Locating Prefix Hijackers using LOCK

Tongqing Qiu\textsuperscript{+}, Lusheng Ji\textsuperscript{*}, Dan Pei\textsuperscript{*}
Jia Wang\textsuperscript{*}, Jun (Jim) Xu\textsuperscript{+}, Hitesh Ballani\textsuperscript{++}

\textsuperscript{+} College of Computing, Georgia Tech
\textsuperscript{*} AT&T Lab – Research
\textsuperscript{++} Department of Computer Science, Cornell University
Outline

• Background & Motivation
• System Architecture
• Basic algorithm and improvements
• Evaluation
• Conclusion
Background

• Autonomous System (AS)

• Border Gateway Protocol (BGP)

• Profit-driven policy
Background (cont.)

- BGP lacks authentication
- Fabricated AS announcement
- Prefix hijacking
  - blackholing
  - imposture
  - interception
State of Art

• Proactive
  – Prevent the happenings of hijacks
    • e.g. [Kent et al. JSAC 00] [Aiello et al. CCS 03], [Subramanian et al. NSDI 04], [Karlin et al. ICNP 06], etc.
  – Deployment issues:
    • Routing infrastructure modification
    • Difficulties of incremental deployment
    • PKI requirement

• Reactive
  – Detection
    • e.g. [Lad et al. Usenix Security 06], [Ballani et al. Sigcomm 07], [Zheng et al. Sigcomm 07], [Hu et al. IEEE S&P 07], [Zhang et al. Sigcomm 08], etc.
  – Recovery
    • e.g. [Zhang et al. CoNext 07]
A Complete and Automated Solution?

- Locating is important
  - Provide key information for recovery/mitigation
- Locating is not trivial
  - Current practice
    - Identify newly appeared origin AS of prefix $p$
System Architecture of LOCK

Input: Target prefix p

Output: A is the hijacker!
Key Components of LOCK

• Monitor Selection (from candidates)
  – Maximize the likelihood of observing hijacking events on the target prefix
  – Maximize the diversity of paths from monitors to the target prefix

• Locating Scheme
  – Using AS path information
  – Infer the hijacker location (how?)
Two key observations

- Countermeasure ability
  - The hijacker cannot manipulate the portion of AS path from a polluted vantage point to the upstream neighbor AS of the hijacker AS.
Two key observations

- Convergence: The trustworthy portion of polluted AS paths from multiple vantage points to a hijacked victim AS prefix converge around the hijacker AS (based on real AS topology).
Basic Locating Algorithm

• Identifying hijacker search space
  – Neighborset of one AS: ASes one-hop away (include itself)
  – Based on existing AS topology
  – The union of neighborset of all ASes on all polluted paths (why?)
  – The hijacker should be in the space (based on observation 1)

• Ranking all ASes in the search space
  – Based on observation 2
  – The more frequently an AS appears, the higher its ranking is
  – Tie breaker: The closer an AS to the monitors, the higher its ranking is
Basic Locating Algorithm Example

<table>
<thead>
<tr>
<th>Monitors</th>
<th>Polluted AS PATH</th>
<th>Neighbor Set</th>
<th>Hijacker List</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>A X</td>
<td>(A H) (H X Y)</td>
<td>H &gt; (4 times)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X &gt; Y &gt; (2 times)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A = B &gt; C (once)</td>
</tr>
<tr>
<td>M2</td>
<td>B X</td>
<td>(B H C) (H X Y)</td>
<td></td>
</tr>
</tbody>
</table>
Improvements

• Search space of basic algorithm
  – Trim the suspect list

• Improvement I: AS relationship
  – Basic algorithm neighborset
  – Valley free
  – Trim the neighborset on “trustworthy” ASes

• Improvement II: excluding “innocent” ASes

• Two improvements may introduce false negative
Evaluation

• Three sets of experiments:
  – Simulating synthetic prefix hijacking events
  – Reconstructed previous known hijacking events
  – Real prefix hijacking events
Simulating Synthetic Prefix Hijacking Events

• Hijacker $h$ and source $s$ from 73 Planetlab nodes
• 451 Target prefix $t$
  – Multiple Origin ASes (MOAS) prefix
  – Single Origin Ases with large traffic
  – Popular website (based on Alexa ranking)
• Emulate all possible hijacking events
  – Based on the combination of ($s$, $h$, $t$)
  – Imposture, interception, and malicious (countermeasure) cases
• Monitor selection
  – From Planetlab nodes
  – Based on the target prefix
Effectiveness and Improvement

- The accuracy of basic algorithm is 85%+
- Combine both improvements, the accuracy is up to 94.3%
- False negative ratio is relatively low.
Reconstruct Previously-known Hijacking Events

7 hijacking events
Locate all hijackers

<table>
<thead>
<tr>
<th>Victim AS</th>
<th>Hijacker AS</th>
<th>Date</th>
<th>#monitors</th>
</tr>
</thead>
<tbody>
<tr>
<td>3691</td>
<td>6461</td>
<td>March 15, 2008</td>
<td>16</td>
</tr>
<tr>
<td>36561 (YouTube)</td>
<td>17557</td>
<td>February 24, 2008</td>
<td>9</td>
</tr>
<tr>
<td>11643 (eBay)</td>
<td>10139</td>
<td>November 30, 2007</td>
<td>7</td>
</tr>
<tr>
<td>4678</td>
<td>17606</td>
<td>January 15, 2007</td>
<td>8</td>
</tr>
<tr>
<td>7018</td>
<td>31604</td>
<td>January 13, 2007</td>
<td>13</td>
</tr>
<tr>
<td>1299</td>
<td>9930</td>
<td>September 7, 2006</td>
<td>5</td>
</tr>
<tr>
<td>701, 1239</td>
<td>23520</td>
<td>June 7, 2006</td>
<td>12</td>
</tr>
</tbody>
</table>
Real Hijacking Events

Prefix: 204.9.168.0/22

Seattle

Berkeley

Pittsburgh

Cornell

victim

hijacker
### Real Hijacking Events (cont.)

<table>
<thead>
<tr>
<th>Victim Site</th>
<th>Hijacker Site</th>
<th>Launch Time (EST)</th>
<th>Response Time (minutes)</th>
<th>Required monitors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cornell</td>
<td>Berkeley</td>
<td>May 2 12:01:31</td>
<td>13</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>Seattle</td>
<td>May 2 16:12:47</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>Pittsburgh</td>
<td>May 2 17:34:39</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>Pittsburgh</td>
<td>Cornell</td>
<td>May 2 19:32:09</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>Berkeley</td>
<td>May 2 22:50:25</td>
<td>11</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>Seattle</td>
<td>May 3 02:26:26</td>
<td>12</td>
<td>15</td>
</tr>
<tr>
<td>Seattle</td>
<td>Cornell</td>
<td>May 3 11:20:42</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Pittsburgh</td>
<td>May 3 13:03:10</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>Berkeley</td>
<td>May 3 19:16:16</td>
<td>8</td>
<td>18</td>
</tr>
<tr>
<td>Berkeley</td>
<td>Seattle</td>
<td>May 3 22:35:07</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>Pittsburgh</td>
<td>May 4 00:01:01</td>
<td>12</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>Cornell</td>
<td>May 4 11:19:20</td>
<td>11</td>
<td>10</td>
</tr>
</tbody>
</table>
Conclusion

• LOCK to locate prefix hijacker ASes
  – First study of hijacker location problem
  – Locate the hijacker even when countermeasures are engaged
  – Extensively evaluation illustrates high location accuracy
Acknowledgement

• Authors Tongqing Qiu and Jun (Jim) Xu would like to acknowledge the generous support from the NSF CyberTrust program (specifically CNS 0716423)
• Thanks You!
• Questions