Remote IP Protection Using Timing Channels

Ariano-Tim Donda\textsuperscript{1,2}  Peter Samarin\textsuperscript{1,2}  Jacek Samotyja\textsuperscript{1}  Kerstin Lemke-Rust\textsuperscript{1}  Christof Paar\textsuperscript{2}

\textsuperscript{1}Bonn-Rhein-Sieg University of Applied Sciences, Germany  
\textsuperscript{2}Ruhr University Bochum, Germany  

December 4, 2014
Motivation: IP Protection Using Watermarks

Suspicious device

- “Trial” binary/bitstream is used in production
- *Given a system: is it my software/bitstream?*
- -> Insert a watermark into the IP

- Challenge: bitstream and binary are encrypted
Motivation: Embedding Watermarks in Side Channels

- **Motivation:**
  - **Side channel leakage:** power, EM, timing
  - **Regular input:**
  - **Regular output:**
  - **Unintentional:**
  - **Microcontroller (μC) / FPGA**

**Problems:**
- Special equipment necessary
- Measurements must be done in proximity to the device

**Sequence generating circuit**

**Leakage circuit**

(Becker et al. 2010)
This Work: Watermarks in the Timing Channel

1. Definition of the timing channel
2. Embedding watermarks in the timing channel
3. Case study: FPGA implementation
4. Measurements: remote and near-field
Definition: The Timing Channel

Timing attacks on AES (Bernstein, 2005)

Sender (using binary method)
- Using the regular data channel:
  - Delays the output by some short time when sending a 1
  - No delay when sending a 0

Receiver (using binary method)
- Using the regular data channel:
  - Observes time differences between input and output: $\Delta_t$
  - Compute $\overline{\Delta_t}$ by observing many $\Delta_t$-s
  - Decode to 1 if $\Delta_t \geq \overline{\Delta_t}$
  - Decode to 0 if $\Delta_t < \overline{\Delta_t}$

Assumptions
- Known or observable input
- Observable output
The Timing Channel: An Example

Send binary sequence: "0110"

- Timing channel can be used as a black box to send any kind of data

*ICISC 2014*
Watermarks

- **Authorship watermarks**
  - Is used to identify the owner of IP
  - Always visible
  - Codeword scheme
  - Challenge response scheme

- **Fingerprint watermarks (Easter egg watermarks)**
  - Hidden most of the time
  - Becomes visible when the owner enters the right passphrase
  - Challenge response scheme
Authorship Watermarks: Codeword Scheme

Verifier $\mathcal{V}$

Codeword $\mathbf{c}_{CW} = (c_{n-1}, \ldots, c_0)_2$

\[
i = 0; \quad \text{suc} = 0;
\]

Trigger $f$ and start timer: $t_s$

Stop timer: $t_e$

\[
\Delta_t = t_e - t_s;
\]

\[
w' = (b', l') = \text{rcvTC}(\Delta_t);
\]

If $(\text{cmp}(\mathbf{c}, i, w') = \text{true})$ \quad $\text{suc} = \text{suc} + 1$;

\[
i = (i + l) \mod n;
\]

Function $f$

Codeword $\mathbf{c}_{CW} = (c_{n-1}, \ldots, c_0)_2$

\[
i = 0;
\]

\[
\text{sndTC}(c, i);
\]

\[
i = (i + l) \mod n;
\]

\[
\text{rcvTC}(\Delta_t);
\]

\[
\text{cmp}(\mathbf{c}, i, w') = \text{true} \quad \text{suc} = \text{suc} + 1;
\]

\[
i = (i + l) \mod n;
\]
Authorship Watermarks: Challenge Response Scheme

Verifier $\mathcal{V}$

<table>
<thead>
<tr>
<th>Secret key $k$</th>
</tr>
</thead>
</table>

- Generate random input $c$
- Trigger $f$ and Start timer: $t_s$

<table>
<thead>
<tr>
<th>$c$</th>
</tr>
</thead>
</table>

Function $f$

| Secret key $k$ |

- $t = E_k(c)$
- $l = \text{sndTC}(t,0)$

- Stop timer: $t_e$

$\Delta_t = t_e - t_s$

$w' = (b', l') = \text{rcvTC}(\Delta_t)$

$t' = E_k(c)$

If $(\text{cmp}(t', 0, w') = \text{true})$ $\text{suc} = \text{suc} + 1$;
Fingerprint Watermarks: Challenge Response Scheme

Verifier $\mathcal{V}$

Secret Key List $[k]$

Generate random input $c$

Trigger $f$

Choose $k' \in [k]$

$c = E_{k'}(r)$

Trigger $f$ and Start timer: $t_s$

Stop timer: $t_e$

$\Delta_t = t_e - t_s$

$w' = (b', l') = \text{rcvTC}(\Delta_t)$;

If $(b' = 1)$ Return$(k')$;

Function $f$

Secret Key $k$

Generate random input $c$

\[ \cdots \]

\[ c \] \rightarrow

\[ \cdots \]

Generate random output $r$

\[ t = E_k(r) \]

\[ \cdots \]

\[ r \] \rightarrow

\[ \cdots \]

If $(c = t)$ $\text{sndTC}(1,0)$;

Generate random output $r$

\[ t = E_k(r) \]

\[ \cdots \]

\[ r \] \rightarrow

\[ \cdots \]
Proof of Concept: CV Application on an FPGA

Camera (100 fps) → FIFO → Ethernet controller → UDP packets to a fixed IP address → FPGA

FPGA → Ethernet controller → Packet capture (using pcap library) → Image reconstruction → PC

Threshold = 127
Proof of Concept: Embedding the Codeword Scheme

- Codeword initialized a circular shift register
- Delays are introduced right before finalizing packets
- PC: packet time stamp inspection to compute $\Delta t$ between two consecutive timestamps
Proving of Concept: Embedding the Challenge Response Scheme

- Challenge response scheme using Trivium with a fixed key
- Use binarized image as a seed value for Trivium
- PC: Compute Trivium stream cipher seeded by received thresholded image
Experiments: Remote Measurement

- FPGA and PC separated by two routers and three switches in the department network of BRSU
- Compare received data with ground truth

<table>
<thead>
<tr>
<th>Timing delays (μs)</th>
<th>Error rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.5047</td>
</tr>
<tr>
<td>20</td>
<td>0.3440</td>
</tr>
<tr>
<td>40</td>
<td>0.2682</td>
</tr>
<tr>
<td>60</td>
<td>0.2521</td>
</tr>
<tr>
<td>80</td>
<td>0.0936</td>
</tr>
<tr>
<td>100</td>
<td>0.0953</td>
</tr>
<tr>
<td>120</td>
<td>0.0583</td>
</tr>
</tbody>
</table>

- Advantages
  - low cost—no extra equipment necessary
  - can be done remotely
Experiments: Measuring in Proximity
Near-field experiments
- Direct Ethernet cable measurement
- EM measurement of Ethernet controller
- EM measurement at the FPGA
- Power traces

Delays of two clock cycles are visible
Can recover the whole watermark without noise
Possible Attacks

- **Reverse engineering the binary/bitstream**
  - no tools publicly available for RE of FPGA bitstreams
  - if tools are available (SW), a complete RE to remove all timing dependencies is hard work
  - better to write from scratch!

- **Wrapper attack**
  - timing-normalizing wrapper to equalize all $\Delta_t$
  - countered by sending several bits at a time (the sliding window approach)
  - increasing the delay decreases the operability of the wrapper
  - EM measurements still can reveal what the code does
Summary

- Timing channel definition
- Watermarks in the timing channel
- Proof-of-concept implementation on an FPGA

- Advantages
  - remote verification
  - low-cost solution

- Future work
  - Robust $\mu$C implementation
  - Fingerprint watermark implementation
  - Less obvious timing channel
    - Use only every 10th I/O pair (for example)
  - Verification over the Internet