Side-channel attacks against white-box cryptography on Android

Michael Eder

Technische Universität München
Department of Informatics
Chair for IT-Security

Fraunhofer AISEC

6.7.2016

Contents

Goals

- reproduce results on possibility of side-channel attacks against white-box cryptography
- Apply attack against white boxes on Android

What are black and white boxes in a cryptographical context? (1)

- Classical black-box approach: Attacker can only observe input and output
- Intermediate values are stored and processed without any protection against environmental attackers
- Ability to observe/modify execution environment $\rightarrow$ Key can be read/recovered

What are black and white boxes in a cryptographical context? (2)

- White-box approach: Create an implementation resistant against attacker in white-box context
- academic schemes mostly use keys obfuscated into further protected tables
- en-/decryption are done via table lookups, key is never stored or processed in plaintext
- Useful for Digital Rights Management and Payment applications
- Also useful for protecting keys in memory, e.g. in order to replace dedicated security hardware or protecting TLS certificate's/password manager's master secrets

Common problems

- Code Lifting
- white-box inversion
- size/performance
- all academic schemes broken
- Side-channel attacks!

DCA and DFA against white-box cryptography

- idea behind DCA: Observe accessed memory addresses and contents
- Addresses can be used to visualize the trace and detect the white-box
- content of memory can be used for attacking the white-box with a DPA-like attack called DCA

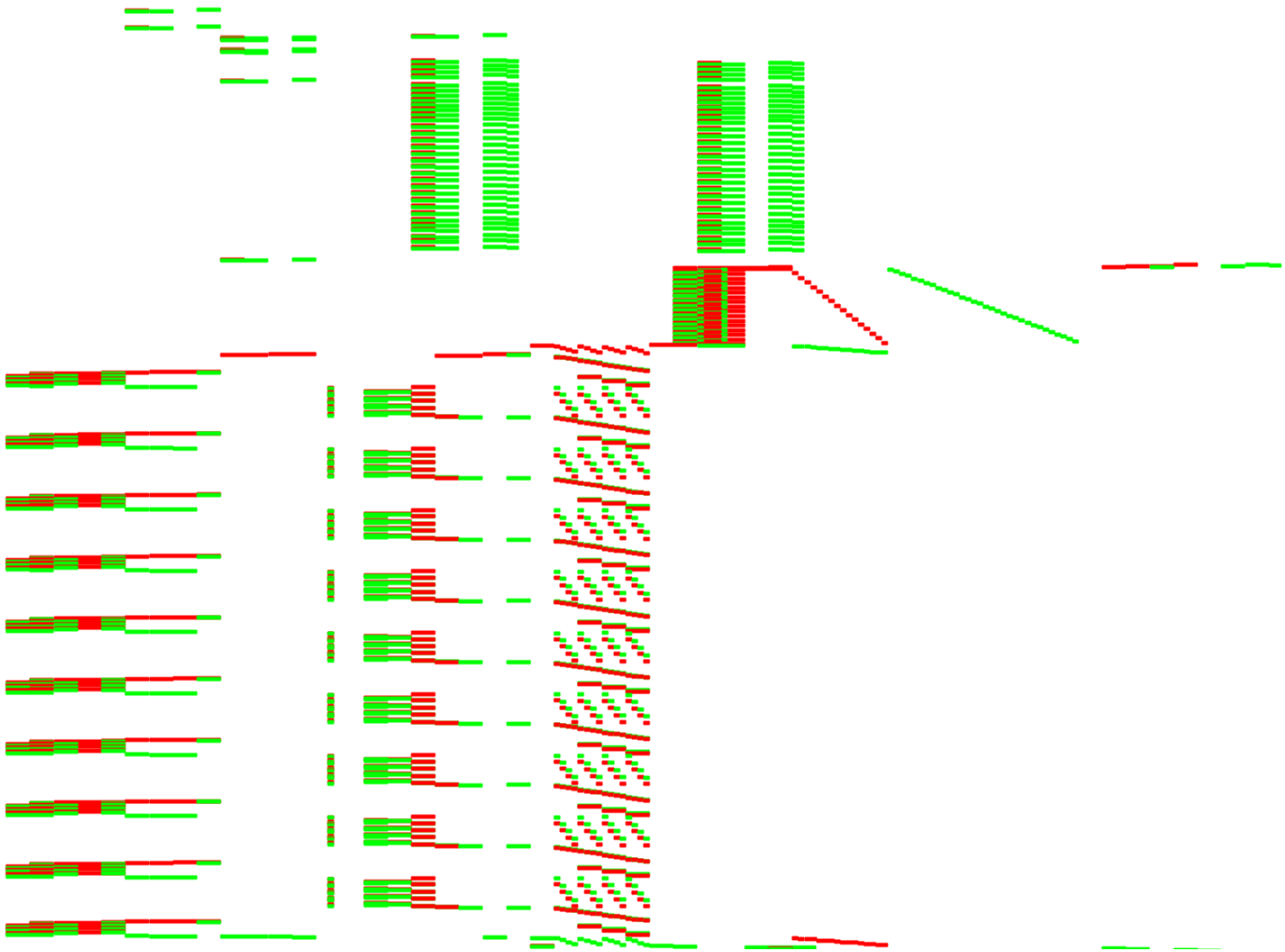
DCA and DFA against white-box cryptography

- idea behind DCA: Observe accessed memory addresses and contents
- Addresses can be used to visualize the trace and detect the white-box
- content of memory can be used for attacking the white-box with a DPA-like attack called DCA
- idea behind DFA: Inject faults into execution, analyze error propagation in order to learn about the key
- Nearly all public white-box implementations can be broken without greater reverse engineering by applying these attacks
- Required traces for current public white-boxes: $\sim$2000 worst case (which is very few)

Side-channel attack successful

- memory tracing and visual identification of white boxes reproduced with self-written tools
- Key recovery reproduced for public white boxes with SCAMarvels tool chain (described later)

Hack.lu2009 CTF challenge, visualized memory trace of white box execution



Search for Android applications containing white boxes

- Scan of a local app archive containing more than 18.000 unique apk files
- few additional DRM media and payment apps crafted by hand
- Shared libraries were searched for buzzwords like `whitebox` or names from white-box vendor products
- No further reverse engineering or analysis since attack is supposed to work without deep knowledge of the white box

Tracing native libraries on Android with Valgrind

- target architecture: Android on ARMv7a
- tracing regular binaries like coreutils works fine
- tracing android applications requires few workarounds
- Some problems: bad performance, some instructions unknown to Valgrind

Tracing Java binaries on Linux

- Extremely slow
- Traces polluted with unrelated data
- unpredictable memory behaviour
- Better approach (future work): modify JVM or use a Java DBI framework

SCAMarvels and our Docker container image

- SCAMarvels: complete, free tool chain for analysis of white boxes
- plugins for Valgrind and PIN to trace memory
- data visualization tool
- DCA attack tool, supporting AES and DES
- DFA attack tool, supporting AES
- Pool containing various public white-boxes, write-ups to analyze and attack them, ready-to-use attack scripts, further literature
- Our contribution: Docker container image containing portable working environment

Conclusion

- Attack reproducible
- low-level DBI approach not efficient against Java
- DBI on Android is currently not that easy because of lack of tools
- Free tool chain available to analyze and attack white boxes and to automate and scale required steps
- Conclusion: white-box cryptography feasible for short term-keys, but more secure academic schemes required