High-performance FLOSS tooling for DPA

Ilya Kizhvatov
Digital Security group

Radboud Universiteit

Joint work with Cees-Bart Breunesse (Riscure North America)

CRYPTACUS workshop, Nijmegen, 2017-11-17

Main points

- In many applications, attack time and not the number of traces is the ultimate metric
- W.r.t. speed, free open-source DPA tooling is on par with industry standard
- Experimental tool written in Julia allows for easy parallelisation

Smartcard vs Embedded

- Smartcard world:
  high security, limit on the number of crypto operations. SCA metric - **#traces**
- Embedded (IoT) world:
  low to moderate security, no trace limit (think encrypted firmware, communication layer, whitebox). SCA metric - **time to perform the attack**

Academia vs Real life

- Academia
  - compare SCA to SCA (#traces, SR, GE, MI, …)
- Real life
  - compare SCA to other attacks (time, expertise, cost, …, but not #traces)

Existing FLOSS DPA tooling

2007: **OpenSCA**. Not maintained. **MATLAB**

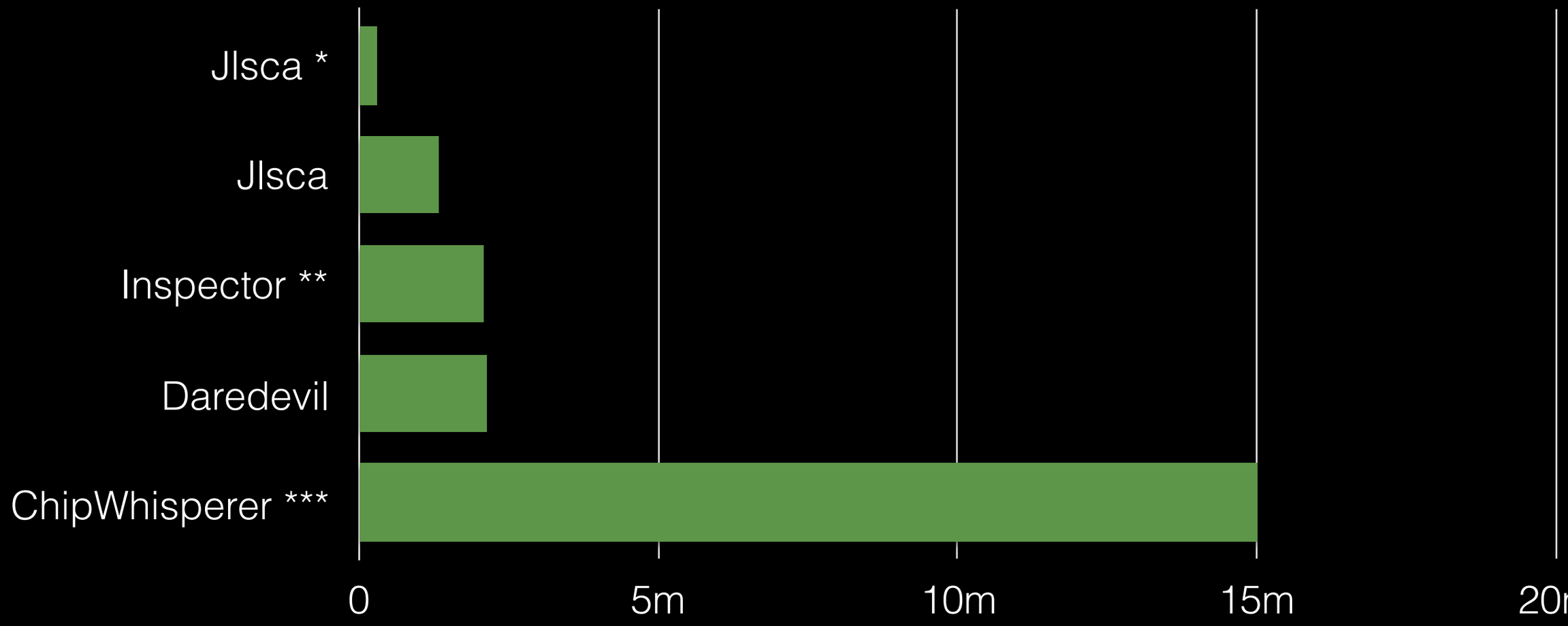
2012: **ChipWhisperer.** HW+SW, SCA+FI. **Python**

2016: **Daredevil**. 1- and 2-order CPA. **C++**

2016: **Jlsca**. CPA, LRA and more. **Julia**

+ Key enumeration and rank estimation tools
+ DPA contest (v1 implementations published)
+ Cache attack tools
+ Some lone repos on GitHub

Performance classical CPA



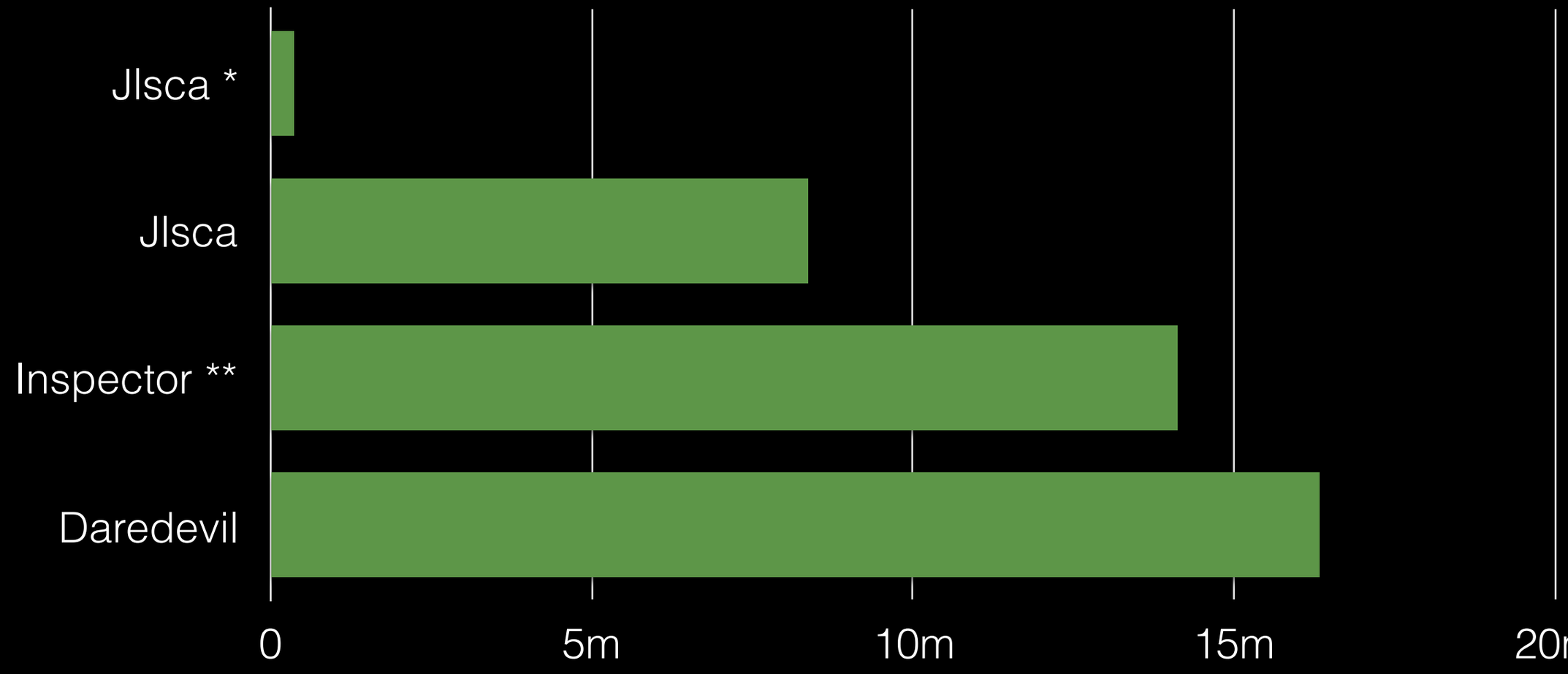
* conditional leakage averaging
** industry standard for reference
*** “C-accelerated” implementation

Target: AES-128 S-box out, Hamming weight
Dataset: 100K traces of 512 float32 samples (200 MB)
Platform: a modest dual-core laptop

https://github.com/ikizhvatov/dpa-tools-benchmarking

Radboud Universiteit

Performance all-bit AS-CPA



* conditional leakage averaging
** industry standard for reference

Target: AES-128 S-box out, all-bit AS-CPA
Dataset: 100K traces of 512 float32 samples (200 MB)
Platform: a modest dual-core laptop

https://github.com/ikizhvatov/dpa-tools-benchmarking

Radboud Universiteit

Some other features

	ChipW.	Dared.	Jlsca	Inspector
Acquisition	+			+
Public key attacks	+			+
Template attacks	+			+
TVLA				+
Advanced trace preprocessing				+
Cluster capable			+	+
GUI	+			+
Command line script	+	+	+	

Jlsca

- https://github.com/Riscure/Jlsca, GPLv3
- started from a toolbox in Python, rewritten and extended in Julia (for parallelism)
- can run on clusters (with one extra config line)
- usage: script / REPL / notebook
- supports trace formats from other tools

Tutorials with traces

- https://github.com/ikizhvatov/jlsca-tutorials
  - Working with various trace formats
  - Classical DPA and LRA
  - DCA on whitebox
  - Running on the cluster
  - DPA on HMAC-SHA1