Remote IP Protection Using Timing Channels

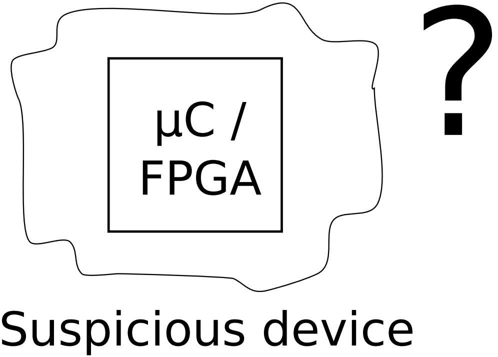
Ariano-Tim Donda[1,2] Peter Samarin[1,2] Jacek Samotyja[1]
Kerstin Lemke-Rust[1] Christof Paar[2]

[1]Bonn-Rhein-Sieg University of Applied Sciences, Germany

[2]Ruhr University Bochum, Germany

December 4, 2014

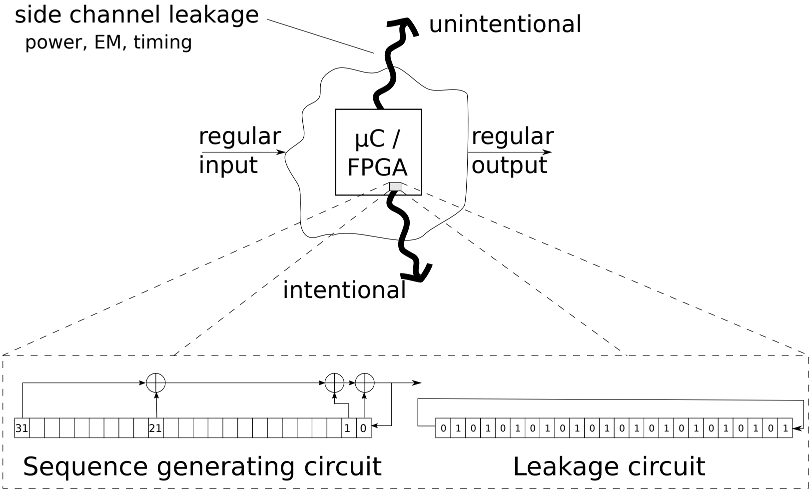
Motivation: IP Protection Using Watermarks



- “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



(Becker et al. 2010)

- Problems:
  - special equipment necessary
  - measurements must be done in proximity to the device

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

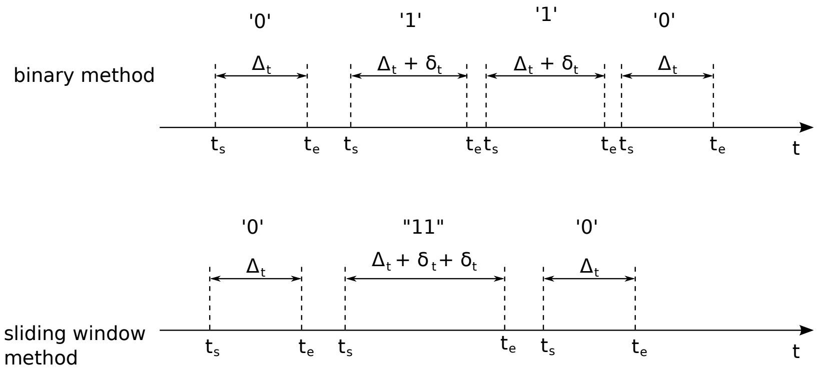
- Timing attacks on AES (Bernstein, 2005)

- 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: Δ_t
  - Compute $\overline{\Delta_t}$ by observing many Δ_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

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 $\mathcal{V}$		Function f
Codeword $c_{CW} = (c_{n-1}, \ldots, c_0)_2$ $i = 0; suc = 0;$		Codeword $c_{CW} = (c_{n-1}, \ldots, c_0)_2$ $i = 0;$
Trigger f and start timer: t_s	$\longrightarrow$	
		$\ldots$ $l = \text{sndTC(c,i)};$ $i = (i + l) \bmod n;$ $\ldots$
Stop timer: t_e $\Delta_t = t_e - t_s;$ $w' = (b', l') = \text{rcvTC}(\Delta_t);$ If $(cmp(c, i, w') = true)$ $suc = suc + 1;$ $i = (i + l) \bmod n;$	$\longleftarrow$	

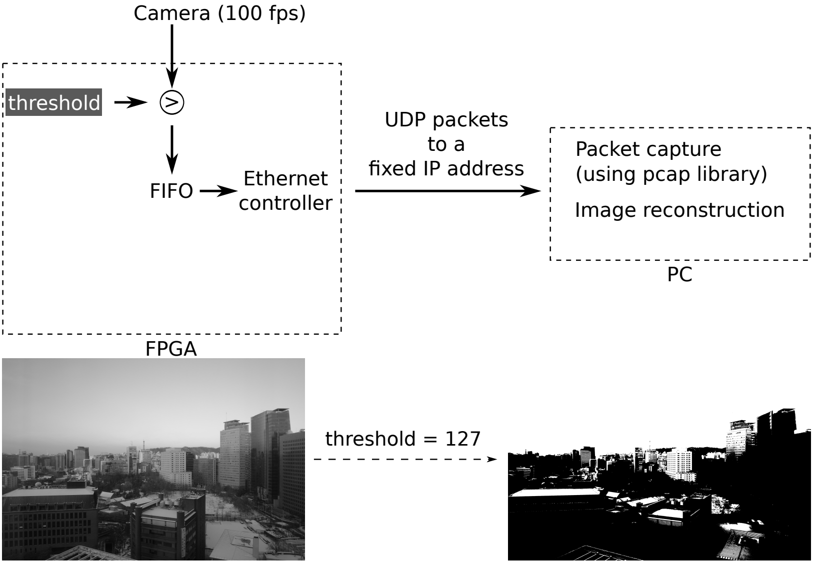
Authorship Watermarks: Challenge Response Scheme

Verifier $\mathcal{V}$		Function f
Secret key k		Secret key k
Generate random input c		
Trigger f and Start timer: t_s	$\xrightarrow{c}$	
		... $t = E_k(c)$ $l = \text{sndTC(t,0)};$ ...
Stop timer: t_e $\Delta_t = t_e - t_s$ $w' = (b', l') = \text{rcvTC}(\Delta_t);$ $t' = E_k(c)$ If $(cmp(t', 0, w') = true)$ $suc = suc + 1;$	$\longleftarrow$	

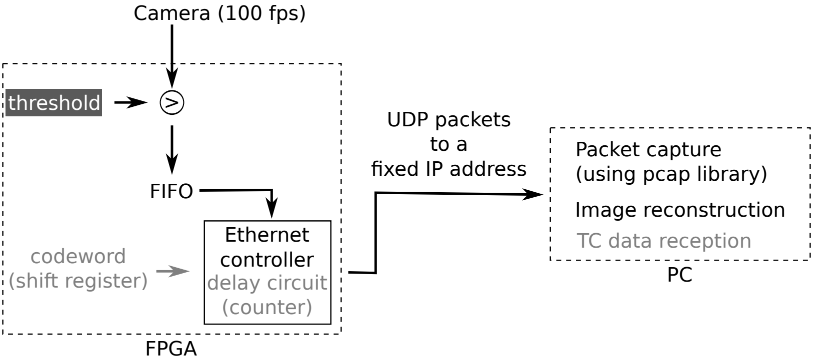
Fingerprint Watermarks: Challenge Response Scheme

Verifier $\mathcal{V}$		Function f
Secret Key List [k]		Secret Key k
Generate random input c		
Trigger f	c →	
		…
		Generate random output r
		$t = E_k(r)$
		…
	← r	
Choose $k' \in [k]$		
$c = E_{k'}(r)$		
Trigger f and Start timer: t_s	c →	
		…
		If $(c = t)$ sndTC(1,0);
		Generate random output r
		$t = E_k(r)$
		…
	← r	
Stop timer: t_e		
$\Delta_t = t_e - t_s$		
$w' = (b', l') =$ rcvTC(Δ_t);		
If $(b' = 1)$ Return(k');		

Proof of Concept: CV Application on an FPGA

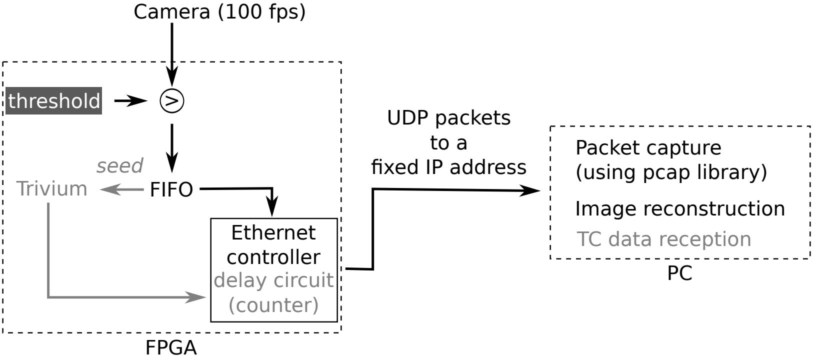


Proof of Concept: Embedding the Codeword Scheme



- Codeword initialized a circular shift register
- Delays are introduced right before finalizing packets
- PC: packet time stamp inspection to compute Δ_t between two consecutive timestamps

Proof 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

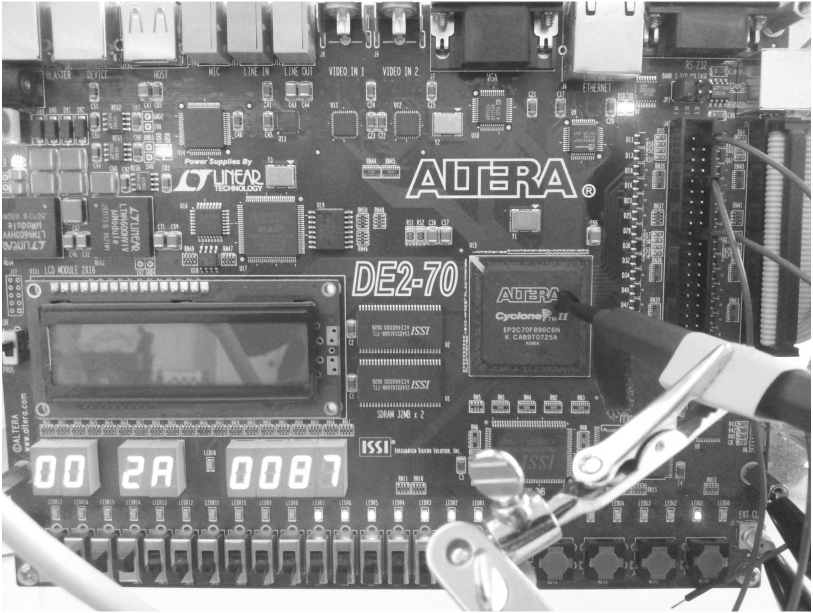
Experiments: Remote Measurement

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

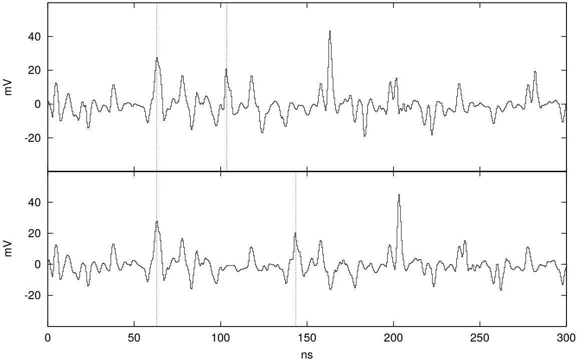
Timing delays (μs)	Error rate
0	0.5047
20	0.3440
40	0.2682
60	0.2521
80	0.0936
100	0.0953
120	0.0583

- Advantages
  - low cost—no extra equipment necessary
  - can be done remotely

Experiments: Measuring in Proximity



Experiments: Measuring in Proximity (contd.)



- 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 Δ_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 μC implementation
  - Fingerprint watermark implementation
  - Less obvious timing channel
    - Use only every 10th I/O pair (for example)
  - Verification over the Internet