Announcements

• HW8 due this week
• HW9 is out
• Student presentations
$ dig uh.edu

; <<>> DiG 9.7.3-P3 <<>> uh.edu
;; global options: +cmd
;; Got answer:
;; >>>HEADER<<< opcode: QUERY, status: NOERROR, id: 44950
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 4

;; QUESTION SECTION:
uh.edu. IN A

;; ANSWER SECTION:
uh.edu. 16306 IN A 129.7.97.54

;; AUTHORITY SECTION:
uh.edu. 90 IN NS mesquite.cc.uh.edu.
uh.edu. 90 IN NS ns2.uh.edu.
uh.edu. 90 IN NS ns1.uh.edu.
uh.edu. 90 IN NS ncc.uky.edu.

;; ADDITIONAL SECTION:
ncc.uky.edu. 80512 IN A 128.163.1.6
ns2.uh.edu. 22524 IN A 129.7.1.6
ns1.uh.edu. 27472 IN A 129.7.1.1
mesquite.cc.uh.edu. 27584 IN A 66.140.111.1

;; Query time: 0 msec
;; SERVER: 129.7.240.1#53(129.7.240.1)
;; WHEN: Mon Mar 26 12:00:34 2012
;; MSG SIZE  rcvd: 188
HW8

- Distance Vector Routing
- Count-to-infinity
- Split-horizon
Today’s Topics

- BGP Wedgies
- IP
- NAT
- Student presentations
Multiple Stable Configurations
BGP Wedgies [RFC 4264]

• Typical policy:
  – Prefer routes from customers
  – Then prefer shortest paths
BGP Wedgies
BGP Wedgries

3 2 ...

2 1 1 1 1

1.2.0.0/16: 1 1 1 1 (Backup Path)

Peering

1 1 1 1 1

1.2.0.0/16: 1
Primary Path

4 3 2 ...

5 4 3 2 ...

4 3 2 ...

1.2.0.0/16: 1 1 1 1 1

Peering
BGP Wedgies

3 prefers customer route: stable configuration!

3 2 ...

1.2.0.0/16: 1 1 1 1 1 (Backup Path)

1 1.2.0.0/16: 1 Primary Path

2 1 1 1 1

4 5 1

5 1
BGP Security Goals

• Confidential message exchange between neighbors
• Validity of routing information
  – Origin, Path, Policy
• Correspondence to the data path
Proposed Solution: S-BGP

- Based on a public key infrastructure
- Address attestations
  - Claims the right to originate a prefix
  - Signed and distributed out of band
  - Checked through delegation chain from ICANN
- Route attestations
  - Attribute in BGP update message
  - Signed by each AS as route along path
- S-BGP can avoid
  - Prefix hijacking
  - Addition, removal, or reordering of intermediate ASes
S-BGP Deployment

• Very challenging
  – PKI
  – Accurate address registries
  – Need to perform cryptographic operations on all path operations
  – Flag day almost impossible
  – Incremental deployment offers little incentive

• But there is hope! [Goldberg et al, 2011]
  – Road to incremental deployment
  – Change rules to break ties for secure paths
  – If a few top Tier-1 ISPs
  – Plus their respective stub clients deploy simplified version (just sign, not validate)
  – Gains in traffic => $ => adoption!
Data Plane Attacks

• Routers/ASes can advertise one route, but not necessarily follow it!
• May drop packets
  – Or a fraction of packets
  – What if you just slow down some traffic?
• Can send packets in a different direction
  – Impersonation attack
  – Snooping attack
• How to detect?
  – Congestion or an attack?
  – Can let ping/traceroute packets go through
  – End-to-end checks?
• Harder to pull off, as you need control of a router
Forwarding with CIDR

• Longest Prefix Match

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Nexthop</th>
</tr>
</thead>
<tbody>
<tr>
<td>a.b.0.0/23</td>
<td>A</td>
</tr>
<tr>
<td>a.b.1.0/24</td>
<td>C</td>
</tr>
</tbody>
</table>

Where to forward these packets?
- dst: a.b.0.5
- dst: a.b.1.6
IP Protocol

• Provides addressing and *forwarding*
  – Addressing is a set of conventions for naming nodes in an IP network
  – Forwarding is a local action by a router: passing a packet from input to output port

• IP forwarding finds output port based on destination address
  – Also defines certain conventions on how to handle packets (e.g., fragmentation, time to live)

• Contrast with *routing*
  – Routing is the process of determining how to map packets to output ports (topic of next two lectures)
Service Model

• Connectionless (datagram-based)
• Best-effort delivery (unreliable service)
  – packets may be lost
  – packets may be delivered out of order
  – duplicate copies of packets may be delivered
  – packets may be delayed for a long time

• It’s the lowest common denominator
  – A network that delivers no packets fits the bill!
  – All these can be dealt with above IP (if probability of delivery is non-zero...)

# IP v4 packet format

<table>
<thead>
<tr>
<th></th>
<th>vers</th>
<th>hdr len</th>
<th>TOS</th>
<th>Total Length</th>
<th>Identification</th>
<th>Fragment offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Identification</td>
<td>Fragment offset</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>TTL</th>
<th>Protocol</th>
<th>hdr checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Source IP address</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Destination IP address</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Options</th>
<th>Padding</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Data
IP header details

• Forwarding based on destination address
• TTL (time-to-live) decremented at each hop
  – Originally was in seconds (no longer)
  – Mostly prevents forwarding loops
  – Other cool uses...
• Fragmentation possible for large packets
  – Fragmented in network if crossing link w/ small frame
  – MF: more fragments for this IP packet
  – DF: don’t fragment (returns error to sender)
• Following IP header is “payload” data
  – Typically beginning with TCP or UDP header
Other fields

• Version: 4 (IPv4) for most packets, there’s also 6
• Header length: in 32-bit units (>5 implies options)
• Type of service (won’t go into this)
• Protocol identifier (TCP: 6, UDP: 17, ICMP: 1, …)
• Checksum over the header
Fragmentation & Reassembly

- Each network has maximum transmission unit (MTU)
- Strategy
  - Fragment when necessary (MTU < size of datagram)
  - Source tries to avoid fragmentation (why?)
  - Re-fragmentation is possible
  - Fragments are self-contained datagrams
  - Delay reassembly until destination host
  - No recovery of lost fragments
Fragmentation Example

- Ethernet MTU is 1,500 bytes
- PPP MTU is 576 bytes
  - R2 must fragment IP packets to forward them
Fragmentation Example (cont)

- IP addresses plus ident field identify fragments of same packet
- MF (more fragments bit) is 1 in all but last fragment
- Fragment offset multiple of 8 bytes
  - Multiply offset by 8 for fragment position original packet
Internet Control Message Protocol (ICMP)

- Echo (ping)
- Redirect
- Destination unreachable (protocol, port, or host)
- TTL exceeded
- Checksum failed
- Reassembly failed
- Can’t fragment
- Many ICMP messages include part of packet that triggered them

See
http://www.iana.org/assignments/icmp-parameters
ICMP message format

20-byte IP header
(protocol = 1—ICMP)

<table>
<thead>
<tr>
<th>Type</th>
<th>Code</th>
<th>Checksum</th>
</tr>
</thead>
</table>

depends on type/code
Example: Time Exceeded

<table>
<thead>
<tr>
<th>Type = 11</th>
<th>Code</th>
<th>Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>unused</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Student presentation: traceroute
Translating IP to lower level addresses

- Map IP addresses into physical addresses
  - E.g., Ethernet address of destination host
  - or Ethernet address of next hop router

- Techniques
  - Encode physical address in host part of IP address (IPv6)
  - Each network node maintains lookup table (IP->phys)
ARP – *address resolution protocol*

- Dynamically builds table of IP to physical address bindings
- Broadcast request if IP address not in table
- All learn IP address of requesting node (broadcast)
- Target machine responds with its physical address
- Table entries are discarded if not refreshed
ARP Ethernet frame format

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware type</td>
<td>Hardware type = 1</td>
</tr>
<tr>
<td>ProtocolType</td>
<td>ProtocolType = 0x0800</td>
</tr>
<tr>
<td>HLen</td>
<td>HLen = 48</td>
</tr>
<tr>
<td>PLen</td>
<td>PLen = 32</td>
</tr>
<tr>
<td>Operation</td>
<td>Operation</td>
</tr>
<tr>
<td>SourceHardwareAddr</td>
<td>SourceHardwareAddr (bytes 0–3)</td>
</tr>
<tr>
<td>SourceHardwareAddr</td>
<td>SourceHardwareAddr (bytes 4–5)</td>
</tr>
<tr>
<td>SourceProtocolAddr</td>
<td>SourceProtocolAddr (bytes 2–3)</td>
</tr>
<tr>
<td>TargetHardwareAddr</td>
<td>TargetHardwareAddr (bytes 0–1)</td>
</tr>
<tr>
<td>TargetHardwareAddr</td>
<td>TargetHardwareAddr (bytes 2–5)</td>
</tr>
<tr>
<td>TargetProtocolAddr</td>
<td>TargetProtocolAddr (bytes 0–3)</td>
</tr>
</tbody>
</table>
Format of IP addresses

• Globally unique (or made seem that way)
  – 32-bit integers, read in groups of 8-bits: 128.148.32.110

• Hierarchical: network + host

• Originally, routing prefix embedded in address

  – Class A (8-bit prefix), B (16-bit), C (24-bit)
  – Routers need only know route for each network
Forwarding Tables

• Exploit hierarchical structure of addresses: need to know how to reach *networks*, not hosts

<table>
<thead>
<tr>
<th>Network</th>
<th>Next Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>212.31.32.*</td>
<td>0.0.0.0</td>
</tr>
<tr>
<td>18.<em>.</em>.*</td>
<td>212.31.32.5</td>
</tr>
<tr>
<td>128.148.<em>.</em></td>
<td>212.31.32.4</td>
</tr>
<tr>
<td>Default</td>
<td>212.31.32.1</td>
</tr>
</tbody>
</table>

• Keyed by network portion, not entire address
• Next address should be local
Classed Addresses

• Hierarchical: network + host
  – Saves memory in backbone routers (no default routes)
  – Originally, routing prefix embedded in address
  – Routers in same network must share network part

• Inefficient use of address space
  – Class C with 2 hosts (2/255 = 0.78% efficient)
  – Class B with 256 hosts (256/65535 = 0.39% efficient)
  – Shortage of IP addresses
  – Makes address authorities reluctant to give out class B’s

• Still too many networks
  – Routing tables do not scale

• Routing protocols do not scale
• Add another level to address/routing hierarchy
• **Subnet mask** defines variable portion of host part
• Subnets visible only within site
• Better use of address space
R1’s Forwarding Table

<table>
<thead>
<tr>
<th>Network</th>
<th>Subnet Mask</th>
<th>Next Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>128.96.34.0</td>
<td>255.255.255.128</td>
<td>128.96.34.1</td>
</tr>
<tr>
<td>128.96.34.128</td>
<td>255.255.255.128</td>
<td>128.96.34.130</td>
</tr>
<tr>
<td>128.96.33.0</td>
<td>255.255.255.0</td>
<td>128.96.34.129</td>
</tr>
</tbody>
</table>
Supernetting

• Assign blocks of contiguous networks to nearby networks
• Called CIDR: Classless Inter-Domain Routing
• Represent blocks with a single pair
  – (first network address, count)
• Restrict block sizes to powers of 2
• Use a bit mask (CIDR mask) to identify block size
• Address aggregation: reduce routing tables
### CIDR Forwarding Table

<table>
<thead>
<tr>
<th>Network</th>
<th>Next Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>212.31.32/24</td>
<td>0.0.0.0</td>
</tr>
<tr>
<td>18/8</td>
<td>212.31.32.5</td>
</tr>
<tr>
<td>128.148/16</td>
<td>212.31.32.4</td>
</tr>
<tr>
<td>128.148.128/17</td>
<td>212.31.32.8</td>
</tr>
<tr>
<td>0/0</td>
<td>212.31.32.1</td>
</tr>
</tbody>
</table>
Obtaining IP Addresses

• Blocks of IP addresses allocated hierarchically
  – ISP obtains an address block, may subdivide
  ISP: 128.35.16/20  
    \text{10000000 \ 00100011 \ 00010000 \ 00000000}
  Client 1: 128.35.16/22  
    \text{10000000 \ 00100011 \ 00010000 \ 00000000}
  Client 2: 128.35.20/22  
    \text{10000000 \ 00100011 \ 00010100 \ 00000000}
  Client 3: 128.35.24/21  
    \text{10000000 \ 00100011 \ 00011000 \ 00000000}

• Global allocation: ICANN, /8’s (ran out!)
• Regional registries: ARIN, RIPE, APNIC, LACNIC, AFRINIC