Routing

Jeff Chase
Duke University
IP Routing

From Click
Internet Map

From CAIDA
IP Address Allocation

- Originally ("classful" addrs), 4 address classes
  - “A”: 0 | 7 bit network | 24 bit host (1M each)
  - “B”: 10 | 14 bit network | 16 bit host (64K)
  - “C”: 110 | 21 bit network | 8 bit host (255)
  - “D”: 1110 | 28 bit multicast group #
- Assign net # centrally, host # locally
  - IBM has class A address
  - Duke has class B address
- What is a network “prefix”?

{razor,vahdat}@cs.duke.edu
IP Address Issues

• We can run out
  - 4B IP addresses; 4B microprocessors in 1997
• We’ll run out faster if sparsely allocated
  - Rigid structure causes internal fragmenting
  - E.g., assign a class C address to site with 2 computers
    • Waste 99% of assigned address space
• Need address aggregation to keep tables small
  - 2 million class C networks
  - Entry per network in IP forwarding tables
    • Scalability?

{razor,vahdat}@cs.duke.edu
Efficient IP Address Allocation

• Subnets
  - Split net addresses between multiple sites

• Supernets
  - Assign adjacent net addresses to same organization
  - Classless routing (CIDR)
    • Combine routing table entries whenever all nodes with same prefix share same hop

• Hardware support for fast prefix lookup

{razor,vahdat}@cs.duke.edu
Physical Networks and IP Addresses

• Originally: network part of IP address identifies exactly one physical network
  – What about large campuses with many physical networks?
Subnetting

• Subnetting: introduce *subnet masks*
  - All hosts on same network already have same network #
  - Subnet mask: hosts on one network have same subnet #
  - Subnet mask: 255.255.255.128, IP: 128.96.34.15
    • This says top 25-bits identify the network
    • Class B: 16-bits for network #, 9-bits for subnet
    • Logical AND Host and mask for Subnet #
    • 128.96.34.15 AND 255.255.255.128 → 128.96.34.0

{razor,vahdat}@cs.duke.edu
Subnetting and Forwarding

• Task of forwarding changes:
  - Hosts check if on same subnet (using mask)
• Task of routers change:
  - Replace \(<\text{network }\#, \text{ next hop}>\) with (must send prefix):
    • \(<\text{subnet }\#, \text{ subnet mask}, \text{ next hop}>\)
  - For each dest IP addr
    • Perform logical AND of IP addr with mask
    • Compare to subnet #
  - How to do this efficiently?

{razor,vahdat}@cs.duke.edu
Classless Interdomain Routing (CIDR)
- Balances between need for fewer entries in forwarding tables and need to efficiently distribute IP address space
- Example: site that requires 16 class-C IP addresses
  - Use 16 contiguous class C addrs, e.g., 192.4.16-192.4.31
  - Top 20 bits are identical
  - Between a class B and class C addr
    - "Classless"
- Need routing protocols to recognize CIDR

{razor,vahdat}@cs.duke.edu
On Network Prefixes

- All these network addresses describe the same network
- 152.3.128.0/17
- 152.3.128.15/17
- 152.3.128/17
- 152.3.128.0/255.255.128.0
- 152.3.128.75/255.255.128.0

- This network has a prefix of 17 (most significant bits in address)
Subnetting vs. Supernetting

- Subnetting attempts to share one address among multiple physical networks
- Supernetting attempts to collapse multiple addresses assigned to single Autonomous System (AS) onto one address
- CIDR essentially discards all class-based addressing
  - Use prefix notation now
Interdomain Routing

• Two kinds of networks/domains
  - Stub
  - Transit (ISP)

• Three kinds of relationships for each hop destination:
  - Provider: transit provides service for a stub or another transit. (uphill: +1)
  - Peer: two networks exchange traffic. (sideways: 0)
  - Customer. (downhill: -1)

• Valley-free paths
  - Type 1: {+1}*{-1}*
  - Type 2: {+1}*0{-1}*

Routes

• BGP speakers know of three kinds of routes:
  - My routes (for traffic destined to me)
  - Routes learned from a provider
  - Routes learned from a peer
  - Routes learned from a customer

• Specific relationships
  - Sibling is a kind of peer (same owner, exchange all routes).
  - Backup: peer or provider that is less preferred, for use only when the primary path fails.
Export Rules

• Driven by self-interest
  - I want to get good service for my customers.
  - I want you to have good service too, but not at my expense.

• Exporting to provider or peer
  - My routes and my customer routes
  - Not routes from peers or other providers

• Exporting to a customer
  - All routes I know
Malicious Routers

• Can a router suppress paths advertised by its neighbors?
• Can a router lie about its own identity?
• Can a router synthesize a fake path to an origin?
  - Hijacking
  - Lie about neighbor advertisements
• Can a router modify the paths advertised by its neighbors?
• Can colluding routers advertise a fake path between them? Why would they do such a thing?
• What defenses do we have against these attacks?
Defenses

• Prevent routers from lying about what someone else has said to them.
• Prevent adversaries from interposing on communication between routers.
• Detect inconsistent paths and suppress paths through the likely adversary?
• How to identify the source of a problem?
Whisper

• Simple hashing can prevent an adversary from faking a shorter path to an origin than the adversary itself has.
• However, an adversary can modify advertised paths as long as it does not change their length.
• “Strong whisper” enables detection of modified paths as “inconsistent” by any other router that learns of multiple paths to the same origin.
Suppressing Bogus Paths

- Problem: whisper cannot identify the adversary, or even which route in an inconsistent pair is bogus.
- Solution: guess.
- The adversary is always present in the AS path for a bogus route.
- Its neighbors can always guarantee this property.
  - (If the neighbor fails to do this then we can consider the neighbor as an adversary.)
- Downgrade the reputation of all AS IDs on any path that is part of an inconsistent pair.
- Avoid paths through disreputable Autonomous Systems.
Listen

- Identify black holes by watching for completed TCP connections.
- Problem: may only see one direction of flow.
- Solution: if you see data after a SYN, it’s probably OK.
- Problem: An adversary can fake completed connections.
- Solution: drop some packets and see if it notices.
- Problem: it can pretend to notice.
- Solution: monitor to see if it is pretending...