Practical Network Support for IP Traceback

Stefan Savage, David Wetherall, Anna Karlin, Tom Anderson

Presented by: Songjie Wei

Outlines

- Motivation of IP traceback
- Related work and their solutions
- Some definitions and assumptions of the marking approach
- Principles of marking approach
- Implementation issues
- Conclusion

Challenge of DoS Attacks

- The frequency and severity of DoS attacks increase every year.
- DoS attack is easy to implement, hard to prevent and difficult to trace.
- Two mechanisms to deal with DoS attacks
  - Tolerating attacks
  - Tracing back the attacks and stopping it

Tracing Back the Attacks

- A perfect solution – identify the true causal origin of the attack.
- A practical solution – identify the machines that directly generate attack traffic
  - Motivated by the operational need to control and contain attacks
  - Even incomplete or approximate information is valuable
  - Still very difficult to implement

Related Work: Ingress Filtering

- Eliminating the ability of forging source addresses
- Routers should block packets with illegitimate source addresses
- Drawbacks
  - Router needs sufficient power
  - Router needs sufficient knowledge
  - Wide deployment is necessary
  - Can’t detect forged addresses inside the domain

Related Work: Link Testing (I)

- Input debugging
  - Starting from the closest router, recursively find the upstream router along the loaded link
- Drawbacks
  - Management overhead
  - Only effective at tracing on-going attacks
Related Work: Link Testing (II)

- Controlled flooding
  - Victims have a map of Internet topology
  - Testing links by flooding them with large bursts of traffic
  - Observing how the attacking traffic changes
- Drawbacks
  - The flooding is another kind of DoS attack
  - It needs good knowledge of Internet topology
  - Not good for DDoS attack
  - Not good for bypass attacks

Related Work: Logging

- Logging packets at core routers
- Use data mining techniques to determine the traveling path of attacks
- Drawbacks
  - Enormous resource requirements
  - Database integration problem

Related work: ICMP Traceback

- By some ratio, copy the contents of packets and construct ICMP messages.
- Send these ICMP messages to the same destination.
- Victim can reconstruct a path back to the attacker with ICMP traceback messages
- Drawback
  - ICMP traffic tends to be rate-limited and filtered
  - Some routers may not have the ability of input debugging
  - Widespread is needed
  - Attackers may send false ICMP Traceback messages

Another Approach: Marking

- Probabilistically or deterministically, the routers mark the packets they transfer.
- Victim uses the information in the marked packets to trace an attack back to its source.
- Two components of the marking approach
  - Marking on the routers
  - path reconstruction on the victim
- Many advantage
  - No management overhead of input debugging
  - No network overhead
  - No significant router overhead
  - Good for attacks that have stopped

Some Definitions

- Victim node
- Attack origin
- Attack path
- Exact traceback
- Approximate traceback
- Convergence time
- Robustness

Basic Assumptions

1. The attacker may generate any packet
2. Multiple attackers may conspire
3. Attackers may be aware they are being traced
4. Packets may be lost or reordered
5. Attackers send numerous packets
6. The route between attacker and victim is fairly stable
7. Router are both CPU and memory limited
8. Routers are not widely compromised
Marking Algorithm: Node Append

Marking procedure at router $R$:
for each packet $w$, append $R$ to $w$

Path reconstruction procedure at victim $v$:
for any packet $w$ from attacker
extract path $(R_i, R_j)$ from the suffix of $w$

- Advantages
  - Robust
  - Fast to converge
- Disadvantages
  - High router overhead
  - Insufficient space in the packet for the marks

Marking Algorithm: Node Sampling

Marking procedure at router $R$:
for each packet $w$
  let $x$ be a random number from $[0, 1)$
  if $x < p$ then,
    write $R$ into $w$ node
Path reconstruction procedure at victim $v$:
let $NodeTM$ be a table of tuples (node, count)
for each packet $w$ from attacker
  $z = \text{lookup} w$ node in $NodeTM$
  if $z = \text{NIL}$ then
    increment $z$.count
  else
    insert tuple ($w$, node, 1) in $NodeTM$
  sort $NodeTM$ by count
  extract path $(R_i, R_j)$ from ordered node fields in $NodeTM$

- Advantages
  - Both time and space efficient
  - Very robust against single attacker
- Disadvantages
  - Path reconstruction is slow
  - Not robust against multiple attackers

Marking Algorithm: Edge Sampling

Marking procedure at router $R$:
for each packet $w$
  let $x$ be a random number from $[0, 1)$
  if $x < p$ then
    write $R$ into $w$.start and 0 into $w$.distance
  else
    if $w$.distance = 0 then
      write $R$ into $w$.end
    increment $w$.distance

Path reconstruction procedure at victim $v$:
let $G$ be a tree with root $v$
let edges in $G$ be tuples (start, end, distance)
for each packet $w$ from attacker
  if $w$.distance = 0 then
    insert edge $(w$.start, $v$, 0) into $G$
  else
    insert edge $(w$.start, $w$.end, $w$.distance) into $G$
remove any edge $(x, y, d)$ with $d < \text{distance from } v$ to $v$ in $G$
extract path $(R_i, R_j)$ by enumerating acyclic paths in $G$

Why we need the field distance and increase it at each router?
Marking Algorithm: Edge Sampling

- The convergence time depends on the time to receive a sample from the furthest router.
- The probability that a packet carries an edge: 
  \[ d \cdot p(1 - p)^{d-1} \]
- The number of marked packets required to get all the edges:
  \[ 1 + \frac{d}{d-1} + \frac{d}{d-2} + \ldots + d/d + \log(d) + O(1) \]
- Therefore, the convergence time (number of packets required to reconstruct the path) \( X \) is
  \[ X < \frac{\text{Mark}(d)}{p(1 - p)^{d-2}} \]

Advantages
- Using tree structure to detect multiple attacks
- The convergence time is linear function of attackers
- The algorithm is robust

Disadvantages
- Additional space required in IP packet header

Encoding of Edge Sampling

- 32+32+8 bits are needed in every IP packet
  - Simply storing the edge sample data in an IP option
  - Sending the edge data in a separate packet
- One possible solution given by the authors: Compressed edge fragment sampling

- Using exclusive-or (XOR) to combine start and end of an edge together

- Dividing each edge into several non-overlapping fragments

- Increase the size of each router address to 64 bits by using a well-known hash function

Bit Interleave

Bit-interleaving doesn’t eliminate the probability of collision.
Encoding of Edge Sampling

- Now less space is required in each packet

<table>
<thead>
<tr>
<th>offset</th>
<th>distance</th>
<th>Edge fragment</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>15</td>
</tr>
</tbody>
</table>

- Space overhead is reduced with the expense of larger computational overhead and relatively weaker robustness

Encoding of Edge Sampling

- Encoding edge fragments into the IP identification field

<table>
<thead>
<tr>
<th>use</th>
<th>flen</th>
<th>TOS</th>
<th>total length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Problems about using IP identification field
  - A datagram may be fragmented upstream from a marking router
    - No mark on the fragments
    - Use another probability $q$ to mark fragments, use an ICMP echo reply to replace the fragment and carry the edge information
  - A datagram may be fragmented downstream from a marking router
    - Set the Don’t Fragment flag on every marked packet

Conclusion

- DoS attacks motivate the development of IP traceback
- Several traceback algorithms based on packet marking
- Edge sampling enables efficient and robust multi-party traceback
- Edge sampling can be deployed by overloading IP header fields

Thank You

Questions & Comments???