Architectural Considerations in Smart Object Networking

IAB RFC 7452

Dave Thaler
Hannes Tschofenig
Mary Barnes (moderator)

Note: Slide contains embedded links and, depending how you view this slide deck, those may not work.

The PPT file can be found at: https://iab.org/wp-content/IAB-uploads/2015/03/92plenary.pptx

Some History Behind This Document

- A couple years ago, the IAB observed that:
  - Many non-IP-based smart object devices are being made and used
  - Various forums exist that defined profiles for non-IP-based devices
  - Belief among some of them that IP is too heavyweight

- RFC 6574 (Smart Object Workshop Report), April 2012 recommended IAB develop architectural guidelines about how to use existing protocols
  - It also pointed out some things for the IETF to address

- We wanted a document that explained to device engineers why/when IP should be used

- This RFC 7452 is the result
  - Thanks to various IETF folks who provided great feedback

Meanwhile, much work happened in parallel

- IETF WGs (6LO, 6TiSCH, ACE, CORE, DICE, LWIG, ROLL, etc.)
- IRTF proposed “Thing-to-Thing” RG
- RFC 7228 “Terminology for Constrained-Node Networks”
  - Three classes of constrained nodes, down to <<10KB memory/100KB code
- ZigBee Alliance created ZigBee IP that uses IPv6 and 6LoWPAN
- Bluetooth SIG and IETF worked on IPv6 over BTLE (Bluetooth Smart)
- IP-based alliances expanded (AllSeen, IPSO, OIC, OMA, Thread, etc.)

- And of course the hackers worked overtime too…

'Smart homes' are vu
hackers

By Heather Kelly, CNN
Updated 2:15 PM ET, Fri August 2, 2013

Hack
secur

Amor
powe
Hackin
contro

The bluetooth-cor t toilet was just one of the many conne security researchers hacked at the Bla n computer security conferences in La . They also opened front door locks, h tlets, took over the hubs that coordinate all the home-automation devices, and did some very creepy things with a toy bunny.

WIRED Rickroll Innocent Televisions With This

RICKROLL INNOCENT
TELEVISIONS WITH THIS
HACK

nsume

CNN M

The Stream

A used thermostat could hack your house

"Smart" online power meters are tracking energy use — and that data may soon be

This guy's light bulb performed a DoS attack on his entire smart house

What's so special about a "smart object"?

- There's many types of smart objects, so various answers might include:
  A. It's very constrained in some way (cost, power, memory, bandwidth, etc.)
  B. It interacts directly with physical world even when no user is around, and so potentially more dangerous
  C. It's physically accessible by untrusted people and so may be more vulnerable
  D. It's physically inaccessible by trusted people and has a long (5-40yr) lifespan

Smart Object Architecture

Information & Data Models

- Schema for exposing device-specific properties/methods/notifications/etc.

Software Stack

- Choice of protocols from app layer to link layer

IETF typically focuses just on this layer

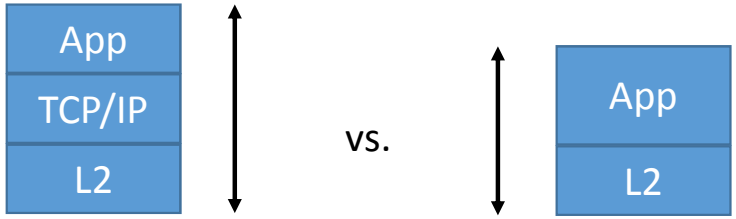
Hardware

- Choice of radio/other technology (Wi-Fi, Bluetooth, IEEE 802.15.4, …)

Internet-connected smart objects are even harder

- Besides all of the other issues, there's
  - Internet protocols to deal with
  - Corresponding attacks to deal with
  - More privacy issues to deal with (e.g., jurisdiction-specific legal requirements)

There's still tradeoffs of putting IP in smart objects



- If you DO put IP in a smart object:
  - You have to devote resources (code/memory/power) to it that might be desirable for other device functionality
  - You have to worry about securing IP from the Internet
- If you DON'T put IP in a smart object:
  - You usually need an Application-Layer Gateway (ALG) deployed
  - You might end up reinventing things IETF already did
  - You can't leverage the large ecosystem of IP-based knowledge, tools, etc.

Four Common Communication Patterns

1. Device-to-device within same network
2. Device-to-cloud
3. Device-to-ALG (to cloud or another local network)
4. Back-end data sharing

Device-to-Device Pattern

- Device talks directly to another local device (often smart phone or a wearable)
- Security & trust often based on direct relationship between the devices (pairing)
- Rarely uses IP today but apps instead directly sit over link layer protocol
  - Bluetooth, Z-Wave, ZigBee, …
  - Such forums often standardize device-specific data models
  - ***Results in many orgs doing somewhat redundant work, with differing information models for the same type of device***

Examples

Hearing Aid

Suunto Ambit 3

StickNFind

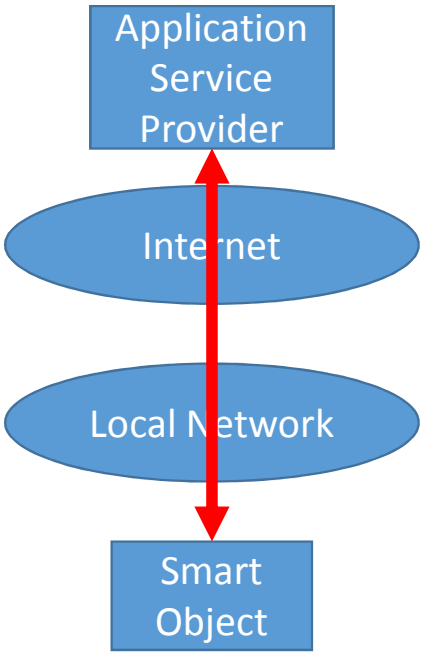
Beacons

Parrot

Cadence Sensor

Device-to-Cloud Pattern

- Device connects directly to some cloud service
- Allows users to access data/device from anywhere
- Requires choosing L2 already widely deployed, e.g. WiFi
  - Many different config. bootstrap solutions exist today
- Often service and device are from same vendor
- Can lead to silos with proprietary protocols
- Device might become unusable if ASP goes away or changes hosting provider
  - Standard protocols and/or open source can mitigate



Examples

LittlePrinter

Shut down this month

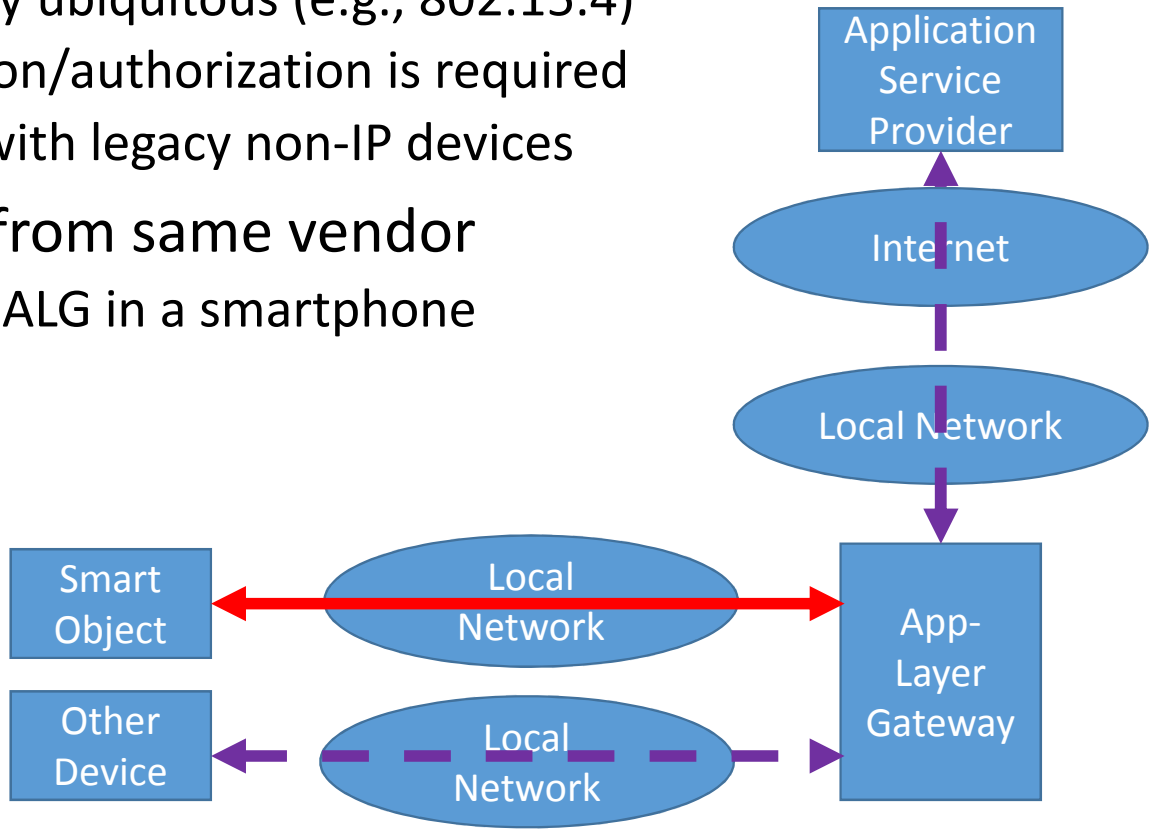
Withings Scale

Dropcam

Tractive

Device-to-ALG Pattern (1/2)

- Typically used in any of these cases:
  a) Uses L2 media not already ubiquitous (e.g., 802.15.4)
  b) Special local authentication/authorization is required
  c) Interoperability needed with legacy non-IP devices
- Often ALG and device are from same vendor
  - Another common model is ALG in a smartphone



Device-to-ALG Pattern (2/2)

- ALG also allows integrating IPv6-only devices and legacy IPv4-only devices/apps/cloud services
- *Cheaper and more reliable generic gateways more likely if devices use standard protocols not requiring an **app-layer** gateway*
  - Lack of standard data models for device types hampers this

Examples of ALGs

Philips Hue

NXP Janet-IP

SmartThings

Nest

Revolv Smart Home Gateway

Example devices with phone as ALG

Zepp Golf

Sensor

Oral-B Toothbrush

Fitbit

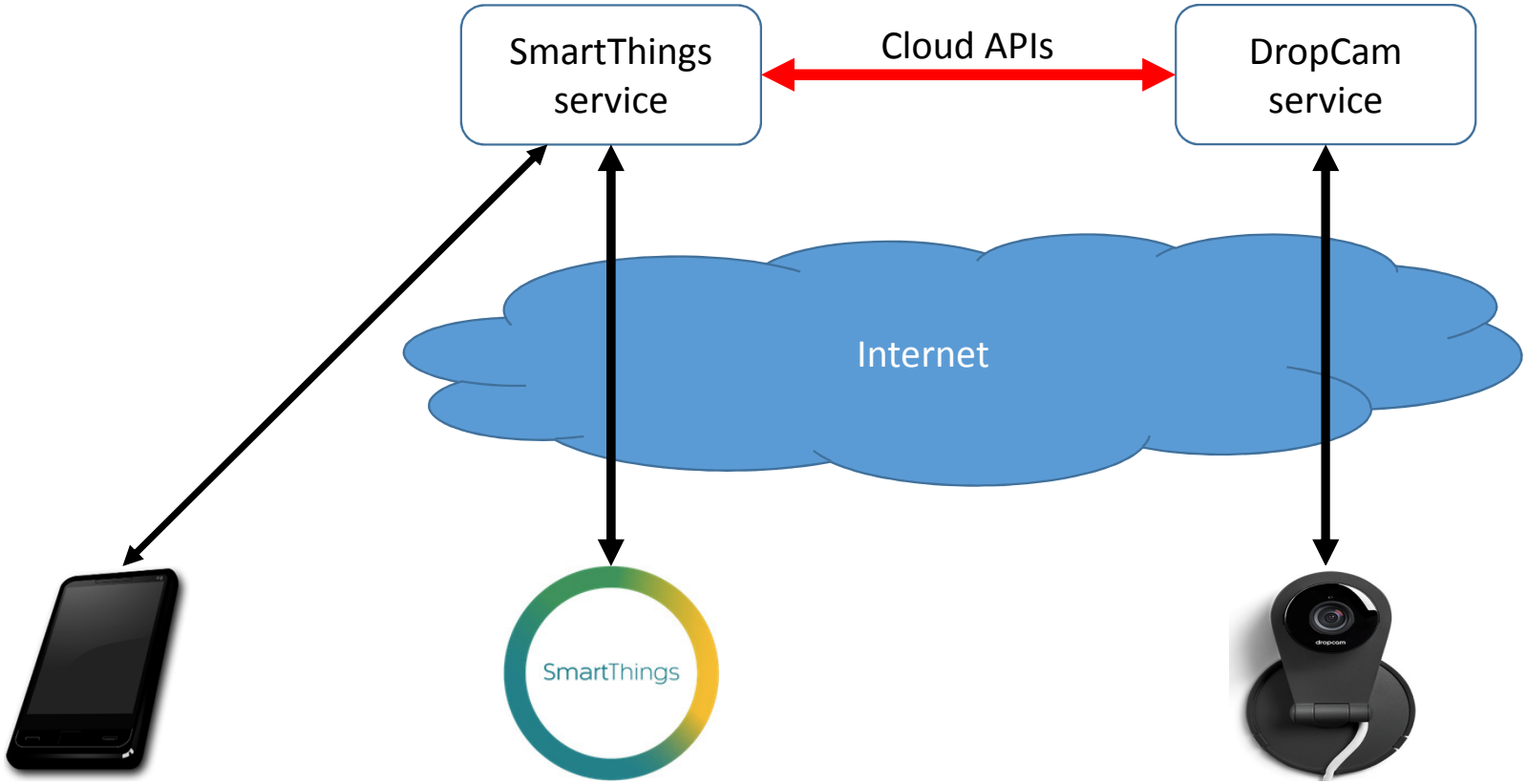
Garmin

Forerunner 920XT

Back-end Data Sharing Pattern

- Data silos result from proprietary schemas
  - Intentionally or simply due to lack of any standardization
- Many usage scenarios need data/devices from multiple sources
- Results in federated cloud services and/or (often RESTful) cloud APIs
- Standard protocols (HTTP, OAuth, etc.) help but are not sufficient
  - Standardized information models generally outside scope of IETF

Example


SmartThings service
Cloud APIs
DropCam service
Internet
SmartThings

Summary of Lack of Standardization

- Information/data models for various types of smart objects
  - Often outside scope of IETF, except for general connectivity models
  - There's lots of other forums in this space
    - "The nice thing about standards is that you have so many to choose from." –Tanenbaum
    - See also http://xkcd.com/927/
- App-layer mechanism to configure Wi-Fi (etc) settings
  - WiFi Alliance has WPS but not ubiquitously accepted
  - Using browser with web server in device avoids "need" to standardize
  - Still some desire for common mechanisms, but unclear where it best belongs
- Smart objects today often compete on time-to-market
  - Standardization seen as too slow

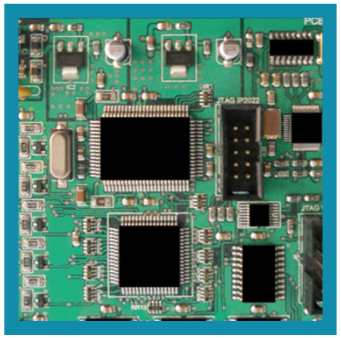
Effect on End-to-End

- IAB RFC 1958: “the goal is … intelligence is end to end rather than hidden in the network”
  - But the smallest of constrained devices need “proxies, gateways, or servers” for Internet communication
- IAB RFC 3724: “Requiring modification in the network … typically more difficult than modifying end nodes”
  - But can be expensive to put a secure software update mechanism in a smart object

Total Cost of Ownership

Total Cost

=



Hardware Cost

+

Energy Cost

+

Development Cost
(amortized, inc. deployment cost)

We care most about *this*.

But it can make sense to spend more *here* …
(e.g., on flash/RAM, CPU, BOM)

… if it results in savings *here* …
(e.g. sophisticated power management)

… and *here*.
(e.g. firmware update, manageability)

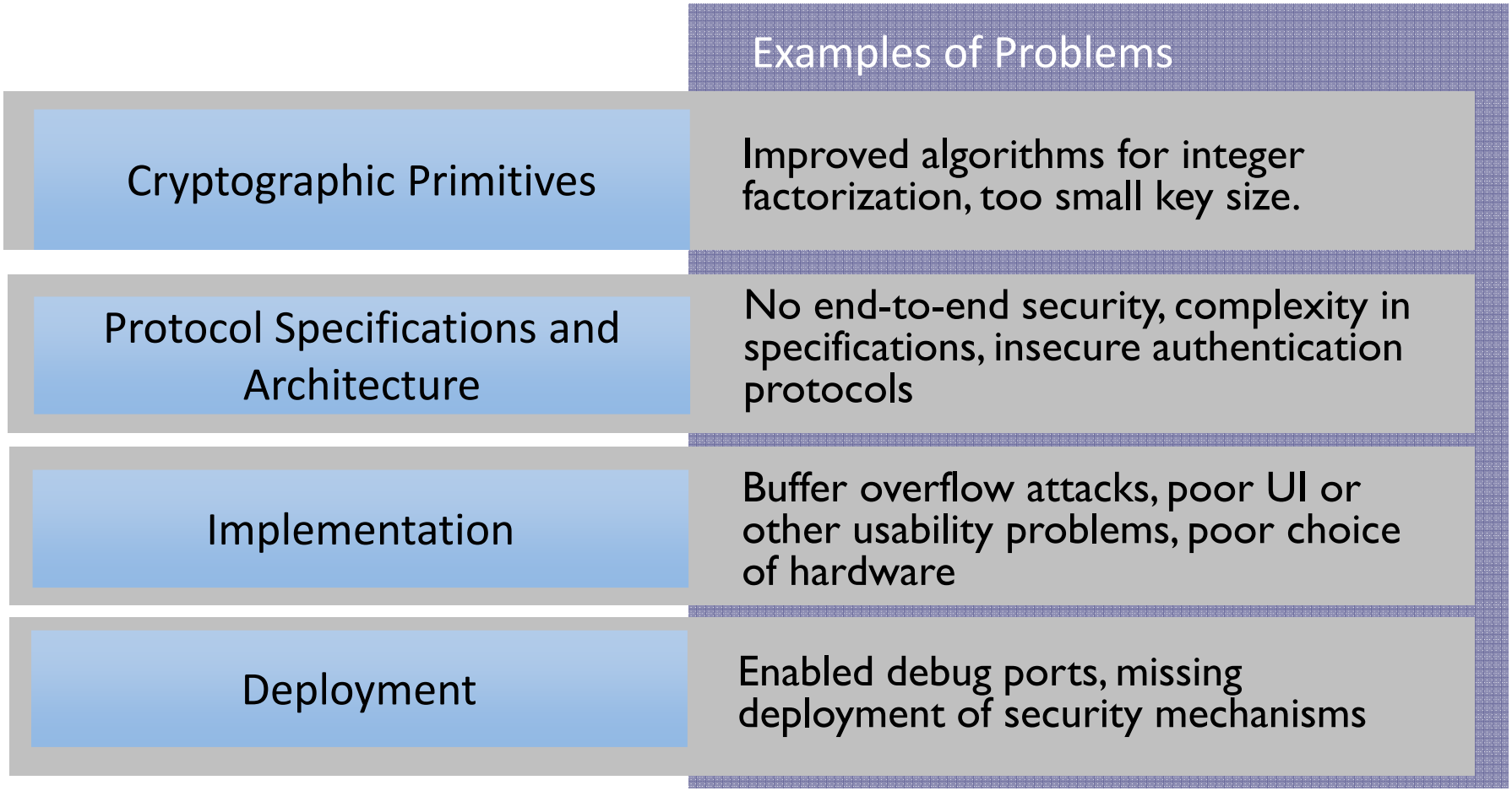
More detailed treatment of this topic in a webinar by Peter Aldworth about "How to Select Hardware for Volume IoT Deployments?"

Securing the Internet of Things

Which approach to take?

- Perform Classical Threat Analysis
- Following Security Recommendations
- Learn from Attacks
- Follow Design Patterns

Areas of Responsibility



Understanding the distributed nature of the development process is essential for tackling security problems.

Security Recommendations (IETF)

- **Key management:** RFC 4107 discusses the **trade-off between manual and automatic key management** and recommends the use of automatic key management.
- RFC 7258 argues that protocols should be designed such that they make **Pervasive monitoring** significantly more expensive or infeasible (such as by using opportunistic security - RFC 7435).
- draft-iab-crypto-alg-agility argues for the ability to migrate from one algorithm to another over time (called **Crypto Agility).**
- **Randomness requirements** and **key length recommendations** → subsequent slide
- Also available are protocol-specific recommendations
  - Using TLS in Applications (uta) working group
  - DTLS In Constrained Environments (dice) working group

Randomness Requirements

- RFC 4086 – “Randomness Requirements for Security”
- Security protocols frequently use random numbers for
  - Nonces for use with authentication and to avoid replay protection
  - Key transport
  - Asymmetric key generation (e.g., ephemeral Diffie-Hellman key pairs)
  - Signature algorithms based on El Gamal
- Unfortunately, most sources of randomness available at laptops and desktop PCs are not available at embedded systems.
  - Startup clock time in nanosecond resolution, input events, disk access timings, IRQ timings.
- The danger is that there is little (to no) randomness in embedded systems, as observed by Nadja Heninger et al. and Kenneth Paterson et al.

Key Length Requirements

- The chosen key length impacts security and performance.
- [I-D.ietf-uta-tls-bcp] recommends at least 112 bits symmetric keys.
- A 2013 ENISA report states that an 80bit symmetric key is sufficient for legacy applications but recommends 128 bits for new systems.
- ECC offers better performance than RSA for the same level of security taking over-the-wire bandwidth into account.
  - For this reason, there is a preference for use of ECC with IoT protocols.

Learn from Attacks

- Selected attacks to illustrate common problems:
  - Limited software update mechanism
  - Missing key management
  - Inappropriate access control
  - Missing communication security
  - Vulnerability to physical attacks
- Don’t forget to secure the server-side as well.
  According to the Open Web Application Security Project (OWASP) this is the #1 security vulnerability.

Limited Software Update Mechanism

- In January 2014 Bruce Schneier published an article where he expresses concerns about the lack of software update mechanisms in IoT deployments.


NEWS

Vulnerability in embedded Web server exposes millions of routers to hacking

By Lucian Constantin | Follow
IDG News Service | Dec 18, 2014 11:30 AM PT

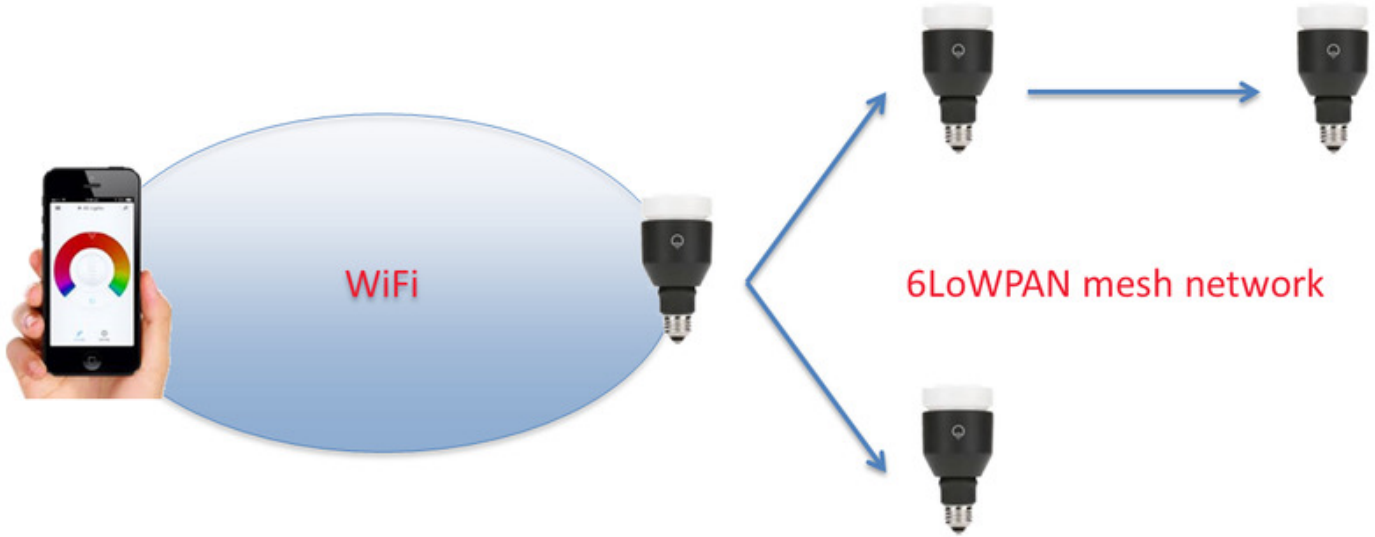
MORE LIKE THIS ::

Misfortune Cookie Vulnerability Affects 12 Million Routers

- In a presentation at the Chaos Communication Congress in December 2014 a security vulnerability of devices implementing the TR69 protocol, which also provides a software update mechanism, was disclosed.
- Real problem: Fix released in 2005 by AllegroSoft already but has not been distributed along the value chain of chip manufacturers, gateway manufacturers, Internet service providers.
- What happens when vendors do not support certain products anymore? Do IoT devices need a “time-to-die”/”shelf-life”?

Missing Key Management Problem

- Example: **LIFX** - Internet connected light bulb



- The attack revealed that an AES key shared among all devices to simplify key management.
- The firmware image was extracted via JTAG using a Bus Blaster. Then, the firmware was analyzed using IDA Pro.
- Mistakes only made by startups? See BMW ConnectedDrive

Pictures taken from http://contextis.co.uk/resources/blog/hacking-internet-connected-light-bulbs

Inappropriate Access Control

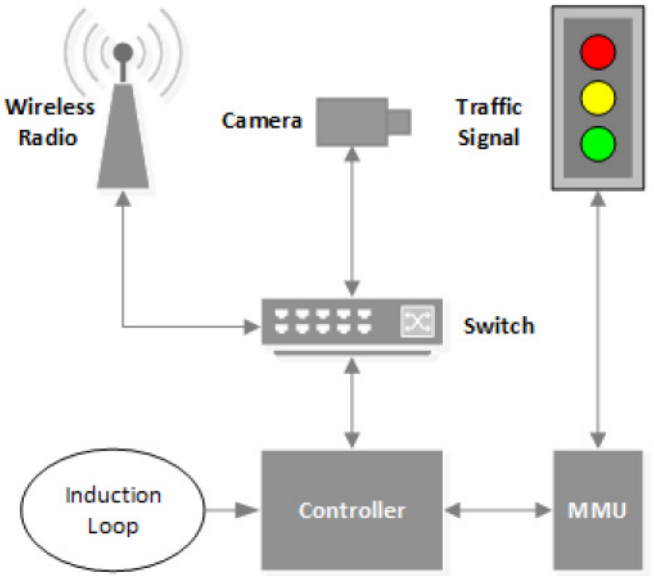
Insteon LED Bulbs

- Insecure default settings have caused problems with Insteon LED Bulbs, as reported in "When 'Smart Homes' Get Hacked: I Haunted A Complete Stranger's House Via The Internet"

- To find IoT devices connected to the Internet global scans have been used, for example, using ZMap.
- Similar problems have been seen with various other appliances, such as surveillance cameras, baby monitoring cameras and gas stations.
- Lacking access control to configuration files can cause problems for the entire system, as demonstrated with attacks against industrial control systems.

Missing Communication Security

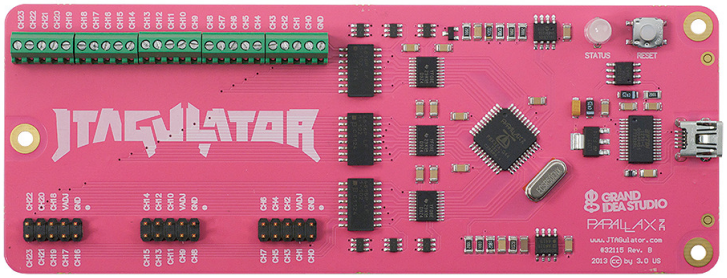
- In "Green Lights Forever: Analyzing the Security of Traffic Infrastructure" Ghena,et al. analyzed the security of the traffic infrastructure.
- Results:
  - "The wireless connections are **unencrypted** and the radios use **factory default usernames and passwords**."
  - "All of the settings on the controller may be configured via the physical interface on the controller, but they may also be modified though the network. An FTP connection to the device allows access to a writable configuration database. This requires a username and password, but they are fixed to **default values which are published online by the manufacturer**."



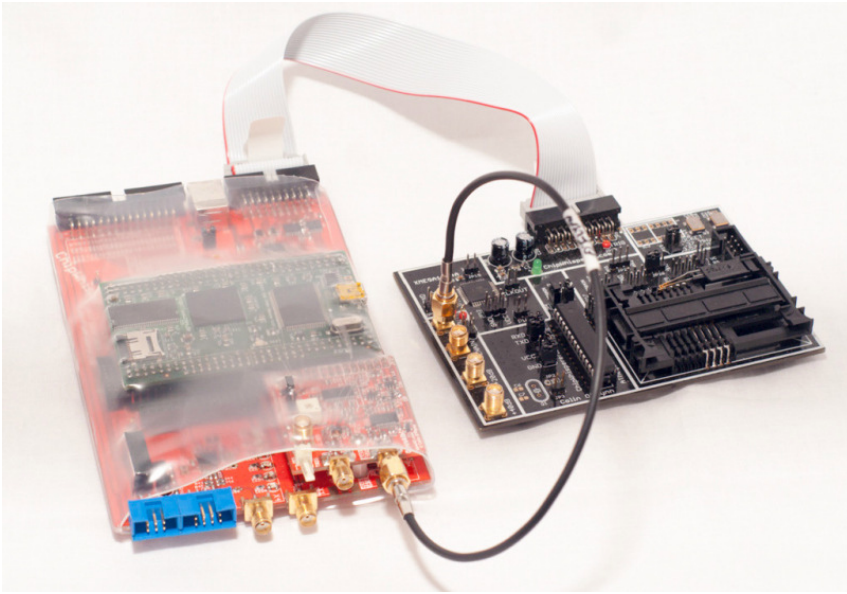
- A similar attack also exploited the unencrypted communication.
  - "I even tested the attack launched from a drone flying at over 650 feet, and it worked!"

Vulnerability to Physical Attacks

- Physical access to IoT devices introduces a wide range of additional attack possibilities.
- In some cases it might be necessary to extract keys contained on chip. This can be accomplished using power analysis, or fault injection (glitching) attacks.
- Tools for physical attacks decrease in cost and become easier to use.
- Important to keep these attacks in mind since we will see more of them in the future.



JTAGulator



Chip Whisperer

Remarks

- Internet of Things security today is like PC security 20 years ago.
- Most attacks on consumer-oriented IoT systems fall under the ”script kiddie” category.
  - For industrial control systems many attacks are already scary (see DragonFly, and attack against German steel factory).
- Risk analysis is often complex since hacked devices may be used for further attacks. Hence, indirect consequences also need to be taken into account.
  - Examples: DDoS attacks using SNMP (used in printers), hacked Femto home router used for spying

Privacy

- RFC 6973 provides generic guidance that is also applicable to IoT protocol engineering.
- Privacy challenges with the deployment of IoT technologies arise, such as
  - Quality of user consent, and
  - Consequences of big data processing and inferences derived from data (such as behavioral pattern)
- See also Article 29 Working Party publication: "Opinion 8/2014 on the Recent Developments on the Internet of Things" from September 2014.

Summary

- Re-use Internet security technologies:
  - Use **state-of-the-art key length**
  - Always **use well-analysed security protocols**.
  - Use **encryption** to improve resistance against pervasive monitoring.
  - Support **automatic key management** and per-device keys.
- Additional IoT relevant security aspects:
  - **Crypto agility** is a hard decision and you need to think deeply about it.
  - Integrate a **software update mechanism** and leave enough “head room”.
  - Include a **hardware-based random number generator**.
  - Threat analysis must take **physical attacks** into account.
  - **Use modern operating system concepts** to avoid system-wide compromise due to a single software bug.