Systems and Internet Infrastructure Security

PennState

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

Z. Berkay Celik

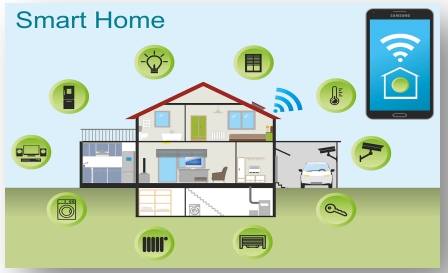
Gang Tan

Patrick McDaniel

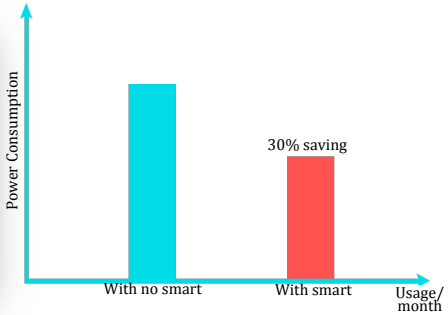
Penn State University

NDSS 2019

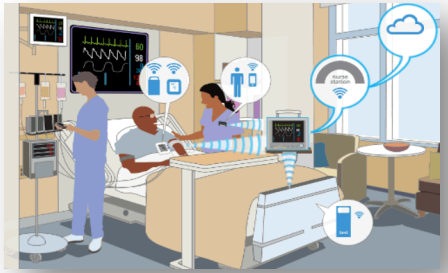
Internet of Things (IoT) enables the future



Smart Homes
Source: Samsung



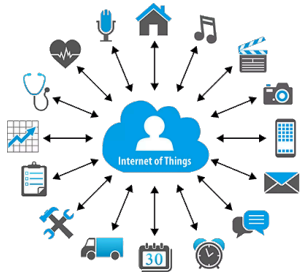
Smart Energy
Source: LG



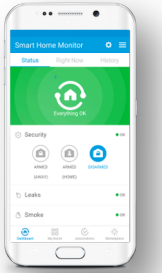
Healthcare
Source: John Hopkins

Smart Farms
Source: Microsoft

IoT is not magic



Connected devices



Mobile app

```
MQTT.sub(topicInLedA, function(conn, topic, msg) {
  print('Topic:', topic, 'message:', msg);
  if (msg === '0'){
    GPIO.write(pinLedA,0);
    isLedAOn = 0;
  } else {
    GPIO.write(pinLedA,1);
    isLedAOn = 1;
  }
}, null);


MQTT.sub(topicInLedB, function(conn, topic, msg) {
  print('Topic:', topic, 'message:', msg);
  if (msg === '0'){
    GPIO.write(pinLedB,0);
    isLedBOn = 0;
  } else {
    GPIO.write(pinLedB,1);
    isLedBOn = 1;
  }
}, null);
```

IoT application

Automation

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

ANDY GREENBERG SECURITY 07.21.15 6:00 AM

HACKERS REMOTELY KILL A JEEP ON THE HIGHWAY—WITH ME IN IT

Their code is an automaker's nightmare: software that lets hackers send commands through the Jeep's entertainment system to its dashboard functions, steering, brakes, and transmission, all from a laptop that may be across the country.

SMART HOME

Have a smart lock? Yeah, it can probably be hacked

While wandering in his neighborhood, he noticed a lot of Bluetooth locks popping up and decided to do some sniffing of those "security" gadgets (read: capturing packets being sent between devices). "I discovered plain-text passwords being sent that anybody could read.

Mirai botnet adds three new attacks to target IoT devices

This new version of the botnet uses exploits instead of brute force attacks to gain control of unpatched devices.

By Danny Palmer | May 18, 2018 -- 13:29 GMT (14:29 BST) | Topic: Security

All of these failures are traditional security problems:

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

IoT environment

IoT Devices

eclipse marketplace

Eclipse SmartHome > IoT (52)

IoT app market

IFTTT ifthisthenthat

Microsoft Flow

zapier

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

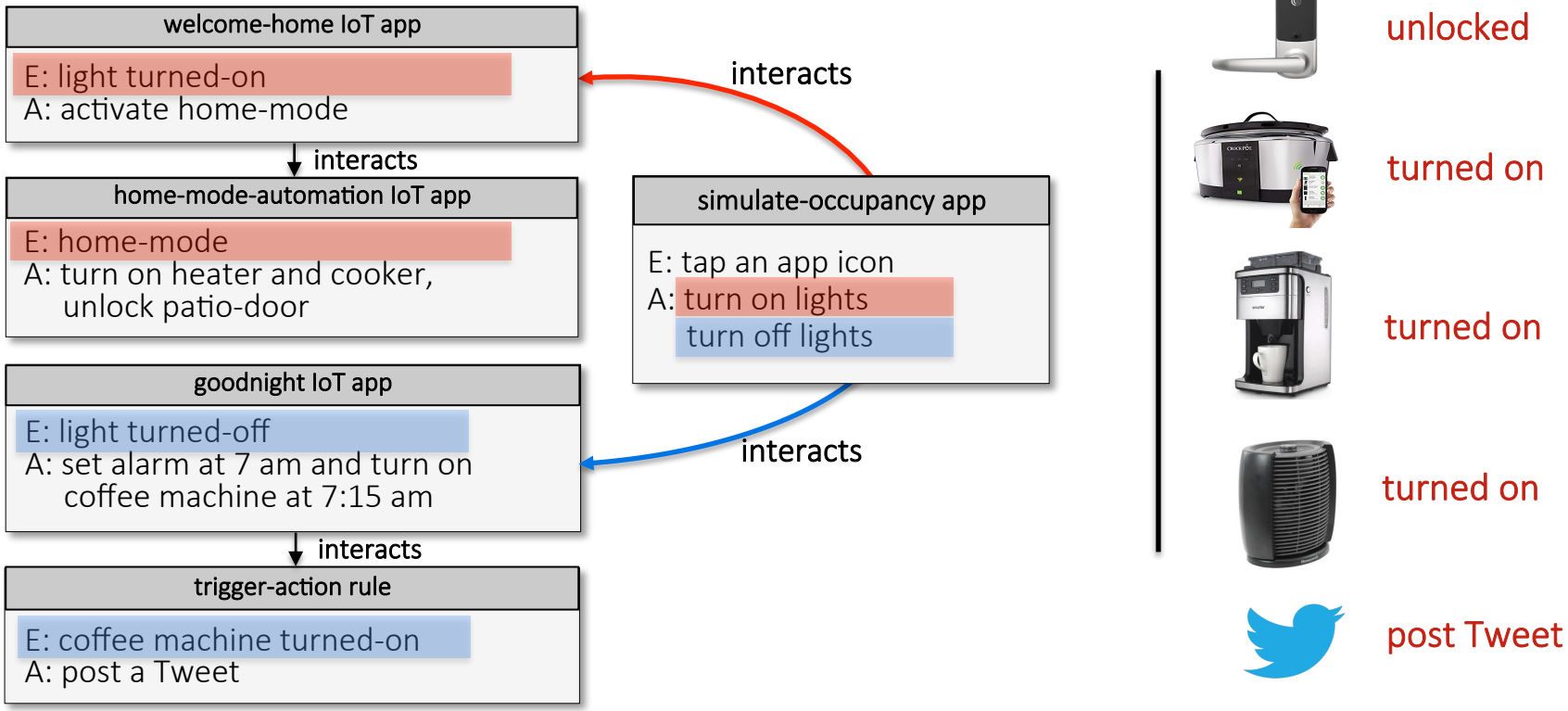
App to turn Lights On/Off automatically while away (Simulate Presence)

Mobile App

lmosenko 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

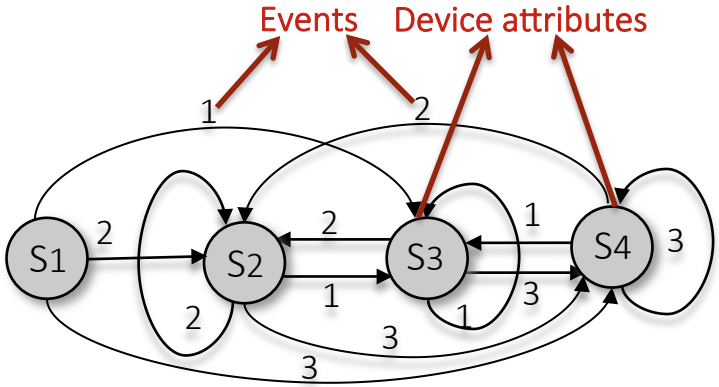


How can we prevent safety and security violations within IoT environments?

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**



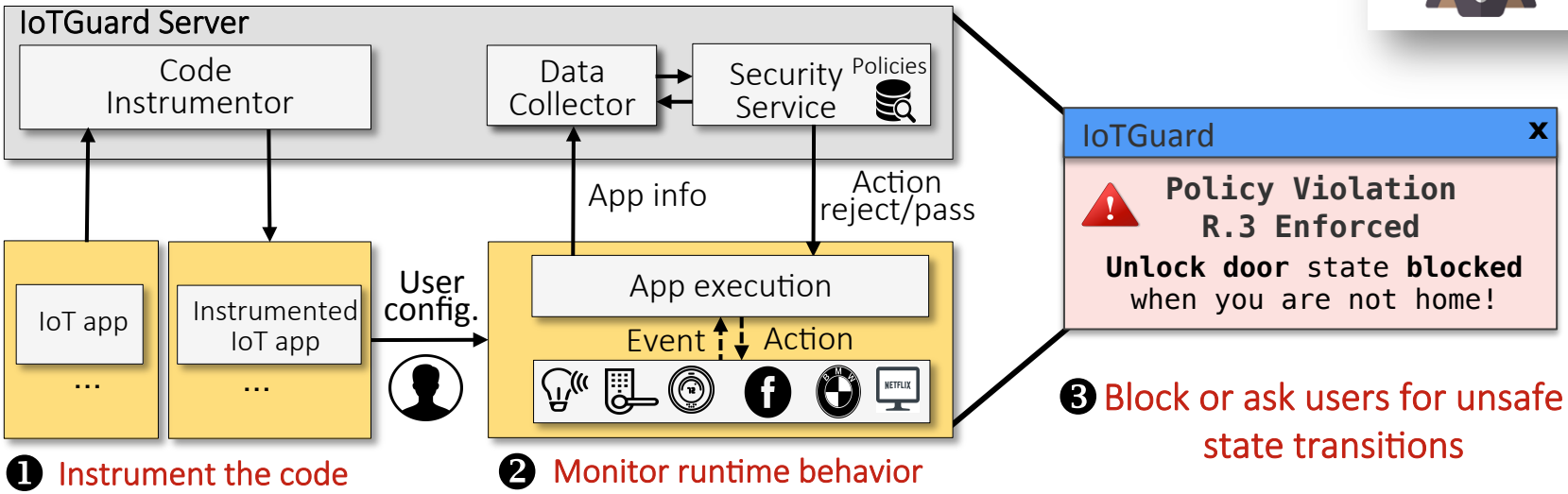
Modeling states and transitions of IoT devices

… 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



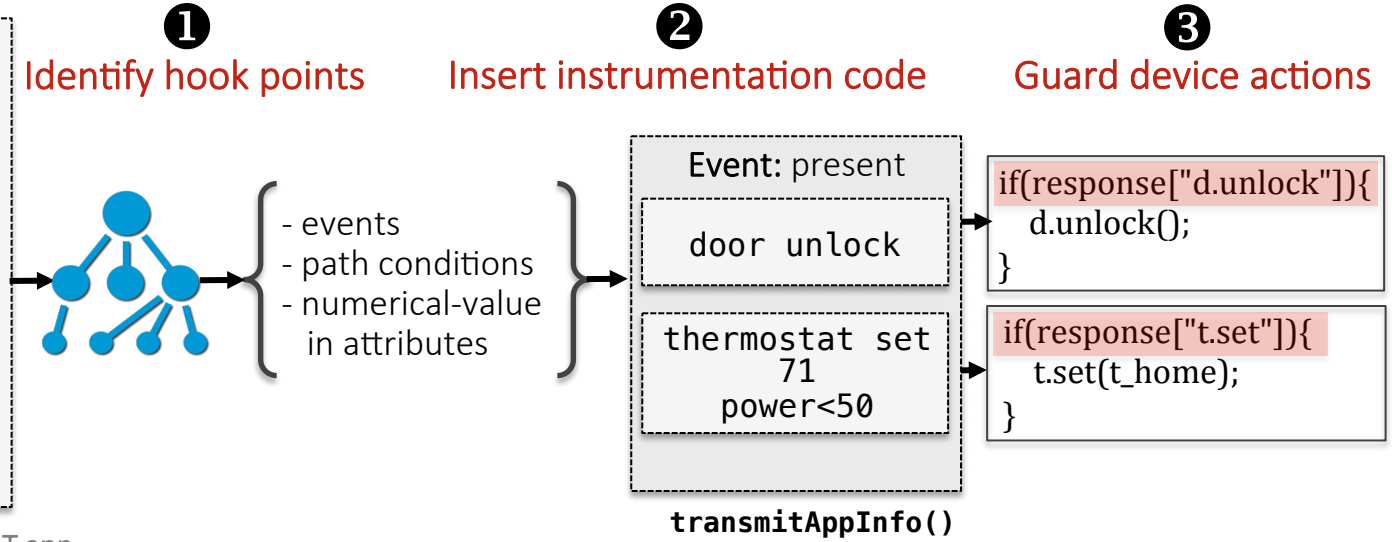
* We refer to IoT and trigger-action apps as IoT apps

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

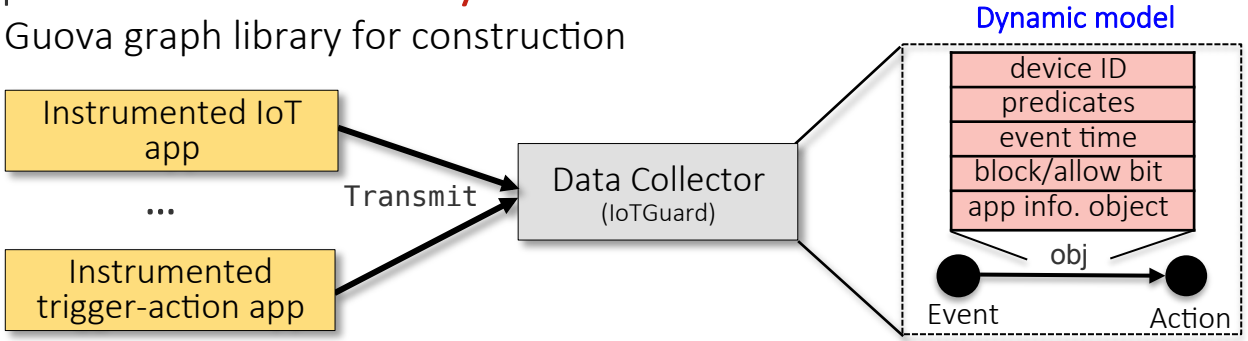
```
1:  // Devices
2:  presence sensor ps
3:  door d
4:  thermostat t
5:  power meter p

6:  when ps.present
7:    t_home=71; thold=50;
8:    d.unlock();
9:    if (p.power<thold){
10:     t.set(t_home);
11:   }
```

Source code of home-automation IoT app

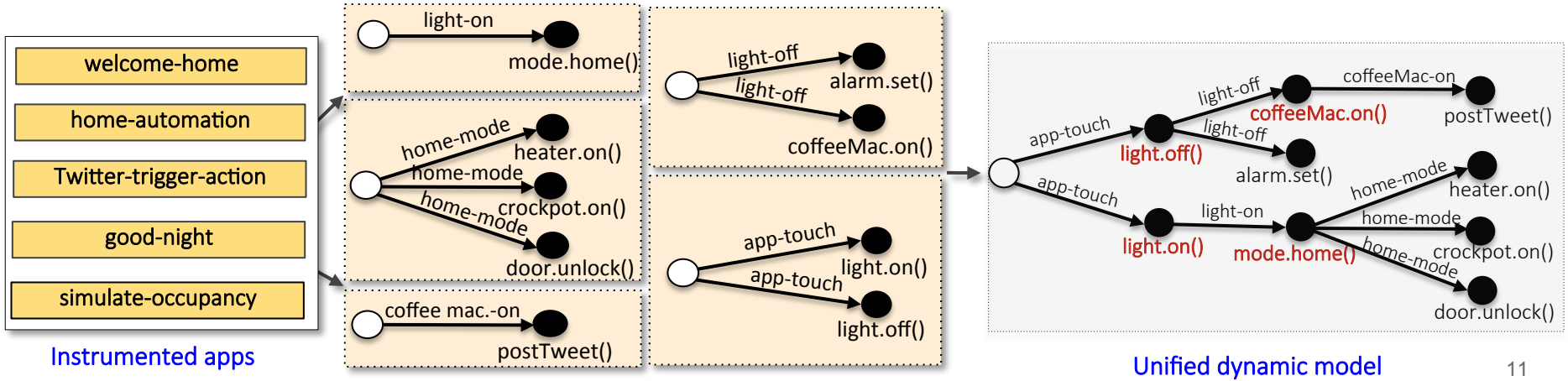


Data collector

- 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

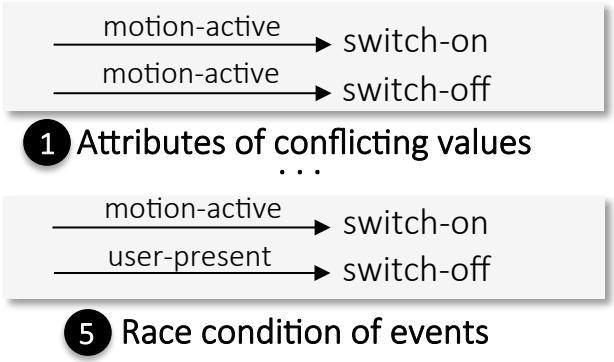


Security service - Property identification

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

General properties

- Constraints on states and transitions



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

. . .

30. 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

Integrity policy

Confidentiality policy

GPL: IoTGuard Policy Language

<policy-set> ::= [<statements>]

<statements> ::= <statement> `;' [<statements>]

<statement> ::= <restrict_clause> | <allow_clause>

<restrict_clause> ::= `restrict' `:' [<transitions>] `:' [<states>]

<allow_clause> ::= `allow' `:' [<transitions>] `:' [<states>]

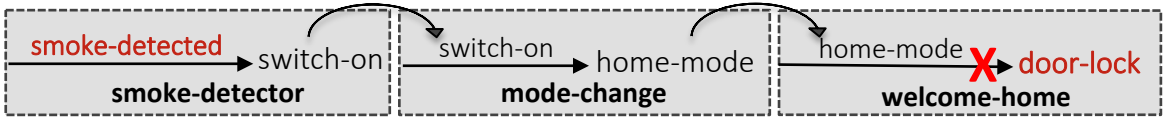
. . .

Overall IoTGuard checks an IoT environment against 36 identified policies

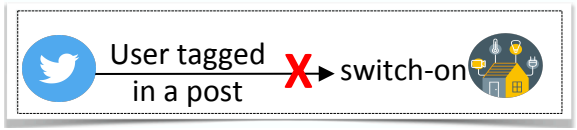
Violations in trigger-action apps: Surbatovich et al. (WWW'17), Celik et al. (arXiv'18)

Security service - Policy enforcement

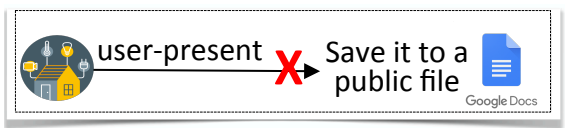
- Security service blocks undesired states before happening
  - Enforce policies by **exploring** their **reachability** and check **state labels** during exploration



Block door-lock state when there is a smoke at home

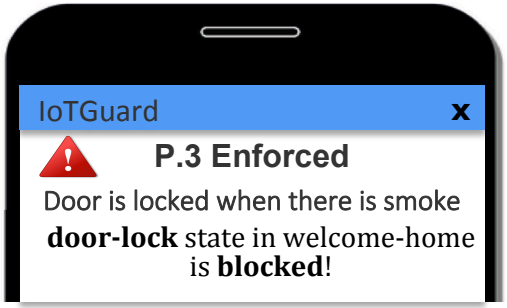


Integrity policy enforced

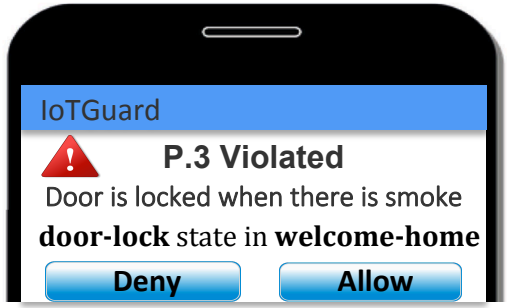


Confidentiality policy enforced

- Two solutions for policy enforcement



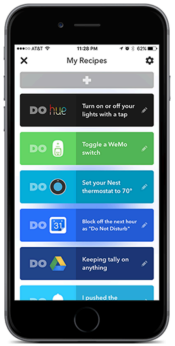
Automated blocking



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



1 Light switch(4)
2 Door lock
3 Presence sensor(2)
4 Motion sensor(3)
5 Contact sensor
6 Temp. measure.
7 AC
8 Heater
9 Coffee machine
10 Crockpot
11 Leak detector(2)
12 Fan
13 Power meter
14 Alarm(2)
15 Smoke detector(2)
16 Humidity sensor
17 Luminance sensor
18 Speakers
19 Window shade
20 Doorbell

- 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

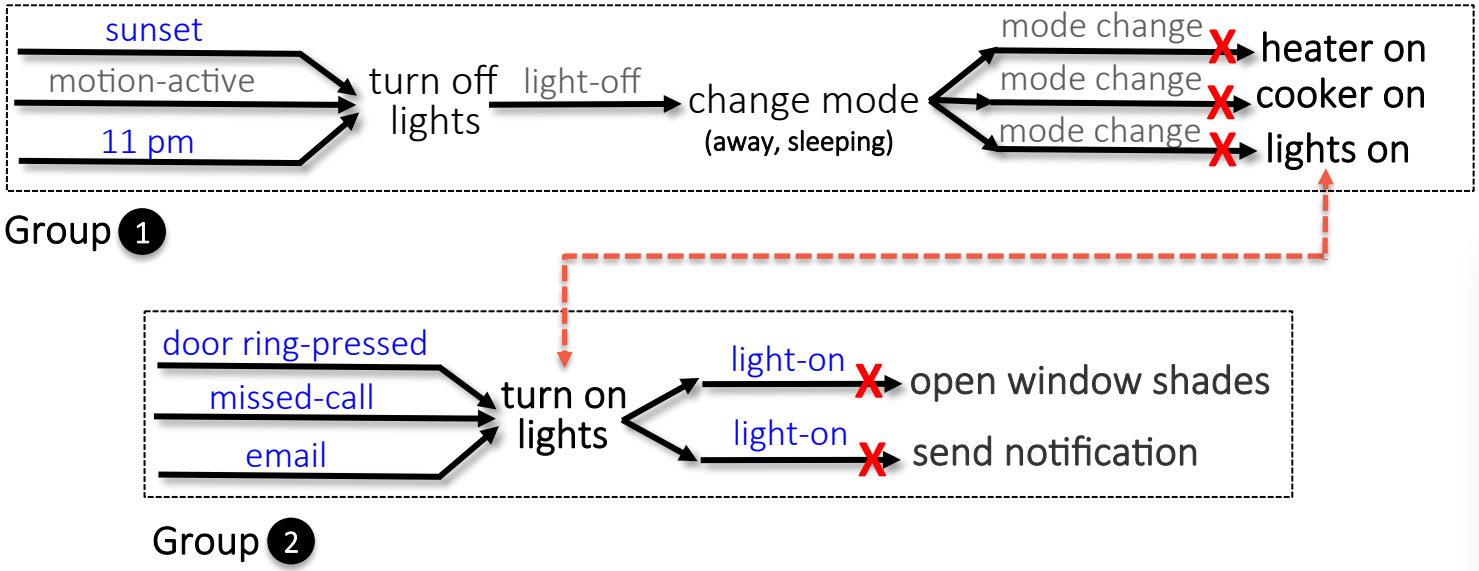
App ID	Violation Description	Policy	Blocked
ST4-ST7	The heater is turned on when user is not at home	R.13	X heater on
IFTTT5	The switch is turned on when someone Tweets a hashtag	S.1	X switch on
ST11-ST12	Heater and AC turned on at the same time	R.17	X AC on

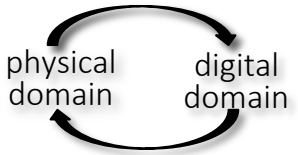
ST = SmartThings IoT apps IFTTT = Trigger-action apps

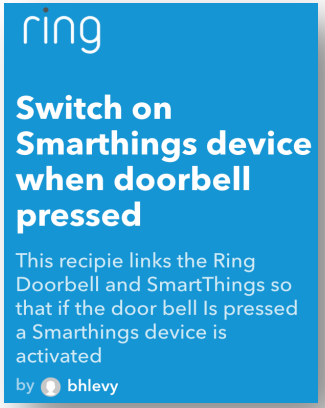
- Source of policy violations
  - R:13: Interactions through abstract attributes
  - S.1: Lack of app-vetting for trigger-action apps
  - R.17: Misconfiguration of numerical-valued device attributes

Policy enforcement in multi-apps

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



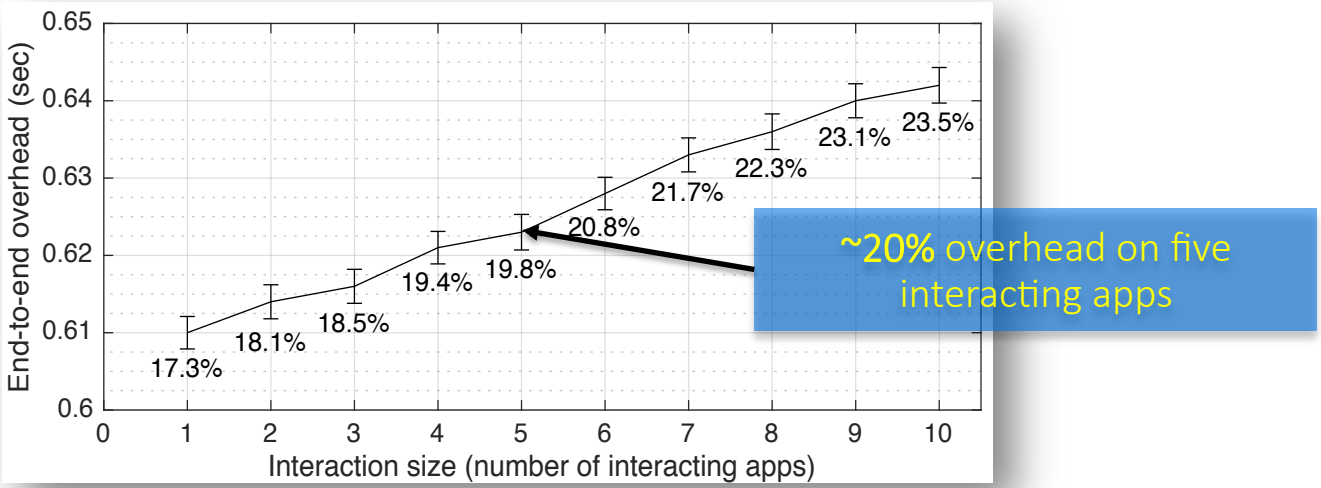




- Each group includes a set of interacting IoT and trigger-action apps

Performance

- Code instrumentation
  - **14±4** Lines of Code (LoC) added to the apps (+ **20** LoC for IoTGuard library)
  - **4.1±2** seconds to add instrumentation code
- Runtime latency



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

https://beerkay.github.io

@ZBerkayCelik

beerkay

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!