Bark: *Default-Off* Networking and Access Control for the IoT

James Hong, Amit Levy, Laurynas Riliskis, Philip Levis
Stanford University

The IoT is everywhere

So are the attacks...

1. Devices easily compromised

   Mirai botnet

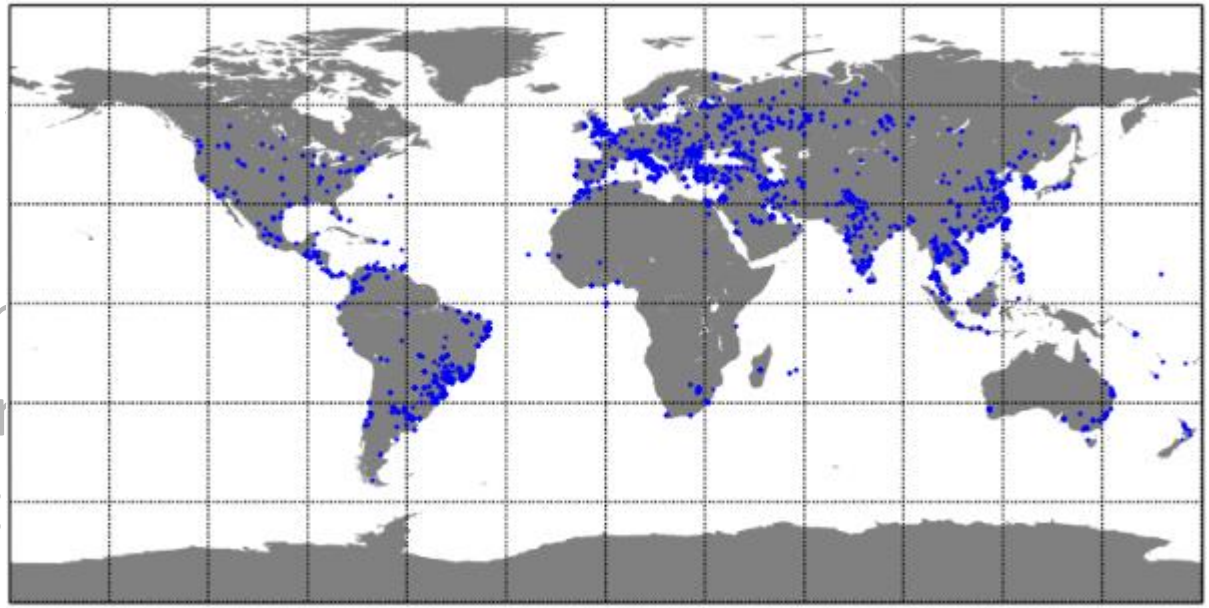
   - Targets IP cameras, DVRs, routers, printers
   - 100,000 IPs and 1.2Tb/s
   - Traffic to port 53 (DNS)

So are the attacks...

1. Devices easil

Mirai botnet

- Targets IP cam
- 100,000 IPs ar
- Traffic to port



Hajime worm

- Targets devices with open telnet port
- Again, default usernames and passwords

Image: https://www.symantec.com/connect/blogs/hajime-worm-battles-mirai-control-internet-things

So are the attacks…

2. Bugs in software

   Dishwasher directory traversal

GET /../../../../../etc/shadow HTTP/1.0

So are the attacks...

3. Ignoring best-security practices
   - No authentication
   - Sending data in the clear
   - Unsecured ad-hoc network

Can this be fixed?

1. Bugs and vulnerabilities can be fixed

   Difficult to patch a large number of devices

Can this be fixed?

1. Bugs and vulnerabilities can be fixed

   Difficult to patch a large number of devices

2. Devices are seamless and unobtrusive

   How would you know your thermostat is hacked?

Can this be fixed?

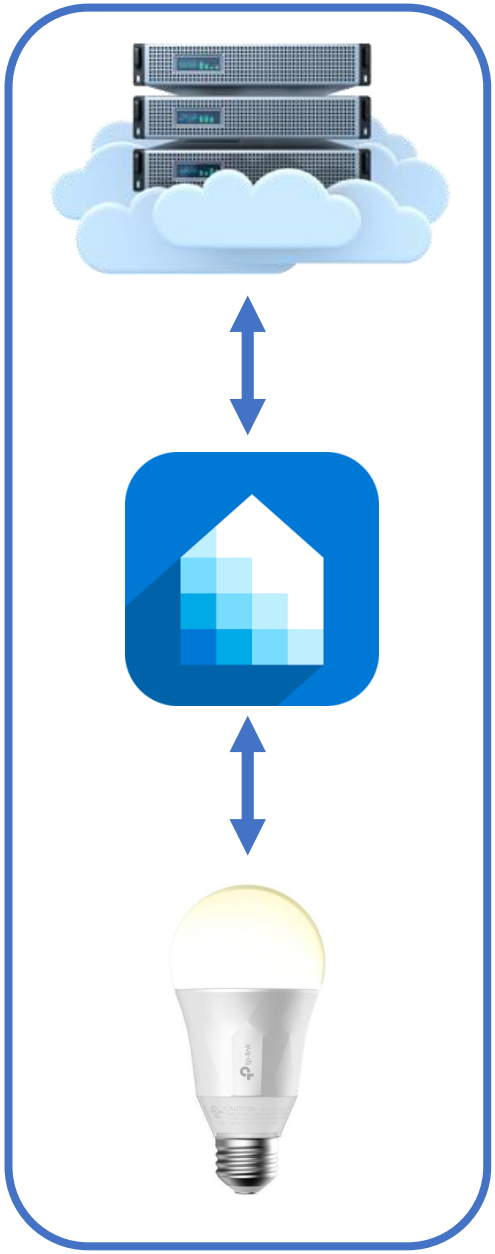
1. Bugs and vulnerabilities can be fixed

   Difficult to patch a large number of devices

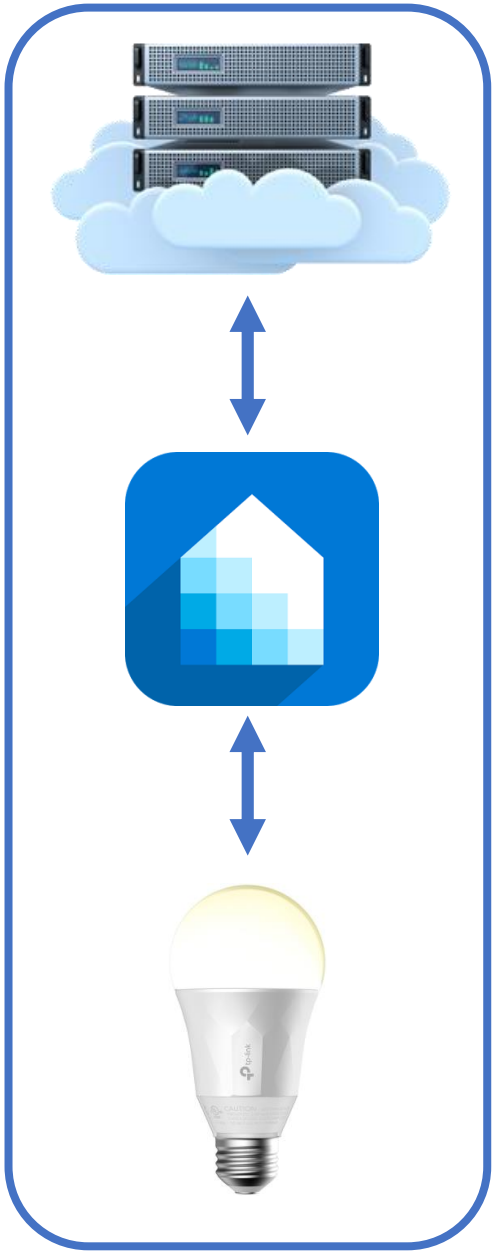
2. Devices are seamless and unobtrusive

   How would you know your thermostat is hacked?

3. Impossible to predict access control needs

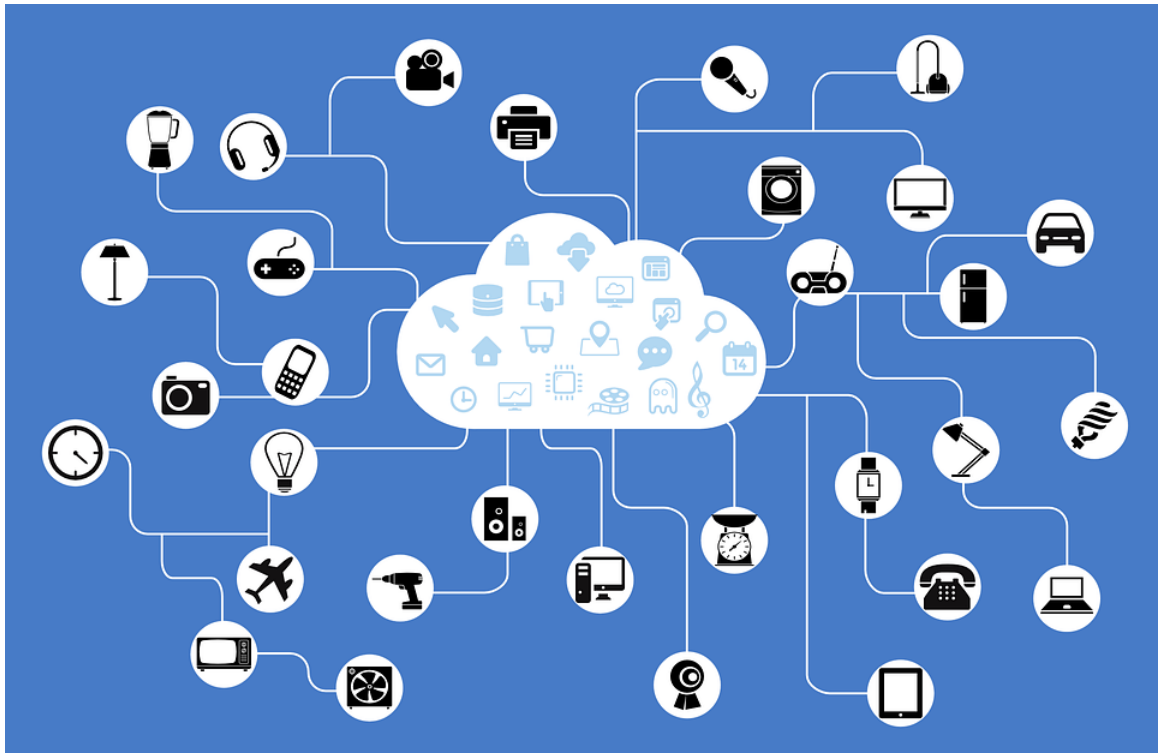


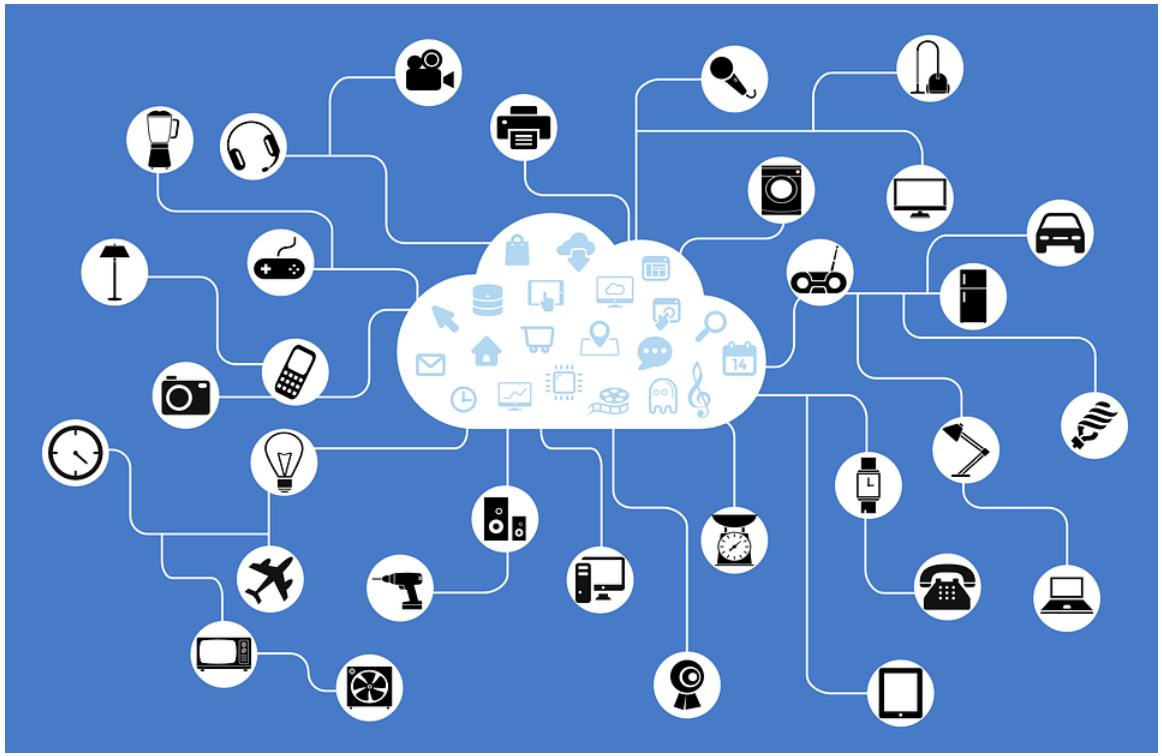
Application Silo



Application Silo

Users, sharing, policies





I don’t know

What is a user?
How is access shared?

What is **Bark**?

1. “*Default-off*”

What is **Bark**?

1. *"Default-off"*

   - IoT devices serve narrow functions and have narrow traffic patterns
   - Disallow everything else

What is **Bark**?

1. *"Default-off"*

   - IoT devices serve narrow functions and have narrow traffic patterns
   - Disallow everything else

   - Devices cannot DoS DNS servers, send spam
   - Random clients cannot telnet, ssh
   - *Hacked cloud cannot send commands when a user is not home*

What is **Bark**?

1. *"Default-off"*
2. Policy language to enable *"on"*

What is **Bark**?

1. *"Default-off"*
2. Policy language to enable *"on"*
   - Expressive *to capture a wide range of applications*
   - Precise *at the granularity of the application layer protocol*
   - Presentable and understandable to humans

What is **Bark**?

1. *"Default-off"*
2. Policy language to enable *"on"*

---

3. Enforceable by gateways
   - Require no changes to devices
   - Exploit existing protocols
     (HTTP, TCP, UDP, DNS, BLE/GATT, etc.)

What is **Bark**?

1. *"Default-off"*
2. Policy language to enable *"on"*

---

3. Enforceable by gateways
   - Require no changes to devices
   - Exploit existing protocols
     (HTTP, TCP, UDP, DNS, BLE/GATT, etc.)
   - *Best effort for each device. (TLS)*

Can we capture meaningful interactions in the network?

Can we capture meaningful interactions in the network?

- Devices have a small number of functions
  1. Gather and upload data (long flow)
  2. Perform actions (short bursts)
  3. Update firmware, etc.

Can we capture meaningful interactions in the network?

- Devices have a small number of functions
  1. Gather and upload data (long flow)
  2. Perform actions (short bursts)
  3. Update firmware, etc.
- Devices connect to a small number clients
  1. Cloud server
  2. Mobile phone
  3. Web browser

Can we capture meaningful interactions in the network?

- Devices have a small number of functions
  1. Gather and upload data (long flow)
  2. Perform actions (short bursts)
  3. Update firmware, etc.
- Devices connect to a small number clients
  1. Cloud server
  2. Mobile phone
  3. Web browser
- Differing security (and privacy) needs
  1. Smart lights vs. door lock
  2. Barometer vs. a smart treadmill

Can we capture meaningful interactions in the network?

- Devices have a small number of functions
  1. Gather and upload data (long flow)
  2. Perform actions (short bursts)
  3. Update firmware, etc.
- Devices connect to a small number clients
  1. Cloud server
  2. Mobile phone
  3. Web browser
- Differing security (and privacy) needs
  1. Smart lights vs. door lock
  2. Barometer vs. a smart treadmill

Expressing Types

Who

Where

When

What

How

Expressing Types

Who Identity of the device, application, etc.
(e.g. BD_ADDR, MAC address, client certificate, etc.)

Where

When

What

How

Expressing Types

Who

Identity of the device, application, etc.
(e.g. BD_ADDR, MAC address, client certificate, etc.)

Where

Verifiable location in the network
(e.g. connected to home gateway, or mobile hotspot.)

When

What

How

Expressing Types

Who — Identity of the device, application, etc.
(e.g. BD_ADDR, MAC address, client certificate, etc.)

Where — Verifiable location in the network
(e.g. connected to home gateway, or mobile hotspot.)

When — Boolean conditions
(e.g., a time constraint, condition such as requiring 2FA)

What

How

Expressing Types

Who — Identity of the device, application, etc.
(e.g. BD_ADDR, MAC address, client certificate, etc.)

Where — Verifiable location in the network
(e.g. connected to home gateway, or mobile hotspot.)

When — Boolean conditions
(e.g., a time constraint, condition such as requiring 2FA)

What — A resource or service of a device named by the protocol
(e.g., a HTTP path, DNS on port 53, BLE heart rate service)

How

Expressing Types

Who	Identity of the device, application, etc. (e.g. BD_ADDR, MAC address, client certificate, etc.)
Where	Verifiable location in the network (e.g. connected to home gateway, or mobile hotspot.)
When	Boolean conditions (e.g., a time constraint, condition such as requiring 2FA)
What	A resource or service of a device named by the protocol (e.g., a HTTP path, DNS on port 53, BLE heart rate service)
How	An action dependent on the “what” (e.g., GET/POST for HTTP, read/write/notify for BLE)

Rules from Types

1. Subject (Who, Where)

   WeMo app on *my smartphone*

Rules from Types

1. Subject (Who, Where)

   WeMo app on my smartphone

2. Object (Who, Where, What)

   WeMo switch connected to home AP, on/off

Rules from Types

1. Subject (Who, Where)

   WeMo app on my smartphone

2. Object (Who, Where, What)

   WeMo switch connected to home AP, on/off

3. Action (How)

   Allow modification

Rules from Types

1. Subject (Who, Where)

   WeMo app on my smartphone

2. Object (Who, Where, What)

   WeMo switch connected to home AP, on/off

3. Action (How)

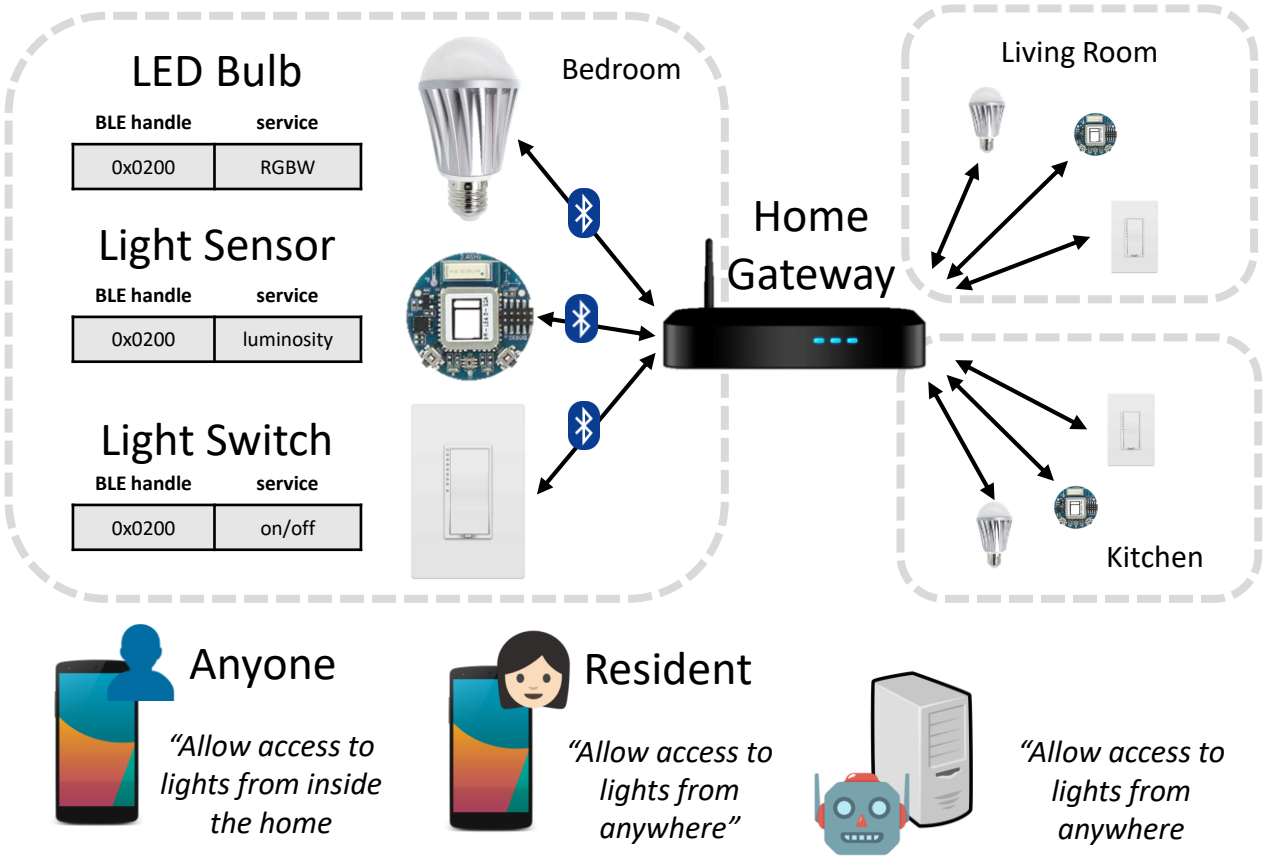
   Allow modification

4. Conditions (When)

   Anytime in the day

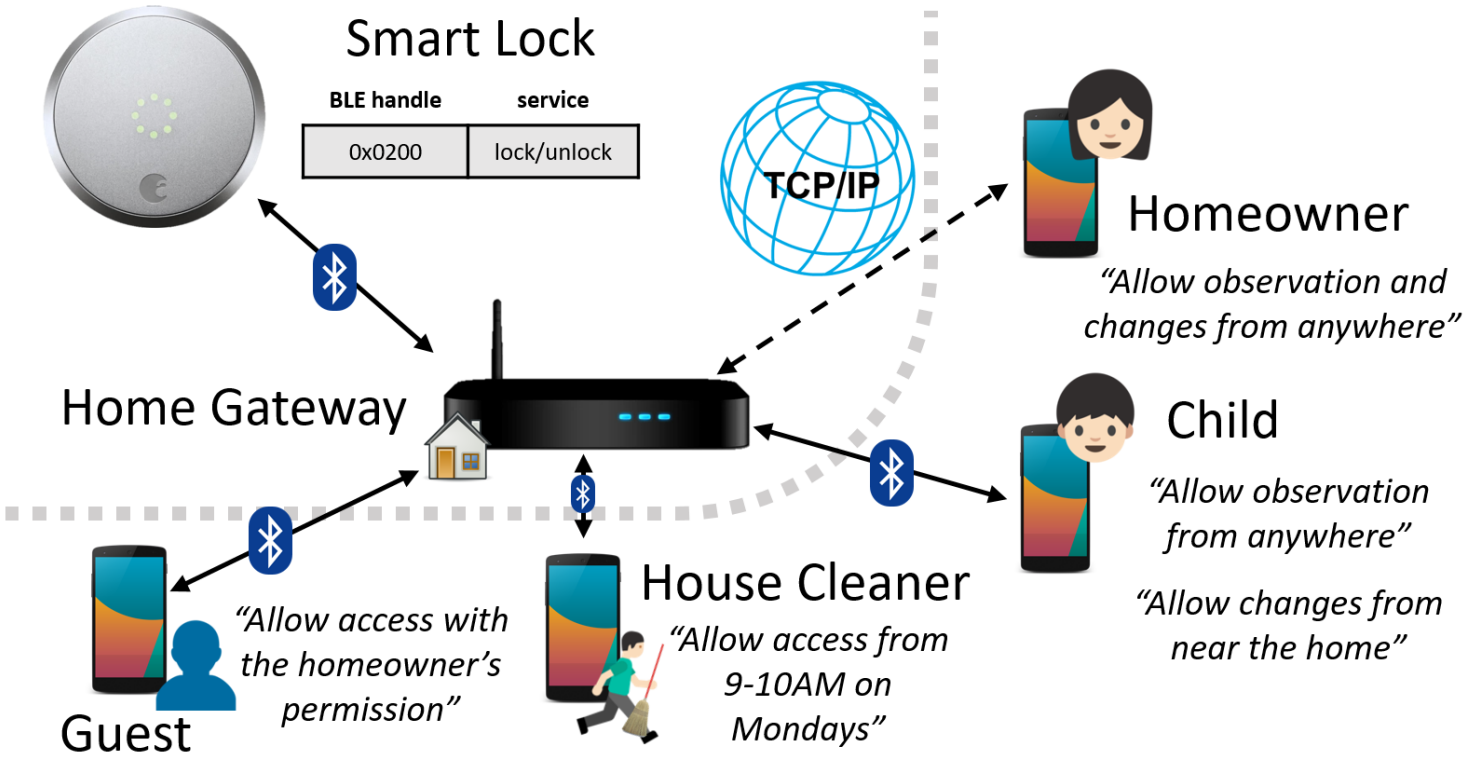

Subject{(WeMo app, James' phone)}
Action{GET/POST}
Who{WeMo app}
Where{James' phone}
How{GET/POST}
Allow WeMo app, at my phone, to modify on/off of WeMo switch, at home, at any time
What{TCP:49153:/upnp/control/basicevent1}
Who{WeMo switch}
Where{Home AP}
When{Cron(* * * * *)}
Object{(TCP:49153:/upnp/..., WeMo switch, Home AP)}
Conditions{Cron(* * * * *)}

This is sufficient for static topologies

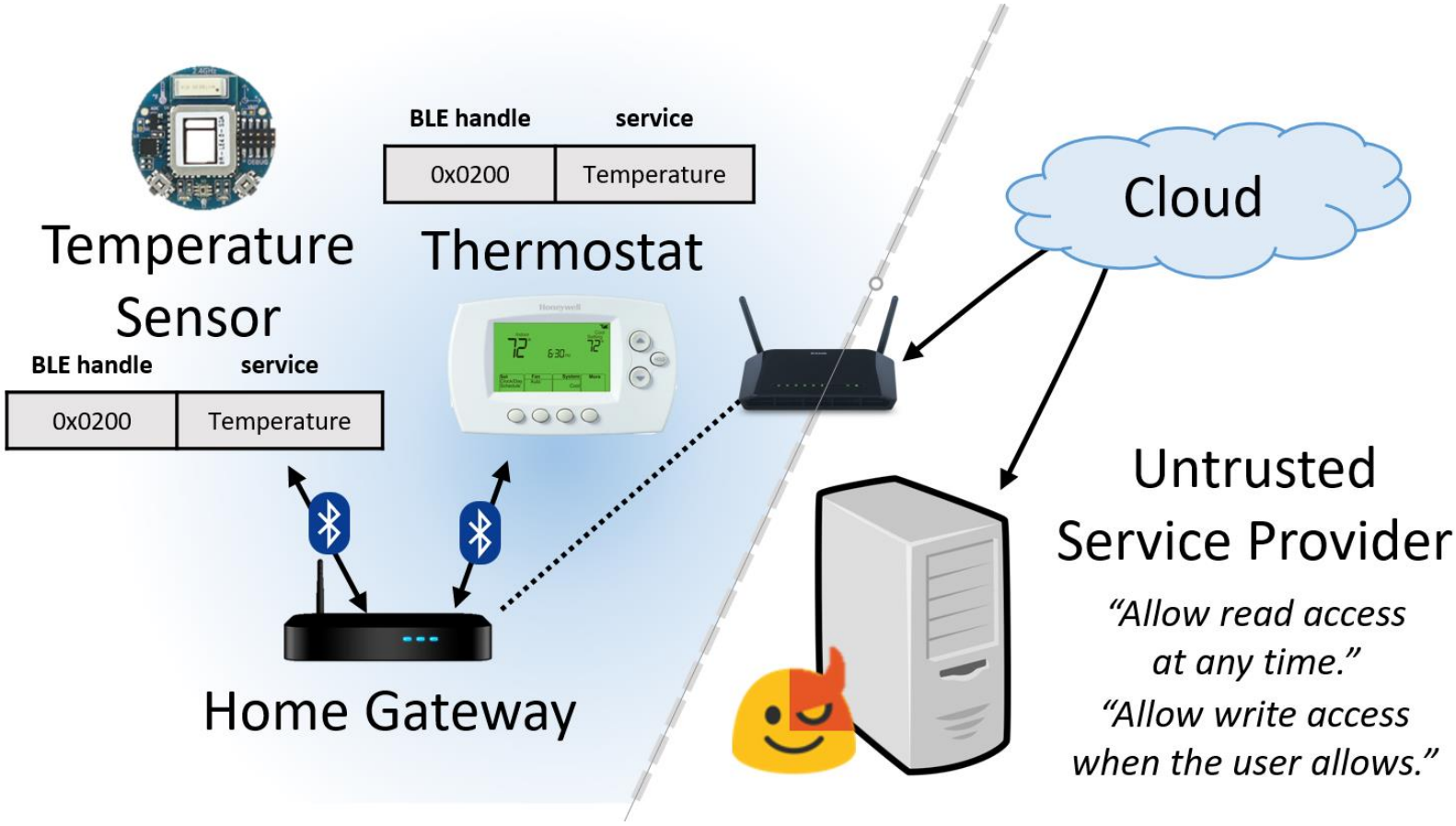


However, …

Sharing a Lock



Distrusting the Cloud



Adding *Conditions (when)*

- Devices do not live in a vacuum
- Sometimes it is ok to ask the user or owner

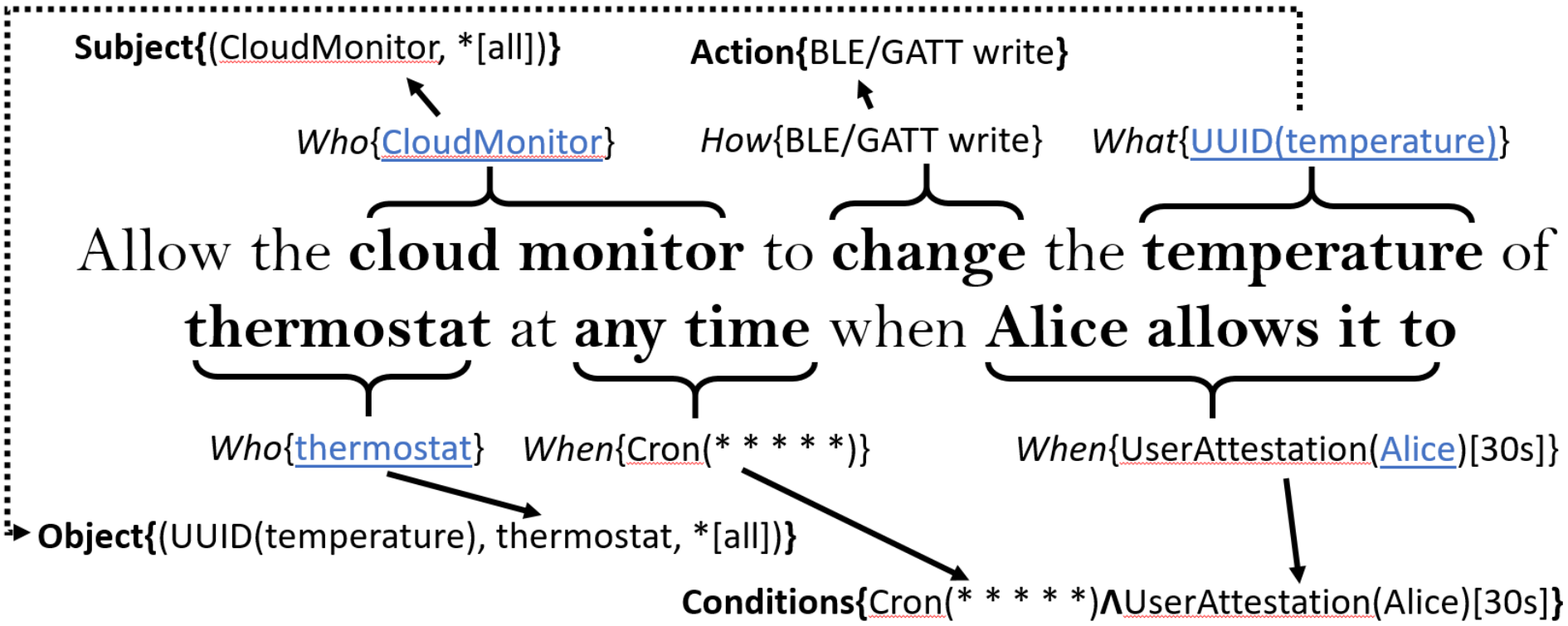
Adding *Conditions (when)*

- Devices do not live in a vacuum
- Sometimes it is ok to ask the user or owner

Four familiar schemes

1. Ask the user
2. Ask the owner
3. Authenticate with a password
4. Exclusive access

Example: dealing with a semi-trusted cloud

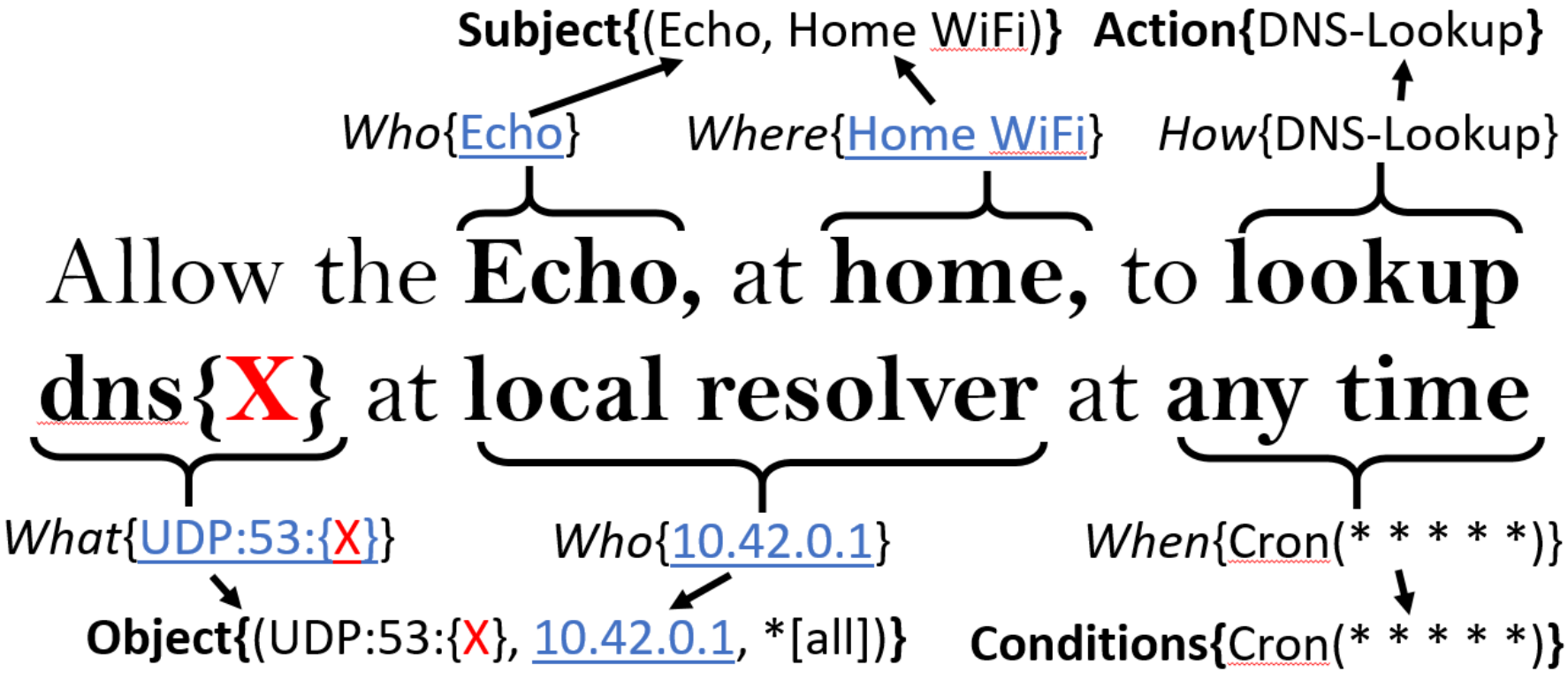


How well are existing IoT apps supported?

UPnP discovery protocol (SSDP)

How well are existing IoT apps supported?

Filtering DNS queries



Wildcards (*) and groups

- Not all of the devices may be known when a rule is specified
- Match patterns in HTTP paths or DNS queries

  (e.g., /event/, *.google.com)*
- Resolve overlaps with (logical-V)

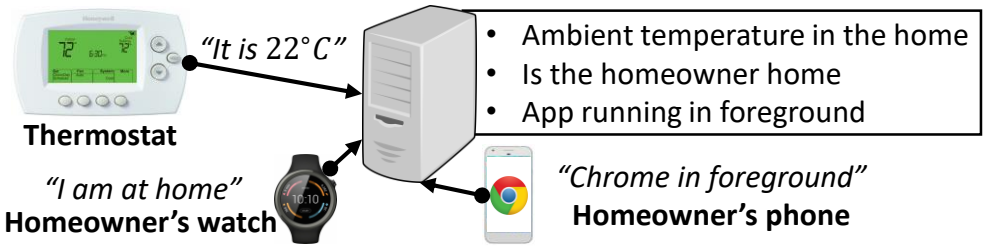
External oracles

1. Automate granting access
2. Express more complex conditions
   - *“only allow the Nest servers to make changes when the Nest app is in the foreground of your smartphone”*
   - *“allow your child to use certain appliances when some adult is present in the home”*

External oracles

1. Automate granting access
2. Express more complex conditions
   - ➢ *"only allow the X's servers to make changes when the X's app is in the foreground of your smartphone"*
   - ➢ *"allow your child to use certain appliances when some adult is present in the home"*

Implemented as server the gateway can query.



Limitations

1. Not always straightforward to write rules about network behavior of devices
   - *Nonstandard ports and protocols are common*

Limitations

1. Not always straightforward to write rules about network behavior of devices
   - *Nonstandard ports and protocols are common*
2. Limited by the level of granularity that we can inspect application traffic at (e.g., TLS)
   - *The finer the better*

Limitations

1. Not always straightforward to write rules about network behavior of devices
   - *Nonstandard ports and protocols are common*
2. Limited by the level of granularity that we can inspect application traffic at (e.g., TLS)
   - *The finer the better*
3. Tasks such as authenticating endpoints, setting up secure channels are still the application's and network's responsibility.

Implementation

1. WiFi access point
   - Uses iptables rules to yield certain decisions to a user application

Implementation

1. WiFi access point
   - Uses iptables rules to yield certain decisions to a user application
2. Bluetooth Low Energy (BLE) with **Beetle**
   - Virtualization system for BLE periperals
   - Paper at MobiSys ’16 (Levy, et al)

Code is public, along with Beetle
https://github.com/helena-project/beetle

Thanks and Questions?