Towards Language-Based Mitigation of Traffic Analysis Attacks

Jeppe Fredsgaard Blaabjerg  
jfblaa@cs.au.dk

Aslan Askarov  
aslan@cs.au.dk

Aarhus University
Is there a message from Alice to Specialist?
Observable Properties of Communication

- **Message count**
- **Message size**
- **Message recipient**
- **Message time**
Mitigating Traffic Analysis

* System level mitigation
  * Black-box
  * Enforcing constant rate/padding on all communication

* X. Fu, B. Graham, R. Bettati, W. Zhao, and D. Xuan, “Analytical and empirical analysis of countermeasures to traffic analysis attacks,” 2003 International Conference on Parallel Processing,
Our Approach

- Language-based mitigation
  - Program source to control traffic shape
  - Information flow control to enforce that traffic shape does not leak secrets
- Fixed-size packets
  - Messages encoded as sequences of packets
  - Packet contents protected by encryption

Programmatic control over traffic shape
SELENE: Language and Runtime

• Simple imperative language with I/O
• Labelled channels – assuming a single channel per level $\ell$
• Runtime support for scheduling and queueing messages

$$e ::= n \mid x \mid e \text{ op } e$$

$$c ::= x = e \mid c_1; c_2 \mid \text{skip} \mid \text{if } e \text{ then } c_1 \text{ else } c_2 \mid \text{while } e \text{ do } c \mid \text{sleep } (e)$$

| $x = \text{input } (\ell)$ | (* input from channel $\ell$ *) |
| $x = \text{sizeof } (e)$ | (* compute runtime size of an expression *) |
| $\text{schedule } (\ell, e_1, e_2)$ | (* schedule packets on channel $\ell$ *) |
| $\text{queue } (\ell, e)$ | (* queue message on channel $\ell$ *) |

**queue** $(\ell, v)$ places value $v$ into the output buffer for channel $\ell$
- splits the value into packets as necessary

**schedule** $(\ell, n, t)$ allocates $n$ packets to be sent on channel $\ell$ starting after time delay $t$
Schedule/Queue semantics

- Corner case 1
  - Packets scheduled but nothing queued - send dummy packets

- Corner case 2
  - Packets queued but not scheduled - buffer until schedule is set

- Schedule is globally deterministic
  - The semantics keeps track of time counter $t$

<table>
<thead>
<tr>
<th>Time</th>
<th>100</th>
<th>101</th>
<th>102</th>
<th>103</th>
<th>104</th>
<th>105</th>
<th>106</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel</td>
<td>Alice</td>
<td>Alice</td>
<td>Bob</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Schedule config at $t=99$

<table>
<thead>
<tr>
<th>Time</th>
<th>100</th>
<th>101</th>
<th>102</th>
<th>103</th>
<th>104</th>
<th>105</th>
<th>106</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel</td>
<td>Alice</td>
<td>Alice</td>
<td>Bob</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Instruction at $t=100$

- Schedule (Charlie, 4, 0)
  - Schedule four packets for Charlie with no time delay

<table>
<thead>
<tr>
<th>Time</th>
<th>100</th>
<th>101</th>
<th>102</th>
<th>103</th>
<th>104</th>
<th>105</th>
<th>106</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel</td>
<td>Charlie</td>
<td>Alice</td>
<td>Alice</td>
<td>Charlie</td>
<td>Bob</td>
<td>Charlie</td>
<td>Charlie</td>
</tr>
</tbody>
</table>

Schedule config at $t=100$
size_result = sizeof(result);
schedule(Doctor, size_result, 5);

size_id = sizeof(id);
schedule(Specialist, size_id, 100);

queue(Doctor, result);

needs_treatment = input(Doctor);

if needs_treatment
    then queue(Specialist, id);
else skip;
Type System

\[
\text{T-IF} \\
\Gamma \vdash e : \texttt{int}@\ell \quad \Gamma, pc \sqcup \ell \vdash c_1 : pc' \quad \Gamma, pc \sqcup \ell \vdash c_2 : pc'' \\
\hline \\
\Gamma, pc \vdash \text{if } e \text{ then } c_1 \text{ else } c_2 : pc' \sqcup pc''
\]

\[
\text{T-SCHEDULE} \\
\begin{array}{ll}
pc = \bot & \Gamma \vdash e_1 : \texttt{int}@\bot & \Gamma \vdash e_2 : \texttt{int}@\bot \\
\hline \\
\Gamma, pc \vdash \text{schedule}(\ell, e_1, e_2) : pc
\end{array}
\]

\[
\text{T-QUEUE} \\
\Gamma \vdash e : \sigma_e@\ell_e \quad \ell_e \sqcup pc \sqsubseteq \ell \\
\hline \\
\Gamma, pc \vdash \text{queue}(\ell, e) : pc
\]
Medical Referral Re-revisited

\[
\begin{align*}
\text{size\_result} &= \text{sizeof} (\text{result}); \\
\text{schedule} (\text{Doctor}, \text{size\_result}, 5); \\
\text{queue} (\text{Doctor}, \text{result}); \\
\text{needs\_treatment} &= \text{input} (\text{Doctor}); \\
\text{if} \ \text{needs\_treatment} \ \text{then} \ {\{} \\
&\quad \text{size\_id} = \text{sizeof} (\text{id}); \\
&\quad \text{schedule} (\text{Specialist}, \text{size\_id}, 5); \\
&\quad \text{queue} (\text{Specialist}, \text{id}); \\
&\{ \text{else skip}; \\
\end{align*}
\]
Security Condition

\[ k(cfg, \tau, \ell) = \{ cfg' | cfg \approx_{\ell} cfg' \land cfg' \rightarrow^{*}_{\tau}, cfg'' \land \tau \approx_{\ell} \tau' \} \]

Attacker knowledge\(^1\)

Soundness theorem

- Well-typed SELENE programs do not leak by their output

\[ k(cfg, \tau \cdot \alpha, \ell) \supseteq k(cfg, \tau, \ell) \]

Security condition

SELENE Limitations

• Scheduling requires low pc
  • Progress-sensitive type system leads to pc-creep
• Possible approaches for mitigating pc-creep
  • Extra precision in the static reasoning
  • Declassification
    • Value declassification (e.g., before branching)
    • PC-declassification\(^2\)

\(^2\) J. Bay and A. Askarov, “Reconciling progress-insensitive noninterference and declassification,” CSF 2020
Towards Language-Based Mitigation of Traffic Analysis Attacks

/ Takeaways

• Traffic analysis is a significant concern
  • IFC models should reflect that

• Language-based solutions
  • Potential to reduce overhead compared to black-box techniques

• Future research
  • New language designs for mitigating traffic analysis

Thank you!

jfblaa@cs.au.dk