EMBEDDED SYSTEM DESIGN AND THE INTERNET OF THINGS

PAT PANNUTO

UNIVERSITY OF MICHIGAN

INTERNET OF THINGS RESEARCH PROGRAM

AUGUST 11, STANFORD UNIVERSITY

Pat Pannuto

3rd Year Ph.D. Student, University of Michigan

- BSE Computer Engineering, University of Michigan

Research:

- Embedded systems, wireless technology, next-generation computing technologies
- “Last Inch” Problem

Prabal Dutta

Assistant Professor, University of Michigan

- Ph.D. in CS from Berkeley, 2009

Research:

- Networked embedded systems with applications to health, energy, and the environment

Regrets he cannot be here

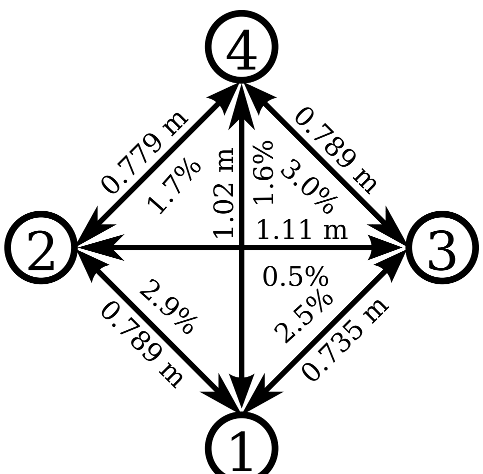
Hosting a DARPA ISAT workshop

Lab11 at Michigan: (Some of) What we do

eMbedded Gateway Cloud

Lab11 at Michigan: (Some of) What we do

eMbedded



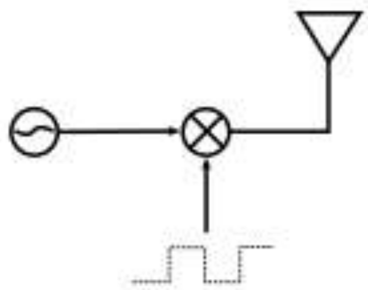
Opo

Goal:
Explore methods to enable high spatiotemporal human interaction tracking

Result:
Novel ultrasonic wakeup circuit enables ~2 s granularity with ~5 cm accuracy for 1 week on a 40 mAh battery

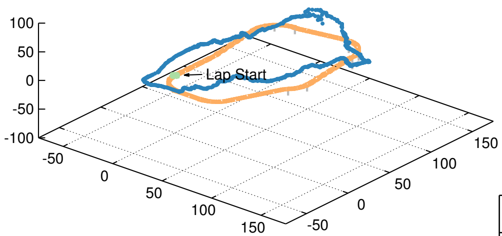
Lab11 at Michigan:
(Some of) What we do

eMbedded

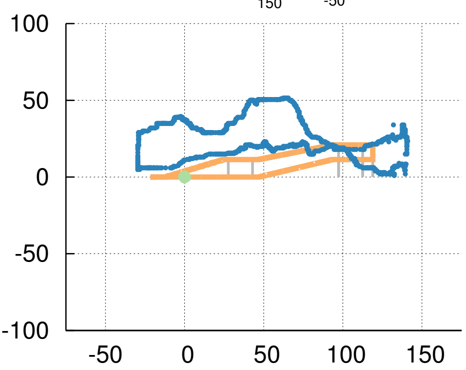


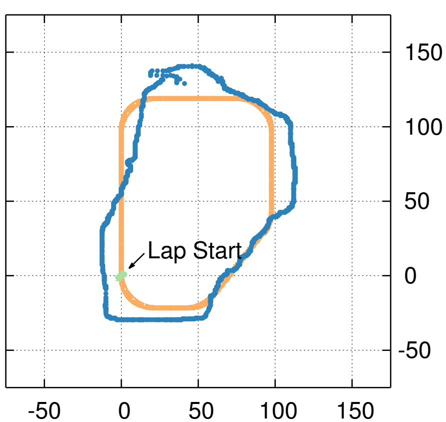
(a) Tag Architecture

(b) Tag Realization



20 cm avg error
40 cm 95%ile
56 Hz samples





Harmonia

Goal:

*Rapid, high accuracy, **indoor** RF TDoA localization*

Track micro-quadcopters in real time

Approach:

UWB accuracy using NB frontends via impulses (TX) and band-stitching (RX)

Lab11 at Michigan: (Some of) What we do

eMbedded

Monjolo Family

Original Hypothesis:
Estimate appliance energy use from side-channel emissions

Result:
Practical, battery-free energy-harvesting sensors*

Lab11 at Michigan: (Some of) What we do

eMbedded

Monjolo Family

Original Hypothesis:
Estimate appliance energy use from side-channel emissions

Result:
Practical, battery-free energy-harvesting sensors*

Lab11 at Michigan:
(Some of) What we do

"A necessary evil"

Gateway

Gen 1: Rpi + CC2520

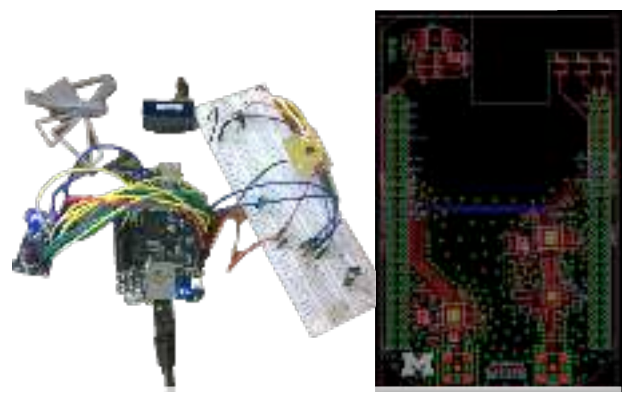
Lab11 at Michigan: (Some of) What we do

"A necessary evil"

That may be evolving into an interesting area of research

Gateway

Gen 1: Rpi + CC2520



Gen 2: BeagleBone Black + 2 x CC2520 + CC2591

Lab11 at Michigan: (Some of) What we do

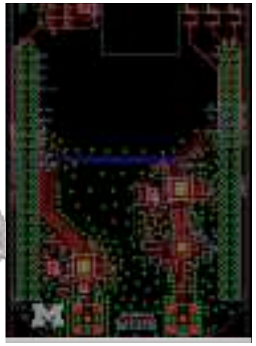
“A necessary evil”

That may be evolving into an interesting area of research

Gateway

Bluetooth Low Energy

Smartphone as a gateway



Gen 2: BeagleBone Black + 2 x CC2520 + CC2591

Lab11 at Michigan: (Some of) What we do

Never say "No"

- Collect and store all data, figure out what to do with it later

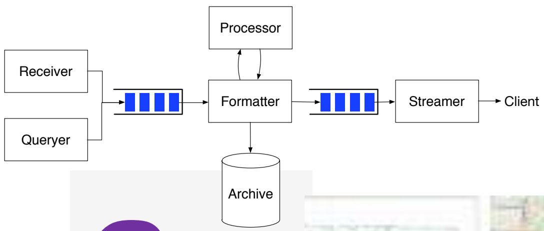
Optimize for real-time / streaming

- Easy to archive a stream to more traditional DB for analysis

Leverage "Web Scale"

- Enough technology exists to build highly scalable infrastructure quickly
- MongoDB + RabbitMQ + SocketIO

Cloud



Lab11 at Michigan: (Some of) What we do

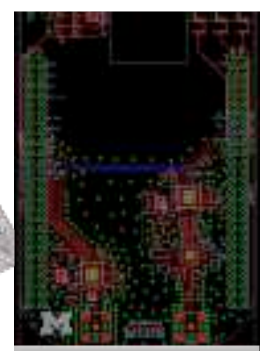
eMbedded

Embedded Systems
Built in the last 365 days

Gateway

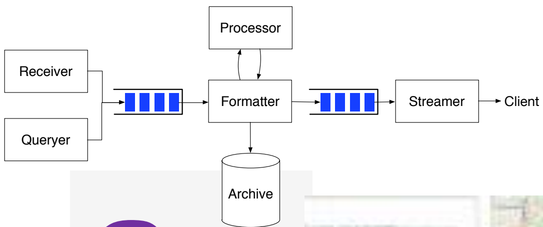
Bluetooth Low Energy

Smartphone as a gateway



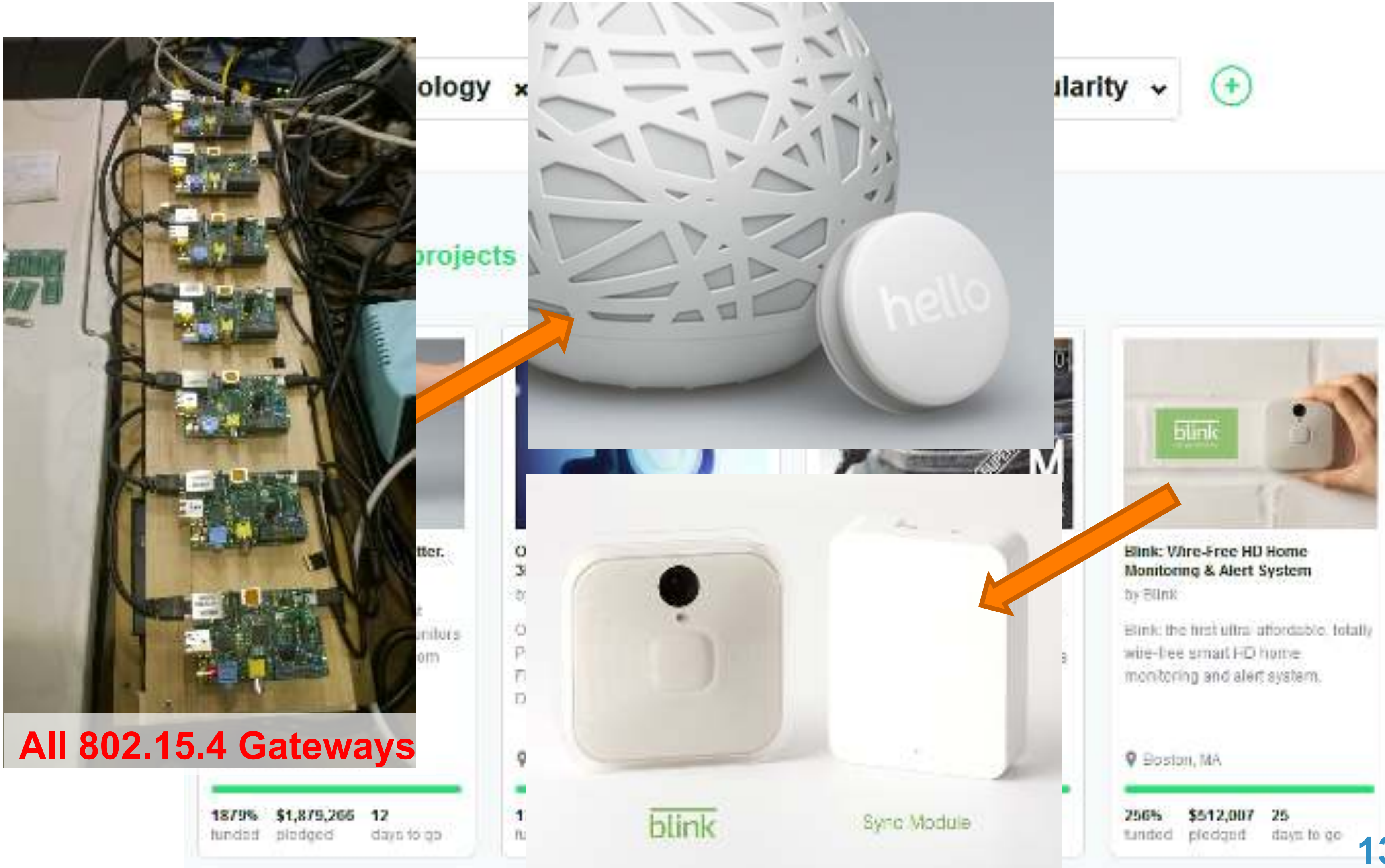
Gen 2: BeagleBone Black + 2 x CC2520 + CC2591

Cloud



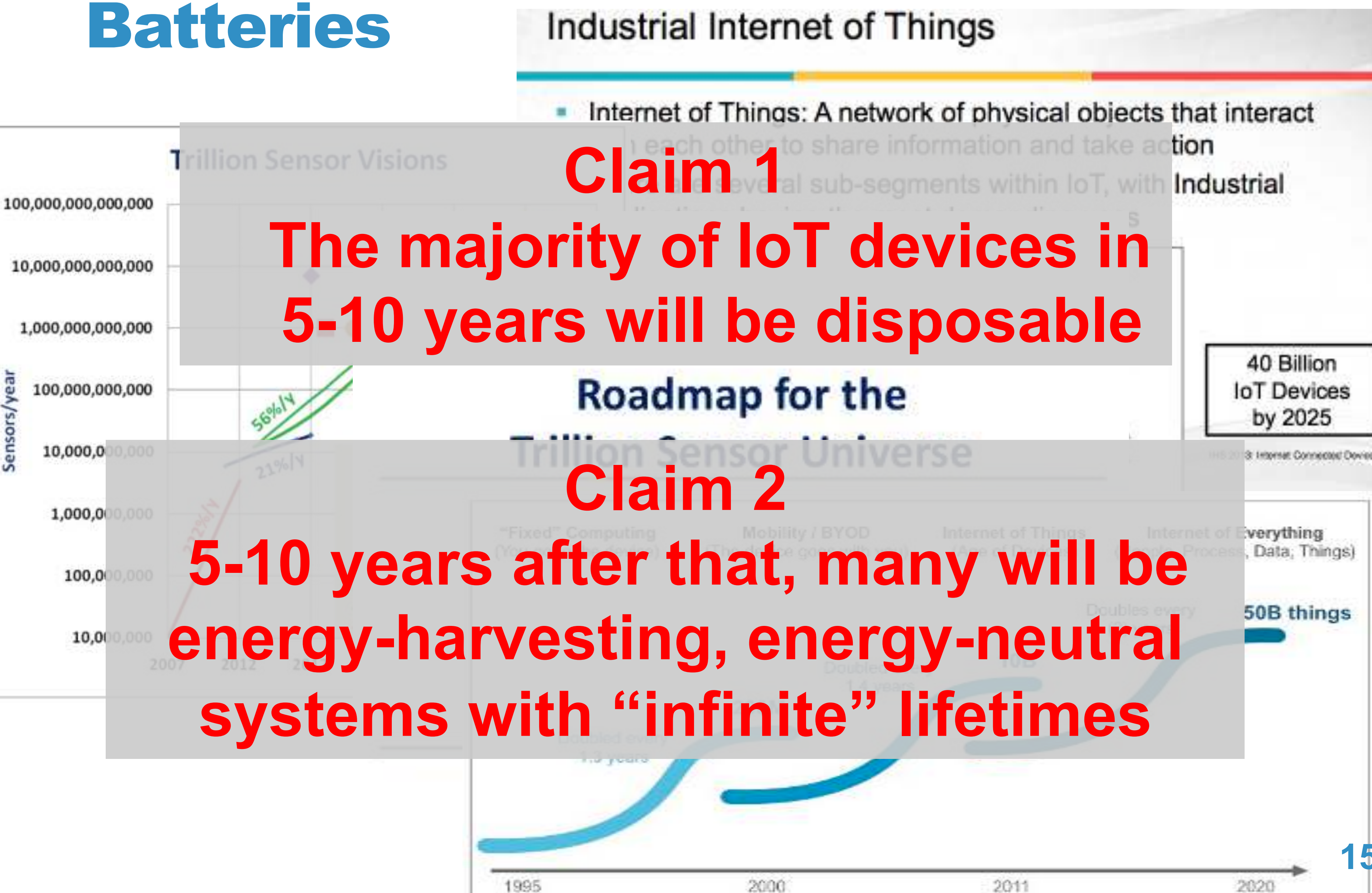
GET ALL THE DATA

The Gateway Problem



The (mostly) universal gateway worked for WiFi, why not us?

A Trillion Sensors is a Trillion Batteries



Claim 1

The majority of IoT devices in 5-10 years will be disposable

Claim 2

5-10 years after that, many will be energy-harvesting, energy-neutral systems with "infinite" lifetimes

The lifetime of a disposable IoT device is defined by the energy it ships with (or can harvest)

Thus, we need something more energy-efficient than 802.11

But what?

Self-Organizing (Sohrabi ’99), LEACH ’00, Adaptive Rate Control (Woo ‘01), S-MAC ‘02, WiseMAC ‘04, B-MAC ‘04, Adaptive LPL ‘07, RI-MAC ‘08, A-MAC ‘10, GLOSSY ‘11, LPB ‘12, Chaos ‘13, [To Appear: EkhoNet ’14]…

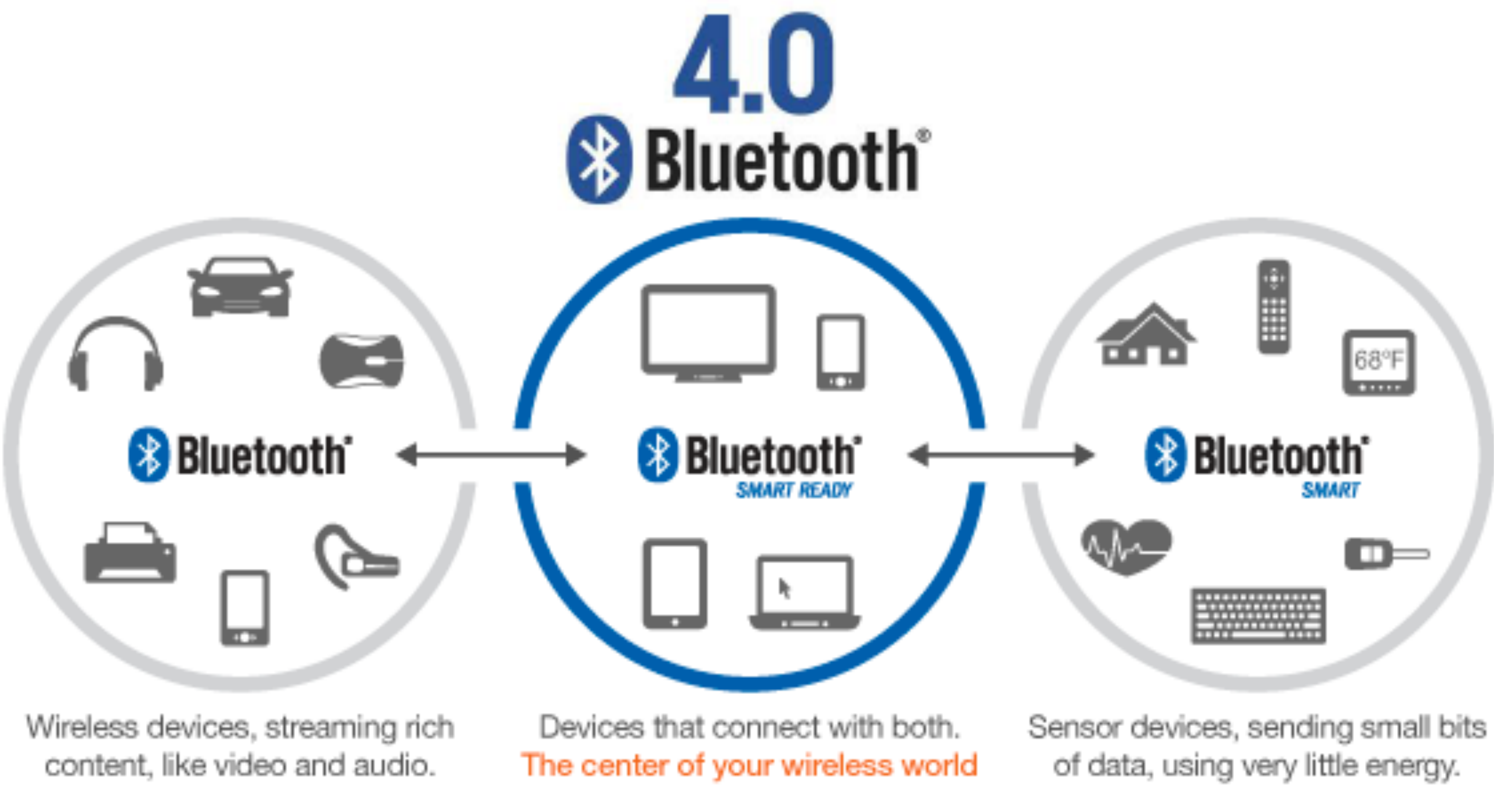
ZigBee, 802.15.4e, CTP

Best choice is system / application dependent

- Wakeup (“LPP”, Musaloiu-E. et al., IPSN’ 08)
- Discovery (“Disco”, Dutta et al., Sensys’ 08)
- Unicast (“RI-MAC”, Sun et al., Sensys’ 08)
- Broadcast (“ADB”, Sun et al., Sensys’ 09)
- Pollcast (“Pollcast”, Demirbas et al., INFOCOM’ 08)
- Anycast (“Backcast”, Dutta et al., HotNets’ 08)

Bluetooth Low Energy – A MAC convergence for non-mesh applications

BLE doesn't mesh, but many applications don't need mesh – especially [primarily] collection-based ones



BLE as a backhaul for Personal Area Networks

This is in commercial technology now

Apps are emerging to provide other services as well

Can this network reliably provide other, more demanding applications (e.g. firmware updates? + different class of trust for this application)

BLE as a backhaul for general sensor networks "Reverse Data Muling"

Can smartphones + BLE act as a semi-universal gateway?

Are there security concerns with auto-connecting to arbitrary Bluetooth devices?

How does an embedded device trust arbitrary phones?

Who pays for the data? Can this run as a carrier service?

[Micropayments?]

The gateway problem is a fundamental problem

Low-power IoT devices require low-power networks

- Which by their nature have limited range

Something (gateway) must bridge a low-power network

- It is part of the architecture for good reason
- But it is burdensome in practice to deploy

One Potential Plus

- Gateway as a privacy-preserving bottleneck

The Gateway Problem
~~Nest~~ Google Gets It

Wireless

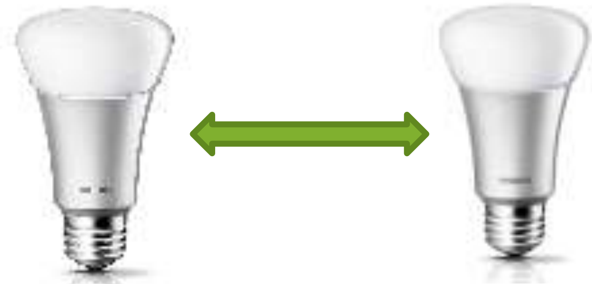
Working Wi-Fi connection: 802.11 - 2.4 GHz
Wireless Interconnect: 802.15.4 - 2.4 GHz

Nest Products are Trojan Horse IoT Gateways

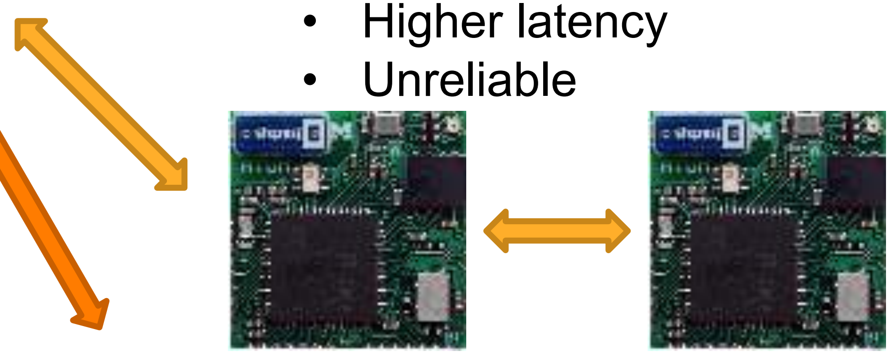
Motivation for burdening the gateway node: Masking less-performant, less-reliable low-power networks

The Internet
- High bandwidth
- Reasonably low latency
- Reliable

Low Power Networking Powered Devices
- Less bandwidth



Low Power Networking Battery-backed
- Less bandwidth
- Higher latency
- Unreliable



Energy-Harvesting
- Short Transmissions
  - Minimal bandwidth
- Non-deterministic latency
- Highly Unreliable
- Possibly Unidirectional

Motivation for burdening the gateway node: Centralizing computation to minimize costs

Minimize Cost

- Limited computing power
- Limited program space, memory

A Case Study:
The risks of relying on the gateway to be anything more than a gateway

Hue authentication: App to base station

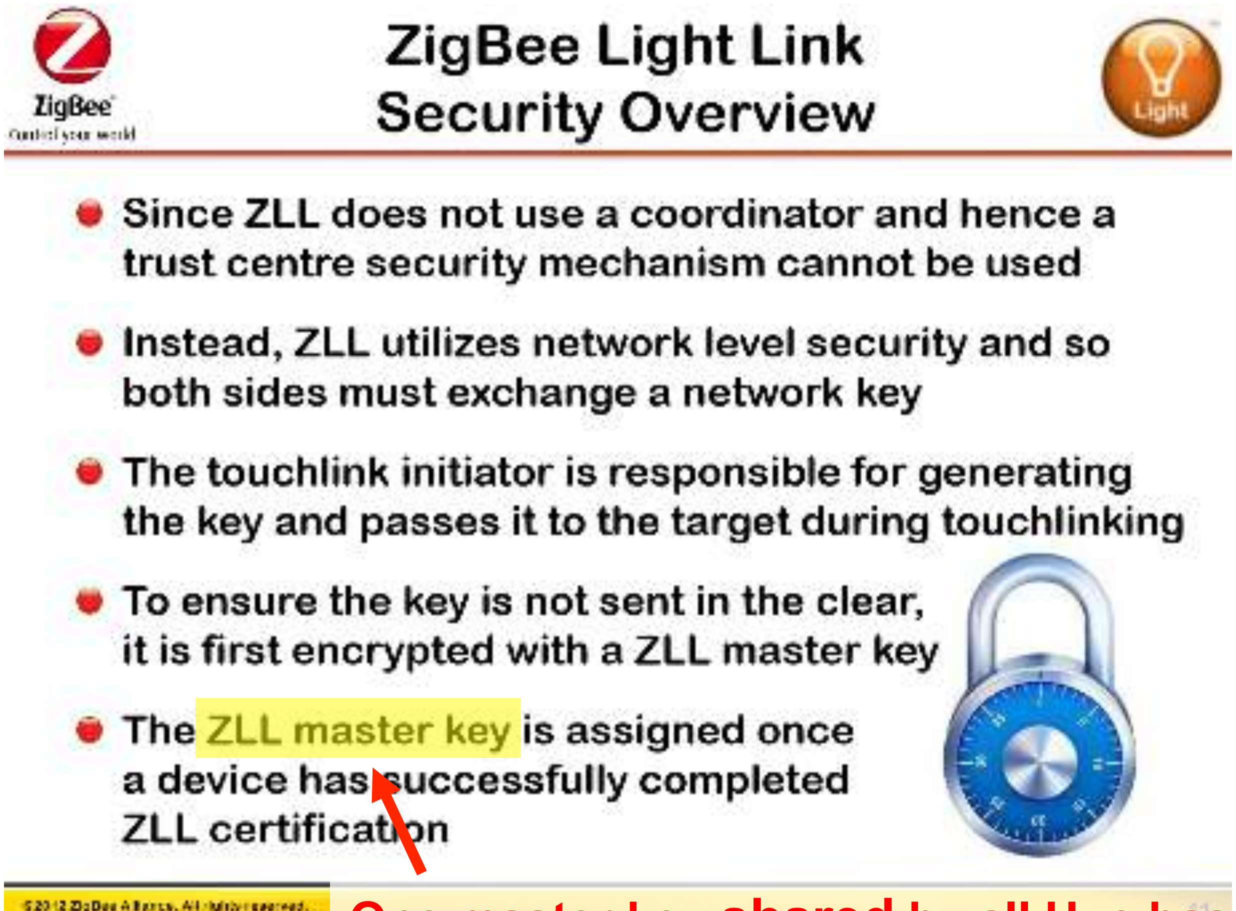
Hue authentication: Base station to bulb

Bulbs and base station ship pre-configured with shared secrets

Hue Security, OR
end-to-end violation by example

Does this architecture guarantee that all commands are sent by an authorized owner of the light?

Any Hue bulb will trust any Hue base station



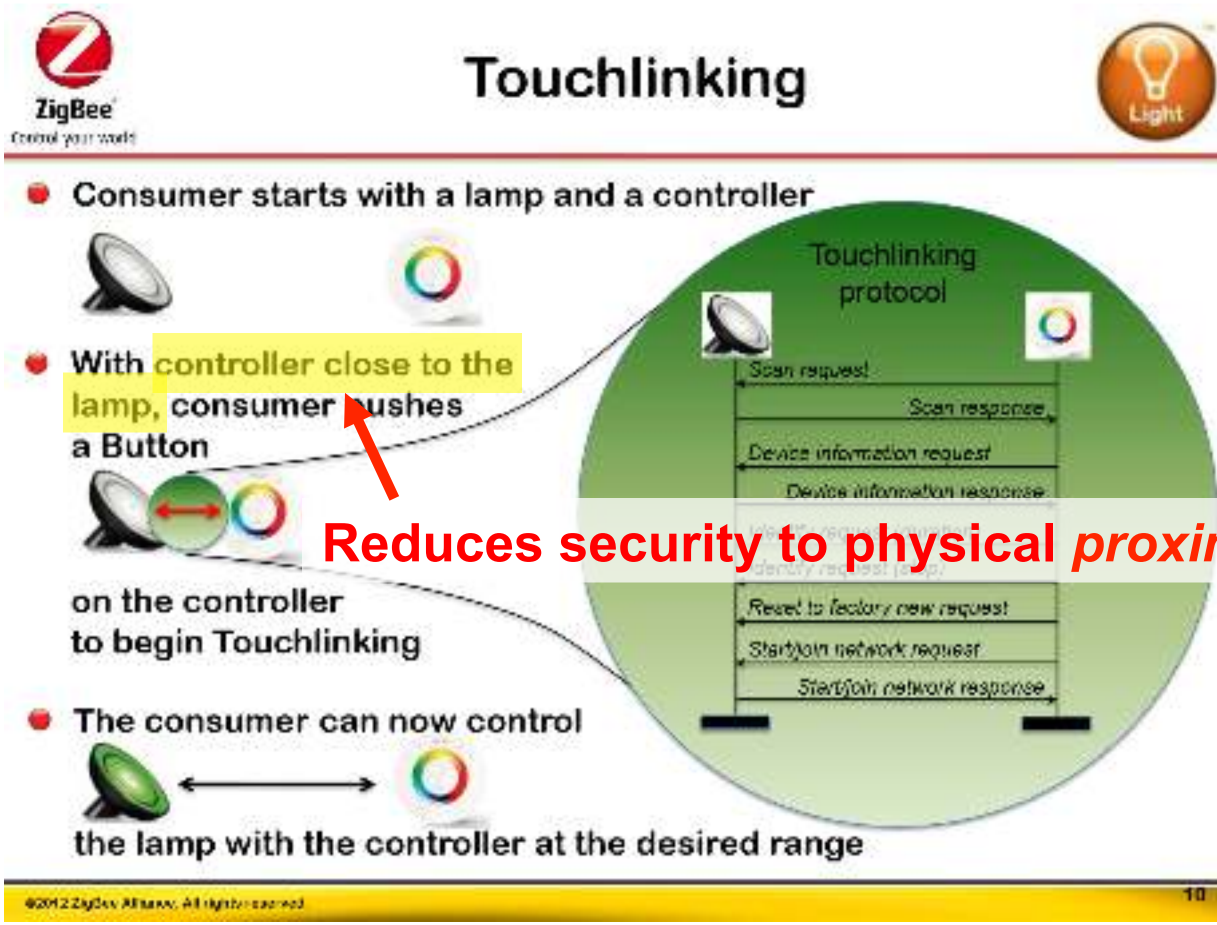
One master key shared by all Hue base stations

http://www.zigbee.org/portals/0/documents/events/2012_04_26_ZLL-Green%20Lighting-Heile.pdf

Two-level trust: Who's trusted now and who to trust

Bulbs and base station ship pre-configured with **TWO** shared secrets

The importance of understanding a threat model (and why what came before actually wasn't so bad)

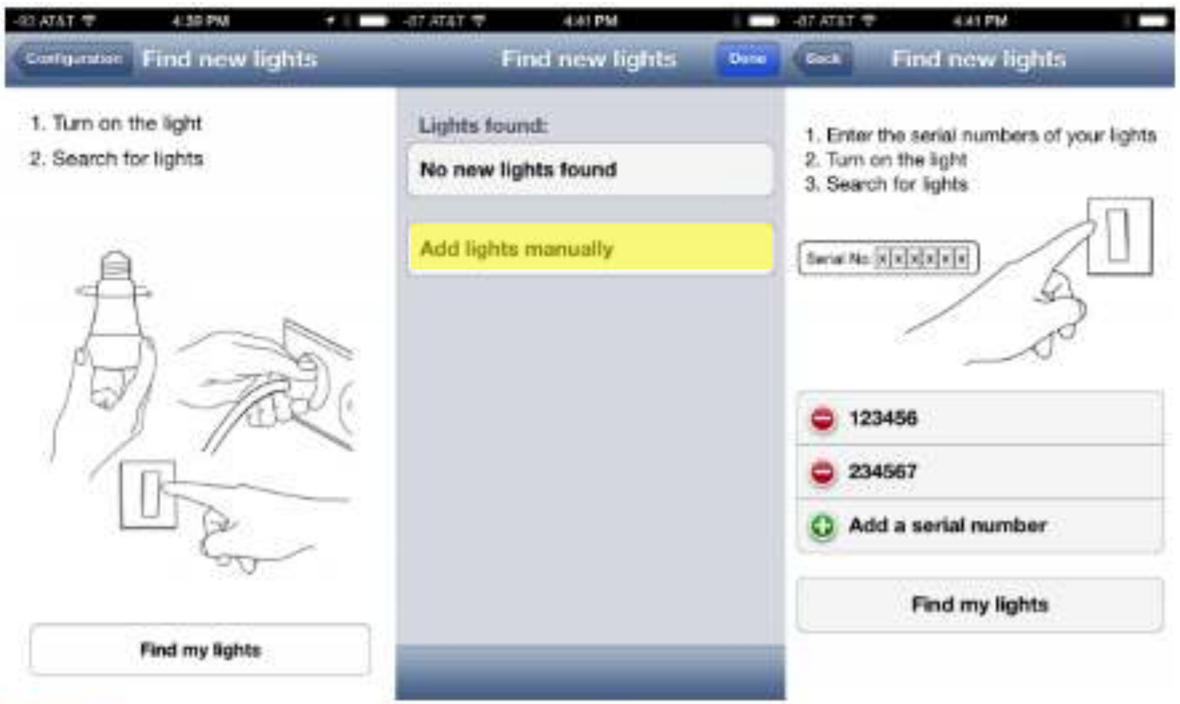


Reduces security to physical *proximity* (RSSI)

http://www.zigbee.org/portals/0/documents/events/2012_04_26_ZLL-Green%20Lighting-Heile.pdf

And then usability demanded an extension that makes it worse

ArsTechnica, 5 Nov 2013:



Enlarge / Rather than the crazy procedure previously required to add pre-paired Hue bulbs to an existing kit, the updated app provides a manual serial number-based method. We were able to add the three downlights to our existing bridge with no trouble.

Lee Hutchinson

Authenticate via **immutable 6-digit** bulb serial number

Off-the-shelf ~40 ms per authentication attempt (due to slow web server)

One-time cost to brute force:
Just over a week

http://arstechnica.com/gadgets/2013/11/philips-hue-family-gets-brighter-with-new-type-of-light/

Embedded Device Design
Small things without buttons

500,000 Belkin WeMo users could be hacked; CERT issues advisory

IOActive researchers uncovered numerous vulnerabilities in all Belkin WeMo home automotation devices that put over half a million WeMo users at risk of being hacked, but when CERT tried to contact Belkin, Belkin chose not to respond at all.

RELATED

Belkin fixes WeMo security holes, updates firmware and app

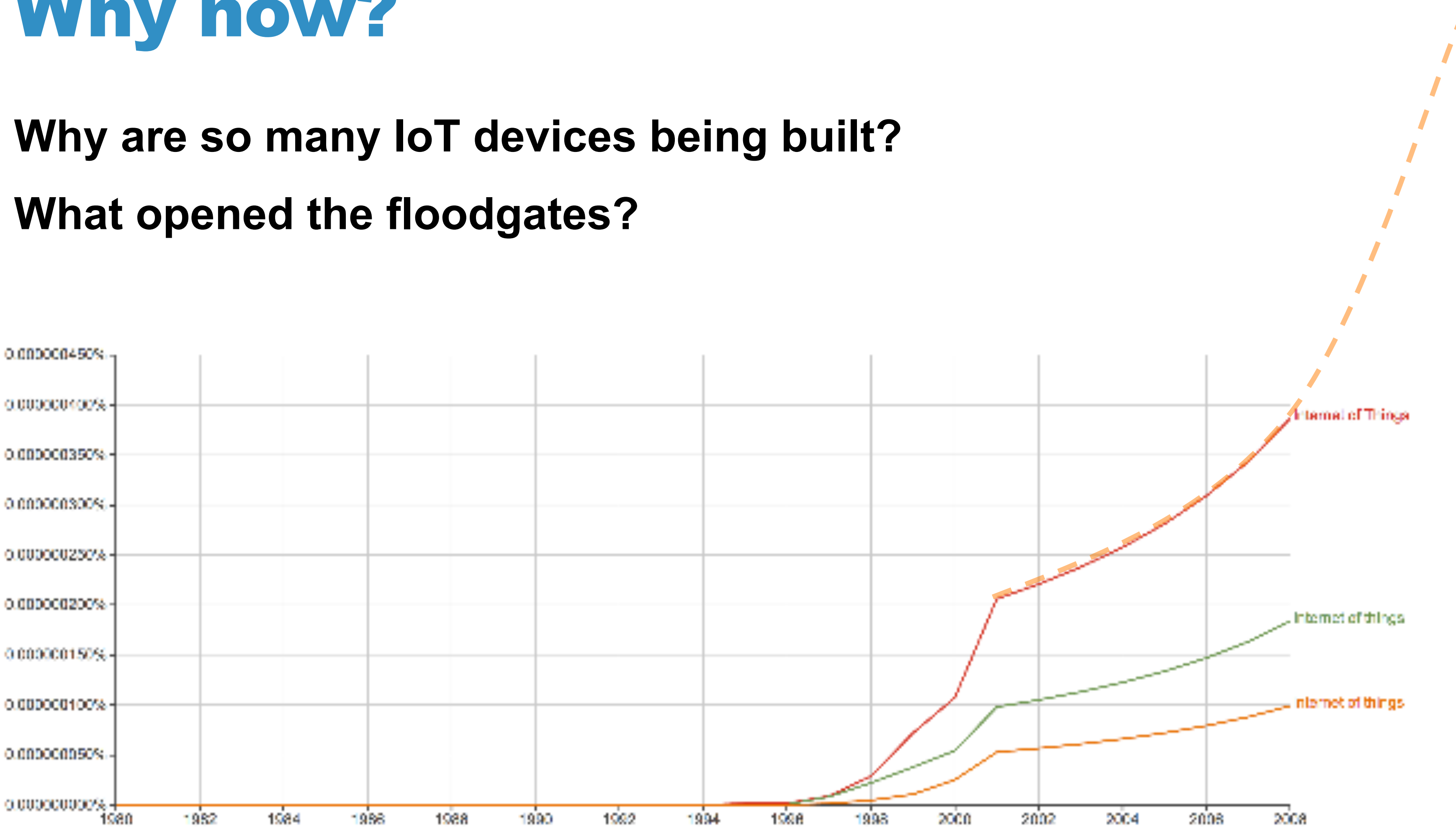
Eavesdropping made easy: Remote spying with WeMo Baby and an iPhone

Bizarre gadgets at CES 2014 that monitor your every move

Why now?

Why are so many IoT devices being built?

What opened the floodgates?

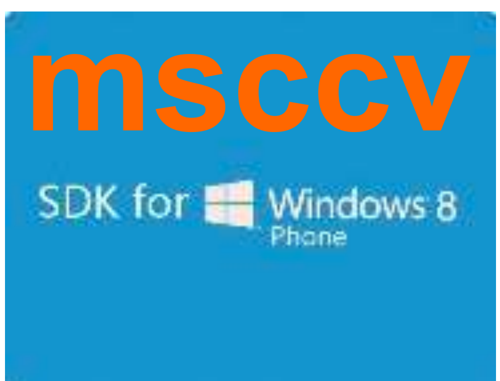


Google Ngram Viewer

The smartphone is an embedded system and a micro-PC

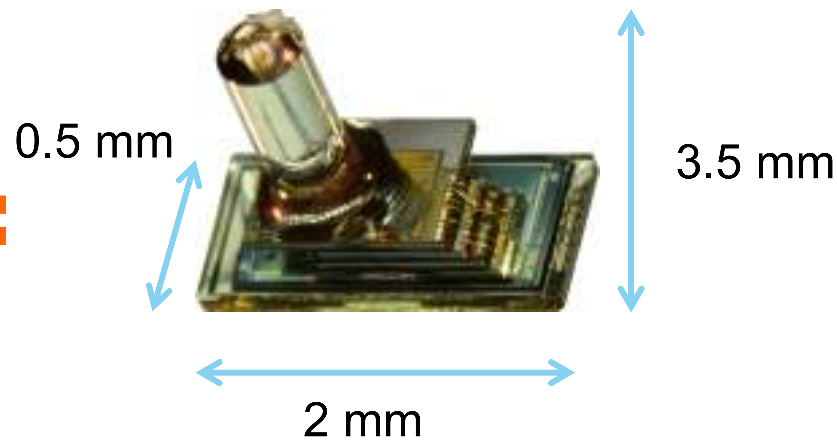
Very mature toolchains for embedded ARM cores

Driving down size, cost, and energy of peripheral sensors



Wait, why do we need IoT if there are smartphones everywhere?

!=



!=

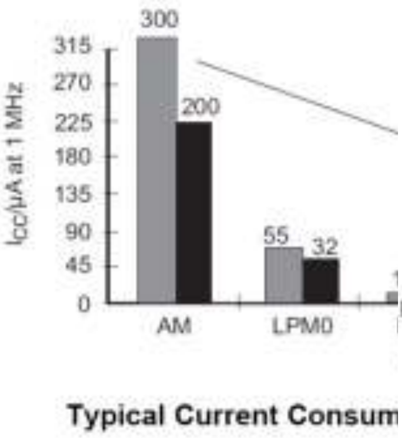
A reminder: Energy is king

CAN ONLY

Life Expectancy:
40 mAh

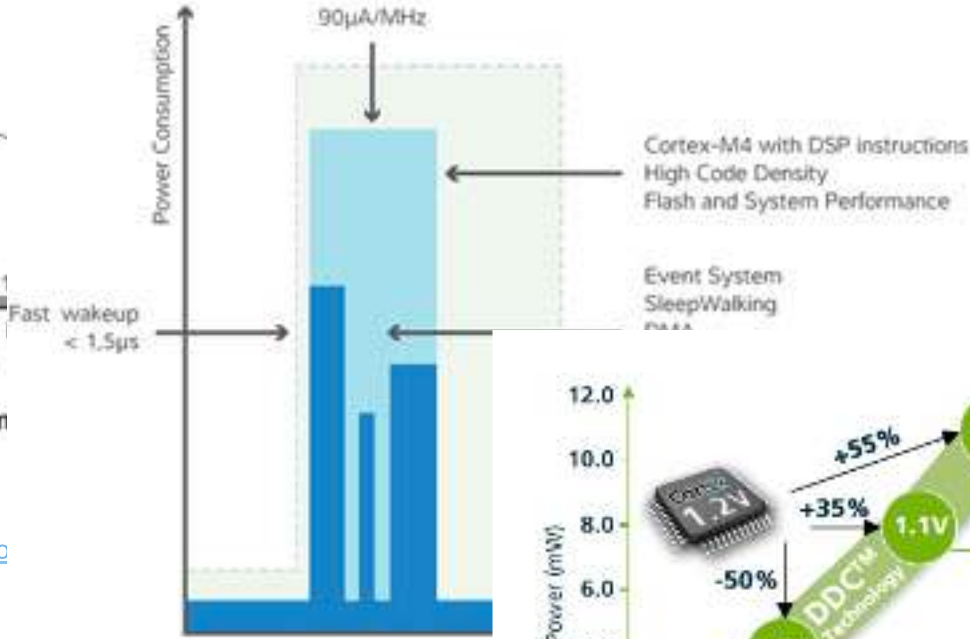
Moore's Law
Computing power in embedded

Low power 32-bit microcontrollers becoming reality



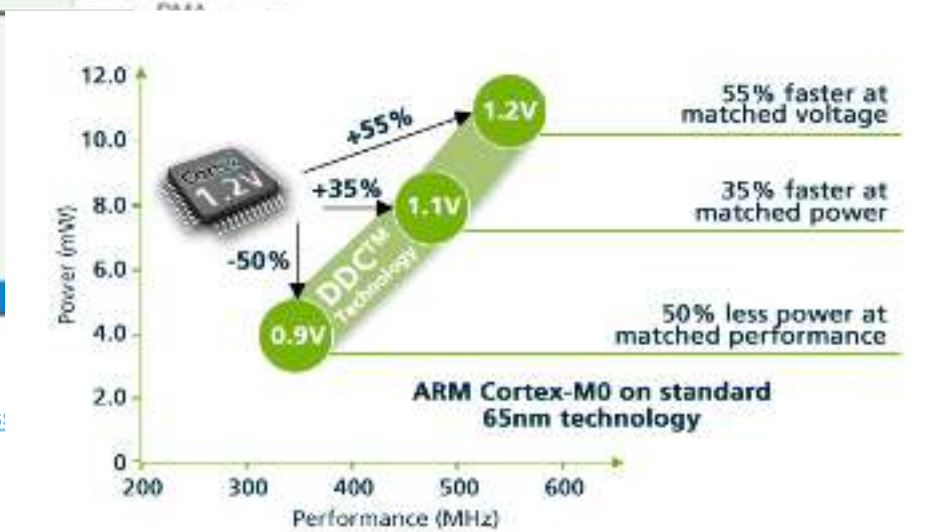
TI MSP430

http://forum.allaboutcircuits.co



Atmel SAM4L

http://atmelcorporation.wordpres
atmels-arm-mcu-picopower/



Suvolta

http://www.suvolta.com/technology/ddc/

Moore's Law and Memory
SRAM hasn't followed the same trend

SRAM ceiling

MSP430	**Atmel SAM4L**
0.125-66 kB	32-64 kB
ST Micro STM32L	**NXP LPC1xxx**
4-80 kB	1-36 kB

Why has embedded memory size not followed the rest of computing?

Q: Servers have terabytes of RAM, why is embedded memory following a slower trajectory?

A1: Demand. Lower-performance cores restricted the scope of embedded applications, limiting demand for RAM.

A2: Cost/Area. Don’t expect to improve on 6T / bit.

Ultra-low power cells are even more demanding, e.g. 11T / bit.

A3: Energy. SRAM contributes to static power of minimum (useful) power state

Some cautious SRAM Predictions

It won't get much bigger in the short term.

Power-State Partitioned SRAM?

Provides larger working set for applications, while enabling a minimum useful low power state

The rise of FRAM.

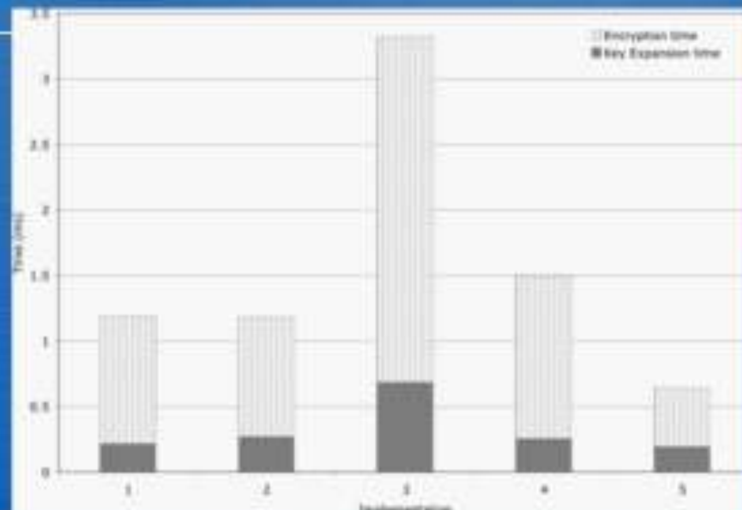
Replace "core" partition above with zero static power.

e.g. TI's Wolverine

What does all of this mean for security? Cryptography is computationally hard…

Other Implementations

- Improved both speed and code size (RAM unknown)
- Note that our versions seem to have varied significantly from published numbers in some cases



Implementation	Reference paper	Measured ROM Usage	Published ROM Usage
1	[6]	5968 bytes	n/a
2	[12]	6780 bytes	12616 bytes
3	[14]	6848 bytes	3322 bytes
4	[10]	n/a	n/a
5	Our implementation	5160 bytes	n/a

Didla et al., Optimizing AES for Embedded Devices and Wireless Sensor Networks
https://engineering.purdue.edu/dcsl/presentations/2008/aes_tridentcom.pdf

Cryptographic systems are designed with hardware acceleration in mind

Sometimes you really need a Cup Holder.

Atmel SAM4L AES-128 coprocessor – **11 cycles / block**

But how to actually USE it?

1. Ensure clock mask includes HSBMASK
2. Configure mode / other settings
    1. §18.4.1-2 “Basic Programming and Operation”: 2 pages / 1200 words
3. DMA + Sleepwalking
    1. Another dozen pages…

We solve these kind of problems with “drivers”

Things are not so elegantly encapsulated in embedded systems

First-generation solved by **Phil Levis**

TinyOS 2.0 abstracts resource management and peripheral power states

Good enough for then, don’t handle now

- “Sleepwalking”
- Multi-clock options / decisions

Open-problem in OS design

Reasoning about performance at odds with security

18.4.5 Security Features

18.4.5.1 Hardware Countermeasures Against Differential Power Analysis Attacks

AESA features four types of hardware countermeasures that are useful for protecting data against differential power analysis attacks:

- Type 1: Randomly add one cycle to data processing
- Type 2: Randomly add one cycle to data processing (other version)
- Type 3: Add a random number of clock cycles to data processing, subject to a maximum of 11 clock cycles for key size of 128 bits
- Type 4: Add random spurious power consumption during data processing

By default, all countermeasures are enabled. One or more of the countermeasures can be disabled by programming the Countermeasure Type (CTYPE) field in the MODE register.

The countermeasures use random numbers generated by a deterministic random number generator embedded in AESA. The seed for the random number generator is written to the DRNGSEED register. Note that access to the DRNGSEED register is by 32-bit words only (i.e., no halfword or byte access). Note also that a new seed must be written after a change in the key size.

Note that enabling countermeasures reduces AESA's throughput. In short, the throughput is highest with all the countermeasures disabled. On the other hand, with all of the countermeasures enabled, the best protection is achieved but the throughput is worst.

Coprocessors can mean crypto libraries are *less portable*

Existing software crypto libraries provide a good interface

Q1: Does HW interface always match the "standard" SW interface?

Q2: What accelerators are available?

Q2.1: How do app developers say,

"I want 'enough', 'efficient' security"

Q2.2: How to build heterogeneous networks with efficient crypto in the face of heterogeneous chips and co-processors?

A partial answer is provided by existing protocols

Bluetooth Low Energy use AES-128-CCM for most operations

And AES-128-CCM accelerators are available on every Bluetooth chip

Master Key exchange is host-side and can change protocol

This is an interoperability trade-off

Security vs Energy tradeoff. BLE requires out-of-band initial pairing to secure against eavesdroppers

Association Models

Not a good property for using smartphones as roving, universal gateways

Blue ... Entr ... equivalent of Numeric Comparison. Each of these association models is similar to Secure Simple Pairing with the following exception; Just Works and Passkey Entry **do not provide any passive eavesdropping protection**. This is because Secure Simple Pairing uses Elliptic Curve Diffie-Hellman and Bluetooth Smart (low energy) does not. The use of each association model is based on the I/O capabilities of the devices in a similar manner as Secure Simple Pairing.

https://developer.bluetooth.org/TechnologyOverview/Pages/LE-Security.aspx

lab11.eecs.umich.edu

Hardware Takeaways

Energy is king.

Dictates system lifetime

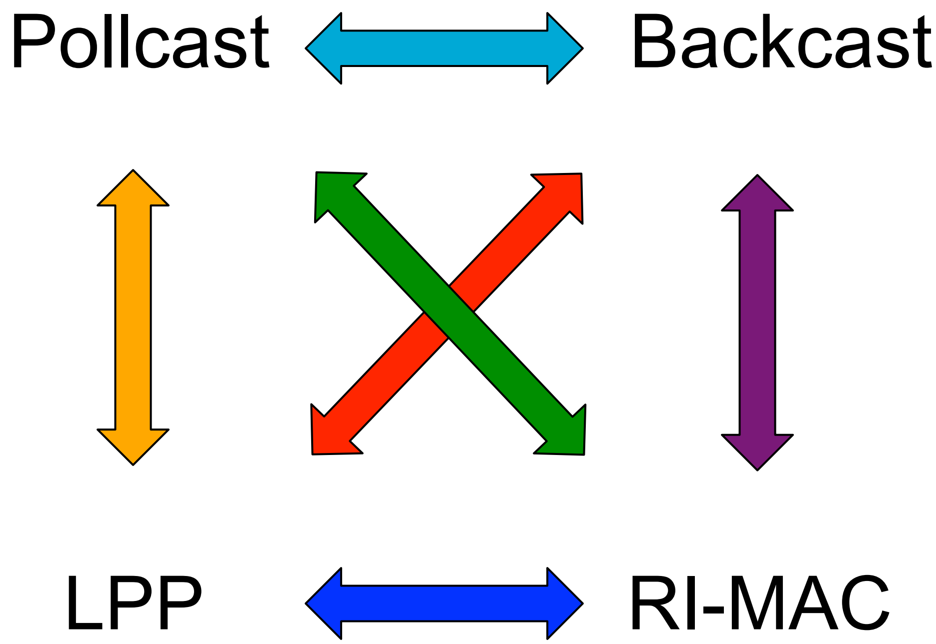
Computational power has come to embedded MCUs, but

Application complexity limited by limited SRAM

Hardware support can enable energy-efficient complex tasks

Perhaps a sweet spot between ASIC and FPGAs {David Brooks}

THE PROBE INCOMPATIBILITY MESS



Probes use hardware acknowledgements

Probes do not use hardware acknowledgements

Probes include only receiver-specific data

Probes include sender-specific data too

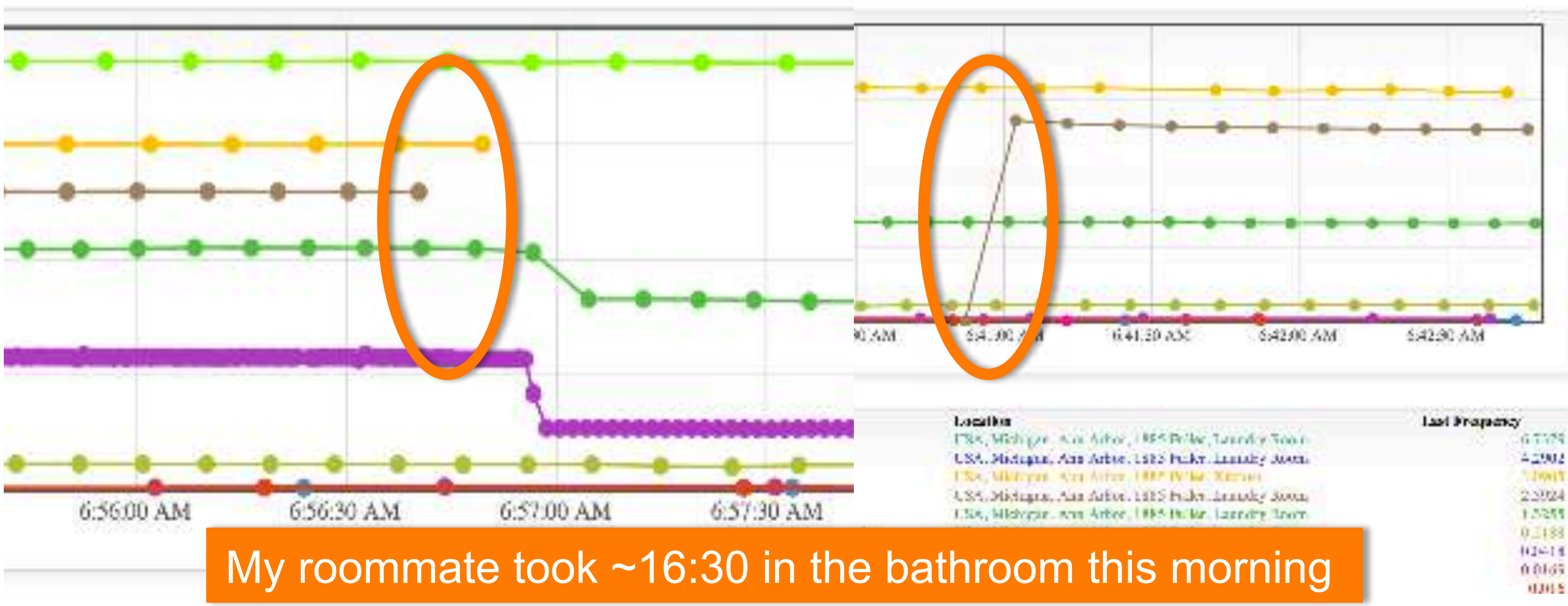
Probes include contention windows

Probes do not include contention windows

Cultural help: We like data too much for our own good

We can see devices turn on / off

Lose sight of the fact that humans are turning them on and off

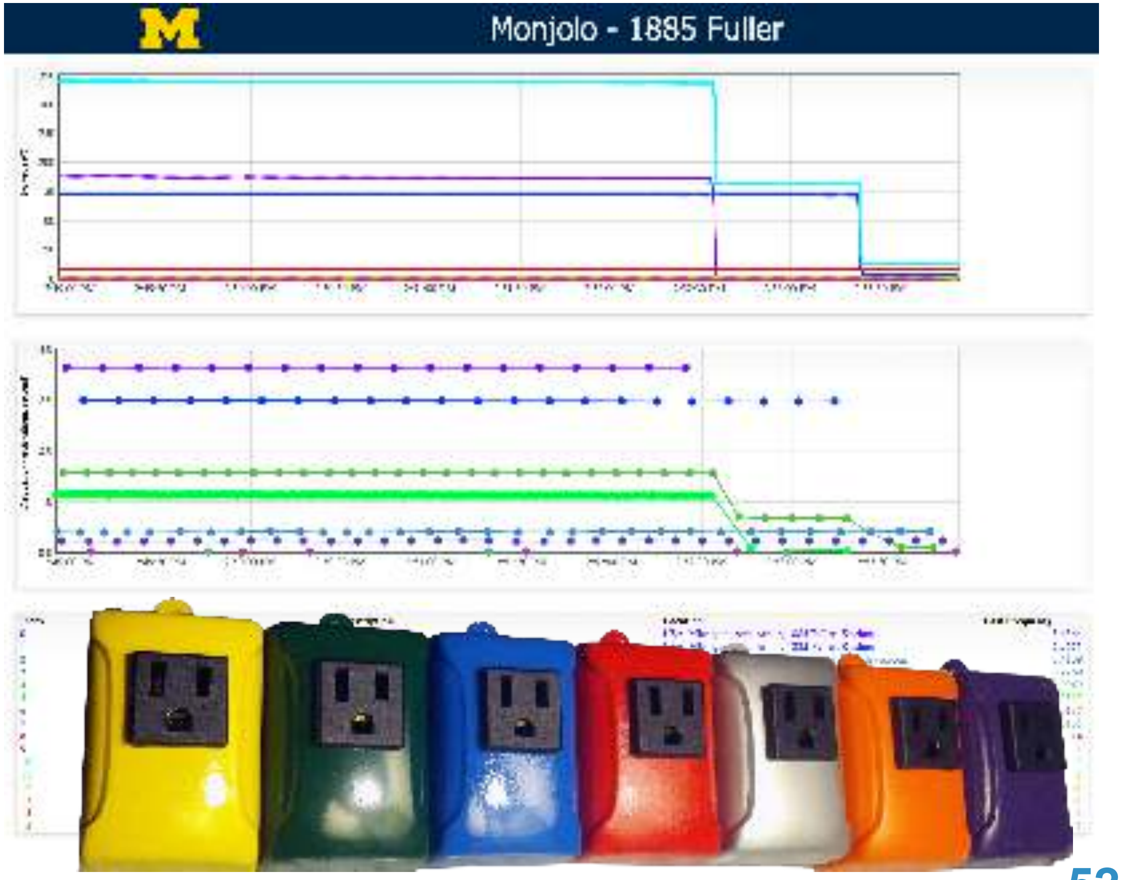


Privacy implications

What should a system like this look like?

Monjolo in action

http://inductor.eecs.umich.edu/pathouse.html



Augmenting smartphones to add "missing" sensors

Geiger Counter

EKG Monitor

Thermometer

CC Terminal

CO Sensor

Soil Moisture

[slide courtesy Prabal Dutta]

Lab11 at Michigan: (Some of) What we do

eMbedded

Energy-Harvesting
Embedded Systems
Built in the last 365 days
(credit: Brad Campbell)





Lab11 at Michigan: (Some of) What we do

eMbedded

Embedded Systems
Built in the last 365 days