A MORE EFFICIENT HYBRID APPROACH FOR SINGLE-PACKET IP TRACEBACK

Yulong Wang, Sen Su, Yi Yang, Ji Ren

Beijing University of Posts and Telecommunications
Beijing, China
OUTLINE

- Introduction
- Related works
- Our Approach
  - Path fragment encoding scheme
  - Path recording procedure
  - Path recovery procedure
  - Overhead analysis
- Conclusion
INTRODUCTION

Why we need an IP Traceback mechanism

• To find attack source so as to punish the attacker or filter attack packets from the source
• source IP address is not trustworthy
  o the source address of IP packets can be easily spoofed
  o packets with spoofed source address can still reach the victims.
• memoryless of computer network
  o packets don’t carry their path information
  o routers don’t memorize which packets they had transferred
INTRODUCTION

- The main idea of IP Traceback
  - To enable the network memorize the path of packets it had transferred in peacetime (path recording phase)
  - Once IDS detects an attack, rebuild the attack path from the victim to the source using the stored information (path recovery phase)

- Two places to store the path information
  - the routers’ memory
  - the packets’ fields

- Three kinds of IP Traceback approaches
  - Marking-based
    - store path info. in packets’ fields
  - Logging-based
    - store path info. in routers’ memory
  - Hybrid
    - store path info. in both packets’ fields and routers’ memory
INTRODUCTION

Why we choose to propose a hybrid approach

- Marking-based approaches, such as PPM, are suitable for flooding attack traceback, but cannot trace single-packet attacks due to their probabilistic nature.
- Logging-based approaches, such as SPIE, can be used to trace single-packet attacks but incurs heavy overhead in storage and time.
- ISPs want to deploy one traceback system which can handle both flooding and single-packet attacks since both kinds of attacks can occur in their network.
- Thus, there is a need to devise a hybrid approach can trace single-packet (thus can trace flooding attack) but with a much less overhead.
RELATED WORKS

- hybrid approaches
  - DLLT (B. Al-Duwairi et al., TPDS’06)
    - keep track of a subset of the routers that are involved in forwarding a certain packets by establishing a temporary link between them using distributed link list.
    - routers probabilistically mark and store the forwarded packets, thus it cannot trace single packet
  - PPPM (B. Al-Duwairi et al., TPDS’06)
    - group the packets with same destination addresses into a flow and let these packets pick up path fragments which are left in the router by previous packets belonging to the same flow
    - Routers probabilistically mark packets, thus it cannot trace single packet
RELATED WORKS

- hybrid approaches
  - HIT (C. Gong et al., TPDS’08)
    - for each packet passing through the routers, the routers mark it deterministically but log it alternately.
    - Decrease storage overhead by ½ compared with SPIE, thus is more suitable for both single-packet and flooding attack
OUR APPROACH

Our Goal
- Maintain the single-packet traceback capability of HIT
- Reduce the overhead incurred by HIT so as to cope with flooding attack more efficiently

The main contribution is
- to reduce the overhead to 2/3 of HIT in both storage and time for recording packet paths
- to reduce the time overhead for recovering packet paths by a calculatable amount.
OUR APPROACH

- Walsh matrix based Hybrid IP Traceback
- The main idea of WHIT is
  - using short IDs (15 or 24 bits) to denote routers instead of IP addresses (32 bits)
  - assign globally unique IDs to large-degree routers;
  - assign IDs to small-degree routers in such a way that their IDs are orthogonal to their small degree neighbors’ IDs
  - In the path recording phase, routers store as many IDs as possible in packets’ fields and dump them in routers’ memory once the packets’ fields are full
  - In the path recovery phase, attack paths can be rebuilt by using the orthogonality of IDs and the neighbor information of routers
OUR APPROACH

- Path fragment encoding scheme
  - a Walsh matrix is a specific square matrix with the property that any two distinct rows are orthogonal.

Walsh matrix
- $W(0,8) = 1, 1, 1, 1, 1, 1, 1, 1$
- $W(1,8) = 1,-1, 1,-1, 1,-1, 1,-1$
- $W(2,8) = 1, 1,-1,-1, 1, 1,-1,-1$
- $W(3,8) = 1,-1,-1, 1, 1,-1,-1, 1$
- $W(4,8) = 1, 1, 1, 1,-1,-1,-1, 1$
- $W(5,8) = 1,-1, 1,-1,-1, 1,-1, 1$
- $W(6,8) = 1, 1,-1,-1,-1,-1, 1, 1$
- $W(7,8) = 1,-1,-1, 1,-1, 1, 1,-1$

- Based on the Walsh matrix, we designed an encode scheme which helps put more IDs in a packet.
OUR APPROACH

- How many IDs can we put in a packet
  - the length of the marking field in the packet
    - the maximum length is 38 bits
      - the 16-bit identification
      - the 1-bit fragmentation flag
      - the 8-bit type of service
      - the 13-bit fragment offset
    - the actual length, L, is determined by
      \[
      L = \left( \left\lfloor \log_2 C \right\rfloor + 1 \right)2^k + \left\lfloor \log_2 C \right\rfloor + 1 \leq 38
      \]
      where \( C \) is the capacity of the marking field, \( 2^k \) is the dimension of the Walsh matrix.
OUR APPROACH

- How many IDs can we put in a packet
  - the router’s degree
    - to distinguish router_i from its neighbors, the marking field of a packet should contain at least three IDs of consecutive routers, thus \( C \geq 3 \)
    - Given \( C \geq 3 \), the number of candidate router IDs and the average router degree should conform to the following inequality.

\[
2^k \geq (C - 2) \lfloor \bar{D} \rfloor + 1
\]

- the average router degree is 6.34 according to ITDK0304 skitter data measured by CAIDA
OUR APPROACH

- How many IDs can we put in a packet
  - the result
    - max \( C \)
    - s.t. \( ([\log_2^C + 1] + 1)2^k + [\log_2^C + 1] \leq 38 \)
    - \( 2^k \geq (C - 2)[\tilde{D}] + 1 \)
    - \( C \geq 3 \)

- \( C \) is 3.57, thus we can put three IDs in a packet
- \( k \) is 3, thus we need \( 2^3 = 8 \) candidate IDs
OUR APPROACH

- Orthogonal router IDs based on Walsh matrix

<table>
<thead>
<tr>
<th>Walsh matrix</th>
<th>Router IDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>$W(0,8) = 1, 1, 1, 1, 1, 1, 1$</td>
<td>ID1 = 001 001 001 001 001 001 001 001</td>
</tr>
<tr>
<td>$W(1,8) = 1, -1, 1, -1, 1, -1, 1, -1$</td>
<td>ID2 = 001 101 001 101 001 101 001 101</td>
</tr>
<tr>
<td>$W(2,8) = 1, 1, -1, -1, 1, 1, -1, 1$</td>
<td>ID3 = 001 001 101 101 001 001 101 101</td>
</tr>
<tr>
<td>$W(3,8) = 1, -1, -1, 1, 1, -1, 1, -1$</td>
<td>ID4 = 001 101 101 001 001 101 101 001</td>
</tr>
<tr>
<td>$W(4,8) = 1, 1, 1, -1, -1, -1, 1, 1$</td>
<td>ID5 = 001 001 001 001 101 101 101 101</td>
</tr>
<tr>
<td>$W(5,8) = 1, -1, 1, -1, -1, 1, -1, 1$</td>
<td>ID6 = 001 101 001 101 101 001 101 001</td>
</tr>
<tr>
<td>$W(6,8) = 1, 1, -1, -1, -1, -1, 1, 1$</td>
<td>ID7 = 001 001 101 101 101 101 001 001</td>
</tr>
<tr>
<td>$W(7,8) = 1, -1, -1, 1, -1, 1, -1, 1$</td>
<td>ID8 = 001 101 101 001 101 001 001 101</td>
</tr>
</tbody>
</table>

The values of elements in $W(i,8)$ are $\{-1, 1\}$;
The values of elements in $W(i,8) + W(j,8)$ are $\{-2, 0, 2\}$;
The values of elements in $W(i,8) + W(j,8) + W(k,8)$ are $\{-3, -1, 0, 1, 3\}$;
Each element in the sum can only be one of 7 values which are $\{-3, -2, -1, 0, 1, 2, 3\}$, that means we only need 3 bits to represent one element in Router IDs.
OUR APPROACH

Marking information
- 24-bit path fragment
  - For large-degree routers, the path fragment contains a 15-bit globally unique router ID.
  - For small-degree routers, the path fragment contains the sum of several orthogonal router IDs, which is 24 bits in length.
- 2-bit path fragment length
  - record the number of small-degree IDs in the path fragment field
- 1-bit path fragment type
  - If it’s 0 and the path fragment length is 0, then the path fragment doesn’t contain any router ID.
  - If it’s 0 and the path fragment length is greater than 0, then the path fragment contains the sum of the small-degree router IDs.
  - If it’s 1, then the path fragment contains a large-degree router ID.

Marking field
- the 16-bit identification field
- 11 bits of the 13-bit fragment offset field
OUR APPROACH

- Path recording procedure
  - logging when
    - packet contains large-degree router ID
    - current router is a large-degree router
    - packet contains 3 small-degree router IDs
  - otherwise
    - marking packet with current router ID
  - forward the packet

```
Let d be the ID number the current router R
FOR each packet P
  IF p.path_fragment_type = 1
    or R is a large-degree router
    or p.path_fragment_length = 3 THEN
      Compute and store the digest of p
      p.path_fragment = 0
      p.path_fragment_type = 0
      p.path_fragment_length = 0
    ELSE
      p.path_fragment += R.d
      p.path_fragment_length += 1
    IF R is a large-degree router THEN
      p.path_fragment_type = 1
    ELSE
      p.path_fragment_type = 0
```
OUR APPROACH

- Path recovery procedure
  - pinpoint the nearest router to the victim

Once R1 is identified, the traceback server can restore the attack packet to the form when it reached R1 by either deleting ID of R1 from the packet’s marking field or restore the packet from R1’s digest table.
OUR APPROACH

- Path recovery procedure
  - if path fragment type = 1
    - recover 1 large-degree ID

```
010101010101010

attack packet
path fragment type: 1
path fragment: 0101010101010101010

neighbor list of R1
ID                IP
010101010101010  a.b.c.d
... ...
```
OUR APPROACH

- Path recovery procedure
  - if path fragment type = 0
    - if path fragment length > 0
      - recover up to 3 IDs using orthogonality of small-degree IDs

\[
\begin{align*}
001001101101 & 001101001101 001001001001 \\
001001101101 & 001101001101 001001001001 \\
\end{align*}
\]

\[
\begin{align*}
\text{attack packet} \\
\text{path fragment type: 1} \\
\text{path fragment length: 3} \\
\text{path fragment: 011001001101101101101101} \\
\end{align*}
\]

\[
\begin{align*}
<011, 001, 001, 101, 011, 001, 101, 101> \\
<001, 001, 001, 001, 001, 001, 001, 001> \neq 0
\end{align*}
\]
OUR APPROACH

- Path recovery procedure
  - if path fragment type = 0
    - if path fragment length = 0
      - recover a router ID by examining the digest tables

![Diagram of path recovery process](image)
OUR APPROACH

- **Compatibility**
  - IP fragmentation
    - record the mapping between the original and fragmented packets
    - logging fragmented packets in the following routers
  - IP transformation
    - NAT, IP in IP tunnel, IPsec etc.
    - record the mapping between the original and transformed packets
    - treat the transformed packet as an ordinary packet in the following routers on the path
ANALYSIS

Path recording overhead

- 5 types of packets will be logged at routers
  - type1: IP fragment
    \[ P_1 = \alpha \]
  - type2: non-fragmented packets transformed at this router
    \[ P_2 = (1 - \alpha)\beta \]
  - type3: non-fragmented untransformed packets with large-degree router ID
    \[ P_3 = (1 - \alpha)(1 - \beta)(1 - \gamma)(1 - P_l) \]
  - type4: non-fragmented untransformed packets with small-degree router IDs and this router is a large-degree router
    \[ P_4 = (1 - \alpha)(1 - \beta)(1 - P_l)(1 - \gamma)\gamma \]
  - type5: non-fragmented untransformed packets with small-degree router IDs and this router is a small-degree router but path fragment length = 3
    \[ P_5 = (1 - \alpha)(1 - \beta)(1 - P_l)^3\gamma^4 \]
ANALYSIS

the percentage of logged packets is

\[ P_l = \sum_{i=1}^{5} P_i, \quad 0 \leq \alpha, \beta, \gamma, P_l \leq 1 \]

Theorem 1: \( P_l \) is an increasing function of \( \alpha \)
- proof:

Let \( \alpha = 1 - \delta, \beta = 1 - \epsilon, \gamma = 1 - \lambda, P_l = 1 - Y \)

\[ \delta \epsilon (1 - \lambda)^4 Y^3 + (1 + \delta \epsilon \lambda (2 - \lambda)) Y - \delta \epsilon = 0 \]

\[ A \frac{\partial Y}{\partial \delta} + B = 0 \quad A = 3 \delta \epsilon (1 - \lambda)^4 Y^2 + \delta \epsilon \lambda (2 - \lambda) + 1 \]

\[ B = \epsilon ((1 - \lambda)^4 Y^3 + \lambda (2 - \lambda) Y - 1) \]

\[ \frac{\partial Y}{\partial \delta} \geq 0 \quad \frac{\partial (P_l)}{\partial \alpha} \geq 0 \]

Theorem 2: \( P_l \) is an increasing function of \( \beta \)
ANALYSIS

- Measurement studies show
  - $\alpha \leq 0.25\%$ and $\beta \leq 3\%$
  - $\gamma = 97.54\%$ (CAIDA, ITDK1104)

- The range of $P_l$
  - $\min(P_l) = 0.3199$, when $\alpha = \beta = 0$
  - $\max(P_l) = 0.3288$, when $\alpha = 0.0025$, $\beta = 0.03$

- Compared with HIT
  - $0.5 \leq P_{\text{HIT}} \leq 0.51$, thus $P_{\text{WHIT}} \approx \frac{2}{3} P_{\text{HIT}}$
ANALYSIS

- Path Recovery Overhead
  - The querying time is composed of two parts
    - those spent on finding next-hop routers in the attack path
      - Suppose an attack path has $h$ hops, and each router on the attack path has $n$ neighboring routers on average
      - WHIT needs to dispatch $h/3$ rounds of queries, querying $n-1$ routers in each round, totaling up to $(n-1)h/3$ routers
      - HIT need to query $(n-1)h/2$ routers
      - WHIT reduces $1/3$ of this part of time

\[ T_W = \frac{2}{3} \times T_H \]
ANALYSIS

Path Recovery Overhead

- The querying time is composed of two parts
  - those spent on examining packets in the digest tables

The time interval covered by the digest table is

\[ t = \frac{s \times r}{u} \]

in which, \( s \), \( r \) and \( u \) denote the digest table’s size, memory efficiency factor and packet digest writing rate.

Since \( u_w = \frac{P_w}{P_h} \times u_h \), \( T \) requires a lower rate to write packet digests into digest tables than HIT, the large-size DRAM digest tables are suitable for WHIT but not for HIT.

Thus, \( s_w = c \times s_h \ (c \geq 1) \)
ANALYSIS

- Path Recovery Overhead
  - The querying time is composed of two parts
    - those spent on examining packets in the digest tables

We have \( t_w = \frac{c \times P_h}{P_w} \times t_h \)

For query time period \( \Delta t \), we have

\[
ND_w = n \times \left\lfloor \frac{\Delta t}{t_w} \right\rfloor \\
ND_h = n \times \left\lfloor \frac{\Delta t}{t_h} \right\rfloor = n \times \left\lfloor \frac{c \times P_h}{P_w} \times \frac{\Delta t}{t_w} \right\rfloor
\]

Since \( \frac{c \times P_h}{P_w} > 1 \)

we have \( ND_w \leq ND_h \)
DISCUSSION

- the visibility of the mechanism
  - attackers can determine the existence of the mechanism by examining if the More Fragments (MF) flag of the received packet is cleared but its ID field is not zero
  - force edge routers to commit logging operations
- Partial deployment differs in two aspects
  - the degree of a traceback enabled router is calculated by counting its neighbors on the traceback overlay network
  - if the packet is fragmented or transformed at an ordinary router, then the packet cannot be traced back to its origin since the mapping information between the changed packet and its original form is lost
DISCUSSION

- how to extend WHIT to suit IPv6 network
  - field selection
    - there is no identification field in the IPv6 header
    - Flow Label is not commonly used but too short (20 bits)
    - Hop-by-Hop Options Header is a candidate

- deal with attacks of botnets
  - WHIT can identify the collection of zombies
  - With the help of IDS, WHIT can identify the control server of the botnet so as to find the real attacker behind the zombies
CONCLUSION

- A more efficient IP traceback approach
  - Reduced 1/3 of storage overhead for path recording
  - Reduced time overhead for path recovery
- Compatible to current IP protocol
  - IP fragmentation
  - IP transformation
- Can be easily extended to IPv6 network
THE END

Thanks!

Any question?