2015-10

Resilience of Deployed TCP to Blind Off-Path Attacks

Luckie, Matthew

http://hdl.handle.net/10945/50650

Downloaded from NPS Archive: Calhoun
Resilience of Deployed TCP to Blind Attacks

Matthew Luckie, Robert Beverly, Tiange Wu, Mark Allman, kc claffy

IMC 2015, October 28th 2015
What is a Blind Attack on TCP?

- A brute-force attempt by an **off-path attacker** to disrupt an in-progress TCP connection.

TCP connection: \(<A,B,x,y>\)

Off-path Attacker

[A,B,\(x+1\),\(y\),\(z\)]

[A,B,\(x\),\(y\),\(z\)]

[A,B,\(x\),\(y\),\(z+1\)]

[A,B,\(x+2\),\(y\),\(z+2\)]

(attack packets trying different combinations)
What is a Blind Attack on TCP?

• A brute-force attempt by an off-path attacker to disrupt an in-progress TCP connection

• Attack methods (RFCs 4953 and 5961):
  - RST attack: cause an existing TCP connection to be reset
  - SYN attack: cause an existing TCP connection to be reset
  - Data attack: cause an existing TCP connection to accept the attacker’s data, or enter an ACK war.

• Problematic with long-lived connections (e.g. BGP, SSH) and large windows (e.g. rsync)
History

• Paul Watson: CanSecWest 2004 “Slipping in the Window”
  - Showed feasibility of a blind reset attack. RFC 793 “a reset is valid if its sequence number is in the window.”
  - Larger receive windows reduce an attacker’s work.
  - Attacker must guess source and destination IP addresses, and source and destination ports of victim’s connections.

• Operating systems in 2004 chose ephemeral ports sequentially from a small range.
Slipping in the Window: RST or SYN

“a reset is valid if its sequence number is in the window”

- RFC 793

attacker’s blind RST and SYN packets

receive window

Theoretical receive window of 32k: up to $2^{17}$ packets.
Attacker constrained by network capacity.
Can complete in <1 second on 100Mbps Ethernet.

receive window

rcv.nxt

rcv.nxt + rcv.wnd

attacker’s successful in-window packet

$2^{32}$
Slipping in the Window: Data

“an acknowledgement value is acceptable as long as it is not acknowledging data that has not yet been sent”  
- RFC 793

receive window

0

rcv.nxt

rcv.nxt + rcv.wnd

snd.nxt

send window

2^{32}

acceptable ack range

acceptable acknowledgement values have a range of $2^{31}$ values, so only twice as hard as RST/SYN attacks
Defenses

- Choose ephemeral ports randomly! IETF BCP 156 (2011)
- Generalized TTL Security Mechanism (GTSM)
- TCP MD5 and Authentication Options
- Discard packets with spoofed source IP addresses at origin

RFC 5961, August 2010:
- strictly validate (challenge) the sequence number in RST and SYN packets
- reduce range of valid acknowledgement numbers in Data packets
RFC 5961 defenses: RST

*a reset is valid if the sequence number is exactly the next expected sequence number*

RFC 793:

```
0
```

```
| receive window
|-----------------
| rcv.nxt         |
```

RFC 5961:

```
0
```

```
| receive window
|-----------------
| rcv.nxt | rcv.nxt + rcv.wnd |
```

Difficulty increased to $2^{31}$ attempts (on average)
RFC 5961 defenses: RST or SYN

- **RST**: If the sequence number in a RST is in the window, receiver MUST send a **challenge ACK**
- **SYN**: Regardless of sequence number, send a **challenge ACK**
- **Challenge ACK purpose**: to elicit a reset with exact sequence number and confirm loss of connection

```
RST 11:-

ACK X:

rcv.nxt = 1
rcv.wnd = 64K
```

challenge ACK
RFC 5961 defenses: Data

*an acknowledgement number must fall in a smaller range*

RFC 793: snd.una - max.rcv.wnd

RFC 5961: send window

snd.nxt

acceptable ack range
What did we do?

- We implemented and used an oracle-based approach to test RFC 5961 support
  - Popular web-servers as a proxy for deployed TCP behavior of general purpose operating systems and middleboxes
  - Laboratory test of BGP routers and SDN switches
  - We tested sequence numbers in (+10) and out (-70,000) of receive window (Reset + SYN attacks)
  - We tested acknowledgement numbers behind (-70,000) and ahead (+70,000) of send window (Data attack)

- Evaluated range and strategy of OS ephemeral port selection:
  - Bro logs of communications to ICSI hosts 2005-2015
  - March 2015 Tier-1 backbone link packet trace
What did we find?

• September 2015, tested webservers:
  - 22% were vulnerable to blind reset and SYN packets
  - 30% were vulnerable to blind data packets
  - 38.4% were vulnerable to at least one attack vector

• Laboratory testing of 14 routers and switches
  - 12 were vulnerable to at least one attack vector (mostly blind data attack) that could impact BGP / SDN

• March 2015, 1 hour packet trace: most ephemeral ports were selected in a small range, 50% of predictable in a 2K range.

• 2005-2015: observed some evidence of an increase in ephemeral port range deployment
Testing resilience to blind reset attacks

This example shows RFC 5961 compliance

- in-window RST (b,e)
- challenge ACK (c,f)
## Blind reset and SYN results summary

**Testing ~41K webservers, randomly selected from Alexa 1M**

<table>
<thead>
<tr>
<th>Result</th>
<th>Blind Reset</th>
<th></th>
<th>Blind SYN</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>in</td>
<td>out</td>
<td>in</td>
<td>out</td>
</tr>
<tr>
<td>Accepted</td>
<td>3.4%</td>
<td>0.4%</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Reset (ack)</td>
<td>—</td>
<td>—</td>
<td>17.1%</td>
<td>0.0%</td>
</tr>
<tr>
<td>Reset (dup-ack)</td>
<td>18.8%</td>
<td>0.6%</td>
<td>5.3%</td>
<td>1.2%</td>
</tr>
<tr>
<td><strong>Vulnerable</strong></td>
<td><strong>22.2%</strong></td>
<td><strong>1.0%</strong></td>
<td><strong>22.4%</strong></td>
<td><strong>1.2%</strong></td>
</tr>
<tr>
<td>Challenge ACK</td>
<td>71.4%</td>
<td>1.1%</td>
<td>37.7%</td>
<td>57.0%</td>
</tr>
<tr>
<td>Ignored</td>
<td>5.1%</td>
<td>91.8%</td>
<td>35.9%</td>
<td>38.3%</td>
</tr>
<tr>
<td><strong>Not Vulnerable</strong></td>
<td><strong>76.5%</strong></td>
<td><strong>93.0%</strong></td>
<td><strong>73.6%</strong></td>
<td><strong>95.3%</strong></td>
</tr>
<tr>
<td>Parallel connection</td>
<td>—</td>
<td>—</td>
<td>1.1%</td>
<td>1.1%</td>
</tr>
<tr>
<td>Early FIN</td>
<td>0.3%</td>
<td>3.3%</td>
<td>1.5%</td>
<td>1.6%</td>
</tr>
<tr>
<td>No Result</td>
<td>1.0%</td>
<td>2.7%</td>
<td>1.3%</td>
<td>0.9%</td>
</tr>
<tr>
<td><strong>Other</strong></td>
<td><strong>1.3%</strong></td>
<td><strong>6.0%</strong></td>
<td><strong>4.0%</strong></td>
<td><strong>3.6%</strong></td>
</tr>
</tbody>
</table>

*Testing ~41K webservers, randomly selected from Alexa 1M*
Testing resilience to blind data attacks

Broke initial request into three pieces; sent third piece second with invalid acknowledgment.

---

1. **Client** sends **DATA 1:1(60)** to **Server**.
2. **Server** sends **ACK 1:61** to **Client**.
3. **Client** sends **DATA 121:-70000(62)** to **Server**.
4. **Server** sends **ACK 1:61** to **Client**.
5. **Client** sends **DATA 121:-70000(62)** to **Server**.
6. **Server** sends **ACK 1:61** to **Client**.
7. **Client** sends **DATA 121:-70000(62)** to **Server**.
8. **Server** sends **ACK 1:61** to **Client**.
9. **Client** sends **DATA 121:-70000(62)** to **Server**.
10. **Server** sends **ACK 1:61** to **Client**.
11. **Client** sends **DATA 61:1(60)** to **Server**.
12. **Server** sends **ACK 1:121** to **Client**.
13. **Client** sends **DATA 121:1(60)** to **Server**.
14. **Server** sends **ACK 1:183** to **Client**.
15. **Client** sends **DATA 121:1(60)** to **Server**.
16. **Server** sends **ACK 1:183(1460)** to **Client**.

**Notes:**
- **(a)** - **Client** sends **DATA 1:1(60)** to **Server**.
- **(b)** - **Client** sends **DATA 121:-70000(62)** to **Server**.
- **(c)** - **Client** sends **DATA 121:-70000(62)** to **Server**.
- **(d)** - **Client** sends **DATA 121:-70000(62)** to **Server**.
- **(e)** - **Client** sends **DATA 121:-70000(62)** to **Server**.
- **(f)** - **Client** sends **DATA 121:-70000(62)** to **Server**.
- **(g)** - **Client** sends **DATA 61:1(60)** to **Server**.
- **(h)** - **Client** sends **DATA 121:1(60)** to **Server**.
- **(i)** - **Client** sends **DATA 121:1(60)** to **Server**.
- **(j)** - **Client** sends **DATA 121:1(60)** to **Server**.

---

**Actions:**
- **Third piece invalid ACK**
- **Second piece**
- **Third piece w/ valid ack (if server’s ack did not cover it)**

---

**Summary:**
- The initial request is broken into three pieces, the third piece is sent second with an invalid acknowledgment.
- The server acknowledges the first and second pieces correctly.
- The third piece is acknowledged incorrectly, leading to a retransmission of the third piece.

---

**Diagram Notes:**
- **(2 Second Pause)** indicates a pause in the data transmission.
- **ACK** indicates an acknowledgment from the server.
- **DATA** indicates data sent from the client to the server.

---

**Concepts:**
- **Data transmission and acknowledgment**
- **Resilience to data attacks**
- **Network protocols**
Blind Data results summary

Testing ~41K webservers, randomly selected from Alexa 1M

<table>
<thead>
<tr>
<th>Result</th>
<th>Blind Data behind</th>
<th>ahead</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted</td>
<td>29.6%</td>
<td>5.4%</td>
</tr>
<tr>
<td>Reset (ack)</td>
<td>0.6%</td>
<td>0.6%</td>
</tr>
<tr>
<td>Reset (dup-ack)</td>
<td>0.1%</td>
<td>0.2%</td>
</tr>
<tr>
<td><strong>Vulnerable</strong></td>
<td><strong>30.3%</strong></td>
<td><strong>6.2%</strong></td>
</tr>
<tr>
<td>ACK</td>
<td>37.1%</td>
<td>8.1%</td>
</tr>
<tr>
<td>Ignored</td>
<td>29.3%</td>
<td>81.3%</td>
</tr>
<tr>
<td><strong>Not Vulnerable</strong></td>
<td><strong>66.4%</strong></td>
<td><strong>89.4%</strong></td>
</tr>
<tr>
<td>Parallel connection</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Early FIN</td>
<td>3.2%</td>
<td>3.7%</td>
</tr>
<tr>
<td>No Result</td>
<td>0.1%</td>
<td>0.7%</td>
</tr>
<tr>
<td><strong>Other</strong></td>
<td><strong>3.3%</strong></td>
<td><strong>4.4%</strong></td>
</tr>
</tbody>
</table>

5.4% accepted data with an ack value invalid in both RFC 793 and 5961
Evidence of Middlebox protection

see paper for full details

• TCP connections with an observed MSS of 1380
  - were almost never vulnerable to blind reset and SYN packets, but were vulnerable to blind data packets
  - sent challenge ACKs that arrived with a different TTL than other TCP packets in the flow
  - suggestive of middle-box protection
Ephemeral Port Selection

see paper for full details

• Goal was to evaluate port selection and range strategies

• Messy problem, no ideal set of data to examine trends with:
  - Packet captures observe subset of traffic from outside hosts
  - Hash-based port-selection (HBPS) could be confused with systems that select ports sequentially.
Ephemeral Port Selection

ICSI Bro Logs

Increase in 95th percentile range 2006 - 2008

Examined ranges of ports chosen over time (not selection strategy, due to sparseness)
Infrastructure testing results

see paper for full details

• Tested 14 BGP routers and OpenFlow switches
  - firmwares from 2004 to 2015
  - newer firmware generally does better in both ignoring packets that could have come from a blind attacker, as well as port selection strategies

• 12 were vulnerable to at least one attack
  - data injection attack is currently poorly addressed

• Implication: use GTSM and TCP MD5 where possible
Summary

• Paul Watson 2004 advice: strictly validate RST packets, choose ephemeral ports randomly

• September 2015: 38.3% of tested connections did not use best practices to reject TCP packets that could have come from off-path attacker

• Poor deployment of ephemeral port selection strategies in general population
  - Default behavior of Windows and MacOS is to choose TCP ephemeral ports sequentially

• TBIT tests for resilience to blind attacks available in scamper

http://www.caida.org/tools/measurement/scamper/
We inferred 38.4% of tested systems to be vulnerable to at least one of the three attacks in September 2015.
Oracle vs. Attacker

(a) Attacker Approach. **We do not do this.**

(b) Our Oracle Approach. We establish our own TCP connection and test response to packets that *could* have come from an attacker.
Largest Observed Window Size for Vulnerable Population

- 19.4% advertised ~16K
- 27.7% advertised ~8K
- 27.2% advertised > 64K
Ephemeral Port Selection

Tier-1 ISP Backbone Link

Cumulative Fraction

Range of Ephemeral Port Selection

predictable
N=138144

unpredictable
N=209738

49K – 64K
Ephemeral Port Ranges

<table>
<thead>
<tr>
<th>Port Range</th>
<th>Size</th>
<th>Operating System</th>
</tr>
</thead>
<tbody>
<tr>
<td>1024-5000</td>
<td>3976</td>
<td>Windows XP and earlier</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FreeBSD &lt;= 4.11 (Jan 2005)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Linux &lt;= 2.2</td>
</tr>
<tr>
<td>49152-65535</td>
<td>16384</td>
<td>FreeBSD &gt;= 5.0 (Jan 2003)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Windows Vista (Jan 2007)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Apple MacOS X</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Apple IOS</td>
</tr>
<tr>
<td>32768-61000</td>
<td>28232</td>
<td>Linux &gt;= 2.4</td>
</tr>
<tr>
<td>10000-65535</td>
<td>55535</td>
<td>FreeBSD &gt;= 8.0 (Nov 2011)</td>
</tr>
</tbody>
</table>
# MSS values observed

<table>
<thead>
<tr>
<th>Server MSS</th>
<th>Blind Reset</th>
<th>Blind SYN</th>
<th>Blind Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1460 (87.2%)</td>
<td>23.9%</td>
<td>24.7%</td>
<td>28.1%</td>
</tr>
<tr>
<td>1380 (5.4%)</td>
<td>2.0%</td>
<td>0.5%</td>
<td>58.8%</td>
</tr>
<tr>
<td>8961 (2.3%)</td>
<td>2.3%</td>
<td>2.3%</td>
<td>4.7%</td>
</tr>
<tr>
<td>1440 (0.8%)</td>
<td>5.9%</td>
<td>4.7%</td>
<td>57.5%</td>
</tr>
<tr>
<td>1436 (0.7%)</td>
<td>22.2%</td>
<td>5.8%</td>
<td>32.5%</td>
</tr>
</tbody>
</table>
## Blind attacks by inferred OS (p0f)

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Blind reset</th>
<th></th>
<th>Blind SYN</th>
<th></th>
<th>Blind data</th>
<th></th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>in</td>
<td>out</td>
<td>in</td>
<td>out</td>
<td>behind</td>
<td>ahead</td>
<td></td>
</tr>
<tr>
<td>FreeBSD 8.x</td>
<td>19.2%</td>
<td>0.5%</td>
<td><strong>93.8%</strong></td>
<td>56.5%</td>
<td><strong>83.9%</strong></td>
<td>None</td>
<td>0.5%</td>
</tr>
<tr>
<td>FreeBSD 9.x</td>
<td>18.8%</td>
<td>1.0%</td>
<td><strong>88.1%</strong></td>
<td>22.2%</td>
<td>54.7%</td>
<td>None</td>
<td>1.5%</td>
</tr>
<tr>
<td>Linux 2.4-2.6</td>
<td>87.4%</td>
<td>3.0%</td>
<td><strong>83.6%</strong></td>
<td>0.4%</td>
<td>54.3%</td>
<td>40.5%</td>
<td>0.6%</td>
</tr>
<tr>
<td>Linux 2.6.x</td>
<td><strong>90.1%</strong></td>
<td>0.9%</td>
<td>None</td>
<td></td>
<td></td>
<td></td>
<td>11.8%</td>
</tr>
<tr>
<td>Linux 3.x</td>
<td>15.3%</td>
<td>0.6%</td>
<td>14.0%</td>
<td>0.1%</td>
<td>11.6%</td>
<td>0.6%</td>
<td>43.4%</td>
</tr>
<tr>
<td>Windows 7/8</td>
<td>5.1%</td>
<td>2.1%</td>
<td>0.3%</td>
<td>0.3%</td>
<td><strong>88.7%</strong></td>
<td>0.9%</td>
<td>9.3%</td>
</tr>
<tr>
<td>Windows XP</td>
<td>7.9%</td>
<td>6.1%</td>
<td>3.0%</td>
<td>3.0%</td>
<td>6.3%</td>
<td>3.5%</td>
<td>2.0%</td>
</tr>
<tr>
<td>Unknown</td>
<td>9.6%</td>
<td>0.8%</td>
<td>12.7%</td>
<td>12.7%</td>
<td>23.9%</td>
<td>3.2%</td>
<td>30.2%</td>
</tr>
</tbody>
</table>
## Blind attacks by router/switch

<table>
<thead>
<tr>
<th>Device</th>
<th>OS date</th>
<th>Blind Reset</th>
<th>Blind SYN</th>
<th>Blind Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>in</td>
<td>out</td>
<td>in</td>
</tr>
<tr>
<td>C 2610</td>
<td>2002-01</td>
<td>✗</td>
<td>✔</td>
<td>✗</td>
</tr>
<tr>
<td>C 2610</td>
<td>2002-01</td>
<td>✗</td>
<td>✔</td>
<td>✗</td>
</tr>
<tr>
<td>C 2650</td>
<td>2005-08</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>C 7206</td>
<td>2008-07</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>C 2811</td>
<td>2010-10</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>C 2911</td>
<td>2012-03</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>J M7i</td>
<td>2007-01</td>
<td>✗</td>
<td>✔</td>
<td>✗</td>
</tr>
<tr>
<td>J EX9208</td>
<td>2014-06</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>J MX960</td>
<td>2015-05</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>J J2350</td>
<td>2015-05</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>HP 2920</td>
<td>2015-01</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>HP e3500</td>
<td>2015-06</td>
<td>✗</td>
<td>✔</td>
<td>✗</td>
</tr>
<tr>
<td>B MLX-4</td>
<td>2014-10</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Pica8</td>
<td>2015-05</td>
<td>✗</td>
<td>✔</td>
<td>✗</td>
</tr>
</tbody>
</table>