Trace Back Overview

- Why do we need trace back?
  - Stop the attacker at the source
  - Hold the attacker accountable
Trace Back Overview (Cont’d)

• Classification
  – IP trace back
    • IP layer
  – Trace back through stepping stones
    • Transport/application layer

IP Trace Back
Approaches for IP Traceback

- **Ingress filtering**
  - Block the packets with illegitimate source addresses
  - Has to examine every packet
  - The router must have sufficient knowledge
    - Not feasible for traffic aggregated from multiple ISPs
  - Requires widespread deployment.
  - DOS attacks are still possible with a customer network.
Approaches for IP Track Back (Cont’d)

- **Link Testing**
  - Input debugging
    - Considerable management overhead
    - Only for ongoing attacks

1. signature
2. deploy signature
3. Find input port
4. signature
5. deploy signature
6. Find input port
7. signature

---

Approaches for IP Trace Back (Cont’d)

- **Link Testing**
  - Controlled flooding
    - Upstream router iteratively floods each incoming link
    - Observe change received from the attacker
    - It is a DOS attack itself
    - Require accurate topology map
    - Unable to trace distributed DOS attacks

1. notification
2. Controlled flooding
3. notification
4. Controlled flooding
5. notification
Approaches for IP Trace Back (Cont’d)

• Logging
  – Log packets at key routers
  – Use data mining techniques to determine the attack path
  – Only work after the attack
  – Storage requirement
  – Database integration problem

• ICMP trace back
  – Every router sample with low probability the forwarded packet.
  – Send selected information to the source or the destination in an ICMP traceback message.
    • Forward or backward link
  – Victim reconstructs the attack path using the above information
  – May be filtered out.
  – Requires authentication, and thus key distribution.
Approaches for IP Trace Back (Cont’d)

• Probabilistic packet marking
  – Probabilistically mark the packets with the routers’ addresses
  – Reconstruct the attack path using these addresses

• Potential advantages
  – No interactive cooperation [little management overhead]
  – No significant additional network traffic
  – Can be used both during and after attacks
  – Low overhead on routers.

Probabilistic Packet Marking

• Our discussion is limited to
Probabilistic Packet Marking (Cont’d)

- Attack path from $A_i$
  - The unique ordered list of routers between $A_i$ and $V$
- The problem
  - Exact traceback
    - Difficult, since the attacker may send false information.
  - Approximate traceback
    - Find the attack path that contains the true attack path as a suffix
- Marking procedure
- Reconstruction procedure
- Convergence time

Assumptions

- An attacker may generate any packet
- Multiple attackers may conspire
- Attackers may be aware they are being traced
- Packets may be lost or reordered
- Attackers send numerous packets
- The route between attacker and victim is fairly stable
- Routers are both CPU and memory limited
- Routers are not widely compromised
Marking Algorithm (1)

• Node Append
  – At each router $R$
    • For each packet $w$, append $R$ to $w$
  – At victim $v$
    • For any packet $w$ from attacker
      – Extract path $(R_i \ldots R_j)$ from the suffix of $w$

• Problems
  – High router overhead
  – Unbounded space requirement in the packets
    • Any reserved space may be filled up by the attacker.

Marking Algorithm (2)

• Node Sampling
  – Reserve one node field in the packet header
  – At each router $R$
    • For each packet $w$, write $R$ into $w$.node with probability $p$
  – At victim $v$
    • The order of the routers is not preserved in the order of received packets.
    • How to construct an ordered path?
Node Sampling (Cont’d)

- Assume all routers mark their addresses with equal probability $p$.
- What’s the probability that $v$ receives a packet marked by a router $d$ hops away?
  - 
- Observation:
  - The farther away $R$ is, the less chance $v$ receives packets marked by $R$.

Node Sampling (Cont’d)

- Reconstruction procedure at victim $v$
  - Count the number of packets marked by each router
  - Sort the routers by this count in increasing order
  - The ordered router list $\Rightarrow$ attack path.
- Limitations
  - Slow convergence
    - $d = 15$ and $p = 0.51$ $\Rightarrow$ need 42,000 packets
  - Cannot deal with multiple attackers
Marking Algorithm (3)

- **Edge Sampling**
  - Idea: put *edge* instead of *node* into the packets
  - Reserve *start*, *end*, and *distance* fields in packet header
    - (start, end): the marked edge (link)
    - distance: the distance from start to the victim

Edge Sampling (Cont’d)

- At each router $R$
  - For each packet $w$
    - With probability $p$, write $R$ into $w.start$ and 0 into $w.distance$
    - If $R$ doesn’t write $w.start$
      - If $w.distance = 0$ (i.e., the previous router just marked start), write $R$ into $w.end$
      - Increment $w.distance$
Edge Sampling (Cont’d)

- At victim $v$
  - Let $G$ be a tree with root $v$
  - Let edges in $G$ be tuples $(\text{start, end, distance})$
  - For each packet $w$ from attacker
    - If $w.distance = 0$ then
      - Insert $(w.start, v, 0)$ into $G$
    - Else
      - Insert edge $(w.start, w.end, w.distance)$ into $G$
      - Remove edges $(x, y, d)$ with $d \neq$ distance from $x$ to $v$ in $G$
        - Remove inconsistent edges.
      - Extract path $(R_i, \ldots, R_j)$ by listing acyclic paths in $G$.

Edge Sampling (Cont’d)

- Effectiveness
  - Can discern multiple attackers
  - Robust: impossible for any edge closer than the closest attacker to be spoofed
  - Number of packets needed to reconstruct all paths is linear in the number of attackers.

- Limitations
  - Require additional space in IP packet header
  - Two 32 bit IP addresses + 8 bit distance $\approx 72$ bits
Encoding

• Problem:
  – Where to save the edge samples?
• Idea
  – compress the edge sample and store it in the identification field (16 bits for fragmentation)
• Three techniques

Encoding (Cont’d)

• Technique 1: Send XOR of the two nodes of an edge.
Encoding (Cont’d)

• Technique 2: reduce per-packet space requirement by splitting each edge-id into $k$ fragments.
  – Each fragment is associated with $32/k$ bit data + $\log_2{k}$ bit offset
  – Reconstruct fragments with the same distance
    • Doesn’t work if there are multiple attack paths.

Encoding (Cont’d)

• Technique 3: Interleaving edge-id and its hash and then fragment
Coding (Cont’d)

- Techniques 3 (Cont’d): reconstruction

\[
\begin{align*}
\text{Bit Interleave} & \quad \text{Combine } k \text{ fragments} \\
0 & \quad k - 1 \\
\text{Edge id} & \quad \text{Hash (Edge id)} \\
\text{Hash} & \quad =? \\
\text{Y: accept} & \quad \text{N: reject}
\end{align*}
\]

Further Reading

- Dawn Song, Adrian Perrig: Advanced and Authenticatd Marking Schemes for IP Traceback, IEEE Infocom 2001