CSCI-1680
Transport Layer Warmup (ish)
Nick DeMarinis

Based partly on lecture notes by Rodrigo Fonseca, Jennifer Rexford, Rob Sherwood, David Mazières, Phil Levis, John Jannotti
Administrivia: Grading for IP

• IP: was due last night
  – Look for email today/tomorrow about interactive grading

  “Between the time you’ve handed in and the demo meeting, you can continue to make minor tweaks and bug fixes, but you shouldn’t be making any major changes.”
Administrivia: Grading for IP

- IP: was due last night
  - Look for email today/tomorrow about interactive grading

“Between the time you’ve handed in and the demo meeting, you can continue to make minor tweaks and bug fixes, but you shouldn’t be making any major changes.”
  - OK: Fixing bugs, code cleanup
  - Not OK: Implementing RIP, adding new features
Administrivia: This week

- Today/Thursday: Intro to transport layer & TCP
- TCP assignment out today/tomorrow
- HW3: Due Monday
Today

• BGP security issues
• Intro to transport layer: ports and sockets
BGP Security Issues
BGP Recap

- Key protocol that holds Internet routing together
- Path Vector Protocol among Autonomous Systems (ASes)
- Route selection based on policy, rather than “optimal” routes
- What can go wrong?
Problem: IP address ownership

- IP address block assignment
  - Regional Internet Registries (ARIN, RIPE, APNIC)
  - Internet Service Providers
- Ideally, AS who owns prefix (or its providers) should advertise it
- However: BGP does not verify this
Prefix Hijacking

- Problem: Who "owns" a prefix?

- Who is allowed to originate a prefix?

  - BGP by default does not verify announce match the network that originates.
  
  \[ \implies \text{AS's have their own security policies but not enforced.} \]

**What can happen?**

An actor who does not own a prefix can advertise it to intercept traffic.
Prefix Hijacking

Consequences for the affected ASes
- Sinkhole: data traffic is discarded
- Snooping: data traffic is inspected, and then redirected
- Impersonation: data traffic is sent to bogus destinations
Hijacking is Hard to Debug

• Real origin AS doesn’t see the problem
  – Picks its own route
  – Might not even learn the bogus route

• May not cause loss of connectivity
  – E.g., if the bogus AS snoops and redirects
  – … may only cause performance degradation

• Or, loss of connectivity is isolated
  – E.g., only for sources in parts of the Internet

• Diagnosing prefix hijacking
  – Analyzing updates from many vantage points
  – Launching traceroute from many vantage points
WHAT CAN YOU DO?
- INTERCEPT OR REDIRECT PACKETS FOR A
- SNEAKING
- MODIFY/SLOW DOWN TRAFFIC.

HARD TO DEBUG BECAUSE MIGHT ONLY
BE VISIBLE FROM CERTAIN PARTS OR
NETWORK.

\[ \text{INTERNET} \]

\[ \text{10.0.0.0/8} \]

\[ \text{10.1.0.0/16} \]

IF B ANNOTISES MORE SPECIFIC PREFIX,
IT WINS! ALL TRAFFIC GOES THERE,
BECAUSE MORE SPECIFIC PREFIXES ARE
USUALLY PREFERABLE.
Sub-Prefix Hijacking

- Originating a more-specific prefix
  - Every AS picks the bogus route for that prefix
  - Traffic follows the longest matching prefix
How to Hijack a Prefix

• The hijacking AS has
  – Router with eBGP session(s)
  – Configured to originate the prefix

• Getting access to the router
  – Network operator makes configuration mistake
  – Disgruntled operator launches an attack
  – Outsider breaks into the router and reconfigures

• Getting other ASes to believe bogus route
  – Neighbor ASes not filtering the routes
  – … e.g., by allowing only expected prefixes
  – But, specifying filters on peering links is hard
Pakistan Youtube incident

- Youtube’s has prefix 208.65.152.0/22
- Pakistan’s government order Youtube blocked
- Pakistan Telecom (AS 17557) announces 208.65.153.0/24 in the wrong direction (outwards!)
- Longest prefix match caused worldwide outage
- [http://www.youtube.com/watch?v=IzLPKuAOe50](http://www.youtube.com/watch?v=IzLPKuAOe50)
Many other incidents

• China incident, April 8\textsuperscript{th} 2010
  – China Telecom’s AS23724 generally announces 40 prefixes
  – On April 8\textsuperscript{th}, announced \textasciitilde37,000 prefixes
  – About 10\% leaked outside of China
  – Suddenly, going to \url{www.dell.com} might have you routing through AS23724!

Russian hackers intercept Amazon DNS, steal $160K in cryptocurrency
What can be done?

Originally: Internet Routing Registries (IRRs): public database listing IP allocations

route: 10.0.0.0/8
descr: University of Blogging
descr: Anytown, USA
origin: AS65099
mnt-by: MNT-UNIVERSITY
notify: person@example.com
changed: person@example.com 20180101
source: RADB

But, database not verified and often incomplete/wrong
What can be done?

Brenda's ISP

$whois -h whois.radb.net AS14325

aut-num: AS14325
as-name: ASN-OSHEAN
descr: OSHEAN, Inc.
import: from AS14325:AS-MBRS accept PeerAS
mp-import: from AS14325:AS-MBRS accept PeerAS
export: to AS-ANY announce AS14325:AS-MBRS
mp-export: to AS-ANY announce AS14325:AS-MBRS
admin-c: Tim Rue
tech-c: Ventsislav Gotov
notify: vgotov@oshean.org
mnt-by: MAINT-AS14325
changed: vgotov@oshean.org 20210512
source: RADB

Can contain some info on this AS's policy.

In theory, should reflect how BGP announcements are sent.
Proposed Solution: RPKI

- Based on a public key infrastructure
- Address attestations
  - Claims the right to originate a prefix
  - Signed and distributed out of band, checked on BGP updates
  - Checked through delegation chain from ICANN
- Can avoid
  - Prefix hijacking
  - Addition, removal, or reordering of intermediate ASes

---

1. Every AS adds a signature of route info to DB, max prefix size.
   - Prevent others from advertising a more specific prefix.

2. ASes accepting routes supposed to validate against DB.

---

Can work, if everyone cooperates.
RPKI deployment

RPKI-ROV Analysis of Unique Prefix-Origin Pairs (IPv4)

- Valid: 35.12%
- Invalid: 0.74%

Unique P-O
TOTAL: 996,018
Not-Found: 638,780
64.13%

- Valid: 349,820
- Not-Found: 638,780
- Invalid: 7,418

Not-Found: 64.13%
RPKI at Brown?

FAILURE

Your ISP (Verizon, AS701) does not implement BGP safely. It should be using RPKI to protect the Internet from BGP hijacks. Tweet this →

▼ Details

- fetch https://valid.rpki.cloudflare.com
  - ✓ correctly accepted valid prefixes

- fetch https://invalid.rpki.cloudflare.com
  - ✗ incorrectly accepted invalid prefixes
Ports & Sockets
Layers, Services, Protocols

- **Application**
  - Service: user-facing application.
  - Application-defined messages

- **Transport**
  - Service: multiplexing applications to “ports”
  - Reliable byte stream to other node (TCP),
  - Unreliable datagram (UDP)

- **Network**
  - Service: move packets to any other node in the network
  - Internet Protocol (IP)

- **Link**
  - Service: move frames to other node across link.
  - May add reliability, medium access control

- **Physical**
  - Service: move bits to other node across link
The Network Layer

• Goal: move packets between hosts (anywhere on Internet)
The Transport Layer

Services for getting data to applications on a single host

- Multiplexing at same IP via port numbers
- Connected-ness: A stream or "session" between two nodes
- Reliable data delivery (TCP)
- Series of packets => Data stream
- Throughput management/Congestion control
Transport Layer

Applications (L7)

TCP

UDP

IP

NET_1

NET_2

NET_n

Use transport-layer protocol to enable application-layer protocols.
From Lec 2: OSI Model
What’s a port number?

• Ports define a communication endpoint, which is usually a process or service running on a host
• 16-bit unsigned number, 0-65535
What’s a port number?

- Ports define a communication endpoint, which is usually a process or service running on a host
- 16-bit unsigned number, 0-65535
- Ports numbered < 1024: “Well known port numbers”
  - Allocated by IANA

  *(Same authority for IPs)*

  *(Need to be root to listen)*
What’s a port number?

• Ports define a communication endpoint, which is usually a process or service running on a host
• 16-bit unsigned number, 0-65535
• Ports numbered < 1024: “Well known port numbers”
  – Allocated by IANA
• Higher numbers: “ephemeral ports” (>26K)
  – For general use by any application
  – Very high numbers (>20K): usually for source ports
### Some common ports

<table>
<thead>
<tr>
<th>Port</th>
<th>Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>20, 21</td>
<td>File Transfer Protocol (FTP)</td>
</tr>
<tr>
<td>22</td>
<td>Secure Shell (SSH)</td>
</tr>
<tr>
<td>23</td>
<td>Telnet (pre-SSH remote login)</td>
</tr>
<tr>
<td>25</td>
<td>SMTP (Email)</td>
</tr>
<tr>
<td>53</td>
<td>Domain Name System (DNS)</td>
</tr>
<tr>
<td>67, 68</td>
<td>DHCP</td>
</tr>
<tr>
<td>80</td>
<td>HTTP (Web traffic)</td>
</tr>
<tr>
<td>443</td>
<td>HTTPS (Secure HTTP over TLS)</td>
</tr>
</tbody>
</table>
Sockets

The kernel maps ports to sockets, which are used in applications like file descriptors to access the network.

Two modes for using ports/sockets:
- Applications “bind” to a port to accept new connections
  - `listen_mode`
  - `listen(ADDR, PORT)` (bind)

Individual connections use 5-tuple of source-dest port
- (protocol, source IP, source port, dest IP, dest port) => connection N

"Outgoing" connection, SRC -> DST
MODES FOR NEW CONNECTIONS

1. LISTEN MODE ("PASSIVE MODE")
   - WAIT FOR SOMEONE TO CONNECT

2. ACTIVE MODE
   - MAKE CONNECTION TO SOMEONE ELSE.
Netstat

deemer@vesta ~/Development % netstat -an
Active Internet connections (including servers)

<table>
<thead>
<tr>
<th>Proto</th>
<th>Recv-Q</th>
<th>Send-Q</th>
<th>Local Address</th>
<th>Foreign Address</th>
<th>(state)</th>
</tr>
</thead>
<tbody>
<tr>
<td>tcp4</td>
<td>0</td>
<td>0</td>
<td>10.3.146.161.51094</td>
<td>104.16.248.249.443</td>
<td>ESTABLISHED</td>
</tr>
<tr>
<td>tcp4</td>
<td>0</td>
<td>0</td>
<td>10.3.146.161.51076</td>
<td>172.66.43.67.443</td>
<td>ESTABLISHED</td>
</tr>
<tr>
<td>tcp6</td>
<td>0</td>
<td>0</td>
<td>2620:6e:6000:900.51074</td>
<td>2606:4700:3108::443</td>
<td>ESTABLISHED</td>
</tr>
<tr>
<td>tcp4</td>
<td>0</td>
<td>0</td>
<td>10.3.146.161.51065</td>
<td>35.82.230.35.443</td>
<td>ESTABLISHED</td>
</tr>
<tr>
<td>tcp4</td>
<td>0</td>
<td>0</td>
<td>10.3.146.161.51055</td>
<td>162.159.136.234.443</td>
<td>ESTABLISHED</td>
</tr>
<tr>
<td>tcp4</td>
<td>0</td>
<td>0</td>
<td>10.3.146.161.51038</td>
<td>17.57.147.5.5223</td>
<td>ESTABLISHED</td>
</tr>
<tr>
<td>tcp6</td>
<td>0</td>
<td>0</td>
<td>*.51036</td>
<td>.*</td>
<td>LISTEN</td>
</tr>
<tr>
<td>tcp4</td>
<td>0</td>
<td>0</td>
<td>*.51036</td>
<td>.*</td>
<td>LISTEN</td>
</tr>
<tr>
<td>tcp4</td>
<td>0</td>
<td>0</td>
<td>127.0.0.1.14500</td>
<td>.*</td>
<td>LISTEN</td>
</tr>
</tbody>
</table>
Ports are part of the transport layer

- Port numbers are the first two fields of these headers! (Not part of IP!)
An interface to applications

- Ports define an interface to applications
- If you can connect to the port, you can (usually) use it!
- Problems?
Demo: netcat
Port scanning

What can we learn if we just start connecting to well-known ports?

- Can discover things about the network
- Can learn about vulnerabilities
Large-scale port scanning

• Can reveal lots of open/insecure systems!
• Examples:
  – shodan.io
  – VNC roulette
  – Open webcam viewers..
  – …
Disclaimer

• Network scanning is easy to detect

• Unless you are the owner of the network, it’s seen as malicious activity

• If you scan the whole Internet, the whole Internet will get mad at you (unless done very politely)

• Do NOT try this on the Brown network. I warned you.
Scanning I have done

- Scanned IPv4 space for ROS (Robot Operating System)
- Found ~200 “things” using ROS (some robots, some other stuff)