Editors: D. Balzarotti, davide.balzarotti@eurecom.fr | W. Enck, whenck@ncsu.edu | T. Holz, thorsten.holz@rub.de | A. Stavrou, astavrou@gmu.edu

# Internet of Things Security Research:

## A Rehash of Old Ideas or New Intellectual Challenges?

**Earlence Fernandes, Amir Rahmati, Kevin Eykholt, and Atul Prakash** | University of Michigan

Devices with computational and networking capabilities are enhancing our homes, hospitals, cities, and industries. This emerging network of connected devices—or Internet of Things (IoT)—promises better safety, enhanced management of patients, improved energy efficiency, and optimized manufacturing processes. Despite these many benefits, security vulnerabilities in these systems can lead to user dissatisfaction (for instance, random bugs), privacy violation (for instance, eavesdropping), monetary loss (for instance, denial-of-service attacks or ransomware), or even loss of life (for instance, attackers controlling vehicles[1]). Therefore, it is critical to secure this emerging technology revolution in a timely manner.

Although the research community has begun tackling challenges in securing the IoT, an often-asked question is: What are the new intellectual challenges in the science of security when we talk about the IoT, and what problems can we solve using currently known security techniques? This article summarizes some similarities and differences between IoT security research and classic IT security research.

We take a broad view of the IoT, touching on consumer-grade, industrial control systems and autonomous vehicles. Other IoT areas, such as smart cities, are outside the scope of this article. A whole set of privacy issues might arise from always-connected devices in the physical environment—this article doesn't go into depth on these challenges, but Nigel Davies and his colleagues discuss possible challenges and solutions in "Privacy Mediators: Helping IoT Cross the Chasm."[2] Our focus is on security and safety issues.

### Similarities and Differences

We classify the similarities and differences based on the standard computing stack: hardware, system software, network, and application layer. The IoT computing stack is structured similarly:

- At the lowest layer are devices that can sense and effect physical change in the environment.
- The next layer comprises IoT platforms—software systems that aggregate multiple devices and controlling software to perform useful tasks.
- Next, various connectivity/network protocols enable software and physical devices to communicate with one another.
- Finally, the application layer runs custom code to control physical processes.

We discuss areas of similarities and differences next. We note that it is not our goal to be exhaustive in our discussion.



Copublished by the IEEE Computer and Reliability Societies

1540-7993/17/$33.00 © 2017 IEEE

## Hardware Layer

The hardware layer often forms a root of trust in modern computing systems, and we expect that hardware security research results developed in the context of desktop, mobile, and cloud systems will transfer in some form to IoT systems. We focus on two themes: security for hardware and hardware for security.

**Security for hardware.** Recent work has shown the possibility of hardware-level Trojans—malicious components or instruction sequences that, when triggered, circumvent security guarantees. Kaiyuan Yang and his colleagues recently showed how fabrication-time attackers can inject analog components that force a flip-flop, which maintains the processor's privilege bit, to a target value.[3] With a large percentage of IoT devices being manufactured by third parties (often overseas), hardware-level attacks are an increasing point of concern.

Given the relative simplicity of IoT devices, such as sensors and microcontrollers, in comparison to general-purpose computer processors, open questions are whether such attacks can remain stealthy, and whether postfabrication testing can be more effective in determining whether hardware Trojans exist in a chip.

**Hardware for security.** Galen Hunt and his colleagues recently discussed this topic in *The Seven Properties of Highly Secure Devices*. Two properties directly concern hardware security techniques: a hardware root of trust and hardware-supported software isolation.[4]

Although the ideas of using hardware mechanisms to securely store cryptographic keys (for example, trusted platform modules and one-time fuses) and to create isolation units (for example, memory management units and Intel Software Guard Extension enclaves) are similar to those in classic IT research, we envision many challenges arising in *applying* these notions of hardware security to IoT systems due to their limited computational and energy constraints.

These computational and energy limitations can affect higher-layer security primitives—some IoT devices might not have very precise real-time clocks, making it harder to implement even the most basic of network security protocols that assume the presence of reliable clocks. For example, Amir Rahmati and his colleagues showed how the natural decay rate of static RAM can be used as a timekeeper for embedded devices without clocks (for instance, smart cards).[5]

**Hardware security research results developed in desktop, mobile, and cloud systems will likely transfer to IoT systems.**

In general, we observe that although the core notions of creating hardware to support security primitives is similar to other computing paradigms, the computational and energy limitations at the hardware layer can impact security mechanisms at higher layers in the context of the IoT computing paradigm. We also observe that, conversely, higher-layer security properties might have to be tuned to the specific limitations of the IoT device through a hardware–software codesign approach.

## System Software Layer

The system software layer consists of firmware, OS code, and any privileged system applications or programming frameworks. This layer builds on hardware mechanisms for establishing trust and isolation. We believe that many security principles developed in the context of mobile, desktop, and cloud computing will be applicable to IoT platforms—software systems that are similar in function to OSs for other computing paradigms. We discuss a few areas of similarities and differences, categorized by security principle.

**Process isolation.** Current OSs provide a basic primitive: a fault in one process doesn't affect other processes on the system. These isolation guarantees depend on the presence of a hardware memory management unit (MMU). In small IoT devices (for instance, devices with 64 Kbytes of RAM), such an MMU is generally absent. A challenge here is to support the classic notion of process isolation without an MMU. The Tock OS is currently exploring a combination of language-based isolation features and memory protection units to provide a process isolation abstraction.[6]

In general, although the notion of process isolation is well-known, enabling it for OSs of resource-constrained IoT devices can require new techniques, whereas enabling it for IoT devices with more resources, for example, Nest thermostats or Amazon Alexa, likely won't be a challenge.

**Access control.** OSs protect resources from untrusted code using access control. A piece of code is either given a token (as in a capability-based system) or assigned an unforgeable unique identity on which access control rules are expressed. Building an access control system for a particular domain is often challenging. Our prior work in analyzing consumer

IoT platforms revealed access control design errors as a security flaw.[7] We performed an empirical security analysis of the SmartThings platform and found that access control granularity wasn't designed appropriately, and it led to exploitable overprivilege. A fundamental reason for such granularity design errors in access control systems stems from the tension between usability and security. This tension has manifested itself before, in mobile OSs[8] and, before them, in desktop OSs.[9]

Although the notion of access control still applies to IoT platforms, there are new challenges in the usability aspect of designing such systems. For example, most prior access control systems dealt with virtual objects such as files and processes. In the IoT space, the objects of access control are physical devices and intuitive physical operations. An interesting challenge is how to exploit our natural intuitions about physical objects while designing an access control system for IoT platforms. For example, Earlence Fernandes and his colleagues recently discussed the notion of a user-perceived-risk-based access control system for IoT platforms.[7]

**Information flow control.** Access control is a gatekeeper—once the code obtains access to sensitive resources, access control doesn't provide any further protection. We analyzed a set of smart home platforms and found that current platforms use only access control. Information flow control (IFC) is a promising technique to control how untrusted code uses its access to sensitive resources.[10]

Although IFC isn't a new concept, as evidenced by the multitude of proposed systems for various domains, the challenge lies in applying it meaningfully to a specific domain. For example, FlowFence is a recent proposal for consumer IoT frameworks that enables a dataflow graph approach to IFC due to the structure of IoT apps.[11] Furthermore, the kinds of confidentiality properties for environments such as homes are well-studied; however, the kinds of integrity properties that we might need, which are arguably more important in the IoT, have been less well-studied.

**Software updates.** Updating software is a fundamental security practice to patch security bugs and include additional features once devices are deployed. For smartphones, PCs, and cloud services, updating software is a well-understood, secure, and common practice. However, for physical devices in the IoT, several challenges arise:

- Upgrading software might require a shutdown of the physical processes under control,[12] which could have an economic impact.
- Updates might require reverification of compliance policies for safety-critical devices in sensitive installations like factories and hospitals.
- Updates on computers in tertiary network functions (for instance, a business network) can have unintended effects on a physical process. A prominent example of a negative effect of this kind was the shutdown of a nuclear reactor due to a software update on a computer in the plant's business network.[13]
- Many IoT devices deployed in the field (such as in concrete bridges) can be difficult to physically access and might be intermittently powered (by harvesting power from vibrations). Updating the software on such intermittently powered devices is a challenge that classical computing systems generally don't face.
- IoT devices might not be updatable fundamentally because the manufacturers didn't build an update channel. In this case, we need to revisit our notion of a software update of the host (the device) and include notions of network-based patches.[14]

Although software updates for security are a well-understood concept, designing update systems for the IoT poses new challenges because of the unique properties of the physical processes that are under the control of software.

**Authentication.** Passwords are currently the most widely used mechanism to authenticate users to their IoT devices, platforms, and services. But, they are also a major point of concern because weak passwords are pervasive and have recently enabled large denial-of-service attacks from botnets.[15] Although there are lightweight techniques to obtain statistical estimations of password strength (github.com/dropbox/zxcvbn), weak passwords are still rampant. We don't view *enforcing* reasonable strength passwords (nondefault) as a technical difference from IT security, but rather we view it as a usability challenge. Some proposals suggest moving away from password-based authentication schemes.[4]

Open challenges in authenticating users to IoT devices include answering the following:

- Are activity-based biometrics (for instance, gait and heart-rate) a better alternative to passwords given that IoT devices interact with physical phenomena?
- IoT devices don't necessarily have classic I/O (for instance, no display in Google Home)—this can affect authentication schemes like passwords. Can we design authentication schemes of equivalent security for different interaction modalities?

## Network Layer

As in a classic computing stack, the network layer in the IoT stack

enables devices and software to communicate with each other. However, different from classic networking, IoT networking is marked by a multitude of protocols and is generally populated with fixed-function devices. We elaborate how these differences results in new security challenges and mechanisms.

**Connectivity protocol diversity.** The network layer in the IoT is marked by various physical media and communication protocols. Part of this connectivity protocol diversity stems from the relative infancy of this technology, and part of it stems from the constraints imposed by devices or from the physical spaces that host these devices. For intermittently powered devices, short-range protocols like Bluetooth Low Energy (BLE) and near-field communication (NFC) are vital in conserving energy. For devices located in existing infrastructure, protocols like physical-line communications avoid expensive infrastructural costs. Similarly, visible-light communication can be useful because lights are ubiquitous in physical spaces. This protocol diversity disrupts the operation of network scanning—a fundamental security practice. We highlight this using a BLE port–scanning case study, described below.

In BLE, a rough analog of a TCP port is a service UUID (Universally Unique Identifier). A device can support multiple UUIDs that define the kinds of functionality it provides. There are UUIDs for fitness machines, heart monitors, and so on (see www.bluetooth.com/specifications/gatt/services). When a BLE device is disconnected, it sends out advertisements that can help controllers (or scanners) discover the device, and attempt connections. Advertisements contain rudimentary information, so connections are required to get a full list of the services a device supports. Therefore, for a scanner to work reliably, a device would have to be in a disconnected state as a BLE device accepts only a single connection for its services, unlike TCP ports, where multiple simultaneous connections can be serviced on the same port. This introduces randomness into the scanning process as the scanner will have to "try again" at a later point in time in the hope that the BLE device is in the disconnected state. Furthermore, if a BLE device is connected, it doesn't send advertisements, further complicating scanner operation. (Sophisticated scanners could try to jam existing connections to force them to drop.)

**Challenges arise in adapting known security principles to make them work for the unique IoT computing paradigm.**

Therefore, scanners for IoT protocols are currently very network specific and offer only limited coverage (BLE scanners will be useful only for BLE devices, but it's common for physical spaces such as a home to contain devices using different connectivity protocols). This contrasts starkly with the Internet in general, in which TCP/IP is a constant presence for online services where network scanning is typically used. Port scanning is further complicated in the consumer IoT space due to the practice of placing devices behind a hub or router. Network scanners situated outside such a network won't be able to conduct internal scans.

Because each protocol has its own notions of how two peers communicate with each other, it's unclear how network security practices such as port scanning translate to networks of devices that use various IoT protocols.

**Repurposing networking technologies in unforeseen ways.** Again, a common IoT system architecture for smart homes is to connect multiple devices to a hub. If all the home IoT devices use Wi-Fi as a connectivity protocol, then a Wi-Fi router can be a hub. This kind of configuration poses new security challenges that Wi-Fi wasn't designed to support. For example, it's very difficult to ensure that only a Wi-Fi-enabled presence detector affects a door lock. Such an isolation boundary is useful because there could be multiple devices on a network, some of which might be malicious or compromised through bugs. The isolation unit would serve as defense in depth against such a situation. Furthermore, as we discussed, some devices might not have update channels, necessitating other means of updates. A central hub like a Wi-Fi router is in a good position to apply updates in the form of filters for known malicious traffic patterns. Anna Simpson and her colleagues discuss the design of a Wi-Fi home hub that can perform such security functions.[16]

In the context of smart homes, we observe that hubs like Wi-Fi routers are being increasingly used to support IoT device networks. Adapting these hubs to natively support security properties such as isolation is an open challenge.

**Anomaly detection in the network.** As defense in depth, detecting misbehaving devices on the network is a common and well-deployed security practice in many computing areas. The main challenge in obtaining useful results from anomaly detectors is tuning them to produce a low number of errors—that

is, to minimize how often they either raise a flag for benign behavior or don't raise a flag for malicious behavior. This challenge arises due to the fundamental complexity of the devices we typically connect to a network—general-purpose computers like mobile phones, desktops, and servers. These devices perform multiple functions and lead to complicated network traces that make it difficult to characterize "normal" behavior. In contrast, IoT devices are simple and have a single purpose (that is, they have fixed functions). This can translate to simpler network dynamics and, hence, easier-to-model behaviors, ultimately leading to fewer errors in anomaly detectors. Recent work in the context of industrial control systems have yielded promising results—David Formby and his colleagues show how predictable network characteristics of relays and circuit breakers can be used to reliably fingerprint them.[17]

A physical process evolves as per physical laws in a generally predictable fashion. For example, a garage door of a certain mass takes a specific amount of time to close, and an oven of a certain volume heats up to a specific temperature in a predictable amount of time. We envision that models of these physical processes can be used to reduce the errors in anomaly detectors. In contrast, general-purpose computers, by definition, don't have well-defined behavior models when applications running on them are taken into account.

## Application Layer

The application layer in the IoT is no different from other computing paradigms—it runs customized code for end-user scenarios. We consider two ways in which IoT application behavior can affect security.

**Physical co-relations.** Consider a simple if-this-then-that rule that closes a garage door after 9:00 p.m. If a speaker were placed in the vicinity of the motors controlling the door, it would record a specific acoustic pattern for a specific amount of time whenever the door closes. There is a natural physical co-relation between this acoustic pattern and the closing of the garage doors.

The natural co-relations between physical phenomena could act as feedback channels that IoT platforms could then use to approximately monitor physical processes for deviations from expected behavior. If deviations exist, then it would mean that a failure or security issue occurred.

**Machine learning and control of physical processes.** In recent years, machine learning (ML) and deep learning have found wide applicability to many computing domains—deep-learning robots can learn to grasp objects, and the Nest thermostat can learn and then control HVAC settings automatically. However, recent work has shown that deep-learning algorithms are susceptible to adversarial manipulations of their input—attackers can craft input that looks indistinguishable from benign input to humans, but can be interpreted in a completely different way by machines. For example, tampered images that are fed into a vision algorithm running on an autonomous vehicle can make the vehicle believe a stop sign was a yield sign, causing a possible crash at an intersection. Building robustness into ML algorithms against such attacks is an active area of research whose details are beyond this article's scope. We refer readers to "Towards the Science of Security and Privacy in Machine Learning" for a more thorough treatment of the topic.[18]

As more physical processes come under the control of ML algorithms, their vulnerabilities in adversarial settings will become pressing security and safety issues. Classic IT security has often applied ML to security problems (for instance, malware detection); however, only recently has work begun on securing the ML algorithms.

Broadly, classic IT security research and IoT security research share the basic secure software and hardware construction principles that have been developed in other computing paradigms. The differences form a spectrum of new intellectual challenges. On one end of this spectrum, challenges arise in applying and adapting known security principles to make them work for the unique challenges posed by the IoT computing paradigm. We believe that overcoming many of these challenges will involve a cross-layer codesign approach. For example, due to limited energy availability, hardware security mechanisms might need to be purpose-built depending on the specific higher-level security property we want to enforce—it's not possible to efficiently accommodate a one-size-fits-all security mechanism.

At the other end of the spectrum, the nature of both physical processes and IoT devices lend themselves to the construction of new security mechanisms. As discussed, natural co-relations between physical phenomena can be exploited to detect security and safety failures. Similarly, the predictability of physical processes is another avenue that can be used to detect anomalous events. Finally, introducing ideas from the control engineering world into IoT platform construction (for instance, specialized feedback loops) could lead to a safer and more secure IoT. ■

## References

1. A. Greenberg, "Hackers Remotely Kill a Jeep on the Highway with Me in It," *WIRED*, 21 July 2015; www.wired.com/2015/07/hackers-remotely-kill-jeep-highway.

2. N. Davies et al., "Privacy Mediators: Helping IoT Cross the Chasm," *Proc. 17th Int'l Workshop Hot Topics in Mobile Computing* (Hot Mobile 16), 2016, pp. 39–44.
3. K. Yang et al., "A2: Analog Malicious Hardware," *IEEE Symp. Security and Privacy* (SP 16), 2016, pp. 18–37; dx.doi.org/10.1109/SP.2016.10.
4. G. Hunt, G. Letey, and E. Nightingale, *The Seven Properties of Highly Secure Devices*, tech. report MSR-TR-2017-16, Microsoft, 31 Mar. 2017; www.microsoft.com/en-us/research/publication/seven-properties-highly-secure-devices.
5. A. Rahmati et al., "Tardis: Time and Remanence Decay in SRAM to Implement Secure Protocols on Embedded Devices without Clocks," *21st USENIX Security Symp.* (USENIX Security 12), 2012, pp. 221–236; www.usenix.org/conference/usenixsecurity12/technical-sessions/presentation/rahmati.
6. A. Levy et al., "Ownership Is Theft: Experiences Building an Embedded OS in Rust," *Proc. 8th Workshop Programming Languages and Operating Systems* (PLOS 15), 2015, pp. 21–26; doi.acm.org/10.1145/2818302.2818306.
7. E. Fernandes, J. Jung, and A. Prakash, "Security Analysis of Emerging Smart Home Applications," *Proc. 37th IEEE Symp. Security and Privacy* (SP 16), 2016; doi:10.1109/SP.2016.44.
8. A.P. Felt et al., "Android Permissions: User Attention, Comprehension, and Behavior," *Proc. 8th Symp. Usable Privacy and Security* (SOUPS 12), 2012, pp. 3:1–3:14; doi.acm.org/10.1145/2335356.2335360.
9. E. Bertino et al., "Some Usability Considerations in Access Control Systems," *Proc. Symp. Usable Security and Privacy* (SOUPS 08), 2008; cups.cs.cmu.edu/soups/2008/USM/bertino.pdf.
10. E. Fernandes et al., "Security Implications of Permission Models in Smart-Home Application Frameworks," *IEEE Security & Privacy*, vol. 15, no. 2, 2017, pp. 24–30; dx.doi.org/10.1109/MSP.2017.43.
11. E. Fernandes et al., "FlowFence: Practical Data Protection for Emerging IoT Application Frameworks," *Proc. 25th USENIX Security Symposium* (USENIX Security 16), 2016; www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/fernandes.
12. A. Cardenas et al., "Challenges for Securing Cyber Physical Systems," *Proc. Workshop Future Directions in Cyber-Physical Systems Security*, Dept. Homeland Security, 2009; chess.eecs.berkeley.edu/pubs/601.html.
13. B. Krebs, "Cyber Incident Blamed for Nuclear Power Plant Shutdown," *Washington Post*, 5 June 2008; www.washingtonpost.com/wp-dyn/content/article/2008/06/05/AR2008060501958.html.
14. T. Yu et al., "Handling a Trillion (Unfixable) Flaws on a Billion Devices: Rethinking Network Security for the Internet-of-Things," *Proc. 14th ACM Workshop Hot Topics in Networks* (HotNets 14), 2015, pp. 5:1–5:7; doi.acm.org/10.1145/2834050.2834095.
15. B. Krebs, "Did the Mirai Botnet Really Take Liberia Offline?," Krebs on Security, 4 Nov. 2016; krebsonsecurity.com/tag/mirai-botnet.
16. A. Simpson et al., *Securing Vulnerable Home IoT Devices with an In-Hub Security Manager*, tech. report UW-CSE-17-01-01, Univ. Washington, Jan. 2017.
17. D. Formby et al., "Who's in Control of Your Control System? Device Fingerprinting for Cyber-Physical Systems," *Network and Distributed System Symp.* (NDSS 16), 2016; www.internetsociety.org/sites/default/files/blogs-media/who-control-your-control-system-device-fingerprinting-cyber-physical-systems.pdf.
18. N. Papernot, "Towards the Science of Security and Privacy in Machine Learning," *Computing Research Repository*, vol. abs/1611.03814, 2016; arxiv.org/abs/1611.03814.

**Earlence Fernandes** is a research associate at the University of Washington. At the time of this writing, he was a PhD candidate at the University of Michigan. Contact him at earlence@cs.washington.edu.

**Amir Rahmati** is a security research engineer at Samsung Research America. He will be joining the Computer Science Department at Stony Brook University in 2018. At the time of this writing, he was a PhD candidate at the University of Michigan. Contact him at amir@rahmati.com.

**Kevin Eykholt** is a PhD candidate at the University of Michigan. Contact him at keykholt@umich.edu.

**Atul Prakash** is a professor of computer science at the University of Michigan. Contact him at aprakash@umich.edu.

myCS
Read your subscriptions through the myCS publications portal at **http://mycs.computer.org**