Sensitive Information Tracking in Commodity IoT

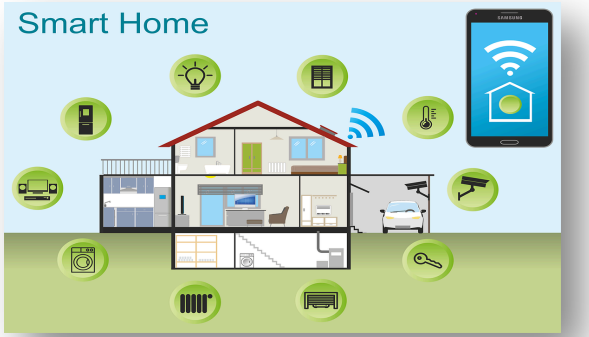
PennState

FIU | FLORIDA INTERNATIONAL UNIVERSITY

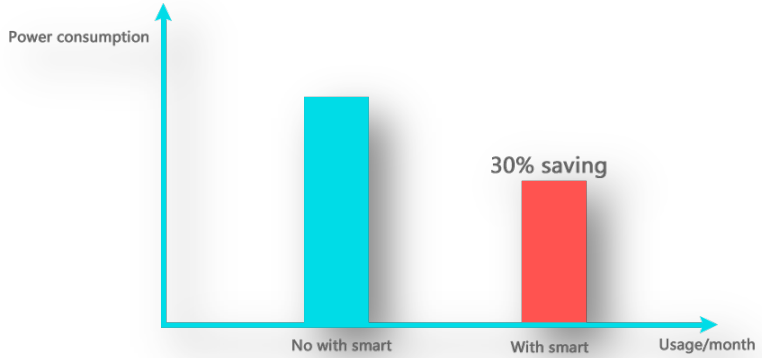
Z. Berkay Celik, Leonardo Babun, Amit K. Sikder, Hidayet Aksu,
Gang Tan, Patrick McDaniel, and Selcuk Uluagac

August 17th, 2018 @ USENIX Security

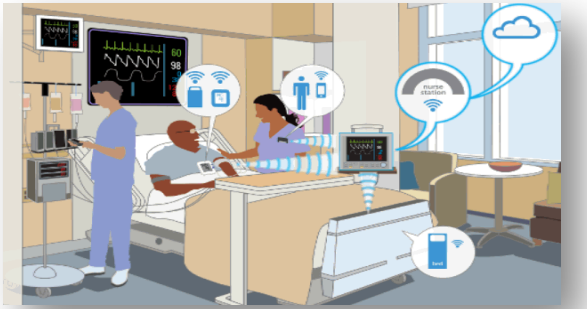
Internet of Things (IoT) enables the future



Smart Homes



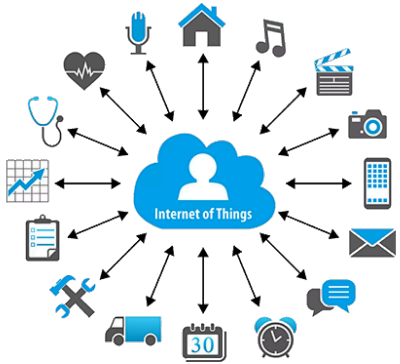
Smart Energy



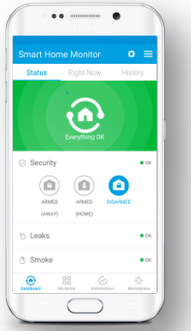
Healthcare

Smart Farms

IoT is not magic



Connected devices



Mobile app

Automation

```
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

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

IoT Devices Are Hacking Your Data & Stealing Your Privacy - Infographic

Ken Savage | 01.11.17 | iot, Infographic

Alexa beware! New smart home tests reveal serious privacy flaws

By Sandra Vogel - February 28, 2018

"Issues such as the fear of oversharing of data by commercial services, insufficient protection of stored personal data, and the possibility of interception of digital traffic by cybercriminals [are] significant."

When you live home …

Whether the door is locked or not…

Sensitive data in IoT apps

Problem: Users lack visibility into who sees their sensitive information

- Look inside of IoT apps to determine how they use privacy sensitive data
  - Device states
  - Device information
  - User inputs
  - Location (physical and geo-location)

SAMSUNG SmartThings™

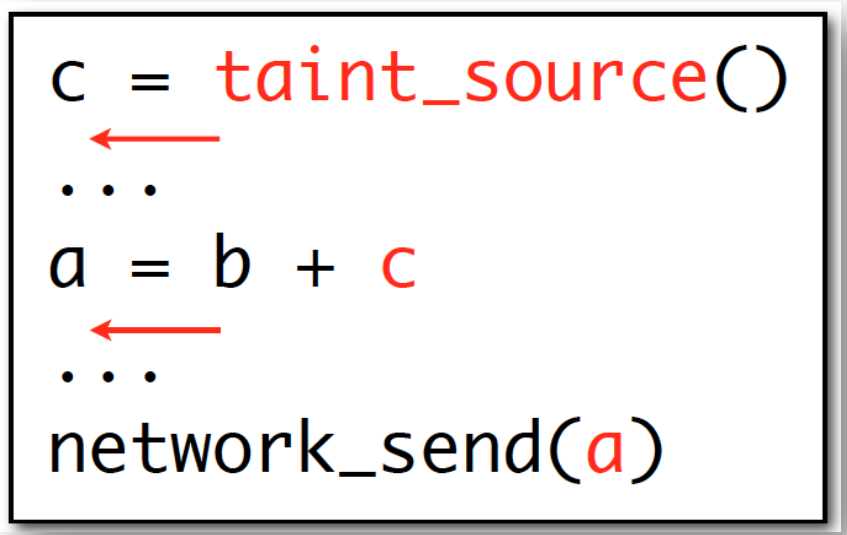
openHAB empowering the smarthome

Works with Apple HomeKit

Static taint analysis

Goal: Analyze app source code to determine when privacy sensitive information leaves the IoT app

- Static taint analysis is a technique that tracks information dependencies from an origin
- Conceptual idea:
  - Taint source
  - Taint propagation
  - Taint sink



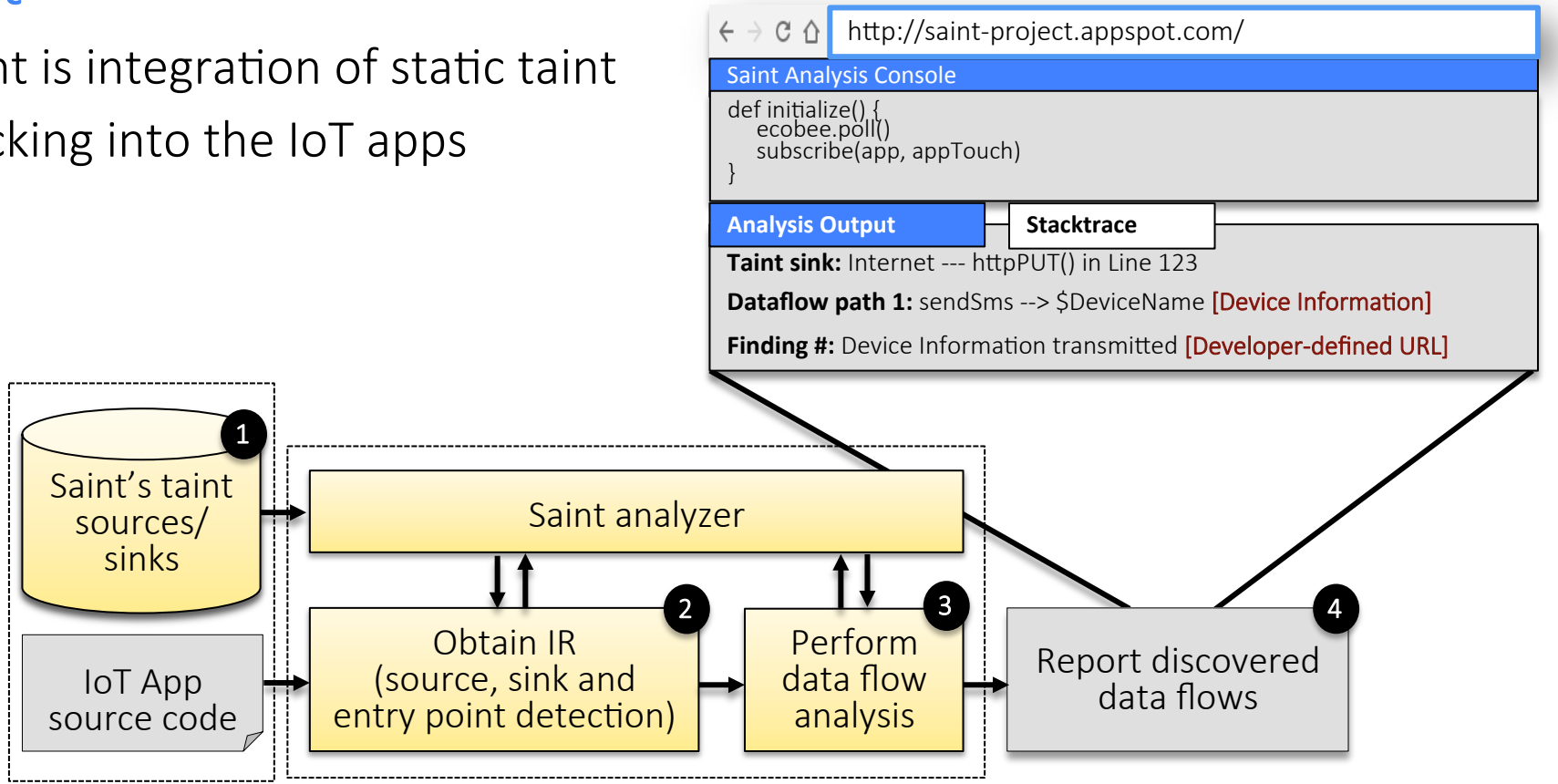
Challenges

- IoT programming platforms are diverse
- Identifying sensitive sources in IoT apps is quite subtle
- Each IoT platform has its idiosyncrasies that require special treatment

- Current data tracking tools are insufficient to address these challenges

Saint

- Saint is integration of static taint tracking into the IoT apps



From app source code to IR

Devices

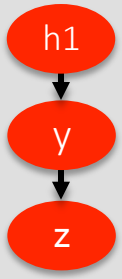
```
input (p, presenceSensor, type:device)
input (s, switch, type:device)
input (d, door, type:device)
input (toTime, time, type:user_defined)
input (fromTime, time, type:user_defined)
input (c, contact, type:user_defined)
```

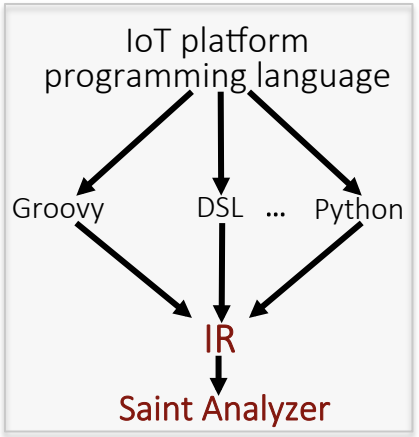
Events

```
subscribe(p, “present”, h1)
```

Computation

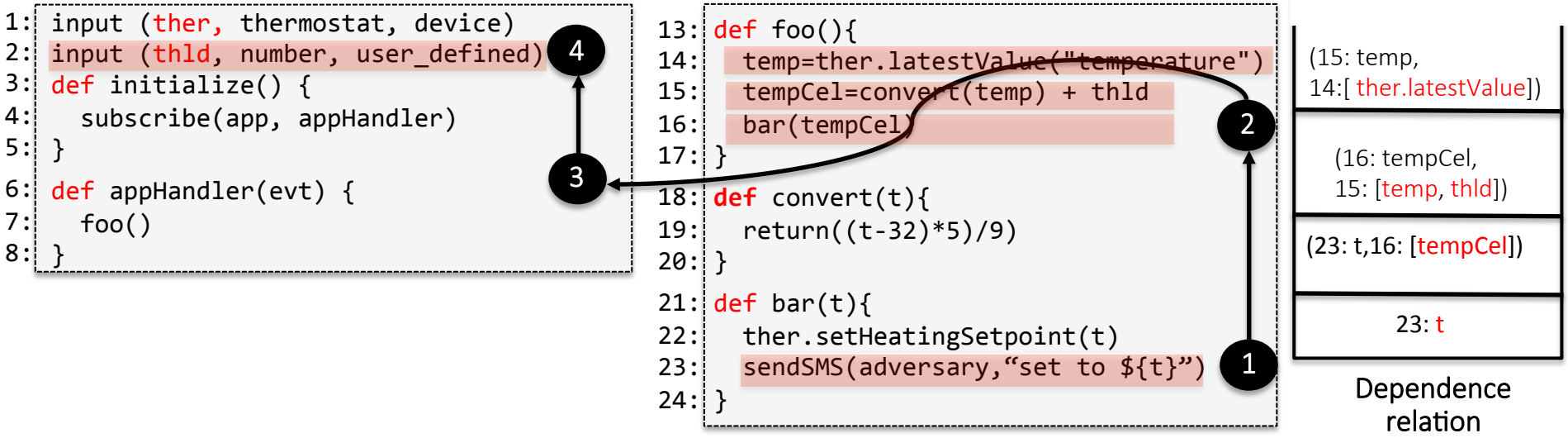
```
h1(){
        s.on()
        d.unlock()
        def between= y()
        if (between){
            z()
        }
}
y(){
        return timeOfDayIsBetween(fromTime, toTime)
}
z(){
        sendSms(c, “...”)
}
```





Backward taint tracking

- Identify sensitive data flow paths



Analysis Sensitivity and Implicit Flows

- Path-sensitivity
  - Collects the evaluation results of the predicates
  - Discards infeasible paths
- Context-sensitivity
  - Implements depth-one call-site sensitivity
  - Discards paths not matching calls and returns
- Implicit flows
  - Determines whether predicates at conditional branches depends on a tainted value
  - Taints all elements in the conditional branch

Algorithms for IoT-specific idiosyncrasies

- On-demand algorithms for analysis precision
  - State variables
    - Field-sensitive analysis
  - Web service apps
    - Allows external entities to access devices
  - Call by reflection
    - Add all methods as possible call targets

```
counter=state.switchCounter //state variable
if (counter){ device actions}
```

```
mappings { // web-service apps
  path("/switches") {
    action: [GET: "listSwitches"] }
def listSwitches() {
  return it.currentValue("switch")
}
```

```
"$methodName"() // call by reflection

def foo() { // add as possible call target }
def bar() { // add as possible call target }
```

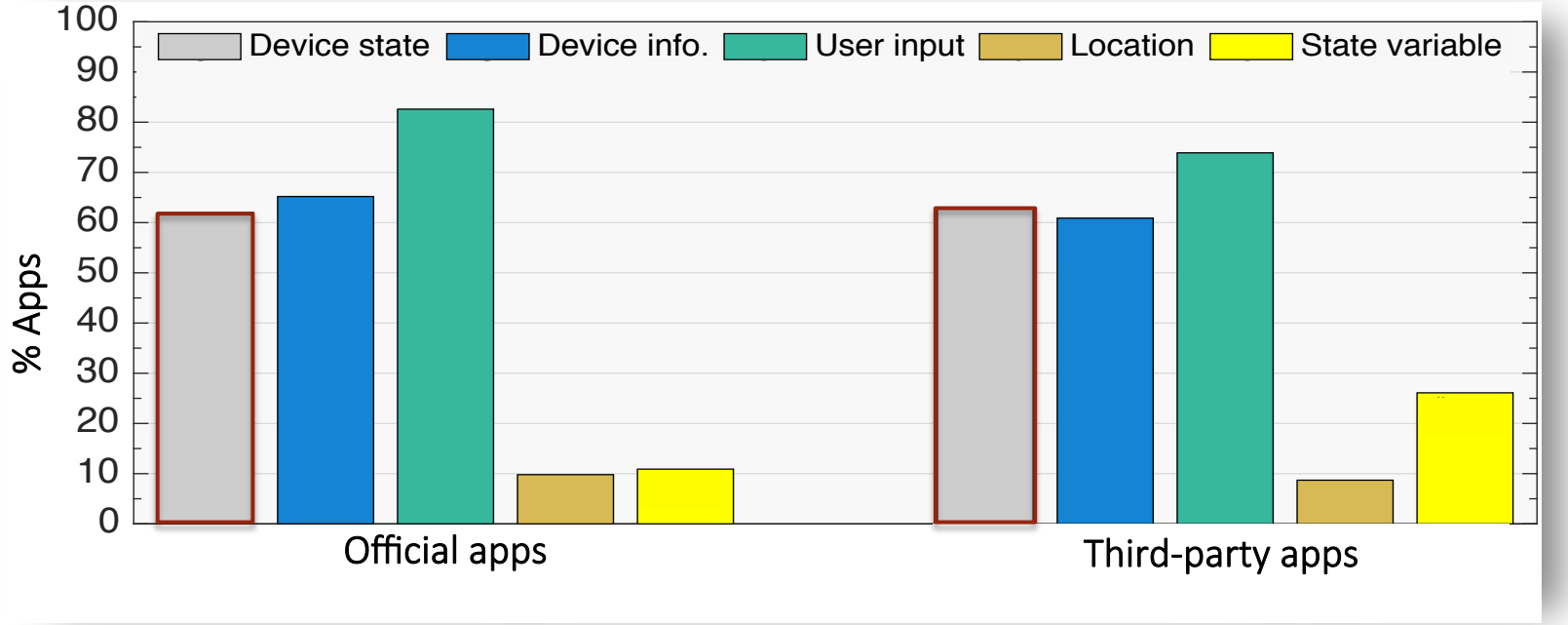
Application Study

- Implemented Saint for SmartThings IoT platform
- Selected **168** official and **62** third-party market apps
- **92** official and **46** third-party apps expose at least one kind of sensitive data

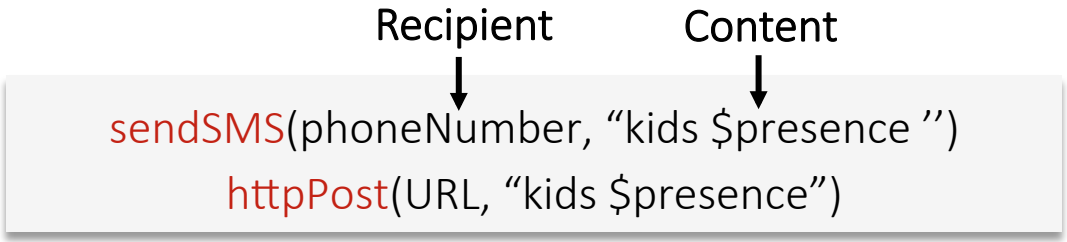
Apps	Internet	SMS	Both	Total
Official	24	63	5	**92**
Third-party	10	36	-	**46**

Application Study

What type of privacy-sensitive information leaves IoT apps?



Who sees privacy-sensitive information?



		Recipient defined by			Content defined by		
Taint Sinks	Apps	User	Developer	External	User	Developer	External
Messaging	Official	154	0	0	5	149	0
	Third-party	67	0	0	4	63	0
Internet	Official	2	48	44	0	54	40
	Third-party	0	13	12	0	13	12

Summary

- Introduced Saint, a static analysis tool that identifies sensitive data flows in IoT apps
- Evaluated Saint on 230 SmartThings apps
- Found %60 of the analyzed apps includes sensitive data flows
- Consumers and developers can use Saint to identify potential privacy risks
- Saint console is available: http://saint-project.appspot.com/

https://github.com/IoTBench/

IoTBench-test-suite

A micro-benchmark suite to assess the effectiveness of tools designed for IoT apps

iot-platform | smartthings | openhab | malicious-behaviors | data-leaks

Groovy ★ 10 2 Updated on May 12

V.1.0.1 Released May 2018

IoTBench

27 data leaks

28 security/safety violations

15 attacks migrated from mobile phone security

500+ official and third party apps

https://beerkay.github.io
@ZBerkayCelik
berkaycelik

https://github.com/LeoBabun
@BabunLeo
leonardo-babun

Thank you for listening!