Formal Security Analysis of Smart Embedded Systems

Farid Molazem, Karthik Pattabiraman

Dependable Systems Lab
University of British Columbia
Internet of Things
Real attacks against IoT

- [Koscher 2010, Zetter 2010]

Hackers Remotely Kill a Jeep on the Highway—With Me in It

Forbes / Tech

09 FBI: Smart Meter Hacks Likely to Spread

A series of hacks perpetrated against so-called “smart meter” installations over the past several years may have cost a single U.S. electric utility hundreds of millions of dollars annually, the FBI said in a cyber-intelligence bulletin obtained by KrebsOnSecurity. The law enforcement agency said this is the first known report of criminals compromising the hi-tech meters, and that it expects this type of fraud to spread across the country as more utilities deploy smart grid technology.

Smart meters are intended to improve efficiency, reliability, and allow the electric utility to charge different rates for...
Security Mechanisms

- Hardware-based techniques [Schellekens 2008]
Security Analysis

- Attack trees [Byres 04, Morais 09]
- Predefined attack goals
- Manual search
- Attack graphs [Jha 02, Sheyner 02]
- Need vulnerabilities of the hosts
- Formal analysis [Delaune 10, Miculan 11]
- Targets well-defined protocols
Idea

- IoT devices perform *specific* tasks
  - Define the right abstraction
    - Not too low level, not too high level

- Opens door to formal analysis
High-level picture

System specification

Formal model of the system

Formal model of attacker

User

Source code

Security expert (Us)
Abstraction – step 1

Rewriting Logic

System Model

Analysis

Rewriting System Model

Changes to the System Model

Rewriting Logic
Abstraction – step 1

Rewriting logic:
- Rewrite rules
- Equations
Abstraction – step 2

[Molazem 14]

Design
Specifications

Comp. 1

Formal
model

Comp. 2

Formal
model

Comp. 3

Formal
model
Abstraction – step 3

Attacker action:
e.g. access to the $i$th sensor channel

Explicit model checking:

$\text{Start} \rightarrow \text{receive}(c_1, v_1) \text{ where } v_1 < 0$
Abstraction – step 4

Formal attack paths

Source code

Control Flow Graph
Case study

SEGMeter: an open source smart meter

- Sensor board: Receive raw data
- Communication board: talk to server
- Code base: Lua and C (~ 3000 LOC)
Threat model

- **Access**
  - Root access to a node in grid network [Mo et al. 2012]

- **Actions**
  - Drop messages
  - Replay messages
  - Reboot meter

Read/Write access to communication interfaces [McLaughlin et al. 2010]
Evaluation

Q1: Performance

Q2: Practicality
Evaluation

Performance

Using Maude [Clavel 15]:
http://maude.cs.illinois.edu/

Less than a second ➡ up to 2 hours

3.4 GHz CPU, 16GB RAM: Reasonable time
Evaluation

Practicality

- Query for paths to unsafe states

\[
\text{search sensor}(N1, M1) \ \text{sensor}(N2, M2) \ \text{sensor}(N3, M3) \Rightarrow \\
\text{stored}(N1, M1) \ \text{stored}(N2, M2)
\]

- Some map to the same execution path
Attack example

start ➔ Receive new data ➔ Add to old data ➔ Send to server ➔ Reboot

S1 ➔ S2 where data(s1) *not sent* & cycle=start
Attack example

Will lose data if reboot

Vulnerability window

Open file in write mode

1. function update_node_list()
2. all_data = get_node_list
3. all_data = merge_table(current,all_data)
4. data_file = assert(io.open(dataFile, “w”))
5. for key, value in pairs(node_list) do
6. data_file::write(data)
7. end
8. assert(data_file::close())
9. end
Attack example (video)
Discussion

• Applicability to other devices
  ● Cars (AUTOSAR)
  ● Medical devices

• Model correctness
  ● Refine the model

• Abstraction level
  ● The model is extensible
Conclusion

- IoT devices perform specific tasks
  - Abstract out their operations
  - Formalize them
  - Formalize the attacker
  - Perform automated analysis

- Find real vulnerabilities