Harmonizing Safety, Security and Performance Requirements in Embedded Systems

Prof Ludovic APVRILLE, Dr Letitia LI
ludovic.apvrille@telecom-paristech.fr
letitia.li@baesystems.com

DATE’2019
Outline

Context: Security for Embedded Systems
Embedded systems

SysML-Sec
Method
SysML-Sec

Case study
Case Study

Conclusion
Conclusion, future work and references
Examples of Threats

Transport systems

► Use of exploits in Flight Management System (FMS) to control ADS-B/ACARS [Teso 2013]
► Remote control of a car through Wifi [Miller 2015] [Tencent 2017]

Medical appliances

► Infusion pump vulnerability, April 2015.
http://www.scip.ch/en/?vuldb.75158
Examples of Threats (Cont.)

Internet of Things

- Proof of concept of attack on IZON camera [Stanislav 2013]

- Vulnerability on fitbit [Apvrille 2015]

- Hacking a professional drone [Rodday 2016]
Vulnerability Identification
Vulnerability Identification (Cont.)

Investigations

- Testing ports (JTAG interface, UART, ...)
- Firmware analysis
- Memory dump
- Side-channel analysis (e.g. power consumption, electromagnetic waves)
- Fault injection
- ...

Secure your systems!

Develop your system with security in mind from the very beginning

Our solution: SysML-Sec, supported by TTool
Designing Safe and Secure Embedded Systems: SysML-Sec

Main idea

- **Holistic approach**: bring together embedded system experts, system architects, system designers and security experts (with SysML)

Common issues (addressed by SysML-Sec):

- Adverse effects due to security on safety/real-time/performance properties
  - Commonly: only the design of security mechanisms
- Hardware/Software partitioning and Design Space Exploration
  - Commonly: no support for security
Context: Security for Embedded Systems

SysML-Sec

**Analysis**
- Requirements
  - Safety
  - Functional
  - Security
- Attack Trees

**HW/SW Partitioning**
- Application
- Architecture
- Mapping

**Verification**
- Safety
- Performance
- Security

**Software Design**

**Safety Countermeasures**
- Redundancy, ...

**Security Countermeasures**
- Firewall, Data Security, ...
- Security Algorithms, ...

**Attacker Scenarios**

**Fault Trees**

**Legend**
- Modeling
- Verification
- Security
- Safety
- User-defined
- Automatic
- Reconsideration

**Code Generation**

**Fully supported by TTool**

8/31 March 2019
Institut Mines-Telecom
SysML-Sec
Partitioning

Before mapping
- Security mechanisms can be captured but not verified

After mapping
- Verify security (confidentiality, authenticity) according to attacker capabilities
  - Whether different HW elements are or are not on the same die
  - Where cryptographic materials (keys) are stored
  - Where encrypt/decrypt operations are performed
- Impact of security mechanisms on performance and safety
  - e.g. increased latency when adding security mechanisms
Partitioning Verification

Modeling

Automatic Verification

Safety

Performance

Security
Safety/Security/Performance

Requirements
- Security
- Safety
- Performance

System design

Verification of design w.r.t. requirements

Security
- Fails
  - Reconsider security req.
  - Add/modify security mechanisms
  - Modify architecture (private bus, etc.)
  - Modify mapping
- Succeeds
  - Security mech. leads to unsafe behavior
  - Security mech. leads to degraded perf. (e.g., increased avg. latency)

Safety
- Fails
  - Reconsider safety req.
  - Add/modify safety mech. (e.g. safe modes)
  - Modify architecture (e.g. redundancy)
  - Modify mapping
- Succeeds
  - Safety mech. leads to insecure behavior
  - Safety mech. leads to degraded performance

Performance
- Fails
  - Reconsider performance req.
  - Reconsider algorithms
  - Modify architecture (Nb of cores, etc.)
  - Modify mapping
- Succeeds
  - Performance issue due to safety mechanisms
  - Performance issue due to security mechanisms

Automated generation
Safety, Security and Perf. Mechanisms

Safety
- Fail-safe mode
- Redundancy
- Resistance to external phenomenon
- System monitoring, event logging and watchdogs
- Plausibility check
- Anomaly detection
- RTOS (determinism)
- ...

Security
- TPMs
- Cryptography
- Security protocols
- Firewalls
- Intrusion detection Systems
- Secure boot
- ...

Performance
- Faster hardware
- Less complex versions of algorithms
- Move software functions to hardware
- ...

12/31 March 2019
Institut Mines-Telecom
SysML-Sec
Safety and Security Mechanisms

Data Encryption/ Authentication

Safety

Security

Performance
Data Security with Hardware Security Module

Safety

Security

Performance
Safety and Security Mechanisms (Cont.)

Redundancy/Coherence Check

Add security

Add security

Safety

Security

Performance
Safety and Security Mechanisms

Failsafe mode

- **System Check**
  - `[systemOk]` -> **defaultMode**
  - `[else]` -> **failsafeMode**

**Safety**

**Security**

**Performance**

March 2019

Institut Mines-Telecom

SysML-Sec
Precise model of security mechanisms (security protocols)
Proof of security properties: confidentiality, authenticity
Channels between software blocks can be defined as private or public
  This should be defined according to the hardware support defined during the partitioning phase
Case Studies

Cyber security of connected vehicles

- Safety/Security/Performance
- EVITA FP7 Partners: Continental, BMW, Bosch, ...
- VEDECOM

H2020 AQUAS

- Automated train sub-systems (ClearSy): Safety/Security/Performance
- Industrial Drives (Siemens): Safety/Security/Performance

Nokia

- Digital architectures for 5G networks (Safety/Performance)
Case Study: VEDECOM Autonomous Vehicle

Tests

Verification

Model

<<MEMORY>>
MemorySystem2
<<BUS-RR>>
MemoryBus2
<<CPURR>>
CameraCPU
Design::Camera
<<BUS-RR>>
EthernetCamera
<<CPURR>>
IMU_CPU
Design::IMU
<<BUS-RR>>
BusIMU
<<CPURR>>
SupervisorCPU
Design::Supervisor
<<BUS-RR>>
CANVedecom
<<MEMORY>>
MemorySystem
<<BUS-RR>>
MemoryBus
<<CPURR>>
PerceptionCPU
Design::Perception
<<BUS-RR>>
supervisorCPU
Design::Supervisor
<<BUS-RR>>
MemoryBox2
<<BUS-RR>>
MemoryBox2

19/31 March 2019
Institut Mines-Telecom
SysML-Sec
Constraints

- Standard: ISO26262
  - SOTIF: Safety Of The Intended Function
- Security: impact of potential attacks on safety
### Requirements

**SafetyMain**

- **ID=0**
  - Text: "The system should be safe."
  - Kind: "Safety"

**PerformanceLatency**

- **ID=5**
  - Text: "The critical system latencies should ensure safety."
  - Kind: "Performance"

**SafeFunction**

- **ID=8**
  - Text: "The vehicle function should be safe."
  - Kind: "Functional"

**LatencyBraking**

- **ID=10**
  - Text: "The system should respond safely to events."
  - Kind: "Functional"

**FunctionalObstacleDetect**

- **ID=11**
  - Text: "A braking order should follow every detection of obstacles in close proximity."
  - Kind: "Functional"

**LidarObstacleDetect**

- **ID=9**
  - Text: "The Navigation unit should always issue an braking order after the Lidar detects an obstacle."
  - Kind: "Functional"

**SafeCommands**

- **ID=18**
  - Text: "The system should prevent an attacker from injecting vehicle commands."
  - Kind: "Functional"

**ReliabilityCommand**

- **ID=16**
  - Text: "The system should ensure that commands will be issued reliably."
  - Kind: "Performance"

**AuthenticityMessage**

- **ID=15**
  - Text: "The system should ensure the Authenticity of all internal messages."
  - Kind: "Integrity"

**AuthenticityCommand**

- **ID=16**
  - Text: "The system should ensure the Authenticity of all vehicle commands."
  - Kind: "Integrity"

**MACdata**

- **ID=20**
  - Text: "The system will use Message Authentication Codes to ensure the Authenticity of data."
  - Kind: "Integrity"

**DataSecurity**

- **ID=19**
  - Text: "The system will ensure Authenticity with data security mechanisms."
  - Kind: "Integrity"

**SafetyMain**

- **ID=0**
  - Text: "The system should be safe."
  - Kind: "Safety"

**SecurityMain**

- **ID=0**
  - Text: "The system should be secure."
  - Kind: "Non-functional"
# Attacks

![Diagram of attacks on a vehicle system]

- **Root Attack**: `attackAutomaticBraking`
  - **OR**: `preventObstacleDetection`, `preventBrakingFunction`
    - **AND**: `manipulateSensors`, `manipulateCamera`, `manipulateLIDAR`
      - **OR**: `preventDataComputation`, `forgeInternalMessages`, `preventDataTransmission`
        - **OR**: `corruptControllerCode`, `forgeObstacleData`, `forgeECUCredentials`
          - **OR**: `jamInCarCommunications`

- **Countermeasure**: `authenticateECUCommands`
Functional View
Safety Verification (Before Mapping)

Reachability/Liveness

Queries

Safety Pragma
A[\] Supervisor.running
Perception.distance<threshold -->
Supervisor.brakingOrder
Architecture and Mapping Views
Safety Verification (After Mapping)

Reachability Graph

Minimized RG

26/31 March 2019 Institut Mines-Telecom
Security Verification

Dialog window

Backtracing

Security Verification
Performance Verification

Latency

Bus/CPU Load

March 2019

Institut Mines-Telecom

SysML-Sec
**SW Design, Code generation, Test**

- First SW model from mapping models
- SW model refinement
- SW model verification (safety, security)
- Code generation
  - (Virtual) Prototyping, test
Conclusion and Future Work

Achievements: SysML-Sec
- Methodology for designing safe and secure embedded systems
- Fully supported by TTool
- Applied to different domains, e.g., automotive systems, IoTs, malware

Future work
- Security risk assessment and backtracing
- Assistance in handling conflicts between security/safety/performance
  - Design space exploration
For more information ...

Web sites

▶ https://ttool.telecom-paristech.fr

References