Active Timing-Based Correlation of Perturbed Traffic Flows with Chaff Packets

Pai Peng, Peng Ning, Douglas Reeves
North Carolina State Univ.

Xinyuan Wang
George Mason Univ.
Attack Through Stepping Stones
Attack Trace-back

• Stepping stone *connection* chain:
  
  \[ h_1 \leftrightarrow h_2 \leftrightarrow \ldots \leftrightarrow h_n \]

• Stepping stone *flows*:
  
  \[ h_1 \leftrightarrow h_2 : h_1 \rightarrow h_2 \text{ and } h_2 \leftarrow h_1 \]
  
  \[ i < j, \ h_i \rightarrow h_{i+1} \text{ is called an *upstream* flow of } h_j \rightarrow h_{j+1}, \]
  
  \[ \text{and } \ h_j \rightarrow h_{j+1} \text{ is called a *downstream* flow of } h_i \rightarrow h_{i+1} \]

• Trace back problem:
  
  \[ \text{Given an upstream flow, to identify its downstream flows.} \]
Attack Trace-back (cont’d)

• Countermeasures:
  – Content encryption
  – Timing perturbations
  – Extra padding packets: *Chaff*
Related Work

• Correlation based on packet contents
  – Thumb-printing
  – Sleepy watermark tracing

• Correlation based on timing characteristics
  – On/off periods
  – Deviation based
  – Watermark scheme based on Inter-packet delay (IPD) quantization
  – Multi-scale
  – Comparing the numbers of packets in the flows
Related Work (cont’d)

• Probabilistic watermark scheme
  – Embed watermark through slightly adjusting packet timing
  – Inter-packet-delay (IPD) of packet $p_j$ and $p_{j+d}$ is:
    $$\text{ipd} = t_{j+d} - t_j$$
  – Randomly construct $2r$ IPDs and divide them into 2 groups: $\text{ipd}^1$ and $\text{ipd}^2$, the average difference between IPDs in group 1 and 2 is:
    $$D = \frac{1}{2r} \sum_{i=1}^{r} (\text{ipd}_{i}^{1} - \text{ipd}_{i}^{2})$$
  – $E(D) = 0$
Probabilistic Watermarking (cont’d)
Probabilistic Watermarking (cont’d)

• Embed watermark
  – Embed bit 1: increase $D$
    • Increase IPDs in the 1$^{\text{st}}$ group, and
    • Decrease IPDs in the 2$^{\text{nd}}$ group.
  – Embed bit 0: decrease $D$

• Decode watermark
  – Check whether $D > 0$ or $D \leq 0$

• Robust to timing perturbation, but not chaff
  – Must known the location of watermark
Related Work (cont’d)

• Zhang et al.:
  – Finding possible matching packets
  – Different correlation schemes aiming at timing perturbation or/and chaff packets
    • Scheme S-IV
Proposed Approach

• Adopt probabilistic watermarking
  – Encode is ok, need to change decode

• Basic idea:
  – Find possible matching packets
  – Decode watermarks from all possible matching flows.
  – Use the “best” watermark that has the smallest hamming distance to the original watermark to determine correlation result.
  – Can detect any flow that probabilistic watermark scheme can.

• Assumptions:
  – No packet loss/merge through stepping stone connections.
  – The delays between corresponding packets are bounded by a maximum delay $\Delta$ (*timing constraint*).
  – The orders of packets are kept the same (*order constraint*).
Matching Packets

For each packet $p_i$ in the upstream flow $f$, we find all its possible matching packets in the suspicious flow $f'$:
- Matching set: $M(p_i) = \{ p_j' | 0 \leq t_j' - t_i \leq \Delta \}$
- Matching sets may overlap
Decoding the “Best” Watermark

- **Brute-force algorithm**
  - high computation cost

- **Greedy algorithm**: choose the packets that are most likely to produce the desired watermark.
  
  - **Pros:**
    - Low
    - Good detection rate
  
  - **Cons:**
    - High false positive rate
    - Brute-force algorithm: high computation cost
    - Greedy algorithm: choose the packets that are most likely to produce the desired watermark.

- **Pros:**
  - Matching Packets of $P_i$
  - Matching Packets of $P_{i+d}$

- **Cons:**
  - Smallest IPD
  - Largest IPD

Time
Decoding the “Best” Watermark (cont’d)

- Use *Greedy* algorithm to filter out the watermark bits that will not match.
- Carefully construct a flow satisfying the order constraint, and decode a watermark $w_b$.
- Gradually improve $w_b$ by switching to other matching packets
  - *Greedy*+: using heuristics
    - Adjust the watermark bit that has the smallest IPD difference $D$ first
    - Cannot affect the bits that are already matched
  - *Greedy**: enumerate all possible combinations of matching packets
Experimental Evaluation

- Compare the detection rates, false positive rates and computation costs of Greedy, Greedy+, Greedy*, probabilistic watermarking, and scheme S-IV.
- Using both real flows and synthetic flows.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\Delta)</td>
<td>0, 1, 2, 3, 4, 5, 6, 7, 8 (second)</td>
</tr>
<tr>
<td>(\lambda_c)</td>
<td>0, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5</td>
</tr>
<tr>
<td>Watermark</td>
<td>24 bits</td>
</tr>
<tr>
<td>Redundancy</td>
<td>4</td>
</tr>
<tr>
<td>WM threshold</td>
<td>7</td>
</tr>
<tr>
<td>WM delay</td>
<td>600ms</td>
</tr>
<tr>
<td>S-IV threshold</td>
<td>3 seconds</td>
</tr>
</tbody>
</table>
Detection Rate

Figure 1: Detection rate changing with $\lambda_c$, $\Delta = 7s$

Figure 2: Detection rate changing with $\Delta$, $\lambda_c = 3$
False Positive Rate

Figure 3: False positive rate changing with $\lambda_c$, $\Delta = 7s$

Figure 4: False positive rate changing with $\Delta$, $\lambda_c = 3$
Computation Cost: Correlated Flows

Figure 5: Computation costs changing with $\lambda_c$, $\Delta = 7s$, correlated flows

Figure 6: Computation costs changing with $\Delta$, $\lambda_c = 3$, correlated flows
Computation Cost: Uncorrelated Flows

Figure 7: Computation costs changing with $\lambda_c$, $\Delta = 7s$, uncorrelated flows

Figure 8: Computation costs changing with $\Delta$, $\lambda_c = 3$, uncorrelated flows
Conclusion

- A correlation scheme that can deal with both timing perturbation and chaff packets
- Different algorithms to achieve the best performance in terms of detection rate, false positive rate and computation cost.
- Through experimental evaluation, Greedy+ has shown the best result.
Thank you!

• Questions?