Leveraging Blockchain-based protocols in IoT systems

Konstantinos Kolias, Angelos Stavrou
Computer Science Department
George Mason University

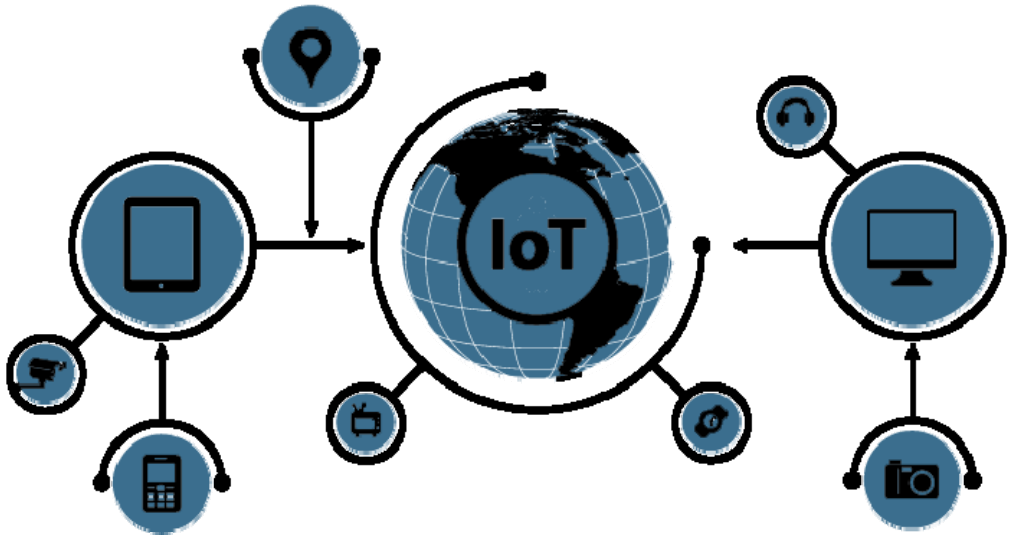
Irena Bojanova, Jeff Voas, Tim Grance
Information Technology Laboratory
National Institute of Standards & Technology

Talk Outline

- IoT Scale & Scope
- Crypto Failures in IoT: A Motivating Use Case
- Understanding the IoT Crypto Needs
- Short Blockchain Primer
- Blockchain in IoT: What are the potential use cases?
- **Why direct use of Blockchain is not practical for IoT**
- **Challenge**: Design practical Blockchain-based protocols for IoT

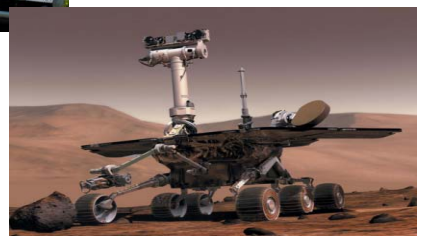
Internet of Things Defined

- Kevin Ashton introduced the term Internet of Things (IoT) in 1999
- Network of devices able to configure themselves automatically
- Human is not the center of the system
- **Motivation**: Better understanding of the environment and response to certain events. Machines are doing better in sensing & reporting on conditions
- **Fact**: Applications of traditional Internet are different than the applications of IoT

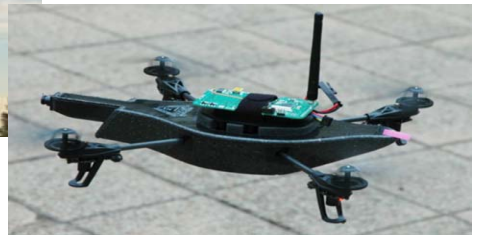


What the Future Holds

Drivables



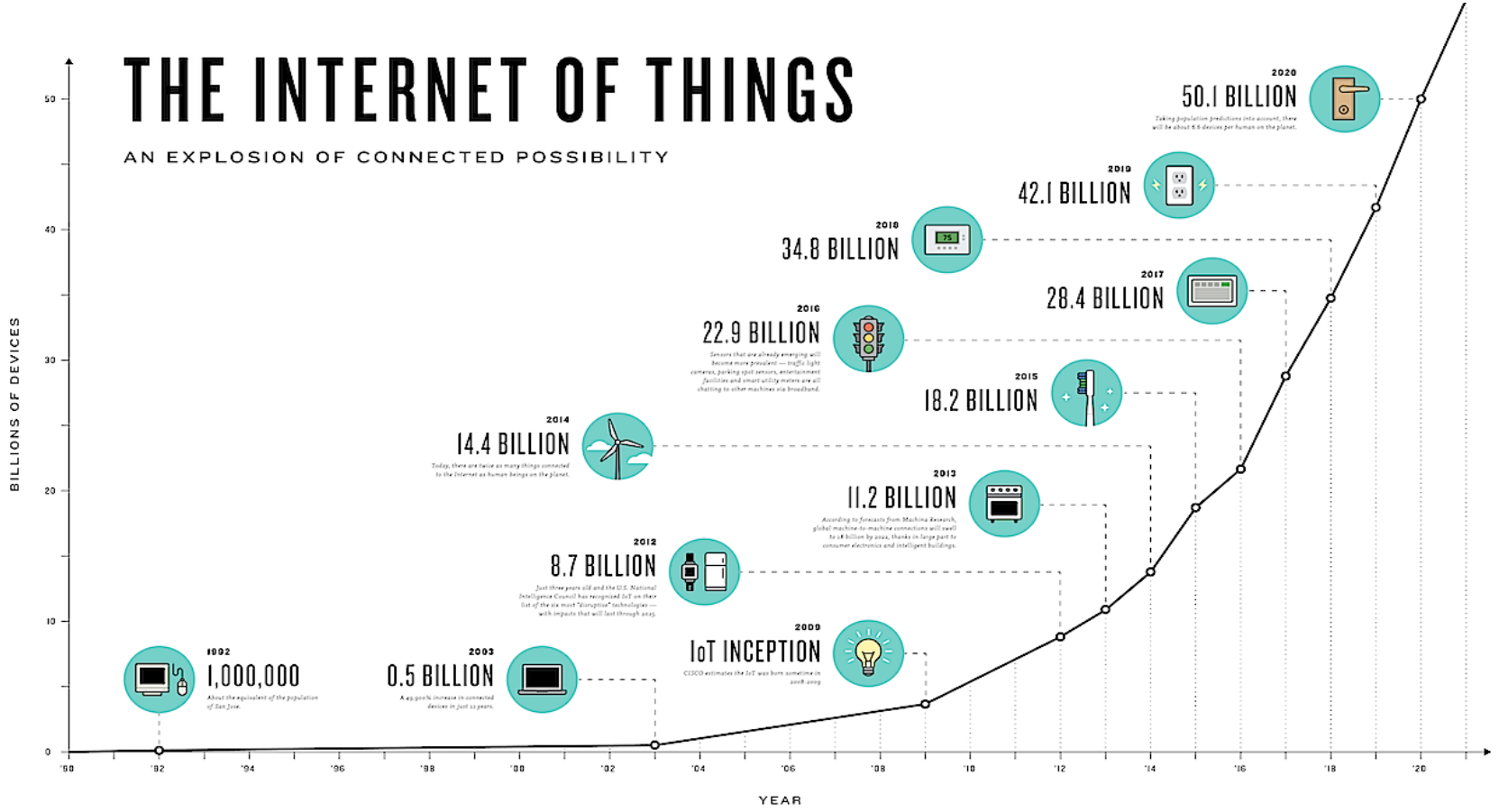
Flyables



Scannables

Wearables

The Growth of IoT

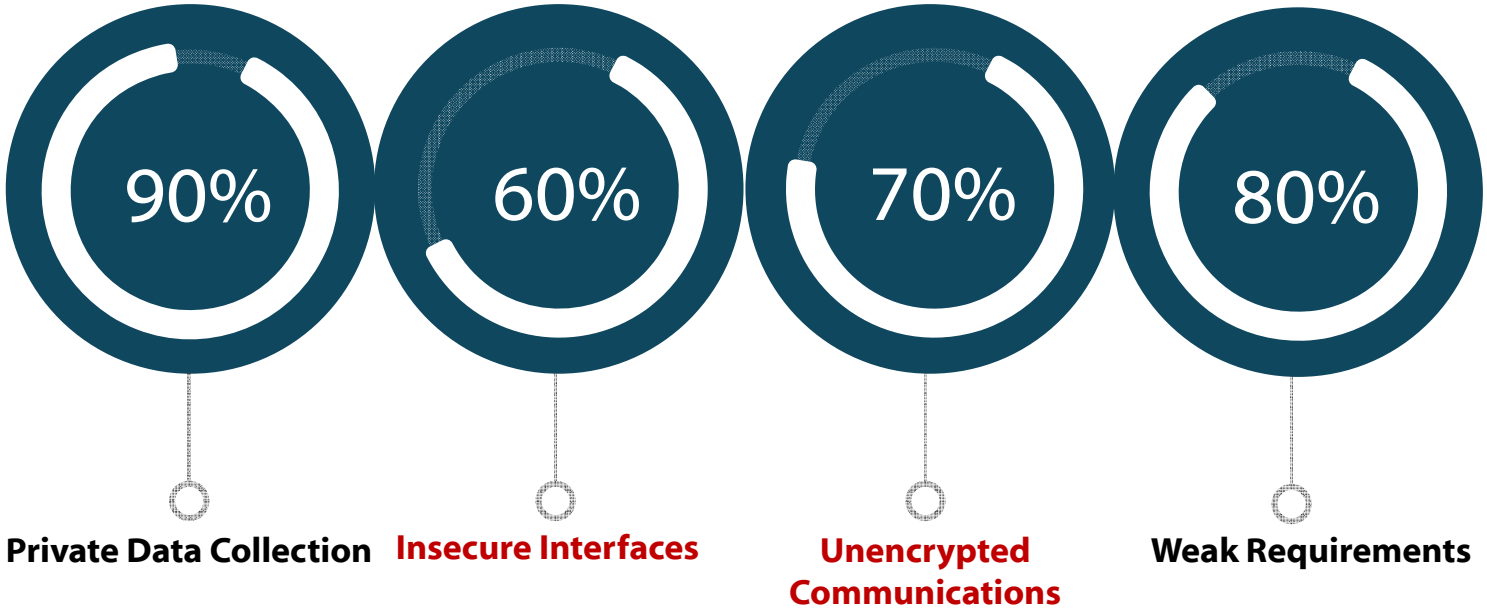


Sectors of IoT Applications

Sensors & Actuators

Connectivity

Common Security Incidents



Top 10 Vulnerabilities (OWASP)

Insecure Web Interfaces
Default accounts, XSS, SQL injection

Inefficient Authentication/Authorizatio
Weak passwords, no two-factor authentication

Insecure Network Services
Ports open, use of UPnP, DoS attacks

Lack of Transport Encryption
No use of TLS, misconfigured TLS, custom encryption

Private Data
Unnecessary private information collected

Insecure Cloud Interfaces
Default accounts, no lockout

Inefficient Mobile Interfaces
Weak passwords, no two-factor authentication

Insufficient Security Configurability
Ports open, use of UPnP, DoS attacks

Insecure Software/Firmware
Old device firmware, unprotected device updates

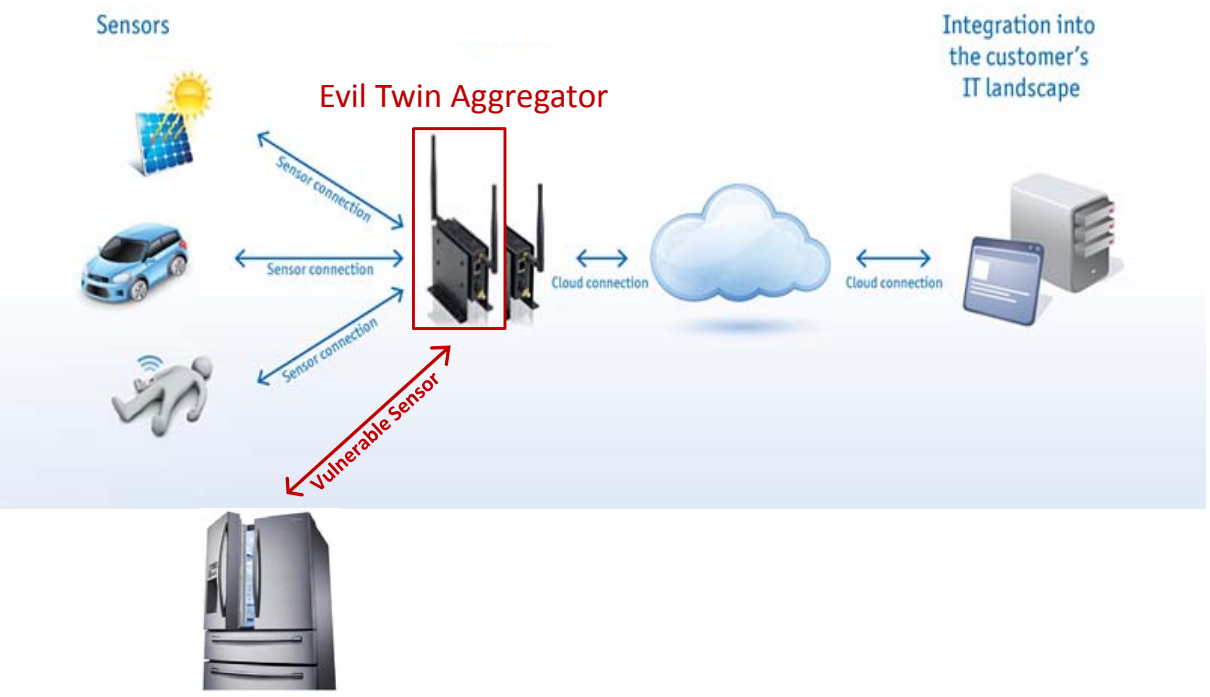
Poor Physical Security
Exposed USB ports, administrative accounts

Use Case: Home Automation

Typical Use Case:

- Sensors and other devices connected to a Home Network
- Devices communicate directly to an “aggregator” gateway

What Can Go Wrong?

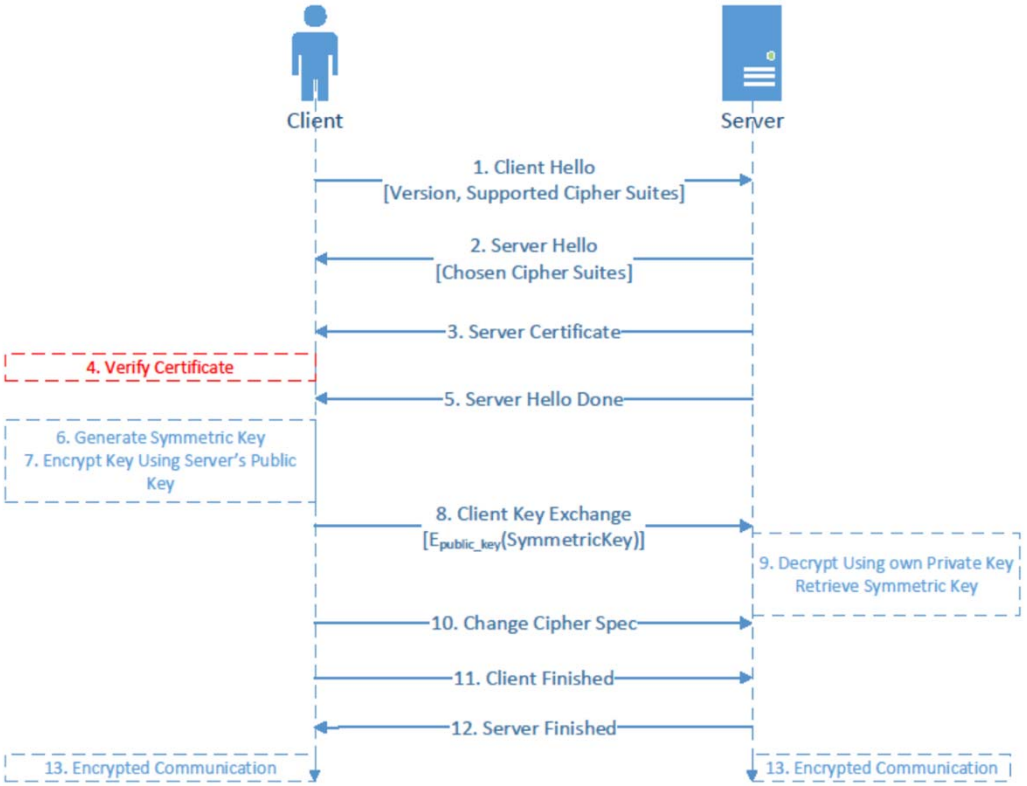


- Attacker introduces a soft-AP or Sensors with the same characteristics
- Custom Crypto
  - Because we can do faster
- No Authentication
- No Encryption

Why Can Go Wrong?

- **Badly Designed System**
  - Platform that cannot handle encryption (SSL/TLS)
  - Cannot communicate securely with standard servers
- **Badly Implemented Crypto**
  - **Example**: Implement "*Custom*" TLS for "*faster*" operation
  - **Challenge**: Make TLS lighter but maintain compatibility
  - **Method**: Remove the "*heaviest*" operations
    - First contender: verification of server certificate
  - **Result**: Minimalistic hardware can support TLS
  - **Gain**: Use of even cheaper hardware
    - Caveat: possible security holes

How Can Go Wrong?



Protocol Hacked at DefCon 2015

- Connects to google calendar to show notes on screen
- Supports SSL/TLS but does not validate server certificates
- Unleash MiM attack
- Steal User's Credentials

Custom Crypto Implementation not a solution

Why Use Custom Crypto?

RSA 1024 Runtime Overhead:

Arduino UNO	16Mhz AVR	==> 12596 ms*	8504 ms#
Arduino Leonardo	16Mhz AVR	==> 12682 ms*	8563 ms#
Arduino Mega	16Mhz AVR	==> 12596 ms*	8504 ms#
Arduino Due	84Mhz ARM	==> 1032 ms*	
Arduino Yún	16Mhz AVR + 400Mhz MIPS	==> 707 ms*	
Intel Galileo	400Mhz x86	==> 192 ms*	

* these numbers are based on a 100% C implementation

these numbers are based on mixed C/AVR assembly implementation

Some of the traditional Crypto is too “expensive” for embedded devices

https://evothings.com/is-it-possible-to-secure-micro-controllers-used-within-iot/

Survey of Crypto Support in IoT

Brand	Name	CPU	Frequency	Sram	Flash	Crypto Acceleration	Energy	Public Key Encryption
Belkin	WeMo Switch	Ralink RT5350F (MIPS)	360 MHz	32MiB	16MiB	No	Wall socket	Yes
Samsung	Smarthings Hub	PIC32MX695F-512H 32 Bit	80Mhz	128KB	512K	No	Wall socket/Battery	Yes
Nest	Thermostat	Texas Instruments AM3703CUS Sitara (ARM Cortex A8)	1Ghz	512Mb	2Gb	Yes	Wall socket	Yes
LIFX	Color 1000	Kinetis K22 (ARM Cortex-M4)	120Mhz	128KB	512K	No	Wall socket	No
Amazon	Echo	Texas Instruments DM3725CUS100 (ARM Cortex A8)	1Ghz	256MB	4GB	Yes	Wall socket	Yes
Philips	Hue Lights	ST Microelectronics STM32F217VE (ARM Cortex-M3)	120Mhz	128KB	1MB	Yes	Wall socket	Yes
Philips	Hue Lights (Bulb)	STM32F100RBT6B (ARM Cortex-M3)	24Mhz	8KB	128KB	No	Wall socket	No
Nest	Smoke/Carbon Alarm	Freescale SCK60DN512VLL10 custom Kinetis K60 (ARM Cortex M4) + Freescale SCKL16Z128V (ARM Cortex M0)	100 Mhz+48Mhz	128KB	512K	Yes	Wall socket/Battery	Yes
Pebble	Time	ST Micro STM32F439ZG (ARM Cortex M4)	180 Mhz	256KB	2MB	Yes	Battery	No
Fitbit	Surge	Silicon Labs EFM32 Giant Gecko (ARM Cortex-M3) EFM32GG395F1024	48Mhz	128KB	1MB	Yes	Battery	No
Fitbit	One	STMicroelectronics 32L151C6 Ultra Low Power (ARM Cortex M3)	32Mhz	16KB	128KB	No	Battery	No

What do we **really** need?

- IoT System Operational Requirements (Empirical)
  - Dynamic but verifiable group membership
  - Authentication & Data integrity
  - Secure against single-node (or small sub-set of nodes) key leakage
  - Lightweight operations in terms of resources
  - Encryption is a plus but not firm requirement
  - Capable of handling sensor “sleep/power-off” periods
  - Handle resource diversity and data of sensors and aggregators

Potential Solutions

- Lightweight Cryptography
  - Security/Cost/Performance trade-off a challenge
  - Size of key material cannot be lowered
  - New crypto designs are promising but not standardized or adopted

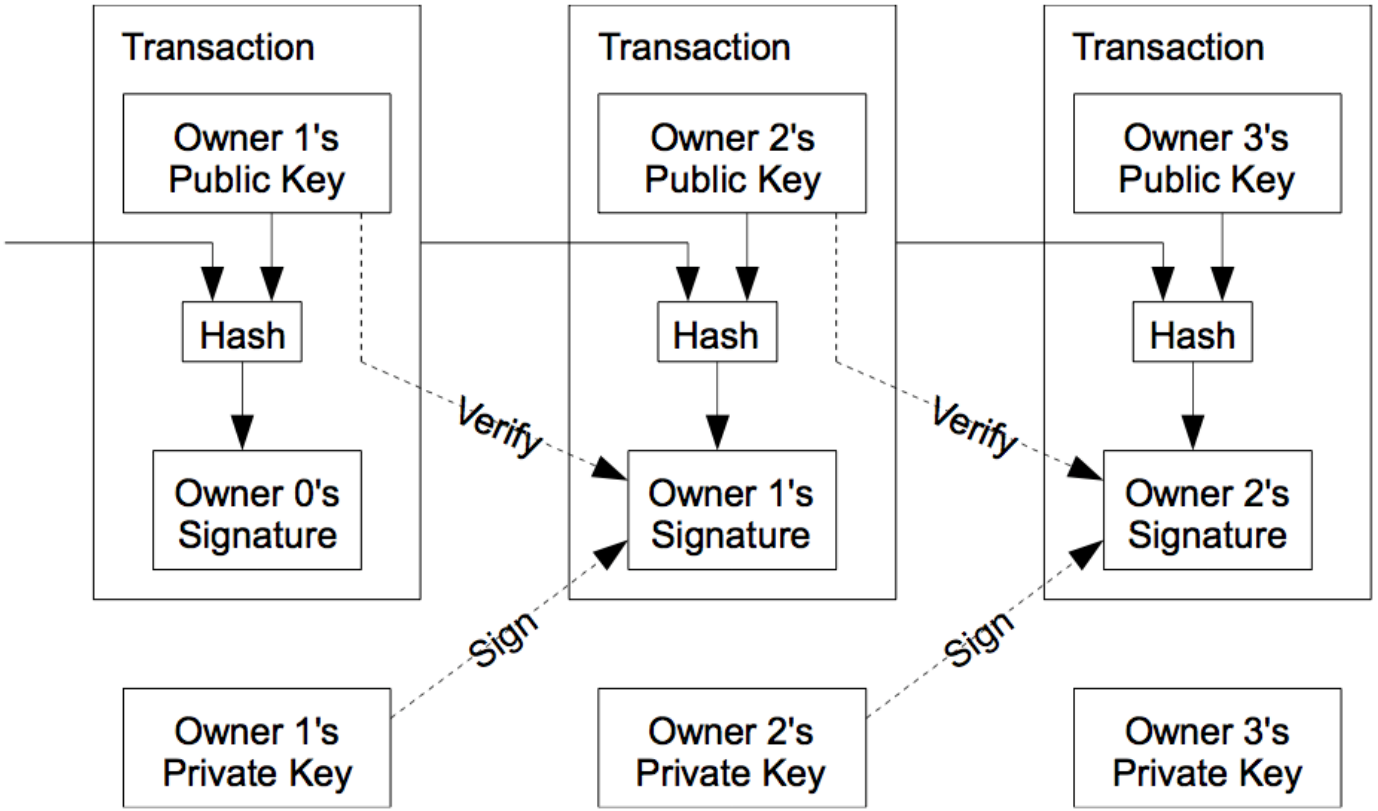
- Can we use existing crypto blocks to meet the requirements?
  - Microcontroller devices support cryptographic hashing
  - Support for AES 256 is pervasive but still expensive
  - Can we leverage Blockchain-based protocols?

Blockchain Primer

Public Distributed Verifiable Cryptographic Leger

- Public
  - All participants gain access to “read”
- Distributed
  - Peer-to-Peer Data Communication, Fully Decentralized
- Cryptographic
  - Digitally signed transactions, proof-of-work limits rate of input
- Ledger
  - Verifiable Transactional Database

Blockchain Primer



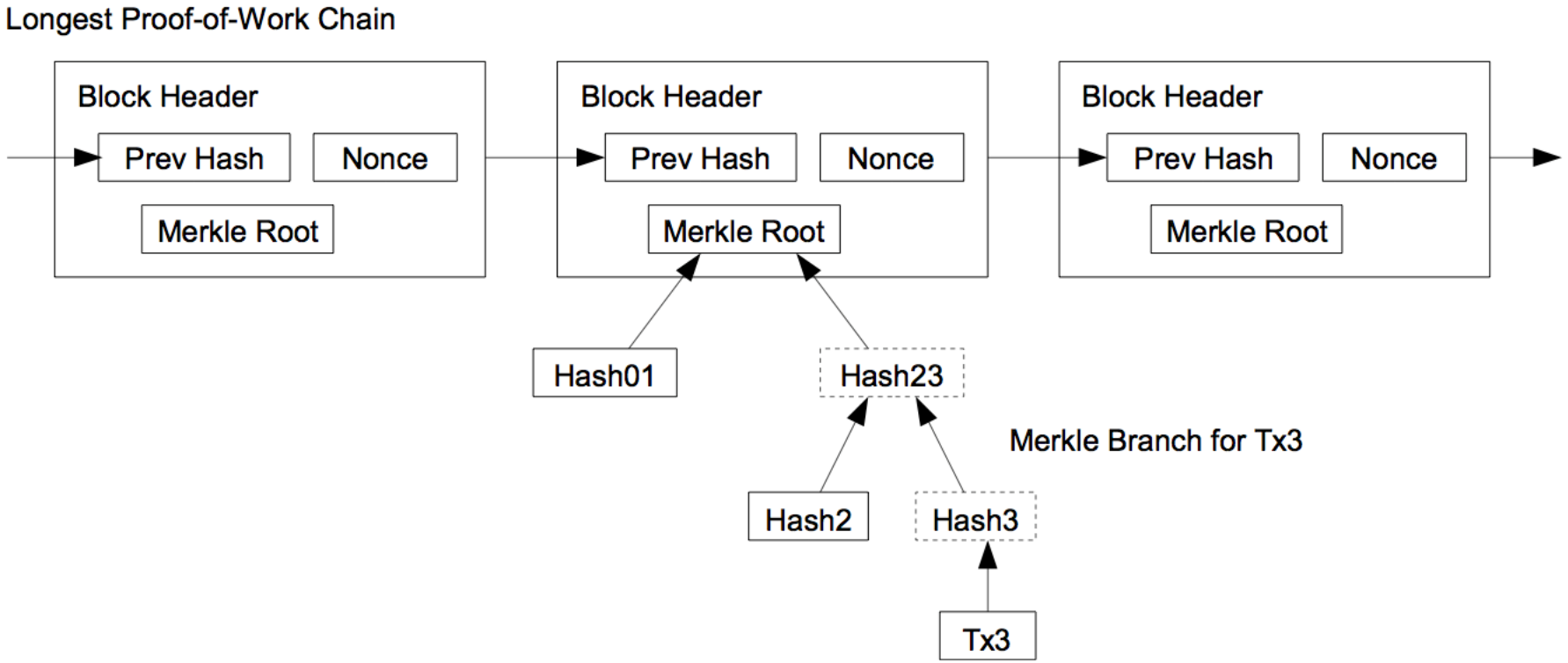
Bitcoin: A Peer-to-Peer Electronic Cash System, Satoshi Nakamoto

Blockchain Primer

Blockchain Blocks

- Sequences of signed and verified transactions
- Published and distributed globally
- Magic number, Size
- Header
  - Hash of previous block (chain)
  - Merkle root hash of block
  - Timestamp
  - Target, nonce (mining)
- Number and list of transactions

Blockchain Primer



Bitcoin: A Peer-to-Peer Electronic Cash System, Satoshi Nakamoto

Is Blockchain Directly Applicable in IoT?

Desirable Properties

- Distributed protocol with verifiable transaction history
- Dynamic membership multi-party signatures

Undesirable Properties

- Requires proof of “work”
- Requires PKI
- Size of the Ledger an issue for “small” devices
- Anonymous (unverifiable) Join/Leave operations

What can we do?

Eliminate undesirable properties

- ~~Requires proof of "work"~~

  Requires proof of earlier participation using history

- ~~Requires PKI~~

  Hash-based signatures (or other Merkle-tree schemes)

- ~~Size of the Ledger an issue for "small" devices~~

  Prune and Compress Ledger. Maintain only device-relevant transaction ledger when device is too resource constrained

- ~~Anonymous (unverifiable) Join/Leave operations~~

  Group signatures using pre-shared group Key(s)

Hash-Chain

One-time hash passwords (Lamport 1981):

- Client generates iteratively a list of hash values (in reverse order of index).

$$
\begin{aligned}
z_\ell &\leftarrow \{0,1\}^n \\
z_i &\leftarrow h(z_{i+1}) \quad \text{for } i \in \{\ell-1, \ell-2, \ldots, 0\}
\end{aligned}
$$

- $z_0 = h(z_1) = h(h(z_2)) = \ldots$ is the "public key"
- Keys are revealed in opposite order, starting from z_1
- Verification of z_i: starting from z_i verify, if z_0 is indeed i-th hash
- Keys can be used only once!

Hash-Chain: PreImage Path

Lamport's one-time-password scheme has either

- $O(\ell)$ storage (whole chain retained) or
- $O(\ell)$ preimage generation time (only z_ℓ retained).

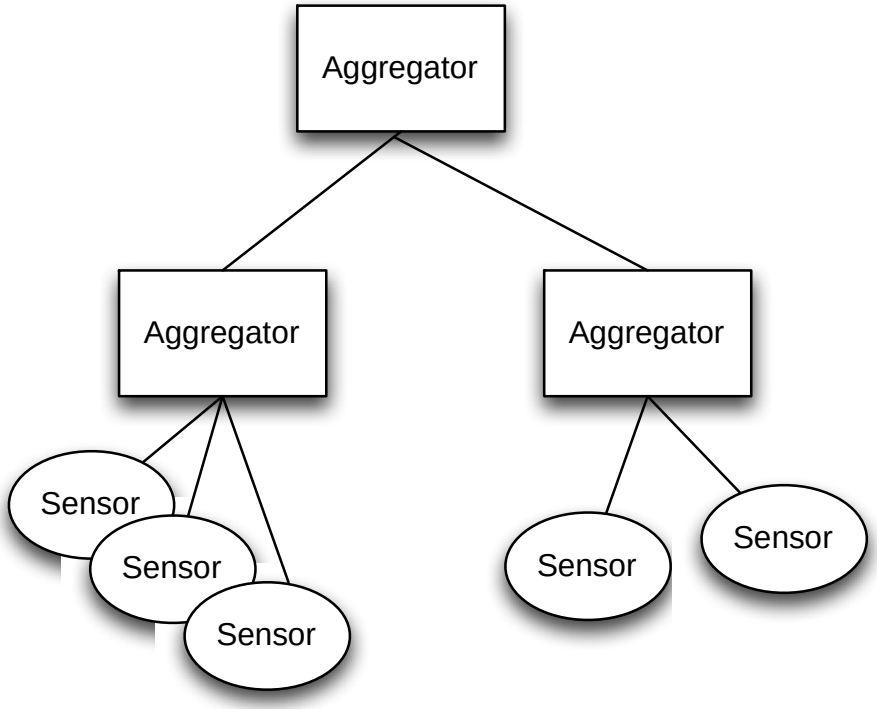
Both extremes are not exactly efficient.

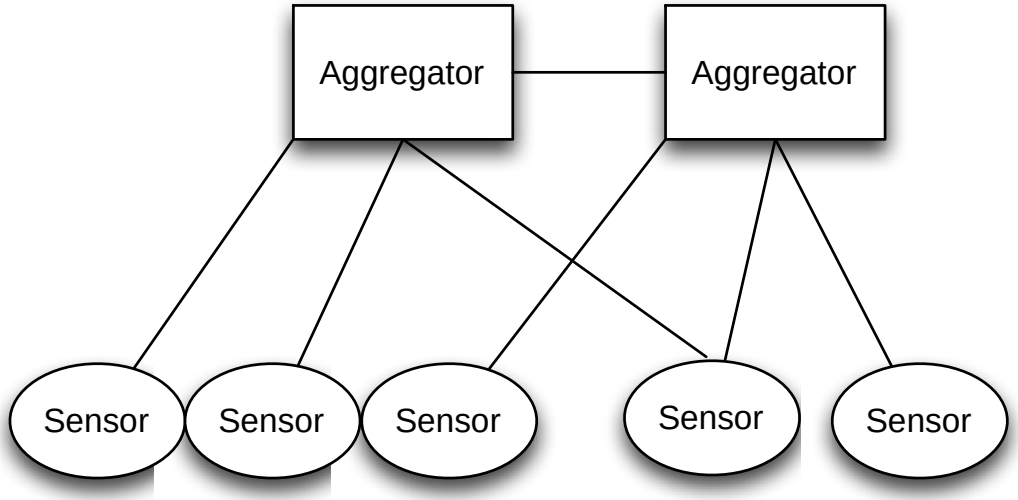
Naive optimization: mark few elements with "pebbles", retain values and use as starting points. If N pebbles are evenly distributed then the worst case is $O(\ell/N)$ hash calculations per key.

Jakobsson (2002): traversal algorithm which amortizes $h()$ calculations. $O(\log \ell)$ memory and $O(\log \ell)$ hashing steps to output a key (preimage).

Pebbles are placed at positions $2^j, j = 1..\lfloor \log \ell \rfloor$; preimages are extracted from left. If a pebble is reached it jumps next to another, and leftover calculations at each step are used to move it gradually into position between neighbors.

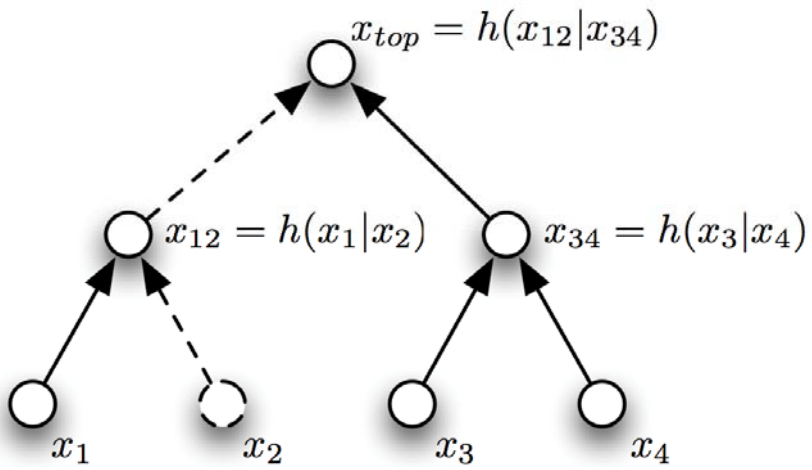
Typical IoT Aggregation Networks

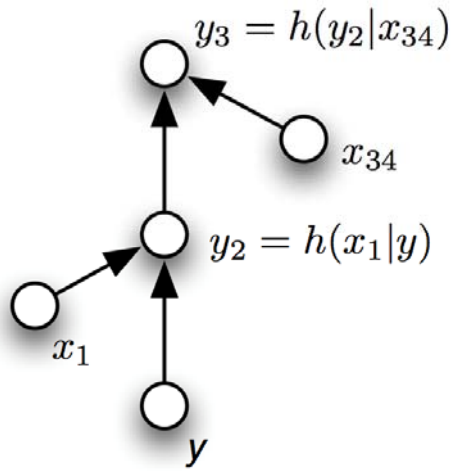




Blockchain-based Protocol for IoT?

We suggest a Blockchain-based protocol that uses the following blocks:



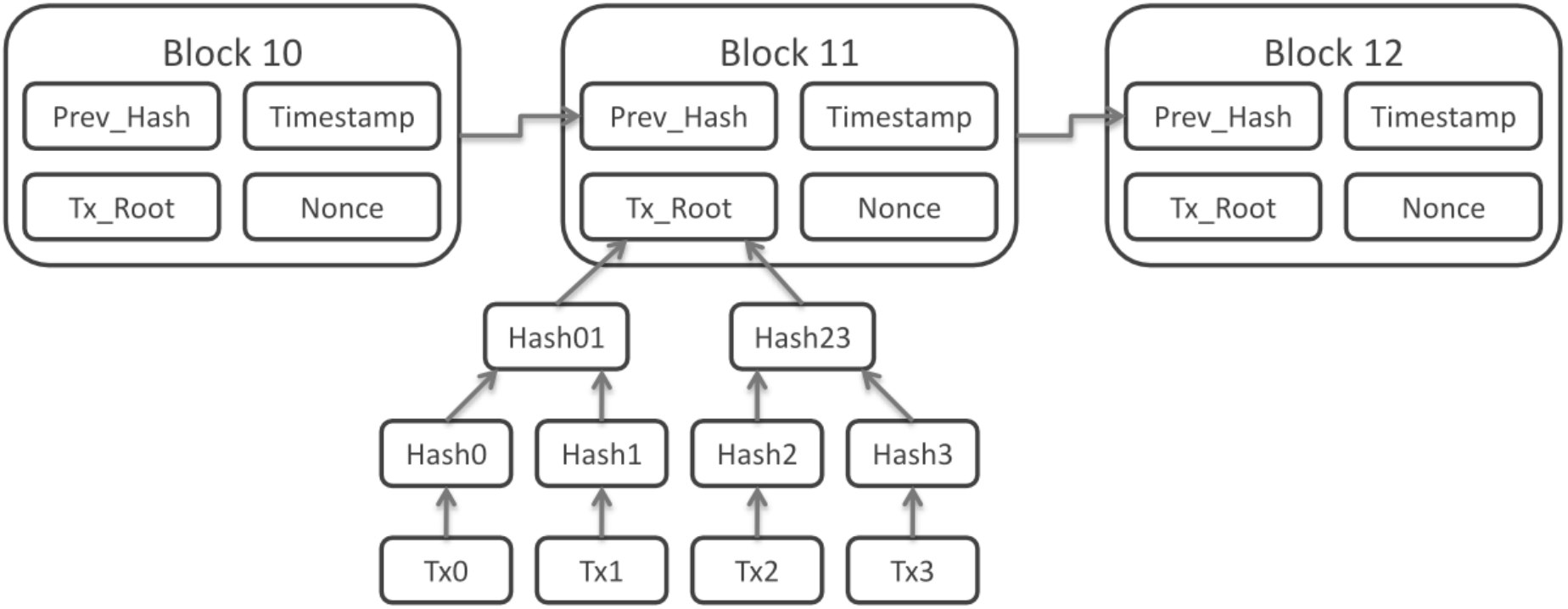


$$x_i = H(Data \,\|\, K_G \,\|\, H(z_i)^n), H(z_i)^{n-1}$$

$$H = Hash,\ K_G = group\ Key,\ z_i = sensor\ i\ "public\ key"$$

Blockchain-based Protocol for IoT?

We suggest a Blockchain-based protocol that uses the following blocks:



[Illustration by Matthäus Wander (Wikimedia)]

Does the Scheme Meet the Requirements?

- IoT System Operational Requirements (Empirical)
  - Dynamic but verifiable group membership
  - Secure against single-node (or small sub-set of nodes) key leakage
    - Only Aggregators can add nodes by issuing a group Key
    - Can be done using Symmetric Encryption or a Hash Chain
    - Node is verified both by group key AND by participation history
    - To add a node, an adversary will have to:

      a) Compromise the group key
      b) Issue an “add node” transaction
      c) Add a sensor node

    - Shape of the tree shows “additions” and “removals” of nodes over time

Does the Scheme Meet the Requirements?

- IoT System Operational Requirements (Empirical)
  - Authentication & Transaction integrity
    - Nodes and transactions are authenticated using the group key and the node Lamport signatures
    - A node uses his Lamport public key to validate inserted DATA, transmits DATA to aggregator(s)
  - Lightweight operations in terms of resources
    - Operations can be lightweight for sensors. Aggregators have more resources
  - Encryption is a plus but not firm requirement
    - No need for encryption

Does the Scheme Meet the Requirements?

- IoT System Operational Requirements (Empirical)
  - Capable of handling sensor “sleep/power-off” periods
    - Nodes can re-authenticate using their knowledge of historical transactions proving their membership specific historical transactions using **predecessors** for Lamport Signatures

$$x_i = H(Data \,||\, K_G \,||\, H(z_i)^n), k, H(z_i)^{n-k}$$

$$where\ n-k\ is\ smaller\ than\ the\ last\ signature\ \ from\ i$$

  - Handle resource diversity and data of sensors and aggregators
    - Different nodes store different portions of the ledger
    - Aggregators fully, others partial

Conclusions

- IoT Scale, Vendors, Technologies increase exponentially
- IoT Devices will always have diverse capabilities & Resources
- Use of Cryptography is done without clear understanding of the implications
- No Current Standards for Lightweight cryptography

- Blockchain inspired protocols combined with new chryptographic primitives might be the path forward

Questions?