IoTGuard: Dynamic Enforcement of Security and Safety Policy in Commodity IoT

Z. Berkay Celik
Gang Tan
Patrick McDaniel

Penn State University
NDSS 2019
Internet of Things (IoT) enables the future

Smart Homes
Source: Samsung

Smart Energy
Source: LG

Healthcare
Source: John Hopkins

Smart Farms
Source: Microsoft

IoT is not magic

Connected devices

Mobile app

IoT application

Automation
IoT enables the future (and a whole lot of problems)

All of these failures are traditional security problems:

Software bugs, user error, poor configuration, or faulty design
IoT environment

IoT Devices

Eclipse SmartHome > IoT (52) IoT app market

Trigger-action app market
1. welcome-home IoT app
   E: light turned-on
   A: activate home-mode

2. home-mode-automation IoT app
   E: home-mode
   A: turn on heater and slow cooker, unlock patio-door

3. goodnight IoT app
   E: light turned-off
   A: set alarm at 7 am, turn on coffee machine at 7:15

4. Trigger-action platform IF rule
   E: coffee machine turned-on
   A: post a Tweet

5. simulate-occupancy app
   E: tap an app icon or at a time
   A: turn on lights
turn off lights

* E is for event, A is for Action

**App to turn Lights On/Off automatically while away (Simulate Presence)**

Mobile App

Imosenko  Community Journeyman  1 Dec '14

Looking for an app to turn lights On/Off while all are away from home to simulate presence. Just in case Mr./Mrs. Burglar want to drop by. Is there one out or can someone help with the code?
Interactions among IoT and trigger-action apps

**Welcome-home IoT app**
- E: light turned-on
- A: activate home-mode

**Home-mode-automation IoT app**
- E: home-mode
- A: turn on heater and cooker, unlock patio-door

**Goodnight IoT app**
- E: light turned-off
- A: set alarm at 7 am and turn on coffee machine at 7:15 am

**Trigger-action rule**
- E: coffee machine turned-on
- A: post a Tweet

**Simulate-occupancy app**
- E: tap an app icon
- A: turn on lights
- A: turn off lights

- E: coffee machine turned-on
- A: post a Tweet
How can we prevent safety and security violations within IoT environments?
Solution...

We need a custom system for IoT to ...

- Model device behavior from app source code
- Construct state transitions of the IoT environment
- Prevent IoT environment from arriving an undesired state

... But code analysis isn’t ideal

- No runtime monitoring: It may not anticipate devices at implementation time
- One sided: Users cannot reason about undesired states at runtime
- Scope: Its analysis is limited to pre-installed devices
IoTGuard

- **IoTGuard** is a dynamic **policy-based enforcement** system on IoT device behaviors

* We refer to IoT and trigger-action apps as IoT apps

![IoTGuard Diagram](image)

1. Instrument the code
2. Monitor runtime behavior
3. Block or ask users for unsafe state transitions

**IoTGuard Server**
- Code Instrumentor
- Data Collector
- Security Service

**IoTGuard**
- Policy Violation
- R.3 Enforced
- Unlock door state **blocked** when you are not home!

**IoTGuard Server**
- Instrument the code
- Monitor runtime behavior
- Block or ask users for unsafe state transitions

* We refer to IoT and trigger-action apps as IoT apps
Code instrumentation

• Add extra code logic to an app source code to work with IoTGuard
  › Perform path-based static analysis to collect app information and guard app actions
  › Optimize number of added instrumentation code block

Source code of home-automation IoT app

1: // Devices
2: presence sensor ps
3: door d
4: thermostat t
5: power meter p
6: when ps.present
7: t_home=71; thold=50;
8: d.unlock();
9: if (p.power<thold){
  t.set(t_home);
}

10: transmitAppInfo()
Data collector

- Store app’s information in a **dynamic model**
  - Extends Guova graph library for construction

- **Dynamic model** represents the **runtime** behavior of **individual** and **interacting** apps

  - **Instrumented IoT app**
  - **Data Collector (IoTGuard)**
  - **Instrumented trigger-action app**

  - Dynamic model contains device ID, predicates, event time, block/allow bit, and app info. object

  - **Event**
  - **Action**

  - Welcome-home
  - Home-automation
  - Twitter-trigger-action
  - Good-night
  - Simulate-occupancy

  - Instrumented apps

  - Individual dynamic models

  - Unified dynamic model
Security service - Property identification

- **Policy** is a system artifact that represents the real world needs of users and environments

**General properties**

- Constraints on states and transitions

  - motion-active → switch-on
  - motion-active → switch-off

  **Attributes of conflicting values**

  - motion-active → switch-on
  - user-present → switch-off

**Application-specific properties**

- Identify use cases of one or more devices

  1. The door must always be locked when the user is not home
  2. The refrigerator and security system must always be on
  3. The water valve must be closed if a leak is detected
  5. The alarm must always go off when there is smoke

* Extends safety and security properties of **Soteria system** (Celik et al., Usenix ATC’18) exercised through model checking
Security service - Policy identification

• Identify safety and security policies for trigger-action apps

• Trigger-action specific policies
  ▸ Label states through NLP techniques
  ▸ Store them in app’s dynamic model object

Overall **IoTGuard** checks an IoT environment against 36 identified policies

Violations in trigger-action apps: Surbatovich et al. (WWW’17), Celik et al. (arXiv’18)
Security service - Policy enforcement

- Security service blocks undesired states before happening
  - Enforce policies by exploring their reachability and check state labels during exploration

- Two solutions for policy enforcement

  ![Diagram](image)

  - **Integrity policy enforced**
    - User tagged in a post is blocked
  - **Confidentiality policy enforced**
    - Save it to a public file is blocked

- **Automated blocking**
  - P.3 Enforced
    - Door is locked when there is smoke door-lock state in welcome-home is blocked!
  - P.3 Violated
    - Door is locked when there is smoke door-lock state in welcome-home

- **User approval via runtime prompts**
Application study

- Implemented IoTGuard for **SmartThings** and **IFTTT** platform
- Selected **35** SmartThings IoT and **30** IFTTT trigger-action market apps

- Executing apps
  - Simulated a smart home including **29** devices with a total of **20** device types
  - Configured apps based on their descriptions
Policy enforcement in individual apps

- Enforced 3 (8%) policies and blocked 3 states in 5 (8%) apps

<table>
<thead>
<tr>
<th>App ID</th>
<th>Violation Description</th>
<th>Policy</th>
<th>Blocked</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST4-ST7</td>
<td>The heater is turned on when user is not at home</td>
<td>R.13</td>
<td>X heater on</td>
</tr>
<tr>
<td>IFTTT5</td>
<td>The switch is turned on when someone Tweets a hashtag</td>
<td>S.1</td>
<td>X switch on</td>
</tr>
<tr>
<td>ST11-ST12</td>
<td>Heater and AC turned on at the same time</td>
<td>R.17</td>
<td>X AC on</td>
</tr>
</tbody>
</table>

ST = SmartThings IoT apps    IFTTT = Trigger-action apps

- Source of policy violations
  - R:13: Interactions through abstract attributes
  - S.1: Lack of app-vetting for trigger-action apps
  - R.17: Misconfiguration of numerical-valued device attributes
Policy enforcement in multi-apps

- Enforced **9 (25%)** unique policies and blocked **18** states
  - Studied violations between interacting apps

```
Group 1

- sunset
- motion-active
- 11 pm

- turn off lights
- light-off
- change mode (away, sleeping)
- mode change
- heater on
- cooker on
- lights on

Group 2

- door ring-pressed
- missed-call
- email

- turn on lights
- light-on
- open window shades
- light-on
- send notification
```

- Each group includes a set of interacting IoT and **trigger-action** apps
Performance

- Code instrumentation
  - $14\pm4$ Lines of Code (LoC) added to the apps (+ 20 LoC for IoTGuard library)
  - $4.1\pm2$ seconds to add instrumentation code

- Runtime latency

---

**End-to-end overhead**:
The time between receiving an event and invoking an action

---

IoTGuard
Through this effort, we introduce a rigorously grounded system for enforcing correct operation of IoT devices through systematically identified IoT safety and security policies, demonstrating the effectiveness and value of monitoring IoT apps with tools such as IoTGuard.

Thanks for listening!