HoMonit: Monitoring Smart Home Apps from Encrypted Traffic

Wei Zhang[1], Yan Meng[1], Yugeng Liu[1], Xiaokuan Zhang[2], Yinqian Zhang[2], and Haojin Zhu[1]

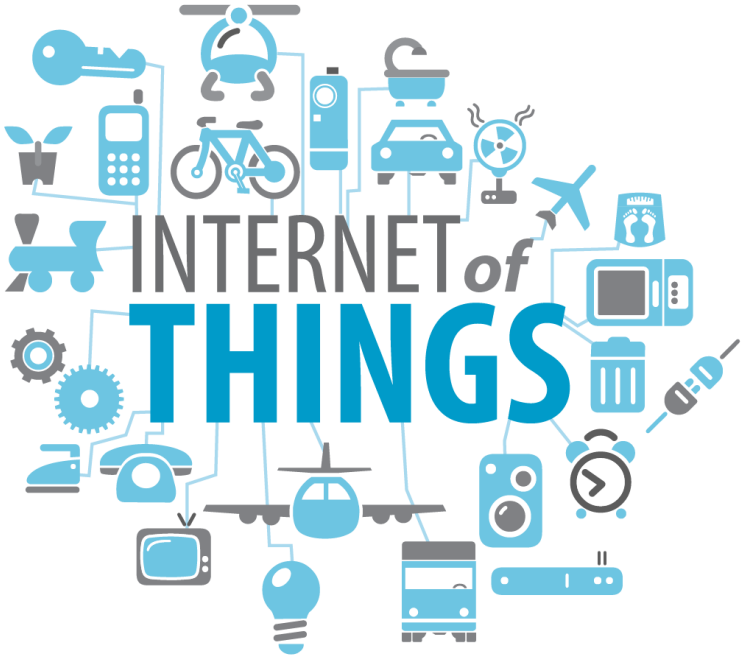
[1]Shanghai Jiao Tong University

[2]The Ohio State University

January, 2019

Background

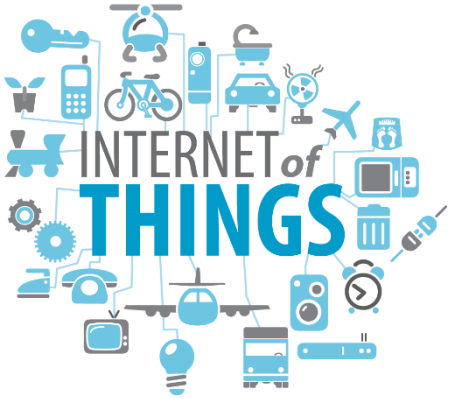
- **Smart home**, a concept of *Internet of Things* (IoT)



Internet of Things

Background

- **Smart home**, a concept of *Internet of Things* (IoT)

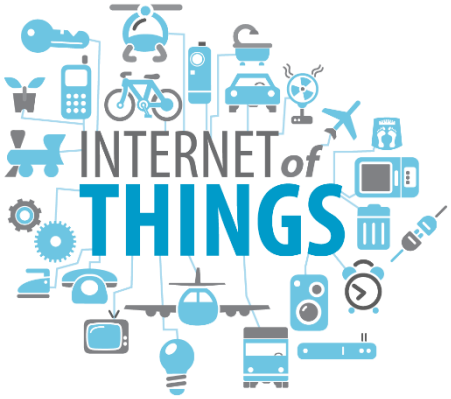


Internet of Things

Smart Home

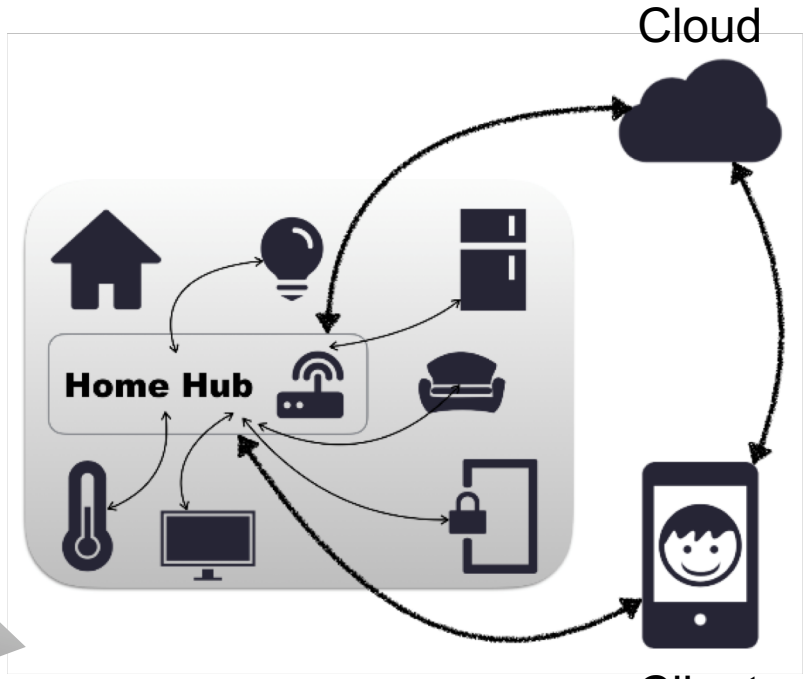
Background

- **Smart home**, a concept of *Internet of Things* (IoT)



Internet of Things

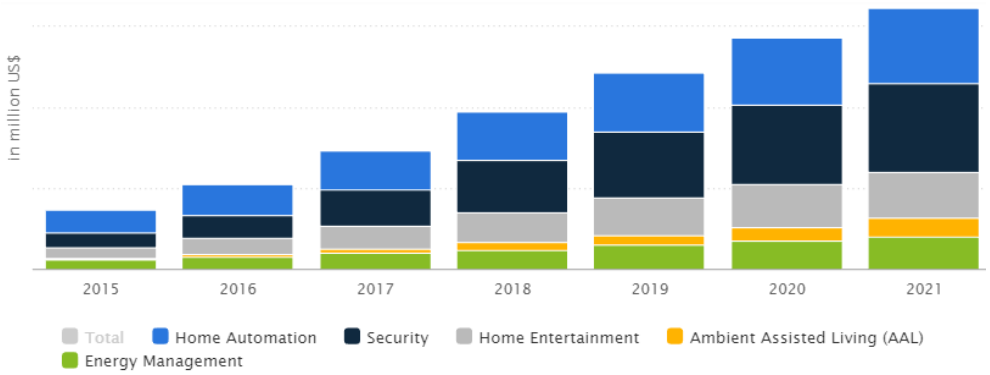
Smart Home



Control multiple devices and let them work together

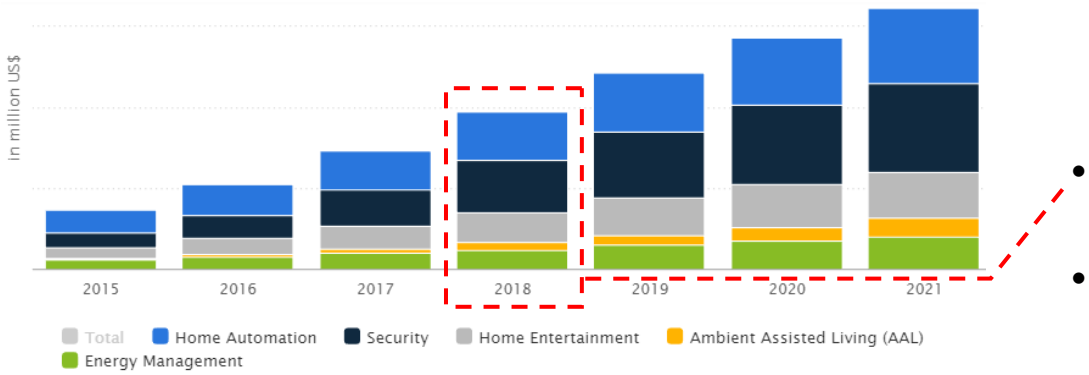
Background

- Rapid growth of **smart home** market



Background

- Rapid growth of **smart home** market

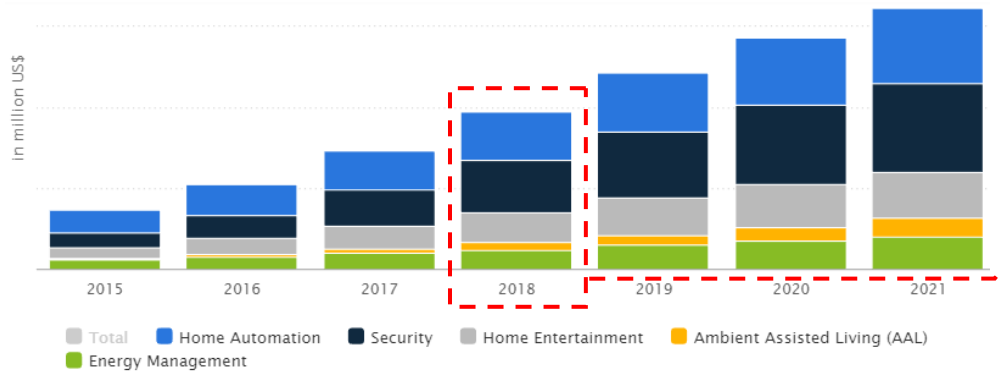


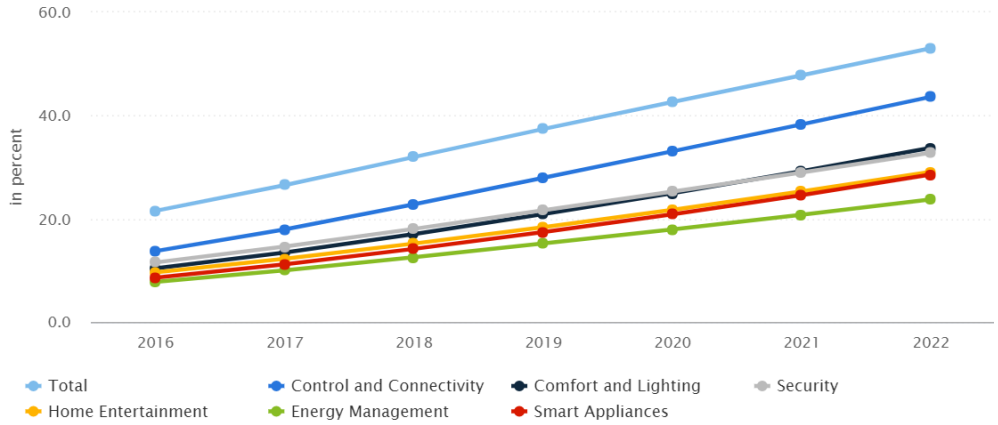
Market Revenue

- $18,877 million in 2018
- Annual growth rate: 14.8%

Background

- Rapid growth of **smart home** market



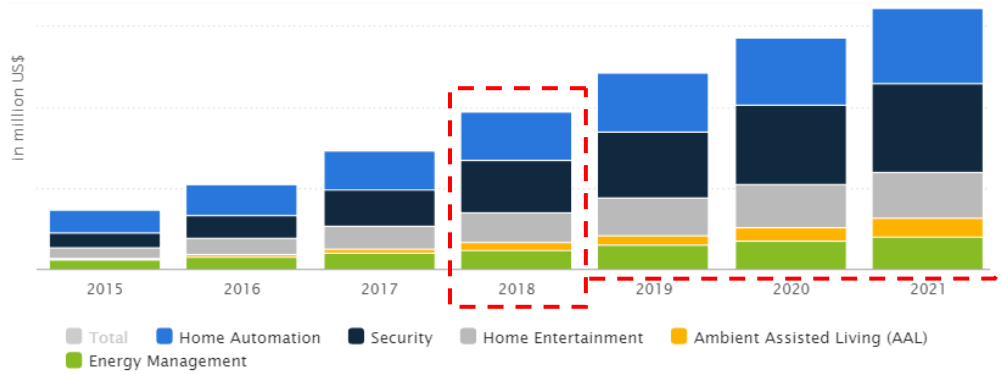


Market Revenue

- $18,877 million in 2018
- Annual growth rate: 14.8%

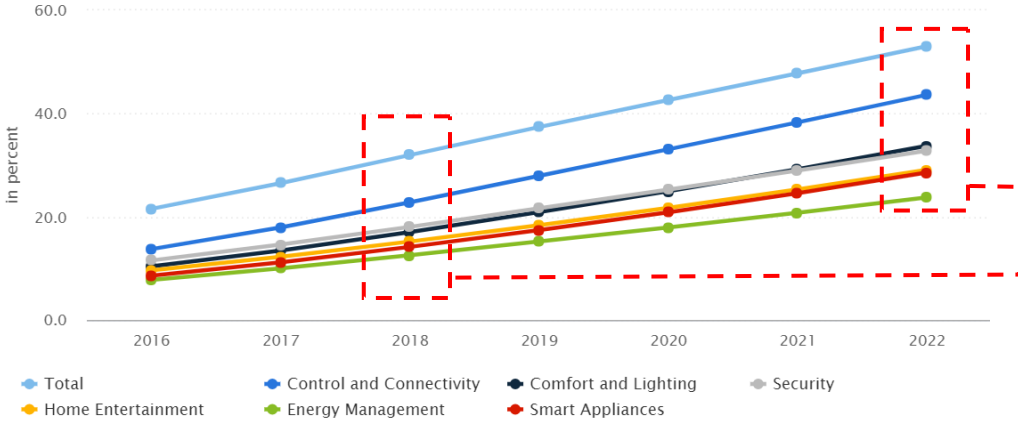
Background

- Rapid growth of **smart home** market



Market Revenue

- $18,877 million in 2018
- Annual growth rate: 14.8%



Household Penetration

- 32.0% in 2018
- Expected to hit 53.1% by 2022

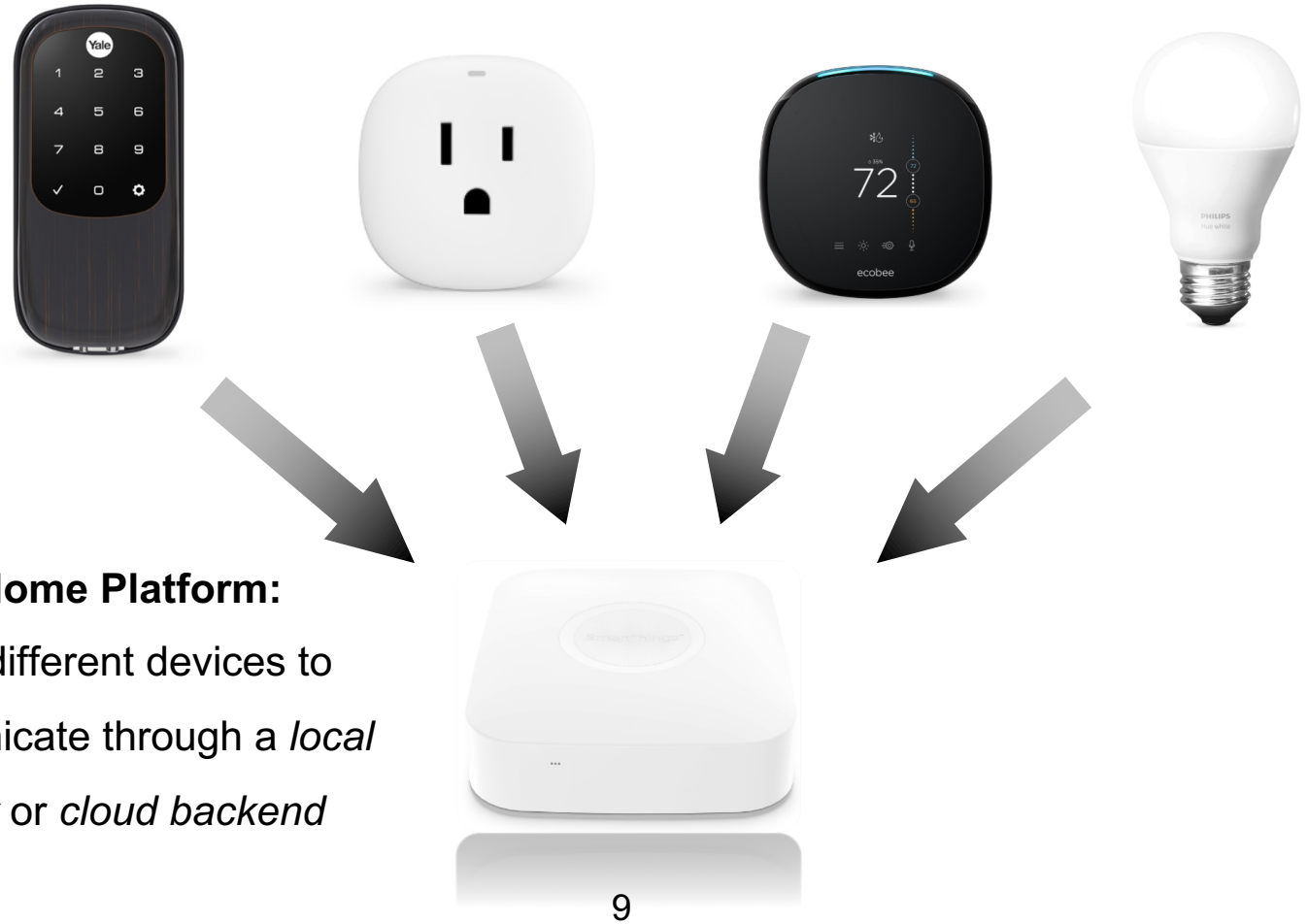
* Source: Statista, 2018

Smart Home Platforms

- Smart devices are produced by different vendors


Smart Home Platforms

- Smart devices are produced by different vendors

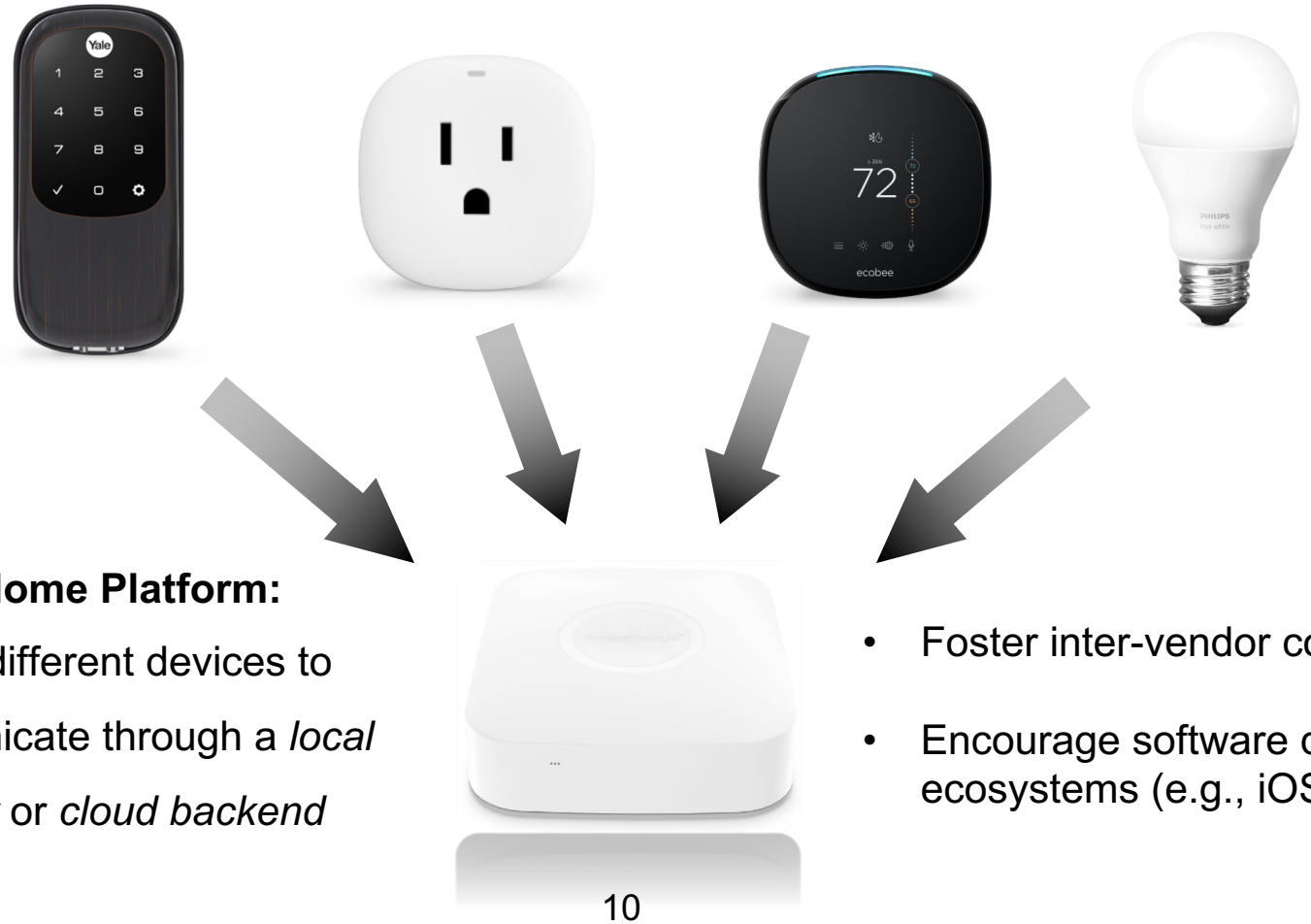


Smart Home Platform:

Enable different devices to communicate through a *local gateway* or *cloud backend*

Smart Home Platforms

- Smart devices are produced by different vendors



Smart Home Platform:

Enable different devices to communicate through a *local gateway* or *cloud backend*

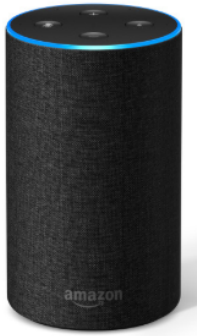
- Foster inter-vendor compatibility
- Encourage software development ecosystems (e.g., iOS App Store)

Smart Home Platforms

- Prominent examples of smart home platforms

Google Home

Apple HomeKit



Amazon Echo

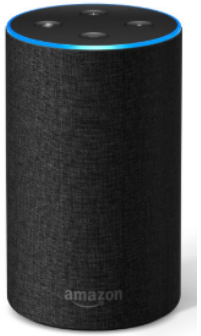
MiHome

Smart Home Platforms

- Prominent examples of smart home platforms

Google Home

Apple HomeKit



Amazon Echo

MiHome

SmartThings

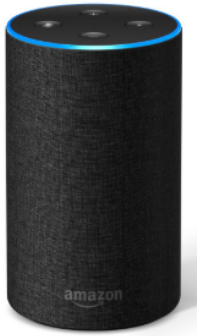
Platform to study in our work

Smart Home Platforms

- Prominent examples of smart home platforms

Google Home

Apple HomeKit



Amazon Echo

MiHome

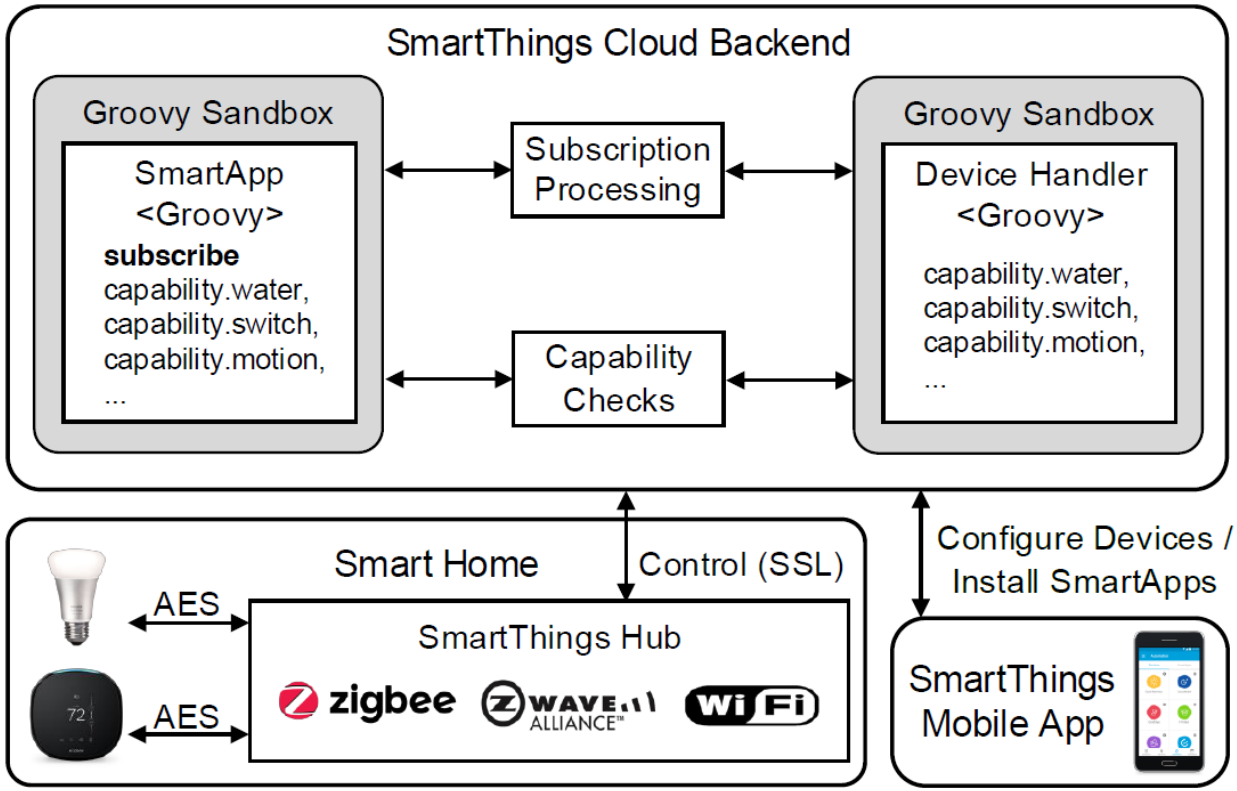
SmartThings

Platform to study in our work

- A popular platform promoted by Samsung
- Support 133 device types and 181 SmartApps in the official GitHub repo
- At least 50 thousand active families (2017)

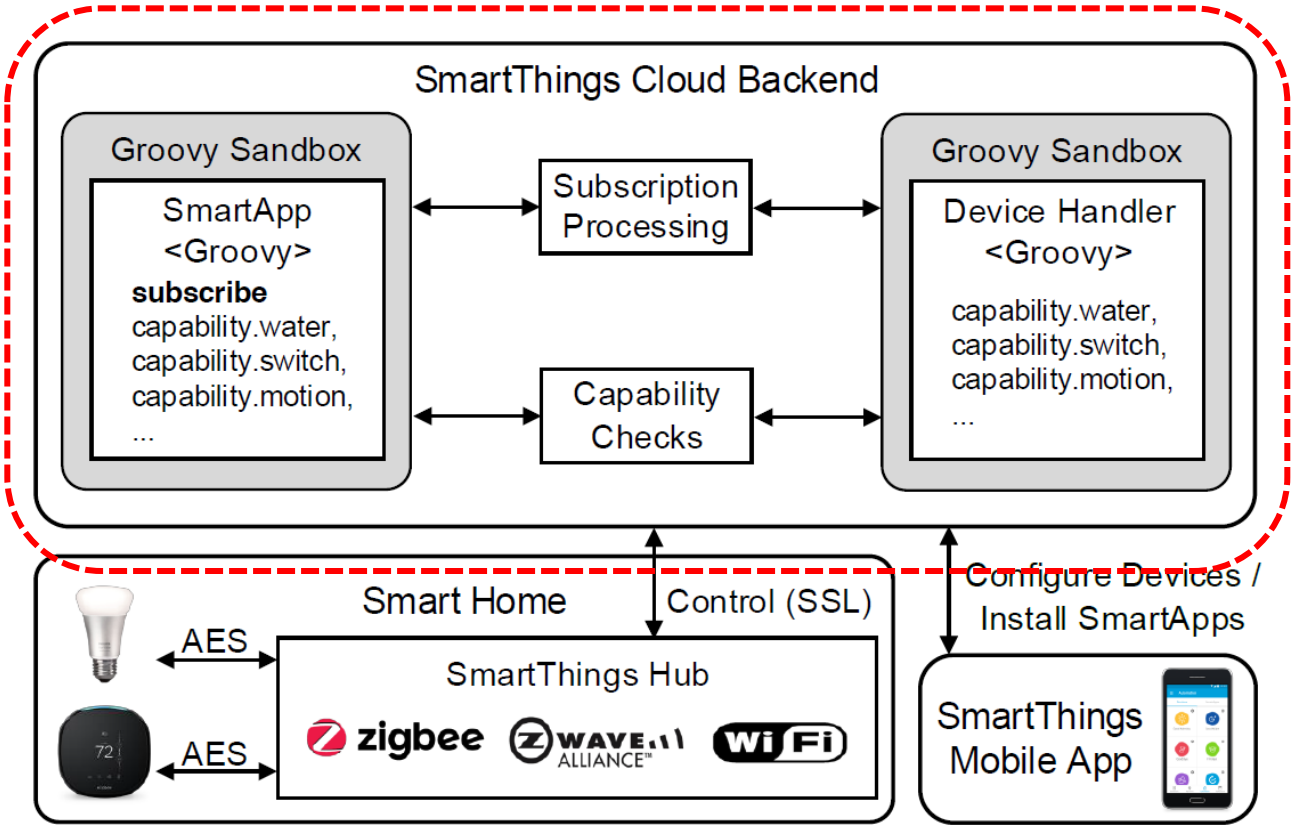
SmartThings

- SmartThings architecture



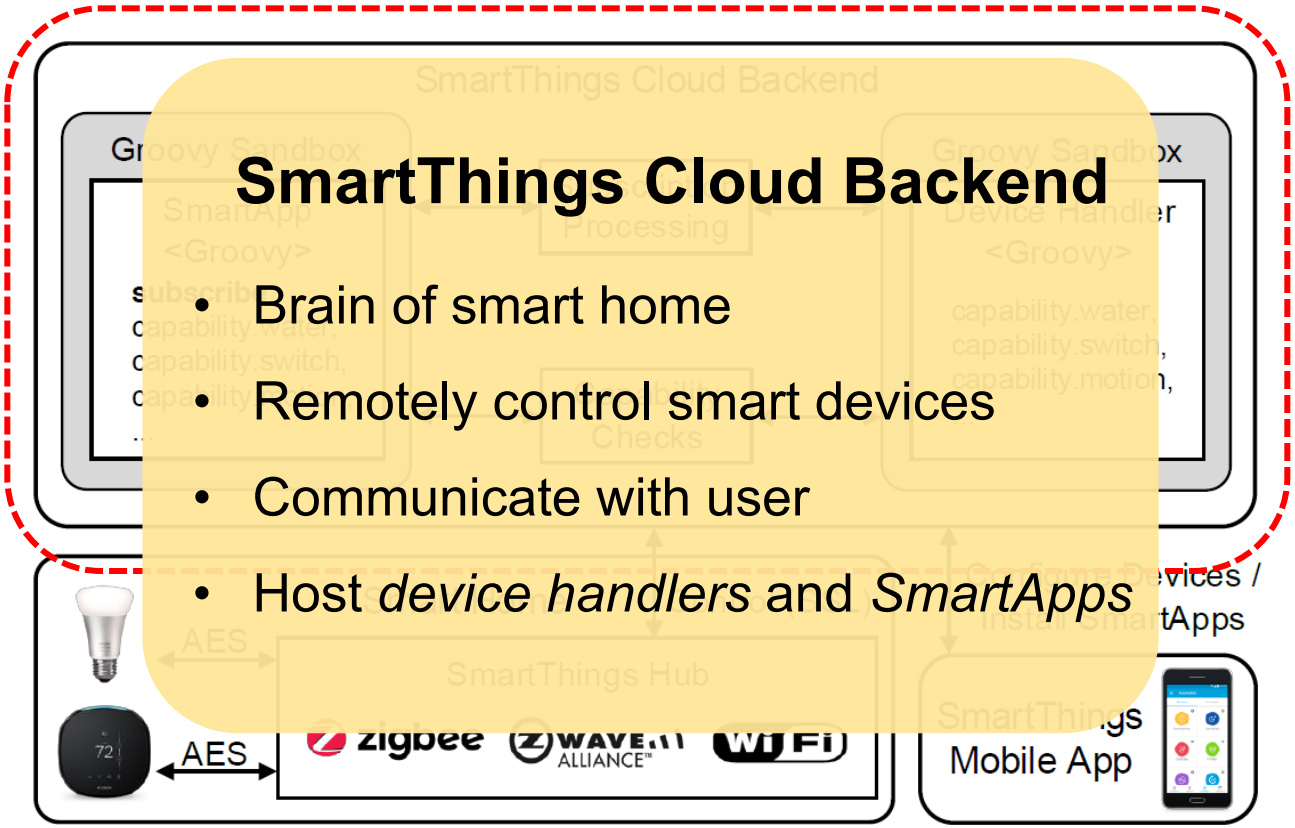
SmartThings

- SmartThings architecture



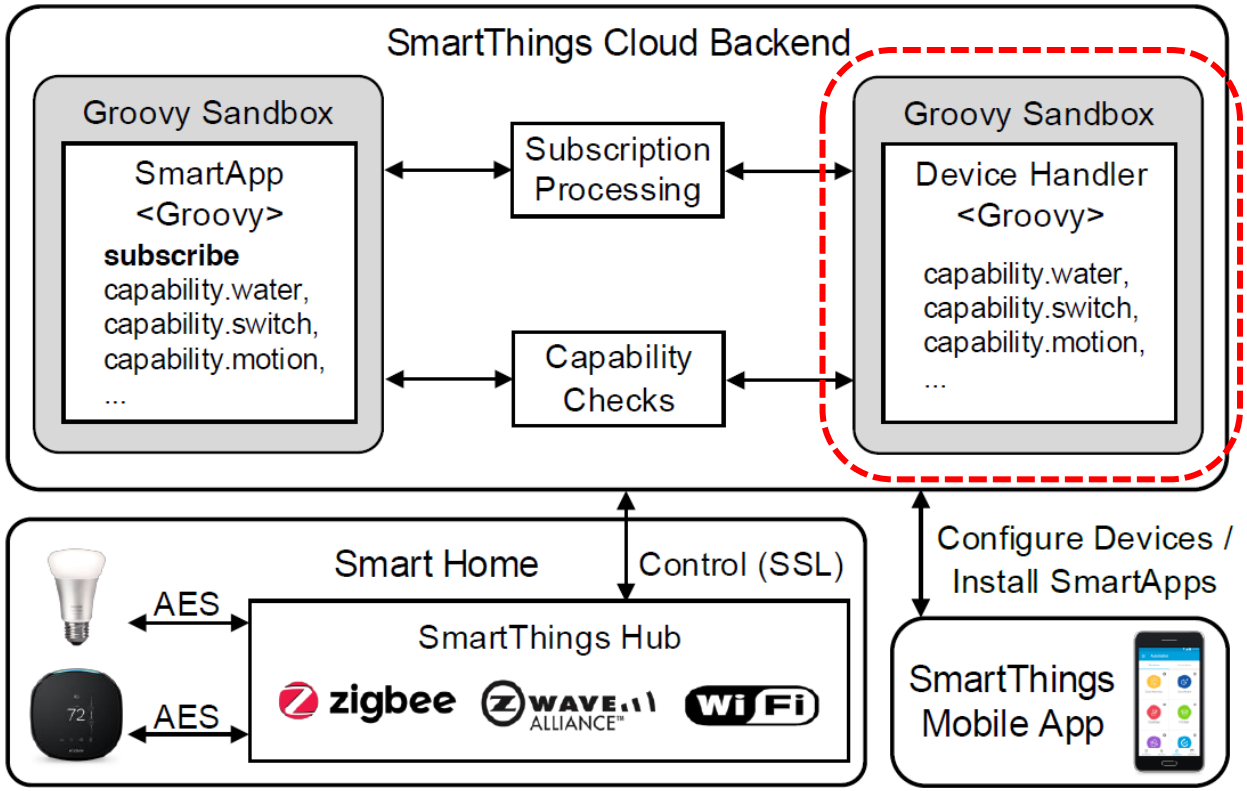
SmartThings

- SmartThings architecture



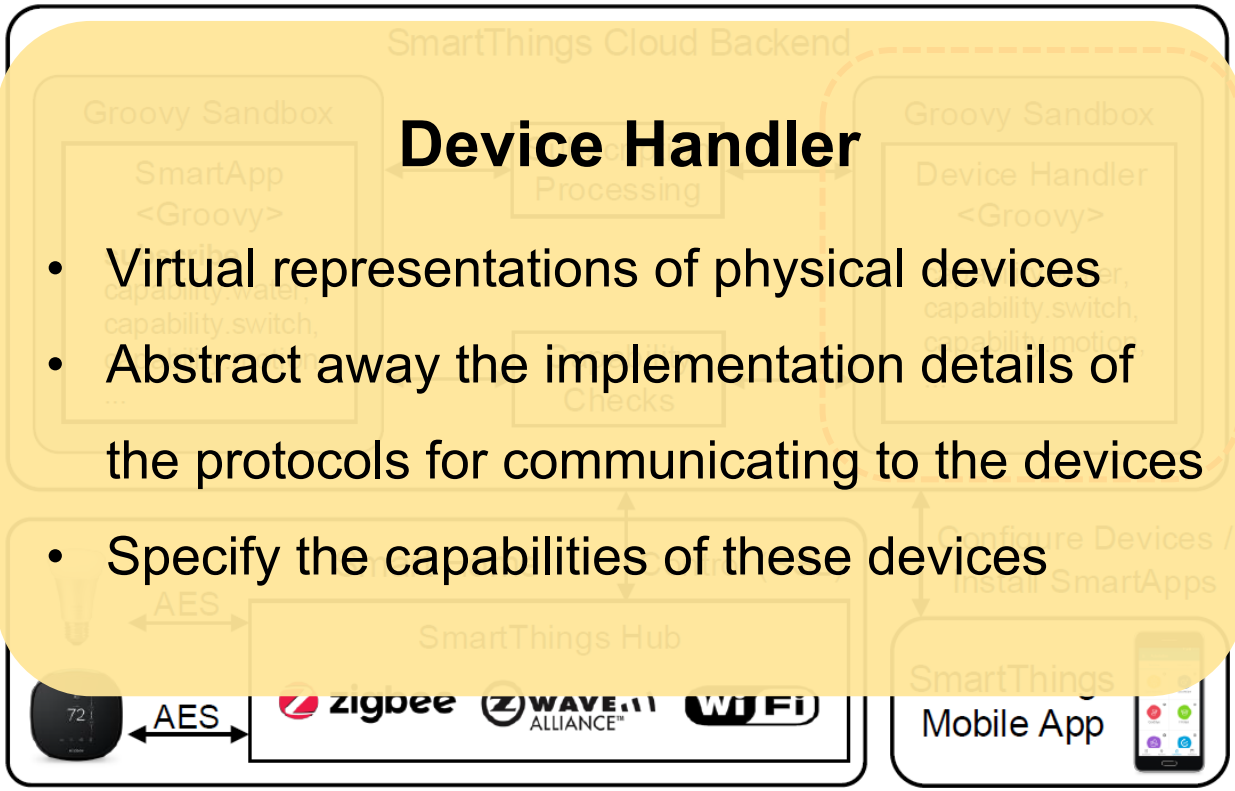
SmartThings

- SmartThings architecture



SmartThings

- SmartThings architecture

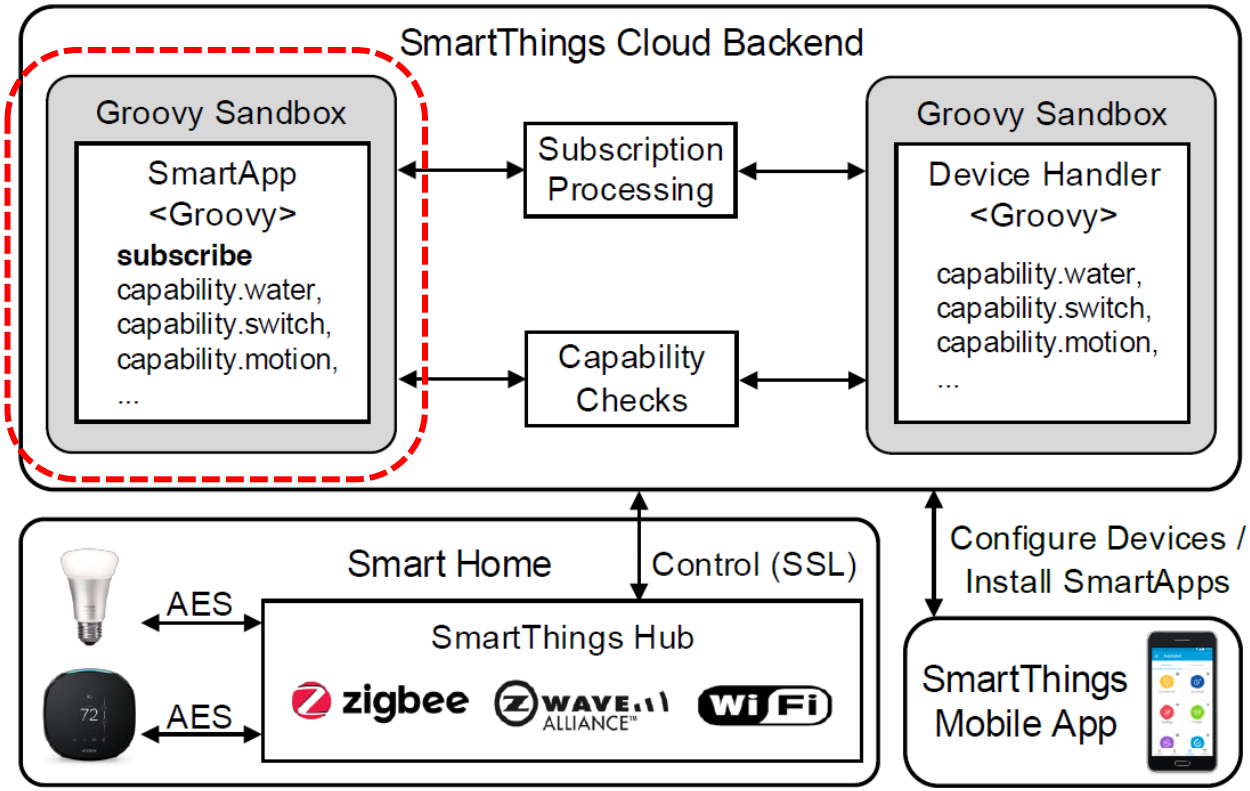


Device Handler

- Virtual representations of physical devices
- Abstract away the implementation details of the protocols for communicating to the devices
- Specify the capabilities of these devices

SmartThings

- SmartThings architecture

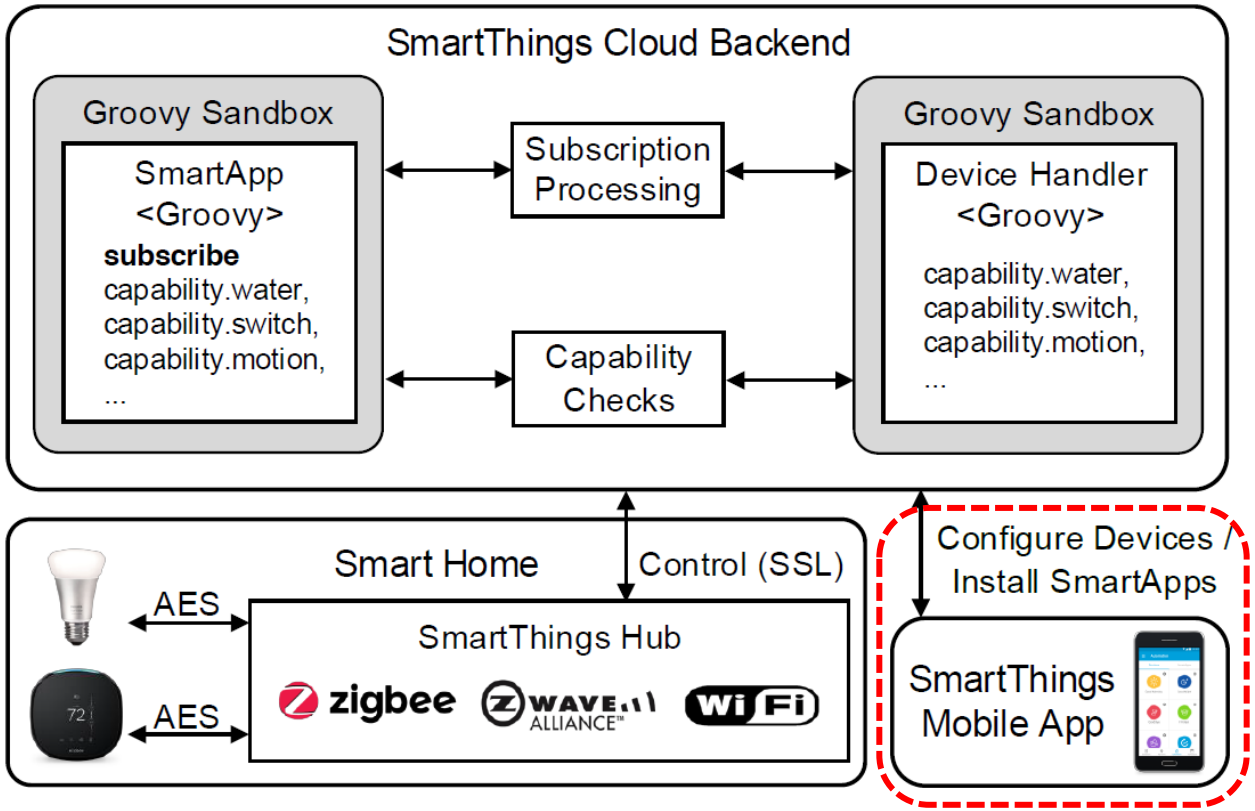


SmartThings

- SmartThings architecture

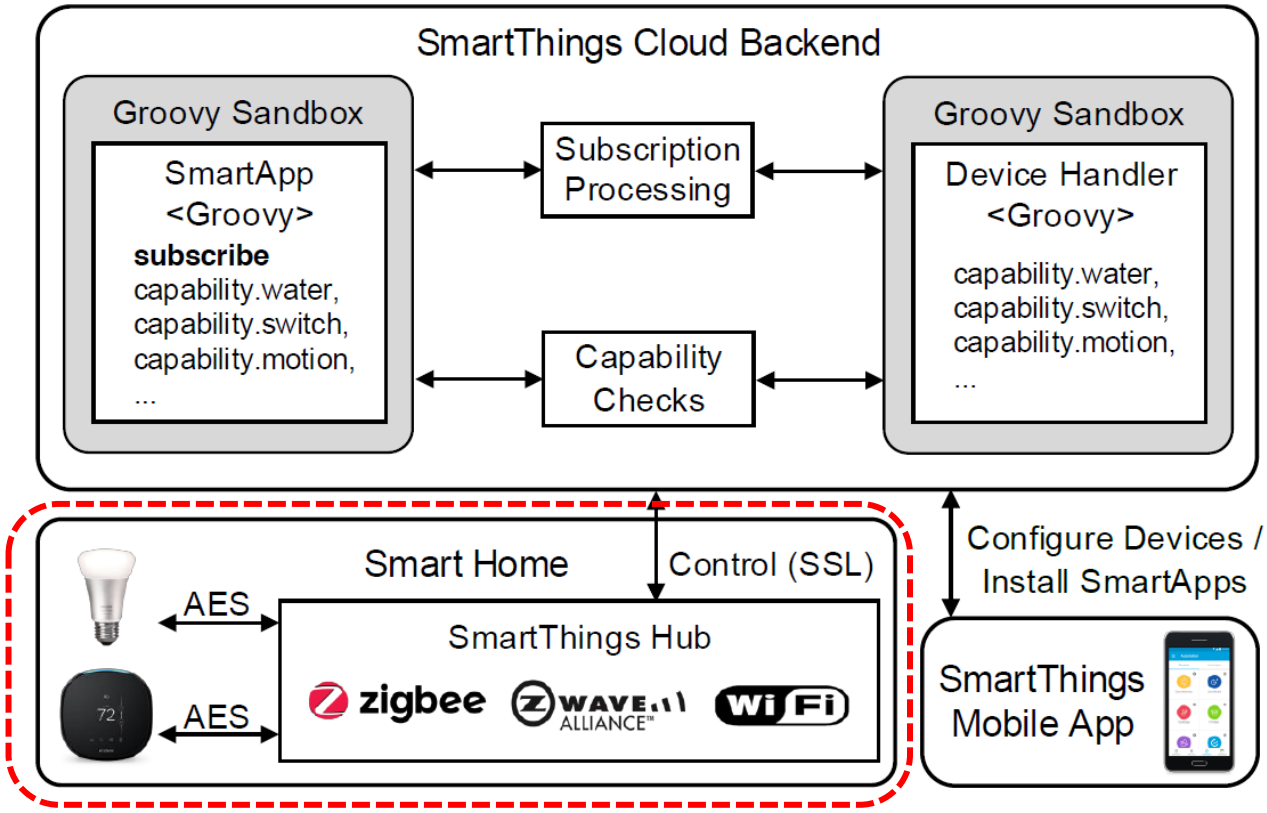
SmartThings

- SmartThings architecture



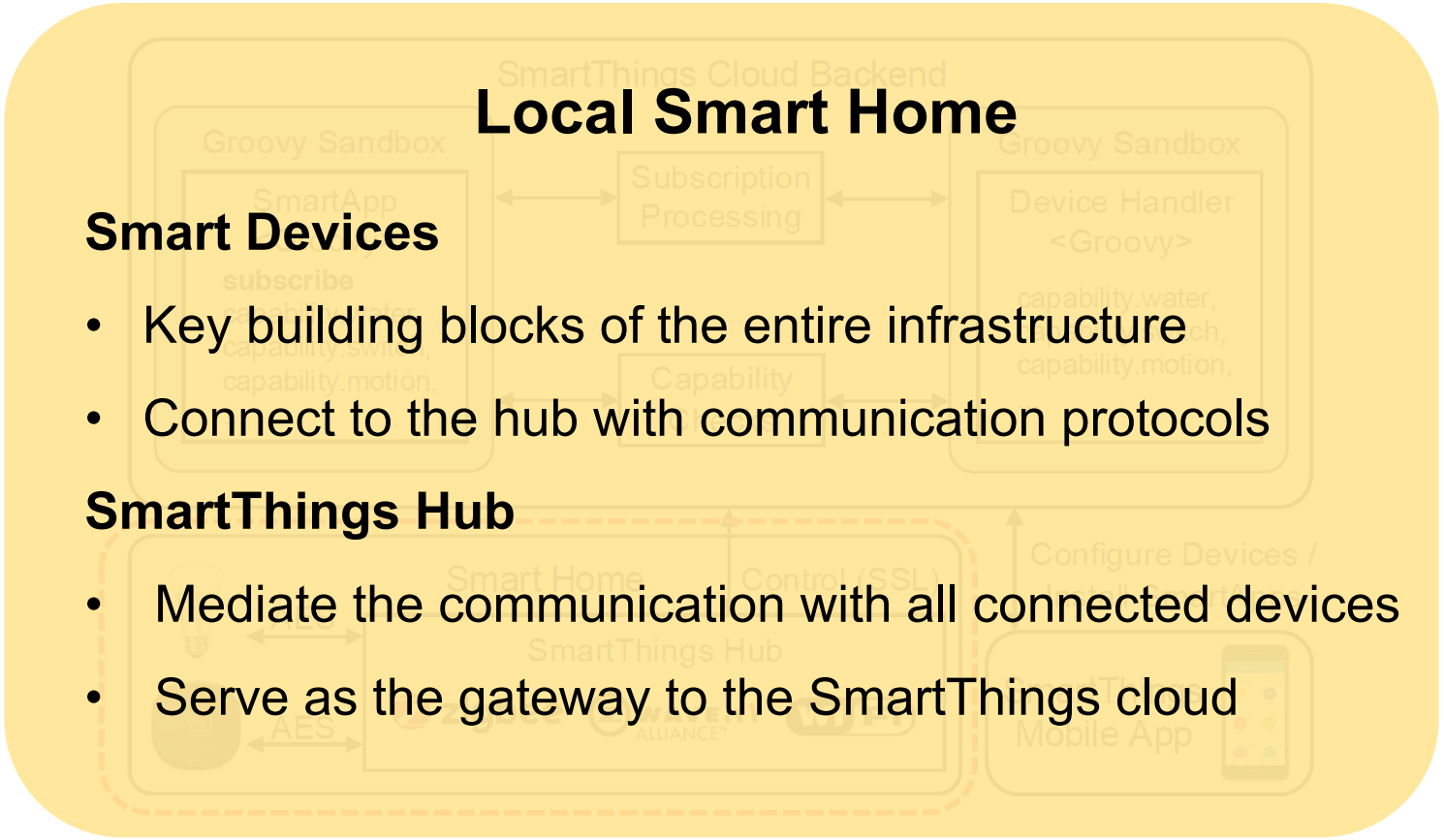
SmartThings

- SmartThings architecture



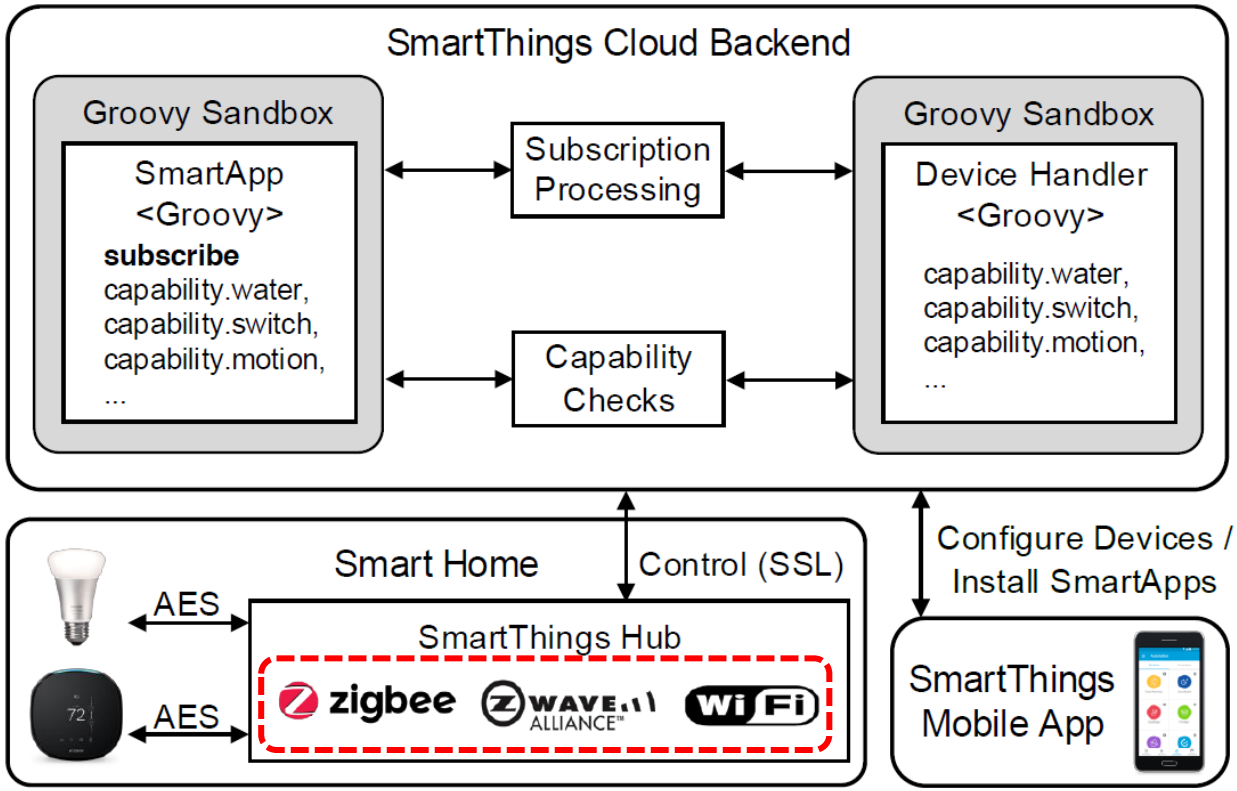
SmartThings

- SmartThings architecture



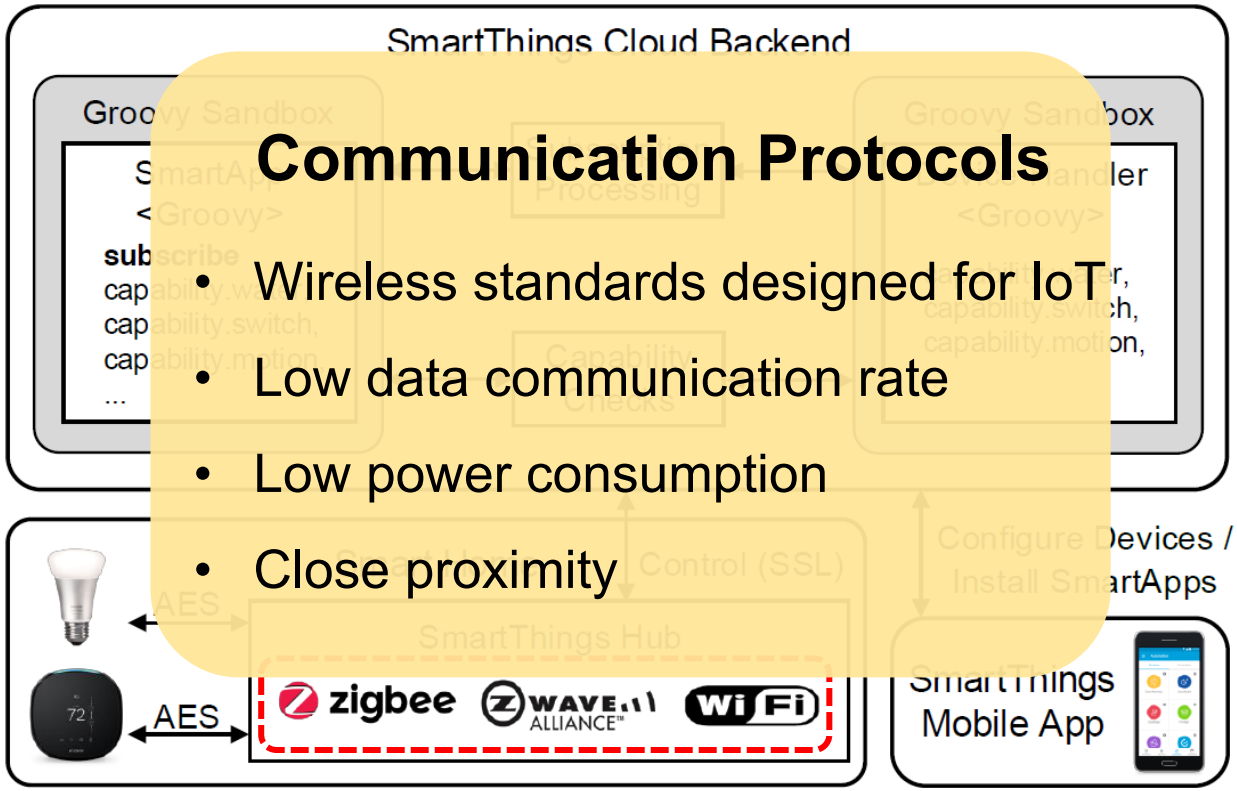
SmartThings

- SmartThings architecture



SmartThings

- SmartThings architecture



SmartThings

- SmartThings architecture

Popular Protocols for Smart Home

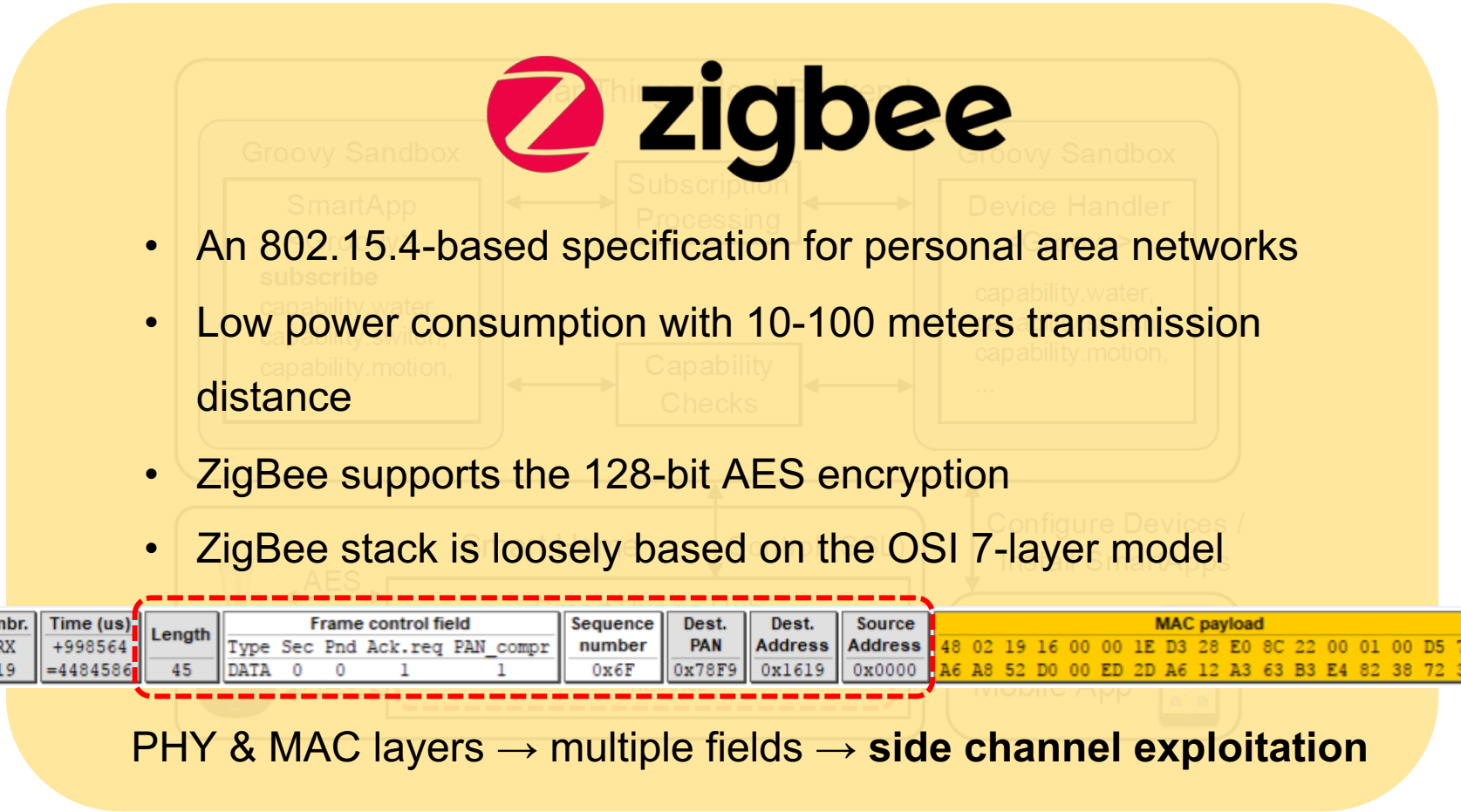
Protocols to Study in Our Work

ZigBee and Z-Wave

- Two dominant wireless protocols in SmartThings device market
- Contribute to about **79.7%** of device market share

SmartThings

- SmartThings architecture



- An 802.15.4-based specification for personal area networks
- Low power consumption with 10-100 meters transmission distance
- ZigBee supports the 128-bit AES encryption
- ZigBee stack is loosely based on the OSI 7-layer model

P.nbr.	Time (us)	Length	Frame control field					Sequence number	Dest. PAN	Dest. Address	Source Address	MAC payload
			Type	Sec	Pnd	Ack.req	PAN_compr					
RX	+998564											48 02 19 16 00 00 1E D3 28 E0 8C 22 00 01 00 D5 79
19	=4484586	45	DATA	0	0	1	1	0x6F	0x78F9	0x1619	0x0000	A6 A8 52 D0 00 ED 2D A6 12 A3 63 B3 E4 82 38 72 37

PHY & MAC layers → multiple fields → **side channel exploitation**

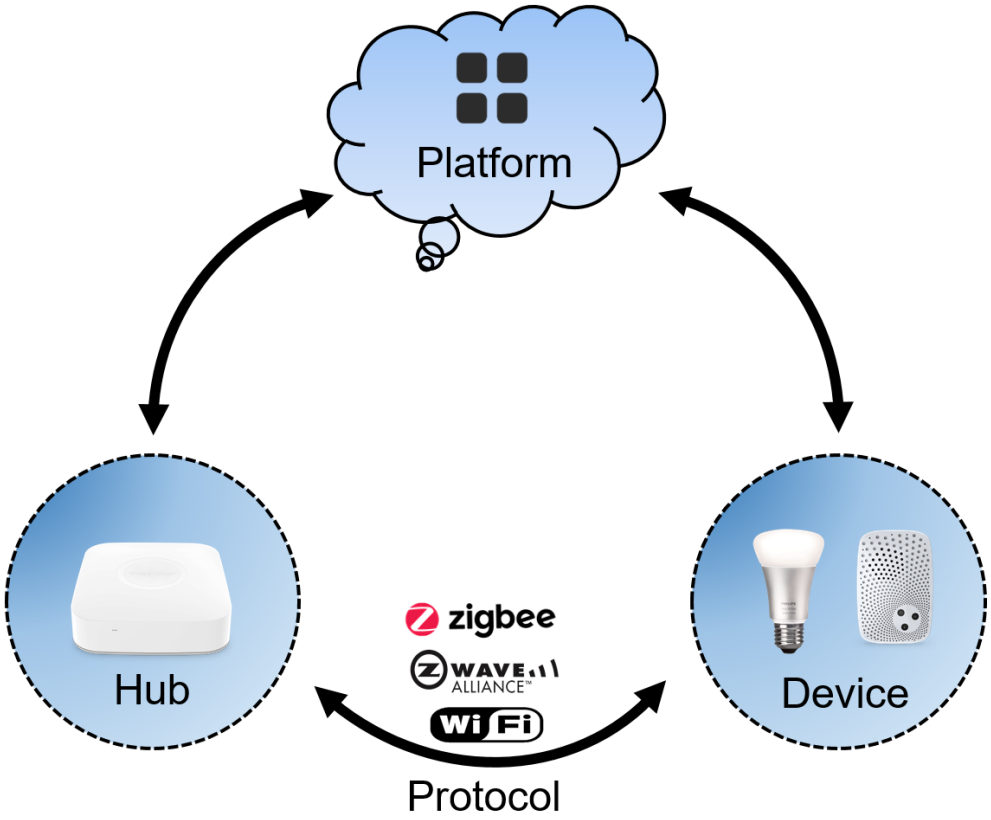
SmartThings

- SmartThings architecture

- A popular low-power consumption communication protocol having frame structure similar with ZigBee
- Follow ITU-T G.9959 recommendation
- In US, Z-Wave devices are specified to work on two frequency bands (908.4MHz & 916MHz)
- Support strong encryption and authentication via Z-Wave S2 security solution, which implements 128-bit AES encryption

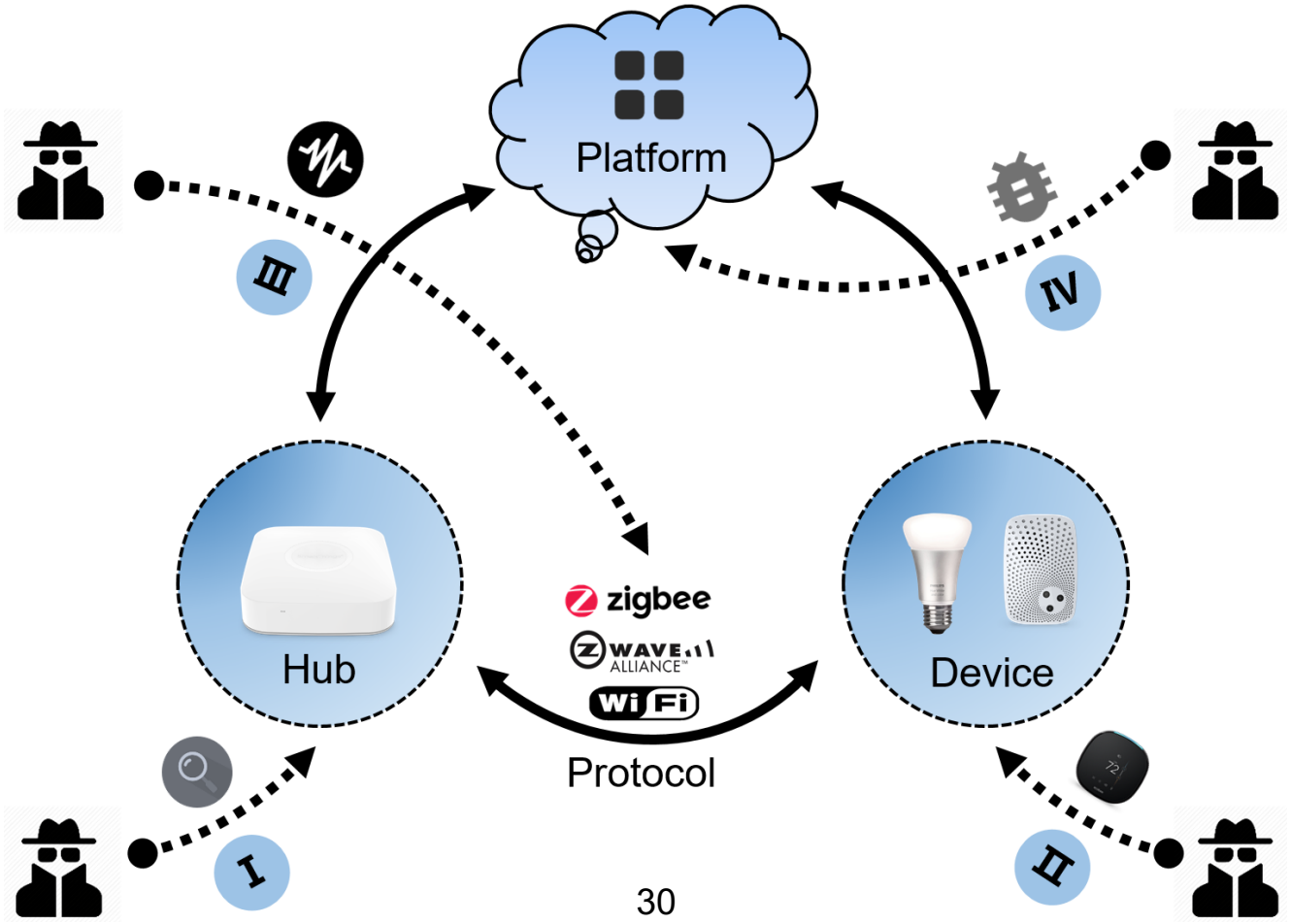
Smart Home Security

- There are four dominant components in the system



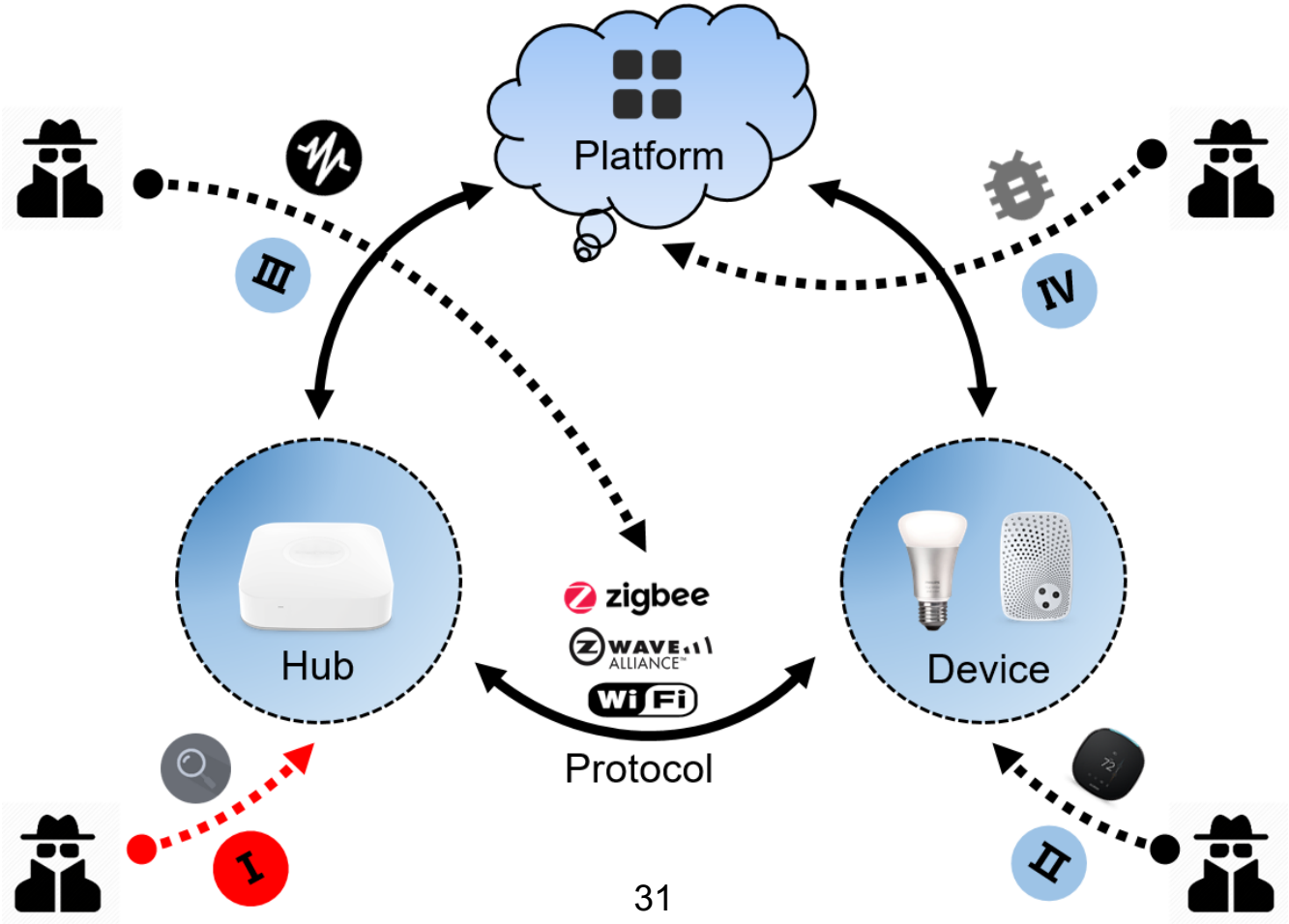
Smart Home Security

- Existing works mainly fall in to four categories



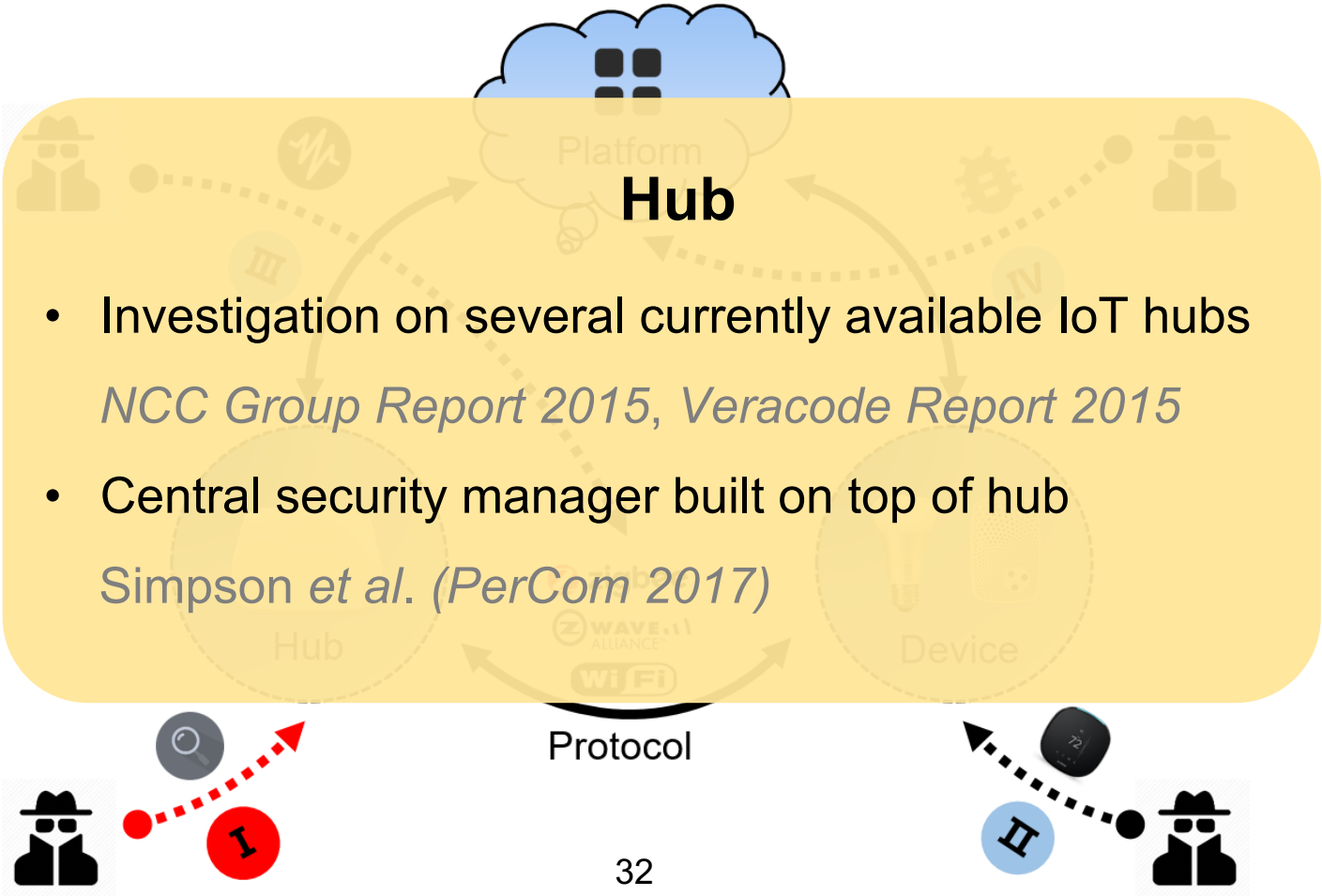
Smart Home Security

- Existing works mainly fall in to four categories



Smart Home Security

- Existing works mainly fall in to four categories



Hub

- Investigation on several currently available IoT hubs

  NCC Group Report 2015, *Veracode Report 2015*

- Central security manager built on top of hub

  Simpson *et al*. *(PerCom 2017)*

Smart Home Security

- Existing works mainly fall in to four categories

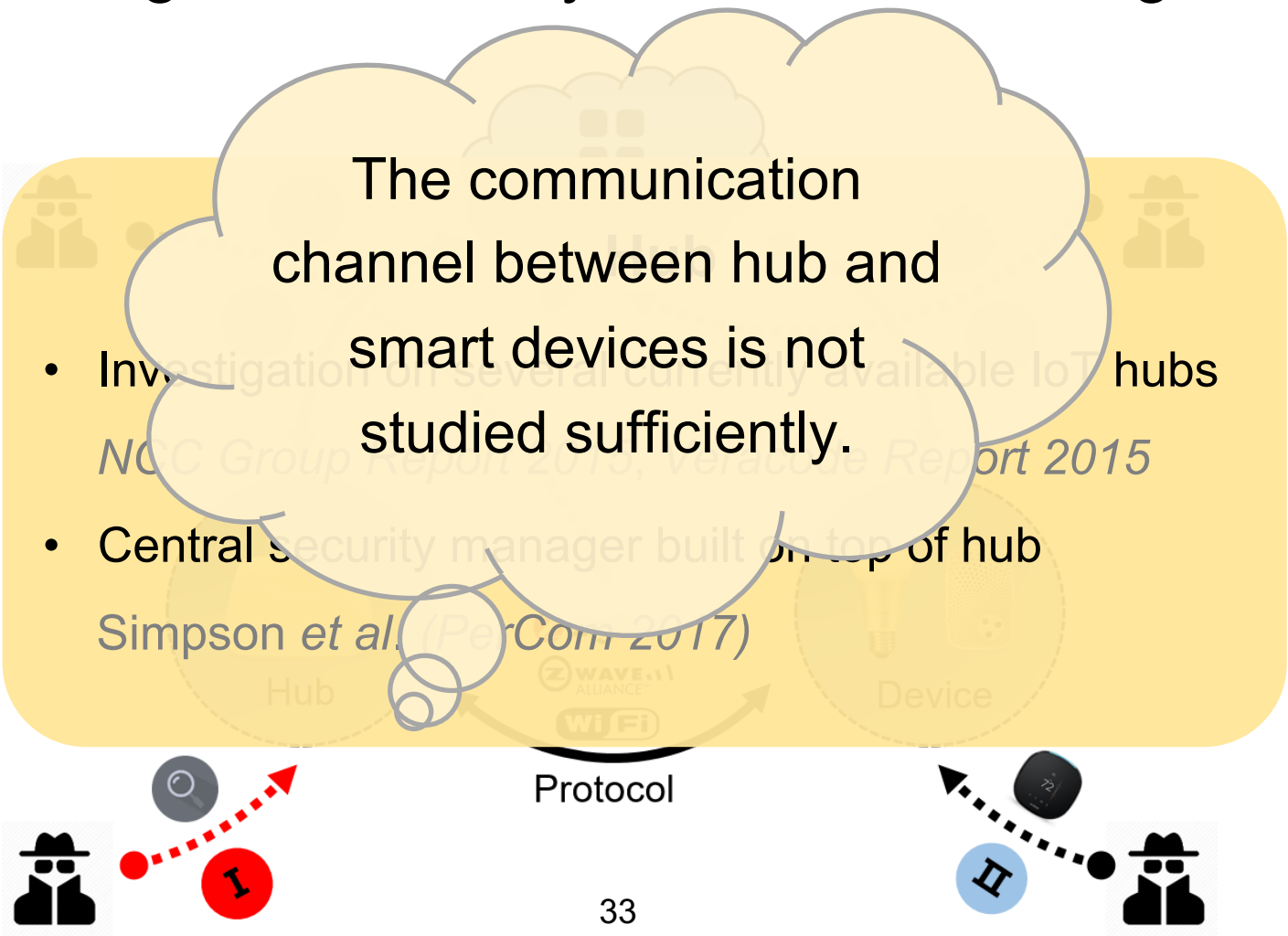
The communication channel between hub and smart devices is not studied sufficiently.

- Inv… hubs

  NCC Group Report 2015, Veracode Report 2015

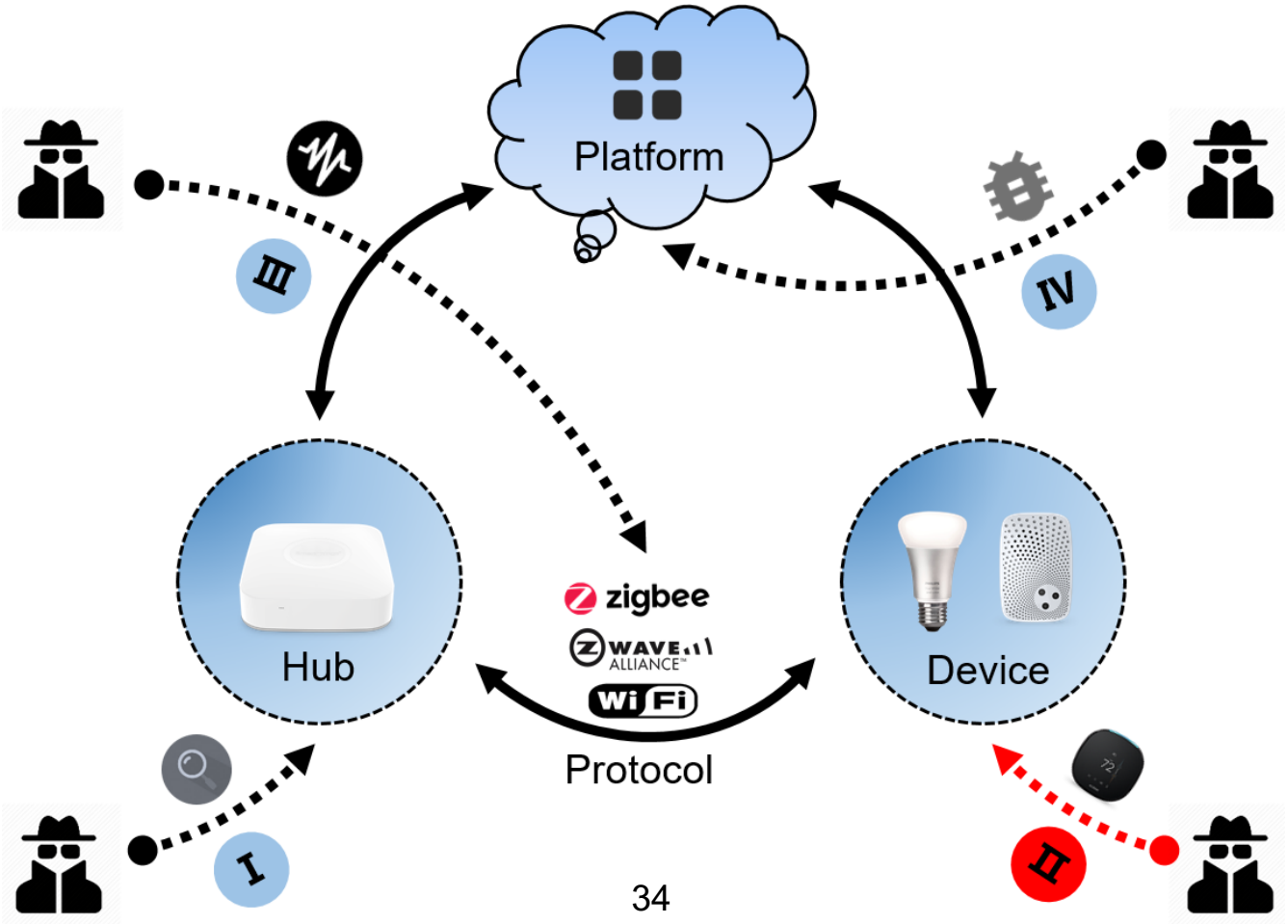
- Central s… of hub

  Simpson *et al.* (PerCom 2017)



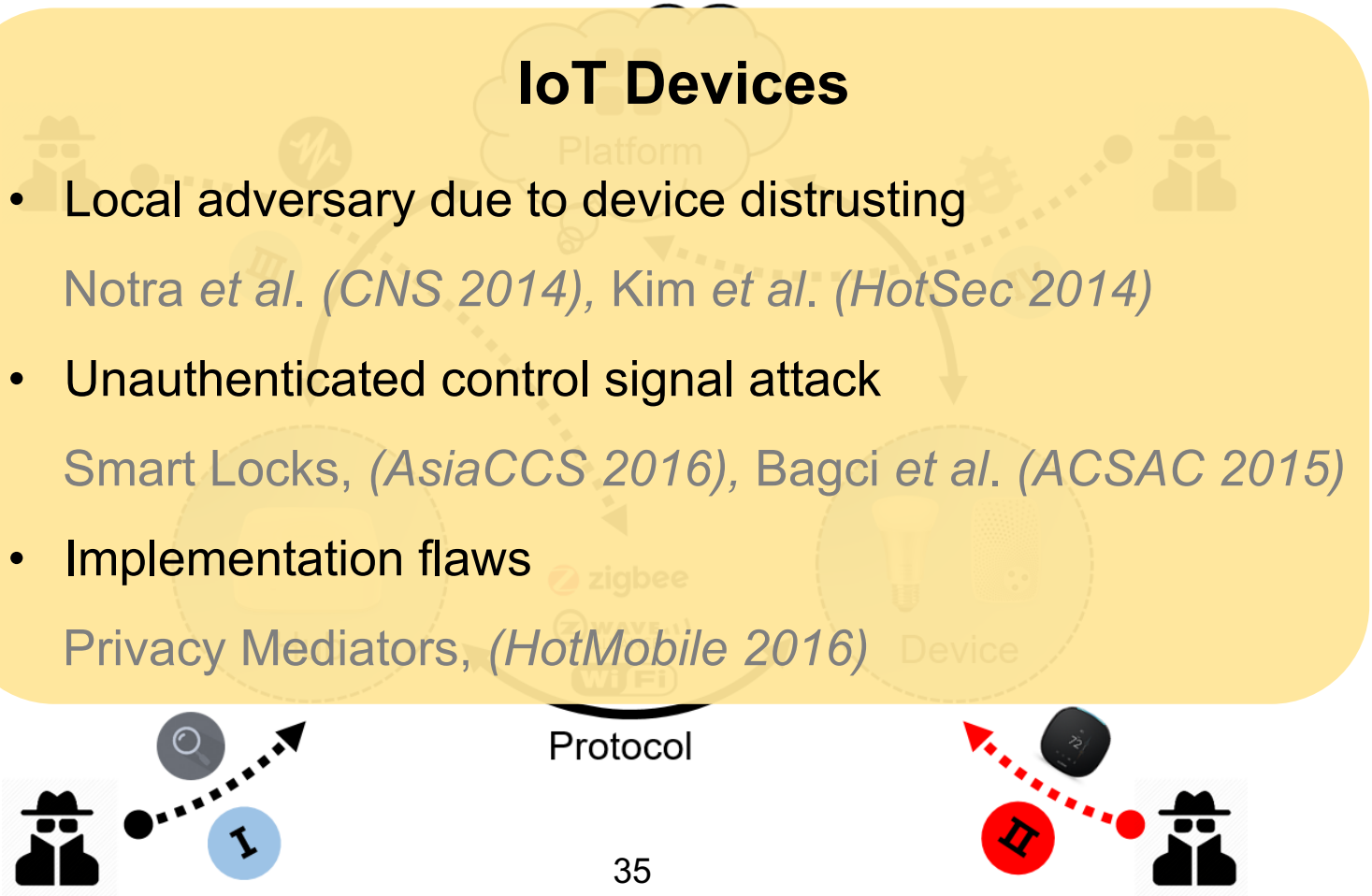
Smart Home Security

- Existing works mainly fall in to four categories



Smart Home Security

- Existing works mainly fall in to four categories



Smart Home Security

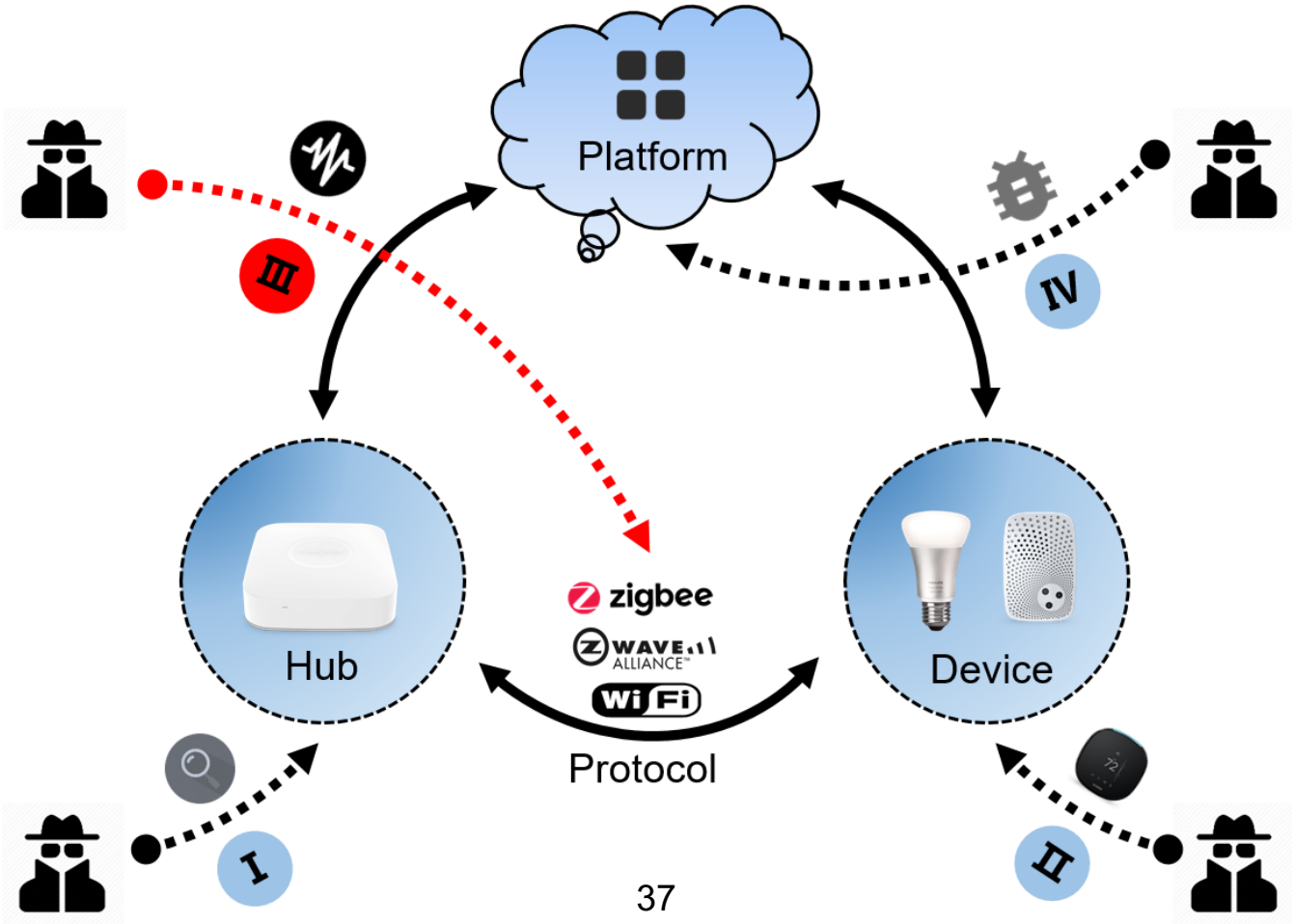
- Existing works mainly fall in to four categories

Can we find approach to protect devices with the semantics of device handlers in smart home context?

- Loc
  Not
- Una
  Smart Locks, *(AsiaCCS 2016)*, Bagci et al. *(ACSAC 2015)*
- Implementation flaws
  Privacy Mediator *(HotMobile 2016)*

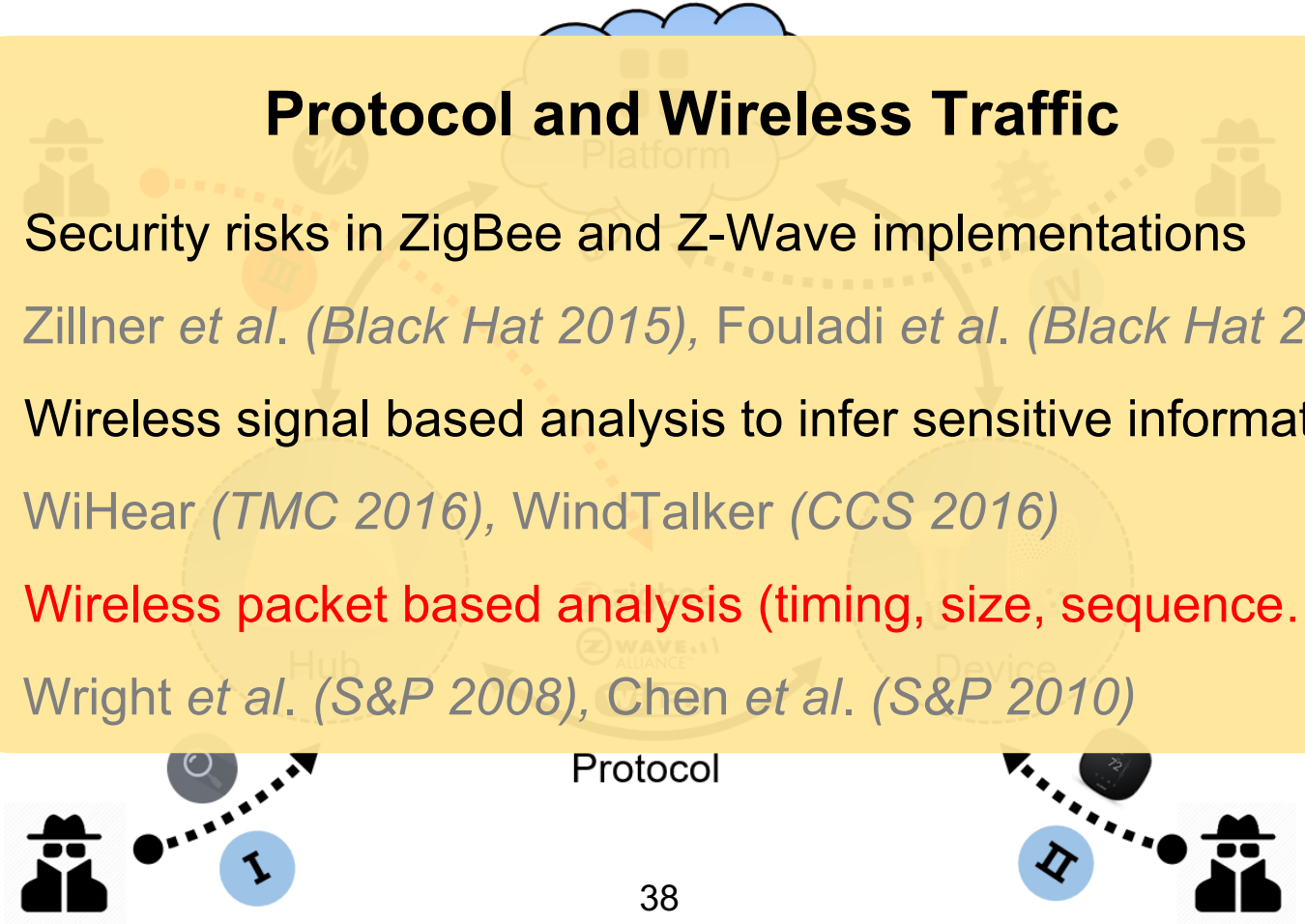
Smart Home Security

- Existing works mainly fall in to four categories



Smart Home Security

- Existing works mainly fall in to four categories



Protocol and Wireless Traffic

- Security risks in ZigBee and Z-Wave implementations

  Zillner *et al. (Black Hat 2015)*, Fouladi *et al. (Black Hat 2013)*
- Wireless signal based analysis to infer sensitive information

  WiHear *(TMC 2016)*, WindTalker *(CCS 2016)*
- Wireless packet based analysis (timing, size, sequence…)

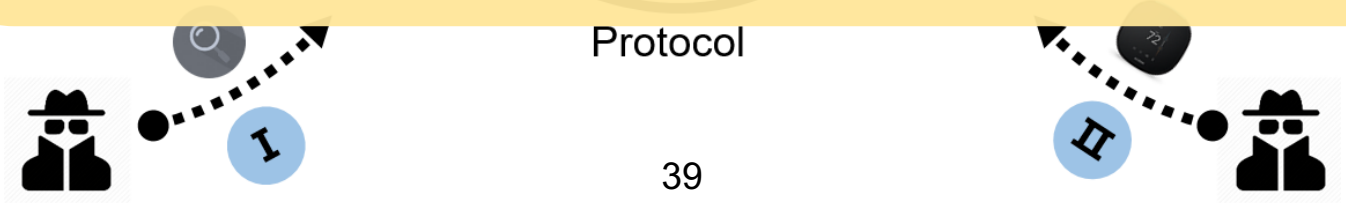
  Wright *et al. (S&P 2008)*, Chen *et al. (S&P 2010)*

Smart Home Security

- Existing works mainly fall in to four categories

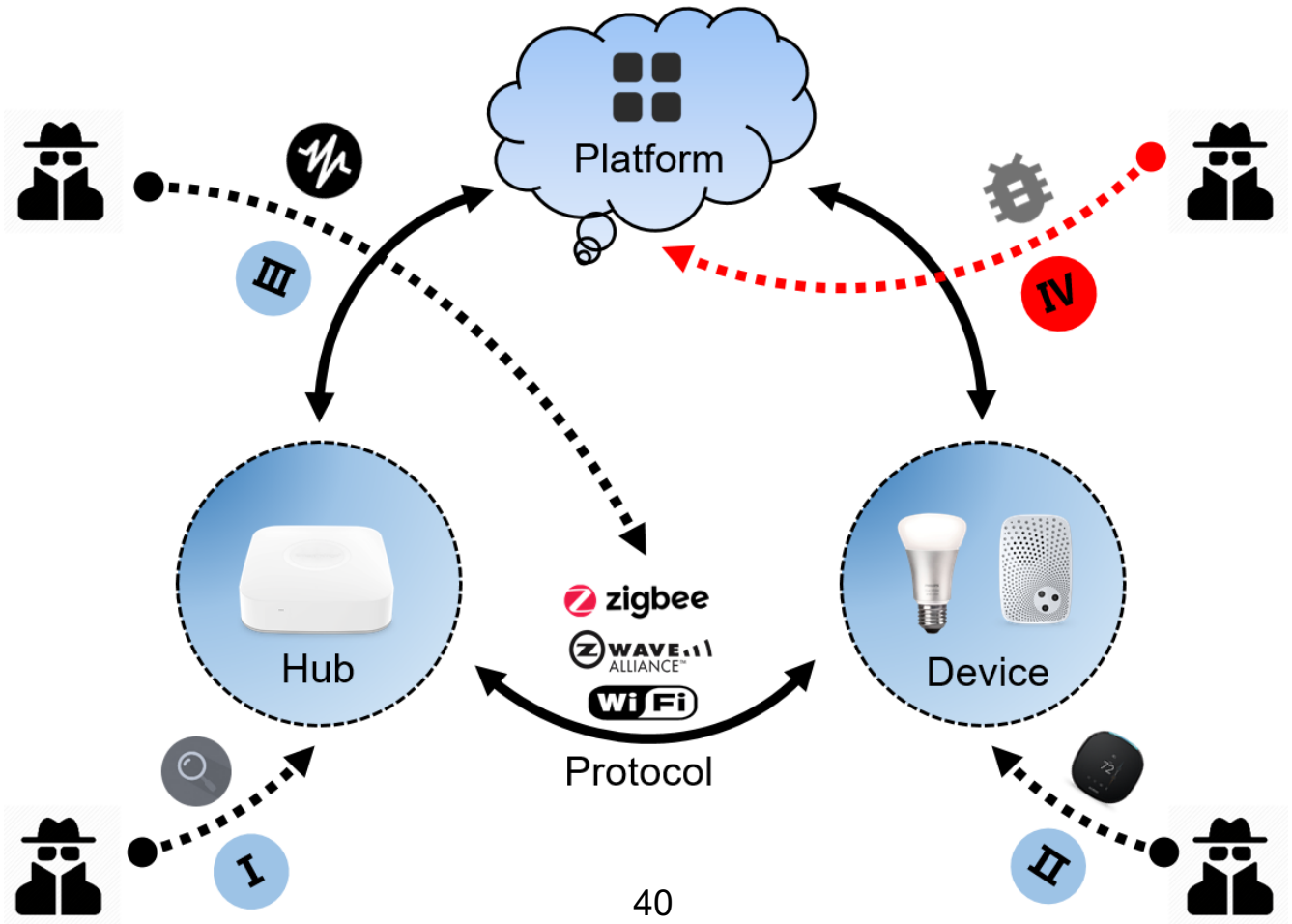
Can we exploit this side channel to implement on the emerging smart home platforms?

- Secu… …ons
  Zillner *et al.* (… …Hat 2013)
- Wirele… …tive information
  WiHear *(TMC 2016)*, WindTalker *(CCS 2016)*
- Wireless packet b… analysis (timing, size, sequence…)
  Wright *et al.* *(S&P 2008)*, Chen *et al.* *(S&P 2010)*



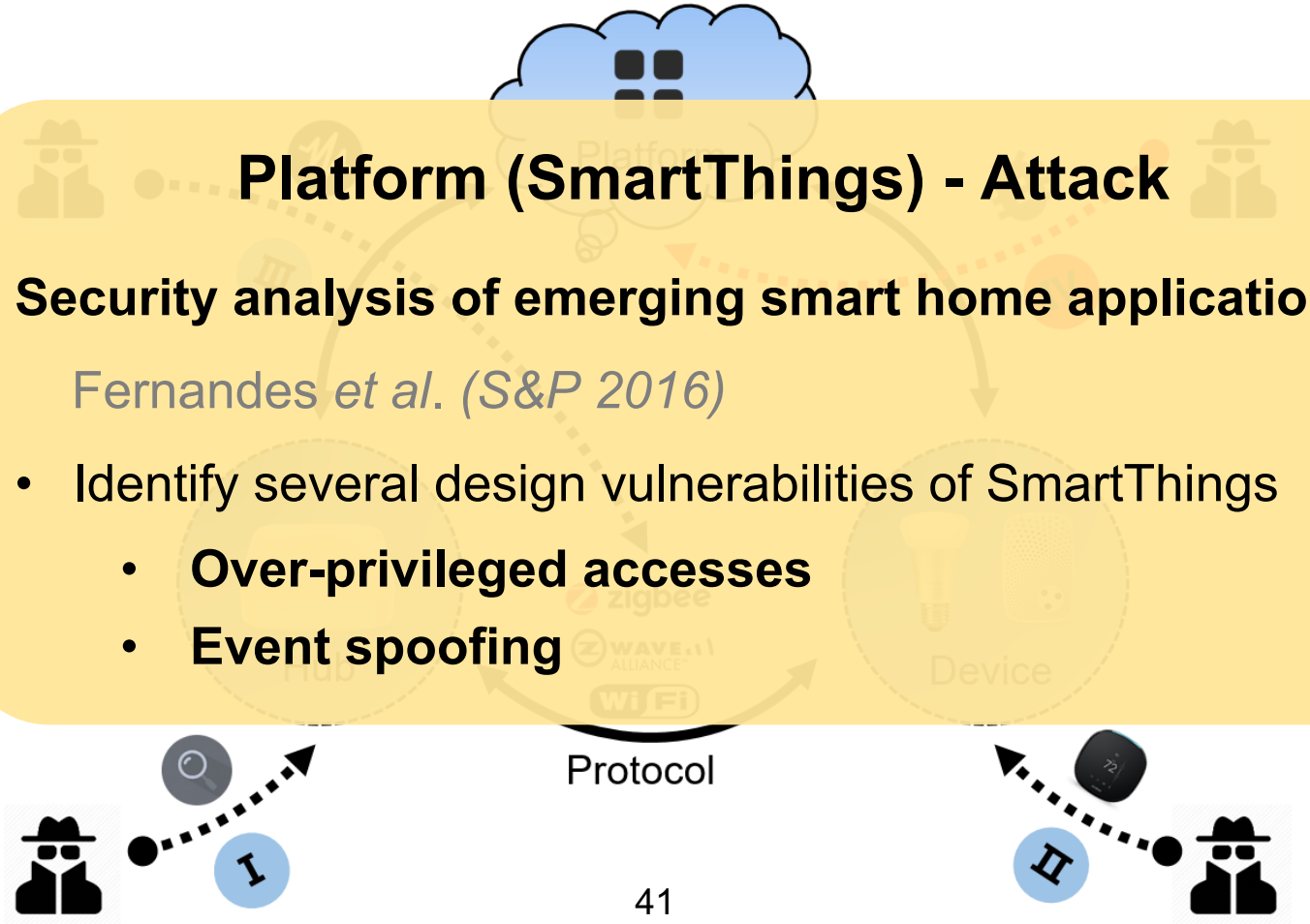
Smart Home Security

- Existing works mainly fall in to four categories



Smart Home Security

- Existing works mainly fall in to four categories



Platform (SmartThings) - Attack

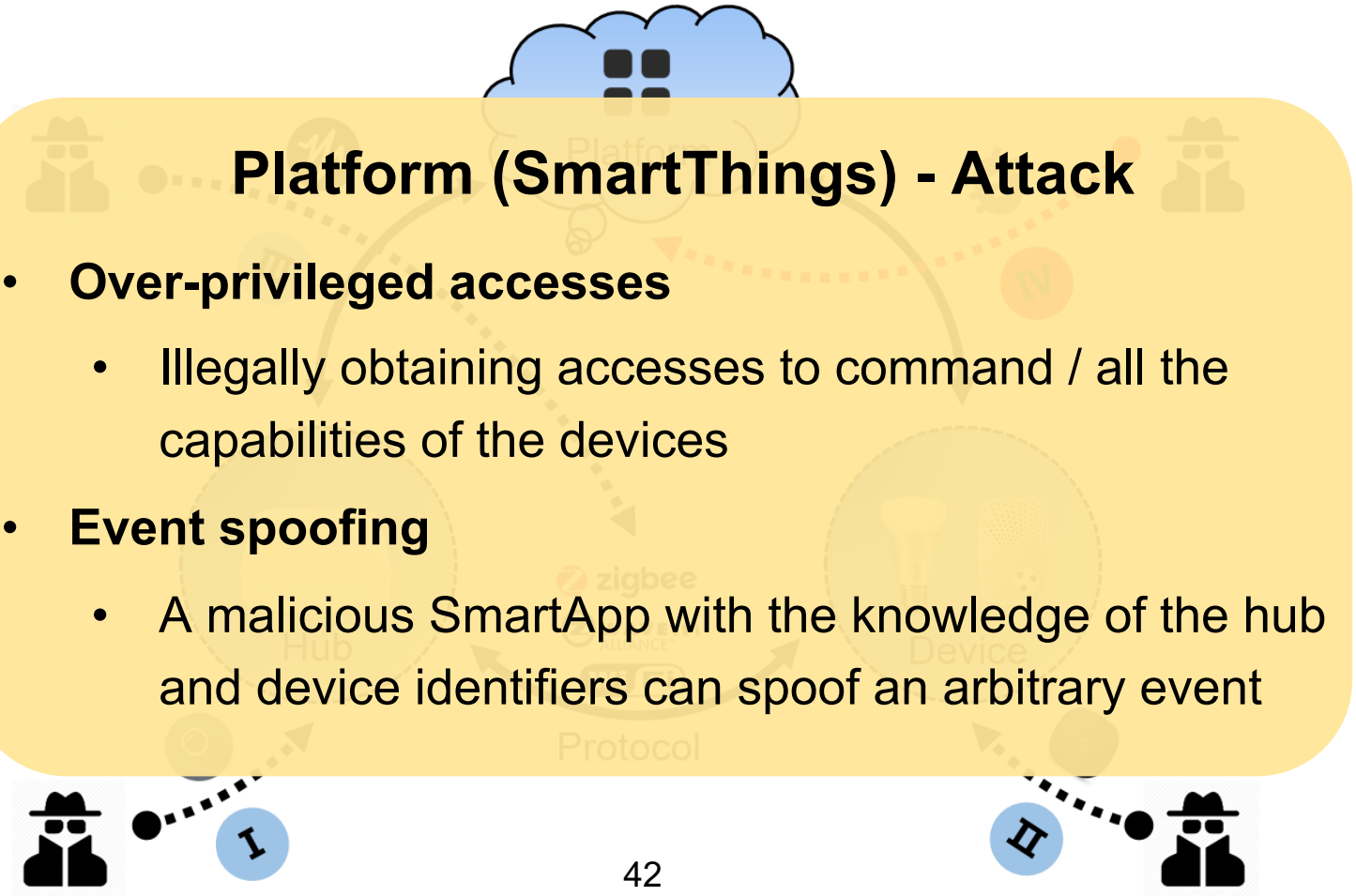
Security analysis of emerging smart home applications

Fernandes *et al*. *(S&P 2016)*

- Identify several design vulnerabilities of SmartThings
  - **Over-privileged accesses**
  - **Event spoofing**

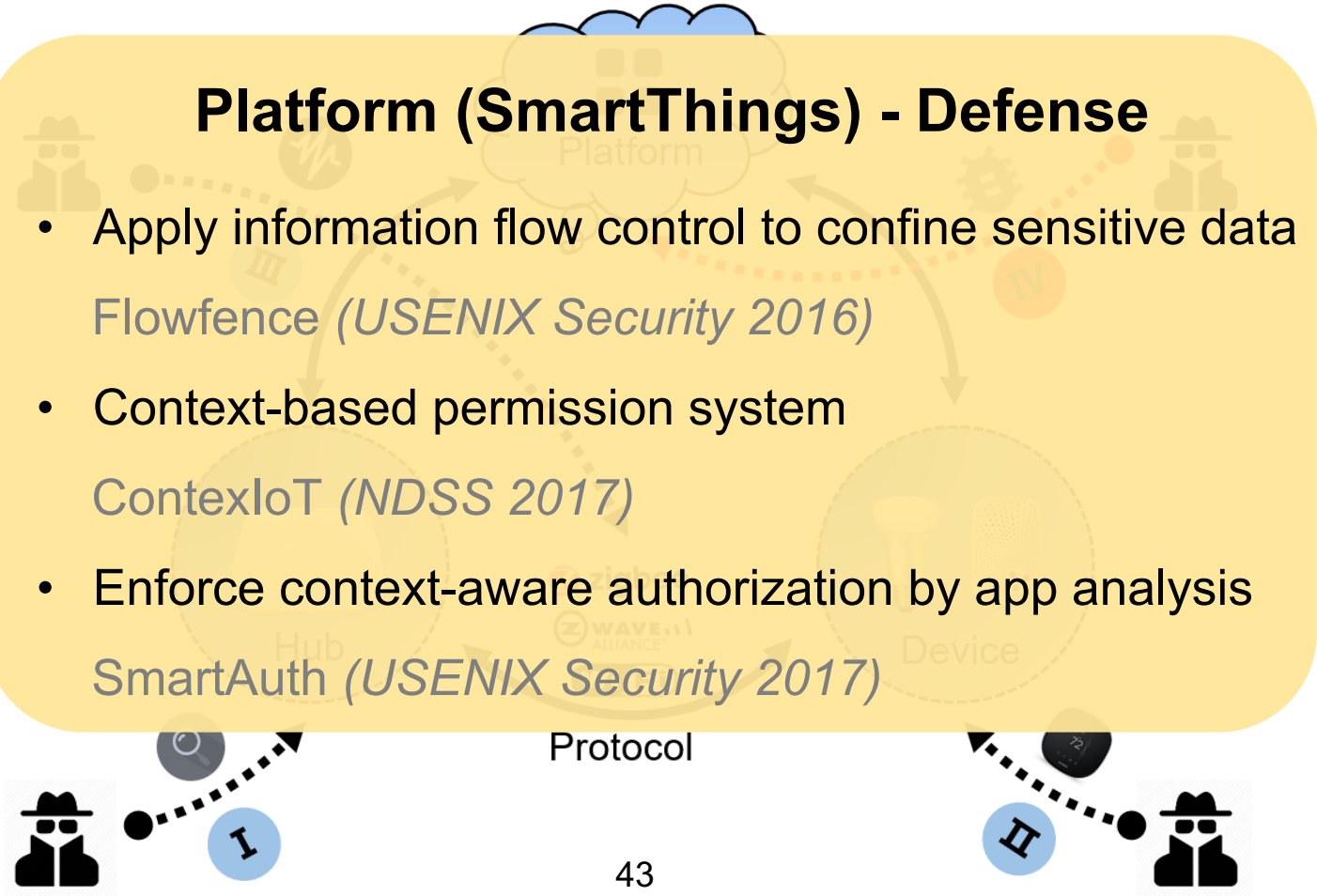
Smart Home Security

- Existing works mainly fall in to four categories



Smart Home Security

- Existing works mainly fall in to four categories



Platform (SmartThings) - Defense

- Apply information flow control to confine sensitive data

  Flowfence *(USENIX Security 2016)*
- Context-based permission system

  ContexIoT *(NDSS 2017)*
- Enforce context-aware authorization by app analysis

  SmartAuth *(USENIX Security 2017)*

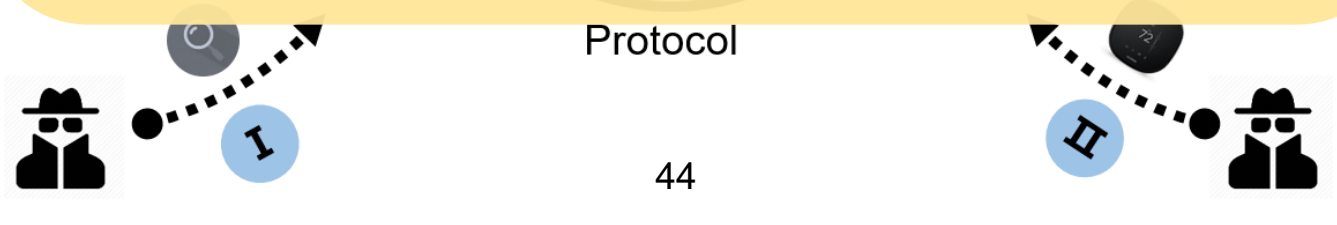
Smart Home Security

- Existing works mainly fall in to four categories

Platform (SmartThings) - Defens**e**

- A[illegible] information flow [illegible] sensitive data
  Flowfence *(USENIX Security 2016)*
- Co[illegible] context-based [illegible]
  ContexIoT *(NDSS 2017)*
- Enforce contex[illegible]-aware authorization by app analysis
  SmartAuth *(USENIX Security 2017)*

Can we find a way to resist these attacks without platform modification or app patching?



Our Design: HoMonit

HoMonit **:** a novel platform-independent anomaly detection system for monitoring smart home apps from encrypted wireless traffic.

Our Design: HoMonit

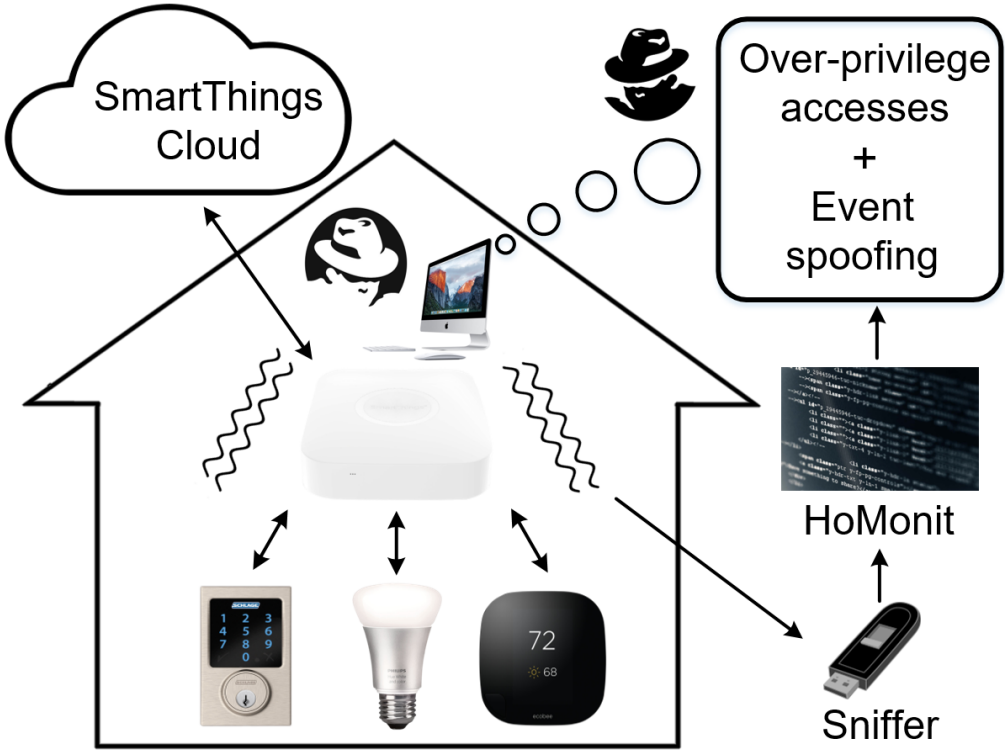
HoMonit **:** a novel platform-independent anomaly detection system for monitoring smart home apps from encrypted wireless traffic.

Contributions

- ***Novel techniques***
  App activities inference using DFA extraction and wireless side-channel
- ***New systems***
  System design, implementation and evaluation
- ***Open-source dataset***
  A dataset of 60 misbehaving SmartApps to benefit research community

Our Design: HoMonit

HoMonit Schema

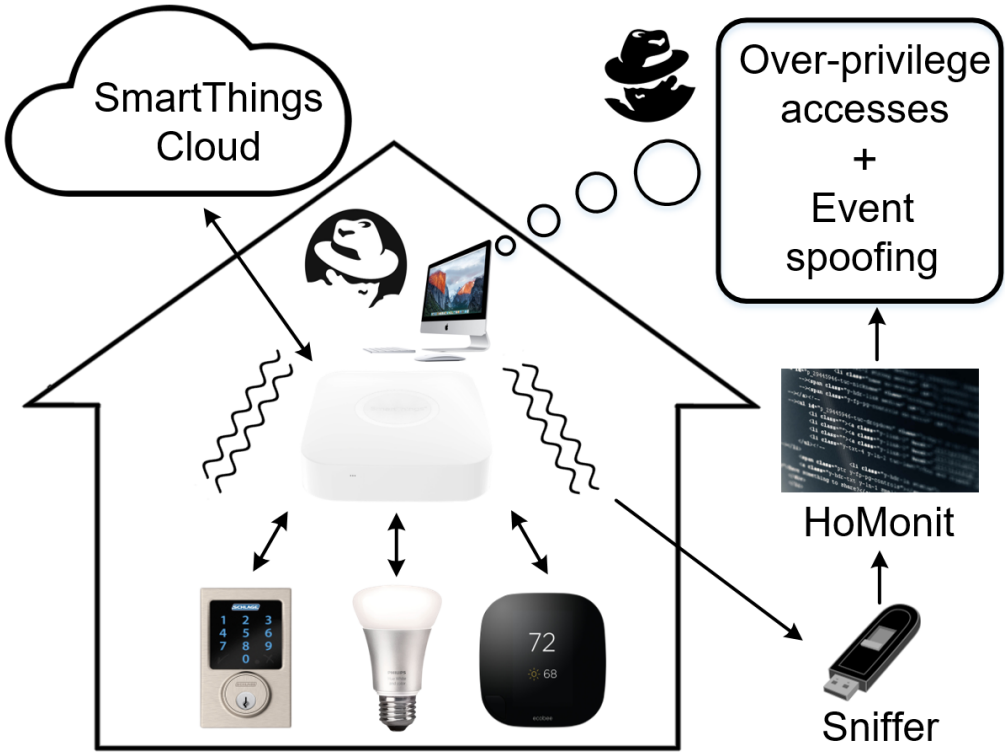


Our Design: HoMonit

HoMonit Schema

Feature

- Platform independent
- SmartApp formulation
- Static code analysis
- NLP analysis
- Encrypted wireless traffic side channel analysis
- Misbehavior detection



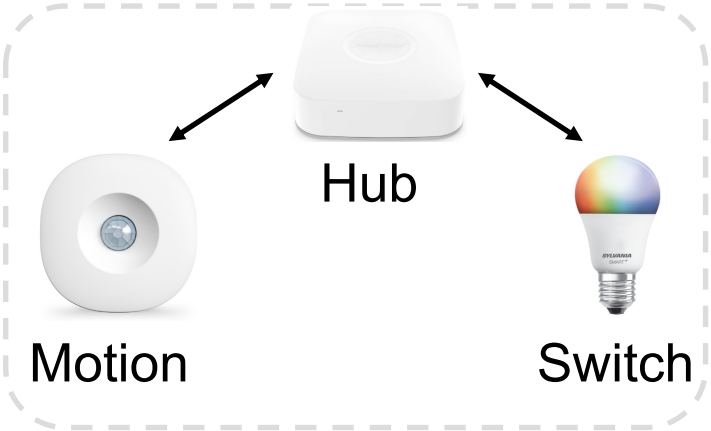
Outline

- Motivation
- Design Overview
- DFA Building via SmartApp Analysis
- Misbehavior Detection via Wireless Fingerprint
- Evaluation
- Discussion
- Conclusion

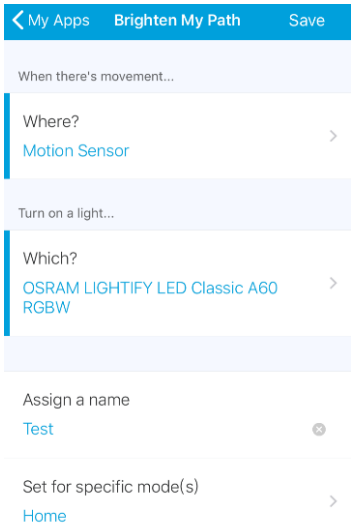
Outline

- Motivation
- Design Overview
- DFA Building via SmartApp Analysis
- Misbehavior Detection via Wireless Fingerprint
- Evaluation
- Discussion
- Conclusion

Motivation



Device



SmartApp

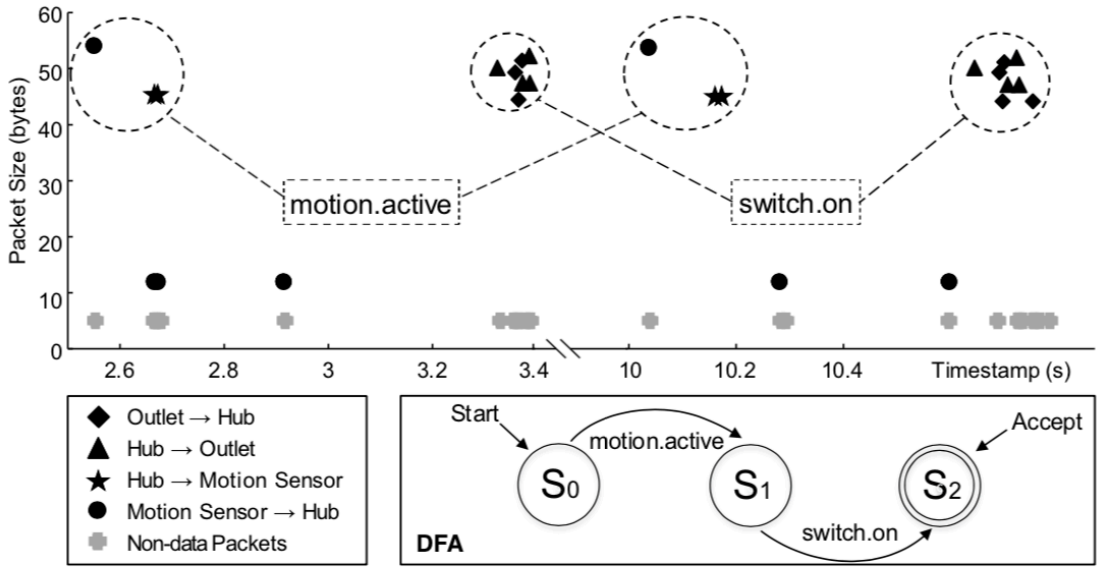
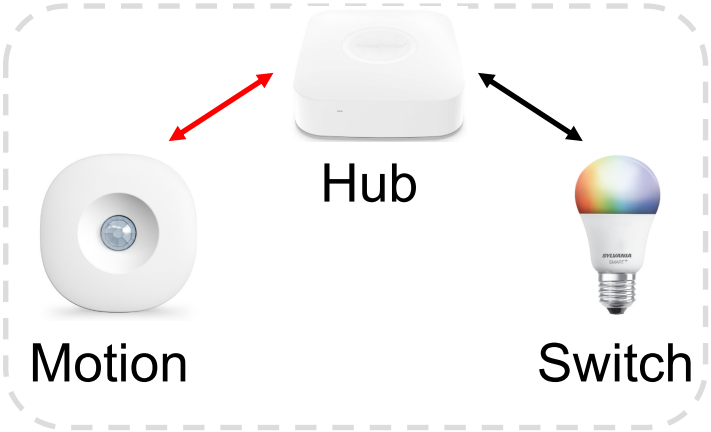
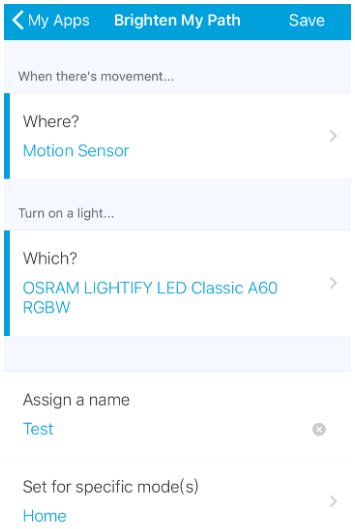


Figure 3: A motivating SmartApp: *Brighten My Path.*

Motivation



Device



SmartApp

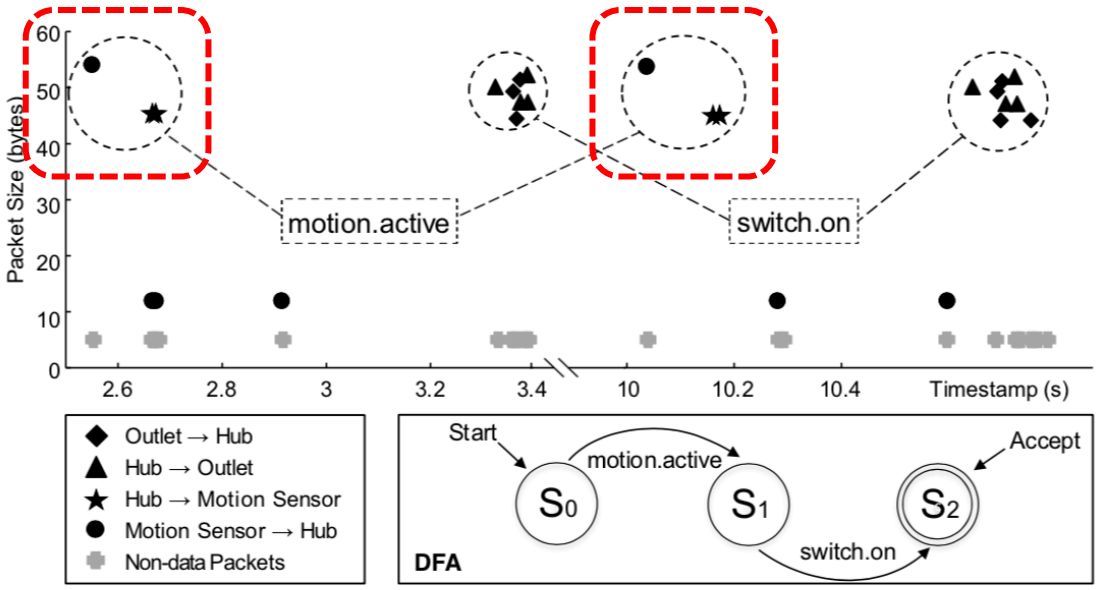
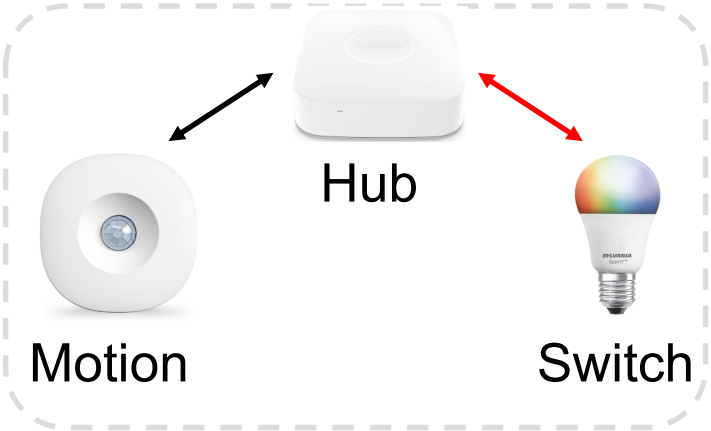
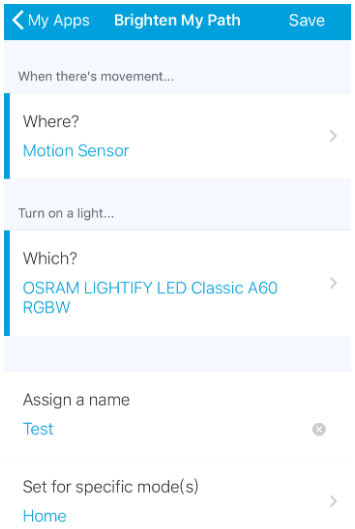


Figure 3: A motivating SmartApp: *Brighten My Path.*

Motivation



Device



SmartApp

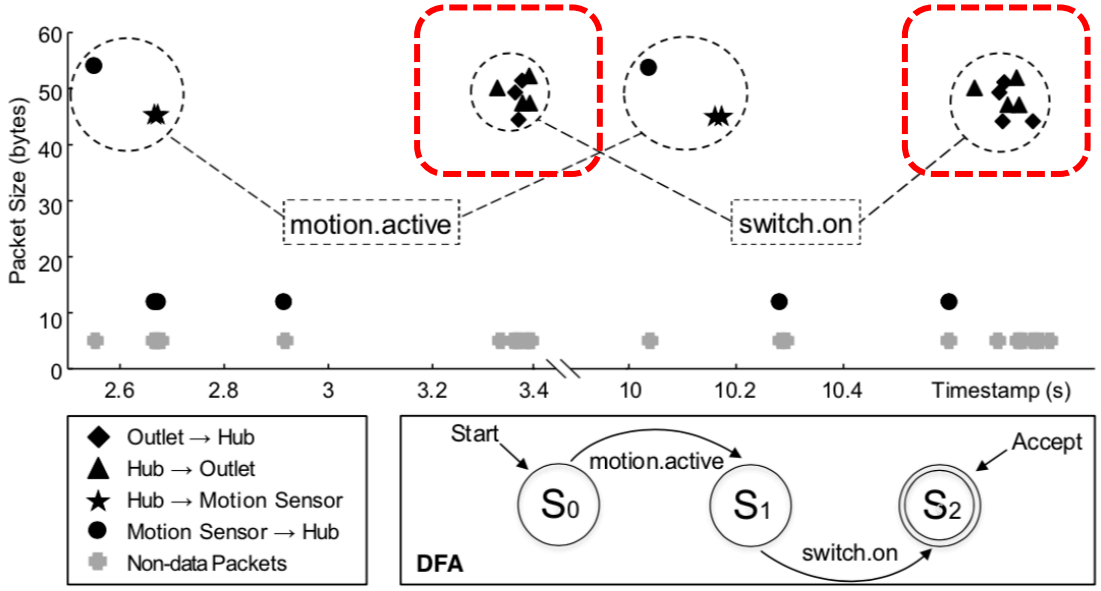
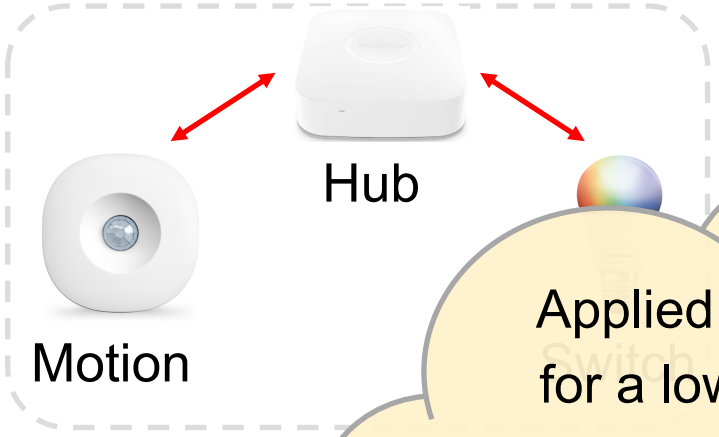


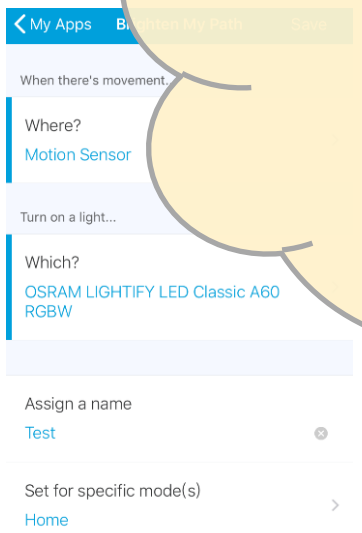
Figure 3: A motivating SmartApp: *Brighten My Path.*

Motivation



Applied protocols are designed for a low transmission rate and reduced data redundancy

Wireless traffic usually shows unique and fixed pattern



SmartApp

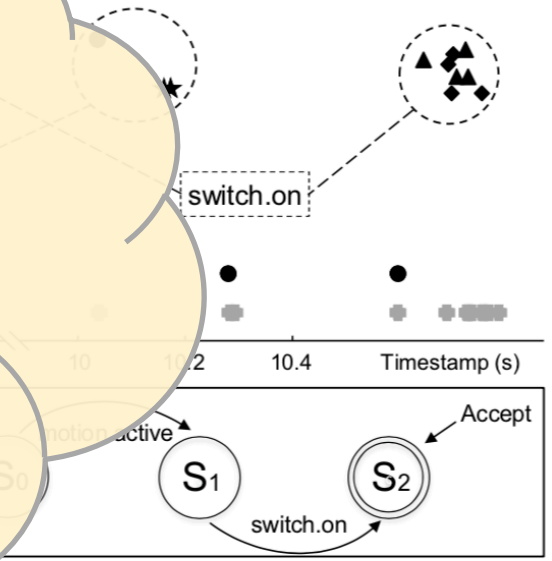
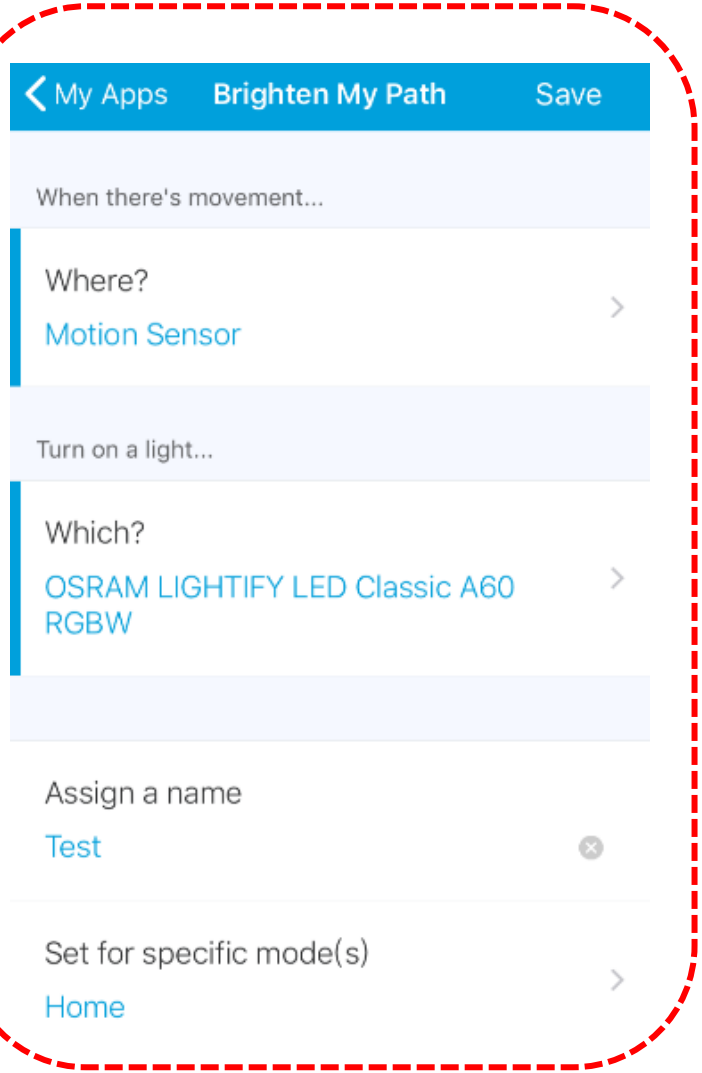


Figure 3: A motivating SmartApp: *Brighten My Path.*

Motivation



SmartApp

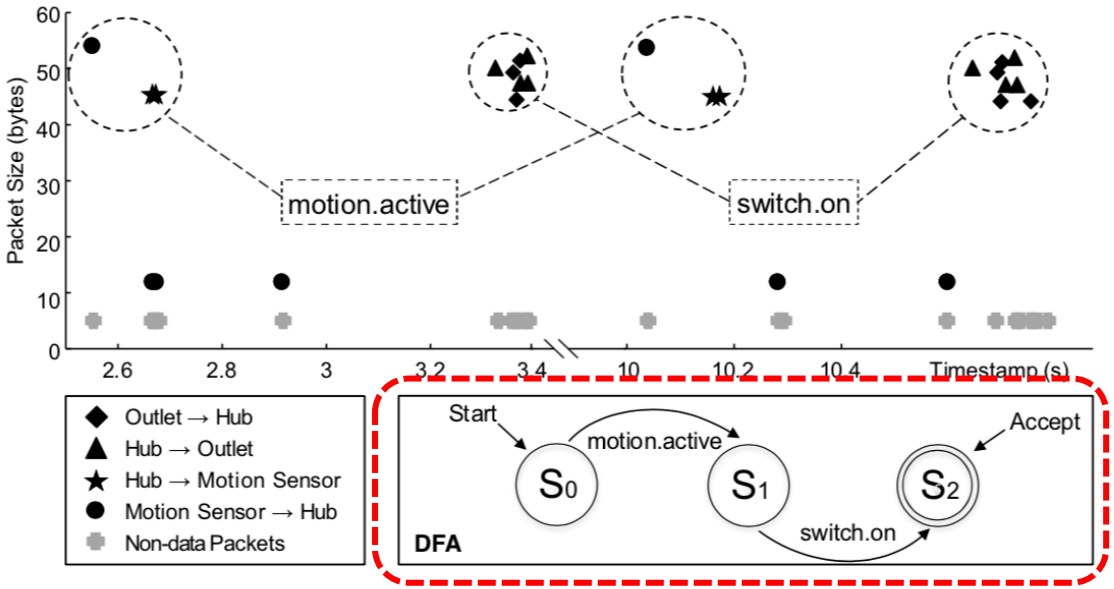
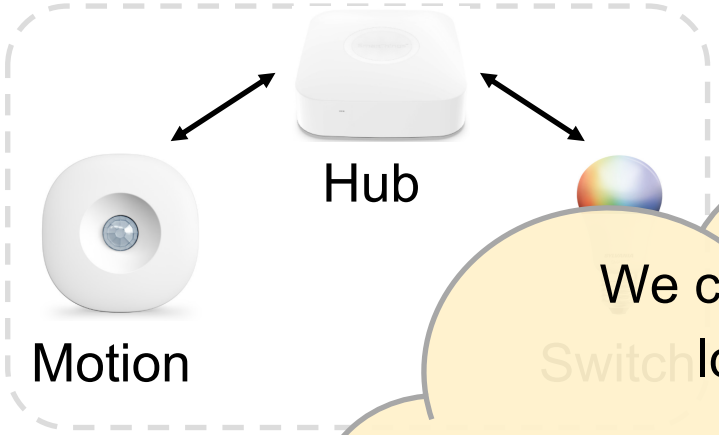


Figure 3: A motivating SmartApp: *Brighten My Path.*

Motivation



We can extract the working logic of SmartApps

By monitoring the metadata, we can determine operating SmartApps and its running state

SmartApp

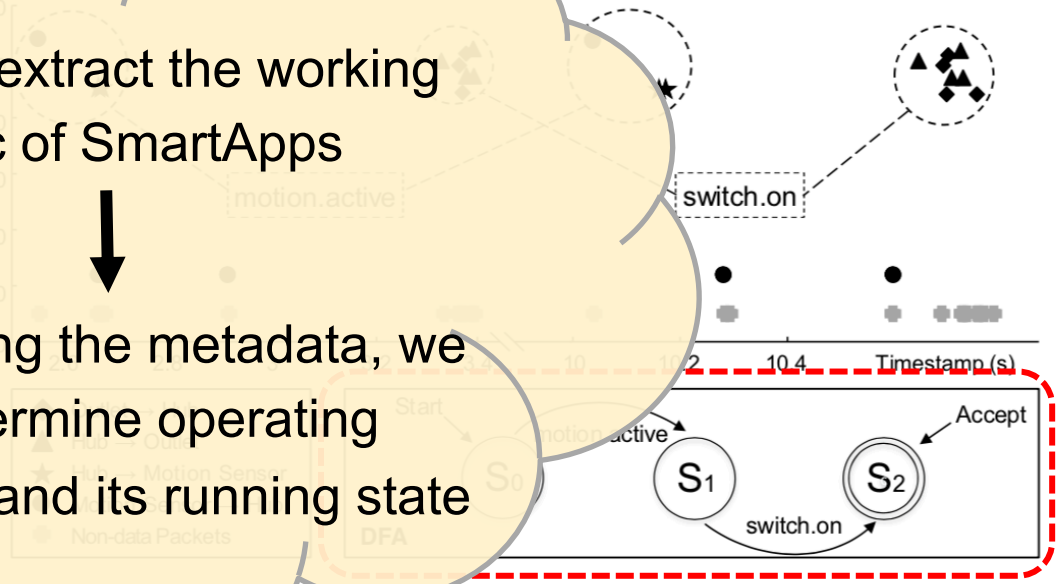


Figure 3: A motivating SmartApp: *Brighten My Path.*

Feasibility

Wireless Side Channel

- ZigBee/Z-Wave wireless traffic can be passively eavesdropped
- Device behavior is tightly corresponding to a packet sequence

Feasibility

Wireless Side Channel

- ZigBee/Z-Wave wireless traffic can be passively eavesdropped
- Device behavior is tightly corresponding to a packet sequence

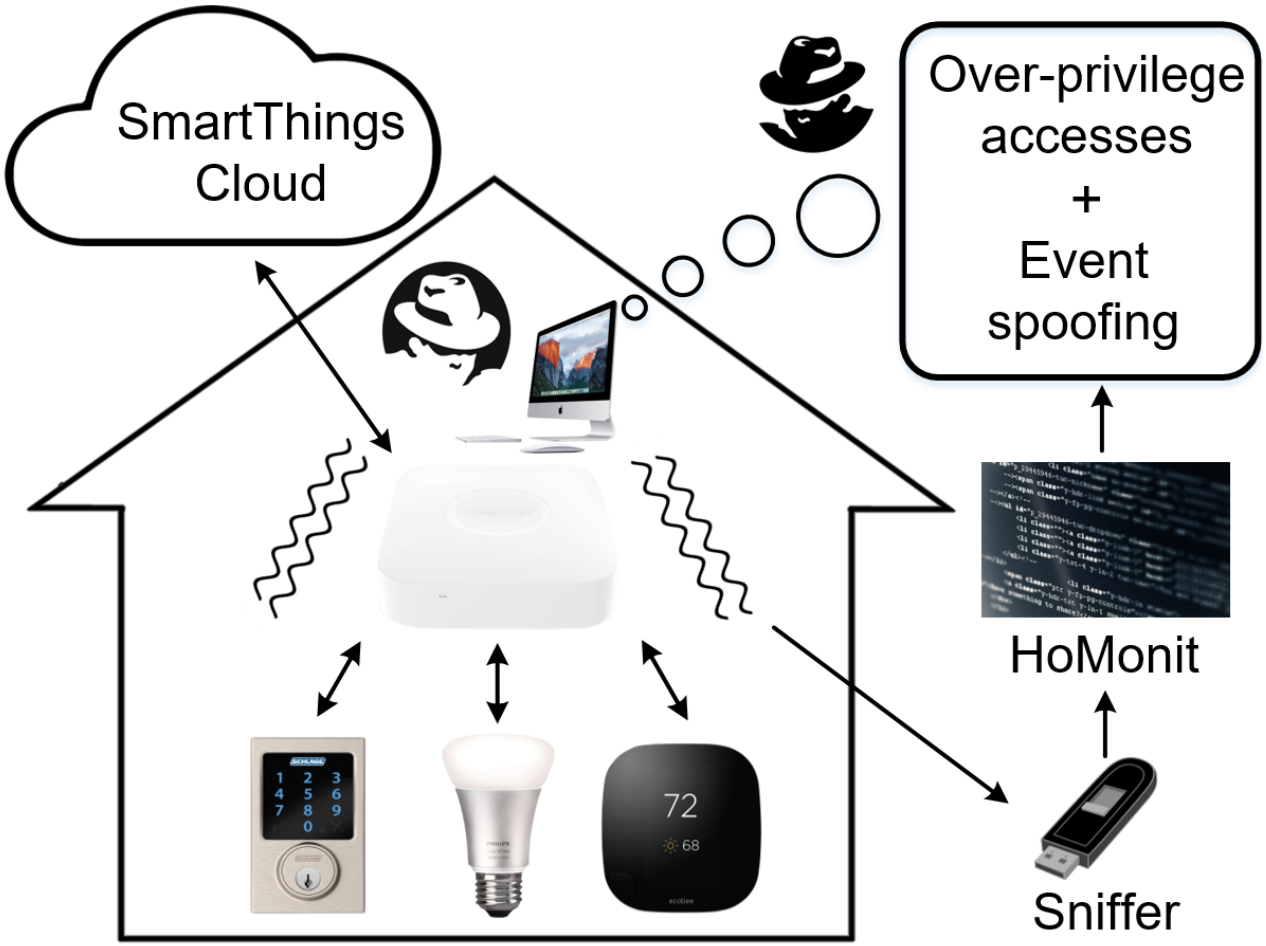
App Workflow Inference

- SmartThings offers a large scale of open/closed source SmartApps
- SmartApps usually control a set of device with certain logic
- We extracting the working logic of SmartApps as DFAs
- The behavior of the SmartApps can be inferred from the DFA transitions: normal behavior sequences are always accepted by the DFA, whereas abnormal ones are not

Outline

- Motivation
- Design Overview
- DFA Building via SmartApp Analysis
- Misbehavior Detection via Wireless Fingerprint
- Evaluation
- Discussion
- Conclusion

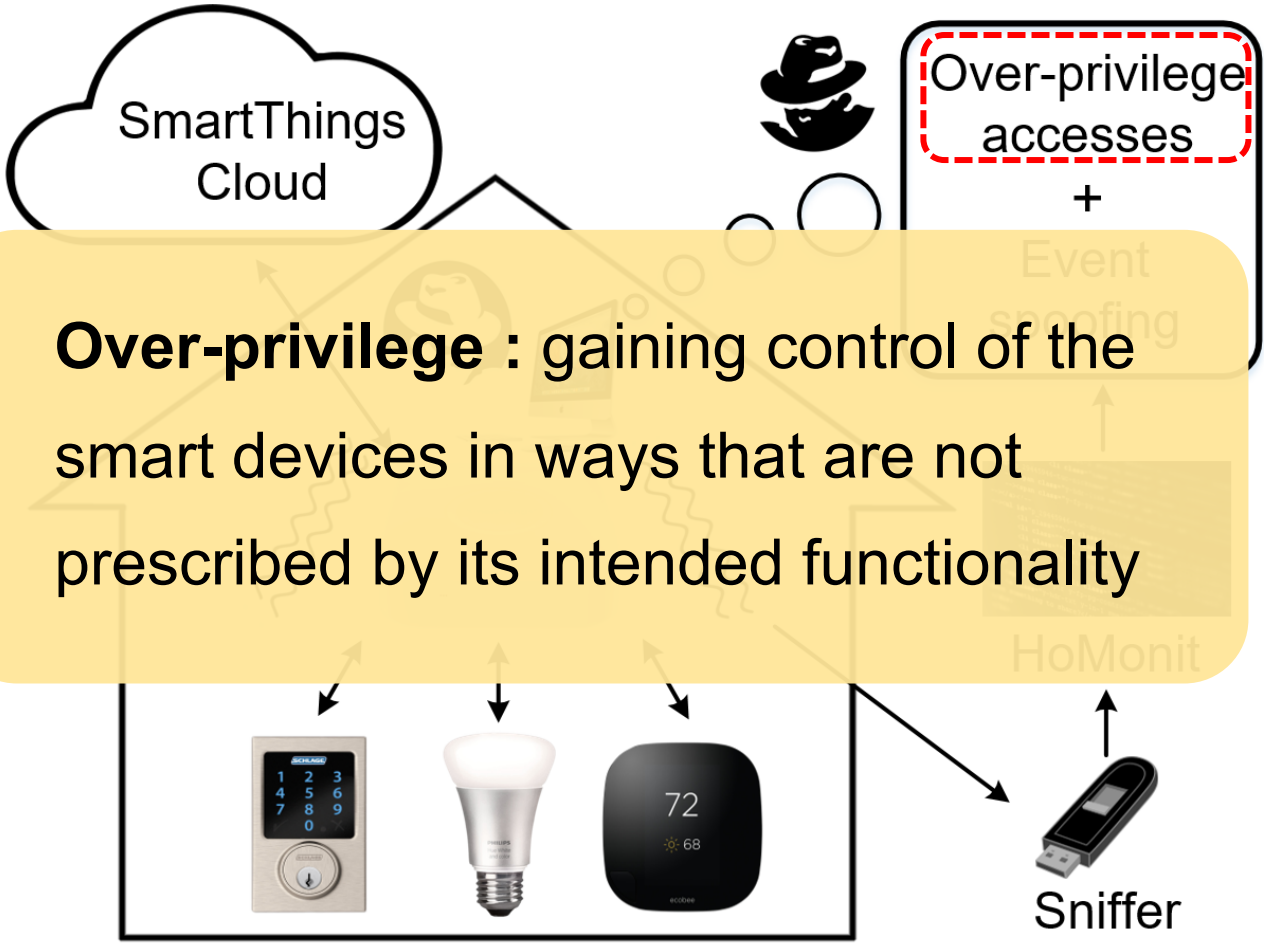
Threat Model



Threat Model

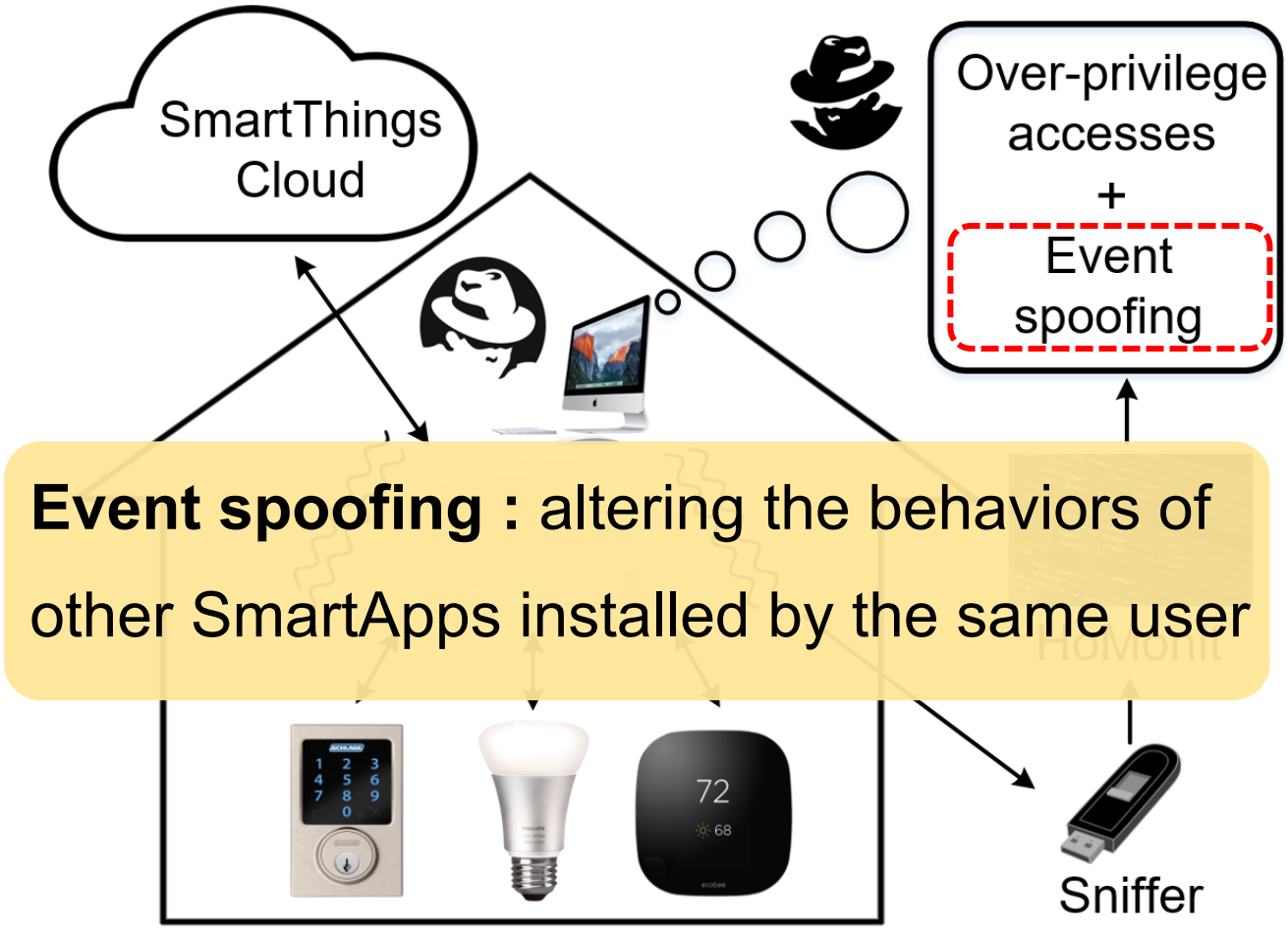
We consider attackers capable of exploiting the vulnerabilities of a benign SmartApp to perform tasks (*i.e.*, ***misbehaviors***).

Threat Model



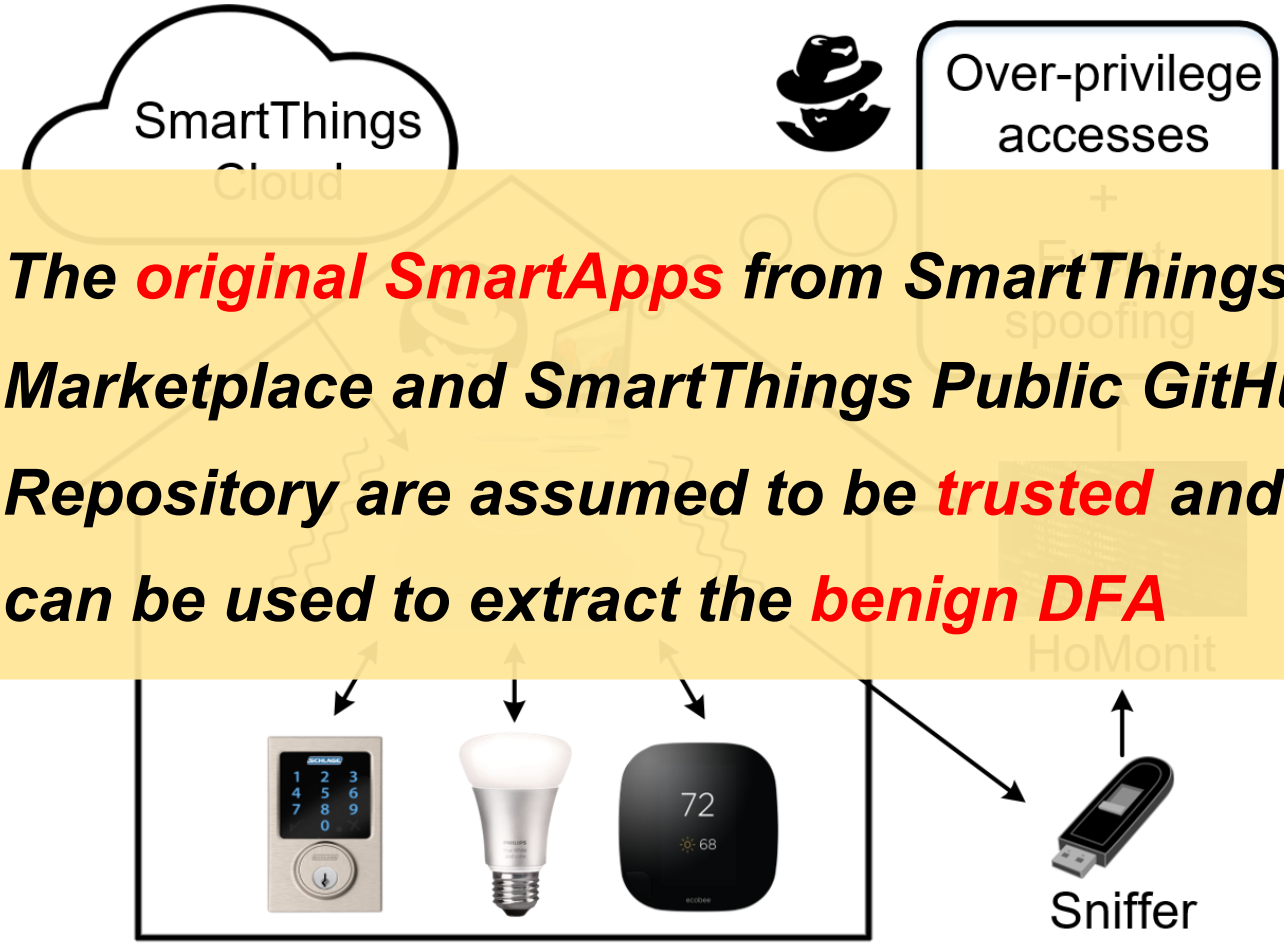
Over-privilege : gaining control of the smart devices in ways that are not prescribed by its intended functionality

Threat Model



Event spoofing : altering the behaviors of other SmartApps installed by the same user

Assumption



The original SmartApps from SmartThings Marketplace and SmartThings Public GitHub Repository are assumed to be trusted and can be used to extract the benign DFA

Challenges

How to automatically extract the control logic (i.e., DFA) of both open-source and closed-source SmartApps?

Challenges

How to automatically extract the control logic (i.e., DFA) of both open-source and closed-source SmartApps?

How to automatically capture wireless traffic (i.e., ZigBee and Z-Wave) and conduct side-channel inference to detect SmartApp misbehaviors?

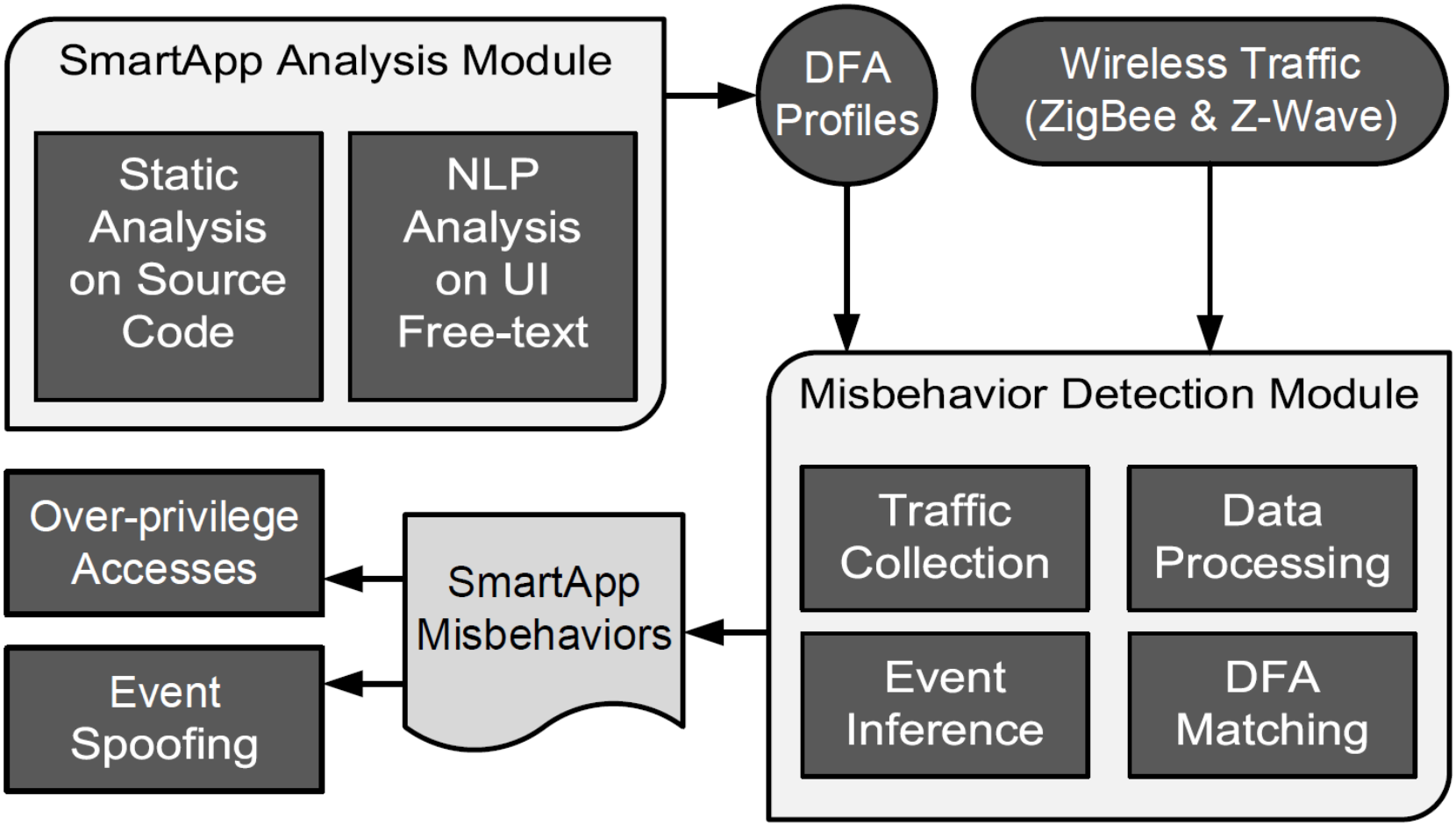
Challenges

How to automatically extract the control logic (i.e., DFA) of both open-source and closed-source SmartApps?

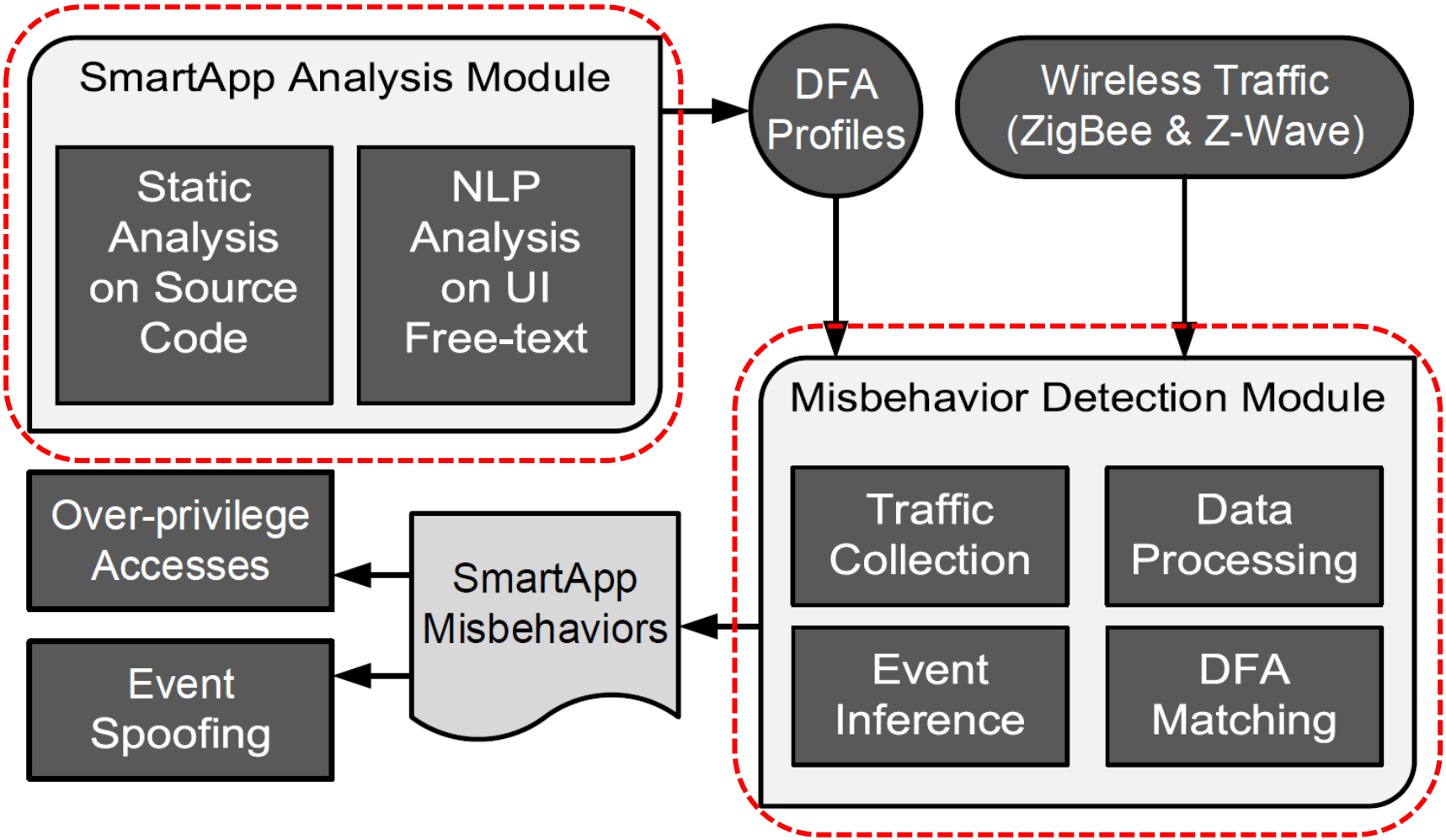
How to automatically capture wireless traffic (i.e., ZigBee and Z-Wave) and conduct side-channel inference to detect SmartApp misbehaviors?

How to implement and evaluate detection accuracy in a real-world smart home deployment?

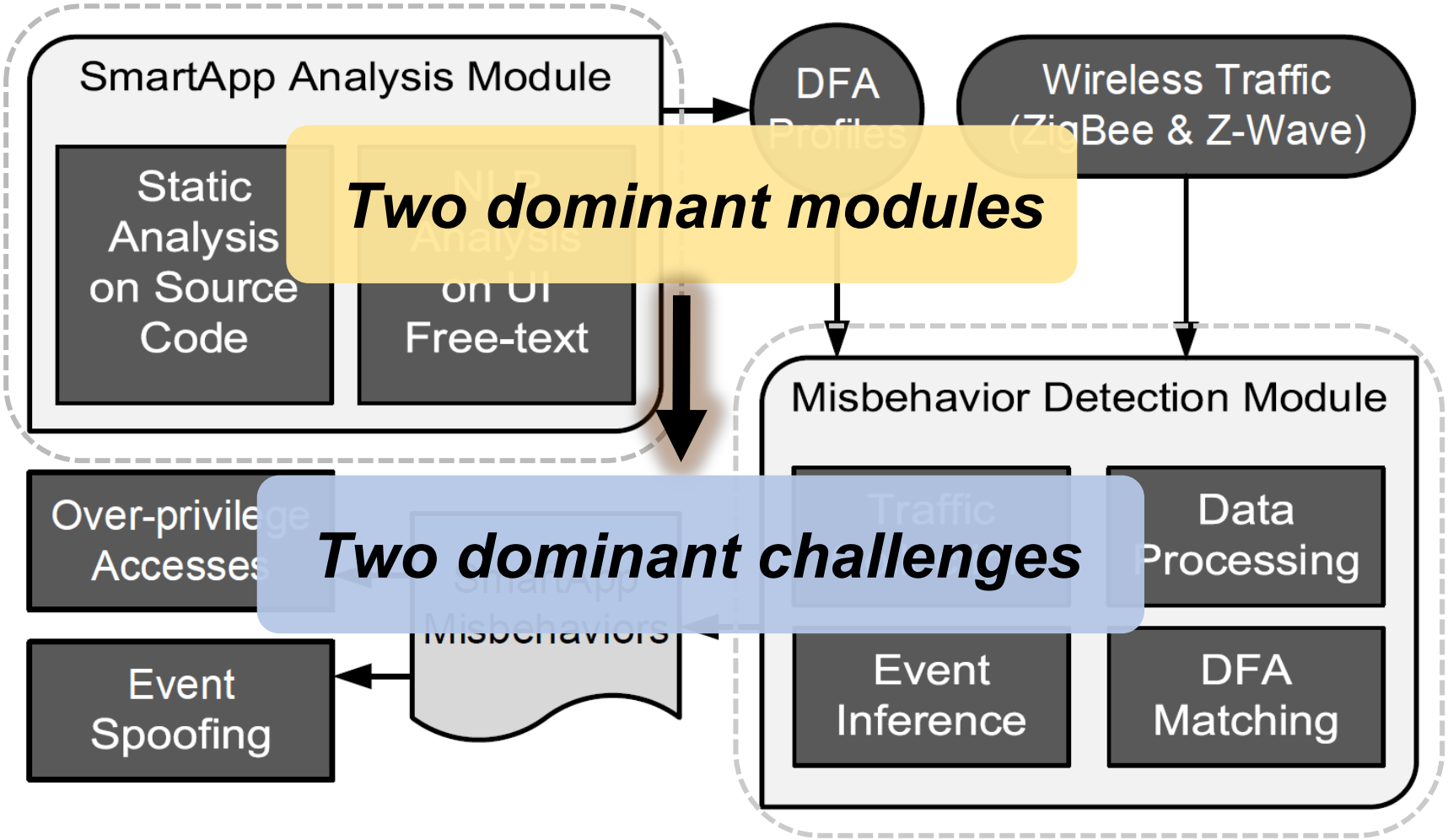
System Workflow



System Workflow



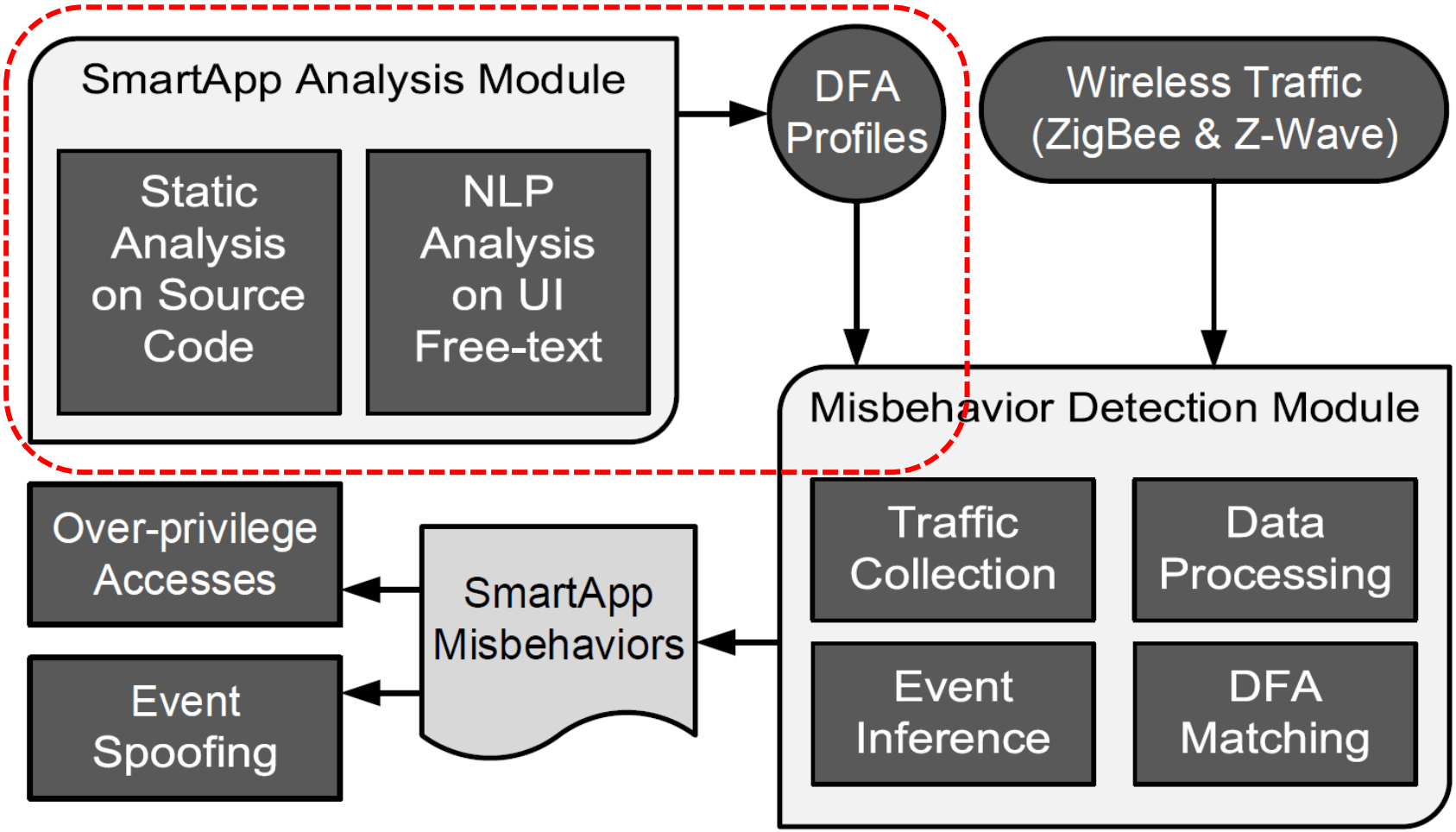
System Workflow



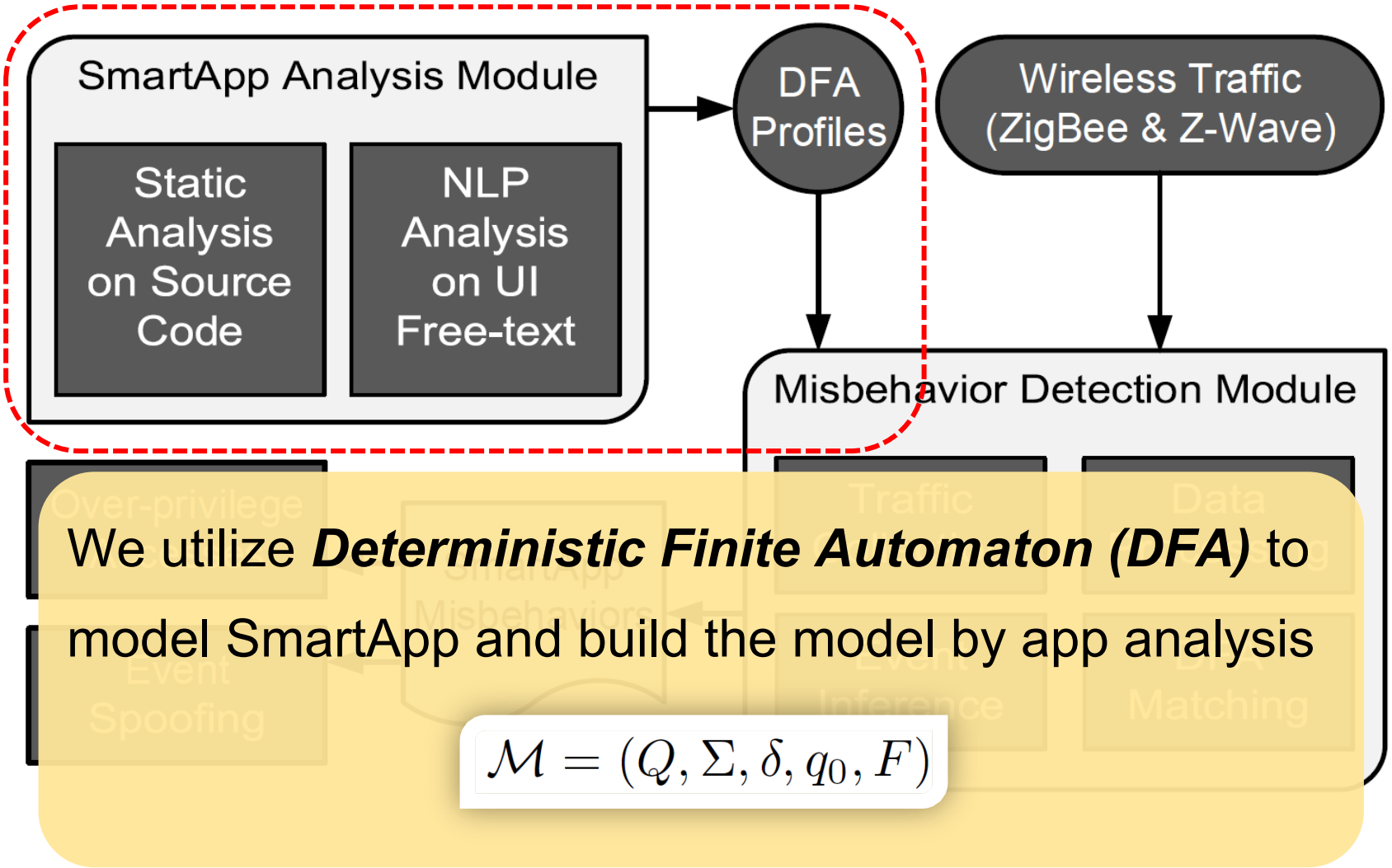
Outline

- Motivation
- Design Overview
- DFA Building via SmartApp Analysis
- Misbehavior Detection via Wireless Fingerprint
- Evaluation
- Discussion
- Conclusion

DFA Building via App Analysis



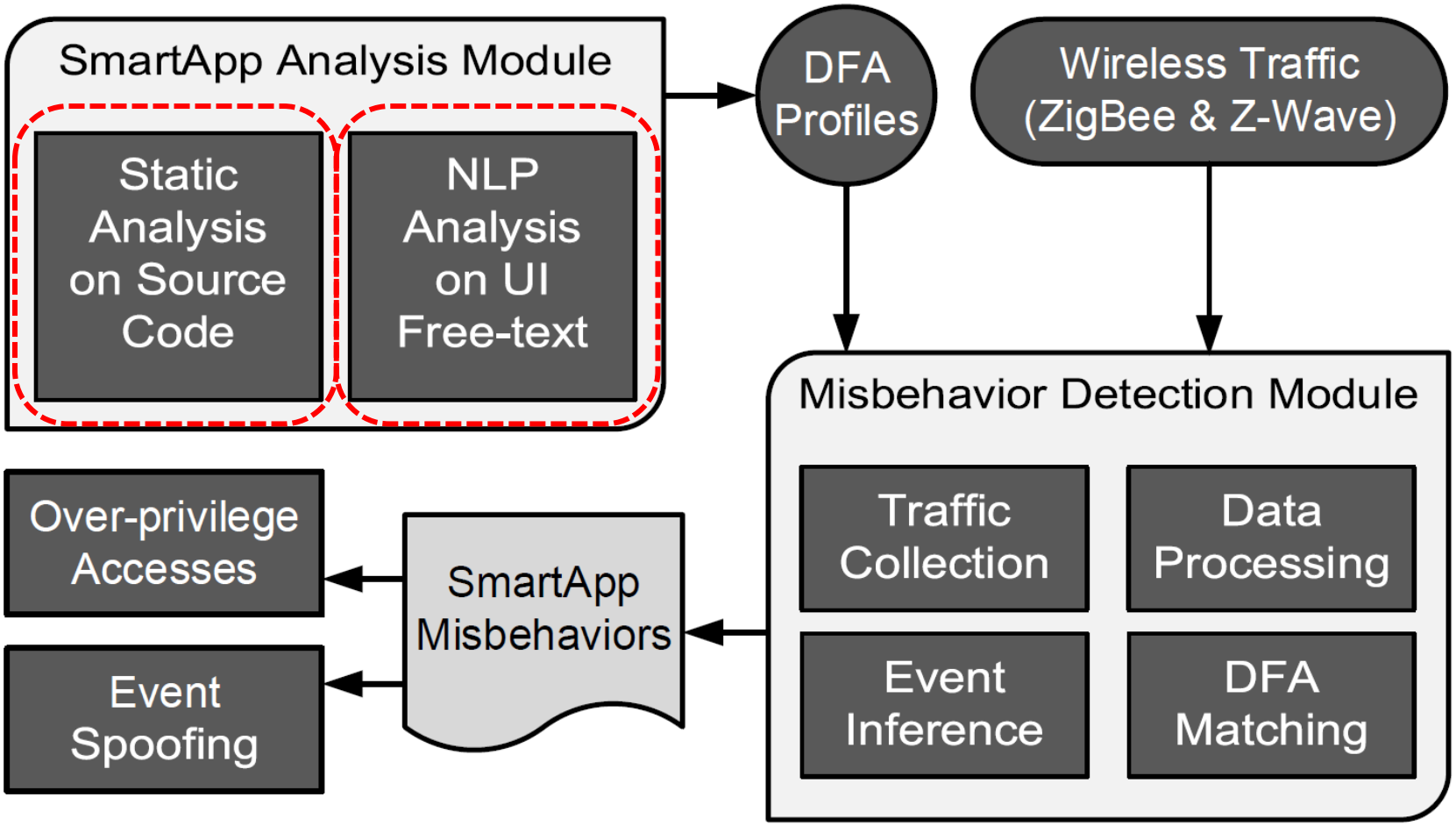
DFA Building via App Analysis



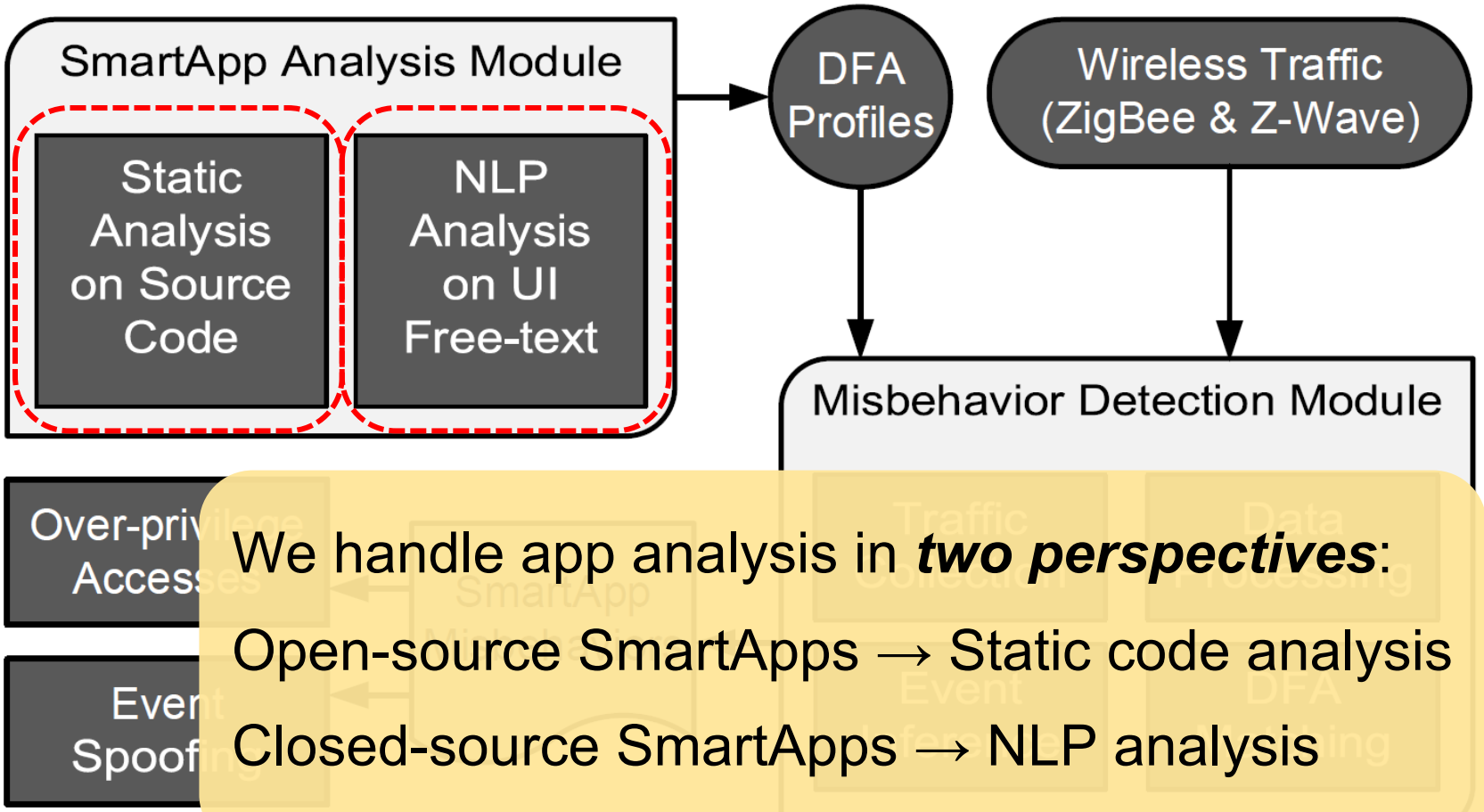
We utilize ***Deterministic Finite Automaton (DFA)*** to model SmartApp and build the model by app analysis

$$\mathcal{M} = (Q, \Sigma, \delta, q_0, F)$$

DFA Building via App Analysis



DFA Building via App Analysis



We handle app analysis in ***two perspectives***:

Open-source SmartApps → Static code analysis

Closed-source SmartApps → NLP analysis

DFA Building for Open-source App

```
definition(
  name: "Smart Light", namespace: "com.example",
  author: "example", category: "Convenience",
  description: "Turn light on when motion detected."
)

preferences {
  section("When there is movement...") {
    input "themotion", "capability.motionSensor",
          title: "Select a motion sensor"
  }
  section("Turn on a light...") {
    input "theswitch", "capability.switch",
          title: "Select a light"
  }
}

def installed() {
  subscribe(themotion, "motion.active", motionHandler)
}

def updated() {
  unsubscribe()
  subscribe(themotion, "motion.active", motionHandler)
}

def motionHandler(evt) {
  theswitch.on()
}
```

DFA Building for Open-source App

```
definition(
  name: "Smart Light", namespace: "com.example",
  author: "example", category: "Convenience",
  description: "Turn light on when motion detected."
)

preferences {
  section("When there is movement...") {
    input "themotion", "capability.motionSensor",
      title: "Select a motion sensor"
  }
  section("Turn on a light...") {
    input "theswitch", "capability.switch",
      multiple: false
  }
}

def installed() {
  subscribe(themotion, "motion.active", motionHandler)
}

def updated() {
  unsubscribe()
  subscribe(themotion, "motion.active", motionHandler)
}

def motionHandler(evt) {
  theswitch.on()
}
```

Take a SmartApp ***Smart Light*** as example

It turns light on when motion detected

Therefore, it automates two devices

DFA Building for Open-source App

```
definition(
  name: "Smart Light", namespace: "com.example",
  author: "example", category: "Convenience",
  description: "Turn light on when motion detected."
)
preferences {
  section("When there is movement...") {
    input "themotion", "capability.motionSensor",
  }
  section("Turn on a light...") {
    input "theswitch", "capability.switch"
  }
}
def installed() {
  subscribe(themotion, "motion.active", motionHandler)
}
def updated() {
  unsubscribe()
  subscribe(themotion, "motion.active", motionHandler)
}
def motionHandler(evt) {
  theswitch.on()
}
```

We achieve DFA building in two steps:

Source code → AST → DFA

1. We use ***AstBuilder*** to convert code into AST during Groovy compilation phase
2. The translation from AST to DFA is completed based on ***SmartApp characteristics***

DFA Building for Open-source App

```
definition(
  name: "Smart Light", namespace: "com.example",
  author: "example", category: "Convenience",
  description: "Turn light on when motion detected."
)

preferences {
  section("When there is movement...") {
    input "themotion", "capability.motionSensor",
          title: "Select a motion sensor"
  }
  section("Turn on a light...") {
    input "theswitch", "capability.switch",
          title: "Select a light"
  }
}

def installed() {
  subscribe(themotion, "motion.active", motionHandler)
}

def updated() {
  unsubscribe()
  subscribe(themotion, "motion.active", motionHandler)
}

def motionHandler(evt) {
  theswitch.on()
}
```

DFA Building for Open-source App

```
definition(
  name: "Smart Light", namespace: "com.example",
  author: "example", category: "Convenience",
  description: "Turn light on when motion detected."
)

preferences {
  section("When there is movement...") {
    input "themotion", "capability.motionSensor",
  }
  section("Turn on a light...") {
    input "theswitch", "capability.switch"
  }
}

def installed() {
  subscribe(themotion, "motion.active", motionHandler)
}

def updated() {
  unsubscribe()
  subscribe(themotion, "motion.active", motionHandler)
}

def motionHandler(evt) {
  theswitch.on()
}
```

Preferences block statement

Extract the capabilities requested by DFA

Subscribe method

Further determine the specific symbols of DFA

DFA Building for Open-source App

```
definition(
  name: "Smart Light", namespace: "com.example",
  author: "example", category: "Convenience",
  description: "Turn light on when motion detected."
)

preferences {
  section("When there is movement...") {
    input "themotion", "capability.motionSensor",
          title: "Select a motion sensor"
  }
  section("Turn on a light...") {
    input "theswitch", "capability.switch",
          title: "Select a light"
  }
}

def installed() {
  subscribe(themotion, "motion.active", motionHandler)
}

def updated() {
  unsubscribe()
  subscribe(themotion, "motion.active", motionHandler)
}

def motionHandler(evt) {
  theswitch.on()
}
```

DFA Building for Open-source App

```
definition(
  name: "Smart Light", namespace: "com.example",
  author: "example", category: "Convenience",
  description: "Turn light on when motion detected."
)

preferences {
  section("When there is movement...") {
    input "themotion", "capability.motionSensor",
      title: "Select a motion sensor"
  }
  section("Turn on a light...") {
    input "theswitch", "capability.switch",
  }
}

def installed() {
  subscribe(themotion, "motion.active", motionHandler)
}

def updated() {
  unsubscribe()
  subscribe(themotion, "motion.active", motionHandler)
}

def motionHandler(evt) {
  theswitch.on()
}
```

Subscribe method

Indicate one transition from start state to intermediate state

Handler method (may contain if-else logic)

Determine how the state will transition to other states

Evaluation for Open-source Apps

Dataset	181 open-source SmartApps from *SmartThings Public GitHub Repository*
Success Rate	150 / 181 (82.9%) were successfully constructed
Marketplace	52 SmartApps, where 36 are open-source 32 / 36 (88.9%) were successfully constructed
Failure Cases	Working logic cannot be modeled as DFA (*e.g.*, the app does not automate any physical device)

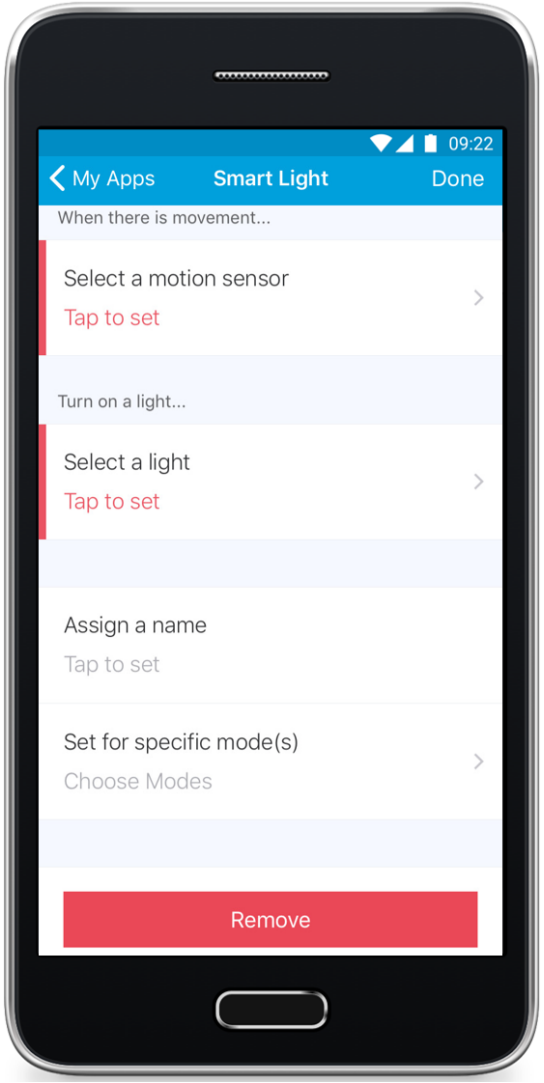
What if we do not have access to the source code of SmartApps?

DFA Building for Closed-source App

Still the same SmartApp

Smart Light

Now turn our eyes towards its app interface (i.e., UI)

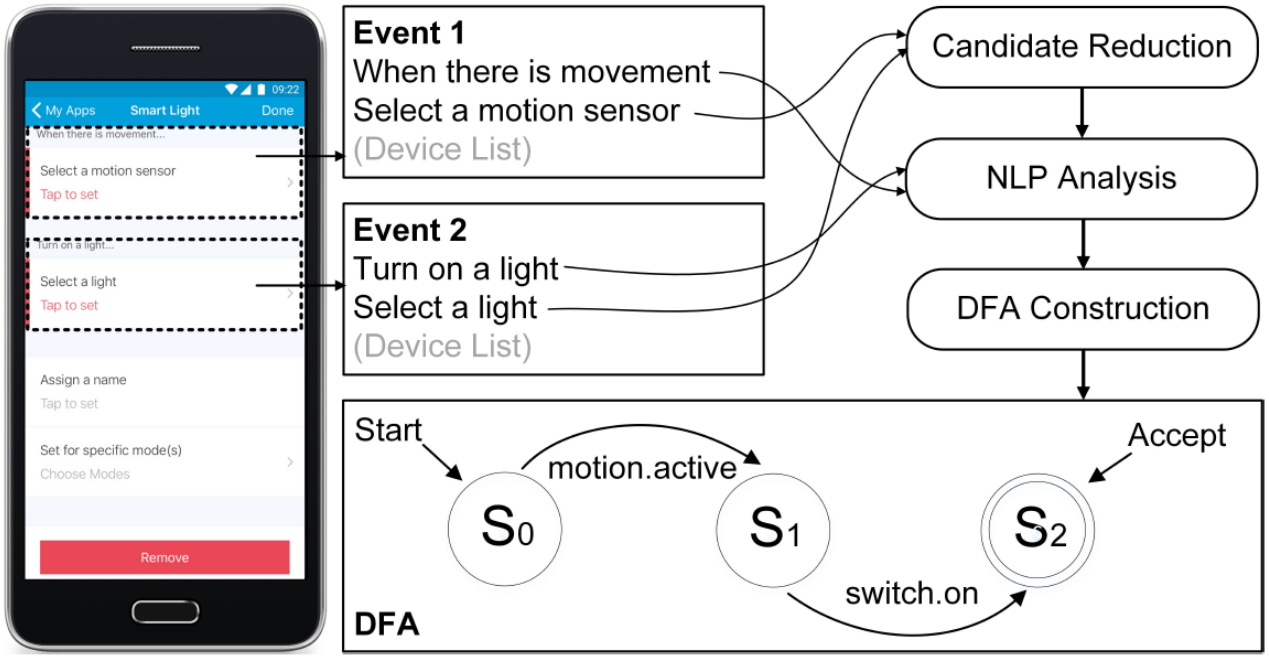


DFA Building for Closed-source App

Text extraction → Symbol inference → DFA building

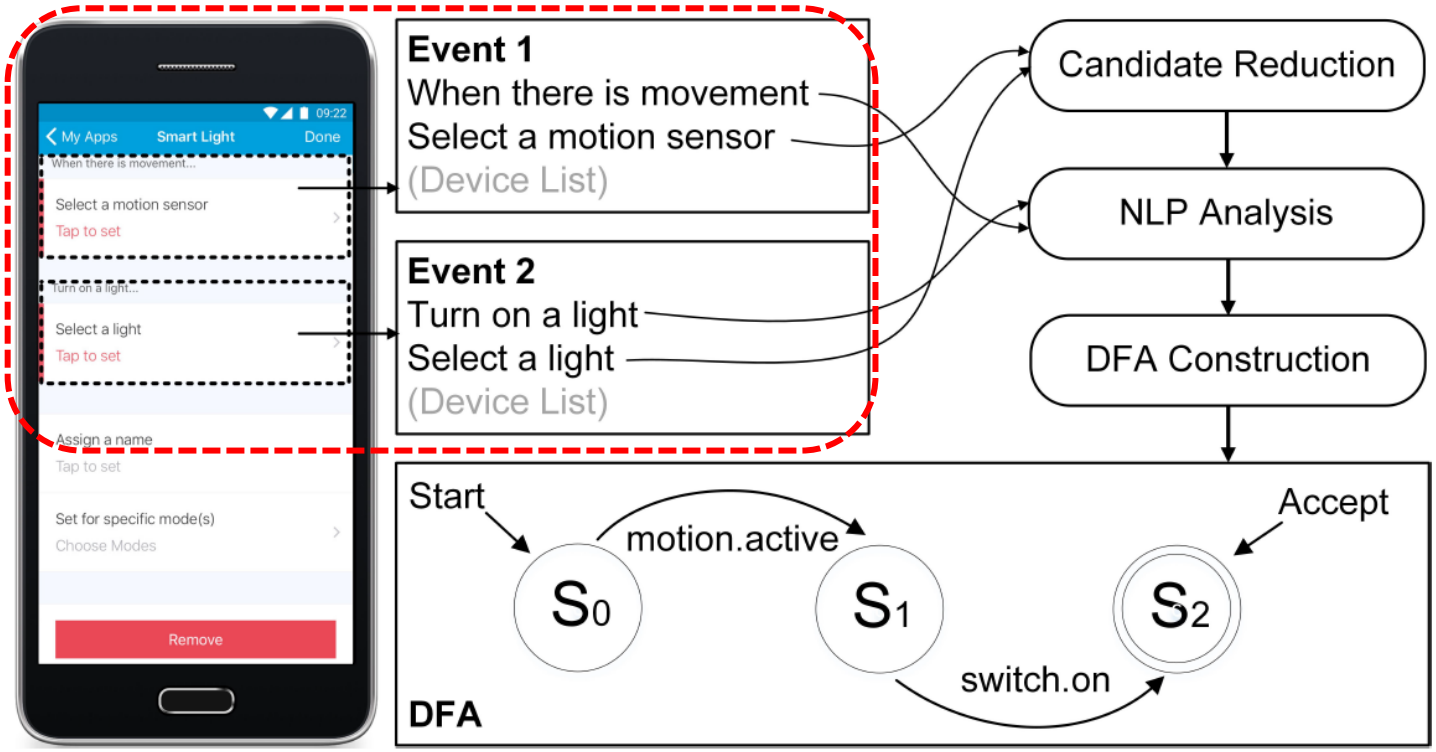
Our method

NLP analysis



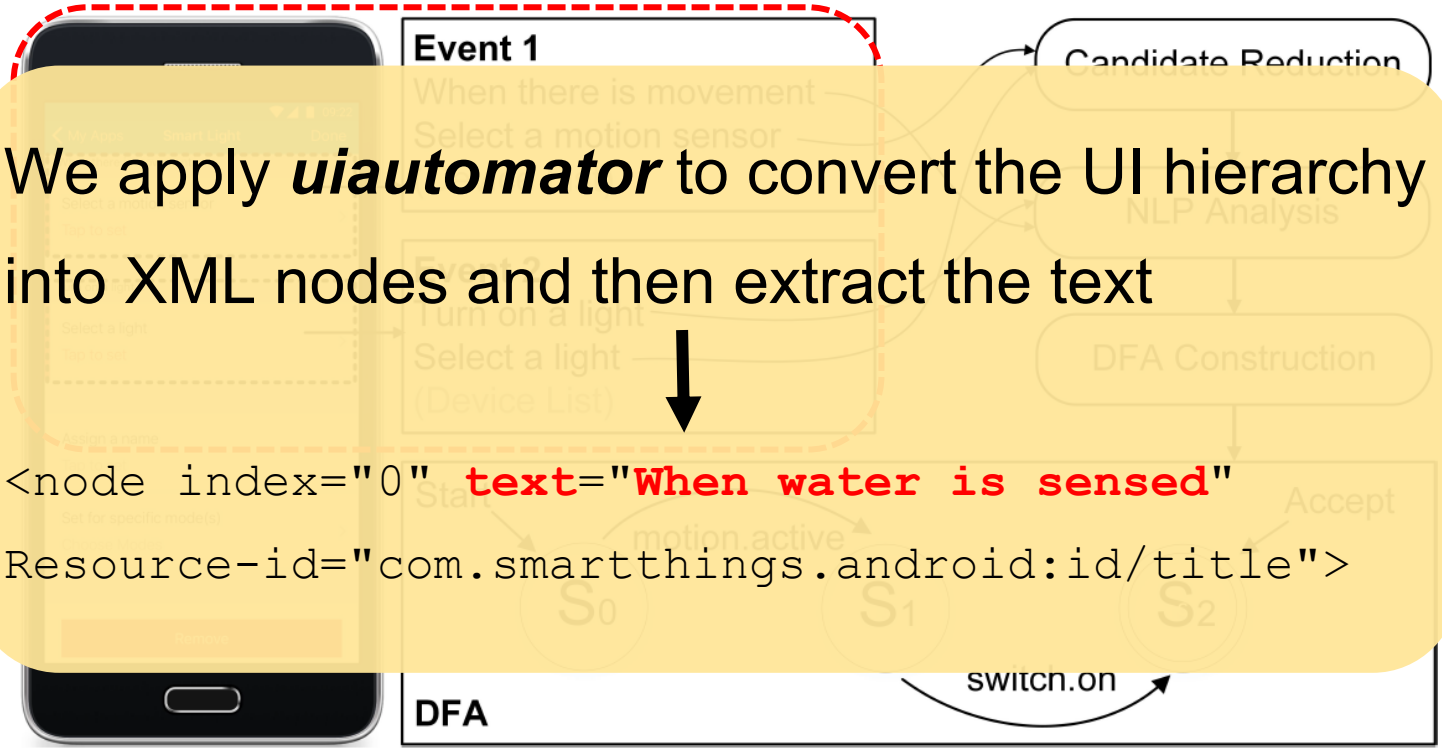
DFA Building for Closed-source App

Text extraction → Symbol inference → DFA building



DFA Building for Closed-source App

Text extraction → Symbol inference → DFA building

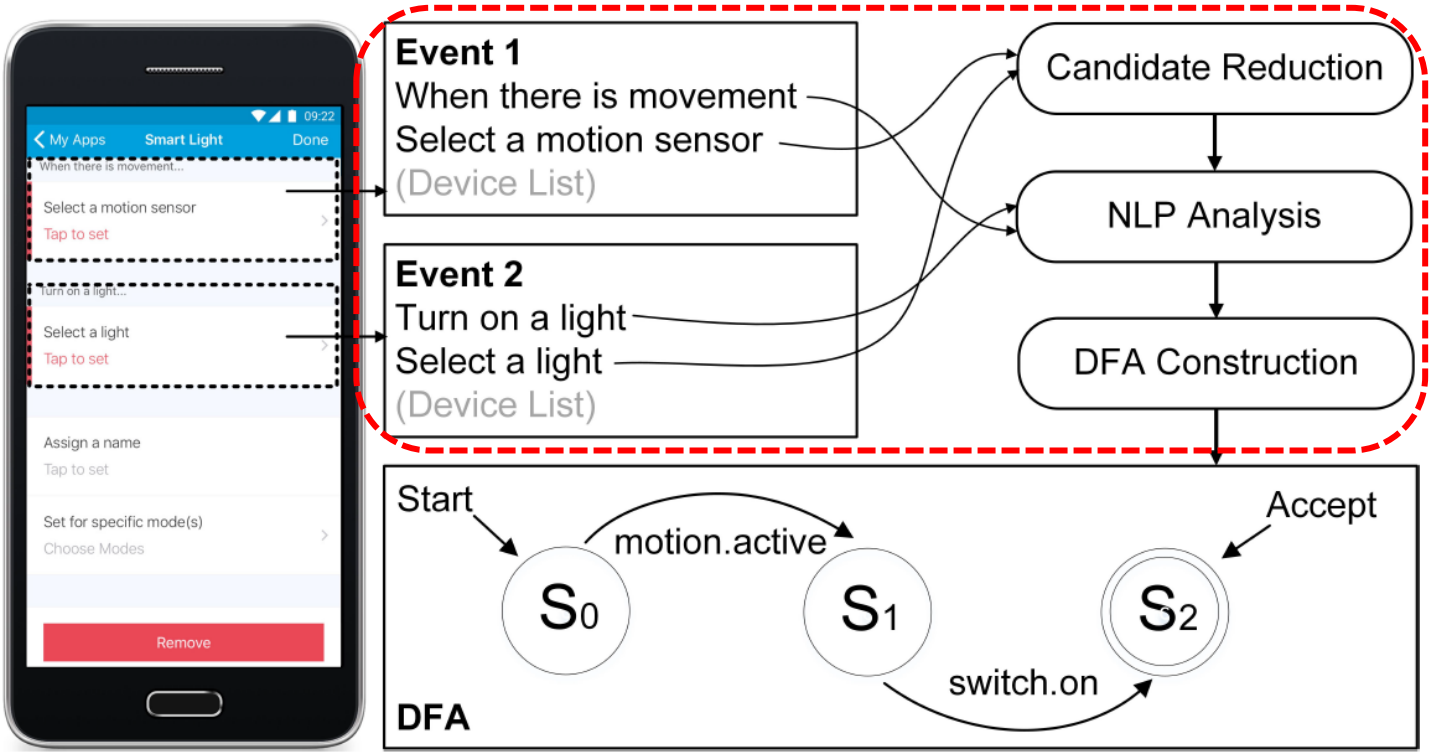


We apply ***uiautomator*** to convert the UI hierarchy into XML nodes and then extract the text

```
<node index="0" text="When water is sensed"
Resource-id="com.smartthings.android:id/title">
```

DFA Building for Closed-source App

Text extraction → ***Symbol inference*** → DFA building



DFA Building for Closed-source App

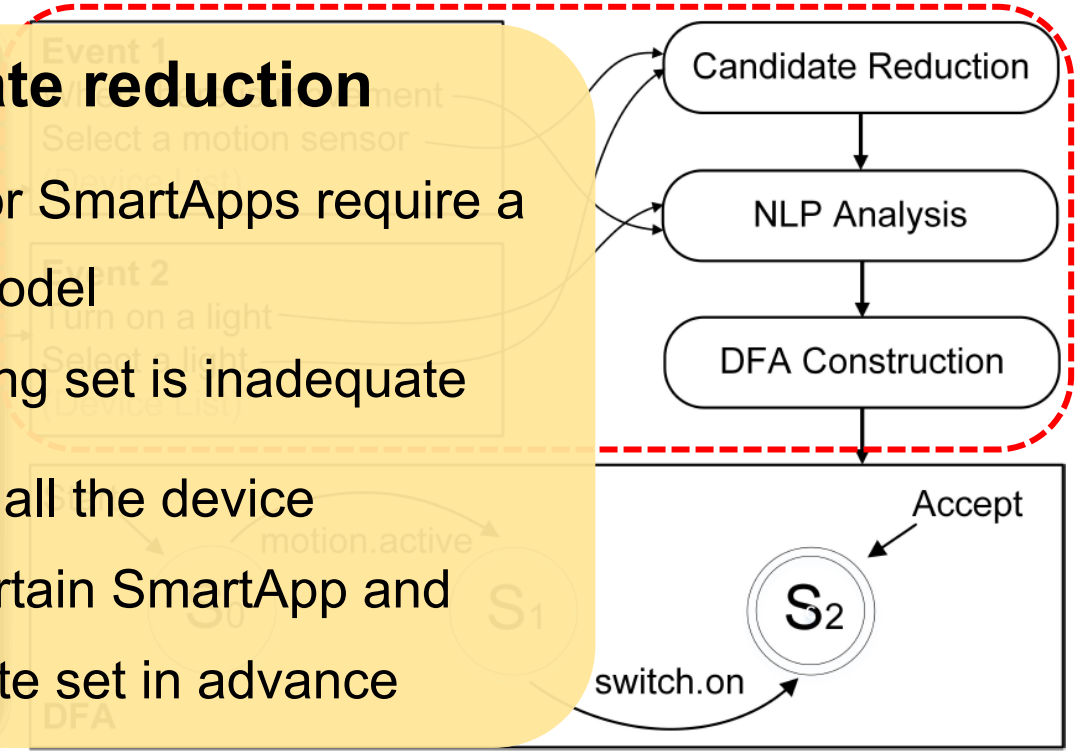
Text extraction → ***Symbol inference*** → DFA building

Candidate reduction

Text descriptions for SmartApps require a unique language model

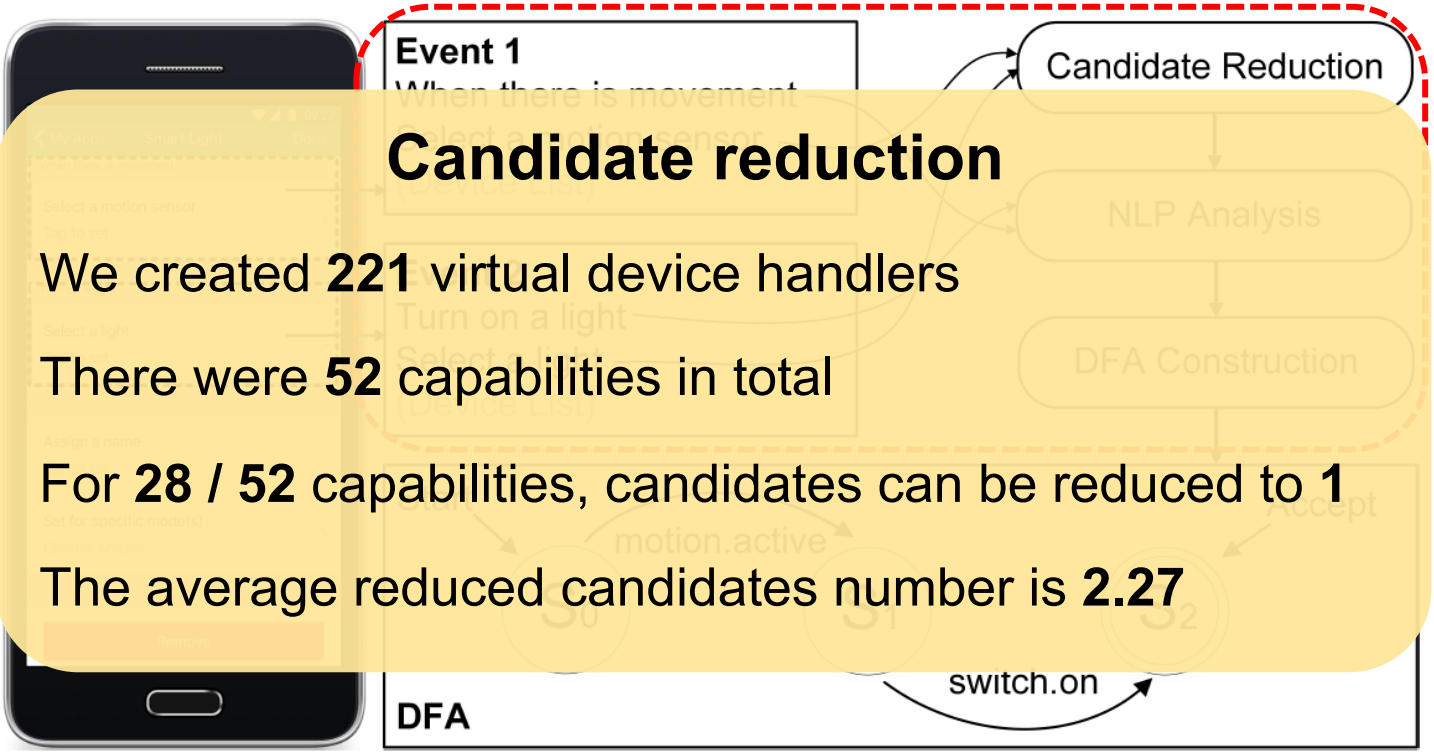
However, our training set is inadequate

Our strategy : test all the device candidates for a certain SmartApp and reduce the candidate set in advance



DFA Building for Closed-source App

Text extraction → ***Symbol inference*** → DFA building



Candidate reduction

We created **221** virtual device handlers

There were **52** capabilities in total

For **28 / 52** capabilities, candidates can be reduced to **1**

The average reduced candidates number is **2.27**

DFA Building for Closed-source App

Text extraction → ***Symbol inference*** → DFA building

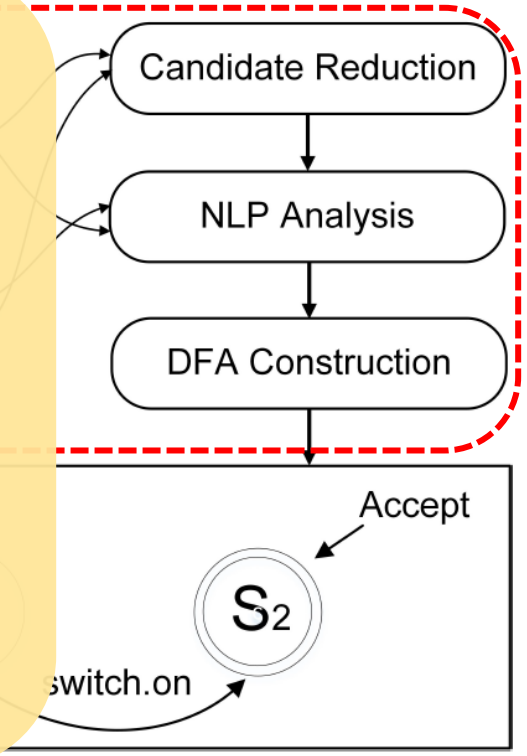
NLP analysis

NLTK : extract the noun phases and verb phrased for capability analysis

water → `Water Sensor`

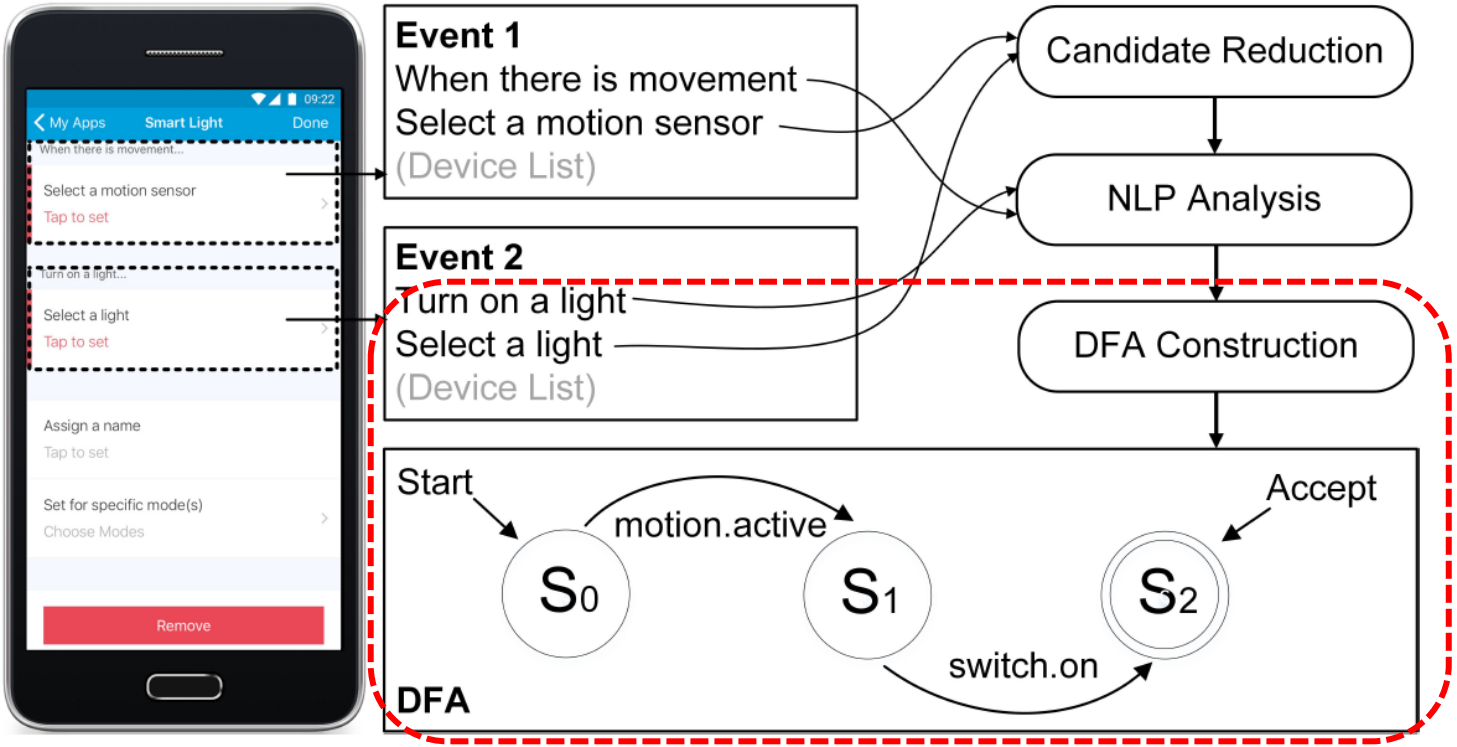
Word2Vec : analyze the context of the verb phrases and then determine the state of attributes or commands

when water is sensed → `water.wet`



DFA Building for Closed-source App

Text extraction → Symbol inference → ***DFA building***



DFA Building for Closed-source App

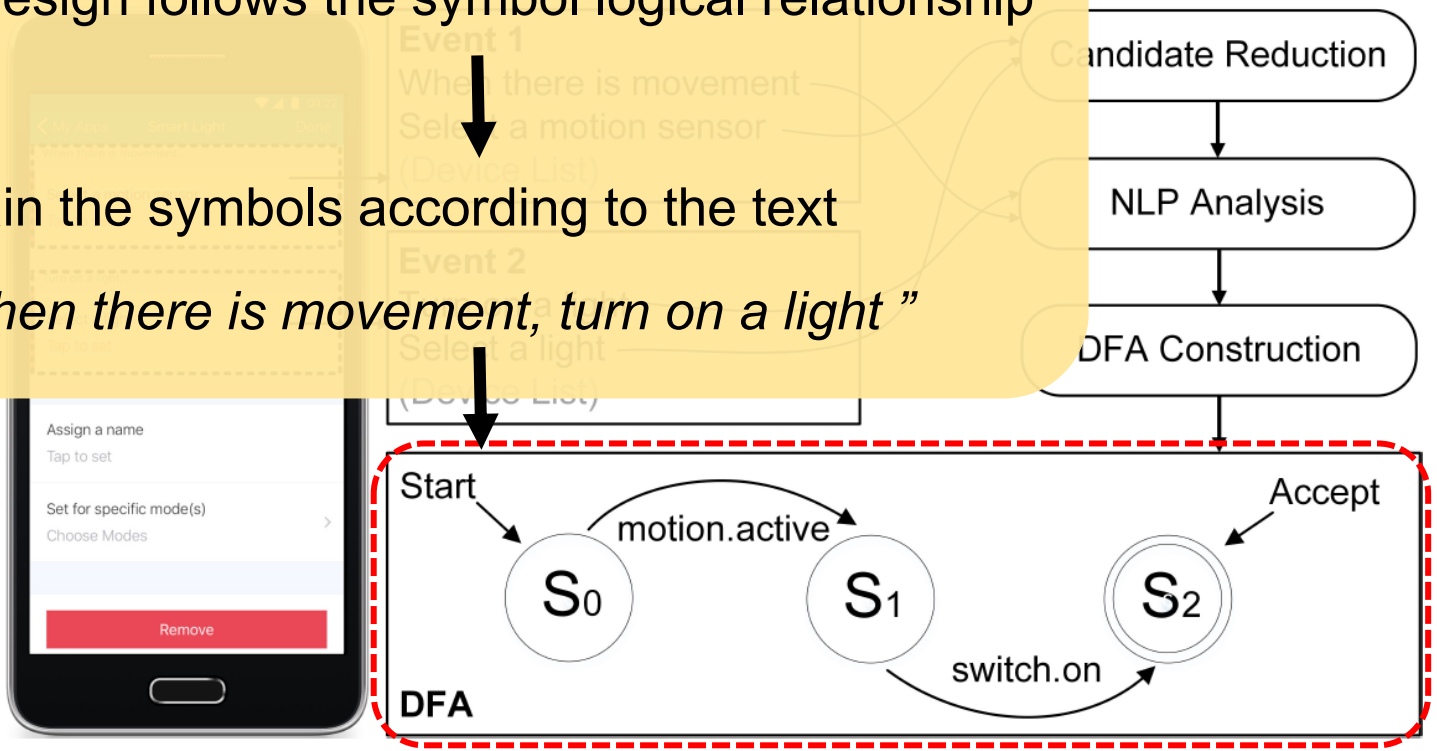
Text extraction → Symbol inference → *DFA building*

DFA building

UI design follows the symbol logical relationship

Chain the symbols according to the text

“ When there is movement, turn on a light ”



DFA Building for Closed-source App

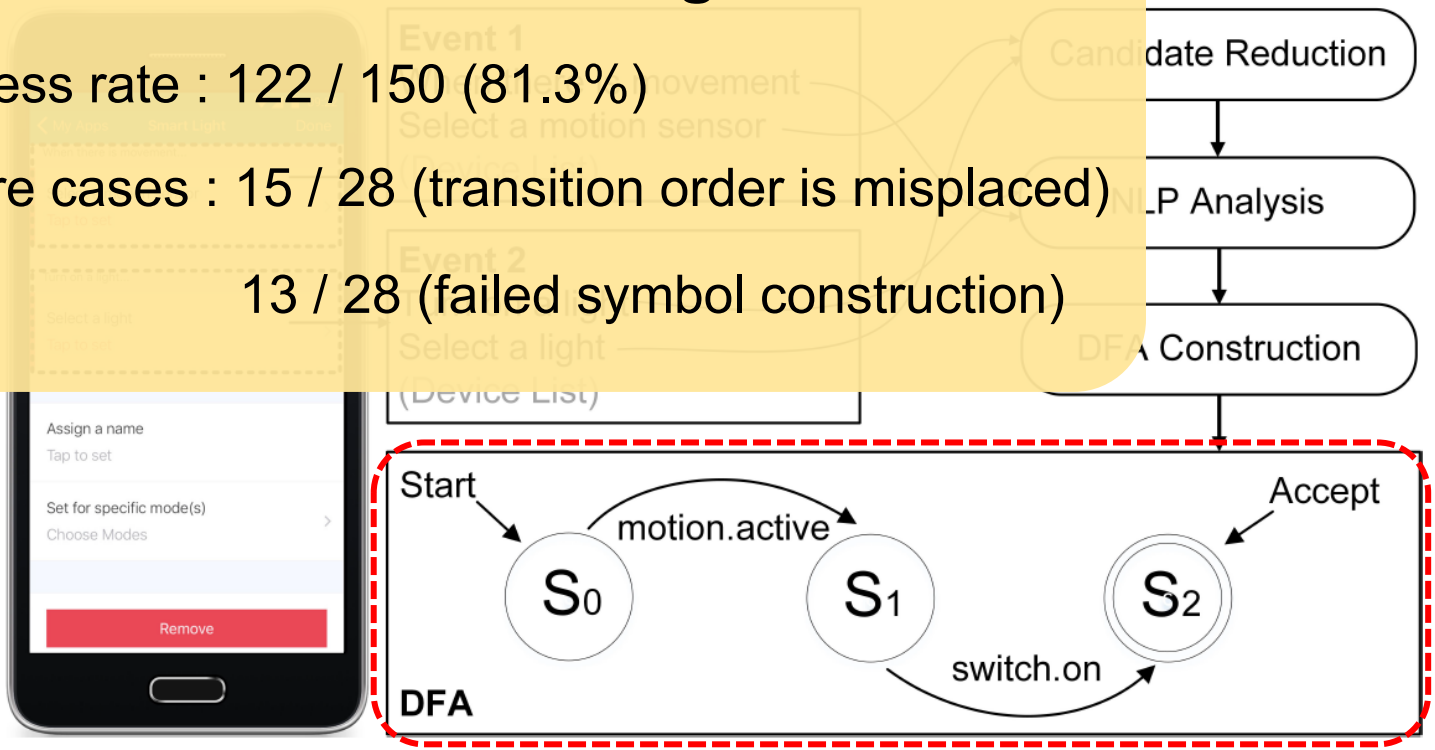
Text extraction → Symbol inference → ***DFA building***

DFA building

Success rate : 122 / 150 (81.3%)

Failure cases : 15 / 28 (transition order is misplaced)

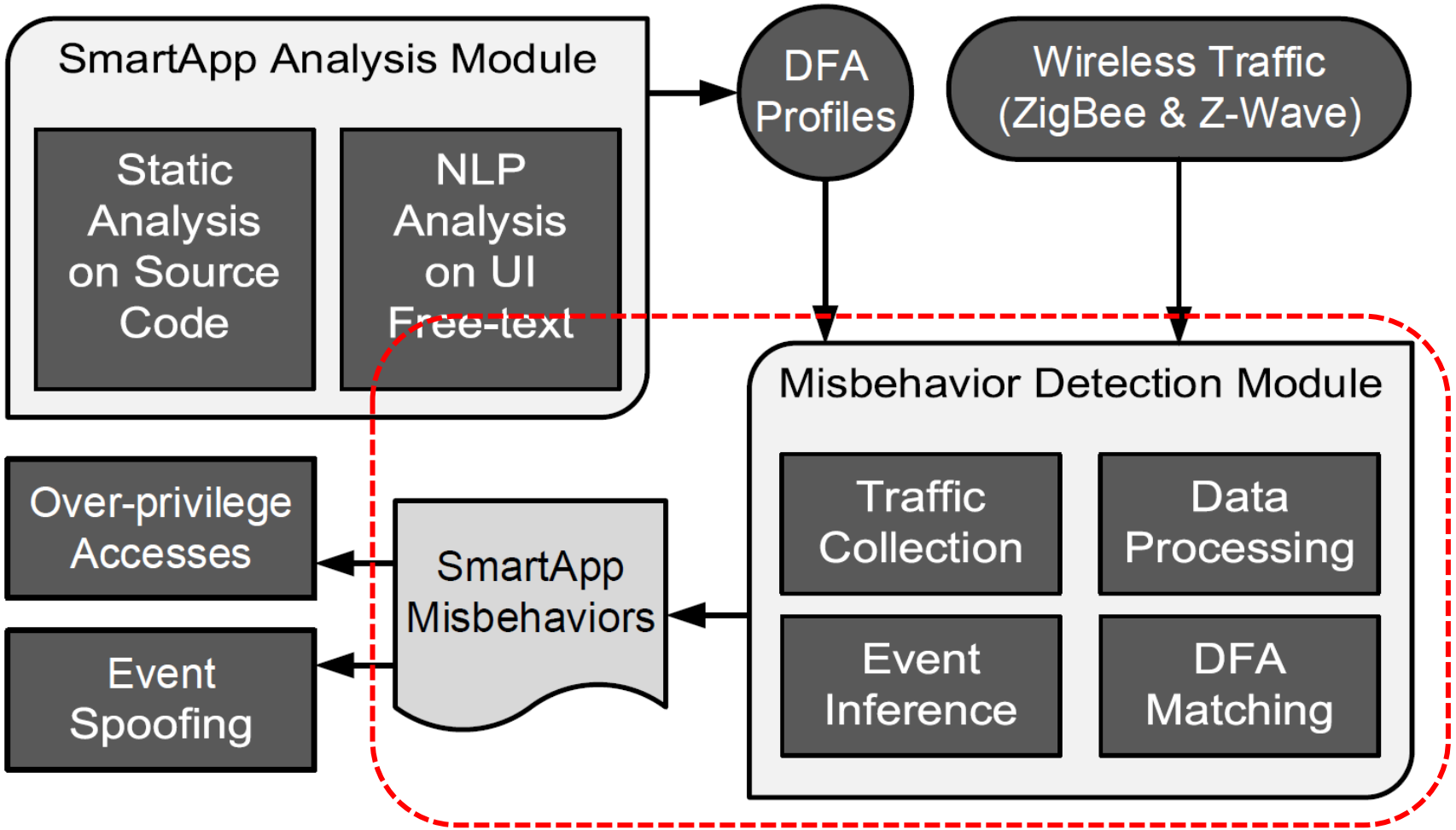
13 / 28 (failed symbol construction)



Outline

- Motivation
- Design Overview
- DFA Building via SmartApp Analysis
- Misbehavior Detection via Wireless Fingerprint
- Evaluation
- Discussion
- Conclusion

Misbehavior Detection via Wireless Fingerprint



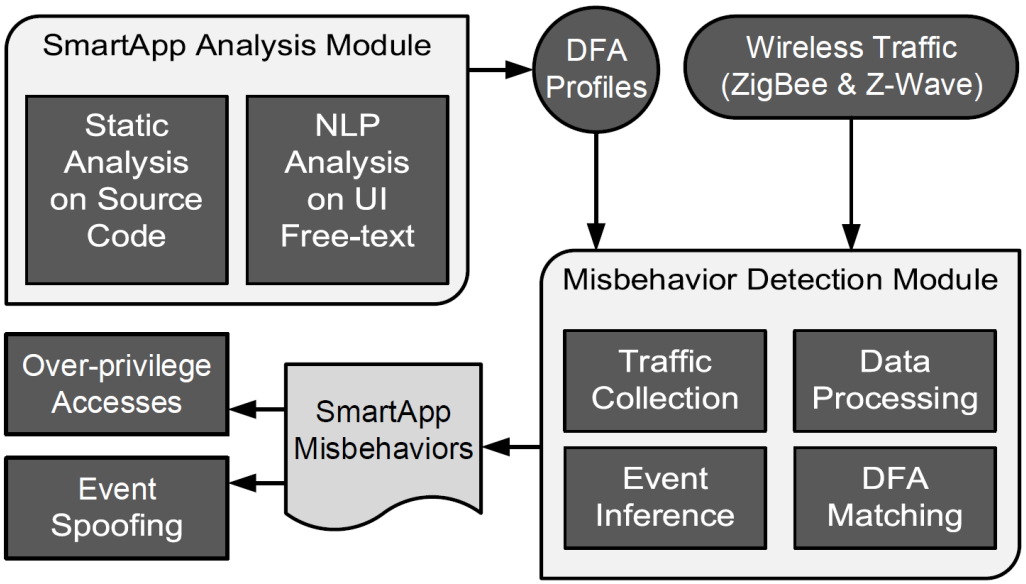
Misbehavior Detection via Wireless Fingerprint

Traffic collection

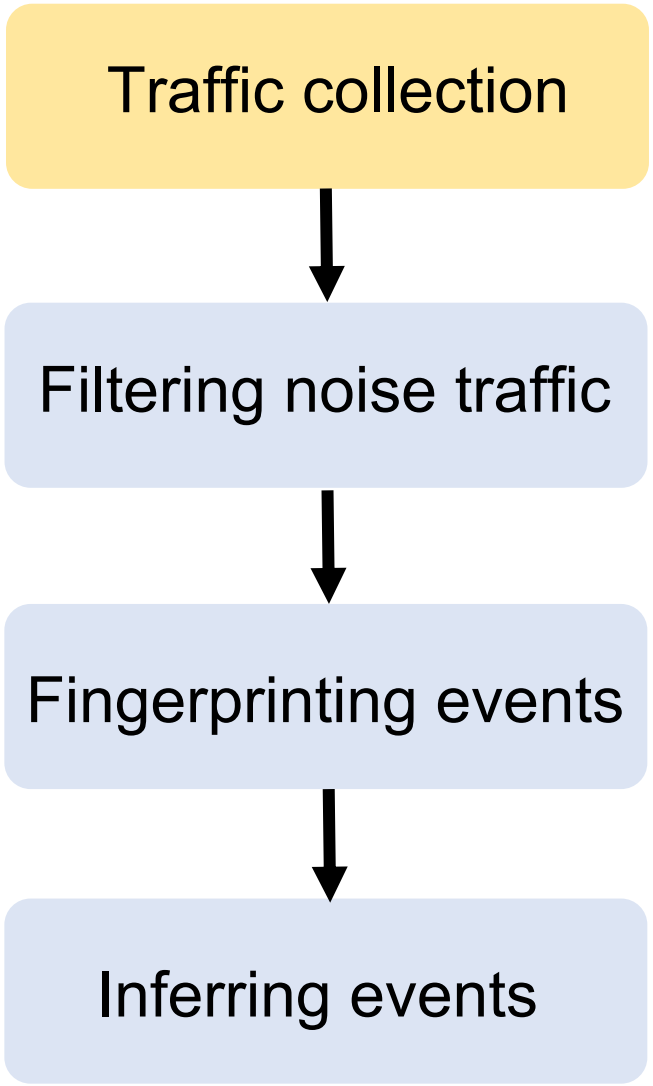
Filtering noise traffic

Fingerprinting events

Inferring events



Event Inference



Traffic collection

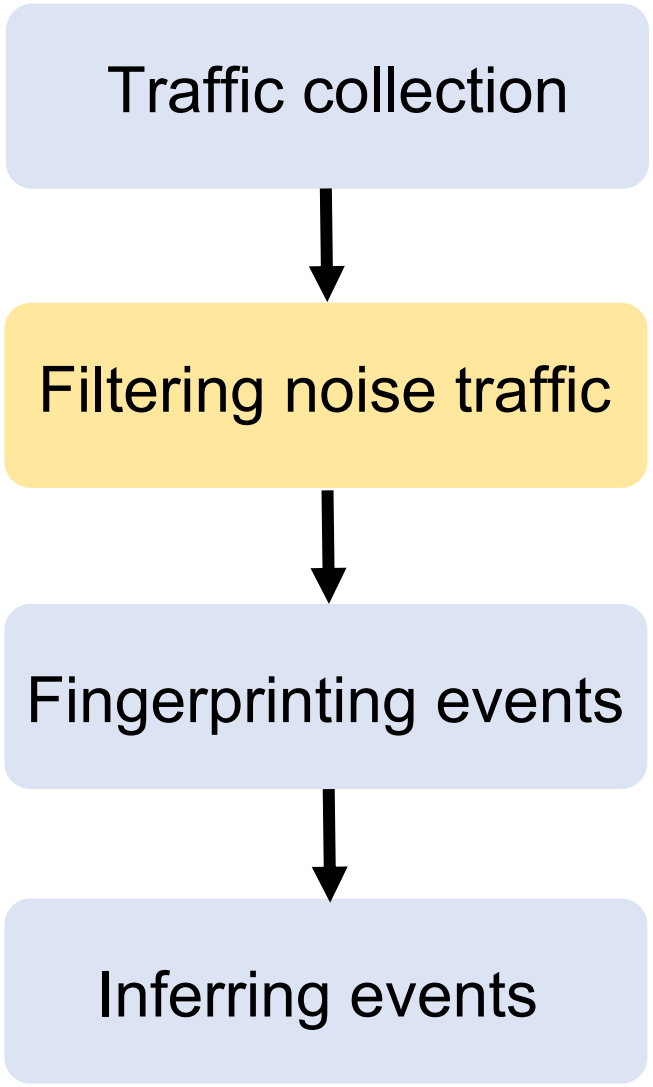
ZigBee traffic

- TI CC2531 USB Dongle
- 802.15.4 monitor
- 2.4GHz

Z-Wave traffic

- USRP
- Scapy-Radio
- 908.4MHz and 916MHz

Event Inference



Filtering noise traffic

Beacon packets

- Used for acknowledging data transmission and maintaining established connection

Retransmission packets

- Used for transmission failure

Unrelated traffic

- Traffic from devices using other wireless standards (*e.g.*, WiFi)

Event Inference

Traffic collection

↓

Filtering noise traffic

↓

Fingerprinting events

↓

Inferring events

Fingerprinting events

We adopt ***Levenshtein Distance*** to measure the sequence similarity between packet sequence

$$\mathcal{F}^{\phi} = \arg\min_{S_i^{\phi} \in \mathbb{S}^{\phi}} \frac{1}{\|\mathbb{S}^{\phi}\|} \sum_{\forall S_j^{\phi} \in \mathbb{S}^{\phi}} dist(S_i^{\phi}, S_j^{\phi}),$$

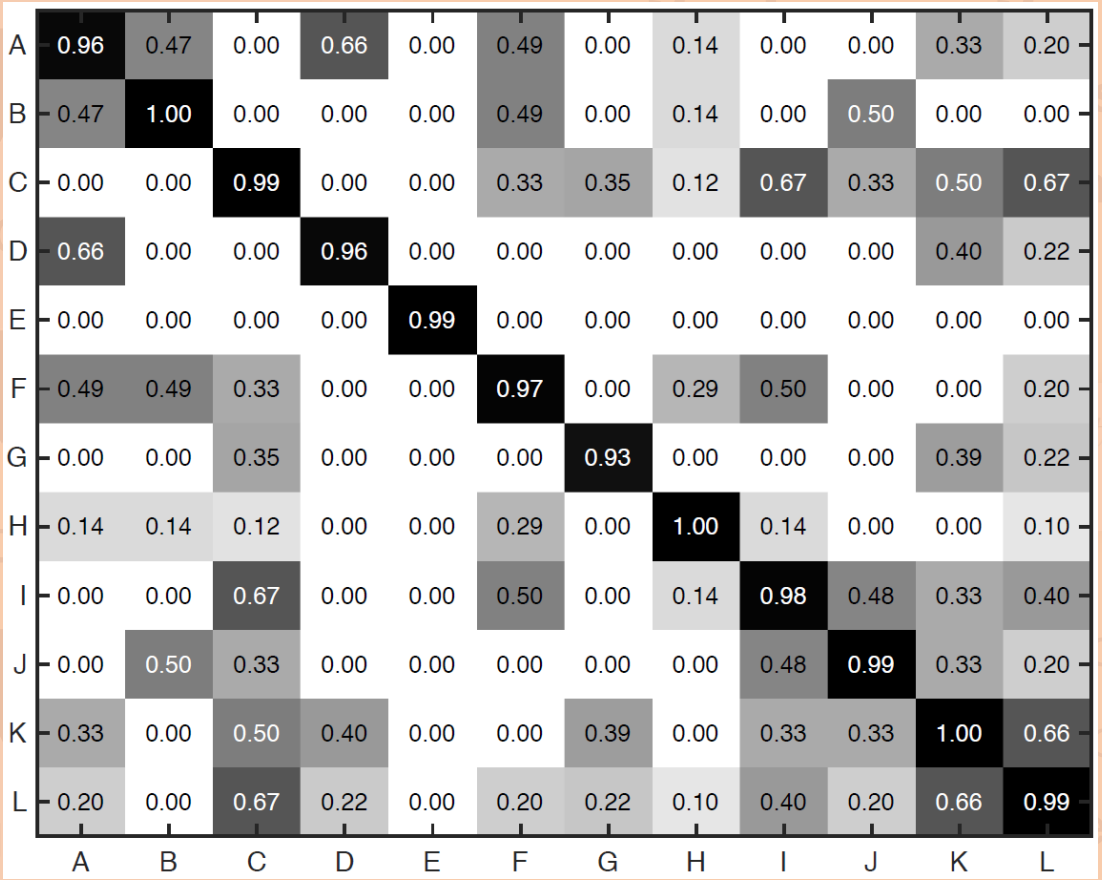
fingerprint

event type

collected packet sequence

Event Inference

Levenshtein ratio for 12 types of ZigBee events



Event Inference

Traffic collection

↓

Filtering noise traffic

↓

Fingerprinting events

↓

Inferring events

Inferring events

Partition the traffic flow into a set of ***bursts*** (a group of consecutive packets within ***burst threshold***)

↓

Match the burst with the possible fingerprint by calculating their *Levenshtein Distance*

Find smallest $dist(S_t^{d_i \leftrightarrow d_j}, \mathcal{F}^{\phi})$

Event Inference

Fingerprints for event types supported by 11 devices

Event Name	Device Name	Protocol	Fingerprint
water.wet[A] *water.dry* *temperature*	Samsung SmartThings Water Leak Sensor	ZigBee	54 ↑ 45 ↑ 54 ↑ 45 ↑ 53 ↑ 45 ↑
motion.active *motion.inactive* *temperature*[B]	Samsung SmartThings Motion Sensor	ZigBee	54 ↑ 54 ↑ 53 ↑ 45 ↑
switch.on[C] *switch.off*	Samsung SmartThings Outlet	ZigBee	50 ↓ 47 ↓ 47 ↓ 52 ↓ 50 ↓ 47 ↓ 47 ↓ 52 ↓
contact.open[D] *contact.closed* *acceleration.active* *acceleration.inactive* *temperature*	Samsung SmartThings Multipurpose Sensor (2016)	ZigBee	54 ↑ 54 ↑ 69 ↑ 65 ↑ 65 ↑ 65 ↑ · · · Occur after event *acceleration.active* finishes 53 ↑
contact.open *contact.closed* *acceleration.active*[E] *acceleration.inactive* *temperature*	Samsung SmartThings Multipurpose Sensor (2015)	ZigBee	54 ↑ 45 ↑ 54 ↑ 45 ↑ 69 ↑ 65 ↑ 65 ↑ 65 ↑ · · · Occur after event *acceleration.active* finishes 53 ↑ 45 ↑
beep[F] *rssi*[G] *presence.present*[H] *presence.not present*	Samsung SmartThings Arrival Sensor	ZigBee	50 ↓ 45 ↓ 52 ↑ 57 ↑ 48 ↑ 45 ↑ 45 ↑ 49 ↑ 45 ↑ 50 ↑ 45 ↑ 50 ↑ 45 ↑ 50 ↑ 45 ↑ Occur after there is no periodic event *rssi*
switch.on[I] *switch.off* *illuminance*[J] *setColorTemperature*[K] *setColor*[L]	Osram Lightify CLA 60 RGBW	ZigBee	50 ↓ 47 ↓ 50 ↓ 47 ↓ 53 ↓ 47 ↓ 54 ↓ 47 ↓ 52 ↓ 47 ↓ 50 ↓ 47 ↓ 54 ↓ 47 ↓ 52 ↓ 47 ↓ 52 ↓ 47 ↓
switch.on *switch.off*	Power Monitor Switch (TD1200Z1)	Z-Wave	13 ↓ 12 ↓ 10 ↓ 13 ↓ 12 ↓ 10 ↓
motion.active *motion.inactive*	Aeotec MultiSensor 6	Z-Wave	14 ↑ 21 ↑ 14 ↑ 21 ↑
contact.open *contact.closed*	Aeotec Door/Window Sensor 6	Z-Wave	17 ↑ 17 ↑ 17 ↑ 17 ↑
alarm.siren	Aeotec Siren (Gen 5)	Z-Wave	13 ↓ 34 ↓ 11 ↓ 33 ↓ 11 ↓ 21 ↓ 11 ↓

SmartApp Misbehavior Detection

DFA matching algorithm

Input	A sequence of events $\mathbb{E} = \{E_{t_1}^{\phi_1}, E_{t_2}^{\phi_2}, \ldots, E_{t_n}^{\phi_n}\}$
Process	Initially, $S_0 = q_0$. For each event in sequence, transition DFA if $E_{t_i}^{\phi_i} \in \Sigma \wedge \delta(S_i, E_{t_i}^{\phi_i}) = S_{i+1} \in Q$
Output	Accept state: $S_n \in F$ Otherwise a misbehaved SmartApp is detected

Outline

- Motivation
- Design Overview
- DFA Building via SmartApp Analysis
- Misbehavior Detection via Wireless Fingerprint
- Evaluation
- Discussion
- Conclusion

Testbed

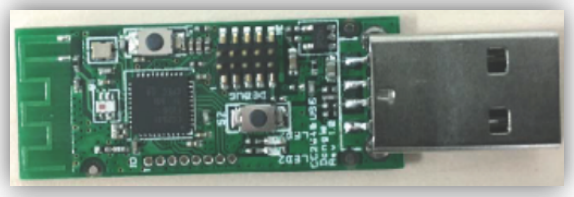
ZigBee sniffing : TI CC2531 USB Dongle

Z-Wave sniffing : two USRPs

Distance between SmartThings hub and *HoMonit* is 7 feet

Dataset : 30 open-source SmartApps from SmartThings Public GitHub Repository

Smart devices : 7 ZigBee devices and 4 Z-Wave devices



Micro-benchmark

Determining the burst threshold

Burst threshold

A parameter used to cluster captured wireless packets for the same event

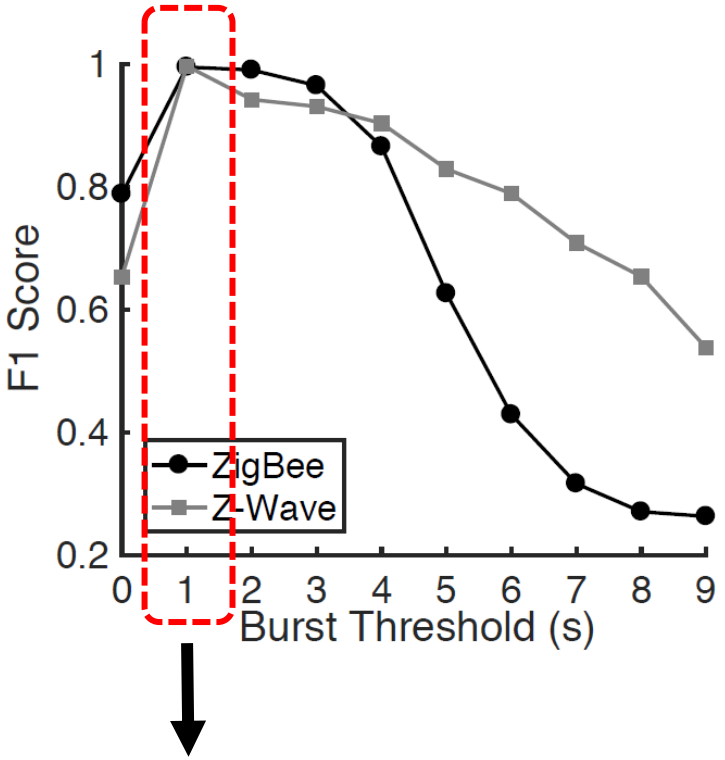
Experiment

Randomly select 8 different devices

Trigger each event for 50 times

Interval range: 3 ~ 10 secs

F1 score to measure the accuracy



We choose the burst threshold as **1s** for remainder evaluation

Micro-benchmark

SmartApp Inference Accuracy

Experiment

20 SmartApps for ZigBee devices

10 SmartApps for Z-Wave devices

Trigger each SmartApp for 20 times

Sniffer distance: 6 feet away

Burst threshold: 1s

Pre, Rec and F1 for measurement

Pre (precision)

The number of *correctly inferred* SmartApp invocation over the total number of inferences made

Rec (recall)

The number of *successfully inferred* SmartApp invocation over the 20 invocation of each SmartApp

F1 (F1-score)

The *harmonic average* of precision and recall

Micro-benchmark

SmartApp inference accuracy result

SmartApp	Protocol	Devices	Pre	Rec	F1
Lights Off When Closed	ZigBee	Samsung SmartThings Multipurpose Sensor (2015), Osram Lightify CLA 60 RGBW	1.00	0.90	0.95
Turn It On When It Opens	ZigBee	Samsung SmartThings Multipurpose Sensor (2016), Samsung SmartThings Outlet	1.00	0.95	0.97
Darken Behind Me	ZigBee	Samsung SmartThings Motion Sensor, Osram Lightify CLA 60 RGBW	1.00	0.95	0.97
Let There Be Light	ZigBee	Samsung SmartThings Multipurpose Sensor (2015), Osram Lightify CLA 60 RGBW	1.00	0.95	0.97
Monitor On Sense	ZigBee	Samsung SmartThings Multipurpose Sensor (2016), Samsung SmartThings Outlet	1.00	0.80	0.89
Big Turn On	ZigBee	Samsung SmartThings Outlet	1.00	1.00	1.00
Big Turn Off	ZigBee	Samsung SmartThings Outlet	1.00	1.00	1.00
Presence Change Push	ZigBee	Samsung SmartThings Arrival Sensor	1.00	1.00	1.00
Door Knocker	ZigBee	Samsung SmartThings Multipurpose Sensor (2016)	1.00	0.95	0.97
Let There Be Dark	ZigBee	Samsung SmartThings Multipurpose Sensor (2015), Osram Lightify CLA 60 RGBW	1.00	0.80	0.89
Flood Alert	ZigBee	Samsung SmartThings Water Leak Sensor	1.00	1.00	1.00
Turn It On When I'm here	ZigBee	Samsung SmartThings Arrival Sensor, Samsung SmartThings Outlet	1.00	1.00	1.00
The Gun Case Moved	ZigBee	Samsung SmartThings Multipurpose Sensor (2015)	1.00	1.00	1.00
It Moved	ZigBee	Samsung SmartThings Multipurpose Sensor (2016)	1.00	1.00	1.00
Light Follows Me	ZigBee	Samsung SmartThings Motion Sensor, Osram Lightify CLA 60 RGBW	1.00	0.95	0.97
Undead Early Warning	ZigBee	Samsung SmartThings Multipurpose Sensor (2016), Osram Lightify CLA 60 RGBW	1.00	0.90	0.95
Cameras On When I'm Away	ZigBee	Samsung SmartThings Arrival Sensor, Samsung SmartThings Outlet	1.00	0.95	0.97
Brighten My Path	ZigBee	Samsung SmartThings Motion Sensor, Osram Lightify CLA 60 RGBW	1.00	1.00	1.00
Dry The Wetspot	ZigBee	Samsung SmartThings Water Leak Sensor, Samsung SmartThings Multipurpose Sensor (2016)	1.00	0.95	0.97
Curling Iron	ZigBee	Samsung SmartThings Motion Sensor, Samsung SmartThings Arrival Sensor, Samsung SmartThings Outlet	1.00	1.00	1.00
Big Turn On	Z-Wave	Power Monitor Switch (TD1200Z1)	1.00	0.90	0.95
Brighten My Path	Z-Wave	Aeotec MultiSensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.85	0.92
Darken Behind Me	Z-Wave	Aeotec MultiSensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.85	9.92
Forgiving Security	Z-Wave	Aeotec MultiSensor 6, Aeotec Siren (Gen 5), Power Monitor Switch (TD1200Z1)	1.00	0.90	0.95
Let There Be Dark	Z-Wave	Aeotec Door/Window Sensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.95	0.97
Let There Be Light	Z-Wave	Aeotec Door/Window Sensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.95	0.97
Light Follows Me	Z-Wave	Aeotec MultiSensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.90	0.95
Lights Off When Closed	Z-Wave	Aeotec Door/Window Sensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.95	0.97
Smart Security	Z-Wave	Aeotec MultiSensor 6, Aeotec Siren (Gen 5), Aeotec Door/Window Sensor 6	1.00	1.00	1.00
Turn It On When It Opens	Z-Wave	Aeotec Door/Window Sensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.95	0.97

Micro-benchmark

SmartApp inference accuracy result

SmartApp	Protocol	Devices	Pre	Rec	F1
Lights Off When Closed	ZigBee	Samsung SmartThings Multipurpose Sensor (2015), Osram Lightify CLA 60 RGBW	1.00	0.90	0.95
Turn It On When It Opens	ZigBee	Samsung SmartThings Multipurpose Sensor (2016), Samsung SmartThings Outlet	1.00	0.95	0.97
Darken Behind Me	ZigBee	Samsung SmartThings Motion Sensor, Osram Lightify CLA 60 RGBW	1.00	0.95	0.97
Let There Be Light	ZigBee	Samsung SmartThings Multipurpose Sensor (2015), Osram Lightify CLA 60 RGBW	1.00	0.95	0.97
Monitor On Sense	ZigBee	Samsung SmartThings Multipurpose Sensor (2016), Samsung SmartThings Outlet	1.00	0.80	0.89
Big Turn On	ZigBee	Samsung SmartThings Outlet	1.00	1.00	1.00
Big Turn Off	ZigBee	Samsung SmartThings Outlet	1.00	1.00	1.00
Presence Change Push	ZigBee	Samsung SmartThings Arrival Sensor	1.00	1.00	1.00
Door Knocker	ZigBee	Samsung SmartThings Multipurpose Sensor (2016)	1.00	0.95	0.97
Let There Be Dark	ZigBee	Samsung SmartThings Multipurpose Sensor (2015), Osram Lightify CLA 60 RGBW	1.00	0.80	0.89
Flood Alert	ZigBee	Samsung SmartThings Water Leak Sensor	1.00	1.00	1.00
Turn It On When I'm here	ZigBee	Samsung SmartThings Arrival Sensor, Samsung SmartThings Outlet	1.00	1.00	1.00
The Gun Case Moved	ZigBee	Samsung SmartThings Multipurpose Sensor (2015)	1.00	1.00	1.00
It Moved	ZigBee	Samsung SmartThings Multipurpose Sensor (2016)	1.00	1.00	1.00
Light Follows Me	ZigBee	Samsung SmartThings Motion Sensor, Osram Lightify CLA 60 RGBW	1.00	0.95	0.97
Undead Early Warning	ZigBee	Samsung SmartThings Multipurpose Sensor (2016), Osram Lightify CLA 60 RGBW	1.00	0.90	0.95
Cameras On When I'm Away	ZigBee	Samsung SmartThings Arrival Sensor, Samsung SmartThings Outlet	1.00	0.95	0.97
Brighten My Path	ZigBee	Samsung SmartThings Motion Sensor, Osram Lightify CLA 60 RGBW	1.00	1.00	1.00
Dry The Wetspot	ZigBee	Samsung SmartThings Water Leak Sensor, Samsung SmartThings Multipurpose Sensor (2016)	1.00	0.95	0.97
Curling Iron	ZigBee	Samsung SmartThings Motion Sensor, Samsung SmartThings Arrival Sensor, Samsung SmartThings Outlet	1.00	1.00	1.00
Big Turn On	Z-Wave	Power Monitor Switch (TD1200Z1)	1.00	0.90	0.95
Brighten My Path	Z-Wave	Aeotec MultiSensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.85	0.92
Darken Behind Me	Z-Wave	Aeotec MultiSensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.85	9.92
Forgiving Security	Z-Wave	Aeotec MultiSensor 6, Aeotec Siren (Gen 5), Power Monitor Switch (TD1200Z1)	1.00	0.90	0.95
Let There Be Dark	Z-Wave	Aeotec Door/Window Sensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.95	0.97
Let There Be Light	Z-Wave	Aeotec Door/Window Sensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.95	0.97
Light Follows Me	Z-Wave	Aeotec MultiSensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.90	0.95
Lights Off When Closed	Z-Wave	Aeotec Door/Window Sensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.95	0.97
Smart Security	Z-Wave	Aeotec MultiSensor 6, Aeotec Siren (Gen 5), Aeotec Door/Window Sensor 6	1.00	1.00	1.00
Turn It On When It Opens	Z-Wave	Aeotec Door/Window Sensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.95	0.97

Micro-benchmark

Smart

Exper

20 Sm

Sniffe

Burst

Pre, R

SmartApp inference accuracy result

SmartApp	Protocol	Devices	Pre	Rec	F1
Lights Off When Closed	ZigBee	Samsung SmartThings Multipurpose Sensor (2015), Osram Lightify CLA 60 RGBW	1.00	0.90	0.95
Turn It On When It Opens	ZigBee	Samsung SmartThings Multipurpose Sensor (2016), Samsung SmartThings Outlet	1.00	0.95	0.97
Darken Behind Me	ZigBee	Samsung SmartThings Motion Sensor, Osram Lightify CLA 60 RGBW	1.00	0.95	0.97
Let There Be Light	ZigBee	Samsung SmartThings Multipurpose Sensor (2015), Osram Lightify CLA 60 RGBW	1.00	0.95	0.97
Monitor On Sense	ZigBee	Samsung SmartThings Multipurpose Sensor (2016), Samsung SmartThings Outlet	1.00	0.80	0.89
Big Turn On			1.00	1.00	1.00
Big Turn Off			1.00	1.00	1.00
Presence Change I			1.00	1.00	1.00
Door Knocker			1.00	0.95	0.97
Let There Be Da			1.00	0.80	0.89
Flood Alert			1.00	1.00	1.00
Turn It On When I'm here	ZigBee	Samsung SmartThings Arrival Sensor, Samsung SmartThings Outlet	1.00	1.00	1.00
The Gun Case Moved	ZigBee	Samsung SmartThings Multipurpose Sensor (2015)	1.00	1.00	1.00
It Moved	ZigBee	Samsung SmartThings Multipurpose Sensor (2016)	1.00	1.00	1.00
Curling Iron	ZigBee	SmartThings Outlet	1.00	1.00	1.00
Big Turn On	Z-Wave	Power Monitor Switch (TD1200Z1)	1.00	0.90	0.95
Brighten My Path	Z-Wave	Aeotec MultiSensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.85	0.92
Darken Behind Me	Z-Wave	D1200Z1)	1.00	0.85	9.92
Forgiving Security	Z-Wave	Switch (TD1200Z1)	1.00	0.90	0.95
Let There Be Dark	Z-Wave	h (TD1200Z1)	1.00	0.95	0.97
Let There Be Light	Z-Wave	h (TD1200Z1)	1.00	0.95	0.97
Light Follows Me	Z-Wave	D1200Z1)	1.00	0.90	0.95
Lights Off When Closed	Z-Wave	h (TD1200Z1)	1.00	0.95	0.97
Smart Security	Z-Wave	Window Sensor 6	1.00	1.00	1.00
Turn It On When It Opens	Z-Wave	Aeotec Door/Window Sensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.95	0.97

SmartApp	Protocol
Lights Off When Closed	ZigBee

Devices
Samsung SmartThings Multipurpose Sensor (2015), Osram Lightify CLA 60 RGBW

Pre	Rec	F1
1.00	0.90	0.95

Micro-benchmark

SmartApp inference accuracy result

SmartApp	Protocol	Devices	Pre	Rec	F1
Light Follows Me	Z-Wave	Aeotec Door/Window Sensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.90	0.95
Lights Off When Closed	Z-Wave	Aeotec Door/Window Sensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.95	0.97
Smart Security	Z-Wave	Aeotec Door/Window Sensor 6	1.00	1.00	1.00
Turn It On When It Opens	Z-Wave	Aeotec Door/Window Sensor 6, Power Monitor Switch (TD1200Z1)	1.00	0.95	0.97

Result

Precision for all SmartApps are **1.00**

Average F1 score: **0.98** for ZigBee **0.96** for Z-Wave

F1 scores for **26** / 30 SmartApps are over **0.95**

Major false inference factors:

- Packet loss
- Unrelated wireless traffic
- Traffic jam splitting a burst

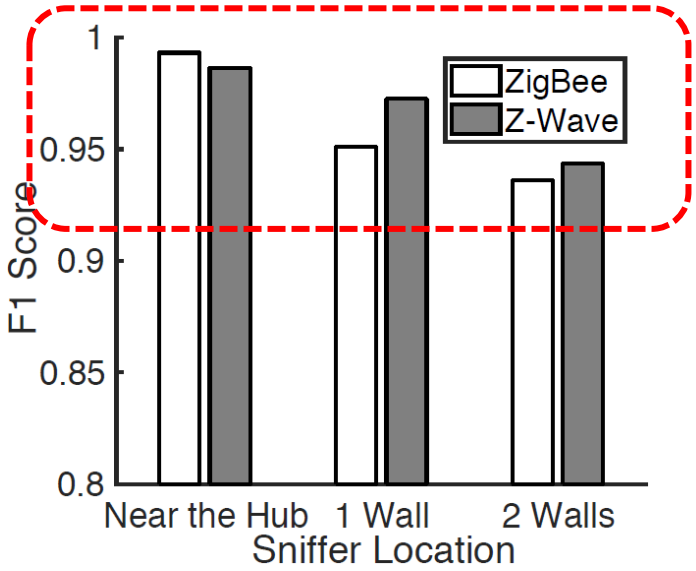
Micro-benchmark

Impact of Distance and Wireless Obstacles

The effectiveness of app inference may be affected by the environment condition (sniffer distance)

Experiment

1. 6 feet without walls
2. 16 feet with 1 wall
3. 33 feet with 2 walls



Recall : all above 0.88
Precision : all 1.00
F1 score : all above 0.94

Macro-benchmark

Reconsider the two types of attacks…

Platform (SmartThings) – Misbehavior types

- **Over-privileged accesses**
  - Illegally obtaining accesses to command / all the capabilities of the devices
- **Event spoofing**
  - A malicious SmartApp with the knowledge of the hub and device identifiers can spoof an arbitrary event

Earlence Fernandes, Jaeyeon Jung, and Atul Prakash. 2016. Security analysis of emerging smart home applications. In IEEE Symposium on Security and Privacy (S&P).

Misbehavior Detection

Over-privileged accesses

Strategy

HoMonit detects this misbehavior by checking additional states in DFA matching process

Experiment

30 benign & misbehaving apps

3 volunteers to simulate residents

20 times triggering for each app

Over-privileged access realization

We develop over-privileged apps by adding malicious code to cause unintended operations

SmartApp Example

Brighten My Path

Turn the light on when motion detected

Only requires command *on()* of *switch*

1. Illegally obtaining accesses to *off()*
2. Illegally gaining accesses to *all capabilities of the device*

Misbehavior Detection

Event spoofing

Strategy

HoMonit detects this misbehavior by checking missing states in DFA matching process

Experiment

Same approach as over-privilege access evaluation

Event spoofing realization

We develop event-spoofing apps by exploiting insufficient event protection mechanism to spoof a physical event and trigger subscribed apps

SmartApp Example

Flood Alert

Trigger a siren alarm for a wet state

Each device is assign with a 128-bit identifier

A fake wet event can be raised by exploiting the device identifier without permission requirement

Misbehavior Detection Result

Metrics

TP (true positive) : Correctly labeled misbehaving SmartApp

TN (true negative) : Correctly labeled benign SmartApp

FP (false positive) : Incorrectly labeled benign SmartApp

FN (false negative) : Incorrectly labeled misbehaving SmartApp

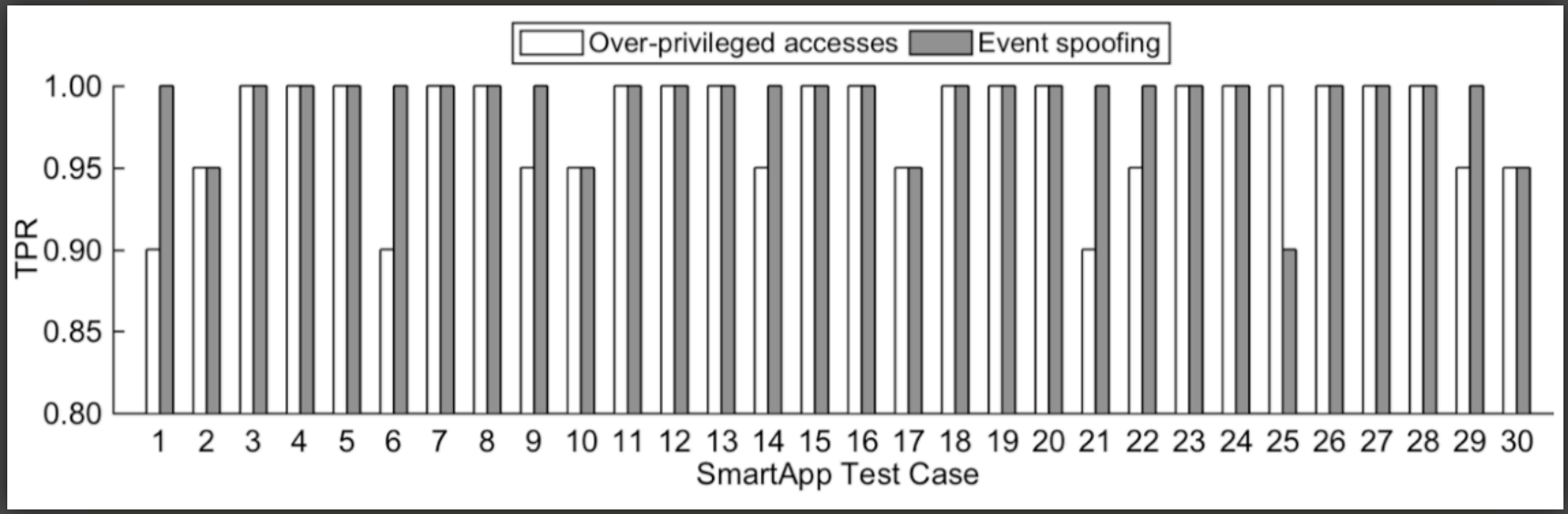
TPR (true positive rate) : TPR = TP / (TP + FN)

TNR (true negative rate) : TNR = TN / (FP + TN)

Misbehavior Detection Result

Metrics

Detection result of 30 misbehaving SmartApps



TP (true positive) : Correctly labeled misbehaving SmartApp

TNR (true negative rate) : TNR = TN / (FP + TN)

Misbehavior Detection Result

Metrics

Detection result of misbehavior occurrence in SmartApps

	ZigBee (20 misbehaving + 20 benign)		Z-Wave (10 misbehaving + 10 benign)	
	Over-privileged accesses	Event spoofing	Over-privileged accesses	Event spoofing
TPR	0.98 (0.03)	0.99 (0.02)	0.98 (0.04)	0.99 (0.04)
TNR	0.95 (0.07)	0.95 (0.06)	0.92 (0.05)	0.92 (0.05)

The average TPR for *over-privileged accesses* is 0.98

The average TPR for *event spoofing* is 0.99

Failed test case reason : packet loss, unexpected wireless traffic, accidental signal reception delay

Outline

- Motivation
- Design Overview
- DFA Building via SmartApp Analysis
- Misbehavior Detection via Wireless Fingerprint
- Evaluation
- Discussion
- Conclusion

Privacy Consideration

Side-channel information leakage is a double-edged sword

Daily routines : The attacker can spy on the victim's daily activities

SmartApp Example

Good Night : change its mode when there is no human activity in the home

Home occupations : The attacker can infer it by detecting the app existence

SmartApp Example

Vacation Lighting Director : deceptively control lights while residents are away

Health conditions : The attacker can infer the health condition of residents

SmartApp Example

Elder Care: Slip & Fall : monitor the behavior of the aged people

Outline

- Motivation
- Design Overview
- DFA Building via SmartApp Analysis
- Misbehavior Detection via Wireless Fingerprint
- Evaluation
- Discussion
- Conclusion

Conclusion

HoMonit	An anomaly detection system for smart home platforms to detect misbehaving SmartApps
Insight	Leverage the side-channel information leakage in the wireless channel to infer device event
Contribution	Program logic extraction from source code or UI, and auto DFA construction & matching algorithms
Performance	*HoMonit* can effectively validate the working logic of SmartApps and achieve a high accuracy in the detection of SmartApp misbehaviors

Thanks!

zhang-wei@sjtu.edu.cn