CHAPTER 1

Conceptualizing the Secure Internet of Things

In this chapter we relate several iconic attacks on cyber-physical IoT systems to illustrate the clever ways attackers are able to achieve their objectives. The physicality of cyber-physical systems and resource limitations of constrained IoT devices present new challenges, both for attackers and systems designers. This chapter explores security trade-off consequences resulting from design decisions aimed at reducing device cost. We advocate more enlightened perspectives that consider the value of the device in terms of the broader network and system value. The security front line often is a constrained device requiring world-class security capabilities such as hardware underpinnings for cryptography, integrity protection, storage, and attestation. Devices that don't provide the basic building blocks of security are the *weak links* in the system – which systems designers aim to quarantine.

S. Cheruvu et al., *Demystifying Internet of Things Security*,
https://doi.org/10.1007/978-1-4842-2896-8_1

The BadUSB Thumb Drive

In 2014 Karsten Nohl and Jacob Lell presented proof-of-concept malicious software at Black Hat USA 2014[1] that demonstrated how USB is fundamentally broken. The malware infects USB firmware rather than simply placing malicious applications on the storage area. USB firmware is trusted by most every USB controller to behave properly, as defined by the USB Consortium specifications.[2] However, as long as USB firmware works within the framework defined by the standard, malware can cause the USB controller to give the USB firmware unintended access to the host computer. This is unfortunate as the lack of attention given to security implies a potential for exploits that includes key-logging, privilege escalation, data exfiltration, identity and access misdirection, session hijacking, and denial-of-service.

Karsten and Jacob not only published their findings but also published the malware on an open source repository known as GitHub.[3] This means virtually anyone can construct their own USB attack device and even improve upon the original design. There have even been "how-to" publications[4] that step the reader through the process, making it easier than ever for even those without prior knowledge of USB architecture and implementation to successfully build an attack device.

Subsequently, the "maker community"[5] has picked up on BadUSB by creating a business around hardware platforms that have BadUSB preintegrated called "MalDuino"[6] – a play on words involving a popular

[1] www.blackhat.com/us-14/speakers/Karsten-Nohl.html
[2] www.usb.org/home
[3] https://github.com/brandonlw/Psychson
[4] https://null-byte.wonderhowto.com/how-to/make-your-own-bad-usb-0165419/
[5] https://en.wikipedia.org/wiki/Maker_culture
[6] www.indiegogo.com/projects/malduino-badusb-arduino-usb#/

"maker" platform named Arduino.[7] Using MalDuino as a development platform, it is possible for attackers to integrate other interesting malware designed to further infiltrate the victim computer or network. Often an attacker exploits a vulnerability in order to stage an attack on another vulnerability. Attack lethality can be amplified by linking several exploits that expose larger attack surfaces and allow the attacker to marshal more resources for the next attack. An attack that began as a compromise of something without network connectivity may morph into a compromise of resources with network connectivity – that broadens the attacker's reach and lethality.

Air-Gap Security

Some of the most secure networks rely on "air-gap" security as a way to prevent the spread of malware through interconnected networks. Air-gap is an isolation technique that ensures there are no wired or wireless connections between a highly sensitive network and one that is commonly accessible to everyone, such as the Internet. The security principle behind air-gapping is to establish physical isolation such that in order to move information back and forth between the secure network and other networks, there needs to be a mechanical system in place – euphemistically termed a "sneaker-net." The idea is that only trustworthy people would have physical access to the air-gap and would follow appropriate security practices and procedures that ensure sensitive networks do not fall victim to the many attack scenarios found on public networks.

However, air-gaps rely on the use of electronic media to "sneaker-net" information to and from air-gapped networks. This often involves the use of USB connected peripherals. The assumption is that a device that isn't

[7] www.arduino.cc

capable of sending or receiving electromagnetic emanations is safe to cross an air-gap. The fallacy of this assumption, of course, is they are not safe as evidenced by BadUSB.

Air-gap security has a significant usability downside in that it is costly to deploy, doesn't scale well, and isn't forward looking. The next generation of industrial IoT looks to other network security mechanisms such as VLANs that segment networks that isolate manufacturing equipment behind routers, static/dynamic whitelisting, and zoning/quarantining using network firewalls.

The lesson learned by air-gap security is that attention to usability cannot be ignored. Security mechanisms must be designed with all other system requirements taken into consideration to find the security mechanisms that optimize trade-offs.

Stuxnet

"Stuxnet"[8] is the name given to a malware found to have successfully infiltrated a top security nuclear research facility in Iran in June 2010. The Natanz uranium enrichment facility employed air-gap security mechanisms due to the safety critical aspect of the uranium enrichment process. Furthermore, uranium enrichment processes rely on SCADA (Supervisory Control And Data Acquisition) systems that are commonly used for industrial control because of their ability to precisely control physical machinery and remain resilient in the face of physical system failures, but also incorporate popular information messaging protocols such as MQTT (Message Queuing Telemetry Transport), AMQP (Advanced Message Queuing Protocol), and DDS (Data Distribution Service).

[8] www2.cs.arizona.edu/~collberg/Teaching/466-566/2012/Resources/presentations/2012/topic9-final/report.pdf

SCADA systems may use programmable logic controllers (PLCs) and a variety of other sensors and actuators that can be customized to suit the needs of the particular mechanical operations in a plant or factory. PLCs often have USB interfaces for uploading the control logic executed by the PLC, but also support serial bus interfaces and protocols such as Modbus or 4-20mA current loops that transfer information reliably and with less wiring and setup. Unfortunately, these techniques did not anticipate security or are simply incapable of stopping attackers who have physical access.

Stuxnet employed a variety of techniques, some seemingly designed as alternative attack strategies in case some other strategy failed to pan out. Among them included a strategy to propagate the Stuxnet malware using Internet "Futbol"–themed web sites. Ultimately, Stuxnet found a way to program USB thumb drives that were used to update PLCs used for uranium enrichment centrifuges.

Stuxnet ultimately was able to cause physical damage to centrifuges by working within the tolerance specifications of the control system, but stealthily controlling the centrifuges to spin faster than usual for longer than usual or to adjust the rate of acceleration and deceleration in ways that exceeded the mechanical designer's expected use case scenarios.

Although there still remains controversy over who created Stuxnet and whether it was targeting Iranian nuclear enrichment or not, statistics gathered by Symantec[9] suggest there were unintended consequences in the form of compromise to "friendly" or untargeted installations. While the majority of infections, 58.85%, occurred in Iran, the remaining 41.15% affected other countries; 8.31% occurred in India, 18.22% in Indonesia, and 1.56% in the United States. 13.05% occurred in other parts of the world.

Stuxnet is interesting because it demonstrates the possibility for information systems to cross over to operational systems in such a way that physical systems, infrastructure, the environment, and ultimately human

[9] "W32.StuxNet". Symantec. 17 September 2010. Retrieved 2 March 2011.

life can be harmed using only commonly available inexpensive electronics and software.

It marks the fusion of Information Technology (IT) with Operational Technology (OT). The acronym Internet of Things (IoT) takes on an additional and apropos meaning of Informational and Operational Technology (IOT).

Designing Safe and Secure Cyber-Physical Systems

The preceding attack scenarios suggest we need to revisit past assumptions that electronic equipment is "secure" because of physical and air-gap isolation is incorrect. The presence of electronic "things" may be sufficient for some form of "networking" to be implemented involving the exchange of electronic things and therefore the exchange of malware that can transform to take advantage of different attack vectors. A more enlightened view of IoT may be the idea that the interconnection of all networks – including the exchange of physical things containing information – is the Internet.

Applying this view of the Internet, there are two additional layers to classes of computers[10] that historically fit into three categories: (1) cloud servers largely composed of mainframes and super computers; (2) mini computers such as workstations and department or team servers; (3) microcomputers such as PCs, laptops, tablets, and smartphones.

IoT more commonly refers to a fourth layer consisting of smart cars, drones, wearable computing, and pervasive computing. However, a fifth layer consists of everything else that is electronic including USB thumb drives, cameras, MEMS,[11] smart construction materials, and "Smartdust."[12]

[10] https://en.wikipedia.org/wiki/Classes_of_computers
[11] https://en.wikipedia.org/wiki/Microelectromechanical_systems
[12] https://en.wikipedia.org/wiki/Smartdust

The layering of technology has many non-security related benefits, but technology layers can present new security challenges. The interaction between layers is often not well understood or clearly specified. This can result in exploitable security weaknesses. Security analysis and design scope should therefore be expanded to include these other layers. Another aspect of security analysis is to determine the "attack surface"[13] – the environment or sum of all points where an unauthorized user can try to extract information or inject control not anticipated by system designers. A basic tenant of security design is to keep attack surface small to limit the potential for unanticipated interactions.

The attack surface of IoT can be viewed as a pyramid (Figure 1-1) where the number of possible interactions is a function of the number of possible "things." Although cloud servers process large workloads, there are only a few cloud servers in terms of possible points of interaction. Cloud servers expose commonly used web interfaces that do largely a small set of things, but in large volumes.

The IoT pyramid also illustrates the importance of defense in depth as nodes at opposite ends of the pyramid tend to be separated by routers, gateways, and other networking equipment that can be repurposed as security enforcement. Network segmentation reduces the effective attack surface by artificially isolating IoT nodes.

Intel predicts there will be 200 billion "objects" by the year 2020.[14] An object is anything that is "smart" – that is anything that has a microcontroller of some kind. If we consider relative population of objects across a five-layer IoT pyramid, the number of objects is roughly exponentially larger in the layer below and the layer above is exponentially smaller. A simple calculation showing exponential distribution across five layers reveals approximately 1.4B objects at the top layer, 1.9B objects at

[13] https://en.wikipedia.org/wiki/Attack_surface

[14] www.intel.com/content/www/us/en/internet-of-things/infographics/guide-to-iot.html

the second layer, 3.6B objects at the third layer, 13.4B objects at the fourth layer, and an amazing 179B objects at the fifth layer.

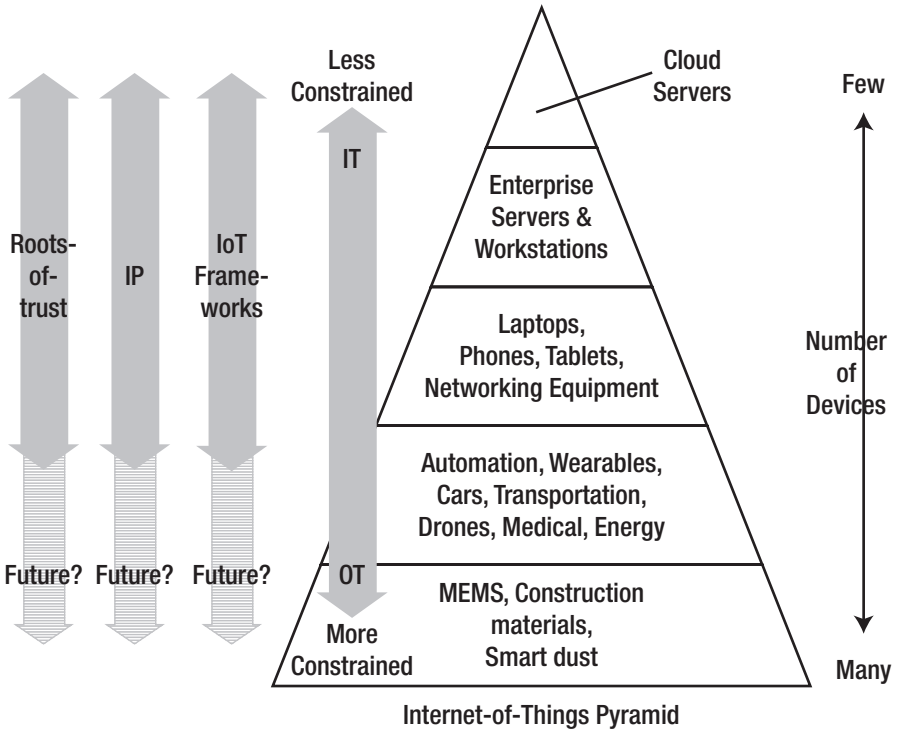

Figure 1-1. *Internet of Things pyramid*

Amazon had around 2M cloud servers and 1M customers in 2014.[15] Alibaba had 765,000 customers in June 2017.[16] Microsoft, IBM, Google, and others also have cloud service offerings that contribute to an estimate in terms of number of cloud server objects that could very well be in the 1B range by 2020.

[15] www.bloomberg.com/news/2014-11-14/5-numbers-that-illustrate-the-mind-bending-size-of-amazon-s-cloud.html

[16] https://intl.aliyun.com/about

In 2015, it was estimated there were 2.6B smartphones[17] and predicted to be 6.1B by 2020. There were about 2B PCs and laptops in 2014.[18] Our simple calculation suggests there would be 3.6B objects at layer 3 – off by a factor of 1.5 or 2, but still in the ballpark.

Even with conservative estimates, these account for only 10B of the 200B forecasted. If layer 4 accounts for 15B objects, that leaves 175B objects unaccounted for at layers 1–4. These estimates suggest, by far, that layer 5 represents the largest attack surface. That suggests there will be many more "Stuxnet"-like attack scenarios going forward. It also suggests mitigation of these attacks will be countered by additional security capabilities being applied to layer 4 and layer 5 objects.

Security capabilities often are required across a spectrum of technologies ranging from hardware to system software to application layers. IoT security also embraces network security and distributed computing security techniques. The potential exists to substantially increase the overall cost and complexity of security functionality for IoT systems. As security professionals anticipate the role security should play given an Internet of 200B connected things, security interoperability and standards are increasingly needed at layers 4 and 5 of the IoT pyramid. This includes the need for hardware-roots-of-trust (specially hardened components in hardware that resist many common vulnerabilities), common networking layers, and common IoT framework and object models. Consolidation of technology choices has a desirable consequence of allowing more security functionality to fit into constrained computing environments.

[17] https://techcrunch.com/2015/06/02/6-1b-smartphone-users-globally-by-2020-overtaking-basic-fixed-phone-subscriptions/

[18] www.reference.com/technology/many-computers-world-e2e980daa5e128d0

Constrained Computing and Moore's Law

In 1965 Gordon Moore made a prediction that computing would dramatically increase in power, and decrease in relative cost, at an exponential pace.[19] The computing industry perspective historically has been one that continually looks for "power-hungry" applications that can soak up the predicted CPU cycles. Ironically, that pursuit has led the computing industry to push the IoT pyramid higher and wider, but only recently has realized a frontier in the form of many (billions) chips that are power constrained. In constrained computing environment, the application that runs on a chip is quite small and functionally is relatively simple. The path to realizing Moore's Law is through the number of chips – increasing in number exponentially.

Rather than consolidating more workloads on increasingly more powerful computers, constrained computing is about distributing workloads across hundreds, thousands, and even millions of nodes. Distributed applications are described more in terms of conceptual notions of computing such as "pervasive," "mobile," "intelligent," "autonomous," "perceptual," "virtual," "emotional," and "augmented." These adjectives describe properties of computation that are realized in large part due to distributed computing that bridges the five layers of the IoT pyramid.

Constrained computing dynamics optimizes the computing environment to fit specialized functions. The function is unique to sensor/actuator capability. Hence, enhancing a distributed application may be realized by adding constrained nodes as well as by adding more powerful nodes or by moving compute-intensive operations to edge servers.

These dynamics aim to provide more flexibility at the lower layers of the technology stack by using, for example, virtualized PLCs where manufacturing equipment can be consolidated into more powerful gateways running multiple, redundant servers that are less expensive to

[19] www.intel.com/content/www/us/en/silicon-innovations/moores-law-technology.html

operate than deployments of multiple less powerful devices. Non-mission critical sensing over wireless technologies is an important trend where the cost driver is low-power sensing solutions (sometimes retrofitted with brownfield sensors and actuators) designed to operate without replacement over many years. Deployment models such as this don't anticipate having extra watts for security processing.

Security however follows a counterintuitive cost model (Figure 1-2) where the motivation is to make nodes more powerful – so they can perform security processing that applies security consistently across all nodes. Workload consolidation, data consolidation, and redundancy result in the deployment of additional nodes or more powerful nodes – all requiring consistently strong security capabilities and hardening.

In the Stuxnet scenario, attackers were able to connect USB thumb drives to air-gapped process control networks because the USB thumb drive didn't have strong cryptography and authentication protections built into the IO control subsystem. Such sophisticated security operations are often determined to be "too costly" to justify bills-of-material cost constraints typically expected in "mass market" products.

Security functionality overhead for layer 1–3 systems typically is expected to be 10–15% of the total system cost. These environments are often very capable of supporting a common set of security features, algorithms, and operations such that the goal of having a network of equivalently protected computers is achieved. However, when moving compute into constrained environments, even with the dynamics of Moore's Law, computing power remains constrained. As such, the percentage of overall functionality that is security related vs. non-security related increases. Our estimates suggest that as much as 60% of a constrained environment computer could be focused on performing security-related computation, leaving 40% for application-specific computing. In other words, the "tinification" (the process of removing unused functionality not needed by purpose-built embedded systems) of an application to fit into constrained environments results in the need to preserve more of

the security functionality than the non-security functionality. This leads business decision makers to question the viability of profits in constrained environments. Often these trade-off decisions lead to justification for weaker security, lack of firmware update capability, and no support for hardware root-of-trust architectures. These economic dynamics have led leading security thinkers to suggest the only resolution is through regulation.[20] However, regulation aimed at even the most insignificant of IoT platforms would affect over 170B things – 85% of everything! If regulation happens to have inefficiencies, those inefficiencies would be multiplied 170B times – a cost that could outweigh the cost of smartly applied security.

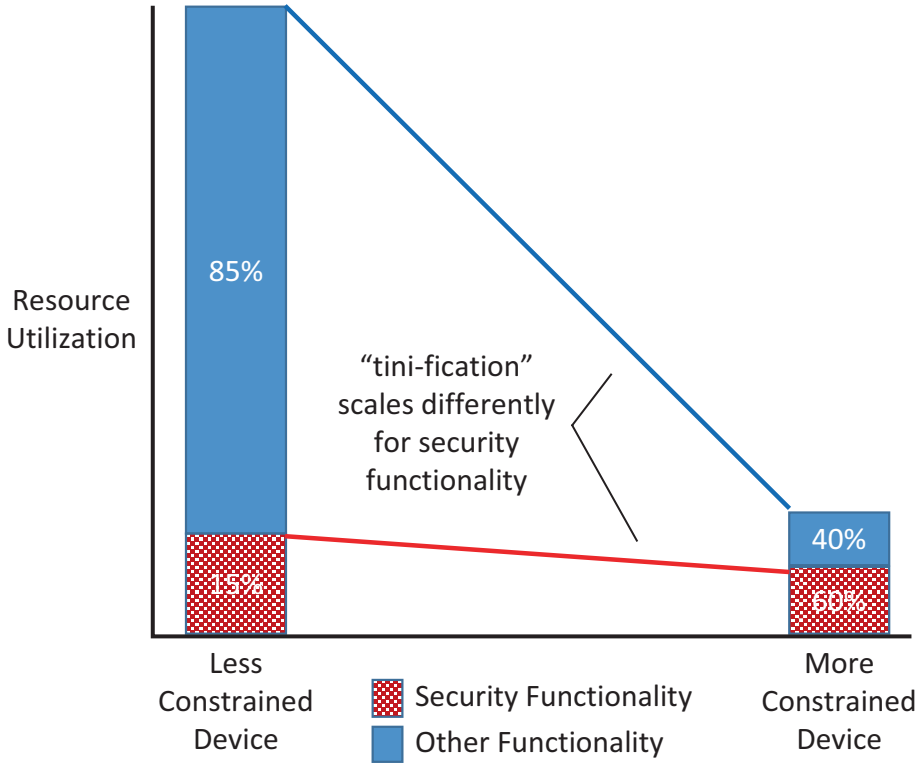

Figure 1-2. *Nonlinear "tinification" of security vs. other functionality*

[20] www.schneier.com/blog/archives/2017/02/security_and_th.html

Trusted IoT Networks and the Network Edge

The Internet of Things is a new term to describe an old concept – connected embedded computing. For as long as there has been electronic control, there has been connected embedded computing. Every dimension of process control and automation is characterized by a flavor of connected embedded control technology.[21] In most cases, process control networks were connected using wires. This is no different a phenomenon for IP networks that first began as Ethernet[22] cable. More recently wireless communications dominate applications where mobility or deployment considerations make using wires infeasible. Nevertheless, the array of wireless networking standards[23] has evolved to take the place of wired equivalents. However, convergence toward a single network protocol remains a promise of IoT which anticipates that IPv6 (Internet Protocol)[24] will become the foundation of IoT networks – and by extension the entire Internet. Nevertheless, there are non-IP protocols that sometimes are included under the umbrella of the IoT buzz word such as Bluetooth[25] and Zigbee.[26] Although these are not technically IP, there are strategies to encapsulate IP over non-IP networks using 6LoWPAN[27] to support larger payloads, compression, and framing that otherwise would not be feasible. IPv6 encapsulation is currently supported with Bluetooth Low Energy (BLE) 5, IEEE 802.15.4, and ZigBee.

The interesting security challenge for encapsulated or bridged networks (Figure 1-3) is the expectation of end-to-end security is often

[21] https://en.wikipedia.org/wiki/List_of_automation_protocols

[22] www.safaribooksonline.com/library/view/ethernet-the-definitive/1565926609/ch01.html

[23] https://en.wikipedia.org/wiki/Comparison_of_wireless_data_standards

[24] www.ietf.org/rfc/rfc2460.txt

[25] www.bluetooth.com/specifications/bluetooth-core-specification

[26] www.zigbee.org/zigbee-for-developers/network-specifications/

[27] https://tools.ietf.org/html/rfc4944

not possible since security applied within one suite of IoT network technology must be mapped, in the clear, to an Internet-based protocol suite. This creates the need for a security appliance, such as a firewall, that maps not only distributed application data but also security semantics and operations. We show a simple security appliance example here. Subsequent chapters provide additional insights into network partitioning, monitoring, and responses facilitated by security appliances.

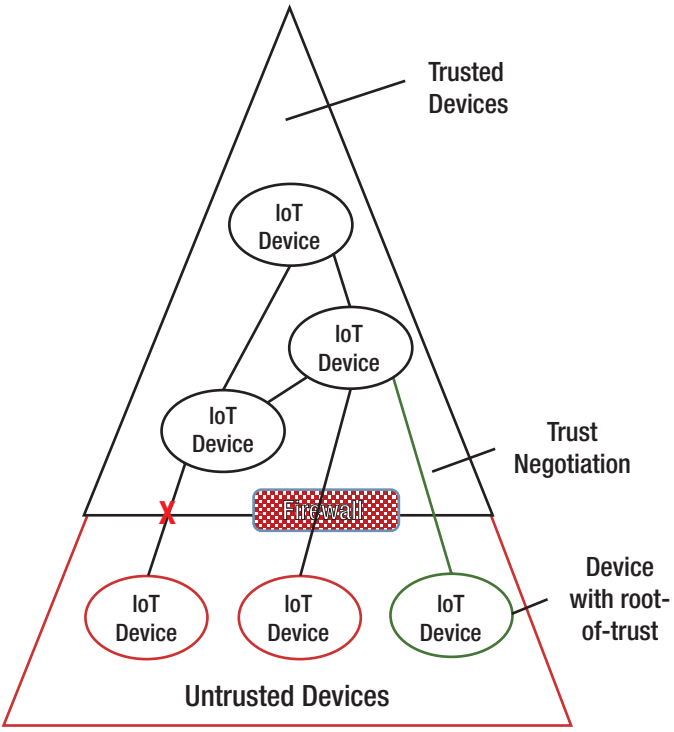

Figure 1-3. *Negotiating trust with IoT devices*

IoT networks are in a constant state of flux forming and re-forming coalitions of devices needed to implement a variety of distributed applications. We use the term "onboarding" to refer to this dynamic. Devices not yet recognized as members of a coalition are considered

"untrusted," while devices already part of the coalition are considered "trusted." Membership in the coalition involves trust negotiation where the device presents evidence of trustworthiness; for example, the device may be equipped with a "root-of-trust" hardened environment containing a manufacturer embedded attestation key. The root-of-trust is designed to meet a set of security features and assurances as a basis for trust. Secure key storage and secure cryptographic operations are important capabilities of a root-of-trust that can be used to implement attestation.

Attestation protocols (Figure 1-4) allow the root-of-trust to prove to a verifier that it is capable of protecting secrets, identities, and data. When an untrusted device is onboarded into a coalition, it first attests to its level of trustworthiness. This allows the attestation verifier to determine if the desired coalition is appropriate or if some other coalition is more appropriate. For example, a coalition of medical devices might expect all coalition member devices to have been approved by a quality control agency and receive a statement of approval that could be included with the attestation exchange at onboarding. If omitted, the verifier might conclude the device hasn't been vetted by the agency and recommend it join a coalition of personal health fitness devices (that don't require agency vetting).

The attestation verifier is a process that operates at a border that separates trusted and untrusted. In practice, these borders are nondescript. They may not align with geographic, topologic, social, or political boundaries. Likewise, such boundary criteria could also be asserted as part of attestation (if combined with additional contextual information), making enforcement of such bounding criteria eminently possible.

Attestation is a form of operational integrity checking that can be pervasive. IoT nodes should respond to changes that might invalidate recent checks and respond proactively by updating integrity profiles and rechecking. If an attack is successful, the attestation check can detect it and respond appropriately.

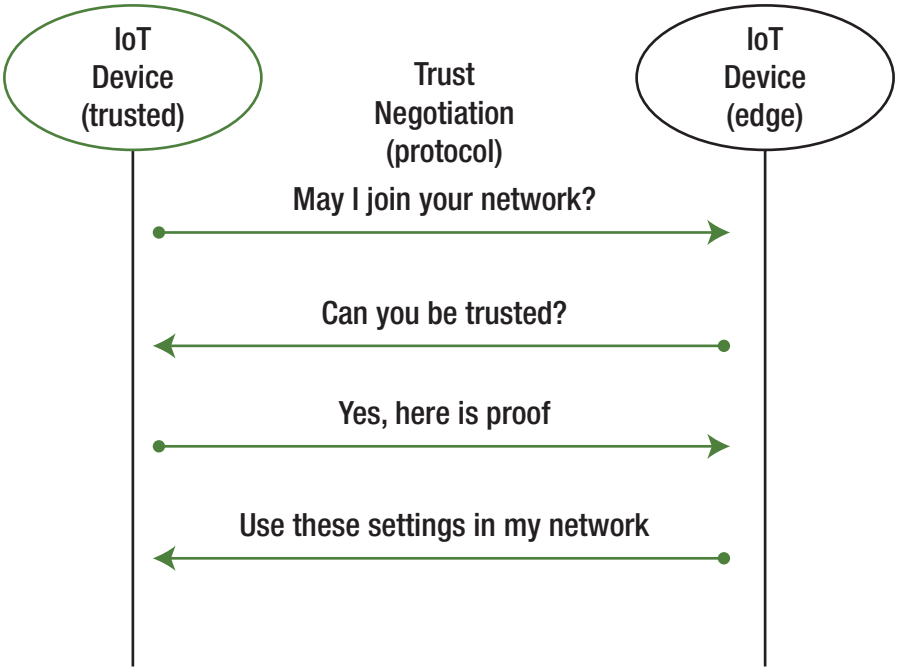

Figure 1-4. *Attestation protocol*

IoT can therefore be thought of as a connectivity graph where certain nodes are simultaneously connected to multiple other coalitions of connected nodes. The connectivity graph reveals relative importance of certain nodes but also relative security and safety risk as more highly connected nodes represent a greater potential for doing harm if compromised or malfunctioning.

Attestation therefore can be thought of as a fundamental capability for anything that is connected. It provides a first-order filter that categorizes IoT devices according to the risk they bring to the established coalition. If we consider all ventures as being composed of a collection of IoT devices, whether they be Smartdust or whether they are cloud servers, the value of the venture is collectively held by the coalition. The introduction of a new IoT device that may have the potential to nullify that value creates the basis for risk-based management approach that relies primarily on attestation and root-of-trust as the primary tools for value preservation and risk management.

An IoT root-of-trust (Figure 1-5) can be constructed in a variety of ways and can vary dramatically in terms of implementation and deployment costs. However, all root-of-trust designs have several minimum capabilities. First the IoT device is partitioned into trusted and traditional functionality. Traditional functionality is everything that isn't essential to satisfying coalition onboarding requirements. An IoT device that can't satisfy onboarding is simply an embedded or stand-alone device. It isn't a "connected" device – at least not a trusted connected device. Trusted functionality is everything else that is needed to satisfy coalition onboarding and is trusted to work correctly.

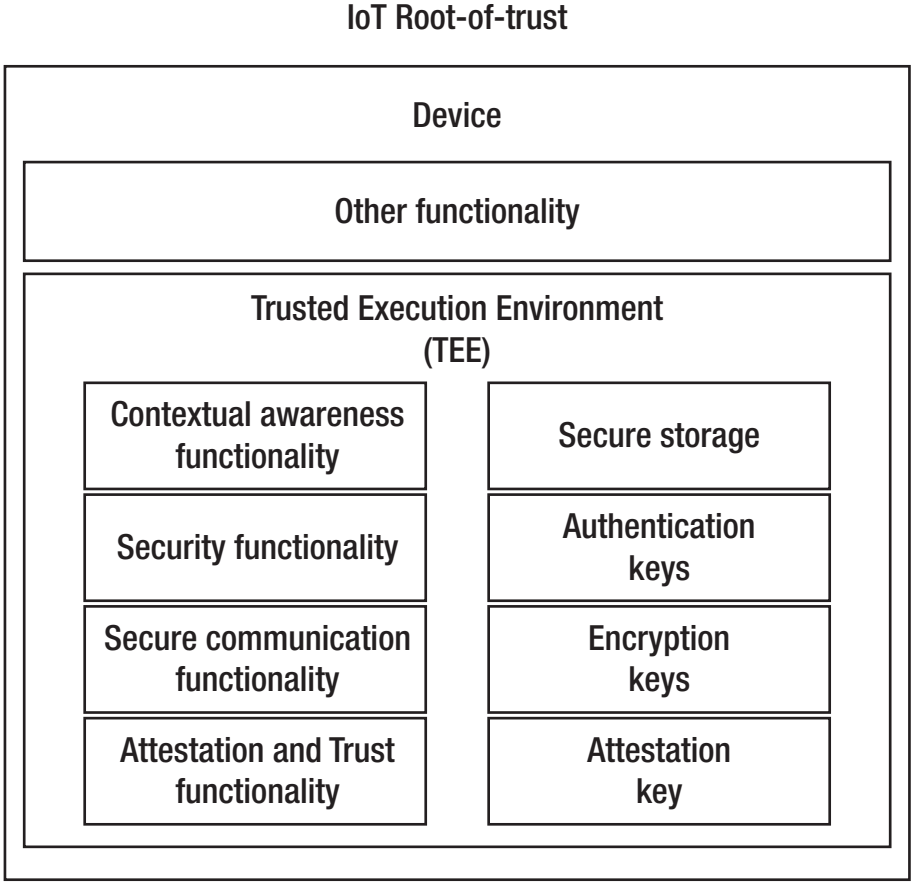

Figure 1-5. *Root-of-trust architecture*

Trusted computing is defined by TechTarget[28] as "Trusted computing is a broad term that refers to technologies and proposals for resolving computer security problems through hardware enhancements and associated software modifications." Wikipedia[29] defines a trusted system as "… a system that is relied upon to a specified extent to enforce a specified security policy. This is equivalent to saying that a trusted system is one whose failure would break a security policy (if a policy exists that the trusted system is trusted to enforce)."

The most essential elements of a trusted system are its trusted computing base (TCB). The TCB of a computer system is the set of all hardware, firmware, and/or software components that are critical to its security, in the sense that bugs or vulnerabilities occurring inside the TCB might jeopardize the security properties of the entire system.

Some devices have a Trusted Execution Environment (TEE) for executing trusted application code. The TCB and TEE cooperate to ensure embedded security functionality can be accessed from within the TEE without a significant security risk. Bugs and vulnerability in these components jeopardize the security properties of the device. The TEE may be effective at detecting, preventing, or countering security events occurring in other parts of the system. It is therefore extremely important that every IoT device have a *trustworthy* TCB!

The authors suggest every TCB for IoT should contain the following:

(A) Attestation key: An asymmetric key supplied by the device manufacturer that establishes device origin authenticity. The Enhanced Privacy Identifier (EPID)[30] can be used to attest device origin without

[28] http://searchsecurity.techtarget.com/definition/trusted-computing

[29] https://en.wikipedia.org/wiki/Trusted_system

[30] Proceedings: WPES '07 Proceedings of the 2007 ACM workshop on Privacy in electronic society, pp 21-30, Alexandria, Virginia, USA – October 29, 2007, ACM New York, NY, USA ©2007, ISBN: 978-1-59593-883-1 doi> https://doi.org/10.1145/1314333.1314337

introducing a trackable identifier that violates privacy.

(B) Attestation functionality: Trusted code that implements attestation and attestation verification logic.

(C) Encryption keys: Symmetric and asymmetric keys used to protect device-device and device-human interactions that may occur in the context of a coalition.

(D) Secure communication: Trusted code that implements cryptographic algorithms used to protect the confidentiality and integrity of information exchanged between devices and TCB peers. It contains support for key management protocols such as Kerberos,[31] PKI,[32] and Fluffy.[33]

(E) Authentication keys: Symmetric and asymmetric keys used to authenticate the originators of messages exchanged device-device and device-human, also in the context of a coalition.

(F) Authentication functionality: Trusted code that implements identity and authentication primitives including support for distributed authentication protocols such as OAuth2[34] and OpenID Connect.[35]

[31] https://web.mit.edu/kerberos/

[32] www.ietf.org/rfc/rfc5280.txt

[33] https://datatracker.ietf.org/doc/draft-hardjono-ace-fluffy/

[34] https://tools.ietf.org/html/rfc6749

[35] http://openid.net/connect/

(G) Secure storage: The ability to store keys, integrity measurements (cryptographic hash), whitelists, settings, and contextual information that if modified or deleted could result in failure of the TCB to correctly apply a security objective.

(H) Contextual awareness functionality: Trusted code that can encrypt and authenticate stored data securely even if the attacker has physical access to the storage resource. The ability to sense and collect security relevant context such as time, location, biometrics, and other context.

(I) Trusted execution environment functionality: Trusted code that correctly implements the TEE environment such that the TEE firmware can be updated securely and computing interfaces into the TEE are resistant to attack.

These security "building blocks" provide the core set of hardened functionalities that enables an IoT device to establish itself as a trustworthy node suitable for inclusion in one or more coalition groups of IoT devices. Once a member of a coalition group, a distributed application can be deployed securely.

Conclusion

The Internet of Things can be described as a dynamic set of distributed computing coalition groups that come into existence seemingly on their own, without a presumption of central control or orchestration. Coalition groups may just as easily disappear, but IoT networks persist as a set of protocols, data structures, and capabilities that enable these dynamics. A secure IoT network is essential to a sustainable and automated distributed

computing on a massive scale where the tiniest of computing nodes needs to support a set of security capabilities that is common to all other nodes in the Internet including the largest cloud servers. Coalitions of devices will work together to manage risk and to preserve the value inherent in the distributed computing venture by vetting coalition memberships. Failure to enforce membership integrity places at risk the value of the coalition. These economic dynamics, once properly understood, motivate proper investment in security capabilities, even among the simplest of IoT devices. This leads to a rethinking for conventional practices that assume security functionality should be less than 15–10% of total system cost. Rather, we think an enlightened approach considers the value of the network is greater than the sum of its constrained endpoints. The cost of security is weighed against the larger value where the percentage investment in security technology, standards, and business practices is aligned. Such a perspective will make it more feasible for most relevant IoT security technology to exist at the right layers of the IoT pyramid.