Formal Modeling and Proving Cryptographic Properties of the Optin Sensor Protocol

HU-MATHS-IN
Hungarian Service Network for Mathematics in Industry and Innovations

The Industrial Problem

In this project aided with authomatic theorem proving we developped a lightweight cryptographic extension of the Optin Sensor Protocol which is heavily used at our industrial partner in IoT devices.

INCLUDE THE MORE APPROPRIATED INDUSTRIAL SECTOR

Research group

SZTE TTIK INFORMATIKAI INTÉZET

Reseach on Mathematical Modleing of Information Security and Cryptography at the Department of Foundations of Computer Science

Company

Opin Kft.

OPTIN

Developer and active user of the OSP in the automotive industry, medical services in other fields using IoT devices

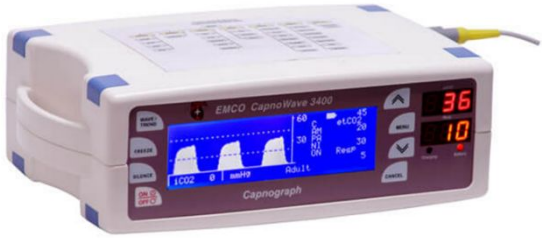
SZÉCHENYI 2020
MAGYARORSZÁG KORMÁNYA
Európai Unió
Európai Strukturális és Beruházási Alapok
BEFEKTETÉS A JÖVŐBE

Formal Modeling and Proving Cryptographic Properties of the Optin Sensor Protocol

HU-MATHS-IN
Hungarian Service Network for Mathematics in Industry and Innovations

Challenges & Goals

- Determine the security requirements for some typical industrial use of the OSP at OPTIN Kft.
- Extend the protocol with cryptographic features
- Maintain both security and usablilty in a restricted environment
- Analyze the security properties of the new extension(s)
- Implement, test and use the new protocol features in real-world applications
- Validate the new design by mathematical modeling
- Publish the new protocol version and t
scientific results



Devices using the OSP protocol at OPTIN Kft.

Formal Modeling and Proving Cryptographic Properties of the Optin Sensor Protocol

HU-MATHS-IN
Hungarian Service Network for Mathematics in Industry and Innovations

Mathematical and computational methods and techniques applied

- We modeled both the new protocol desing and the security requirements by logic formulas.
- We used the TAMARIN-Proover softwer for formal verification of the key security properties.

```
Unsecure client:
Test   1: Succesful connection                PASSED
Test   2: Bad SessionID                       PASSED
Test   3: Bad SeqNum                          PASSED
Test   4: Bad MessageType                     PASSED
Test   5: Bad flags                           PASSED
Test   6: Large PacketSize                    PASSED
Test   7: Small PacketSize                    PASSED
Test   8: Bad ConnState                       PASSED
Test   9: Bad DeviceType                      PASSED
Test  10: Bad ModuleID                        PASSED

Secure Client:
Test   1: Successful four-way handshake       PASSED
Test   2: Plaintext Data                      PASSED
Test   3: Authenticated, encrypted Data       PASSED
Test   4: EAX Data, bad MAC                   PASSED
Test   5: EAX Data, good MAC, bad Ciper       PASSED
Test   6: Plaintext Ping                      PASSED

Problems with Init New Connection Message:
Test   1: Bad SessionID                       PASSED
Test   2: Bad SeqNum                          PASSED
Test   3: Bad MessageType                     PASSED
Test   4: Bad Flags                           PASSED
Test   5: Large PacketSize                    PASSED
Test   6: Small PacketSize                    PASSED
Test   7: Bad ConnState                       PASSED
Test   8: Bad DeviceType                      PASSED
Test   9: Bad ModuleID                        PASSED
```

Proof of the key_secret lemma

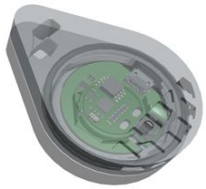
```
lemma key_secret:
  all-traces
  "∀ k #i #j. ((Secret( k ) @ #i) ∧ (K( k ) @ #j)) ⇒ (⊥)"
simplify
solve( Secret( k ) @ #i )
  case Connect_Step_2
  solve( Init_Client( cIV ) ▶₀ #i )
    case Connect_Step_1
    solve( !Key( ~k ) ▶₃ #i )
      case setup
      by solve( !KU( ~k ) @ #vk )
    qed
  qed
next
  case Connect_Step_3
  solve( Init_Server( senc(<sIV, cIV>, ~k) ) ▶₁ #i )
    case Connect_Step_2
    solve( !Key( ~k ) ▶₂ #i )
      case setup
      by solve( !KU( ~k ) @ #vk )
    qed
  qed
next
  case Connect_Step_4
  solve( Client_Auth( w ) ▶₀ #i )
    case Connect_Step_3
    solve( !Key( ~k.1 ) ▶₂ #i )
      case setup
      by solve( !KU( ~k.1 ) @ #vk )
    qed
  qed
next
  case Send_Data
  solve( !Authenticated( x ) ▶₁ #i )
    case Connect_Step_4
    solve( !Key( ~k.1 ) ▶₂ #i )
      case setup
      by solve( !KU( ~k.1 ) @ #vk )
    qed
  qed
qed
```

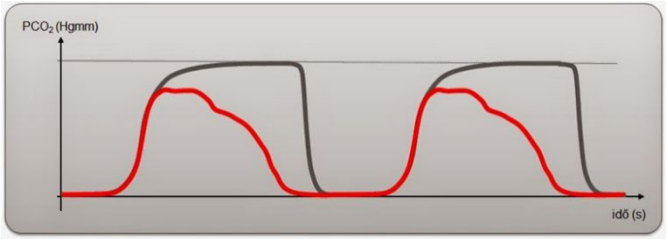
Formal Modeling and Proving Cryptographic Properties of the Optin Sensor Protocol

HU-MATHS-IN
Hungarian Service Network for Mathematics in Industry and Innovations

Results & Benefits to the company

- The new OSP 2.0 protocol standard is available for download from the website of the company
- The new more secure design is in use in several ongoing projects
- Further applications of the protocal are planned
- The new public standard encourages wider industrial cooperation





Cryptographic and mathematical modeling insights are beneficial for the system engineers of the company