RUNTIME CODE GENERATION TO SECURE DEVICES
Damien Couroussé | CEA Grenoble

Workshop Interdisciplinaire sur la Sécurité Globale,
Paris 14 et 15 sept. 2017
ANR INS 2013. 42 months -- October 2013 → March 2017

Three institutes

- CEA
- XLIM Limoges → INRIA Rennes – LHS
- École Nationale Supérieure des Mines de Saint-Étienne

4 post-docs funded by the project

- Hélène Le Bouder, INRIA Rennes (2015-2016)
- Karim Abdellatif, ENMSE (2015-2016)
- Abderrahmane Seriai, CEA (2016-2017)

Project participants

- Karim Abdellatif (ENMSE), Thierno Barry (CEA), Nicolas Belleville (CEA), Damien Couroussé (CEA), Philippe Jaillon (ENMSE), Jean-Louis Lanet (INRIA), Hélène Le Bouder (INRIA), Hassan Noura (CEA), Olivier Potin (ENMSE), Bruno Robisson (CEA), Abderrahmane Seriai (CEA)
PHYSICAL ATTACKS

One of the major threats against secure embedded systems

- The only effective class of attacks against crypto-systems
- Relevant in many cases against cyber-physical systems: bootloaders, firmware upgrade, reverse-engineering, etc.

Observation-based: side channel attacks

Perturbation-based: fault attacks

- hiding
- masking
- tolerance
- detection
One of the major threats against secure embedded systems
- The only effective class of attacks against crypto-systems
- Relevant in many cases against cyber-physical systems: bootloaders, firmware upgrade, etc.

**Observation-based:** side channel attacks

**Perturbation-based:** fault attacks

**Application of software countermeasures**
1. SmartCard industry
2. IoT industry

**Our approach**

**Source code**

Source to source approach

**Compiler**

Assembly approach

**Binary code**

+ runtime code generation
COGITO: CODE POLYMORPHISM

Code polymorphism: regularly changing the behavior of a (secured) component, at runtime, while maintaining unchanged its functional properties,

- Protection against physical attacks: side channel & fault attacks
  - Changes the spatial and temporal properties of the secured code
  - Can be combined with other state-of-the-Art HW & SW Countermeasures
- Implementation with runtime code generation

STM32 (Cortex-M3)
COGITO: CODE POLYMORPHISM

Code polymorphism: regularly changing the behavior of a (secured) component, at runtime, while maintaining unchanged its functional properties,

- Protection against physical attacks: side channel & fault attacks
  - Changes the spatial and temporal properties of the secured code
  - Can be combined with other state-of-the-Art HW & SW Countermeasures
- Implementation with runtime code generation

Observation-based: side channel attacks
- hiding
- masking

Perturbation-based: fault attacks
- tolerance
- detection

© CEA 2017. All rights reserved | 15/09/2017 | 6
PROJECT CHALLENGES

- Demonstrate applicability to constrained embedded systems (IoT, SmartCard...)
  - Experiment target: ARM Cortex-M3 (32-bits), 8 kB RAM
  - Current dynamic compilation frameworks incur a too large overhead.
  - Solution: generate ad hoc runtime code generators

- Automated application from C source code

- Small performance overhead

- Certification
  - Polymorphism can be used in certified components (Collaboration with ANSSI)

- Effectiveness against side-channel attacks
Polymorphism is a hiding countermeasure against side-channel attacks – does not *remove* information leakage; *reduces* SNR only

However, information leakage is sufficiently blurred such that it is *not found* in observation traces, with a confidence level of 99.999%

Configurable level of polymorphism

### CPA attack results

Unprotected reference

*Low polymorphic variability*

t-tests results (2x 10000 observations)
Proof-of-concept implementation of code polymorphism
  - A practical solution, even on constrained embedded systems, to diversify the runtime behaviour of a software component.
  - Increases the resistance against side channel attacks
  - Application of polymorphism can be fully automated

Code polymorphism is compatible with certification standards

On-going work
  - Combination of polymorphism with other countermeasures
  - Validation of a polymorphic component
COGITO – Runtime Code Generation to Secure Devices

damien.courousse@cea.fr

Workshop Interdisciplinaire sur la Sécurité Globale,
Paris 14 et 15 sept. 2017