Internet of Things Security Primer

IoT North America, Chicago, April 2015

Your presenter: Eric Üner

CTO of Redwall

Device security for mobiles, SCADA, etc.

Integrity checking, DPA resistance, anti-rooting, n-persona, and more

Offensive work

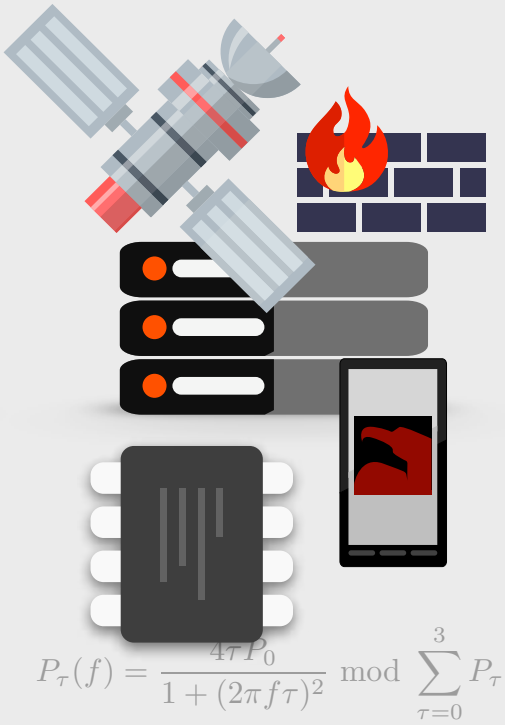
Not a researcher - call it "applied ethical hacking," typically with a specific mission

All manner of devices and infrastructures

Defensive work

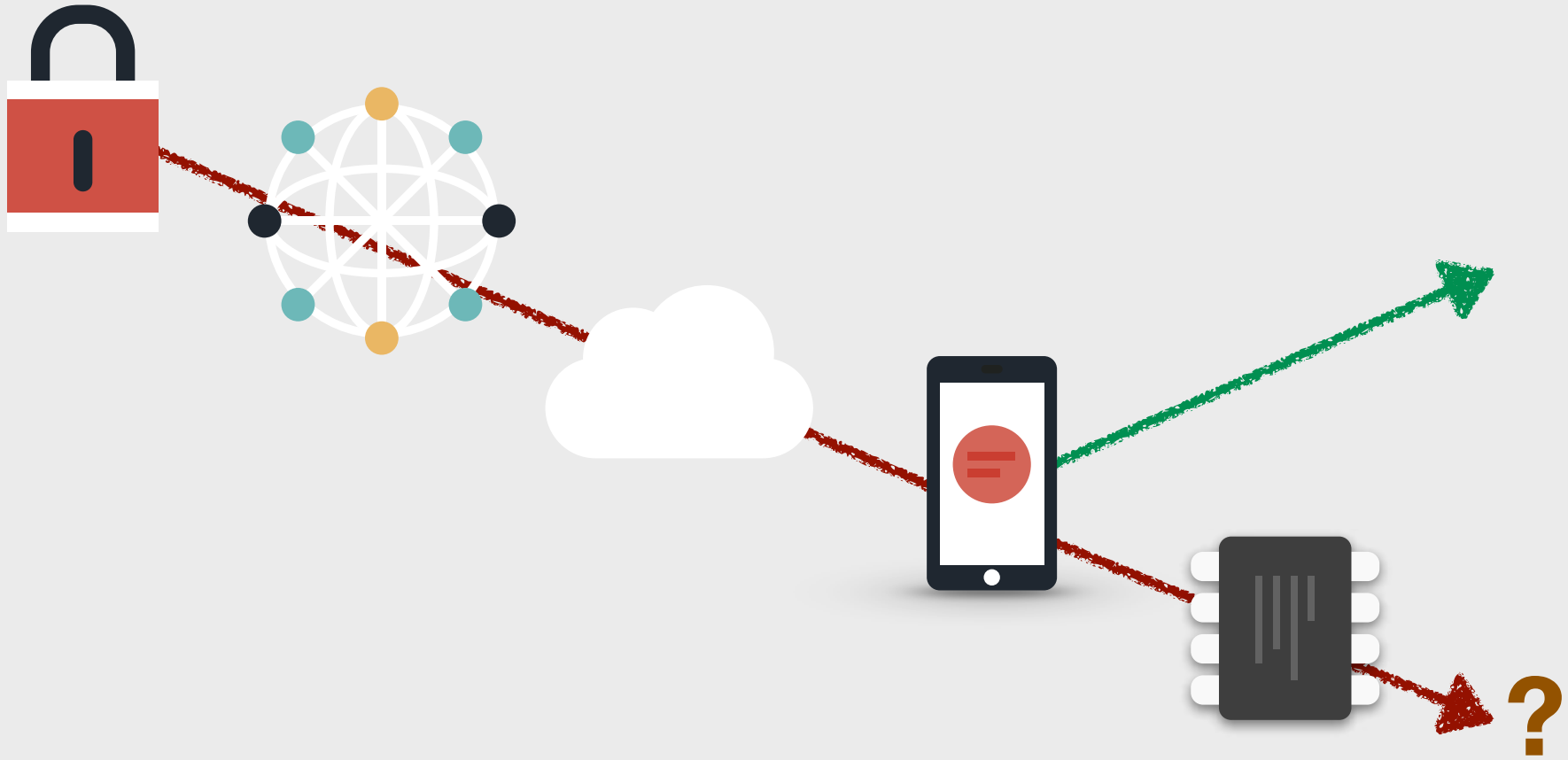
Often direct response to offensive work

Systems, algorithms, architectures



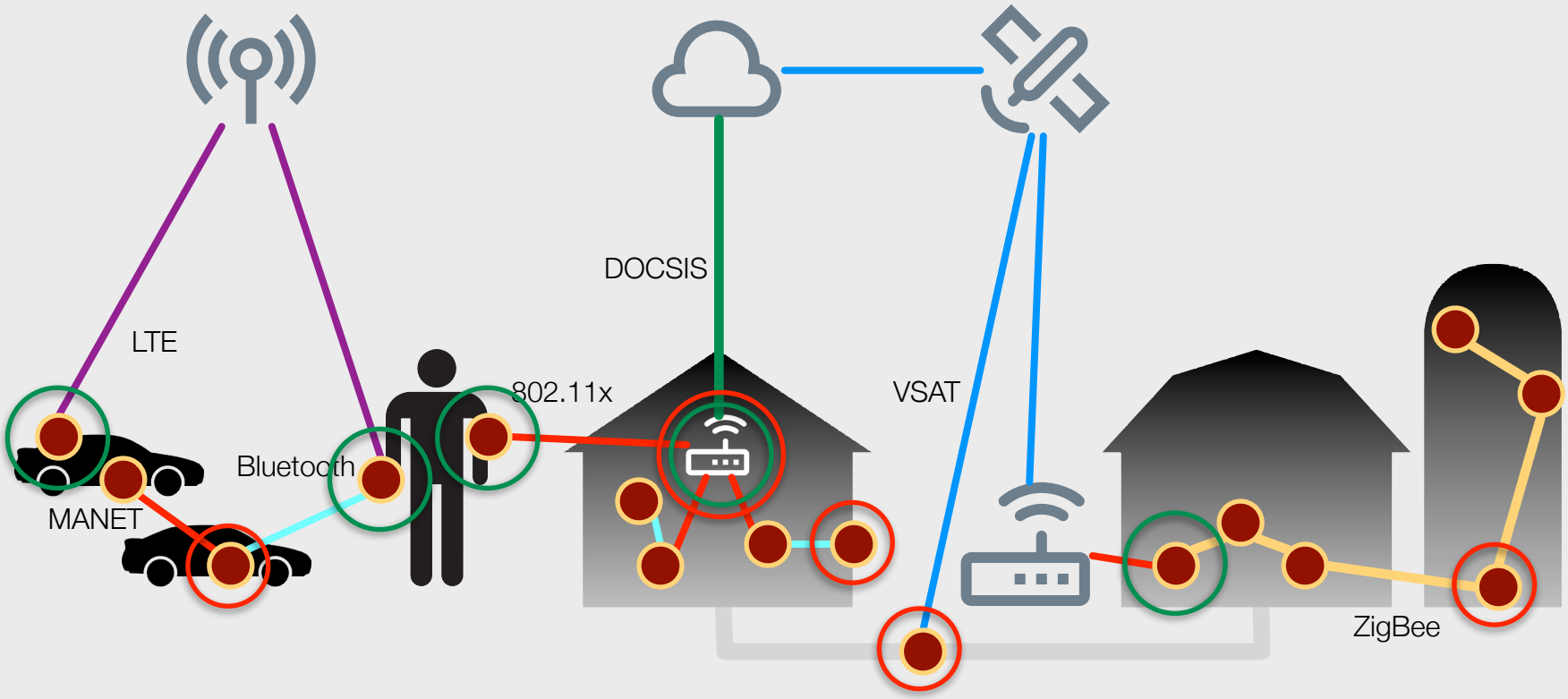
$$P_\tau(f) = \frac{4\tau P_0}{1+(2\pi f\tau)^2} \bmod \sum_{\tau=0}^{3} P_\tau$$

Goal for today



IoT Architecture(s) and roles

There are more possibilities, protocols, vendors, and standards than I can possibly illustrate in one diagram, but here's one simple example…

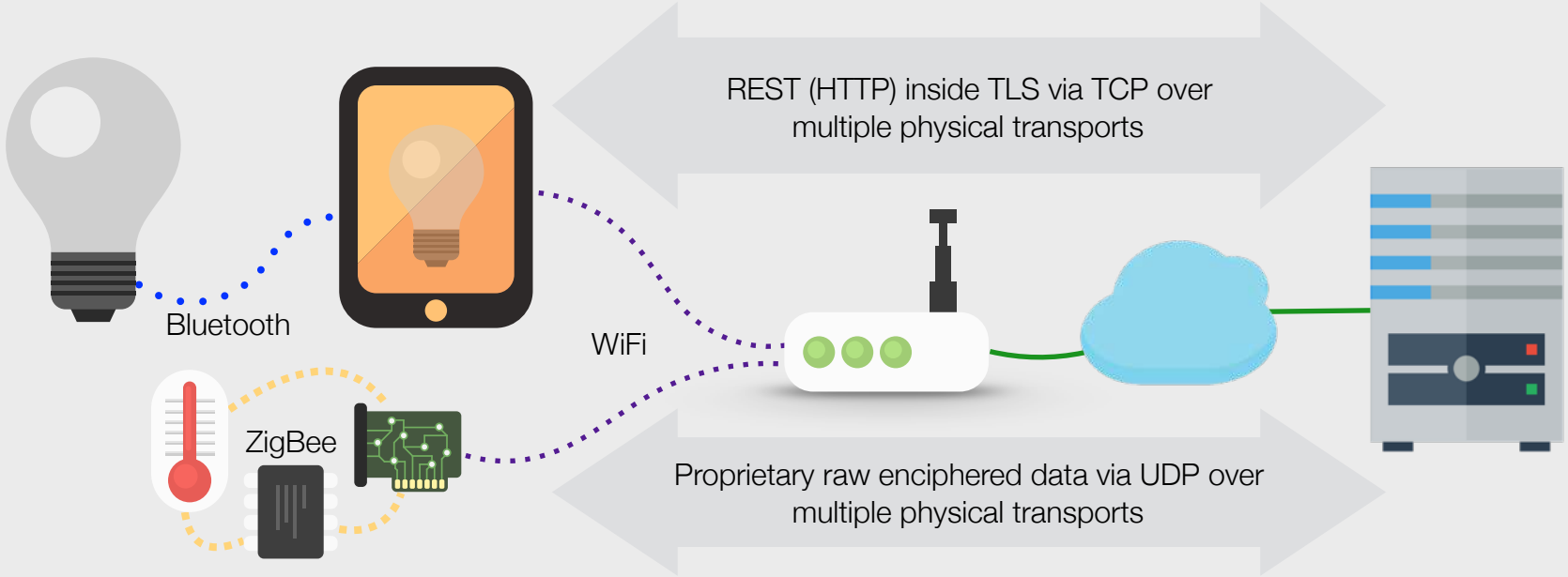


Users: individuals, businesses (i.e. users)

Manufacturers: ODMs, OEMs, service providers, etc.

Often they overlap
Often they are at odds wrt "security"

How are all these things connected?



Dozens of protocols inside protocols

Just as many ways to connect

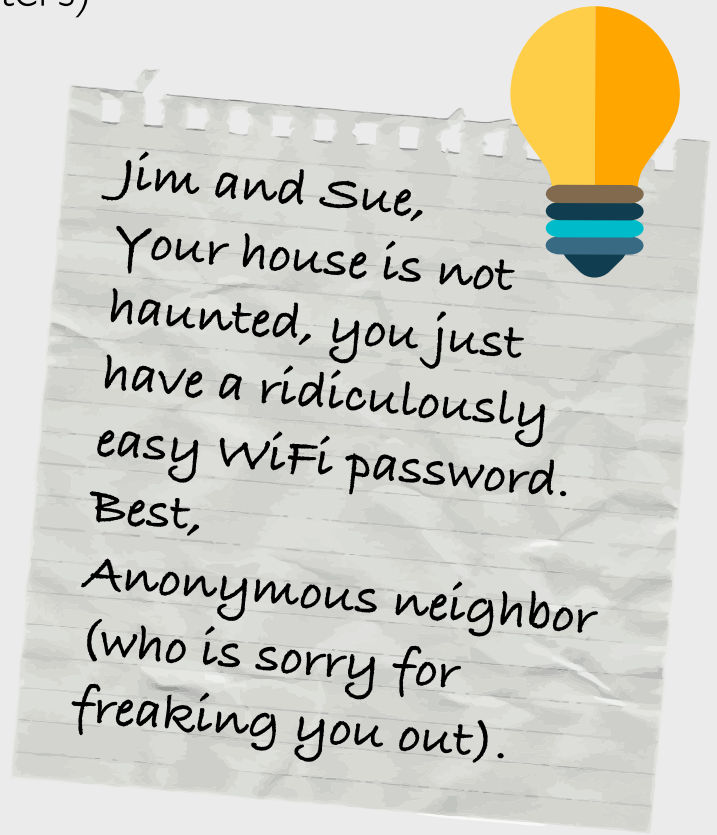
Which ones you use depend on your needs and capabilities

Depends on SWAP constraints, scalability, and connectivity needs

And of course security!

Pairing and provisioning

- Bluetooth and WiFi (802.11)
  - Smart bulbs have no UI for BT PIN or WiFi password
  - Simple for Users = simple for attackers (and pranksters)
- Zigbee
  - Pre-installed keys
- 4G and others
  - Pre-installed keys
  - SIM card or soft-SIM
- Often device limitations can be our friend
- Other considerations
  - Key distribution and management
  - Server-side security
  - Registration can help thwart rogue devices

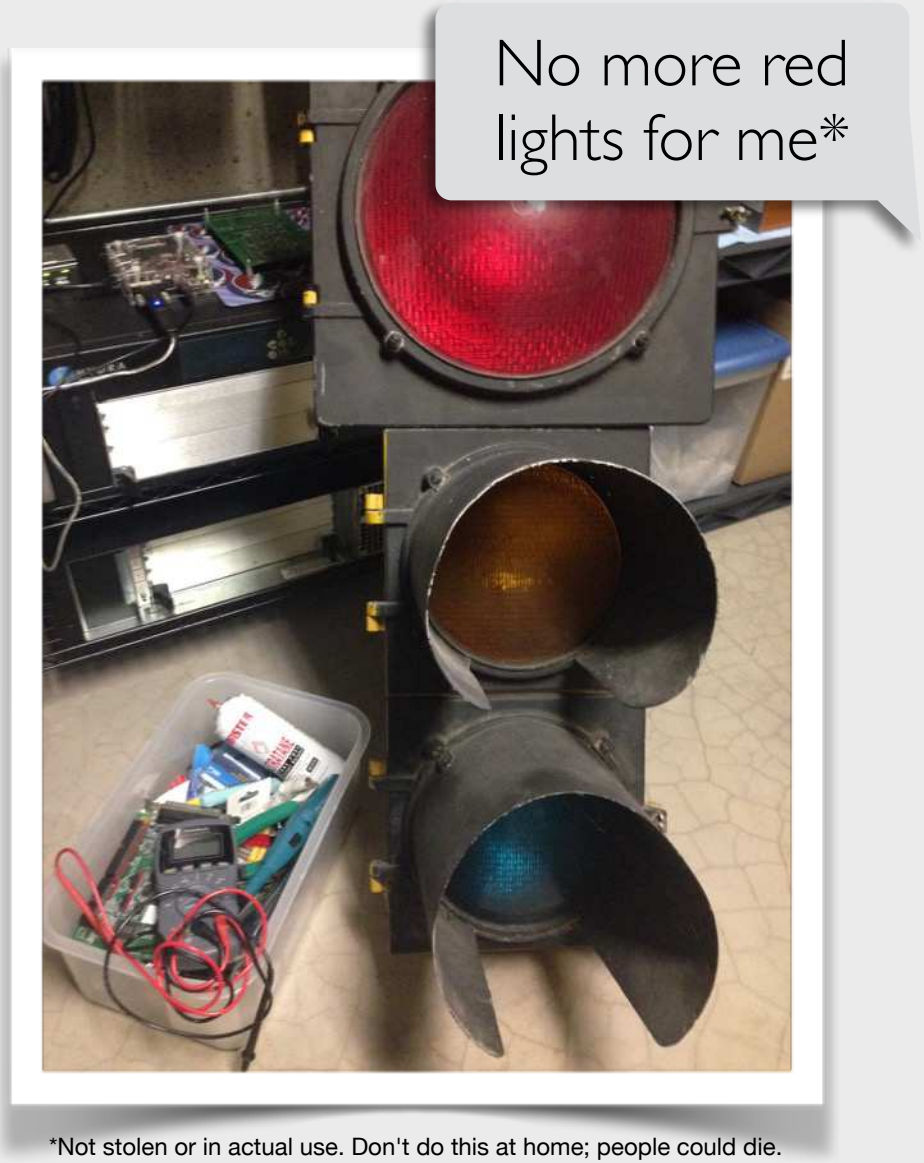


Make no assumptions

Don't overestimate your effectiveness at reducing physical access

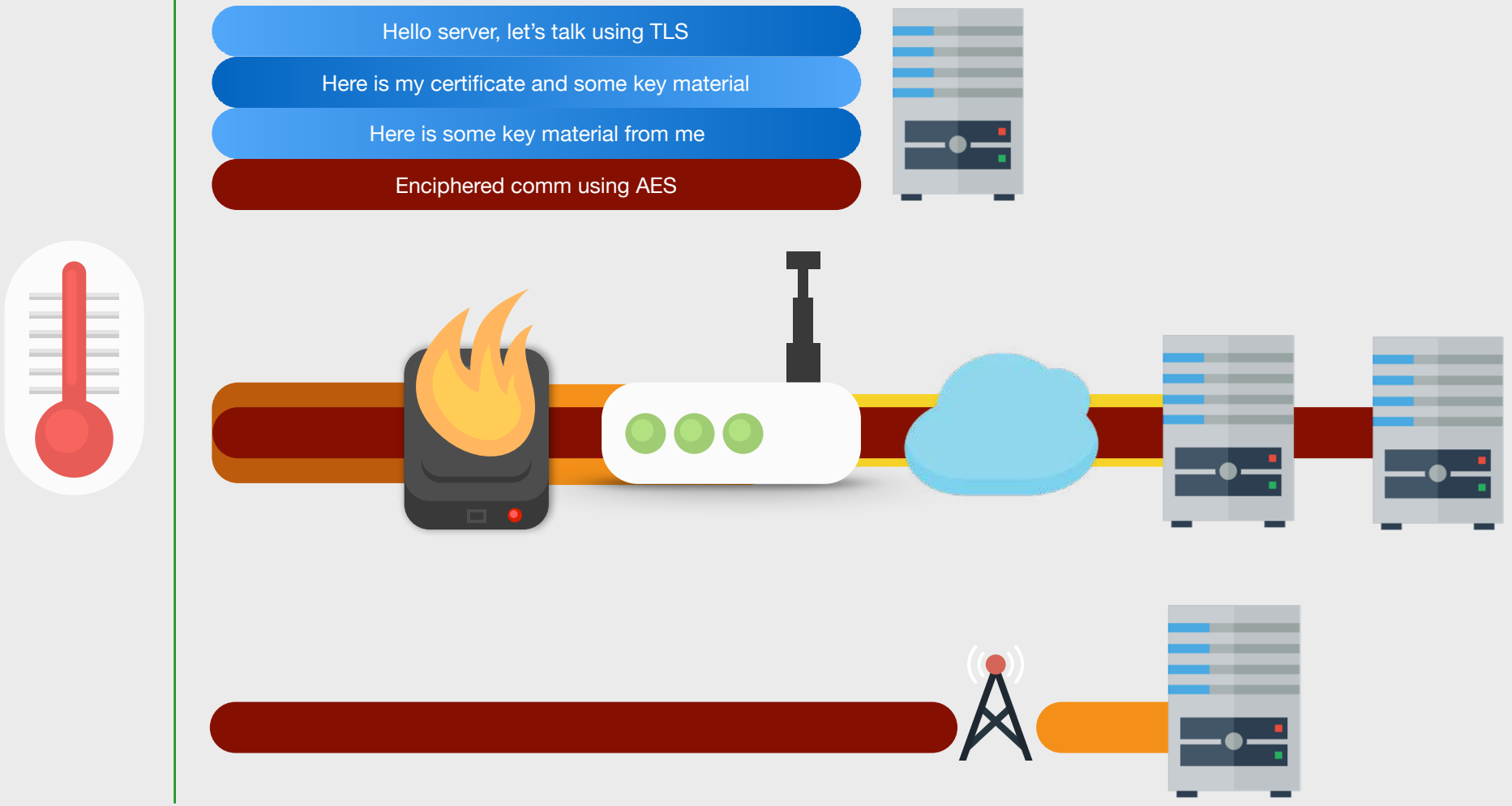
Reduce window of attack

More on chain of custody and anti-tamper later

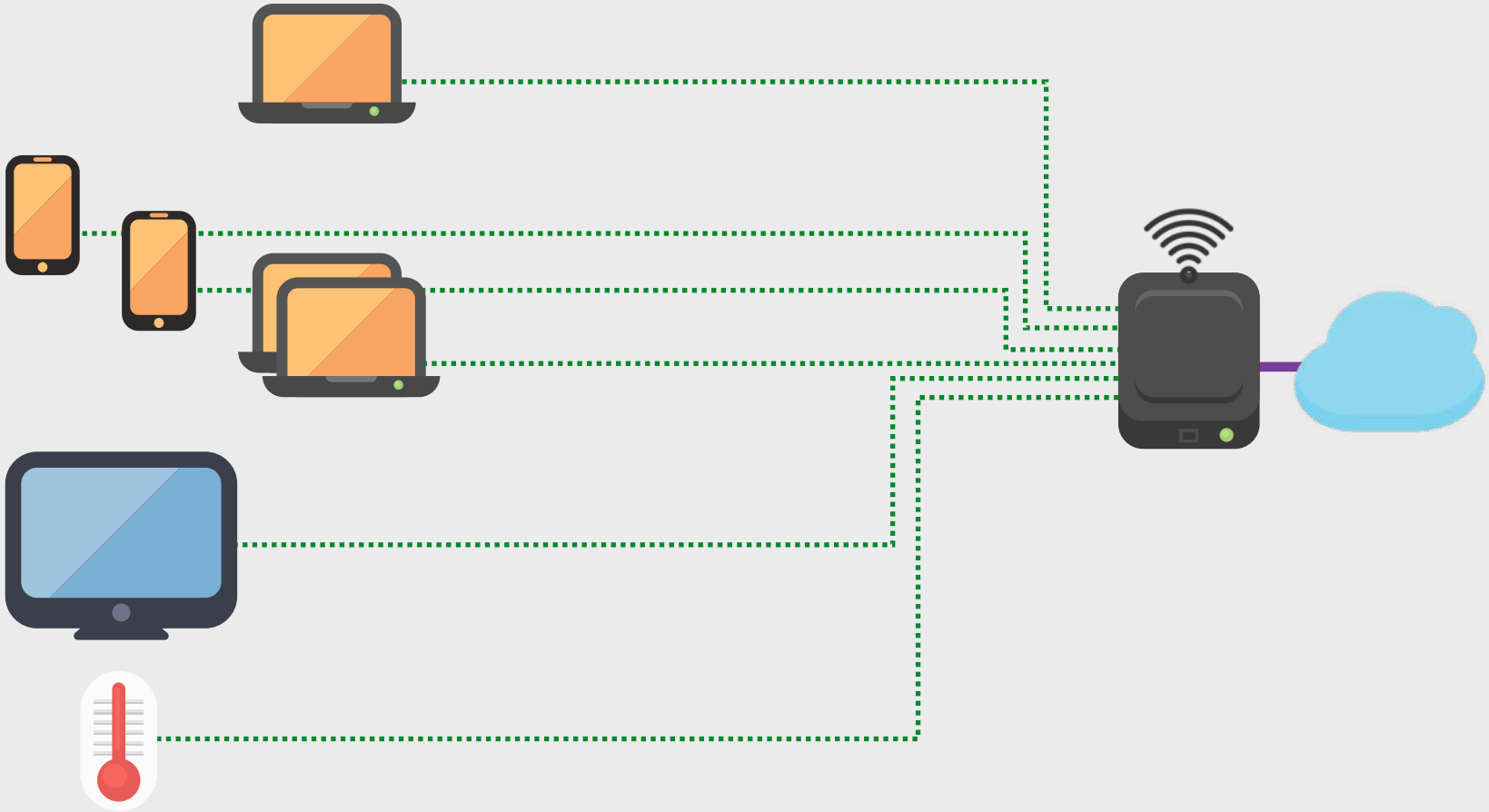


*Not stolen or in actual use. Don't do this at home; people could die.

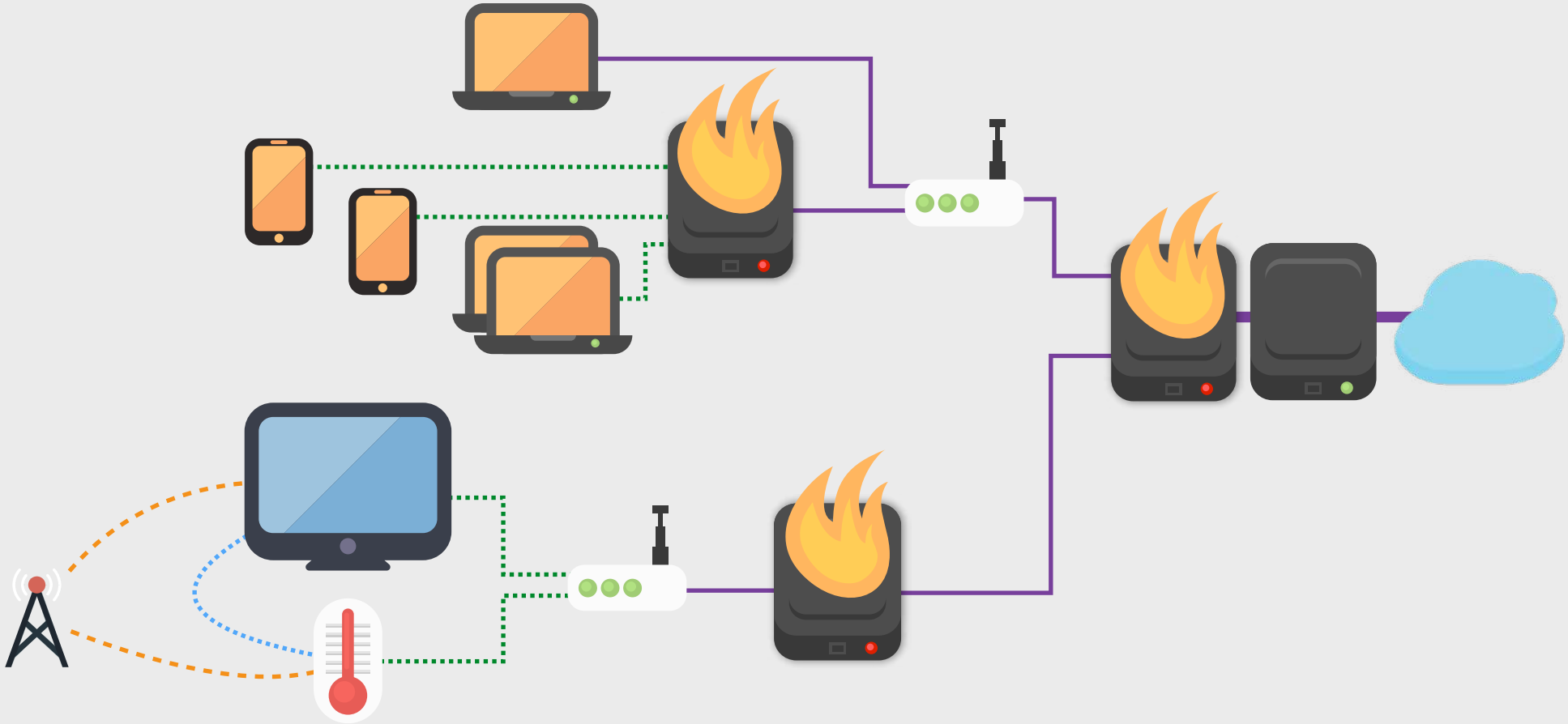
Encryption


Hello server, let's talk using TLS
Here is my certificate and some key material
Here is some key material from me
Enciphered comm using AES

Typical household or SMB



Advanced household (or enterprise)



I like having firewalls and routers and proxies

But this does complicate IoT device integration, and worse…

IPv6 and proprietary protocols were bad enough

But this will be my undoing - how do I protect my devices and network?

Encryption algorithms

In general, use NSA Suite B

- Advanced Encryption Standard (AES) with key sizes of 128 or 256 bits in CTR or GCM mode
- Elliptic Curve Digital Signature Algorithm (ECDSA) for digital signatures
- Elliptic Curve Diffie-Hellman (ECDH) for key agreement
- Secure Hash Algorithm 2 (SHA-256 and SHA-384) for message digest

Keep in mind even encrypted data is useful to attackers

- If I see encrypted traffic on your WiFi, I know you're home
- And metadata can be as useful (or more so) than data

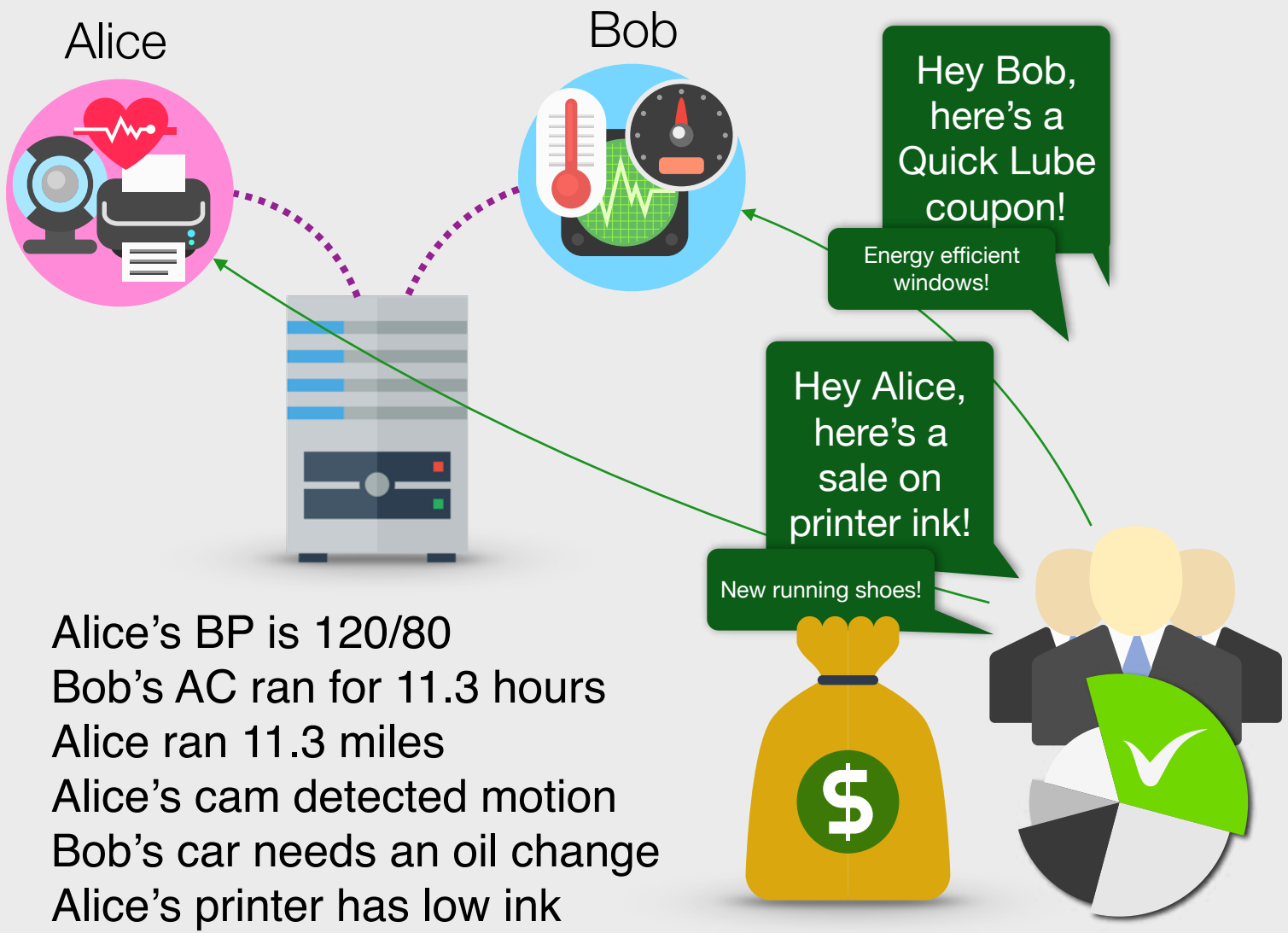
Encryption

Users	Manufacturers
Makes them feel safer	Key distribution and management can be a challenge
Prevents attacks from trivially getting data to and from devices	Extra cpu required
In some cases, actually does protect their information	Debugging issues is more difficult

The bottom line: you have to do it, and you have to do it correctly

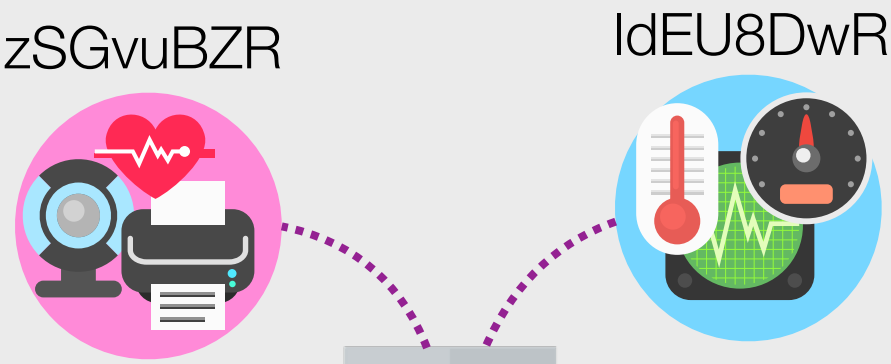
Encryption does not equal security (it's a very small part of it in fact)

Anonymity



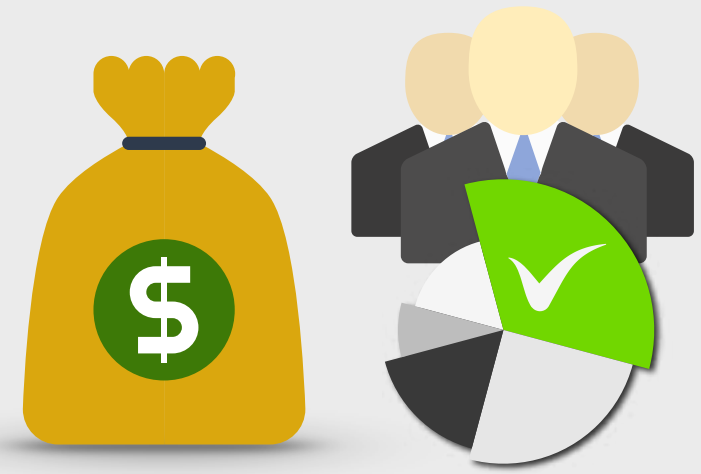
Alice's BP is 120/80
Bob's AC ran for 11.3 hours
Alice ran 11.3 miles
Alice's cam detected motion
Bob's car needs an oil change
Alice's printer has low ink

Anonymity



Alice zSGvuBZR
Bob IdEU8DwR

zSGvuBZR's BP is 120/80
IdEU8DwR's AC ran for 11.3 hours
zSGvuBZR ran 11.3 miles
zSGvuBZR's cam detected motion
IdEU8DwR's car needs an oil change
zSGvuBZR's printer has low ink



Anonymity

Users	Manufacturers
Are terrified when big data firms correlate their habits	Makes data less valuable as a product, but creates upsell opportunity
And annoyed when those firms get it wrong	Reduces liability and improves compliance
Are willing to pay premiums for increased privacy	

Use your privacy policy as a differentiating factor

Don't just hash emails or other PII (personally identifiable information)

Know the laws where you are selling (US, EU, CA, etc.)

Securing the code in a device

Anti-tamper means many different things in different contexts

Prevent the firmware from being re-written or modified

Detect if the unit has been altered or accessed (tamper evident)

Verify the crypto is intact (per FIPS)

Destroy keys if device is tampered with

Permanently disable the device if it is tampered with

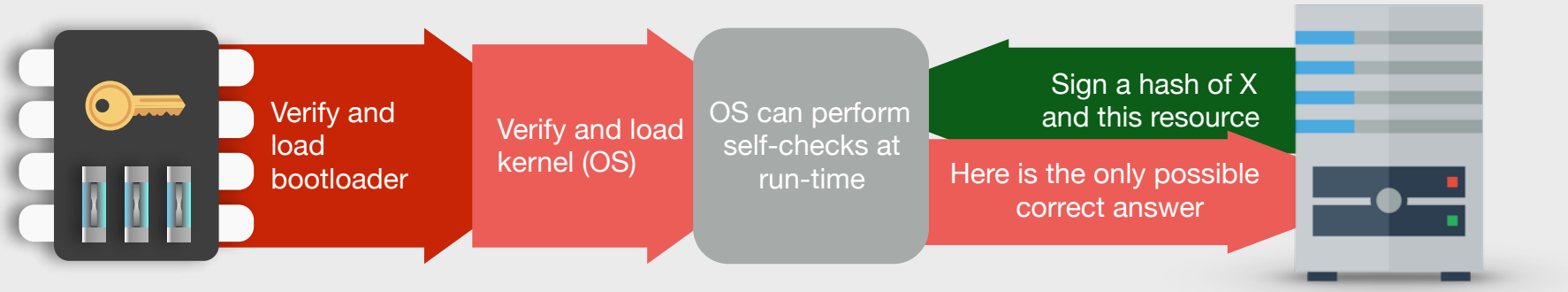
Prevent peripherals that could perform debugging or reverse engineering

Trusted boot can ensure integrity of code as it loads

Run-time integrity checks can verify that critical resources have not changed

Remote attestation can prove assurance level of device

Without integrity checking and trusted boot, device can lie



Securing the code in a device

Users	Manufacturers
Hobbyists want to be able to update devices	Requires extra or more expensive hardware in a few cases
For remote attestation, users don't like required connectivity	Adds very significant complexity
	Makes working with third parties and extending features more difficult

Worth it to thwart attackers and reduce support burden

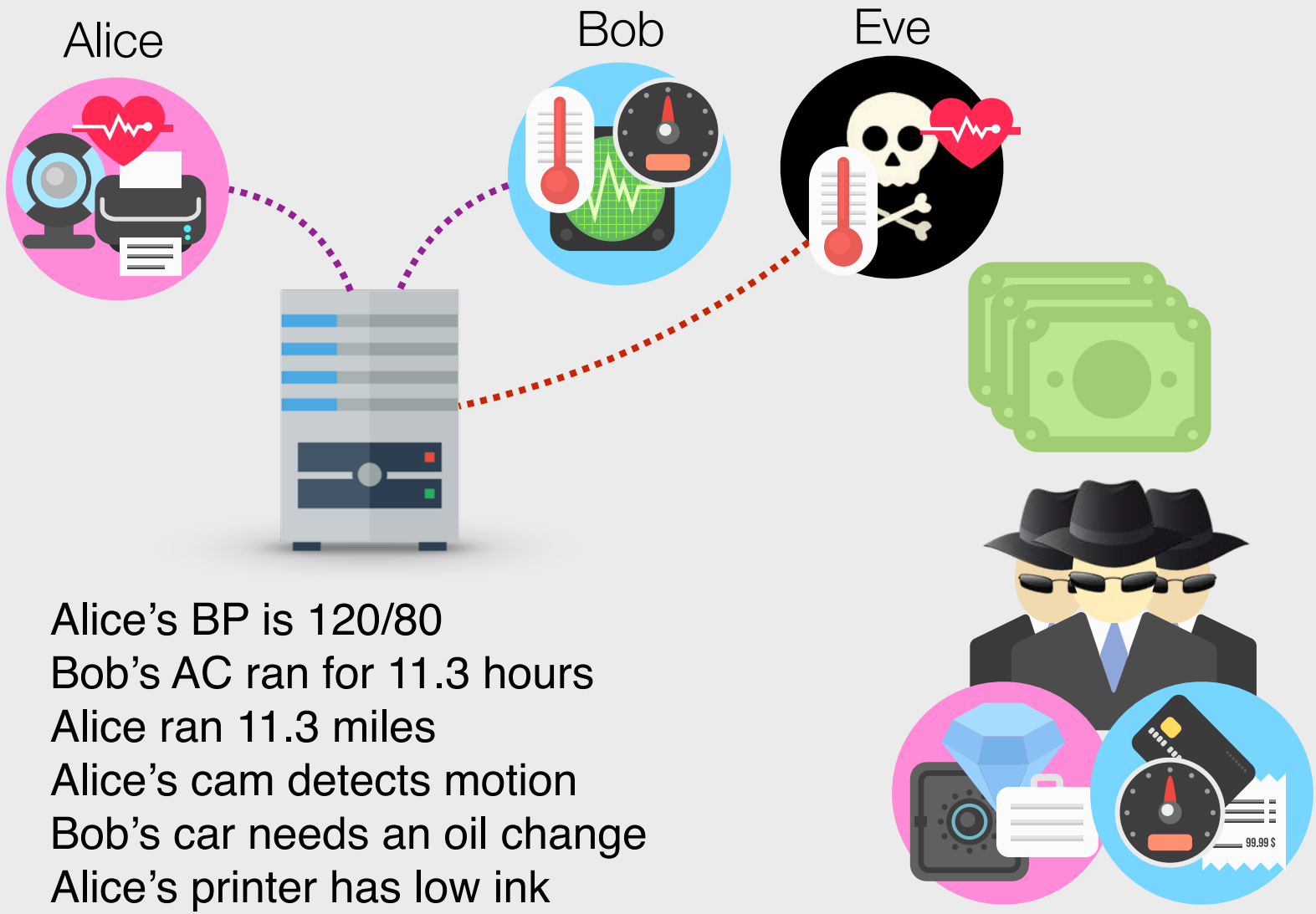
Simple enough to sign keys for trusted third parties or developers when needed (e.g. the Apple and Google models)

Must balance SWAP constraints with measurements

Must implement crypto properly

Disclaimer: Redwall helps device manufacturers with all this, so of course I would say that

Server-side security



Alice's BP is 120/80
Bob's AC ran for 11.3 hours
Alice ran 11.3 miles
Alice's cam detects motion
Bob's car needs an oil change
Alice's printer has low ink
Alice's identity is stolen
Bob just bought a new Macbook

Server-side security

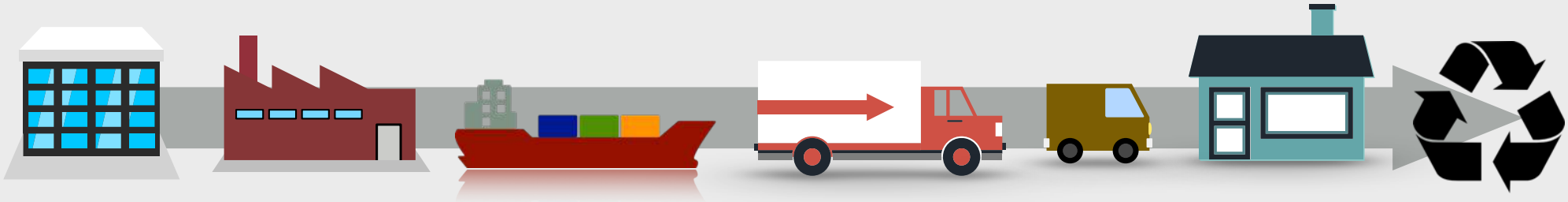
Common attack vectors I see

- Your devices
- Third party libraries for protocols and APIs
- Insecure logins to hosting services
- Users never changing default credentials
- Inadequate proxy or hosting setup

Don't bet on all developers to be security experts

Overlay your own security on top of hosting facility's (even if you are the hosting facility)

Chain of custody



Have a policy that documents responsibilities and procedures and follow it

Does not start with manufacturing or end with delivery

- Programmers and manufacturers can insert covert channels (or inadvertent backdoors)
- Materials can come from untrusted (and non-eco and worker-unfriendly) sources
- Discarded devices can create risks and liability

Chain of custody

Users	Manufacturers
Commercial customers typically don't care	Can use pallet tracking technologies (IoT for IoT)
Defense, CNI, medical and others may have specific legal requirements or acquisition regulations	Helps quality and inventory control as much as security
	Typically do not even understand risks of complete chain

About your enemy

- Hobbyists and users looking for features
  - Over 1/4 of users root android phones, about 7% of iPhone users Jailbreak (in the US, but almost 1/3 in China)
  - Reflashing can add features, or remove limitations
  - They often create security risks and become an attack vector
- Users often looking to circumvent paid services
  - Imagine free smart bulbs that require a paid app - how long before a counterfeit app would come out?
  - Can you manipulate your gas meter to cut your bill?
- Skilled and sponsored attackers with an agenda

Conclusion

Let's make the attackers really work for it

Make security a differentiating feature

- Customers are aware of cyber-crime and snooping

Build security into the product from the start, and make it part of your culture

Get design and code reviews by security experts (like Redwall)

Don't just depend on some API or library for security

Contact me if you need information, advice, pointers, or even opinions

Thank you!

Eric Ridvan Üner
eric.uner@redwall.us
http://www.redwall.us/

HTTP://WWW.REDWALL.US

Backup: Recommended products

For device security software/firmware, integrity measuring, and red team security analysis see Redwall Technologies, http://www.redwall.us

For pallet tracking see Private Pallet Security Systems, http://www.p2s2llc.com

Backup: Crypto concepts

PKI/Asymmetric cipher - public key infrastructure. In asymmetric ciphers, data is encrypted with one key and decrypted with another. Splitting the keys into a public and private key allows for both encryption and identification.

AES - advanced encryption standard. A symmetric algorithm, wherein the same key is used for encryption and decryption. AES operates in "modes" like Counter mode or ECB, which have different benefits in different situations.

Key and certificate management - the process of distributing keys and the certificates (standard file format for transmitting keys and chain of signatures).

NSA-approved - an often misused term, but generally refers to algorithms that the NSA approves of for use in different situations (search for "Suite B" and "Type II cryptography").

FIPS - federal information processing standards. A set of US standards that define approved algorithms and requirements for implementation of information security-related modules.

CC - Common Criteria. International standards for computer security certification.

NIST, NIAP - Bodies that maintain the above standards.

Watch out for "NSA approved" or "the same encryption used for Top Secret" and similar ambiguous marketing-speak. Saying algorithms are approved does not mean the implementation is certified.

ECC - elliptic curve cryptography. A standard asymmetric encryption algorithm based on elliptic curves.

SHA - secure hash algorithm. A hash transforms data in a way that, unlike encryption, cannot be (easily) reversed, in part because different data of different sizes may map to the same fixed-size result.

Digital signature - the result of a hash that is encrypted, typically with asymmetric algorithms to prove who performed the encryption and to prevent forgery.

Backup: Common APIs

HTTP - hypertext transfer protocol. The protocol used by Web browsers and servers, which because of its ubiquity and ability to pass through common firewall configurations has become a common catch-all in IoT protocols.

REST - representational state transfer. A common protocol that typically leverages HTTP to communicate almost anything, usually between devices and servers.

Ajax - client-side code that typically updates web pages while you are looking at them. Allows your browser or app to update device information in what appears to be real-time.

JSON - JavaScript object notation. A simple data exchange format that both humans and machines can create and interpret.

Backup: Common lingo

IoT - Internet of things (but if you don't know that, you may be reading the wrong slide deck).

M2M - machine to machine. Part of the IoT concept, it's devices communicating to other devices, typically with no user interaction.

API - application programming interface. The code to access an interface, often vendor-specific, or used to abstract vendor-specific programming libraries and simplify or aggregate access into something more "standard."

PII - personally identifiable information. Information that can be traced back to an individual, such as an email address (or probabilistically a hash of an email address), phone number, or combination of IP address, location, and access logs.

Backup: References

Uner, *Bad assumptions lead to bad security*, http://www.embedded.com/design/connectivity/4006466/Bad-assumptions-lead-to-bad-security

Complying with the Personal Information Protection and Electronic Documents Act, https://www.priv.gc.ca/resource/fs-fi/02_05_d_16_e.asp

Jon P. Daries et al, *Privacy, Anonymity, and Big Data in the Social Sciences*, http://queue.acm.org/detail.cfm?id=2661641

Rainer Böhme, The Economics of Information Security and Privacy, ASIN: B00GZZ9WE4

Robert Cragie, ZigBee Security, https://docs.zigbee.org

NSA, *Suite B Cryptography*, https://www.nsa.gov/ia/programs/suiteb_cryptography/