MPInspector: A Systematic and Automatic Approach for Evaluating the Security of IoT Messaging Protocols

Qinying Wang    Shouling Ji    Yuan Tian    Xuhong Zhang    Binbin Zhao
Yuhong Kan    Zhaowei Lin    Changting Lin    Shuiguang Deng    Alex X. Liu    Raheem Beyah

2021
Cloud based IoT Platforms

Most IoT platforms offer MP (Messaging Protocol) implementations.
Typical Architecture of IoT platform

Device → Telemetry and event message → Message Broker → Telemetry and event message → User Application

Control command message

MQTT/CoAP/AMQP/…
An MP Workflow

An example workflow of MQTT standards:

CONNECT(ClientId, Username, Password, WillTopic, WillMessage)

CONNACK(returnCode)
An MP Workflow

An example workflow of MQTT standards:

Connection | SUBSCRIBE | PUBLISH | UNSUBSCRIBE | DISCONNECT

Application

SUBSCRIBE(Topic, packetId, qos)

SUBACK(packetId, returnCode)

PUBLISH(Topic, Payload, packetId, qos)

PUBACK(packetId, returnCode)

Broker

IoT Device

PUBLISH(Topic, Payload, packetId, qos)

PUBACK(packetId, returnCode)
Security and Privacy Threats on MP

Several MP flaws have been spotted including denial of service, sensitive data theft, malicious command injection, etc.
Threat Model

_neighbor scenario_

✓ The victim and the attacker are in the same network.
✓ The attacker can _eavesdrop, drop, modify, inject messages._

_tenant scenario_

✓ The victim uses some devices previously being used by the attacker.
✓ The attacker can _collect the device identity_ (e.g., password).
✓ The attacker can _leave a backdoor_ on the device.
How to build a **systematic** and **automatic** tool to evaluate the security of IoT MPs?
Challenges

- **Diverse and customized MP implementations**
  - Multiple types of MPs
  - Customized implementations on different platforms

- **Complex and closed-source MP workflow**
  - Stateful procedure with multiple messages
  - The implementation are closed-source
Customized MP Implementations

Topic: smart/device/out/03613xxx
Message: 2.1edxxxxxx+1+3SvBBejkIKA==

Customized parameters Topic and Message
Customized MP Implementations

MP interaction logic

Broker on Google

Brooker on Bosch

CONNECT(ClientId1)

Device1 Online

Device2

Device1 Online

Device2
Customized MP Implementations

MP interaction logic

Customized interaction logic on duplicate connections with the same ClientId
Challenges

☑️ Diverse and customized MP implementations
  ✓ Multiple types of MPs
  ✓ Customized implementations on different platforms

☑️ Complex and closed-source MP workflow
  ✓ Stateful procedure with multiple messages
  ✓ The implementations are closed-source
Limitations of Existing Attack Finding Strategies for IoT Protocols

- Few analysis on implementation, mostly analyze the specification
- Few logic vulnerabilities which do not cause crashes
- No systematic and automatic approach
Insight

A property-driven and model-based testing philosophy.

State machines

Property sets

Tamarin prover

Property validation
Insight

A **property-driven** and model-based testing philosophy.

- **Secrecy properties extracted from the specification**
  - A set of **parameters** from messages that should be confidential
  
  $\text{SecProp}_\text{Set} = \{\text{ClientId, Password, PUBLISH payload, ...}\}$

- **Authentication properties extracted from the specification**
  - A set of **messages** that should be authenticated
  
  $\text{AuthProp}_\text{Ser} = \{\text{CONNECT, PUBLISH, SUBSCRIBE, ...}\}$
A property-driven and **model-based** testing philosophy.

**Model the interaction logic with messages as a state machine**

State machine of the broker and device
A property-driven and **model-based** testing philosophy.

State machine of the broker and device
Overview of MPInspector

MPInspector has 5 modules and 3 inputs.

MP state machine extraction

Formal representation and formal verification

Parameter Decoder*  Traffic-based Semantics Extraction  Semantics Map

Parameter Searching  NLP-based Semantics Extraction  Semantics Assignment

1. Message Semantics Extraction

Semantics Map

MP Traffic

IoT Platform Documents

Communication Configurations

Adaptor*  Membership Queries  Refinement

Equivalence Queries

2. Interaction Logic Extraction

State Machine

Meta Properties*  Property Extending

3. Properties Generation

Helping Oracle

5. Formal Verification

Lemmas Rules  ...

State Machine Translation*

4. Formal Code Translation

Security Property Translation

Tamarin Code

Property Violation
Message Semantics Extraction Workflow

- Extracting the **customized** message semantics is not trivial.
  - Traffic and document based analysis
  - Patter matching & NLP

Will Message: {
  "clientId" : "036130xxx",
  "username" : "light123/dev1"
}

User Name: light123/dev1
Password: 5570ff002f8bd758

Traffic file

Pattern Matching
Message Semantics Extraction

NLP assisted IoT platform documents analysis.

Will Message: {“clientId”:“036130xxx”, “username”:”light123/dev1”}
User Name: light123/dev1
Password: 5570ffxxxxxbd758

mqttPassword:sign_hmac
(deviceSecret,content)

{iotubhostname}/{deviceId} in the Username field

Traffic file

IoT platform documents
Message Semantics Extraction

NLP assisted IoT platform documents analysis.

(1) `mqttPassword = sign_hmac(deviceSecret, timestamp, rawPassword)`  
Pattern matching

(2) The password field contains the SAStoken.  
The NLP method

(3) The format of the SAStoken is the same as for both the HTTPS and AMQP protocols:  
`SharedAccessSignatures={signature-string}&se={expiry}&sr={URL-encoded-resourceURI}`  
The NLP method
Message Semantics Extraction Workflow

- Extracting the **customized** message semantics is not trivial.
- Traffic and document based analysis
- Pattern matching & NLP

EXP: 
{"CONNECT": {"ClientID": "", "username": {"composition": ["authid", "tenantid"]}, "password": {"encryption": "HMAC", "encryptedTerms": ["authid", "timestamp"]}}}

2021/7/7
Interaction Logic Extraction

Apply active model learning to infer the interaction logic.
Interaction Logic Extraction

- Only supports **two parties and response messages**.
  - Extend the adapter to support **multi-parties** and monitoring the connection state.

Diagrams:
- Membership Queries
  - Abstract inputs
  - Abstract outputs
  - Sequences of message types

- Equivalence Queries
  - Abstract inputs
  - Abstract outputs

- Adapter
  - Concrete stimuli
  - Concrete reactions
  - Sequences of real messages

- MP Broker
  - Concrete stimuli
  - Concrete reactions
Interaction Logic Extraction Workflow

- The equivalence query is **time-consuming** while dealing multitype of messages.
  - A customized equivalence algorithm to **cut down unnecessary queries**

- **Model learning may be trapped into an endless procedure;**
  - An enhance learning algorithm

- **Cond. 1:** If an counterexample is found?
- **Cond. 2:** Is the number of same hypotheses greater than the threshold?
Interaction Logic Extraction

Modeling validity predicates.

Message example

```
Delivery-Id:3
...
Password: 5570ffxxxxxbd758
...
```

Broker

Validity predicate testing by sending mutated message to the broker
Overview of MPInspector

MPInspector performs formal representation and formal verification.
Formal Verification

✗ The search space of possible states may potentially **explode**.
   ✓ An inherent limitation of Tamarin Prover

✓ **Helping oracle** ranks the open goals based on our strategies

- The source of a state (The ones contain a state of a longer trace rank first)
- The existence of an action fact indicating the attacker knows secret key or password.
- The existence of an action indicating the attacker knows an encrypted parameter.
- Other goals
New Extension for New Types of MP

- A one-short effort for each new MP type
  - Message structure, meta properties and initial state of MP
  - Concluded from the MP specification
Experiment settings

- Test **ten** MP implementations from **nine** leading IoT platforms

- Test the SaaS applications on our own services
- Validate our attack on our own devices

- MQTT V3.1.1
- MQTT V5.0
- AMQP V1.0
- CoAP
# Findings

- **Uncovered 11 types of MP attacks**

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Attacks</th>
<th>Affected Protocol</th>
<th>Affected Platforms</th>
<th>Related Pr.</th>
<th>Attack Success</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neighbor Scenario</td>
<td>Man-in-the-middle</td>
<td>All protocols</td>
<td>All platforms</td>
<td>MA1-MA9, AA1-AA13, CA1-CA8</td>
<td>●</td>
</tr>
<tr>
<td></td>
<td>MQTT V3.1.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MQTT V5.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CoAP</td>
<td>EMQ X</td>
<td>CA1-CA4</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AMQP V1.0</td>
<td>ActiveMQ</td>
<td>AA1-AA13</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Transfer sync. failure</td>
<td>AMQP V1.0</td>
<td>ActiveMQ</td>
<td>AA1-AA9</td>
<td>●</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Client Identity Hijacking</td>
<td>MQTT V3.1.1</td>
<td>Google IoT Core</td>
<td>MS1-MS7, MA1, MA3, MA5, MA7, MA9, R2</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MQTT V5.0</td>
<td>Azure IoT Hub</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>AWS IoT Core</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Aliyun Cloud</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AMQP V1.0</td>
<td>ActiveMQ</td>
<td>AS1-AS5, AS1, AS3, AS5, AS7, AS9, AS11, AS13</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CoAP</td>
<td>EMQ X</td>
<td>CS1, CS11, CA1, CA3, CA5, CA7</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Aliyun Cloud</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reflection attack</td>
<td>CoAP</td>
<td>EMQ X</td>
<td>CS1, CA1, CA3, CA5, CA7</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td>Malicious Topic Subscription</td>
<td>MQTT V3.1.1</td>
<td>AWS IoT Core</td>
<td>S5, MS7, MA3</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AMQP V1.0</td>
<td>ActiveMQ</td>
<td>AS2, AS4, AA9</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td>Malicious Topic Publish</td>
<td>MQTT V3.1.1</td>
<td>AWS IoT Core</td>
<td>MS5, MS7, MS9, MA7</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CoAP</td>
<td>EMQ X</td>
<td>CS1, CA3</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td>Malicious Response Topic Publish</td>
<td>MQTT V5.0</td>
<td>Mosquito</td>
<td>MS5, MS7-9, MA7</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td>Unauthorized Will Message</td>
<td>MQTT V3.1.1</td>
<td>AWS IoT Core</td>
<td>MA1, MA10</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td>Unauthorized Retained Message</td>
<td>MQTT V5.0</td>
<td>Mosquito</td>
<td>MA1, MA10</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td>Illegal Occupation</td>
<td>AMQP V1.0</td>
<td>ActiveMQ</td>
<td>AS1, AA1, AA3</td>
<td>●</td>
<td></td>
</tr>
</tbody>
</table>
Denial of Service (Neighbor Scenario)
Unauthorized Response Topic publish (Tenant Scenario)

Attacker

CONNECT(ClientId’, Username’, Password’)

PUBLISH(topic, data, properties(responseTopic (victim’s topic), correlationData(evil data)))

Broker

CONNECT(ClientId, Username, Password)

SUBSCRIBE(topic)

PUBLISH(topic, data, properties(responseTopic (victim’s topic), correlationData(evil data)))

PUBLISH(victim’s topic, evil data)

Victim Device
AMQP illegal occupation (Tenant Scenario)
Performance

The overhead of MPInspector

- The average precision of property violations is 1.00
- The average overhead is ~4.5 hours

<table>
<thead>
<tr>
<th>IoT Platform</th>
<th>MP</th>
<th>Message semantics Extraction</th>
<th>Interaction Logic Extraction</th>
<th>Formal code Translation</th>
<th>Total Time (h:mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Time (ms)</td>
<td>Precision</td>
<td>States</td>
<td>Time Delay</td>
</tr>
<tr>
<td>Google IoT Core</td>
<td>MQTT V3.1.1</td>
<td>115</td>
<td>1.00</td>
<td>3</td>
<td>8s</td>
</tr>
<tr>
<td>AWS IoT Core</td>
<td>MQTT V3.1.1</td>
<td>102</td>
<td>1.00</td>
<td>3</td>
<td>3s</td>
</tr>
<tr>
<td>AWS IoT Core(with)</td>
<td>MQTT V3.1.1</td>
<td>103</td>
<td>1.00</td>
<td>8</td>
<td>5s</td>
</tr>
<tr>
<td>Azure IoT Hub</td>
<td>MQTT V3.1.1</td>
<td>107</td>
<td>1.00</td>
<td>3</td>
<td>8s</td>
</tr>
<tr>
<td>Bosch IoT Hub</td>
<td>MQTT V3.1.1</td>
<td>106</td>
<td>1.00</td>
<td>5</td>
<td>9s</td>
</tr>
<tr>
<td>Aliyun Cloud</td>
<td>MQTT V3.1.1</td>
<td>105</td>
<td>0.96</td>
<td>3</td>
<td>4s</td>
</tr>
<tr>
<td>Tuya Smart</td>
<td>MQTT V3.1.1</td>
<td>110</td>
<td>1.00</td>
<td>3</td>
<td>8s</td>
</tr>
<tr>
<td>Mosquitto</td>
<td>MQTT V5.0</td>
<td>106</td>
<td>1.00</td>
<td>2</td>
<td>1s</td>
</tr>
<tr>
<td>Mosquitto(with)</td>
<td>MQTT V5.0</td>
<td>106</td>
<td>1.00</td>
<td>6</td>
<td>5s</td>
</tr>
<tr>
<td>Mosquitto(retain)</td>
<td>MQTT V5.0</td>
<td>106</td>
<td>1.00</td>
<td>8</td>
<td>7s</td>
</tr>
<tr>
<td>EMQ X</td>
<td>CoAP</td>
<td>928</td>
<td>1.00</td>
<td>1</td>
<td>1s</td>
</tr>
<tr>
<td>Aliyun Cloud</td>
<td>CoAP</td>
<td>2152</td>
<td>1.00</td>
<td>2</td>
<td>1s</td>
</tr>
<tr>
<td>ActiveMQ</td>
<td>AMQP V1.0</td>
<td>1808</td>
<td>1.00</td>
<td>9</td>
<td>1s</td>
</tr>
</tbody>
</table>
Discussion

☒ Mitigate security risks

✓ Fine-grained testing and more flexible model learning strategies
✓ Automatic meta property extraction based on NLP
✓ Applying MPIInspector on more MPs and devices

☒ Limitation and future work
Summary

- The first systematic and automatic framework for evaluating the security of MP implementations.

- A large-scale experiment on 3 popular MPs on 9 leading IoT platforms.

- Uncover 11 kinds of attacks.

- [https://github.com/wqqqqy/MPInspector](https://github.com/wqqqqy/MPInspector)
wangqinying@zju.edu.cn