Project Report
LSP-228

Securing the Instrumented Battle Space: FY17 Line-Supported Cyber Security Program

D. Whelihan

4 April 2018

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LEXINGTON, MASSACHUSETTS

This material is based upon work supported by the Assistant Secretary of Defense for Research and Engineering under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001.

This report is the result of studies performed at Lincoln Laboratory, a federally funded research and development center operated by Massachusetts Institute of Technology. This material is based upon work supported by the Assistant Secretary of Defense for Research and Engineering under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Assistant Secretary of Defense for Research and Engineering.

Massachusetts Institute of Technology
Lincoln Laboratory

Securing the Instrumented Battle Space: FY17 Line-Supported Cyber Security Program

D. Whelihan
Group 53

Project Report LSP-228

4 April 2018

Lexington Massachusetts

This page intentionally left blank.

TABLE OF CONTENTS

This page intentionally left blank.

LIST OF ILLUSTRATIONS

This page intentionally left blank.

1. INTRODUCTION

Internet of Things (IoT) is a buzzword that is commonly used to describe the proliferation of wirelessly connected embedded systems such as cellular phones, sensors, appliances, vehicles, and more. The IoT has been enabled by a confluence of technologies, namely: increasingly capacious batteries, extreme miniaturization of processing horsepower, and ubiquitous access to broadband wireless communication. The end result of improvements in these areas is that computer systems that can sense and actuate their environment can be made smaller and cheaper and hence more numerous. The proliferation of these devices, and the fact that they are connected to the internet on equal footing with traditional desktop and server computers has blurred the lines on what exactly computing is, and where the work to do the computation is performed.

In this report, we will use the term IoT to mean the general pattern exemplified by wirelessly and widely accessible sensors, actuators, and computers dedicated to cooperatively streamlining our activities. We will refer to *an* IoT system as a group of wirelessly connected and communicating sensors and actuators dedicated to a relatively narrow function. An IoT system is made of IoT elements, which generally have some form of wireless communication, computation, and sensing and/or actuation capabilities. When we speak of IoT in the DoD, we are referring to the general pattern of IoT. We are not implying that DoD systems will be plugged into the world-wide Internet, rather that we will have networks of sensors and actuators that cooperate to aid the warfighter and make support infrastructure more efficient.

To best guess the future of DoD IoT, we can examine the commercial (e.g., personal and home) IoT. The most common IoT elements and systems are designed to operate in the home. Generally, a home IoT is a set of distributed sensors such as thermostats [1], smart refrigerators [2], and actuators such as door locks [3] or light controllers [4] that ideally work together to streamline our lives. Ideally, they make us more efficient and more in control of our environment. In addition, the aphorism “knowledge is power” serves as the motivation for the home IoT. If the thermostat knows your pattern of life, then your thermostat can more efficiently heat your home by turning the heat on only when you are there to enjoy it. Increasingly, knowledge is not only power to the consumer, but also to industry. With the advent of the modern smartphone came a stunning opportunity: People were willing to instrument *themselves* with connected accelerometers, GPSes, light sensors, sound sensors, and others. The always-on connectedness via wireless communication enabled a new kind of situational awareness. This situational awareness was/is largely still free in dollar terms, provided one shares their situation with the manufacturer of the device or service delivering that awareness. For example, one can know that a thunderstorm is impending, provided one gives access to the location sensor on the phone. This does not feel like much of a trade, but it is worth real money to marketers, sellers (e.g., Amazon), demographers, and decision makers, primarily when it is aggregated to establish trends in populations. This is a centralized approach to IoT, in which sensors and actuators are part of a wider, more communication-intensive web of systems and services.

Such a centralized approach to the IoT, in which the IoT elements are the leaves in a larger, more complex tree of functionality that is rooted in the cloud, has some benefits in that it is easier to maintain a coherent picture of a large number of devices and the environments they are experiencing. Maintenance and code updates are easier because all devices can call back to a central service. However, there are drawbacks: privacy being one of the most glaring, but also potentially reliability, especially if the communications medium is insecure or unreliable. If a smart thermostat requires internet connectivity to work, what happens when internet access goes down in bad weather?

These drawbacks are all worrisome in the face of random failures, but purposeful cyber-effects induced by a determined adversary significantly raise the stakes. This is what happens when security vulnerabilities meet mass-produced, always-on IoT elements. Since the expansion of IoT in the commercial world was driven by a confluence of technological factors, it stands to reason that military systems owned by the United States Department of Defense (DoD) would start to reflect improvements in those areas also. In this report, we hope to shed light on the following questions:

- What does the DoD analog of IoT look like?
- How does the commercial world's IoT differ from the DoD's IoT?
- What is needed to secure DoD IoT systems?

The idea of a decentralized style of IoT, in which devices do not have to reach back to a cloud resource, and can mutually authenticate each other without a central authority (or at least only sporadic communication with one) is a necessary condition for the DoD, as their systems operate in communication-challenged environments. Many of the recommendation we make in this report should be seen in that context, and the presence of a challenging, highly adversarial environment with extremely high stakes should be a founding assumption in any DoD IoT system.

2. DOD'S USE OF IOT

As described in the previous section, the allure of the IoT pattern is that it enables efficiencies gained by instrumenting distributed systems with connected sensors, actuators, and computers. For the DoD, those systems may be weapons platforms such as ships or vehicles, squads of soldiers, or the heating systems in any of the thousands of buildings that make up the DoD physical plant. The DoD will always be resource-constrained in that the projection of power requires efficient and timely redistribution of resources all across the globe. The promise of IoT is to streamline that redistribution, and be more efficient with distributed resources resulting in a more effective fighting force.

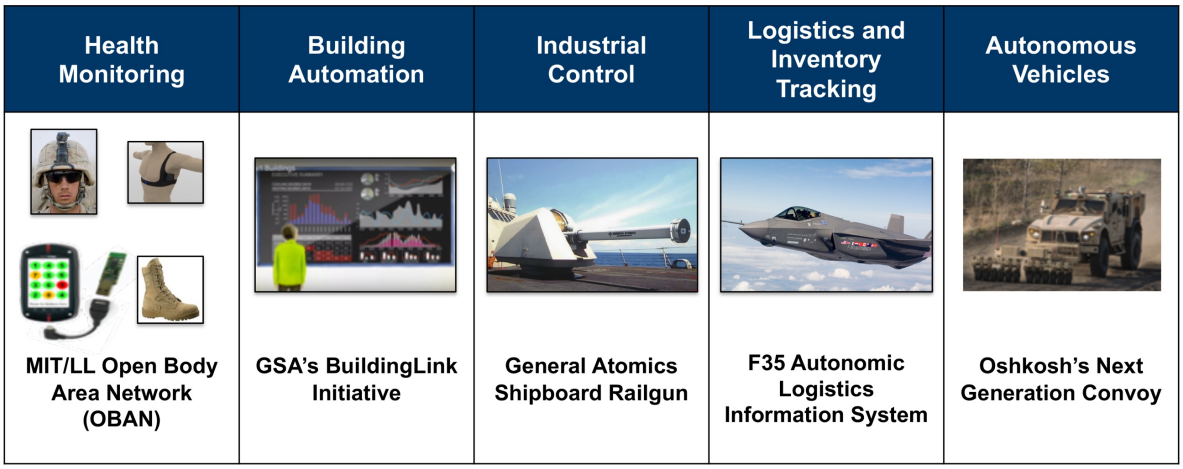

Figure 1. Example of the IoT pattern in DoD systems.

In this section, we will describe five future and current systems, shown in Figure 1, in the DoD that fit the IoT pattern. This is by no means an exhaustive list, but it should serve to describe a number of usage modes for DoD IoT-patterned systems.

2.1 SOLDIER HEALTH MONITORING

Monitoring the health and status of soldiers in the field is a promising application for IoT in the DoD. This was explored in the Open Body Area Network (OBAN) program at MIT Lincoln Laboratory [7]. The goal of such monitoring is to provide fine-grained situational awareness on the state of individual troops and their squads to their leadership. Because soldiers must be mobile and able to move quickly, physiological monitors for heart rate, carried load, body temperature, and others must be lightweight and able to communicate wirelessly to aggregate data at the squad-level and higher. This is a very interesting case because by default, the IoT network of instrumented soldiers is intermingled with a determined adversary, both physically and wirelessly.

2.2 SMART BUILDING

One of the closest overlaps between the DoD and the commercial world is in smart buildings. Smart buildings are engineered with the goal of achieving higher efficiency, convenience, and effectiveness of the workforce they support. With many thousands of buildings worldwide, the DoD has a serious stake in modernizing its physical plant [8]. Smart buildings fit the IoT pattern because they are heavily instrumented with sensors such as thermostats or motion sensors whose data is used to actuate lighting, heating, and security systems. Increasingly, these sensors are wireless for ease of deployment.

2.3 INDUSTRIAL CONTROL

Large weapon systems such as ships share many characteristics with industrial processing systems. They require tight coordination between complex sub-systems such as actuators, power-generation systems, etc. Some of the same technologies used to control industrial processes, Programmable Logic Controllers (PLCs) are used to control DoD weapons platforms. These devices are distributed around large DoD systems to control and report the status of the many moving parts that are used to achieve the mission. On ships, the advent of electrically driven weapons for directed energy weapons (lasers) [9] to rail guns [10] will impact the way large complex systems are build and operated. Many of these systems require tremendous instantaneous energy, which must be provided by the system power plant, perhaps at the expense of other less critical systems. A secure and efficient electrical nervous system provided by distributed controllers is vital to maintaining mission effectiveness.

2.4 LOGISTICS AND INVENTORY TRACKING

The DoD requires vast amounts of supplies to maintain effectiveness in peace-time, let alone in war where those supplies must be quickly transported to almost anywhere in the world. Lockheed Martin has implemented a system called ALIS (Autonomic Logistics Information System) [11] for the F-35 Joint Strike Fighter that allows the aircraft to report its own maintenance needs to a global network. Such systems are enabled by the connectedness that is the hallmark of IoT systems. The ALIS system is an example of a full-featured centralized system that aims to improve many aspects of sustainment of the F-35 platform.

2.5 AUTONOMOUS VEHICLES

The DoD already employs remotely operated vehicles in the form of aerial drones such as the weaponized MQ-9 Reaper [12], and the PackBot 510 [13]. These devices are wirelessly operated sensing/actuated machines that connect to control stations that in turn can be connected to larger networks. In addition, significant research dollars are being poured into swarms of these vehicles [14] that behave autonomously. These swarms have all of the characteristics of the IoT pattern, with the added complication of closed-loop control.

2.6 IOT THREATS

There has been no shortage of IoT hacks in the news from the Mirai botnet [5], which exploited vulnerabilities in web-enabled devices to create a huge, highly effective botnet out of webcams and home routers to the Stuxnet worm [6] that targeted networked industrial equipment. These are two examples of network-borne threats. A key difference however was their goal: Wheras Mirai-infected devices were used to deny access to infrastructure and services by flooding them with data on the internet, Stuxnet was used to infiltrate an industrial facility to destroy critical infrastructure. Stuxnet's kinetic or physical effect had a great impact on the geopolitical landscape by severely checking Iran's nuclear ambitions.

The threats against the DoD IoT systems are similar to those in the commercial world. What is different however, is the magnitude of the effect of a successful cyberattack against DoD systems. We already have examples of cyberattacks changing the fate of nations with Stuxnet, and significant evidence of their use in the Ukraine [19]. In that example, a compromised phone app used to compute artillery trajectories was reporting the location of the users back to enemy forces. Lives were lost. We consider this an IoT hack in that a wirelessly connected small-form factor embedded information delivery system. That system had a sensor on board that was compromised by the enemy to great effect. Many, if not most attacks on these and other systems are remote, network-borne attacks, though Stuxnet got into its target networks via an infected USB drive.

One important thing to note is that the IoT pattern results in a proliferation of small devices in many locations, and owned by a variety of types of people with an even wider variety of technical acumen. Should the soldiers firing the artillery pieces in the Ukraine have been suspicious of an app they found on the internet? Absolutely. Can we expect your average soldier to think these things through? Probably not. This raises the point that just like in the commercial world, the most potent threat to the security of these systems is the legitimate users themselves.

A vital though as-yet largely unfulfilled property of modern computer systems in general is *usable security*. Usable security is comprised of policies and procedures that are clear, easy to follow, and most importantly, do not seem arbitrary to users. An example of a rather arbitrary and hard-to-understand security policy is the Android ecosystem's permissions mechanism. At least as late as the KitKat release (4.4.4), loading an app will result in a screen popping up asking whether the user wanted to grant permission for the app to utilize sensors such as the microphone or GPS. Because the implications of a yes/no for any of these are nebulous, most users will simply click past this because they are anxious to get to the efficiency improving functionality of the app. In addition, our perception of risk will be skewed by the most proximate threat. If someone is shooting at you with a rifle, you might just permit an app to utilize your GPS despite the fact that a compromised app can call in much heavier artillery.

It is safe to say that one of the salient features of IoT-patterned systems is that they have an expanded attack surface owing to a proliferation of small devices, each with their own vulnerabilities. This is compounded by a desire for the devices to be low-power, which limits options for security. In the following

section, we describe a universal threat model for these systems that permits an evaluation of the overall security not just of the IoT elements alone, but the full IoT system.

3. A GENERALIZED IOT THREAT MODEL

Threat models have been generated for many systems and systems of systems. From commercial software exploitation to attacks on DoD systems [20], these models must incorporate the concept of operations (CONOPS) of the machine, its exposure to threats, attack surface [21], and its value to an attacker. The threat model for DoD IoT is complicated based on the necessarily decentralized nature of DoD IoT systems. These systems generally cooperate to realize a purpose for the warfighter, whether it is to obtain fine-grained situational awareness across a battlefield or to ensure that the proper materials are in the right place to carry out the mission. Despite necessary decentralization due to challenged communications and/or the fog of war, the connection of an IoT system at the tactical edge with computers in the broader DoD enterprise (perhaps a DoD cloud) is an important capability. This means that one of the shifts in the computing landscape is that we are no longer relying on systems that can be effectively secured by the staples of military macro-level security, Guns, Gates, and Guards. It does not matter that armed guards surround a data center if it is connected to a compromised device outside of the secure perimeter. This is the true danger of the IoT pattern.

As was mentioned in the previous section, usability is a primary driver of functionality, and a must-have in DoD systems. To be truly effective, the devices that sense, compute, and actuate must be agile such that they can properly instrument and actuate the DoD enterprise. The ability to scale to that level is not limited by technology. The commercial world is already scaling rapidly. Rather the ability will be limited by our ability to balance security with ease-of-use.

The purpose of this section is to outline a threat model that will allow the characterization of distributed, communicating heterogeneous collections of sensors, actuators, and processing nodes to make sense of the threat environment and guide application of security technologies. Understanding the system CONOPS and threats will allow the application of security technology while minimally impacting user workload. It is fine (and expected) to remove work by making target functionality easier to use, but adding work is largely unacceptable unless it is obvious that the work expended to adopt or use the offered capability is justified by the use of that capability. This is especially true for security features, as users usually view them as impediments to getting their work done.

A holistically useful metric for security has eluded the community thus far. The metrics that have been developed are relative to system features and/or dependent on a system model that may not reflect reality [22] as it is generally infeasible to prove all properties of a system conform to the security goals. Validating any model-based metric is difficult if data is not available. In the commercial world, data on successful exploits is available, but it can be closely held. In the DoD, this problem is even worse, with the existing data often being useful only to support an existence proof of a vulnerability. Analytical techniques, often called formal methods, show great promise in helping to secure systems by permitting formal proofs of security, but even formally verified code can still be hacked [23]. This is because such techniques tend to verify that the system is equivalent to a particular model. The trick is to equate that model with reality,

which is usually the technique's downfall. Complex systems in complex environments beset by complex adversaries are difficult to model.

Nonetheless, the singular element of any security analysis must be the threat model. The threat model is the set of assumptions we make about the system, user needs and tendencies, and adversaries that motivates choices of security properties in the system. The threat model is critical because it establishes a framework for understanding the likelihood of a successful attack (driven by exposure and vulnerability) and the consequence of a successful attack (loss of capability, data, or both). Since the canonical definition of risk is likelihood times consequence, the threat model is an integral part of cyber-risk assessment.

To build a threat model, it helps to think from the perspective of an attacker. The following are questions that an attacker might ask herself before taking action:

- What is the value of the information on the target?
- What is the value of diminishing or eliminating the availability of the target to the owner?
- Can the capability afforded by the target be co-opted and used against the owner?
- How exposed is the target? What interfaces are available to me to effect change on the target?
- How much time do I have on those interfaces before I am detected?
- What is my ability to stay resident on the target system once I compromise it?
- What is my ability to move laterally once I compromise a single part of a system?

First, a determined attacker will assign a certain value to compromising the system. There is a value proposition associated with compromising the system, whether it is stealing cryptographic keys to listen in on communication, critical data about the system, its owners, or the mission. The advent of IoT has changed that value proposition however. It is increasingly possible to deny and/or steal *capability* from the owner. For example, an adversary may take control of a small drone and use it to spy on the drone's owner. The consequence of this attack is severely magnified when the compromised system has a kinetic warfare function, or can be used as a weapon.

Based on our earlier definition, an IoT sub-system is a component of a larger system that conveys a capability to the user. That system is usually connected to other systems via one or more wireless channels. Unless highly directional, that channel can be observed by any adversary with the right antenna [24]. This exposure of the channel over which the system communicates is largely unavoidable. In addition to attacking the system from a distance, many IoT devices can easily find their way into an adversary's hands and be attacked physically. Attackers have many methods, both invasive [25] and non-invasive (destructive) [26, 27] for analyzing and compromising systems.

The question of whether exploitation of a particular vulnerability is detectable, and in what time, is context dependent. It is dependent on the methods and goals of the attacker. All of the methods of manipulation discussed thus far attack one or more elements of the CIA triad of Confidentiality, Integrity, and Availability [28]. Confidentiality may be the most difficult property to protect in wireless, distributed devices as there is always a chance that despite our best efforts, the adversary has gained access to confidential information and it is nearly impossible to know. In fact, we usually detect such failures by looking at the other members of the triad: we measure the state of the sub-system to establish its integrity (i.e., that it is in an acceptable and unmodified state), or availability (if it is down or not responding, an adversary may have compromised the system).

Confidentiality can be lost in a number of ways. Chief among them is by simple eaves dropping. If wireless communication between the IoT devices and the larger system is not secured, the attacker need only listen to the proper frequencies to steal information. Another way to violate confidentiality is to force the system to divulge its secrets by manipulating the IoT device into divulging secrets despite protections such as encryption. Almost invariably, IoT systems, and embedded systems in general, incorporate microprocessors that execute code that defines the sophisticated behaviors and capabilities we expect of the IoT device. Those capabilities include communication for configuration, updating code, and command and control. Each of these functions can have bugs, and allow an attacker masquerading as a legitimate user to gain elevated privileges on a system and therefore access to confidential information. The sophistication of IoT systems is likely to be fairly low due to cost and SWaP restrictions in both commercial/home and DoD systems.

The microprocessors in these systems are relatively straightforward devices, and may run the application software “bare-metal,” without an operating system. Such systems have very little capability to monitor or check their own behavior, therefore an attacker may be able to stay resident for long periods of time. Conversely, a lack of sophistication usually entails a lack of value to the attacker in traditional systems. In IoT however, the value of simply injecting bad or misleading data into the broader system may be extremely valuable.

3.1 TRUST

One real danger of IoT compromise is the possibility that an adversary can move laterally to other devices in the network. Unsecured IoT can give attackers a window into our larger systems from which to conduct more sophisticated attacks. They can use that window to elicit cyber-effects over the entire extended capability by abusing information stolen in one sub-system, such as cryptographic keys. The attacker will attempt to use the system to modify or co-opt the overall functionality of the extended system. This can be done by:

- Exploiting a trust relationship between the compromised sub-system and other sub-systems to co-opt more of the full system
- Feeding incorrect data to other devices in the network to affect the overall behavior

- Destroying the aggregate capability by denying the system the input of the compromised device.

There is a significant lack of tools, techniques, and consistency in threat modeling for systems in general, and especially embedded systems. The discussion above defines threats in general terms, but a useful model requires a deeper treatment of threat vectors. Aside from the defining characteristics of IoT systems, one major difference between IoT systems and the desktop and server systems that have been widely studied from a security perspective is the variety of hardware and software that comprises IoT sub-systems.

In our other work analyzing embedded systems, we have found it helpful to assume a *communication and compute* mindset when modeling such systems for threat analysis. Because IoT systems are essentially embedded computers, we believe this technique, which is described below scales not just to IoT sub-systems, but also to larger collections of IoT sub-systems.

We assume that components in IoT systems can be broken down into two fundamental operations: communication of data (critical or not), and sources of that data, which we will call computers. We use the term computer here very generally to represent any system that can store or permute data. Computers generally have inputs and outputs, and they can have multiple functions. A valid question at this stage is to ask why we separate system components into only two bins? As defined earlier, an IoT is usually comprised of a microprocessor, wireless communications, and sensors and actuators. The IoT will have long-term storage such as random access memory (RAM) and non-volatile storage such as flash memory. Why not differentiate between all of these things and model the threat and CONOPS in the context of the actual system components? One answer is that establishing the security of an extended IoT system requires establishment of parity between the function and vulnerabilities of various subsystems. Performing a high-level security analysis at a level of computers and communications permits just such parity to be established to make judgements about the whole system. We argue that there is no loss of specificity however, as a deeper look can always be performed on any one sub-system to focus on specific threats or vulnerabilities.

We claim that calling memories (both static and dynamic) computers in this model does not lead to a reduction in fidelity because: 1. Many stand-alone memories themselves contain processors, 2. We believe the temporal aspect of data storage is a secondary consideration when evaluating threats to IoT systems. By temporal, we mean the length of time data is stored. For instance, a traditional micro-processor stores data in registers for a much shorter time than it stores it in RAM, and for a much shorter time than it is stored on a solid state disk. Where these three differ is in their exposure to threats, and their mode of operation to store data. The microprocessor registers are only accessible to code running on that microprocessor, whereas RAM may be accessible by multiple microprocessors running different code, and finally the solid state disk may be accessible to external devices. As for mode of operation, both RAM and microprocessors require the presence of power to store data. Generally, this means that data is lost (and made unavailable to

an attacker) when power is removed[1]. Therefore, systems have nothing to protect when power is off. This is in contrast with the solid state disk, which stores data between power cycles. Attacks against such systems are important and have spawned programs at Lincoln [29] and commercially [30]. Despite these differences in threats, the fundamental mode of access to all of these is a logical request architecture in which a transaction is sent to the device that usually includes a location in the RAM and the operation to perform (read or write). Thus, from the outside, a processor and a memory look very similar.

Communication elements and computers are attacked differently. Generally, communication elements are eavesdropped upon. An adversary taps into the communication medium directly—by connecting analysis equipment to wires or by co-opting part of the system to monitor shared resources such as busses [31] or processor caches [32,33], or indirectly by detecting, recording, and correlating electromagnetic emanations [34].

Examples of attacks on communication elements:

1. Direct bus tapping to learn shared secrets [35]
2. Power tapping to learn critical program information. This is an example of an unintended communication link being exploited.
3. Measuring timing on interfaces to establish data dependence on length or frequency of transmission
4. Comparison of data (even encrypted) over time to establish patterns. A good example is extreme data leakage that arises when using AES ECB mode on data [36]
5. Eavesdropping on purposeful Electromagnetic Emanations (such as a wireless link between an IoT sub-system and a base station)

Examples of attacks on compute elements:

1. Protocol attacks- Violating rules of communication to force the computer to perform improperly.
2. Glitch attacks – Manipulating the power supply to get the system to perform improperly
3. BIOS attacks – replacing critical software with software controlled by an attacker

[1] There are attacks that can recover data when power is removed, such as “cold boot attacks” [55] that rely on memory remnants.

The goal of attacks on communication elements is usually to obtain secret information about running applications, or the data on which they are computing or data they have computed. The goal of attacking the computers can be the same, but also to obtain control of the system – to co-opt it for the attacker's use.

4. IOT LIFECYCLE

IoT devices have a lifecycle from creation to decommissioning that can take them into the hands of many authorized users, especially in the DoD CONOP. That lifecycle includes events from construction to deployment, adding of users, parameterization, enrollment in the broader system, code update, and decommissioning. Many of these events may occur multiple times and in multiple geographic locations. One of the salient features of an IoT is its proximity to potential adversaries. Those adversaries may participate in any one of these events, which should reinforce the point that very little can be taken for granted in terms of trust.

We divide the life of IoT sub-systems into four phases: manufacturing, deployment, sustainment, and decommissioning. These phases are described below:

Manufacturing – The act of physically creating the device. This phase will consist of designing the electrical and physical characteristics of the device, sourcing parts, constructing the device from those parts, programming, testing, and shipping.

Deployment – The sequence of steps required to take a manufactured device and put it into operation as part of a larger system. This includes transportation, provisioning of vital information such as mission parameters and trust anchors and or cryptographic keys, and potentially user enrollment.

Sustainment – The steps necessary to keep the system in operation, including periodic reprovisioning of trust anchors and other cryptographic information, re-enrollment into other systems, ownership changes, etc.

Decommissioning – The act of rendering a system unusable or untrustable for disposal.

To understand the path through these phases, we will focus on a use case that will provide a detailed picture of the players involved in each phase, as well as the impact of design decisions. With this, and the threat model from Section 3, the technology gaps will be more apparent. After describing the device lifecycle, we will discuss a design that may meet the exposed requirements.

4.1 USE CASE: BUILDING A SOLDIER-AS-A-SENSOR IOT SYSTEM

A program office puts out a request for proposals to build an environmental monitoring system for soldiers in the field. The system is intended to record and characterize the local Electro Magnetic (EM) environment as well as the soldier's critical status such as heart rate, breathing, and body temperature, as well as number of shots fired by the soldier's weapon. The information is to be aggregated over a wireless link to a centralized receiver which will send the information over a common data link (CDL) modem to a command element thousands of miles away. The sensors and body mounted transmitter must run for 12 hours between charges.

Figure 2 shows such a soldier-as-as-a-sensor IoT in which soldiers are outfitted with a variety of electronic sensors such as cameras and body-mounted sensors that feed back to an aggregator gateway over a high-frequency link, which in turn feeds back via a Common Data Link (CDL) uplink to a satellite, then down to a shared compute resource (a cloud). In our compute and communicate model, this would be represented as follows:

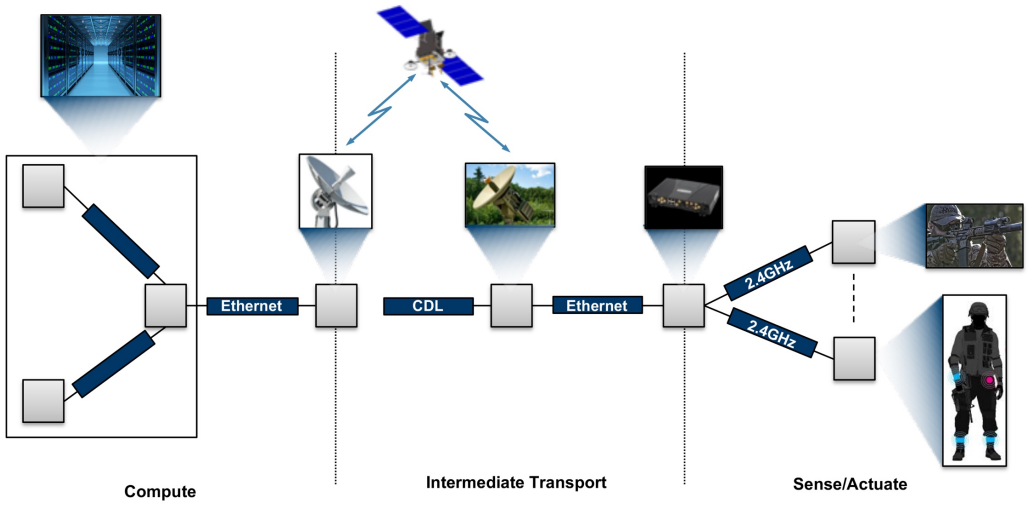

Figure 2. A soldier-as-a-sensor IoT system.

On first blush, this does not look significantly different from any COMMS-enabled mission with reach back. Where the IoT threat model changes this is between the Intermediate Transport and Sense/Actuate regions, where it becomes more difficult to call systems closer to the right (the tactical edge) trustworthy. While it is relatively easy to say that the systems to the left of the satellite link are protected (by Guns, Gates, and Guards and dedicated IT professionals), it is progressively more difficult to make that claim going right. Despite the fact that these devices are, in fact, worn or handled by guards, they can be lost, substituted for other devices, or misconfigured to jeopardize the security of the system, as well as the success of the mission. The key in that environment is to dissect the IoT devices themselves into the compute and communicate model to understand the vulnerability of the entire system.

The threat model for this system will be highly aggressive. As we mentioned earlier however, the system must not just be functional and secure. It must also be manufacturable and sustainable. In the following sub-sections, we will discuss the steps in the IoT system lifecycle to convey the challenges and considerations that must be incorporated into the design.

4.2 THE DOD IOT LIFECYCLE

The lifecycle of DoD equipment incorporates similar events as their commercial counterparts, but the details, especially those that pertain to security, are where significant differences lie. While commercial systems can simply be thrown away when they break or grow obsolete, DoD systems must be decommissioned to avoid leaking sensitive data such as cryptographic key, and to avoid giving the enemy a valuable resource. In addition, whereas the commercial world can rely on "security through obscurity" there is only one DoD, and it is a big target. Thus ANY deployed device could be used to gain an advantage and subvert United States interests and priorities. In the following sections we discuss the realities of four phases of DoD IoT life: Manufacturing, Deployment, Sustainment, and Decommissioning with the goal of informing the subsequent discussion of the design of our hypothetical system.

4.2.1 Manufacturing

The device will be manufactured from chips and components, many of which will be made overseas. This is an unfortunate reality inherent to the sourcing of Commercial-Off-The-Shelf (COTS) parts, which are a staple of DoD procurement. Because of this, many components will be *not* made in trusted foundries. This makes vetting of those components of paramount importance. Without this step, a device designed for high-assurance is still vulnerable to the impact of counterfeit components which may not have the right reliability characteristics and could fail at inopportune moments—an Availability failure, or more insidious threats such as hardware Trojans [37].

The creation of an embedded system requires that a trusted relationship between it and the devices it connects with be established. This usually is accomplished by injecting or locally generating root secrets (keys) that are programmed into the device and stored long term. While this programming may be done later, say in deployment, the design and manufacturing must support those secrets generation and storage. Some systems utilize characteristics of the chip itself, so-called Physical Unclonable Function (PUF)-enabled systems will likely have their root-secret available at this time. Keeping the characteristics of that PUF (and the secrets it can generate) confidential in an untrusted or semi-trusted manufacturing environment is challenging. One way to keep some amount of confidentiality despite the potential for adversary access is to mix that PUF value with yet another secret that is programmed later. It is arguable that IoT devices are potentially very susceptible to attacks at manufacturing time, as they tend to be more numerous, and so keeping track of any one device is more difficult.

Packaging for DoD devices is generally much more heavyweight than for commercial devices as the DoD devices must meet stringent requirements on reliability in harsh environments. This means that these devices will tend to be heavier and bulkier than their commercial counterparts. Further, the decision to place secret root-of-trust key material inside of the device has significant implications on physical structure as that key material must be protected. Device enclosures must have AT properties, which affects materials, mass, and arrangement of the systems.

DoD systems that secure national security information, whether it is cryptographic key material or SECRET or TOP SECRET sensor "take," must be certified by the National Security Agency (NSA). This is a stringent process that can easily add one to two years to the creation of a system. DoD IoT, which will have a more disposable nature and faces a high probability of loss, may force us to upend the current method, which attempts to guarantee that loss will not jeopardize the mission. Regardless, the burden of certification is a significant differentiator between commercial IoT and DoD IoT.

4.2.2 Deployment

Once a system is manufactured and certified for use, it must be enabled and *provisioned* for use. The process of provisioning involves placing any mission-specific information on the device such as key material, functional configuration, security configuration, and information about authorized users. For DoD IoT devices, elements of this process may be done at a protected depot, but some tasks will need to be done more frequently, potentially in the field or in a forward operating base and so relegated to the sustainment process.

For the device we are designing in this report, it is reasonable to assume that long-term secrets such as the device identity and roots of trust can be provisioned infrequently in a protected location. However, given a Public Key Infrastructure (PKI)-based scheme, this places a great deal of importance on maintenance of the chain-of trust that enables the overall system to trust the IoT device. Recall that a PKI-enabled device must not only maintain the secrecy of its private identity (its private key), it must also maintain the integrity of a root public key with which it will verify the authenticity and trustworthiness of other devices to which it will pair. This means that a consistent set of certificate authorities must be present, and they must bless all devices by providing their public keys. This can raise problems when different organization within the DoD wish for their IoT devices to communicate and provide a coherent picture of the battlefield.

The way that keys get into a DoD IoT device is an important consideration. For many devices in use by the DoD, keys are injected into a device in the "red" (unencrypted). A maintenance worker in a depot will plug a device called a Simple Key Loader (SKL) into a special terminal that distributes NSA-approved key material (also known as keyfill) which loads a key. The worker then plugs the SKL into a keyfill port on the device and injects the key into the device memory. This means that the depot must have the capability to retrieve and inject keys. Keys generally have a finite lifetime, and if a device is to be out of the depot for longer than that lifetime, chains of keys intended to be used sequentially are injected. Those keys are often referred to as Pre-Placed Keys (PPKs). A principle reason for rotating the keys in this way is to deny an adversary a large amount of cipher text to study to break the keys. Frequent change of keys is generally a policy used for symmetric traffic keys however. A PKI scheme like the one discussed earlier would likely require key changes, but the private key that is used to derive traffic keys will be used infrequently, requiring infrequent key changes. However, there lies a vulnerability: If an adversary can induce the device to frequently re-establish shared keys, then the root keys are more exposed. In addition, an adversary could use the fact that PKI-based key agreement is computationally complex to burn power on the device, resulting in a denial of service when the battery-based IoT devices run out of power. This means that some

sort of throttling or rate limiting on key agreement is a necessary feature of DoD IoT devices that utilize a PKI scheme for key distribution. An implication of throttling is that the system have some concept of the passage of time, which places added requirements on the system composition.

IoT devices are typically required to use very little electrical power, which is achieved by a combination of efficient circuitry and special modes that put the device to "sleep." Such modes require a wakeup signal that can be supplied by real-time clock (RTC). That clock can also be used to maintain time for crypto functions (primarily key expiration). RTCs are very low power. One example is the ST Microelectronics M41T62 [38], which draws 350nA while keeping time and can run for over 600K hours on a CR2032 coin-cell battery.

4.2.3 Sustainment

The DoD's definition of sustainment is: "The provision of logistics and personnel services required to maintain and prolong operations until successful mission accomplishment." [40]

We divide the sustainment of DoD IoT systems into four broad activities:

Maintenance of Functionality – Ensuring proper composition of the broader IoT system (e.g., which devices are connected to maintain operational effectiveness). In the system in question, this involves fixing or removing broken devices and replacing them with newer functional devices. Since these systems will be exposed to a combat environment, a high degree of breakage should be assumed despite ruggedization. In order to have maximum functionality of the system, all devices on all members of a squad should be instrumented and that information made available to decision makers in real time. This requires that all soldier-mounted gateways be operational and all sensors on those gateways, including the wirelessly connected rifle sensor, be operational.

Maintenance of Connectivity – Our IoT system must maintain connectivity in order to be useful. It must be able to operate wirelessly outside of the continental United States. This means that frequencies and modes must be compatible with local standards. Failing that, interference could become a serious concern. The CONOPS for our example case uses a combination of low-power, high-proximity communications, a higher-power 802.11 Wi-Fi standard, and CDL. The low-power link between the rifle and the gateway would likely utilize a standard such as Bluetooth or Zigbee, each of which operate in the 2.4 GHz Industrial, Scientific, and Medical (ISM) bands. Zigbee also operates at other frequencies for compatibility in some regions. Wi-Fi also has modes in the ISM band, but the newer IoT-specific modes 802.11ah/f operate at 900 MHz and below. 802.11.af utilized cognitive radio [41] techniques which would be very useful in maintaining connectivity not only in diverse geographic regions, but also as resilience against jamming.

Maintaining security – Ensuring that the right cryptographic key material is in the right places to enable functionality of the broad IoT system. This requires periodic rekeying and inclusion/exclusion of devices in the trusted network. To do this, the maintaining organization must know precisely where devices are in order to establish whether they can still be trusted. For instance, a lost device may necessitate rekeying

an entire set of connected devices. In the case in question, use of a PKI scheme permits selective trust/distrust, whereas a scheme that uses pre-placed traffic keys could require rekeying of the entire IoT system. For instance, if an aggregator is lost, all devices that connected securely to that aggregator must ignore communication from the lost device, and establish communication with a new one. Loss of a soldier's gateway would require a rekey between the aggregator and the new gateway. The old gateway would have to be blacklisted to prevent an adversary from accessing the network. Loss of a rifle would have similar effects and require a repairing of the rifle and the soldier's gateway.

Maintaining compatibility – As new sub-systems and upgrades come online, making sure that they are compatible and upgrading software and hardware as appropriate. New versions of commercial systems are released frequently. Due to the stringent security and compliance requirements of DoD systems, the system in question would probably not require updates nearly as frequently primarily due to the complexity of the processes involved in creating such upgrades. However, the presence of upgradability functionality greatly complicates sustainment, as there must be a way to securely and quickly update systems.

One of the hallmarks of an IoT system is composability. Our soldier-mounted sensor must be able to attach securely to different aggregators as needs change on the battlefield. Also, when one group of soldiers merges with another group, or reinforcements need to be integrated, compatibility will become a serious concern. This is especially true for devices with a high probability of loss.

4.2.4 Decommissioning

Devices that reach end-of-life must be removed from service. For IoT devices or any device that engenders trust, that trust must be revoked permanently. In addition, any national-security assets such as cryptographic keys must be securely and completely destroyed. Even if those keys are long expired, they can still be used to retroactively decrypt stored messages. Finally, the device must be disabled such that it cannot be repurposed by the enemy.

5. SYSTEM DESIGN

In this thought exercise, in which a "soldier-as-a-sensor" IoT system is requested, the winning bid proposes to build four sensors that communicate over a soldier's body using a combination of wired interconnect and wireless channels as shown in in Figure 3. A soldier will be instrumented with: A heart rate monitor, a temperature sensor, and a six-axis accelerometer. In addition, the soldier's rifle will be fitted with a "Shot-counter" that will link wirelessly using the ZigBee standard, defined by IEEE 802.15.4 [42]. These sensors would communicate with a body-mounted gateway transmitting using 802.11ah Wi-Fi [43] and 6LowPAN[44], a compressed IPv6 protocol built for low-power networks. The signal from the personal gateway is detected by an 802.11ah access point where it is retransmitted via a CDL modem to the command structure.

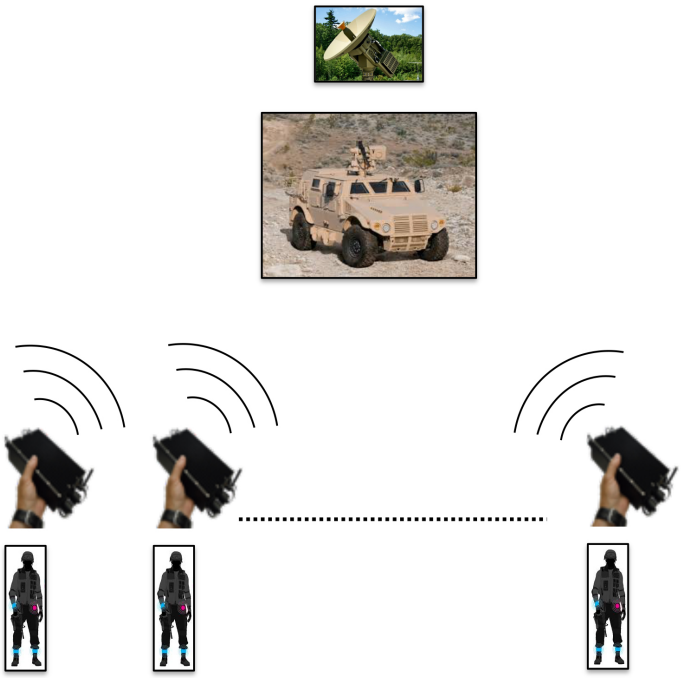

Figure 3. A squad of soldier-sensors communicates to an aggregator, which communicates to decision makers.

All soldier-mounted sensors will communicate with a wearable computer called a Data Aggregator that is also worn by the soldier. The body-mounted sensors will communicate with the aggregator with a standard serial i2c bus [45]. The aggregator will authenticate and transmit with high confidentiality and integrity the soldier's state to a mobile gateway device using an IoT amenable Wi-Fi standard such as 802.11ah.

Each soldier's aggregator device will have a secure, tamper-resistant root-of-trust that is used to connect to a gateway running on a mobile system that is connected via Ethernet to a CDL modem for satellite uplink.

The individual sensors on the body must be built to use low power for longevity in the field. This means that their aggregate energy budget must consume less than the soldier's battery capacity over the time of use. Locally, wireless communication can be quite costly from an energy perspective – ~3.5uJ/bit[46] at 1kb/s data rates. Using a small 160 maH lithium polymer battery [47], we find that the absolute maximum data that can be transmitted assuming perfect (and complete) use of all capacity is 164.5 Mb. Of course this assumption leaves no energy for data collection. The take-away is that wireless communication can severely impact SWaP and should be used sparingly, therefore when possible, a wired connection should be used to connect sensors. Another reason to avoid wireless technologies when possible is security. With the exception of near field communication (NFC) which features ranges in the centimeters, most IoT protocols have ranges in the tens of meters. This means that to connect across a body, one might have to broadcast such that an adversary 30 feet away can intercept the communication. That adversary could also attempt to register as a valid sensor to fool the eventual consumer of the information. Connecting sensors on a body with wires is reasonable. This decision also impacts the overall IoT threat model in that now, rather than having multiple sensors, each of which can be classified as a separate IoT sub-system, we have a single IoT system (the gateway) with a number of connected sensors (each of which can be significantly less sophisticated than if they needed a wireless channel for connectivity). Generally, complexity is anathema to security so it is reasonable to say that this design decision makes the system more secure.

Where does that complexity come from? It comes from the need to authenticate and protect a wireless communication link. IoT systems rarely have directivity in their transmitters, therefore it is difficult to say to whom you have connected with a link without the help of a shared secret between two devices. Salient design questions include:

- Where does the secret that is used to authenticate two devices come from?
- How is the secret stored when the device is off-line?
- How is secret used to authenticate?
- How is the secret invalidated after compromise or as part of normal operation?

In the case of the soldier's rifle, a wire to the soldier's gateway is probably not reasonable, so we must have a wireless telemetry channel. One of the critical facets of DoD IoT design is illustrated in the design decisions in this sub-system: the tension between security and usability. A soldier does not always carry his rifle. He may not even use the same rifle every time he reports for duty (the same can be said for the body monitoring system). The soldier needs to be ready for duty with all of his gear in a timely manner. Mission planners and commanders want situational awareness. Both of these stakeholders want the process

of meeting mission goals to be reliable and effective. The soldier wants to pick up a rifle at an armory and be able to use it immediately. The planners and commanders want to know that the soldier is battle-ready as soon as possible, so the communication link between the soldier and headquarters should be established immediately. In addition, soldiers who may make life and death decisions at a moment's notice have a low tolerance for added complexity, especially if that complexity does not obviously enhance their effectiveness. The need for a wireless channel injects a great deal of complexity into any design. The first problem to solve, however, is how to add or remove a device from a trusted network. The process by which an IoT device is connected to the larger system is called *enrollment*.

5.1 ENROLLMENT

Enrollment requires the establishment of trust between an IoT device and the broader IoT system. In our example system, this means establishing trust between the rifle and the soldier's gateway. Implicit in this is that the soldier himself is authenticated to the system also, as the soldier is now trusted to input valid information (biometric data, rate of fire). The fact that connected devices can be lost, stolen, or traded vastly complicates authentication and starts to move us away from the "pick-up-and-fire" capability the soldier expects. Commercially available technologies to perform this kind of enrollment or pairing almost invariably depend on the use of a low-bandwidth side channel for authentication. The principle vulnerability such schemes avoid is the initial insecurity of a newly established RF channel. These techniques, described here [48] depend on at least one of several properties:

1. Possession – Some technologies require pressing a button to put a device into pairing mode.

2. Proximity – An ancillary, extremely low-range RF channel is used to exchange an initial shared secret

3. Activity – A code is provided on the device that is entered into the system that it is attaching to. The code serves as a bootstrap secret

Each of these three techniques imposes SWaP and usability constraints. This is a prime example of the tradeoff between security and usability.

Proof of possession of both devices can be seen in commercial IoT devices that connect via Bluetooth Pairing. The act of pressing a button on each device conveys intent on the part of the user. This method is often used with a low-bandwidth, low-range channel. That channel can be an alternate wireless channel, a lower-power mode of the primary channel, or an interaction with a user. That interaction usually constitutes an activity in which a user types the same code into one or both devices to derive a secret session key to encrypt the primary communication channel. Any solution that requires user input, however, requires an interface. In some cases, that interface may be as simple as a button on each device. However, this would probably not support a secure solution as proof of possession is not particularly useful to authenticate the devices (or the user who is trying to pair them). A better solution would require that each device have a

securely held identity and the ability to verify the identities of other devices. There exists a wide variety of pairing mechanisms in the Bluetooth specification as described in NIST Special Publication 800-121 [49].

Such a solution would enable a great deal of flexibility when enrolling devices into the broader system, as ostensibly the secret value would have been provisioned by the manufacturer. Since almost all cyber-security is based on the existence of a shared secret, the capability to have all devices possess one is powerful indeed. The problem arises when those devices fall into the hands of the enemy. We assume an enemy with state-level resources, therefore, has the capability to extract all data from a device (in this case, a rifle) given enough time is assumed. If an adversary obtains the identity of the device, she can impersonate the device, and enroll a device of her choosing into the IoT system. Therefore provisioning devices with a shared secret usually requires an increase in SWaP, as volume protection or even active circuitry may be necessary to protect those secrets adequately. In addition, the security of that shared secret may be jeopardized by improper handling at many points in the lifecycle of the IoT sub-system, or the full system as a whole. For instance, we will assume that the secret value is "burned" into the IoT device somehow. There exist many technologies to do so, from so-called eFuse [50] to integrated NVRAM [51]. A secret that is injected exists in at least two places (inside and outside the device) and must be protected, first by the manufacturer, then by the operator. Further, if the operator wishes that other devices communicate with our device, the secret must be shared with them. Even if that secret is encrypted locally, the web of trust has just expanded to include all of the devices that possess the secret.

Another method for provisioning identities is to never allow the true identity of the device to ever leave. This may sound counterintuitive, but the capabilities afforded by PKI make it possible. In PKI systems, each device has two identities: a public one and a private one. The public identity, a large bit vector of hundreds to thousands of bits, is mathematically related to the private identity, another bit vector. This relationship is described in the mathematics of asymmetric cryptography, which are structured such that deriving the public key from knowledge of the private key is easy, but the reverse process is prohibitively difficult without some extra information. This means that a device with a private key can export a public key as its public identity while keeping its private identity secret. The public key is distributed in a standardized data structure called a certificate. Certificates can be forged however, so to ensure the *authenticity* of the public key (i.e., the property that the public key is paired with the private key whose possessor is trusted), the certificate is *digitally signed.* Digital signatures are generated by executing a mathematical function that combines a cryptographic hash of a message, a random nonce (number used once) and the private key. A signature can be verified by executing another function that takes a public key and a hash of the message. Verification of the signature with the public key proves that the signer had the corresponding private key. The signature used to verify the authenticity of the certificate is the certificate

of a mutually trusted entity such as the organization that provisioned and deployed both devices. This public key must be stored unmolested in any device that wishes to authenticate another.

PKI is a powerful technology that underpins the security of all of the modern internet. The downside of using PKI in the IoT system realm is that it is very compute-intensive. The graph in Figure 4 shows the

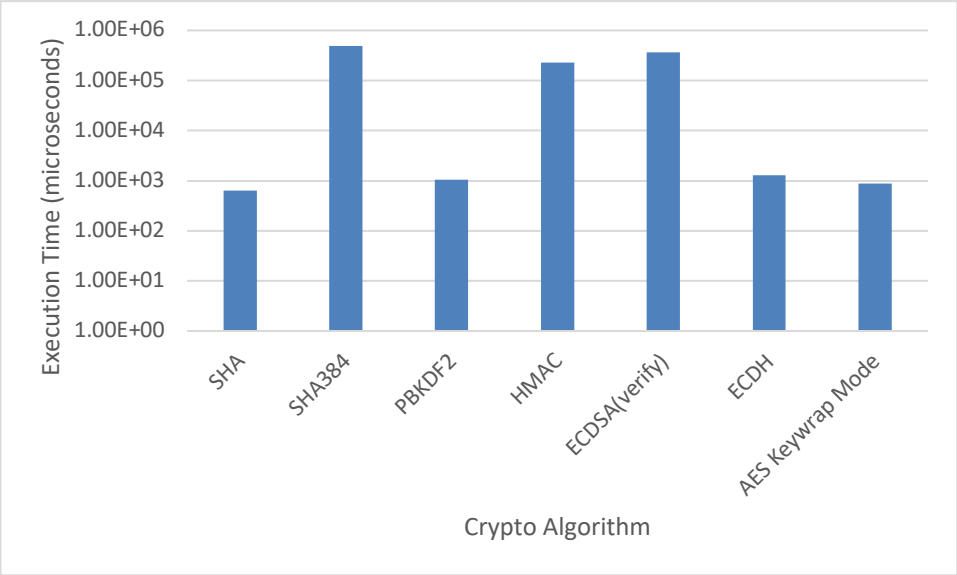

Figure 4. Time to execute a number of common crypto primitives on an embedded processor.

time required to execute a number of cryptographic primitives including Elliptic Curve Cryptography (ECC), an asymmetric cryptography technique. The processor this is executing on is a 32-bit ARM Cortex A-53 processor, which approaches the top-end of what would be used in a small IoT device. There exist many small processors in 8 and 16 bit architectures that could also run these algorithms, but their performance will suffer super-linearly, as breaking a 32 bit operation down into 8 bit operations does not incur only a 4x hit to performance. It is for this reason that many researchers have shunned this technology for IoT despite its promise. We believe this to be somewhat misguided however. Often, the point is made that a processor is simply not powerful enough to execute these operations in a timely manner. This ignores the fact that what constitutes a timely manner is highly subjective. Key agreement, the process of arriving at a shared secret between two parties and which is the critical point in enrollment, need not be executed often. If the rifle takes 30 seconds to pair with the soldier's gateway in a depot, before mission start, the lag does not impact the soldier's effectiveness. If however, enrollment needs to happen during a mission, at best the soldier and command structure is inconvenienced, at worst, the monitoring capability afforded by the system is unavailable, perhaps purposefully (if an adversary can somehow force a re-enrollment).

Processing power is not the only limitation in small devices however. Some of the most popular embedded microprocessors are also extremely limited in terms of RAM. The smallest, such as the

ATMega328, popular in the Arduino [52] ecosystem, have 2k bytes of RAM. Such devices are truly SoCs in that they are self-contained and include the processor pipeline, RAM, NVRAM, and even a clock on the same chip. Therefore the option to expand any of these resources is severely limited. Though devices at this scale are extremely attractive, they are also extremely limited for processing loads that exceed this amount of memory, such as PKI schemes. The still popular RSA algorithms typically have keys that exceed 2048 kb in length, and may hit as high as 4096 kb. That means that just the key alone can take up a quarter of the RAM available in small devices. A better option is the aforementioned Elliptic Curve approach to asymmetric cryptography. ECC keys are much smaller, and currently top out at 384 bits for government-approved implementations.

Enrollment of the overall soldier-mounted system that includes the gateway, attached sensors, and smart rifle may be easier however, as such a system could be made with the same complexity as a smartphone. Smartphones are already more than capable of operating as full-fledged devices on the mobile internet. That is, they are capable of significant processing, which includes the use of PKI. They also do so with relatively small batteries which could be made significantly larger without noticeably adding to a soldier's load. Even in this environment, the overhead of more advanced cryptographic operations is small relative to the available processing power.

If we assume that each soldier has a PKI-enabled gateway connected to a centralized aggregation point via a long-range IoT-friendly Wi-Fi and using a protocol-compression scheme such as 6LowPAN, all devices in the network need to store:

- A private key for which confidentiality and integrity are vital
- A public key embedded in a signed certificate for which integrity is critical
- A public key from a trusted authority with which the system can verify the authenticity of other system's certificates, stored with high integrity, though not confidentiality

DoD systems generally use NSA generated key material. Because those keys are themselves considered SECRET or TOP SECRET, they must be adequately protected. This protection must be both logical (i.e., protection from malware) and physical (i.e., protection from physical tampering). Technology to physically protect a device is usually referred to as anti-tamper (AT), and has significant implications on the physical design as well as the logical design. Some AT technologies require constant power to maintain security, which is a significant burden in low-power designs.

Because the NSA has not yet come to a usable notion of "Transient Crypto," or disposable crypto, in which key material is used so sparingly or for such a short duration that the burden of protecting that material is smaller, we assume that our DoD IoT system would use NSA secret keys, and would therefore include all necessary protection (logical and physical) to achieve NSA certification for use on the battlefield.

6. THREAT ANALYSIS

In this section, we will identify the participants in the scenario as well as their individual goals. Next we will analyze the system using the model described earlier.

6.1 PARTICIPANTS

Squad leader – The squad leader is responsible for leading the members of the squad. He will be the aggregation point for all information from the squad elements. He will use some of this information to make decisions on tactics.

Squad elements – Each element is a single instrumented soldier. Their information is fed back to the squad leader via a connection between the element's gateway and the squad leader's aggregator. The squad leader orders squad elements to take action.

Mission commander – A command element that may control multiple squads. The commander receives information from the aggregators held by each squad leader. This information is used to make tactical and strategic decisions, including mobility and resupply. If a squad is running low on ammunition, the mission commander will know based on the sensors on each soldier.

Adversary - The adversary may attack the system at multiple levels using electromagnetic or physical means.

6.2 GOALS

Squad leader – To effectively and safely meet mission goals by directing his squad. The squad leader identifies the critical needs of his squad with information gleaned from his squad's sensor network, and information fed down from the mission commander.

Squad elements – To safely meet mission goals as directed by the squad leader. The squad elements receive situational awareness from the sensor net filtered through the squad leader.

Mission commander – To use all squads under his command to safely meet mission goals. The mission commander does this by maintaining situational awareness via voice communication and information gleaned from the soldier-borne sensor net.

Attacker goal – To disrupt the commander, squad leader, and elements' ability to meet mission goals. The attacker will do this by disrupting the flow of information to decision makers, or by injecting incorrect information that, if believed, will be advantageous to the attacker. Disruption of information will result in a loss of situational awareness.

6.3 ATTACKER METHODS

Blinding – The attacker may simply wish to remove the squad's ability to communicate their status by jamming the communication medium.

Infiltration – Attempting to enter an attacker's device on the IoT network as a valid device to disrupt or otherwise inject bad information.

Co-option – Attempting to "hack" a legitimate piece of the system to disrupt or otherwise inject bad information. This attack abuses already-formed trust relationships

Re-purpose – Hack a legitimate piece of the system to use it in an attacker's network.

6.4 ANALYSIS OF THE SYSTEM DESCRIBED

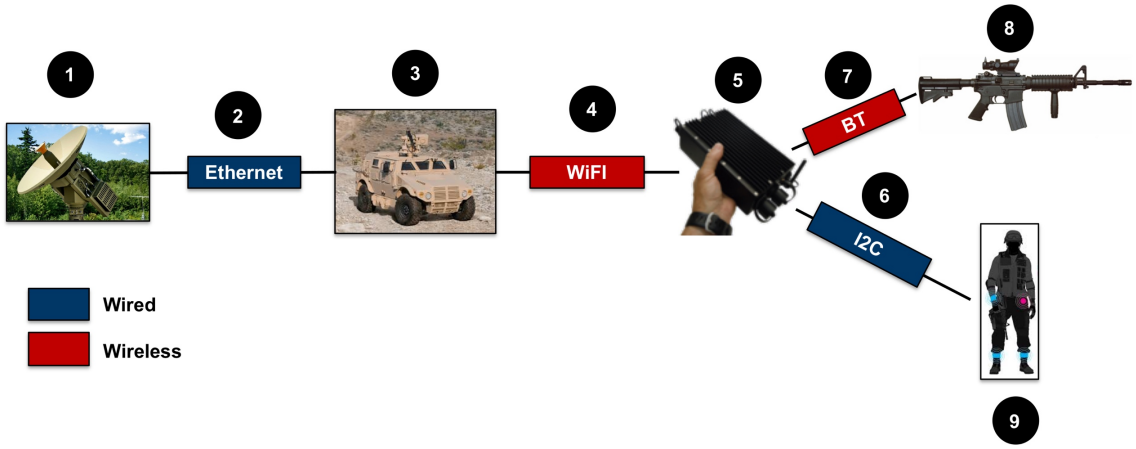

Figure 5. The soldier-as-a-sensor system decomposed in the comput/communicate model.

A high-level diagram of the system in our compute/communicate style is shown in Figure 5. There, we show nine points in the minimal IoT system that must be analyzed. Of course, in reality, points 5 through 9 will be multiplied many times over as every squad member will have one.

Uplink – The uplink uses the Common Data Link (CDL) waveform specification and an NSA-certified CDL modem. These modems utilize strong encryption and generally receive keys in the red, meaning that they are keyed by hand at a depot with keys supplied by the NSA. Assuming they are properly keyed and encryption is enabled and not bypassed, these devices are assumed to be secure.

Wired transport to system aggregator – The communication with the aggregation device is Ethernet in our system. Ethernet itself does not enforce encryption, therefore this communication medium may be prone to attack. Generally, this could be achieved by placing an unauthorized device between the modem and the aggregator on the truck. Such a device could then **blind**, **infiltrate,** or **co-opt** the overall system.

Aggregator – The aggregator is a compute device with both wired and wireless connectivity. It can be attacked via either of these interfaces (attacks on the wired interface require that the Ethernet connection be compromised). The probability of unreported loss on this system is fairly low as, in this design, it is installed in a large piece of equipment. Therefore physical tampering attacks are not likely without tampering being reported. However, network-borne attacks, especially via the wireless channel, could be mounted to affect aberrant behavior of the embedded computer. Compromise of the aggregator can be used to **blind**, **infiltrate,** or **co-opt** the overall system.

Wi-Fi Channel – The squad-level Wi-Fi channel is probably the most exposed element of the system. The chosen IoT-style Wi-Fi standards use frequencies below the 2.4 GHz ISM band to achieve longer range. Therefore contact between the squad elements and the enemy implies contact between the enemy and the wireless network. Even without direct contact, surreptitious attack is possible. Compromise of the wireless channel can be used to **blind**, **infiltrate,** or **co-opt** the overall system.

Gateway – The gateway has a high probability of loss, which necessitates a deeper look at the design as more interfaces are exposed. The gateway, like many of the devices in this system, is an embedded system, therefore many of the threats described below apply to points 8 and 9. A exploded system is shown in Figure 6.

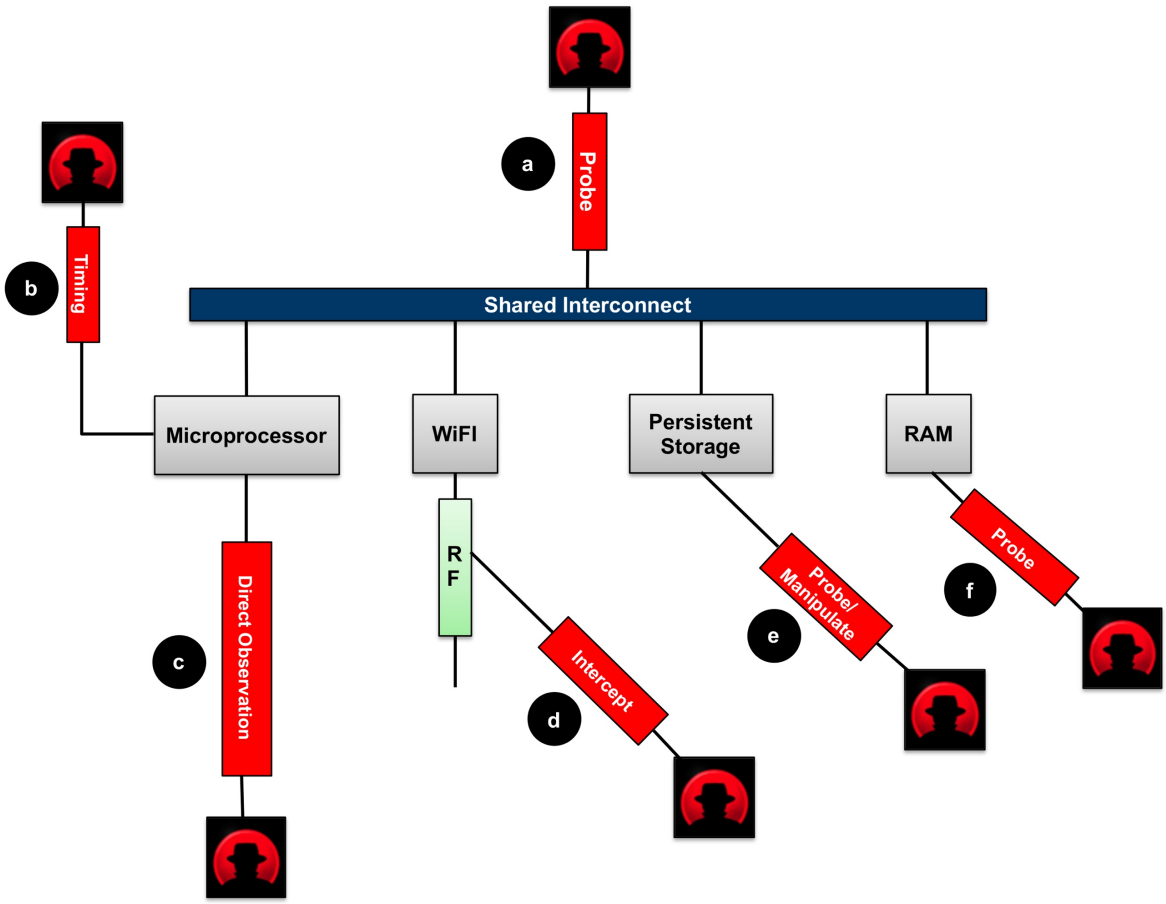

Figure 6. Threats against the local (soldier-wearable) computer.

An adversary attacking at this level wishes to either inject or affect code running on the microprocessor, or exfiltrate security parameters (usually cryptographic keys) such that he can compromise the wireless communication of the system from the aggregator down. One of the most enticing attack vectors is simple probing of the board-level interconnect. Famously, this was done with great success to hack the original X-box [35]. Doing this can yield secret keys because the processing engine must get keys

from persistent storage (Non-volatile RAM). Many microprocessors still do not possess internal persistent storage, meaning that key material must be exposed in plain text at some point.

Microprocessors can be non-invasively probed if the adversary can get their own code to run on the microprocessor. Timing attacks (b) utilize side effects of multiple processes running on the same hardware. This kind of attack benefits heavily from features meant to increase efficiency, such as symmetric multi-processing, in which multiple simultaneously executing processor cores share a common resource such as a cache memory. By measuring access times to that memory, an adversaries program can glean critical information about other programs running on the system, even if they are running at a higher privilege level.

Of course, if malware is able to "pop root" (c), or improperly escalate privilege, such indirect methods are unnecessary, and secrets can be directly obtained and exfiltrated. Such malware can enter the system via standard communication interfaces such as Wi-Fi. Access to the external Wi-Fi channel is a significant vector into the machine (d). This is one of the aspects of IoT that is so dangerous in DoD systems: Any adversary can attempt to communicate (and corrupt) the IoT devices.

An adversary with physical access to the gateway, as would occur when a device is lost, may be able to extract critical information by directly probing memories such as RAM (f) or non-volatile (persistent) storage (e). These methods are easy to detect if the system is still at least partially in a soldier's control. However, a lost device could effectively have its identity stolen and used by adversary's device.

The level of control gained from any of these vectors can lead an adversary to **blind**, **infiltrate,** or **co-opt or repurpose** the **device.**

This page intentionally left blank.

7. RECOMMENDATIONS

The activity of designing a DoD-style IoT system exposed a number of critical technology gaps. These primarily centered around how to mete out trust in dynamically changing, wirelessly connected sets of low-power sensors and actuators. This is not substantially different from the challenges in commercial systems. However, the environment in which DoD IoT systems live is makes a great deal of difference.

Some salient environment considerations:

- DoD IoT systems must operate within communication distance of state-sponsored actors
- Compromise of DoD systems results in strategic or tactical advantage to adversaries
- People can die when DoD systems are compromised
- DoD systems often operate in communications-challenged environments
- The exact location of DoD IoT devices may not be known

The DoD is a massive enterprise with many working parts that must function together to meet mission goals. As such, the infrastructure required to support any security-related features is a vital consideration when considering the feasibility of sustaining the many small devices that make up an IoT. Sustainment includes software and hardware updates, cryptographic key fill, user management, and configuration management. Some considerations on sustainment include:

- Ensuring that all available IoT elements are compatible will be challenging as different weapons systems will have different upgrade roadmaps
- Key management of tactical devices is still often done “in-the-red,” and utilizes an established infrastructure, both materiel and non-materiel, to key equipment in the field.
- Devices that travel with troops may be far away from the maintenance depot for long periods of time.
- IoT devices may be used by multiple soldiers, complicating authentication

These key differences will result in different designs for fielded DoD systems. The commercial IoT area has been able to grow in an organic manner. As technologies have been miniaturized, batteries improved and wireless technologies more prevalent, new devices have come online individually. The concept of a “Smart house” is not owned by any one company selling sensors and actuators (though most companies would love to corner the market). Rather, Nest thermostats are being connected to home networks with Schlage smart locks and D-Link web-enabled security cameras. In effect, there is no real architecture to commercial IoT. DoD systems are not generally put together in an ad-hoc manner as there is simply too much riding on their successful operation. However, historically, these requirements have resulted in “Stove-piped” solutions with minimal interoperability. While stove-piped IoTs may satisfy

many of the characteristics of an IoT, such a thing invalidates one of the major benefits of IoT, which is reliance on relatively standard interfaces and methodologies.

Recommendation #1: MIT LL should continue to push the use of open-architecture techniques and technologies into the low-power embedded space.

One major aspect of this is to enable the use of commercial COTS components and systems in a DoD context. While systems will still need to be architected, they can be architected and deployed *faster*. This should involve input to IoT communication and cryptography standards. As was mentioned earlier, the DoD has unique challenges in the distribution of cryptographic key material. To realize the promise of flexibility in a DoD IoT system, we need to be able to trust our data. This means that data must be stored and transmitted with at least high integrity, and in most cases confidentiality also. Integrity requires low-power integrity-protected cryptographic modes. Note that this is a bit of a departure from the current focus of confidentiality first, and integrity second. IoT tends to aggregate data into useable information. The sensor data is not necessarily sensitive or secret, but unauthorized modifications of that data can influence the end information product (which itself may be sensitive or secret).

Recommendation #2: Focus on authentication and integrity at the IoT end-points over bulk encryption.

Both encryption and authentication/integrity require a root secret key, however, they use those keys differently, resulting in different key lifetimes. In addition, the processing power required to encrypt vs. authenticate (say, using a Message Authentication Code) will be different. Nonetheless, the DoD needs to move to a paradigm that permits more key agility—that is, a capability to locally manage ephemeral keys and renegotiate/regenerate when necessary. This capability needs to be supportable in low-SWaP environments. Symmetric crypto algorithms have been developed in this direction (See Simon and Speck [53]), but limited progress has been made toward asymmetric schemes.

Recommendation #3: Develop technologies for low-power asymmetric key management facilitating the use of short-lived keys.

Asymmetric algorithms that make up most PKI schemes are founded on the concept of a private secret that serves as the identity of the holder of that secret. The secret is used for key agreement between mutually trusting devices and for authentication. Ideally, that secret is an immutable property of each device, making it impossible to change and difficult to forge. That identity must be used sparingly, reducing exposure to adversaries. In addition, it should be easily used and not exposed to software running on the device. The function to authenticate based on the native identity should be exported to the software, but the mechanics of that process, including the key/identity that was used to authenticate, should remain hidden.

Recommendation #4: Develop technologies and standards for the generation, protection, and use of immutable silicon-level IDs.

One of the most powerful characteristics of IoTs is their flexibility. However, reconfiguration and merging of IoT systems to provide an aggregate capability requires that those systems identify and trust

each other. Doing this in an efficient and usable manner is difficult, especially within large networks of heterogeneous systems. Given many of the challenges unique to DoD systems, deriving a secure, scalable and usable method to allow systems to trust one another is vital.

The process of introducing a new device into an IoT network is commonly called *enrollment.* Ease-of-use in enrollment requires not only a firm basis with which we can trust devices, verifying their provenance and integrity, but also an easy and intuitive way to initiate enrollment without substantially increasing the size, weight, and power of the system.

In DoD systems, ease-of-use is critical, as soldiers typically have little tolerance for unintuitive systems (they have other things to worry about). DoD IoT systems must be resilient to changes in ownership, and facilitate the "mass merge" of multiple devices, as when two squads are combined into one, or must work together.

Recommendation #5: Focus on technologies and standards to ease the burden of enrollment in DoD IoT systems.

Trusting systems requires more than verifying ownership, and that the system attempting to enroll in the network is the exact silicon it says it is. Almost all systems today derive great flexibility by encoding advanced functionality in software. The authenticity, provenance, and integrity of that software is of paramount importance. Bad software on one device can result in an infiltrated network, regardless of the integrity of the local identity.

Therefore, a method to convey the integrity of the software (which can be based on the integrity of the hardware) is necessary. Work has been done in the area of secure execution, or execution in which the integrity of the system is provable [54]. But these techniques are far from standardized. In addition however, even these techniques are only as good as the software they are trusting. Therefore, a standardized way to securely update code on DoD systems is necessary.

Recommendation #6: Develop and standardize a secure update and execution environment and methodology for DoD systems.

The static integrity of a system is important, but a record of the actions taken by that device, as well as any configuration changes and the date and provenance of those changes, is necessary to understand the aggregate security posture of the whole IoT system. This requires a way to securely audit and validate the state of low-level devices even over low-bandwidth, challenged links.

Recommendation #7: Secure auditing of low-level IoT devices.

This list is certainly not exhaustive, but it highlights a number of critical gaps or areas for improvement whose remedies would greatly facilitate the expansion of IoT technologies within the U.S. DoD.

This page intentionally left blank.

8. CONCLUSION

The Internet-of-things is increasingly relevant in the commercial world. We are rapidly approaching a time where the IoT should be considered *the* Internet. We distinguish between the IoT and the Internet as a whole because IoT changes the nature of the Internet from an information-passing medium to an information-passing *and control* medium. The information passed is no longer simply web pages or Voice-over-IP traffic; it is data from sensors spread around the planet that can be fused with other data to provide actionable information. IoT devices do not have to be connected to the broader Internet. They can be connected to air-gapped sub-networks hosting arbitrary protocols and still have the same effect: distributed sensing and control of day-to-day and even life-critical functions.

IoT is the result of simultaneous technological improvement in numerous areas, from low-power electronics, system integration, and wireless technology to long-term storage, displays, and battery technology. Regrettably, security has not kept pace with these technology improvements, leading to a massive amount of insecure devices being connected to the Internet. In the commercial world, we have seen these devices used to spy on unsuspecting people via compromised webcams, and IoT systems weaponized to launch massive DDoS attacks.

As useful as these devices are, it should give us pause when considering their use in the DoD. Of course, when we speak of DoD IoT, we are not really talking about commercial webcams or D-Link routers (though their use in DoD systems is not totally unrealistic), but distributed systems of sensors and actuators that enable efficient and effective use of military force. In the commercial world, an insecure sensor or actuator may be safe simply by virtue of there being many other similarly insecure devices to compromise. Unfortunately, there is only one DoD, one target, and huge value to compromise its systems.

Avoiding IoT technologies for these reasons is no longer a possibility. As we have shown, many systems are now wirelessly connected, low-power, and capable of sensing and actuating their environments. In any event, the trend toward using COTS devices, often designed for the commercial world, continues. Through an analysis of current systems, as well as the design exercise of a soldier-as-a-sensor system, we have identified a number of critical gaps that must be addressed for the DoD to securely scale-out IoT architectures. These are mostly technologies, but it is important to not forget that the DoD is a huge enterprise that must sustain these systems over time. Therefore, technologies that enable IoT systems in the DoD must be accompanied by easy-to-use policies and procedures for maintenance

This page intentionally left blank.

REFERENCES

1. https://nest.com/, Retrieved 1/29/18.
2. http://www.lg.com/us/discover/smartthinq/thinq, Retrieved 1/29/18.
3. https://www.schlage.com/en/home/products/products-smart-locks.html, Retrieved 1/29/18.
4. http://www.belkin.com/us/Products/home-automation/c/wemo-home-automation/, Retrieved 1/29/18.
5. Kambourakis, G et al. "The Mirai Botnet and the IoT Zombie Armies," in the proceedings of MILCOM 2017.
6. Kushner, D. "The Real Story of Stuxnet," IEEE Spectrum Volume: 50, Issue: 3, March 2013.
7. Holliman, J. "Building low-power trustworthy systems: Cyber-security considerations for Real-Time Physiological Status Monitoring," proceedings of MILCOM 2016.
8. https://www.gsa.gov/real-estate/facilities-management/gsa-smart-buildings, Retrieved 1/29/18.
9. http://www.cnn.com/2017/07/17/politics/us-navy-drone-laser-weapon/index.html, Retrieved 1/29/18.
10. http://www.janes.com/article/60546/general-atomics-commits-private-funding-to-develop-10-mj-medium-range-railgun, Retrieved 1/29/18.
11. http://www.janes.com/article/60546/general-atomics-commits-private-funding-to-develop-10-mj-medium-range-railgun, Retrieved 1/29/18.
12. http://www.af.mil/About-Us/Fact-Sheets/Display/Article/104470/mq-9-reaper/, Retrieved 1/29/18.
13. http://www.army-technology.com/projects/irobot-510-packbot-multi-mission-robot/, Retrieved 1/29/18.
14. https://www.defense.gov/Portals/1/Documents/pubs/Perdix%20Fact%20Sheet.pdf, Retrieved 1/29/18.
15. https://iadgov.github.io/simon-speck/papers/simon-speck-asic-2014.pdfD. Dinu et. al "Triathlon of Lightweight Block Ciphers"

16. Dinu, D et al. “Triathlon of lightweight block ciphers for the internet of things,” Cryptology ePrint Archive, Report 2015/209, 2015. http://eprint.iacr.org/

17. Mirzadeh, S. et al. “Secure Device Pairing: A Survey,” Communications Surveys & Tutorials, IEEE. 16. 17-40. 10.1109/SURV.2013.111413.00196.

18. Seshadri, A. “SAKE: Software attestation for key establishment in sensor networks,” in the Journal of Ad Hoc Networks, Volume 9, Issue 6, pp1059-1067, August 2011.

19. “Russian hackers tracked Ukrainian artillery units using Android implant: report,” https://www.reuters.com/article/us-cyber-ukraine/russian-hackers-tracked-ukrainian-artillery-units-using-android-implant-report-idUSKBN14B0CU, Retrieved 1/26/18.

20. “Exclusive: Iran hijacked US drone, says Iranian engineer,” Christian Science Monitor, https://www.csmonitor.com/World/Middle-East/2011/1215/Exclusive-Iran-hijacked-US-drone-says-Iranian-engineer, Retrieved 1/26/18.

21. Manadhata P. K. and Wing, J. M. “An Attack Surface Metric,” Software Engineering, IEEE Transactions on, vol. 37, no. 3, pp. 371 –386, Jun. 2011.

22. Pendleton, M. et al. “A Survey on Systems Security Metrics,” ACM Comput. Surv. 49, 4, Article 62 (December 2016), 35 pages. DOI: https://doi.org/10.1145/3005714

23. Dodds, J. “Formal Methods and the KRACK Vulnerability,” https://galois.com/blog/2017/10/formal-methods-krack-vulnerability/, retrieved 1/26/18.

24. Mount, M. and Quijano, E. “Iraqi insurgents hacked Predator drone feeds, U.S. official indicates,” http://www.cnn.com/2009/US/12/17/drone.video.hacked/index.html, retrieved 1/26/18.

25. Anderson, R. J. and Kuhn, M. G. “Tamper Resistance - a Cautionary Note,” The Second USENIX Workshop on Electronic Commerce, Oakland, California, November 18-21, 1996.

26. Courtland, R. “3D X-ray Tech for Easy Reverse Engineering of ICs,” https://spectrum.ieee.org/semiconductors/processors/3d-xray-tech-for-easy-reverse-engineering-of-ics, retrieved 1/26/18.

27. Kocher, P. et al. Differential Power Analysis. In: Wiener M. (eds) Advances in Cryptology — CRYPTO’ 99. CRYPTO 1999. Lecture Notes in Computer Science, vol 1666. Springer, Berlin, Heidelberg.

28. Andress, J. Chapter 1 - What is Information Security?, In The Basics of Information Security, Syngress, Boston, 2011, Pages 1-16, ISBN 9781597496537, https://doi.org/10.1016/B978-1-59749-653-7.00001-3.

29. Nahill, Ben et al., "Towards a Universal CDAR Device: A High-Performance Adapter-Based Inline Media Encryptor," In the proceedings of MILCOM 2017, Baltimore, MD.

30. https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/hh831627(v=ws.11), Retrieved 1/29/18.

31. Hoppe, T. et al. "Security threats to automotive CAN networks–practical examples and selected short-term countermeasures," International Conference on Computer Safety, Reliability, and Security. Springer, Berlin, Heidelberg, 2008.

32. Bernstein, D. J. "Cache-timing attacks on AES," (2005): 3.

33. Kocher, P. et al. "Spectre Attacks: Exploiting Speculative Execution," arXiv preprint arXiv:1801.01203 (2018).

34. Gandolfi K, Mourtel C, Olivier F Electromagnetic analysis: Concrete results. Cryptographic Hardware and Embedded Systems. Springer Berlin Heidelberg. pp. 251–261. doi:10.1007/3-540-44709-1_2

35. Huang, A. "Keeping secrets in hardware: The Microsoft Xbox TM case study," International Workshop on Cryptographic Hardware and Embedded Systems. Springer, Berlin, Heidelberg, 2002.

36. https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation, Retrieved 1/29/18.

37. Yang, K, et al. "A2: Analog malicious hardware," Security and Privacy (SP), 2016 IEEE Symposium on. IEEE, 2016.

38. http://www.st.com/en/clocks-and-timers/m41t62.html, Retrieved 1/29/18.

39. http://www.dtic.mil/doctrine/new_pubs/dictionary.pdf, Retrieved 1/29/18.

40. http://www.dtic.mil/doctrine/new_pubs/dictionary.pdf, Retrieved 1/29/18.

41. Liang, Ying-Chang, et al. "Cognitive radio networking and communications: An overview," IEEE transactions on vehicular technology 60.7 (2011): 3386-3407.

42. http://www.zigbee.org/zigbee-for-developers/network-specifications/

43. Sun, et al. “IEEE 802.11 ah: A long range 802.11 WLAN at sub 1 GHz,” Journal of ICT Standardization 1.1 (2013): 83-108.

44. Mulligan, G. “The 6LoWPAN architecture,” Proceedings of the 4th workshop on Embedded networked sensors. ACM, 2007.

45. https://www.i2c-bus.org/specification/

46. Dementyev, A, et al. “Power consumption analysis of Bluetooth Low Energy, ZigBee and ANT sensor nodes in a cyclic sleep scenario,” Wireless Symposium (IWS), 2013 IEEE International. IEEE, 2013.

47. https://www.amazon.com/Tenergy-160mAh-Battery-Helicopter-Flite/dp/B007R8ZN2I, retrieved 1/29/18.

48. Gollakota, S. et al. “Secure In-Band Wireless Pairing,” USENIX security symposium, 2011.

49. http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-121r1.pdf, Retrieved 1/29/18.

50. Rizzolo, R. F. et al. “IBM System z9 eFUSE applications and methodology,” IBM Journal of Research and Development51.1.2 (2007): 65-75.

51. http://www.imflash.com/

52. http://www.microchip.com/wwwproducts/en/ATmega328p, Retrieved 1/29/18.

53. Beaulieu, R, et al. “Implementation and Performance of the Simon and Speck Lightweight Block Ciphers on ASICs.”

54. Seshadri, A. et al. “SAKE: Software attestation for key establishment in sensor networks,” International Conference on Distributed Computing in Sensor Systems. Springer, Berlin, Heidelberg, 2008.

55. Halderman, J. A. et al. “Lest we remember: cold-boot attacks on encryption keys,” Communications of the ACM 52.5 (2009): 91-98.