be updated promptly when routing change occurs

"SAVE: Source Address Validity Enforcement Protocol",
J. Li, J. Mirkovic, M. Wang, P. Reiher and L. Zhang, Proceedings of INFOCOM 2002

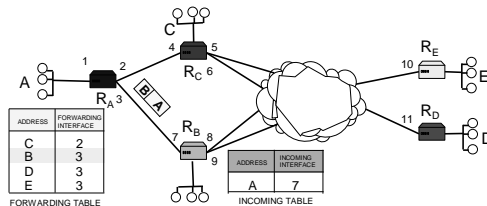
16

SAVE Protocol

- Assumes complete deployment
 - At least within the island of routers
- SAVE builds *Incoming table* at each router through:
 - Generating *SAVE updates*
 - Processing and forwarding *SAVE updates*
- Final result is that all routers build proper tables

17

SAVE Protocol



18

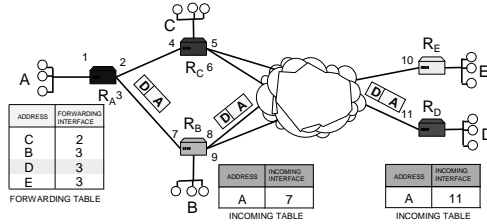
SAVE Update Generation

- Each SAVE router is assigned a *source address space (SAS)*
 - A range of IP addresses who use this router as an exit router for some set of destinations
- Independent of the underlying routing protocol
- A *periodic* SAVE update is generated for every entry in the forwarding table and sent to the next hop
- Forwarding table change invokes the generation of *triggered* SAVE update for the changed entry

19

Updates Benefit Multiple Routers

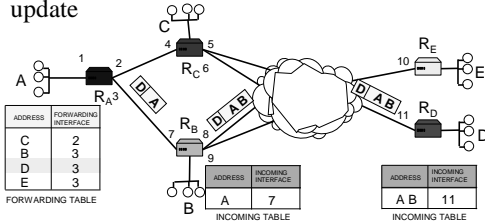
- Intermediate routers update their incoming tables



20

Updates Can Be Aggregated

- Intermediate router can piggyback its incoming interface information to a passing update

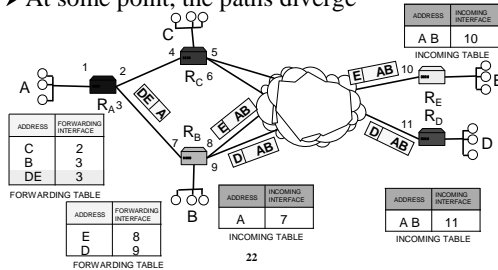


21

21

Sometimes Updates Must Be Split

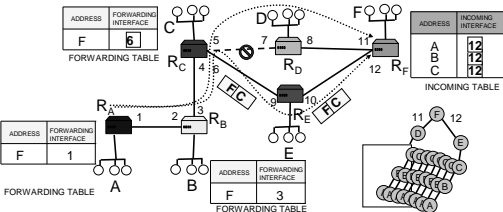
- Addresses in forwarding tables are aggregated
- At some point, the paths diverge



22

22

Reacting to Routing Changes

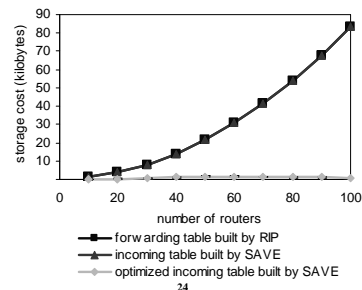


- *Incoming tree* is constructed at every SAVE router
- Each node represents a source address space
- A child inherits same incoming interface as its parent
- Change applies to all descendants on the tree

23

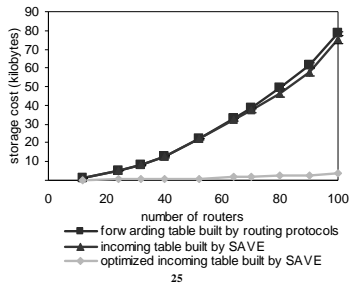
23

Storage Cost Within AS



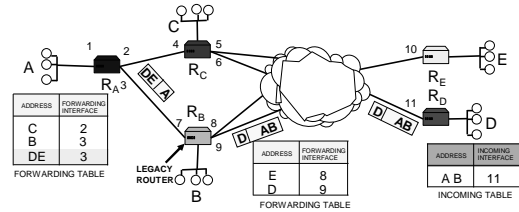
24

Storage Cost Between ASes



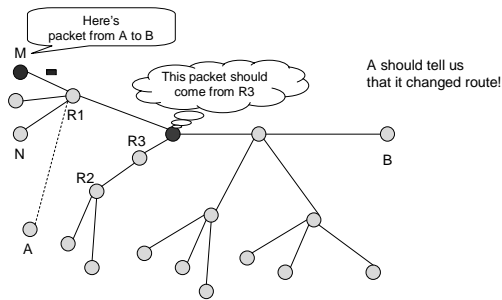
Partial Deployment

► Problem when path is split at legacy router

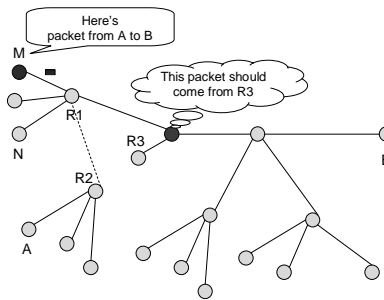


► Problem when routing change affects only the legacy router

Route Change



Route Change



A Way Around — En-Route Spoofing

