Internet of Things (IoT) Trust Concerns

Jeffrey Voas
Rick Kuhn
Computer Security Division
Information Technology Laboratory

Phillip Laplante
Penn State University

Sophia Applebaum
The MITRE Corporation
McLean, Virginia

October 17, 2018

National Institute of Standards and Technology
U.S. Department of Commerce

Abstract

The Internet of Things (IoT) refers to systems that involve computation, sensing, communication, and actuation (as presented in NIST Special Publication (SP) 800-183). IoT involves the connection between humans, non-human physical objects, and cyber objects, enabling monitoring, automation, and decision making. The connection is complex and inherits a core set of trust concerns, most of which have no current resolution This publication identifies 17 technical trust-related concerns for individuals and organizations before and after IoT adoption. The set of concerns discussed here is necessarily incomplete given this rapidly changing industry, however this publication should still leave readers with a broader understanding of the topic. This set was derived from the six trustworthiness elements in NIST SP 800-183. And when possible, this publication outlines recommendations for how to mitigate or reduce the effects of these IoT concerns. It also recommends new areas of IoT research and study. This publication is intended for a general information technology audience including managers, supervisors, technical staff, and those involved in IoT policy decisions, governance, and procurement.

Keywords

Internet of Things (IoT); computer security; trust; confidence; network of 'things'; interoperability; scalability; reliability; testing; environment; standards; measurement; timestamping; algorithms; software testing

Disclaimer

Additional Information

For additional information on NIST's Cybersecurity programs, projects and publications, visit the Computer Security Resource Center, csrc.nist.gov. Information on other efforts at NIST and in the Information Technology Laboratory (ITL) is available at www.nist.gov and www.nist.gov/itl.

Public Comment Period: ***October 17, 2018 through November 16, 2018***

National Institute of Standards and Technology
Attn: Computer Security Division, Information Technology Laboratory
100 Bureau Drive (Mail Stop 8930) Gaithersburg, MD 20899-8930
Email: iot@nist.gov

Executive Summary

The Internet of Things (IoT) is utilized in almost every aspect of personal life and is being adopted within nearly every industry. Governments are taking notice and looking at IoT from a variety of dimensions. One dimension is how IoT systems can improve efficiency, analytics, intelligence, and decision making. Another dimension deals with regulation (i.e., whether is IoT a technology that needs governance, legislation, and standards due to its universal reach and impact). For example, IoT carries security concerns due to its high degree of connectivity. Should there be rules or laws specific to IoT security issues? The same question applies to privacy, safety, and dependability.

As with any new, unproven technology, questions about trustworthiness arise. Those questions often boil down to this: are the benefits worth the risks? Are there more positive reasons to adopt a new technology than to avoid it? If answered with "yes," a secondary question is: how can you minimize the risks to make the technology more acceptable and therefore "suitable for use" by a wider audience? Most new technologies are created to benefit humanity. However, those technologies in the wrong hands can enable new and unforeseen nefarious actions.

This publication is not directly focused on risk assessment and risk mitigation, but rather on trust. That is, will an IoT product or service provide the desired operations with an acceptable level of quality? To answer this question, the analysis begins with a simple understanding of trust. Here, trust is the probability that the intended behavior and the actual behavior are equivalent given a fixed context, fixed environment, and fixed point in time. Trust is viewed as a level of confidence. In this publication, trust is considered at two levels: (1) whether a "thing" or device trusts the data it receives, and (2) whether a human trusts the "things," services, data, or complete IoT offerings that it uses. In this document, we are more focused on the human trust concern than the concern of "things" to trust data. However, both are important.

This publication promotes awareness of 17 technical concerns that can negatively affect one's ability to trust IoT products and services. It is intended for a general information technology audience including managers, supervisors, technical staff, and those involved in IoT policy decisions, governance, and procurement. This publication should be of interest to early adopters and persons responsible for integrating the various devices and services into purposed IoT offerings. The following is a brief synopsis of each technical concern.

Scalability

This trust concern occurs from a combinatorial explosion in the number of "things" that are part of a system. "Things" and the services to interconnect them are often relatively inexpensive and therefore create an opportunity for functionality bloat. This allows complexity to skyrocket, causing difficulty for testing, security, and performance. If the average person is associated with 10 or more IoT "things," the number of "things" requiring connectivity explodes quickly, as do bandwidth and energy demands. Combinatorial explosion and functionality bloat are trust concerns.

Heterogeneity

This trust concern results from competition in the marketplace. The argument goes that with more choices, the competition will result in lower prices. While true, the ability of heterogeneous "things" to interoperate and integrate creates a different tension related to emergent behaviors. Moreover, heterogeneity will almost definitely create *emergent behaviors* that will enable new and unknown security vulnerabilities as well as impact other concerns such as reliability and performance. Potential vulnerability issues related to heterogeneity also occur with *supply chain* applications.

Ownership and Control

This trust concern occurs when much of the functionality within an IoT system originates from third-party vendors. Third-party black-box devices make trust more difficult for integrators and adopters to assess. This is particularly true for security and reliability since the internal workings of black-boxes are not observable and transparent. No internal computations can be specifically singled out and individually tested. Black-box "things" can contain malicious trojan behaviors. When IoT adopters better understand the magnitude of losing access to the internals of these acquired functions, they will recognize limitations to trust in their composite IoT systems.

Composability, Interoperability, Integration, and Compatibility

This trust concern occurs because hardware and software components may not work well when composed, depending on whether: (1) the "right" components were selected; (2) the components had the proper security and reliability built in; and (3) the architecture and specification of the system that the components will be incorporated into was correct. Further, problems arise if components cannot be swapped in or out to satisfy system requirements; components cannot communicate; and components cannot work in conjunction without conflict. Integration, interoperability, compatibility, and composability each impact IoT trust in a slightly different manner for networks of "things," and each "thing" should be evaluated before adoption into a system for each of these four properties.

"Ilities"

This trust concern deals with the *quality* attributes frequently referred to as "ilities." Functional requirements state what a system *shall* do. Negative requirements state what a system *shall not* do, and non-functional requirements (i.e., the "ilities") typically state what *level of quality* the system shall exhibit both for the functional and negative requirements. One difficulty for IoT adopters and integrators is that there are dozens of "ilities," and most are not easily measured. Another difficulty is that technically, a system cannot have high levels of all "ilities" since some are in technical conflict. For example, higher security typically means lower performance. Finally, deciding which "ilities" are more important and at what level and cost is not a well understood process. No cookbook approach exists. So, although quality is desired, getting it is the challenge.

Synchronization

This trust concern stems from IoT systems being distributed computing systems. Distributed computing systems have different computations and events occurring concurrently. There can be numerous computations and events (e.g., data transfers) occurring in parallel, and those computations and events must need some degree of synchronization. For that to occur, a timing mechanism is needed that applies to all computations and events. However, no such global clock exists. Therefore, timing anomalies will occur, enabling vulnerabilities, poor performance, and IoT failures.

Measurement

This trust concern stems from a lack of IoT metrics and measures. Metrics and measures are keystones of trust. Since IoT is a relatively young set of technologies, few metrics and measures are available to adopters and integrators. To date, there are few ways to measure IoT systems other than by *counting* ‘things” or dynamic testing. Because of this, it becomes difficult to argue that a system is trustable or even estimate the amount of testing that a system should receive.

Predictability

This trust concern stems from an inability to predict how different components will interact. The ability to design useful IT systems depends at a fundamental level on predictability, the assurance that components will provide the resources, performance, and functions that are specified when they are needed. This is hard enough to establish in a conventional system, but an extensive body of knowledge in queueing theory and related subjects has been developed. IoT systems will provide an even greater challenge since more components will interact in different ways and possibly not at consistent times.

Testing and Assurance

This trust concern stems from the additional testing challenges created by IoT beyond those encountered with conventional systems. The numerous number of interdependencies alone create testing difficulty because of the large numbers of tests that are needed to simply cover some percentage of the interdependencies. Testing concerns always increase when devices and services are black-box and offer no transparency into their internal workings. Most IoT systems will be built from only black-box devices and services. Also, IoT systems are highly data driven, and assuring the integrity of the data and assuring that a system is resilient to data anomalies will be required. These are just a few of the many testing and assurance problems related to IoT.

Certification

This trust concern occurs because certification is difficult and often causes conflict. Questions immediately arise as to what criteria will be selected and who will perform the certification. Other questions that arise include: (1) What is the impact on time-to-market if the system undergoes certification prior to operation? (2) What is the lifespan of a “thing” relative to the time required to certify that “thing?” and (3) What is the value of building a system from "things," very few of which received certification? Without acceptable answers to such questions, it is unlikely that certification can offer the degree of trust most IoT adopters would want.

Security

Security is a trust concern for all "things" in IoT systems. For example, sensor data may be tampered with, stolen, deleted, dropped, or transmitted insecurely, allowing it to be accessed by unauthorized parties. IoT devices may be counterfeited, and default credentials are still widely used. Further, unlike traditional personal computers, there are few security upgrade processes for "things," such as patches or updates.

Reliability

Reliability is a trust concern for all IoT systems and "things." It will rarely be possible to claim that an IoT system works perfectly for any environment, context, and for any anomalous event that the system can experience. What this means for trust is that reliability assessments depend heavily on correct knowledge of the context and environment and resilience to handle anomalous events and data. Rarely will such knowledge exist and provide complete resilience.

Data Integrity

This trust concern focuses on the quality of the data that is generated by or fed into an IoT system. The quality of the data flowing between devices and from sensors will directly impact whether an IoT system is fit-for-purpose. Data is the "blood" flowing through IoT systems. The ability to trust data involves many factors: (1) accuracy, (2) fidelity, (3) availability, and (4) confidence that the data cannot be corrupted or tampered with. Cloud computing epitomizes the importance of trusting data. Where data resides is important. Where is the cloud? Can the data be leaked from that location? It is a tendency to think of "your data" on "your machine," but in some cases, the data is not just "yours." Leased data can originate from anywhere and from vendors at the time of their choosing and with the integrity of their choosing. These trust concerns should be considered during IoT system development and throughout operation.

Excessive Data

This trust concern is overwhelming amounts of data that get generated and processed in an IoT system. IoT systems are likely to have a dynamic and rapidly changing dataflow and workflow. There may be numerous inputs from a variety of sources such as sensors, external databases or clouds, and other external subsystems. The potential for the generation of vast amounts of data over time renders IoT systems potential "big data" generators. The possibility of not being able to guarantee the integrity of excessive amounts of data or even process that data is a trustworthiness concern.

Performance

This trust concern is too much performance. This may seem counterintuitive. The speed at which computations and data generation can occur in an IoT system is increasing rapidly. Increased computational speed inhibits a system's ability to log and audit transactions as the rate of data generation exceeds the speed of storage. This situation, in turn, makes real-time forensic analysis and recovery from faults and failures more difficult as data is lost and computational deadlines become harder to meet. Consequently, there are fewer ways to "put on the brakes," undo incorrect computations, and fix internal and external data anomalies. Furthermore, computing faster to a wrong outcome offers little trust.

Usability

This trust concern deals with whether users understand how to use the devices that they have access to. How "friendly" are IoT devices to use and learn? This quality is an important consideration for most IT systems, but it may be more of a challenge with IoT, where the user interface may be tightly constrained by limited display size and functionality or where a device can only be controlled via remote means. User interfaces for some device classes, such as Smart Home devices, are often limited to a small set of onboard features (e.g., LED status indicators and a few buttons) and a broader set of display and control parameters accessible remotely via a computer or mobile device. Usability and other trust concerns to which usability is intimately tied have significant implications for user trust.

Visibility and Discovery

The visibility trust concern manifests when technologies become so ingrained in daily life that they disappear from users. If you cannot see a technology, how do you know what else it might be doing? For example, consider voice response technology, such as smart speakers. When you talk to the device, do you know if it is the only system listening? Do you know if the sounds that it hears are stored somewhere for eternity and linked to you?

The discovery trust concern stems from the fact that the traditional Internet was built almost entirely on the TCP/IP protocol suite with HTML for web sites running on top of TCP/IP. Standardized communication port numbers and internationally agreed web domain names enabled consistent operation regardless of the computer or router manufacturer. This structure has not extended to IoT devices because they generally do not have the processing power to support it. This has enabled many new protocol families, causing a vast number of possible interactions among various versions of software and hardware from many different sources. These interactions are prone to security and reliability problems.

In addition to these the 17 concerns, this publication concludes with two non-technical, trust-related appendices. Appendix A reviews the impact that many of the 17 technical concerns have on insurability and risk measurement. Appendix B discusses how a lack of IoT regulatory oversight and governance affects users of IoT technologies by creating a vacuum of trust in the products and services that they can access.

Table of Contents

List of Appendices

1 Introduction

The Internet of Things (IoT) is being utilized in almost every aspect of life today, although this fact is often unknown and not advertised. The incorporation of IoT into everyday processes will continue to increase.

According to Forbes magazine [5] there will be a significant increase in spending on the design and development of IoT applications and analytics. Furthermore, the biggest increases will be in the business-to-business (b2b) IoT systems (e.g. manufacturing, healthcare, agriculture, transportation, utilities, etc.), which will reach $267 billion by 2020. In addition to b2b, smart products are becoming more prevalent, such as smart homes, smart cars, smart TVs, even smart light bulbs, and other basic commodities. In other words, products that can sense, learn, and react to user preferences are gaining acceptance and being deployed in modern living.

The term “Internet of Things" (IoT) is a phrase that was coined by Kevin Ashton in 1999 [2], although he prefers "Internet *for* things” [8]. IoT is an acronym comprised of three letters: I, o, and T. The “o” matters little, and, as already mentioned, “of” might be better replaced by “for.” The Internet (I) existed long before the IoT acronym was coined, and so it is the “things” (T) that makes IoT different from previous IT systems and computing approaches. “Things” are what make IoT unique. Many people question whether IoT is just marketing hype or whether there is a science behind it. That is a fair question to ask about any new, unproven technology.

The acronym IoT currently has no universally-accepted and actionable definition. However, attempts have been made. A few examples include:

- “*The term Internet of Things generally refers to scenarios where network connectivity and computing capability extends to objects, sensors and everyday items not normally considered computers, allowing these devices to generate, exchange and consume data with minimal human intervention.”* [33]

- “*Although there is no single definition for the Internet of Things, competing visions agree that it relates to the integration of the physical world with the virtual world—with any object having the potential to be connected to the Internet via short-range wireless technologies, such as radio frequency identification (RFID), near field communication (NFC), or wireless sensor networks (WSNs). This merging of the physical and virtual worlds is intended to increase instrumentation, tracking, and measurement of both natural and social processes.”* [59]

- “*The concept of Internet of Things (IOT)…is that every object in the Internet infrastructure is interconnected into a global dynamic expanding network.*” [11]

Instead of offering an official definition of IoT in 2016, NIST published a document titled “Networks of ‘Things’” to partially address the deficit of having an accepted IoT definition [44]. In that document, five primitives were presented that can be visualized as Lego™-like building blocks for any network of “things.” The primitives are the (T)s.

The primitives are: (1) sensors—a physical utility that measures physical properties; (2) aggregators—software that transforms big data into smaller data; (3) communication channels—data transmission utilities that allow “things” to communicate with “things;” (4) *e*-Utilities—software or hardware components that perform computation; and a (5) decision trigger—an algorithm and implementation that satisfies the purpose of a network of “things” by creating the final output. Note that any purposed network of “things” may not include all five. For example, a network of “things” can exist without sensors. Also note that having a model of the components of a network of “things” is still not a definition of IoT.

Before leaving the problem of having no universally accepted and actionable definition for IoT, it should be stated that IoT is increasingly associated with Artificial Intelligence (AI), automation, and “smart” objects. So, is “IoT” any *noun* onto which you can attach the adjective “smart” (e.g., smart phone, smart car, smart appliance, smart toy, smart home, smart watch, smart grid, smart city, smart tv, smart suitcase, smart clothes, etc.)? No answer is offered here, but it is something to consider because the overuse of the adjective “smart” adds confusion as to what IoT is about.

Now consider the question: what is meant by “trust?” No formal definition is suggested in this publication, but rather a variation on the classical definition of reliability. Here, trust is the probability that the *intended* behavior and the *actual* behavior are equivalent given a fixed context, fixed environment, and fixed point in time. Trust should be viewed as a *level of confidence*. For example, cars have a trusted set of behaviors when operating on a roadway. The same set of behaviors cannot be expected when the car is sunken in a lake. This informal trust definition works well when discussing both “things” and networks of “things.”

The value of knowing intended behaviors cannot be dismissed when attempting to establish trust. Lack of access to a specification for intended behaviors is a trust concern. Even if there is little difficulty gluing “things” to other “things,” that still only addresses a network of “things” architecture, and that is one piece of determining trust. Correct architecture does not ensure that the actual behavior of the composed “things” will exhibit the intended composite behavior. Hardware and software components may not work well when integrated, depending on whether they were the right components to be selected, whether they had the proper levels of “ilities” such as security and reliability built in, and whether the architecture and specification for the composition was correct.

The Internet (I) is rarely associated with the terms “trust” or “trustable.” Identity theft, false information, the dark web, breakdown in personal privacy, and other negative features of (I) have caused some people to avoid the Internet altogether. However, for most, avoidance is not an option. Similar trust concerns occur for (T) because “things” carry their own trust concerns, and the interactions between “things” can exacerbate these concerns. From a trust standpoint, the Internet should be viewed as an untrustworthy backbone with untrustworthy things attached—that becomes a perfect storm. Hence, there are three categories of IoT trust that must be addressed: (1) trust in a “thing,” (2) trust in a network of “things,” and (3) trust that the environment and context that the network will operate in is known and that the network will be *fit for purpose* in that environment, context, and at a specific point in time.

Understanding what IoT is and what trust means is the first step in confidently relying on IoT. IoT is a complex, distributed system with temporal constraints. This publication highlights 17 technical concerns that should be considered before and after deploying IoT systems. This set has been derived from the six trustworthiness elements presented in NIST SP 800-183 (the six are reprinted in Appendix C.)

The 17 technical concerns are: (1) scalability, (2) heterogeneity, (3) control and ownership, (4) composability, interoperability, integration, and compatibility, (5) "ilities", (6) synchronization, (7) measurement, (8) predictability, (9) IoT-specific testing and assurance approaches, (10) IoT certification criteria, (11) security, (12) reliability, (13) data integrity, (14) excessive data, (15) speed and performance, (16) usability, and (17) visibility and discovery. The publication also offers recommendations for ways to reduce the impacts of some of the 17 concerns.

This publication also addresses two non-technical trust concerns in Appendix A and Appendix B. Appendix A discusses insurability and risk measurement, and Appendix B discusses a lack of regulatory oversight and governance.

In summary, this document advances the original six IoT trust elements presented in [44]. This document also serves as a roadmap for where new research and thought leadership is needed. This publication is intended for a general audience including managers, supervisors, technical staff, and those involved in IoT policy decisions, governance, and procurement.

2 Overwhelming Scalability

Computing is now embedded in products as mundane as lightbulbs and kitchen faucets. When computing becomes part of the tiniest of consumer products, scalability quickly becomes an issue, particularly if these products require network connectivity. Referring back to the primitives introduced earlier, scalability issues are seen particularly with the sensors and aggregators components of IoT. Collecting and aggregating data from tens to hundreds of devices sensing their environment can quickly become a performance issue.

Consider this analysis. If the average person is associated with 10 or more IoT "things," the number of "things" requiring connectivity explodes quickly, as do bandwidth and energy demands. Therefore, computing, architecture, and verification changes are inevitable, particularly if predictions of 20-50 billion new IoT devices being created within the next three years come true. More "things" will require a means of communication between the "things" and the consumers they serve, and the need for inter-communication between "things" adds an additional scalability concern beyond simply counting the number of "things" [54].

Increased scalability leads to increased complexity. Note that although increased scalability leads to complexity, the converse is not necessarily true. Increased complexity can arise from other factors such as infinite numbers of dataflows and workflows.

Unfortunately, complexity does not lend itself to trust that is easy to verify. Consider an analogous difficulty that occurs during software testing when the number of Source Lines of Code (SLOC) increases. Generally, when SLOC increases, more test cases are needed to achieve greater testing coverage.[1] Simple statement testing coverage is the process of making sure that there exists a test case that touches (executes) each line of code during a test. As SLOC increases, so may the number of paths though the code, and when conditional statements are considered, the number of test cases to exercise all of them thoroughly (depending on the definition of thoroughness) becomes combinatorically explosive.[2] IoT systems will likely suffer from a similar scalability concern that will impact their ability to have trust verified via testing.

Thus IoT systems will likely suffer from a similar combinatorial explosion to that just mentioned for source code paths. The number of potential dataflow and workflow paths for a network of "things" with feedback loops becomes intractable quickly, leading to a combinatorial explosion that impacts the ability to test with any degree of thoroughness. This is due to the expense in time and money. Further, just as occurs in software code testing, finding test scenarios to exercise many of the paths will not be feasible.[3] IoT testing concerns are discussed further in Section 10.

[1] This difficulty does not occur for straight-line code that contains no branches or jumps, which is rare.

[2] There are software coverage testing techniques to address testing paths and exercising complex conditional expressions. However, for these more complex forms of software testing coverage, the ability to generate appropriate test cases can become unfeasible due to a lack of reachability (i.e., is there any test case in the universe that can execute this scenario?).

[3] This is the classic test case generation dilemma (i.e., what can you do when you cannot find the type of test case you need?).

In summary, avoiding the inevitable concern of large scale for many IoT systems will not be practical. However, a network of “things” can have bounds placed on it (e.g., limiting access to the Internet). By doing so, the threat space for a specific network of “things” is reduced, and testing becomes more tractable and thorough. By considering sub-networks of “things,” divide-and-conquer trust approaches can be devised that at least offer trust to higher level components than simple “things.”

3 Heterogeneity

The heterogeneity of "things" is economically desirable because it fosters marketplace competition. Today, IoT creates technical problems that mirror past problems when various flavors of Unix and Postscript did not interoperate, integrate, or compose well. Then, different versions of Postscript might or might not print to a specific printer, and moving Unix applications to different Unix platforms did not necessarily mean the applications would execute. It was common to ask which "flavor of Unix" a vendor's product operated on.

As with scalability, issues concerning heterogeneity are inevitable as IoT networks are developed. A network of "things" is simply a system of "things" that are made by various manufacturers, and these "things" will have certain tolerances or intolerances to the other "things" to which they connect and communicate.

The marketplace of "things" and services (e.g., wireless communication protocols and clouds) will allow for the architecture of IoT offerings with functionality from multiple vendors. Ideally, the architecture for a network of "things" will allow IoT products and services to be swapped in and out quickly, but often, that will not be the case.

Heterogeneity will create problems in getting "things" to integrate and interoperate with other "things," particularly when they are from different and often competing vendors, and these issues must be considered for all five classes of IoT primitives [44]. This is discussed more in Section 5. Heterogeneity will almost definitely create *emergent behaviors* that will enable new and unknown security vulnerabilities, as well as impact other concerns such as reliability and performance.

Finally, this is an appropriate place to mention potential vulnerability issues related to *supply chain*. For example, how do you know that a particular "thing" is not counterfeit? Do you know where the "thing" originated from? Do you trust any documentation related to the specification of a "thing" or warranties of how the "thing" was tested by the manufacturer? While supply chain is a concern that is too large to dwell on here with any depth, a simple principle does appear: as heterogeneity increases, it is likely that supply chain concerns will also increase.

4 Loss of Ownership and Control

Third party black-box devices make trust more difficult for integrators and adopters to assess. This is particularly true for security and reliability in networks of “things.” When a “thing” is a black-box, the internals of the “thing” are not visible. No internal computations can be specifically singled out and individually tested. Black-box “things” can contain malicious trojan behaviors. Black boxes have no transparency.

Long-standing black-box software reliability testing approaches are a prior example of how to view this dilemma. In black-box software reliability testing, the software under test is viewed strictly by (input, output) pairs. There, the best that can be done is to build tables of (input, output) pairs, and if the tables become large enough, they can offer hints about the functionality of the box and its internals. This process becomes an informal means by which to attempt to reverse engineer functionality. In contrast, when source code is available, white-box testing approaches can be applied. White-box software testing offers internal visibility to the lower-level computations (e.g., at the line-of-code level).

This testing approach is particularly important for networks of “things.” It is likely that most of the physical “things” that will be employed in a network of “things” will be third-party, commercial, and are therefore commercial off-the-shelf (COTS). Therefore, visibility into the inner workings of a network of “things” may only be possible at the communication interface layer [45].

Consider the following scenario. A hacked refrigerator's software interacts with an app on a person’s smartphone, installing a security exploit that can be propagated to other applications with which the phone interacts. The user enters their automobile, and their phone interacts with the vehicle’s operator interface software which downloads the new software, including the defect. Unfortunately, the software defect causes an interaction problem (e.g., a deadlock) that leads to a failure in the software-controlled safety system during a crash, leading to injury. A scenario such as this is sometimes referred to as a *chain of custody*.

The above scenario demonstrates how losing control of the cascading events during operation can result in failure. This sequence also illustrates the challenge of identifying and mitigating interdependency risks and assigning blame when something goes wrong using techniques such as propagation analysis and traceability analysis. Liability claims are hard to win since the “I agree to all terms” button is usually non-avoidable [54]. (See Section 13.)

Public clouds are important for implementing the economic benefits of IoT. Public clouds are black-box services. Public clouds are a commercial commodity where vendors rely on service-level agreements for legal protection from security problems and other forms of inferior service from their offerings. Integrators and adopters have few protections here. Further, what properties associated with trust can integrators and adopters test for in public clouds?

There are examples of where an organization might be able to test for some aspects of trust in a public cloud: (1) performance (i.e., latency time to retrieve data and the computational time to execute a software app or algorithm) and (2) data leakage. Performance is a more straightforward measure to assess using traditional performance testing approaches. Data leakage is harder but not impossible. By storing data that, if leaked, is easy to detect (i.e., credit card information), a bank can quickly notify a card owner when an illegitimate transaction was attempted. Note, however, that such tests that do not result in the observation of leakage do not prove that a cloud is not leaking since such testing does not guarantee complete observability and is not exhaustive. This is no different than the traditional software testing problem where 10 successive passing tests (meaning that no failures were observed) does not guarantee that the 11^{th} test will also be successful.

In summary, concerns related to loss of ownership and control are often human, legal, and contractual. Technical recommendations cannot fully address these. It should be mentioned, though, that these concerns can be enumerated (e.g., as misuse or abuse cases) and evaluated during risk assessments and risk mitigation in the design and specification phases of a network of "things." This risk assessment and risk mitigation may and possibly should continue throughout operation and deployment.

5 Composability, Interoperability, Integration, and Compatibility

Hardware and software components may not work well when composed, depending on whether: (1) the "right" components were selected; (2) the components had the proper security and reliability built-in (as well as other quality attributes); and (3) the architecture and specification of the system that the components will be incorporated into was correct.

Note there is a subtle difference between composability, interoperability, integration, and compatibility. *Composability* addresses the issue of sub-systems, components, and the degree to which a sub-system or component can be swapped in or out to satisfy a system's requirements. *Interoperability* occurs at the interface level, meaning that when interfaces are understood, two distinct sub-systems can communicate via a common communication format without needing knowledge concerning the functionality of the sub-systems. *Integration* is a process of often bringing together disparate sub-systems into a new system. *Compatibility* simply means that two sub-systems can exist or work in conjunction without conflict.

Integration, interoperability, compatibility, and composability each impact IoT trust in a slightly different manner for networks of "things," and each "thing" should be evaluated before adoption into a system for each of these four properties.

Consider previous decades of building S*ystems of Systems* (SoS). Engineering systems from smaller components is nothing new. This engineering principle is basic and taught in all engineering disciplines. Building networks of "things" should be no different. However, this is where IoT's concerns of heterogeneity, scalability, and a lack of ownership and control converge to differentiate traditional SoS engineering from IoT composition.

Consider military-critical and safety-critical systems. Such systems require components that have prescriptive requirements. The systems themselves will also have prescriptive architectures that require that each component's specification is considered before adoption. Having access to information concerning the functionality, results from prior testing, and expected usage of components is always required before building critical systems.

IoT systems will likely not have these prescriptive capabilities. IoT's "things" may or may not even have specifications, and the system being built may not have a complete or formal specification. It may be more of an informal definition of what the system is to do, but without an architecture for how the system should be built. Depending on: (1) the grade of a system (e.g., consumer, industrial, military, etc.), (2) the criticality (e.g., safety-critical, business-critical, life-critical, security-critical, etc.), and (3) the domain (e.g., healthcare financial, agricultural, transportation, entertainment, energy, etc.), the level of effort required to specify and build an IoT system can be approximated. However, no cookbook-like guidance yet exists.

In summary, specific recommendations for addressing the inevitable issues of composability, interoperability, integration, and compatibility are: (1) understand the actual behaviors of the "things;" (2) understand the environment, context, and timing that each "thing" will operate in; (3) understand the communication channels between the "things" [43]; (4) apply systems design and architecture principles when applicable; (5) and apply the appropriate risk assessment and risk mitigation approaches during architecture and design based on the grade, criticality, and domain.

6 Abundance of "Ilities"

A trust concern for networks of "things" deals with the quality attributes termed "ilities" [52]. Functional requirements state what a system shall do; negative requirements state what a system shall not do; and non-functional requirements (i.e., the "ilities") typically state what level of quality the system shall exhibit both for the functional and negative requirements. "Ilities" apply to both "things" and the systems they are built into.

It is unclear how many "ilities" there are—it depends on who you ask. This document mentions each of these "ilities" in various contexts and level of detail: availability, composability, compatibility, dependability, discoverability, durability, fault tolerance, flexibility, interoperability, insurability, liability, maintainability, observability, privacy, performance, portability, predictability, probability of failure, readability, reliability, resilience, reachability, safety, scalability, security, sustainability, testability, traceability, usability, visibility, and vulnerability. Most of these will apply to "things" and networks of "things." However, not all readers will consider all of these to be legitimate "ilities."

One difficulty here is that for some "ilities" there is a subsumes hierarchy. For example, reliability, security, privacy, performance, and resilience are "ilities" that are grouped into what LaPrie et. al termed as *dependability*. While having a subsumes hierarchy might appear to simply be the relationship between different "ilities," that is not necessarily the case. This can create confusion.

Building levels of the "ilities" into a network of "things" is costly, and not all "ilities" cooperate with each other (i.e., "building in" more security can reduce performance [53]. Another example would be fault tolerance and testability. Fault-tolerant systems are designed to mask errors during operation. Testable systems are those that do not mask errors and therefore make it easier for a test case to notify when something is in error inside of a system. Deciding which "ilities" are more important is difficult from both a cost-benefit trade-off analysis and a technical trade-off analysis. Also, some "ilities" can be quantified and others cannot. For those that cannot be quantified, qualified measures exist.

Further, consider an "ility" such as reliability. Reliability can be assessed for: (1) a "thing," (2) the interfaces between "things," and (3) the network of "things" itself [46]. These three types of assessments apply to most "ilities."

Deciding which "ilities" are more important—and at what level and cost—is not a well understood process. No cookbook approach exists. The point here is that these non-functional requirements often play just as important a role in terms of the overall system quality as do functional requirements. This reality will impact the satisfaction of the integrators and adopters with the resulting network.

In summary, deciding which "ility" is more important than others must be dealt with on a case-by-case basis. It is recommended that the "ilities" are considered at the beginning of the life cycle of a network of "things." Failure to do so will cause downstream problems throughout the system's life-cycle, and it may continually cause contention as to why intended behaviors do not match actual behaviors.

7 Synchronization

A network of "things" is a distributed computing system. Distributed computing systems have different computations and events occurring concurrently. There can be numerous computations and events (e.g., data transfers) occurring in parallel.

This creates an interesting dilemma similar to that in air traffic control: trying to keep all events properly synchronized and executing at the precise times and in a precise order. When events and computations get out of order due to delays or failures, an entire ecosystem can become unbalanced and unstable.

IoT is no different and is possibly more complex than air traffic control. In air traffic control, there is a basic global clock that does not require that events be timestamped to high levels of fidelity (e.g., a microsecond). Further, events are regionalized around particular airspace sectors and airports.

There is nothing similar in IoT. Events and computations can occur anywhere, be transferred at any time, and occur at differing levels of speed and performance. The desired result is that all these events and computations converge toward a single decision (output). The key concern is "any time" because these transactions can take place geographically anywhere, at the microsecond level, with no clear understanding of what the clock in one geographic region means with respect to the clock in another geographic region.

There is no trusted universal timestamping mechanism for practical use in many or most IoT applications. The Global Positioning System (GPS) can provide very precise time, accurate up to 100 nanoseconds with most devices. Unfortunately, GPS devices have two formidable limitations for use in IoT. First, GPS requires unobstructed line-of-sight access to satellite signals. Many IoT devices are designed to work where a GPS receiver could not receive a signal, such as indoors or otherwise enclosed in walls or other obstructions. Additionally, even if an IoT device is placed where satellite signal reception is available, GPS power demands are significant. Many IoT devices have drastically limited battery life or power access, requiring carefully planned communication schedules to minimize power usage. Adding the comparatively high-power demands of GPS devices to such a system could cripple it. In general, GPS may not be practical for use in many networks of things.

Consider a scenario where a sensor in geographic location *v* is supposed to release data at time *x*. There is an aggregator in location *z* waiting to receive this sensor's data concurrently with outputs from other sensors. Note that *v* and *z* are geographically far apart, and the local time *x* in location *v* does not agree, at a global level, with what time it is at *z*. If there existed a universal timestamping mechanism, local clocks could be avoided altogether, and this problem would go away. With universal timestamping, the time of every event and computation in a network of "things" could be agreed upon by using a central timestamping authority that would produce timestamps for all events and computations that request them. Because timing is a vital component needed to trust distributed computations, such an authority would be beneficial. However, such an authority does not exist [40]. Research is warranted here.

8 Lack of Measurement

Standards are intended to offer levels of trust, comparisons of commonality, and predictions of certainty. Standards are needed for nearly everything, but without metrics and measures, standards become more difficult to write and against which to determine compliance. Metrics and measures are classified in many ways.

Measurement generally allows for the determination of one of two things: (1) what currently exists and (2) what is predicted and expected in the future. The first is generally easier to measure. One example is *counting*. For example, one can count the number of coffee beans in a bag. Another approach is *estimation*. Estimation approximates what you have. By using the coffee example and having millions of beans to count, it might be easier to weigh the beans and use that weight to estimate an approximate count.

Prediction is different from estimation, although estimation can be used for prediction. For example, an estimate of the current reliability of a system given a fixed environment, context, and point in time might be 99%. Note the key phrase is "point in time." In comparison, a prediction might say that based on an estimate of 99% reliability today, it is believed that the reliability will also be 99% tomorrow. However, after tomorrow, the reliability might change. Why? The reason is simple: as *time* moves forward, components usually wear out, thus reducing overall system reliability, or as time moves forward, the environment may change such that the system is under less stress, thus increasing predicted reliability. In IoT, as "things" may be swapped in and out on a quick and continual basis, predictions and estimations of an "ility" such as reliability will be difficult.

To date, there are few ways to measure IoT systems other than by *counting* "things" or dynamic testing. Counting is a static approach. Testing is a dynamic approach when the network is executed. Note that there are static testing approaches that do not require network execution (e.g., a walkthrough of the network architecture). Thus, the number of "things" in a system can be counted just like how lines of code in software can be counted, and black-box testing can be used to measure certain "ilities."

In summary, several limited recommendations have been mentioned for mitigating the current lack of measurement and metrics for IoT. To date, counting measures and dynamic approaches such as estimating reliability and performance are reasonable candidates. Static testing (e.g., code checking) can also be used to show that certain classes of IoT vulnerabilities are likely not present. IoT metrology is an open research question.

9 Predictability

The ability to design useful IT systems depends, at a fundamental level, on predictability—the assurance that components will provide the resources, performance, and functions that are specified when they are needed. This is hard enough to establish in a conventional system, but an extensive body of knowledge in queueing theory and related subjects has been developed. IoT systems will provide an even greater challenge since more components will interact in different ways and possibly not at consistent times.

Two properties of IoT networks have a major impact on predictability: (1) a much larger set of communication protocols may be involved in a single network, and (2) the network configuration changes rapidly. Communication protocols for networks of “things” include at least 13 data links, three network layer routings, five network layer encapsulations, six session layers, and two management standards [35]. Data aggregators in the network must thus be able to communicate with devices that have widely varying latency, throughput, and storage characteristics. Since many small devices have limited battery life, data transmission times must be rationed so devices are not always online. For example, Bluetooth Low Energy (BLE) devices can be configured to broadcast their presence for periods ranging from 0.2 seconds to 10.2 seconds.

In addition to second-by-second changes in the set of devices currently active, another issue with network configuration changes stems from the embedding of computing devices within the physical world. Even more than conventional systems, humans are part of IoT systems and necessarily affect the predictable availability of services, often in unexpected ways. Consider the story of a driver who took advantage of a cell phone app that interacts with his vehicle's onboard network to allow him to start the car with the phone. Though probably not considered by the user, the starting instructions are routed through the cellular network. The car owner started his car with the cell phone app and later parked the car in a mountainous area, only to discover that it was impossible to re-start the car because there was no cell signal [29].

This rather amusing story illustrates a basic predictability problem for IoT networks: node location and signal strength may be constantly changing. How do you know if a constantly changing network will continue to function adequately and remain safe? Properties such as performance and capacity are unavoidably affected as the configuration evolves, but you need to be able to predict these to know if and how a system can be used for specific purposes. Modeling and simulation become essential for understanding system behavior in a changing environment, but trusting a model requires some assurance that it incorporates all features of interest and accurately represents the environment. Beyond this, it must be possible to adequately analyze system interactions with the physical world, including potentially rare combinations of events.

Recommendations for design principles will evolve for this new environment, but it will take time before users are able to trust systems composed often casually from assorted components. Here again, the importance of a central theme of this document is shown: to be able to trust a system, it must be bounded, but IoT by its nature may defy any ability to bound the problem.

10 Several IoT-specific Testing and Assurance Approaches

To have any trust in networks of "things" acting together, assurance will need to be much better than it is today. A network of "things" presents a number of testing challenges beyond those encountered with conventional systems. Some of the more significant include:

- *Communication among large numbers of devices*. Conventional Internet-based systems typically include one or more servers responding to short communications from users. There may be thousands of users, but the communication is typically one-to-one, with possibly a few servers cooperating to produce a response to users. Networks of "things" may have several tens to hundreds of devices communicating.
- *Significant latency and asynchrony*. Low power devices may conserve power by communicating only on a periodic basis, and it may not be possible to synchronize communications.
- *More sources of failure*. Inexpensive, low power devices may be more likely to fail, and interoperability problems may also occur among devices with slightly different protocol implementations. Since the devices may have limited storage and processing power, software errors in memory management or timing may be more common.
- *Dependencies among devices matter*. With multiple nodes involved in decisions or actions, some nodes will typically require data from multiple sensors or aggregators, and there may be dependencies in the order this data is sent and received. The odds of failure increase rapidly as the chain of cooperating devices grows longer.

The concerns listed above produce a complex problem for testing and assurance, exacerbated by the fact that many IoT applications may be safety critical. In these cases, the testing problem is harder, but the stakes may be higher than for most testing. For essential or life-critical applications, conventional testing and assurance will not be acceptable.

For a hypothetical example, consider a future remote health monitoring and diagnosis app with four sensors connected to two aggregators, which are connected to an e-Utility that is then connected to a local communication channel, which in turn connects to the external Internet and, finally, with a large artificial intelligence application at a central decision trigger node. While 99.9% reliability might seem acceptable for a $3.00 device, it will not be if included in a critical system. If correct operation depends on all 10 of these nodes, and each node is 99.9% reliable, then there is nearly a 1% chance that this network of things will fail its mission—an unacceptable risk for life-critical systems. Worse, this analysis has not even considered the reverse path from the central node with instructions back to the originating app.

Basic recommendations to reduce this level of risk include redundancy among nodes and much better testing. This means not just more conventional tests and review activities, but different kinds of testing and verification. For some IoT applications, it will be necessary to meet test criteria closer to what are used in applications such as telecommunications and avionics, which are designed to meet requirements for failure probabilities of 10^{-5} and 10^{-9}, respectively. Redundancy is part of the answer, with a tradeoff that interactions among redundant nodes become more critical, and the redundant node interactions are added to the already large number of interacting IoT nodes.

One additional testing and assurance issue concerns the *testability* of IoT systems [56]. There are various meanings of this "ility," but two that apply here are: (1) the ability of testing to detect defects and (2) the ability of testing to cover[4] (execute) portions of the system using a fixed set of test cases. The reason (1) is a concern is that IoT systems may have small output ranges (e.g., a system may only produce a binary output). Such systems, if very complex, may inherit an ability to hide defects during testing. The reason (2) is a concern is that if high levels of test coverage cannot be achieved, more portions of the overall system will go untested, leaving no clue as to what might happen when those portions are executed during operation.

The key problem for IoT testing is apparent from the test issues discussed above—huge numbers of interactions among devices and connections coupled with order dependencies. Fortunately, methods based on combinatorics and design of experiments work extremely well in testing complex interactions [31][9][60]. Covering array generation algorithms compresses huge numbers of input value combinations into arrays that are practical for most testing than would be possible with traditional use case-based testing, making the problem more tractable and coverage more thorough. Methods of dealing with this level of testing complexity are the subject of active research [56].

[4] Coverage, too, comes in different types. For instance, the ability to execute each 'thing' once is different from executing each path through a system once.

11 Lack of IoT Certification Criteria

Certification of a product (not processes or people) is a challenge for any hardware, software, service, or hybrid system [22][47][48][49][50][51][56]. IoT systems are hybrids that may include services (e.g., clouds) along with hardware and software.

If rigorous IoT certification approaches are eventually developed, they should reduce many of the trust concerns in this publication. However, building certification approaches is generally difficult [49]. One reason is that certification approaches have less efficacy unless correct threat spaces and operational environments are known. Often, these are not known for traditional systems, let alone for IoT systems.

Certification economics should also be considered (e.g., the cost to certify a "thing" relative to the value of that "thing"). The *criteria* used during certification must be rigorous enough to be of value. A question of who performs the certification and what their qualifications are to perform this work cannot be overlooked. Two other considerations are: (1) the impact on the time-to-market of a "thing" or network of "things" and (2) the lifespan of a "thing" or network of "things." These temporal questions are important because networks of "things," along with their components, may have short lives that are far exceeded by the time needed to certify.

Certifying "things" as standalone entities does not solve the problem of system trust, particularly for systems that operate in a world where their environment and threat space is in continual flux.

If "things" have their functional and non-functional requirements defined, they can be vetted to assess their ability to: (1) be integrated, (2) communicate with other "things," (3) not create conflict (e.g., no malicious output behaviors), and (4) be swapped in and out of a network of "things" (e.g., when a newer or replacement "thing" becomes available).

When composing "things" into systems, special consideration must be given if all of the "things" are not certified. For example, not all "things" in a system may have equal significance to the functionality of the system. It would make sense to spend vetting resources on those that have the greatest impact. Therefore, weighing the importance of each "thing" should be considered before deciding what to certify and what to ignore. Even if all "things" are certified, that still does not mean they will interoperate correctly in a system because the environment, context, and threat space all play a key role in that determination.

Perhaps most importantly, what functional, non-functional, or negative behavior is being certified? Are forms of vetting available to do that? For example, how can a network of "things" demonstrate that certain security vulnerabilities are not present?

In summary, limited recommendations can be considered for how to certify "things" and systems of "things." Software testing is a first line of defense for performing lower levels of certification. However, it is costly and can overestimate quality (e.g., you test a system twice, potentially leading to a false assumption that the system is reliable and does not need a third test). A good first step here is to first define the type of quality with which you are concerned. (See Section 6.) From there, you can assess what can be certified in a timely manner and at what cost.

12 Security

Like traditional IT or enterprise security, IoT security is not a one-size-fits-all problem, and the solutions deployed to solve this problem tend to only be quick fixes that push the issue down the line. Instead, it should be recognized that the issue of IoT security is both multi-faceted and dependent on the effort to standardize IoT security. This section walks through several of these important facets, highlighting solutions that do exist and problems that remain to be solved.

12.1 Security of "Things"

Security is a concern for all "things." For example, sensors and their data may be tampered with, stolen, deleted, dropped, or transmitted insecurely, allowing them to be accessed by unauthorized parties. Further, sensors may return no data, totally flawed data, or partially flawed data due to malicious intent. Sensors may fail completely or intermittently and may lose sensitivity or calibration due to malicious tampering. Note, however, that building security into specific sensors may not be cost effective, depending on the value of a sensor or the importance of the data it collects. Aggregators may contain malware affecting the correctness of their aggregated data. Further, aggregators could be attacked (e.g., denying them the ability to execute or feeding them false data). Communication channels are prone to malicious disturbances and interruptions.

The existence of counterfeit "things" in the marketplace cannot be dismissed. Unique identifiers for every "thing" would be ideal for mitigating this problem, but that is not practical. Unique identifiers can partially mitigate this problem by attaching Radio Frequency identifier (RFID) tags to physical primitives. RFID readers that work on the same protocol as the inlay may be distributed at key points throughout a network of "things." Readers activate a tag, causing it to broadcast radio waves within bandwidths reserved for RFID usage by individual governments internationally. These radio waves transmit identifiers or codes that reference unique information associated with the item to which the RFID inlay is attached. In this case, the item would be a physical IoT primitive.

The time at which computations and other events occur may also be tampered with, not by changing time (which is not possible), but by changing the recorded time at which an event in the workflow is generated or computation performed (e.g., sticking in a **delay()** function call), thus making it unclear when events actually occurred. Malicious latency to induce delays are possible and will affect when decision triggers are able to execute.

Thus, networks of "things," timing, and "things" themselves are all vulnerable to malicious intent.

12.2 Passwords

Default credentials have been a problem plaguing the security community for some time. Although many guides recommend that users and administrators change passwords during system setup, IoT devices are not designed with this standard practice in mind. In fact, most IoT devices often lack intuitive user interfaces with which credentials can be changed. While some IoT device passwords are documented either in user manuals or on manufacturer websites, some device passwords are never documented and are unchangeable. Such scenarios can be leveraged by botnets. The Mirai botnet and its variants successfully brute-forced IoT device default passwords to ultimately launch distributed denial-of-service attacks against various targets [19].

Many practitioners have proposed solutions to the problem of default credentials in IoT systems, ranging from the usual recommendation to change credentials—perhaps with more user awareness—to more advanced ideas like encouraging manufacturers to randomize passwords per device. While not explicitly mitigating the problem of default credentials, the Manufacturer Usage Description (MUD) specification [21] allows manufacturers to specify authorized network traffic, which can reduce the damage caused by default credentials. This specification employs a defense-in-depth strategy intended to address a variety of problems associated with the widespread use of sensor enabled end devices such as IP cameras and smart thermostats. MUD reduces the threat surface of an IoT device by explicitly restricting communications to and from the IoT device to sources and destinations intended by the manufacturer. This approach prevents vulnerable or insecure devices from being exploited and helps alleviate some of the fallout of manufacturers leaving in default credentials.

12.3 Secure Upgrade Process

On a traditional personal computer, weaknesses are typically mitigated by patches and upgrades to various software components, including the operating system. On established systems, these updates are usually delivered via a secure process where the computer can authenticate the source pushing the patch. While parallels exist for IoT devices, very few manufacturers have secure upgrade processes with which to deliver patches and updates. Often, attackers can man-in-the-middle the traffic to push their own malicious updates to the devices, thereby compromising them. Similarly, IoT devices can receive feature and configuration updates, which can likewise be hijacked by attackers for malicious effect.

Transport standards such as HTTPS, as well as existing public-key infrastructure, provide protections against many of the attacks that could be launched against upgrading IoT devices. These standards, however, are agnostic on the implementations of the IoT architecture and do not cover all edge cases. However, the IoT Firmware Update Architecture [24]—recently proposed to the IETF—provides the necessary details needed to implement a secure firmware update architecture, including hard rules defining how device manufacturers should operate. Following this emerging standard could easily mitigate many potential attack vectors targeting IoT devices.

12.4 Summary

Addressing the security of IoT devices is a prescient issue as IoT continues to expand into daily life. While security issues are widespread in IoT ecosystems, existing solutions such as MUD to remediate password weaknesses and transport standards for secure upgrades can be leveraged to boost the overall security of devices. Deploying these existing solutions can yield significant impacts on overall security without requiring significant amounts of time spent researching new technologies.

13 Reliability

IoT reliability should be based on the traditional definition in [25]. The traditional definition is simply the probability of failure-free operation of individual components, groups of components, or the whole system over a bounded time interval and in a fixed environment. Note that this is the basis for the informal definition of trust mentioned earlier. This definition assumes a static IoT system, meaning new “things” are not continually being swapped in and out. Realistically, that will not be the case since new “things” will be added dynamically and on-the-fly, either deliberately or inadvertently. Thus, the instantaneously changing nature of IoT systems will induce emergent and complex chains of custody and make it difficult to ensure and correctly measure reliability [23][55]. The dynamic quality of IoT systems requires that reliability be reassessed when components and the operating environment change.

Reliability is a function of context and environment. Therefore, to perform reliability assessments, *a priori* knowledge of the appropriate environment and context is needed. It will rarely be possible to make a claim such as: *this network of “things” works perfectly for any environment, context, and for any anomalous event that the system can experience*. Unfortunately, wrong assumptions about environment and context will result in wrong assumptions about the degree to which trust has been achieved.

To help distinguish between context and environment, consider a car that fails after a driver breaks an engine by speeding above the manufacturer’s maximum expectation while driving in excellent road conditions and good weather. Weather and road conditions are the environment. Speeding past the manufacturer’s maximum expectation is the context. Violating the expected context or expected environment can both impact failure. Here, failure occurred due to context.

The relationship between anomalous events and “things” is important for a variety of reasons, not the least of which is the loss of ownership and control already mentioned. Assume worst-case scenarios from “things” that are complete black boxes.

Consider certain scenarios: (1) a “thing” fails completely or in a manner that creates bad data which infects the rest of the system, and (2) a “thing” is fed corrupt data, and you wish to know how that “thing” reacts (i.e., is it resilient?). Here, resilience means that the “thing” still provides acceptable behavior. These two scenarios have been referred to as *propagation across* and *propagation from* [46]. *Propagation across* is the study of “garbage in garbage out.” Propagation across tests the strength of a component or “thing.” *Propagation from* is the study of how far through a system an internal failure that creates corrupt data can cascade. Possibly, it propagates all the way, and the system fails, or possibly, the corrupted internal state of the system is not severe enough to cause that. In this case, the system shows its resilience.

A related concern involves who is to blame when a “thing” or network of “things” fails. This trust concern (and legal liability) becomes especially problematic when there are unplanned interactions between critical and noncritical components. In discussing IoT trust, there are two related questions: (1) what is the possibility of system failure, and (2) who is liable when the system fails [54].

Consider the first question: what is the possibility of system failure? The answer to this question is very difficult to determine. A powerful technique for determining the risks of a system-level failure would involve fault injection to simulate the effects of real faults as opposed to simulating the faults themselves. Until these risks can be accurately and scientifically measured, there likely will not be a means for probabilistically and mathematically bounding and quantifying liability [54].

Now consider the second question: who is liable when the system fails? For any non-interconnected system, the responsibility for failure lies with the developer (i.e., the individual, individuals, company, or companies, inclusive). For systems that are connected to other systems locally and through the Internet, the answer becomes more difficult. Consider the following legal opinion.

In the case of (planned) interconnected technologies, when there is a “malfunctioning thing,” it is difficult to determine the perimeter of the liability of each supplier. The issue is even more complex for artificial intelligence systems that involve a massive amount of collected data so that it might be quite hard to determine the reason why the system made a specific decision at a specific time [6].

Both planned and spontaneous interactions between critical and noncritical systems create significant risk and liability concerns. These interacting, dynamic, cross-domain ecosystems create the potential for increased threat vectors, new vulnerabilities, and new risks. Unfortunately, many of these will remain unknown unknowns until after a failure or successful attack has occurred.

In summary, this publication offers no unique recommendations for assessing and measuring reliability. The traditional reliability measurement approaches that have existed for decades are appropriate for a “thing” and a network of “things.” These approaches, as well as assessments of resilience, should be considered throughout a system’s life cycle.

14 Data Integrity

Data is the “blood” of any computing system, including IoT systems. If a network of “things” involves many sensors, there may be a significant amount of data.

The ability to trust data involves many factors: (1) accuracy, (2) fidelity, (3) availability, and (4) confidence that the data cannot be corrupted or tampered with. Whether any of these is more important than the other depends on the system’s requirements. However, with respect to a network of “things,” the timeliness with which the data is transferred is of particular importance. Stale, latent, and tardy data are trust concerns, and while that is not a direct problem with the “goodness” of the data itself, it is a performance concern for the mechanisms within the network of “things” that transfer data. In short, stale, latent, and tardy data in certain situations will be no worse than no data at all.

Cloud computing epitomizes the importance of trusting data. Where data resides is important. Where is the cloud? Can the data be leaked from that location? It is a tendency to think of “your data” on “your machine,” but in some cases, the data is not just “yours.” Leased data can originate from anywhere and from vendors at the time of their choosing and with the integrity of their choosing. Competitors can lease the same data [23][44].

The production, communication, transformation, and output of large amounts of data in networks of “things” creates various concerns related to trust. A few of these include:

- *Missing or incomplete data.* How does one identify and address missing or incomplete data? Here, missing or incomplete data could originate from a variety of causes, but in IoT, it probably refers to sensor data that is not released and transferred or databases of information that are inaccessible (e.g., clouds). Each network of “things” will need some level of resilience to be built in to allow a potentially crippled network of “things” to still perform even when data is missing or incomplete.

- *Data quality.* How does one address data quality? To begin, a definition is needed for what data quality means for a particular system. Is it fidelity of the information, accuracy of the information, or something else? Each network of “things” will need some description for an acceptable level of data quality.

- *Faulty interfaces and communication protocols.* How does one identify and address faulty interfaces and communication protocols? Since data is the “blood” of a network of “things,” then the interfaces and communication protocols are the veins and arteries of that system. Defective mechanisms that perform data transfer within a system if “things” are equally as damaging to the overall trust in the data as poor data quality and missing or incomplete data. Therefore, trust must exist in the data transfer mechanisms. Each network of “things” will need some level of resilience to be built in to ensure that the data moves from point A to point B in a timely manner. This solution might include fault tolerance techniques, such as redundancy of the interfaces and protocols.

- *Data tampering*. How does one address data tampering or even know it occurred? Rarely can tamperproof data exist if someone has malicious intent and the appropriate resources to fulfill that intent. Each network of "things" will need some type of a reliance plan for data tampering, such as a back-up collection of the original data in a different geographic location.

- *Data security and privacy*. How secure and private is the data from delay or theft? There are a seemingly infinite number of places in the dataflow of a network of "things" where data can be snooped by adversaries. This requires that the specification of a network of "things" have some risk assessment that assigns weights to the value of the data if it were to be compromised. Each network of "things" will need a data security and privacy plan.

- *Data leakage*. Can data leak, and, if so, would you know that it had? Assume a worst-case scenario where all networks of "things" leak. While this does not directly impact the data, it may well impact the business model of the organization that relies on the system of "things." If this is problematic, an analysis of where the leakage originates can be performed. However, this is technically difficult and costly.

While conventional techniques such as error-correcting codes, voting schemes, and Kalman filters could be used, specific recommendations for design principles need to be determined on a case-by-case basis.

15 Excessive Data

Any network of “things” is likely to have a dynamic and rapidly changing dataflow and workflow. There may be numerous inputs from a variety of sources, such as sensors, external databases or clouds, and other external subsystems. The potential for the generation of vast amounts of data over time renders IoT systems as potential “big data” generators. In fact, one report predicts that global data will reach 44 zettabytes (44 billion terabytes) by 2020 [7]. Note, however, that there will be networks of “things” that are not involved in receiving or generating large quantities of data (e.g., closed loop systems that have a small and specialized purpose). An example here would be a classified network that is not tethered to the Internet.

The data generated in any IoT system can be corrupted by sensors, aggregators, communications channels, and other hardware and software utilities [44]. Data is not only susceptible to accidental corruption and delay, but also malicious tampering, delay, and theft. As previously mention in Section 14, data is often the most important asset to be protected from a cybersecurity perspective.

Each of the primitives presented in [44] is a potential source for a variety of classes of corrupt data. Section 13 already discussed the problems of *propagation across* and *propagation from*. Although hyperbole, it is reasonable to visualize an executing network of “things” as a firework show. Different explosions occur at different times, although all are in timed coordination during a show. Networks of “things” are similar in that internal computations and the resulting data are in continuous generation until the IoT system performs an actuation or decision.

The dynamic of data being created quickly and used to create new data cannot be dismissed as a problem for testing. The vast amount of data that can be generated by networks of “things” makes the problem of isolating and treating corrupt data extremely difficult. The difficulty pertains to the problem of identifying corrupt data and the problem of making the identification quickly enough. If such identification cannot be made for a certain system in a timely manner, then trust in that system is an unreasonable expectation [56].

Certain data compression, error detection and correction, cleaning, filtering, and compression techniques may be useful both in increasing trust in the data and reducing its bulk for transmission and storage. No specific recommendations, however, are made.

16 Speed and Performance

The speed at which computations and data generation can occur in a network of “things” is increasing rapidly. Increased computational speed inhibits a system’s ability to log and audit any transactions as the rate of data generation exceeds the speed of storage. This situation, in turn, makes real-time forensic analysis and recovery from faults and failures more difficult as data is lost and computational deadlines become harder to meet. Consequently, there are fewer ways to “put on the brakes,” undo incorrect computations, and fix internal and external data anomalies. Furthermore, computing faster to a wrong outcome offers little trust.

A related problem is that of measuring the speed of any network of “things.” Speed-oriented metrics are needed for optimization, comparison between networks of “things,” and the identification of slowdowns that could be due to anomalies, all of which affect trust.

There are no simple speed metrics for IoT systems and no dashboards, rules for interoperability and composability, rules of trust, or established approaches to testing [55].

Possible candidate metrics to measure speed in an IoT system include:

- Time to decision once all requisite data is presented (an end-to-end measure)
- Throughput speed of the underlying network
- Weighted average of a sensor cluster’s “time to release data”
- Some linear combination of the above or other application domain-specific metrics

Note here that while better performance will usually be an “ility” of desire, it makes the ability to perform forensics on systems that fail much harder, particularly for systems where some computations occur so instantaneously that there is no “after the fact” trace of them.

Traditional definitions from real-time systems engineering can also be used, for example:

- *Response time*: the time between the presentation of a set of inputs to a system and the realization of the required behavior, including the availability of all associated outputs
- *Real-time system*: a system in which logical correctness is based on both the correctness of the outputs and their timelines
- *Hard real-time system*: a system in which failure to meet even a single deadline may lead to complete or catastrophic system failure
- *Firm real-time system*: a system in which a few missed deadlines will not lead to total failure, but missing more than a few may lead to complete or catastrophic system failure
- *Soft real-time system*: a system in which performance is degraded but not destroyed by failure to meet response-time constraints [20]

These traditional measures of performance can be recommended as building blocks for next-generation IoT trust metrics. For example, taking a weighted average of response times across a set of actuation and event combinations can give a “response time” for an IoT system. Once “response time” is defined, then notions of deadline satisfaction and designation of hard, firm, or soft real-time can be assigned. Furthermore, repositories of performance data for various types of IoT systems, devices, and communications channels should be created for benchmarking purposes and eventual development of standards.

17 Usability

One of the larger concerns in IoT trust is usability—the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of the user. It is, essentially, how "friendly" devices are to use and learn. This factor is an important consideration for most IT systems but may be more of a challenge with IoT where the user interface may be tightly constrained by limited display size and functionality or where a device can only be controlled via remote means. User interfaces for some device classes, such as Smart Home devices, are often limited to a small set of onboard features (e.g., LED status indicators and a few buttons) and a broader set of display and control parameters accessible remotely via a computer or mobile device. Some "smart" household items such as lightbulbs or faucets may have no direct interface on the device and must be managed through a computer or smart phone connected wirelessly.

Such limited interfaces have significant implications for user trust. How do users know what action to take to produce a desired response, and how does the device issue a confirmation that will be understood? Devices with only a small display and one or two buttons often require complex user interactions that depend on sequences and timing of button presses or similar non-obvious actions. Consequently, many basic security functions can only be accomplished using a secondary device such as a smart phone. For example, if the IoT device has only two buttons, a password update will have to be done through a secondary device. As a result of this usability problem, users become even less likely to change default passwords, leaving the device open to attack. This is just one example of the interplay between usability and other trust factors. The following discussion illustrates some of the complex interactions between usability engineering and factors such as performance, security, and synchronization.

Limited interfaces may, to some extent, be unavoidable with small devices but go against secure system principles, harkening to Kerckhoff's rules for crypto systems from the 19th century [18] and later extended to IT systems [36]. Among these is the principle that a secure system must be easy to use and not require users to remember complex steps. IoT systems run counter to this principle by their nature. Today, device makers are inventing user interfaces that often vary wildly from device to device and manufacturer to manufacturer, almost ensuring difficulty in remembering the right steps to follow for a given device.

One of the challenges of designing for IoT usability is the asynchronous operation imposed by device processing and battery limitations. Since devices may only be able to communicate periodically with possibly minutes to hours between transmissions, conditions at a given time may be different than indicated by the last data received from a device. Since decision triggers may require readings from multiple devices, it is likely that decisions may be based on at least some currently invalid values or that actions may be delayed as the system waits for updated values. In the worst case, badly-implemented IoT can "make the real world feel very broken" [42], such as when flipping a light switch results in nothing happening for some time while devices communicate.

18 Visibility and Discoverability

More than anything else, IoT represents the merger of information and communications technology with the physical world. This is an enormous change in the way that humans relate to technology and whose full implications will not be understood for many years. As with many aspects of technology, the change has been occurring gradually for some time but has now reached an exponential growth phase. However, by its nature, this merger of information technology with the physical world is not always obvious. Mark Weiser, who coined the term “ubiquitous computing” nearly 30 years ago, said, “The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it” [58]. Today, this vision is coming true as IoT devices proliferate into every aspect of daily life. According to one study, within four years there will be more than 500 IoT devices in an average household [13] so that they truly are beginning to disappear.

Is this disappearance uniformly a good thing? If a technology is invisible, then users will not be aware of its presence or what it is doing. Trust issues related to this new technological world made news when reports suggested that smart televisions were "eavesdropping" on users [43][54]. Voice-operated remote controls in smart televisions can only work if the televisions are always “listening,” but the trust implications are obvious. To resolve trust concerns in cases like this, appliances need to be configurable for users to balance convenience with their personal security and privacy requirements, and device capabilities need to be visible with clear explanation of implications.

A different set of trust concerns is involved with technical aspects of device discovery in networks of “things.” The traditional Internet was built almost entirely on the TCP/IP protocol suite with HTML for web sites running on top of TCP/IP. Standardized communication port numbers and internationally agreed web domain names enabled consistent operation regardless of the computer or router manufacturer. Smartphones added the Bluetooth protocol for devices. This structure has not extended to IoT devices because they generally do not have the processing power to support it. Instead, a proliferation of protocol families has developed by different companies and consortia, including Bluetooth Low Energy (BLE), ZigBee, Digital Enhanced Cordless Telecommunications Ultra Low Energy (DECT ULE), and a collection of proprietary technologies for Low Power Wide Area Networks (LPWAN). These many technologies result in a vast number of possible interactions among various versions of software and hardware from many different sources.

Most computer users are familiar with problems that arise when some business application or other software will not run because other software was changed on the system and the two packages are no longer compatible. At least with PCs and mainframes, a person generally has a good idea of what is running on the system. With 500 IoT devices in a home, will the homeowner even know where the devices are located? How do devices make their presence known with multiple protocols? It may not be clear from day-to-day what devices are on a network or where they are, much less how they are interacting.

Device discovery is a complex problem for networks of “things” [3][41], but the general problem of discovery within networks has been studied for decades. There are generally two approaches:

- *Centralized:* Nodes register with a central controller when they are brought into a network. The controller manages a database of currently available devices and periodically sends out heartbeat messages to ensure devices are available, dropping from the database any that do not respond.
- *Distributed:* In this case, devices conduct a search for partner devices with the necessary features by broadcasting to the local network. This approach avoids the need for a central controller, providing flexibility and scalability.

Scalability requirements for networks of hundreds of things often lead to implementing the distributed approach, but trust issues have enormous implications for device discovery in a large network. Without sophisticated cryptographically-based authentication mechanisms, it becomes very difficult to ensure trusted operation in a network. For example, it has been shown that malware installed on a smartphone can open paths to other IoT devices, leaving the home network fully vulnerable to attack [38]. This is possible primarily because many IoT devices have little or no authentication, often due to the resource constraints described earlier.

Discoverability of IoT devices is thus a key problem for trust. Its dimensions include human factors, such as users' trust in behavior of devices (e.g., the smart TV example and technical issues of authentication among devices). Solutions will require the adoption of some common protocols, and it may take years to develop consensus standards or for *de facto* proprietary standards to emerge. In many cases, there will also be organizational challenges since different kinds of devices may be installed by different departments. Organizations will need to know what devices are present to manage security or to simply avoid duplication of effort. This need can be addressed with audit tools that can identify and catalog devices on the network, reducing dependence on user cooperation but requiring trust in the audit tools.

19 Summary

This publication has enumerated 17 technical trust concerns for any IoT system based on the primitives presented in [44]. These systems have significant differences compared to traditional IT systems, such as much smaller size and limited performance, larger and more diverse networks, minimal or no user interface, lack of consistent access to reliable power and communications, and many others. These differences necessitate new approaches to planning and design. An essential aspect of developing these new systems is understanding the ways in which their characteristics can affect user trust and avoiding a "business as usual" approach that might be doomed to failure in the new world of IoT.

For each of the technical concerns, this publication introduced and defined the trust issues, pointed out how they differ for IoT compared to traditional IT systems, gave examples of their effects in various IoT applications, and, when appropriate, outlined solutions to dealing with trust issues. Some of these recommendations apply not only to IoT systems but to other traditional IT systems as well. For some of the trust issues, IoT introduces complications that defy easy answers in the current level of development. These are noted as requiring research or industry consensus on solutions. This document thus offers the additional benefit of providing guidance on needed standards efforts and research into how to better trust IoT systems.

Appendix A—Insurability and Risk Measurement

IoT trust issues truly come to the fore in assessing the impact of this new technology on insurability and risk management because insurance requires that risk be measured and quantified. In this area, the emergence of IoT can have significant tradeoffs—networks of “things” can make it easier to estimate risk for the physical systems in which devices are embedded but estimating risk for the device networks themselves may be much more difficult than for conventional IT systems.

Cars, homes, and factories with embedded sensors provide more data than ever, making it possible to estimate their risks more precisely, which is a huge benefit for insurers [5]. For example, auto insurance companies have begun offering lower rates for drivers who install tracking devices in their vehicles to report where, how, and how fast they drive. Depending on a user's privacy expectations, there are obvious trust issues, and the legal aspects of employers installing such devices to monitor employee driving are just now being developed [14]. Additionally, an often neglected aspect of such devices is the possible tradeoff between reducing risk by measuring the physical world, such as with driving, and the potential increased risks from a complex network of things being introduced into a vehicle or other life-critical system. Already, there have been claims that vehicle tracking devices have interfered with vehicle electronics, possibly leading to dangerous situations [28]. Examples include claims of losing headlights and tail lights unexpectedly and complete shutdown of the vehicle [16] resulting from unexpected interactions between the vehicle monitor and other components of the car's network of “things.”

In addition to estimating risk—and thus insurability—of systems with embedded IoT devices, cybersecurity risks may become much harder to measure. Quantifying potential vulnerability even for conventional client-server systems, such as e-commerce, is not well understood, and reports of data loss are common. As a result, insurance against cybersecurity attacks is expensive—a $10 million policy can cost $200,000 per year because of the risk [17]. It will be much more difficult to measure risk for IoT networks of thousands of interacting devices than it is even for a corporate system made up of a few hundred servers and several thousand client nodes. IoT interactions are significantly more varied and more numerous than standard client-server architectures. Risk estimation for secure systems requires measurement of a *work factor*, the time and resource cost of defeating a security measure. The same principle has been applied to vaults and safes long before the arrival of IT systems. The cost of defeating system security must be much higher than the value of the assets protected so that attackers are not motivated to attempt to break in. The problem for networks of things is that there are few good measures of the work factor involved in breaking into these systems. They are not only new technology but have vast differences depending on where they are applied, and it is difficult to evaluate their defenses.

From a protection-cost standpoint, IoT systems also have a huge negative tradeoff—the typical processor and memory resource limitations of the devices make them easier to compromise, while at the same time, they may have data as sensitive as what is on a typical PC or, in extreme cases, may present risks to life and health. Implantable medical devices can be much harder to secure than a home PC, but the risks are obviously much greater [30][34]. Determining the work factor in breaking the security of such devices and "body area networks" is an unsolved problem.

A basic goal may be to ensure that life-critical IoT devices adhere to sound standards for secure development [15], but estimating risk for such systems is likely to remain a challenge.

To complicate matters further, IoT systems often provide functions that may inspire *too much* trust from users. Drivers who placed unwarranted trust in vehicle autonomy have already been involved in fatal crashes, with suggestions that they were inattentive and believed the car could successfully avoid any obstacle [37]. Establishing the *right level of trust* for users will likely be a human factor challenge with IoT systems for many years to come.

No specific recommendations are made here. It is inevitable that insurers and systems engineers will eventually develop appropriate risk measures and mitigation strategies for IoT systems.

Selected acronyms and abbreviations used in this paper are defined below.

Appendix B—Regulatory Oversight and Governance

Regulations have the power to significantly shape consumer interaction with technologies. Consider motor vehicles, whose safety is regulated by the National Highway Traffic Safety Administration (NHTSA) [26]. NHTSA enforces the Federal Motor Vehicle Safety Standards, which specify minimum safety compliance regulations for motor vehicles to meet. Notable stipulations include requiring seatbelts in all vehicles, which can help reduce fatalities in the case of vehicular accidents. NHTSA likewise licenses vehicle manufacturers—helping regulate the supply of vehicles that consumers can buy—and provides access to a safety rating system that consumers can consult. Multiple studies have shown the potential for regulations to continue to increase the safety of motor vehicles (e.g., [27]).

Regulatory oversight and governance have been established in most domains for the safety of critical systems. However, there is no parallel to the NHTSA for IoT systems:

1. There are no regulations on the security of IoT devices.
2. There is no oversight on the licensing of IoT device manufacturers.
3. There are no governing authorities evaluating the security of IoT devices.

These problems are compounded due to the economics behind IoT: the barrier to entry to constructing an IoT device is low, meaning that the market contains many different devices and models from many different manufacturers with very few authoritative bodies attesting to the security of any of these devices. While these problems extend into the traditional computing market (i.e., laptops and personal computers), market mechanics have since driven most products toward consolidated products and features, making it easier for consumers to evaluate and understand the security offered by the devices and manufacturers.

Nonetheless, while there is no central entity regulating the security of IoT devices, recent progress has been seen as regulatory participants consider how they want to approach this complex problem. As an example, the Internet of Things Cybersecurity Improvement Act [57] was introduced in 2017 with the goal of setting standards for IoT devices specifically installed in government networks. The bill contains several important stipulations, including requiring devices to abandon fixed, default passwords and not have any known vulnerabilities. The Act also relaxes several other acts that could be used to prosecute security researchers looking to test the safety of these devices.

The mandates of several agencies border the IoT security space. A good example of this is the Federal Trade Commission (FTC). In January 2018, VTech Electronics agreed to settle charges by the FTC that they violated not a security law, but rather U.S. children's privacy law, collecting private information from children without obtaining parental consent and failing to take reasonable steps to secure the data [12]. The key phrase is that last point: VTech's products were Internet-connected toys (i.e. IoT devices) which collected personal information, and due to security risks in how these devices handled and managed data, the company was fined. This case shows that if IoT devices don't have reasonable security, a manufacturer may be held liable.

The U.S. Consumer Product Safety Commission has called for more collaboration between lawyers and experts in the area [1]. Outside of the U.S., the European Union Agency for Network and Information Security (ENISA) has published recommended security guidelines for IoT [10]. As more calls for security and recommendations arise, standardization and regulation may follow, increasing the security and safety of deployed IoT systems.

Regulations offer a serious means to help increase the security and safety of IoT systems, as evidenced by their successes in other industries such as vehicle manufacturing. While some improvements have been noticed as some agencies and organizations attempt to wield influence in IoT regulation, no single, central organization has mandated rules regarding the use and development of IoT systems. Such an organization could have a significant positive impact on the security and safety of IoT systems and consumers' lives.

Appendix C—Six Trustworthiness Elements in NIST SP 800-183

Six trustworthiness elements are listed in Section 3 of NIST SP 800-183. The verbatim text for those six is given here, and note that NoT stands for network of "things":

[begin verbatim text]

To complete this model, we define six elements: *environment, cost, geographic location, owner, Device_ID*, and *snapshot*, that although are not primitives, are key players in trusting NoTs. These elements play a major role in fostering the degree of trustworthiness[5] that a specific NoT can provide.

1. **Environment –** The universe that all primitives in a specific NoT operate in; this is essentially the *operational profile* of a NoT. The environment is particularly important to the sensor and aggregator primitives since it offers context to them. An analogy is the various weather profiles that an aircraft operates in or a particular factory setting that a NoT operates in. This will likely be difficult to correctly define.

2. **Cost –** The expenses, in terms of time and money, that a specific NoT incurs in terms of the non-mitigated reliability and security risks; additionally, the costs associated with each of the primitive components needed to build and operate a NoT. Cost is an estimation or prediction that can be measured or approximated. Cost drives the design decisions in building a NoT.

3. **Geographic location –** Physical place where a sensor or *e*Utility operates in, e.g., using RFID to decide where a 'thing' actually resides. Note that the operating location may change over time. Note that a sensor's or *e*Utility's geographic location along with communication channel reliability and data security may affect the dataflow throughout a NoT's workflow in a timely manner. Geographic location determinations may sometimes not be possible. If not possible, the data should be suspect.

4. **Owner** - Person or Organization that owns a particular sensor, communication channel, aggregator, decision trigger, or *e*Utility. There can be multiple owners for any of these five. Note that owners may have nefarious intentions that affect overall trust. Note further that owners may remain anonymous. Note that there is also a role for an **operator**; for simplicity, we roll up that role into the owner element.

5. **Device_ID –** A unique identifier for a particular sensor, communication channel, aggregator, decision trigger, or *e*Utility. Further, a Device_ID may be the only sensor data transmitted. This will typically originate from the manufacturer of the entity, but

[5] *Trustworthiness* includes attributes such as security, privacy, reliability, safety, availability, and performance, to name a few.

it could be modified or forged. This can be accomplished using RFID[6] for physical primitives.

6. **Snapshot** – an instant in time. Basic properties, assumptions, and general statements about snapshot include:

    a. Because a NoT is a distributed system, different events, data transfers, and computations occur at different snapshots.

    b. Snapshots may be aligned to a clock synchronized within their own network [NIST 2015]. A global clock may be too burdensome for sensor networks that operate in the wild. Others, however, argue in favor of a global clock [Li 2004]. This publication does not endorse either scheme at the time of this writing.

    c. Data, without some "agreed upon" time stamping mechanism, is of limited or reduced value.

    d. NoTs may affect business performance – sensing, communicating, and computing can speed-up or slow-down a NoT's workflow and therefore affect the "perceived" performance of the environment it operates in or controls.

    e. Snapshots maybe tampered with, making it unclear when events actually occurred, not by changing time (which is not possible), but by changing the recorded time at which an event in the workflow is generated, or computation is performed, e.g., sticking in a **delay()** function call.

    f. Malicious latency to induce delays, are possible and will affect when decision triggers are able to execute.

    g. Reliability and performance of a NoT may be highly based on (e) and (f).

[end verbatim text]

This publication has taken Section 3 from NIST SP 800-183 and expanded it into a richer discussion as to why trusting IoT products and services is difficult. This document has derived 17 new technical trust concerns from the six elements in NIST SP 800-183. For example, the snapshot element briefly mentioned in NIST SP 800-183 is discussed in detail in Section 7 concerning a lack of precise timestamps.

[6] RFID readers that work on the same protocol as the inlay may be distributed at key points throughout a NoT. Readers activate the tag causing it to broadcast radio waves within bandwidths reserved for RFID usage by individual governments internationally. These radio waves transmit identifiers or codes that reference unique information associated with the item to which the RFID inlay is attached, and in this case, the item would be a primitive.

1322 ## Appendix D—References

[1] American Bar Association. Consumer Product Safety Administration seeks collaboration in managing internet of things, https://www.americanbar.org/news/abanews/aba-news-archives/2017/05/consumer_productsaf.html [accessed 7/21/18].

[2] K. Ashton, "That 'Internet of Things' Thing," *RFID Journal*, June 2009. http://www.rfidjournal.com/articles/view?4986 [accessed 5/9/17].

[3] O. Bello, S. Zeadally, & M. Badra, "Network layer inter-operation of Device-to-Device communication technologies in Internet of Things (IoT)," *Ad Hoc Networks*, pp. 57, 52-62, 2017.

[4] H. Chung, M. Iorga, & J. Voas, "Alexa, can I trust you?", *IEEE Computer*, September 2017.

[5] L. Columbus, "Internet of Things market to reach $267B by 2020", *Forbes*, January 2017. https://www.forbes.com/sites/louiscolumbus/2017/01/29/internet-of-things-market-to-reach-267b-by-2020/#f2ddc5609bd6 [accessed 1/5/2018].

[6] G. Coraggio, "The Internet of Things and its legal dilemmas," *VC Experts Blog*, December 2016. https://www.blog.vcexperts.com/2016/12/15/the-internet-of -things-and-its-legal-dilemmas.

[7] DataIQ News, "Big data to turn 'mega' as capacity will hot 44 zettabytes by 2020," April 2014. http://www.dataiq.co.uk/news/20140410/big-data-turn-mega-capacity-will-hit-44-zettabytes-2020.

[8] P. Day. (2015, March 19). *Peter Day's World of Business* [Audio podcast] retrieved from http://downloads.bbc.co.uk/podcasts/radio/worldbiz/worldbiz_20150319-0730a.mp3.

[9] G. Dhadyalla, N. Kumari, & T. Snell, "Combinatorial Testing for an Automotive Hybrid Electric Vehicle Control System: A Case Study," Proc. IEEE 7th Int'l Conf. Software Testing, Verification and Validation Workshops (ICSTW 14), pp. 51–57, 2014.

[10] European Union Agency for Network and Information Security. Baseline Security Recommendations for IoT, November 2017. https://www.enisa.europa.eu/publications/baseline-security-recommendations-for-iot [accessed 7/21/2018].

[11] M. S. Farasha et al. "An efficient user authentication and key agreement scheme for heterogeneous wireless sensor network tailored for the Internet of Things environment," *Ad Hoc Networks*, 36(1), January 2016. https://www.sciencedirect.com/science/article/pii/S1570870515001195.

[12] Federal Trade Commission, "Electronic toy maker VTech settles FTC allegations that it violated children's privacy law and the FTC Act," January 2018. https://www.ftc.gov/news-events/press-releases/2018/01/electronic-toy-maker-vtech-settles-ftc-allegations-it-violated [accessed 7/21/2018].

[13] Gartner, "Gartner says a typical family home could contain more than 500 smart devices by 2022," September 2014. http://www.gartner.com/newsroom/id/2839717.

[14] K. Grossenbacher, "The legality of tracking employees by GPS," *Employment Law Lookout*, January 2016. https://www.laborandemploymentlawcounsel.com/2016/01/the-legality-of-tracking-employees-by-gps/.

[15] T. Haigh & C. Landwehr, "Building code for medical device software security," *IEEE Cybersecurity*, May 2015.

[16] G. Horcher, "Concerns with insurance devices that monitor for safe driver discounts," *Atlanta Journal Constitution*, 2014.

[17] J. Kamp & S. Calvert, "Ransom demands and frozen computers: Hackers hit towns across the U.S.," *Wall Street Journal*, June 2018. https://www.wsj.com/articles/ransom-demands-and-frozen-computers-hackers-hit-towns-across-the-u-s-1529838001.

[18] A. Kerckhoffs, "La cryptographie militaire," *Journal des sciences militaires*, Vol. IX, pp. 5-83, 161-191, 1883. http://www.petitcolas.net/kerckhoffs/la_cryptographie_militaire_i.htm.

[19] C. Kolias, G. Kambourakis, A. Stavrou., & J. Voas, "DDoS in the IoT: Mirai and other botnets," *IEEE Computer*, pp. 80-84, 2017.

[20] P. Laplante & S. Ovaska, *Real-time systems design and analysis*, 4th Ed., IEEE Press, 2012.

[21] E. Lear, R. Droms, & D. Romascanu, "Manufacturer Usage Description Specification," IETF draft, 2017.

[22] K. Miller & J. Voas," Software Certification Services: Encourage trust and reasonable expectations," *IEEE IT Professional*, pp. 39-44, 2006.

[23] K. Miller, J. Voas, & P. Laplante, "In trust we trust," *IEEE Computer*, pp. 91-92, 2010.

[24] B. Moran, M. Meriac, & H. Tschofenig, "A firmware update architecture for Internet of Things devices," *IETF*, 2017. https://tools.ietf.org/id/draft-moran-suit-architecture-00.html.

[25] J. Musa, A. Iannino, & K. Okumoto, "Software Reliability: Measurement, prediction, application," McGraw-Hill, 1987.

[26] National Highway Traffic Safety Administration. https://www.nhtsa.gov/. [accessed 7/20/2018].

[27] G.W. Neeley & L.E. Richardson, Jr., "The effect of state regulations on truck-crash fatalities," *American Journal of Public Health*, pp. 408-415, 2009.

[28] Neilson, "Insurance tracking device blamed for car damage," April 2014. https://www.programbusiness.com/News/Insurance-Tracking-Device-Blamed-for-Car-Damage.

[29] P.G. Neumann, *Risks Forum*, June 2018.

[30] L.H. Newman, "Medical devices are the next security nightmare," *WIRED*, 2017.

[31] A.H. Patil, N. Goveas, & K. Rangarajan, "Test suite design methodology using combinatorial approach for Internet of Things operation systems," *J. Software Engineer Applications*, Vol. 8, No. 7, pp. 303, 2015.

[32] R. Reiss, "5 ways the IoT will transform the insurance industry," *Forbes*, February 2016. https://www.forbes.com/sites/robertreiss/2016/02/01/5-ways-the-iot-will-transform-the-insurance-industry/.

[33] K. Rose et al., "The Internet of Things (IoT): An overview," *The Internet Society*, pp. 5, 2015. http://www.internetsociety.org/sites/default/files/ISOC-IoT-Overview-20151221-en.pdf.

[34] M. Rushanan, A.D. Rubin, D.F. Kune, & C.M. Swansons, "SoK: Security and privacy in implantable medical devices and body area networks," *Security and Privacy, IEEE Symposium*, pp. 524-539, 2014.

[35] T. Salman, "Internet of Things protocols." https://www.cse.wustl.edu/~jain/cse570-15/ftp/iot_prot/.

[36] J.H. Saltzer & M.D. Schroeder, "The protection of information in computer systems," *Proceedings of the IEEE*, pp. 1278-1308, 1975.

[37] F. Siddiqui & M. Laris, "Self-driving Uber vehicle strikes and kills pedestrian," *Washington Post*, March 2018. https://www.washingtonpost.com/news/dr-gridlock/wp/2018/03/19/uber-halts-autonomous-vehicle-testing-after-a-pedestrian-is-struck/.

[38] V. Sivaraman, D. Chan, D. Earl, & R. Boreli, "Smart-phones attacking smart-homes," *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Devices*, pp. 195-200, 2016.

[39] S. Soper, "This is how Alexa can record private conversations," 2018. https://www.bloomberg.com/news/articles/2018-05-24/amazon-s-alexa-eavesdropped-and-shared-the-conversation-report.

[40] A. Stavrou & J. Voas, "Verified time," *IEEE Computer*, Vol. 50, 2017.

[41] J. Sunthonlap, P. Nguyen, H. Wang, M. Pourhomanyoun, Y. Zhu, & Z. Ye, "SAND: A social-aware and distributed scheme for device discovery in the Internet of Things," *2018 International Conference on Computing, Networking, and Communication (ICNC)*, pp. 38-42, 2018.

[42] M. Treseler, "How is UX for IoT different?" http://radar.oreilly.com/2014/11/how-is-ux-for-iot-different.html.

[43] H. Tsukayama, "Samsung: Our televisions aren't secretly eavesdropping on you," https://www.washingtonpost.com/news/the-switch/wp/2015/02/10/samsung-our-televisions-arent-secretly-eavesdropping-on-you/?noredirect=on&utm_term=.5322af153a88.

[44] J. Voas, *Networks of Things*, NIST Special Publication (SP) 800-183, National Institute of Standards and Technology, Gaithersburg, Maryland, 2016.

[45] J. Voas, F. Charron, & K. Miller, "Tolerant Software Interfaces: Can COTS-based systems be trusted without them?" *Proceedings of the 15th International Conference on Computer Safety, Reliability and Security (SAFECOMP'96)*, Springer-Verlag, pp. 126-135, October 1996.

[46] J. Voas, "Error propagation analysis for COTS systems," *IEEE Computing and Control Engineering Journal*, 8(6), pp. 269-272, December 1997.

[47] J. Voas, "Certifying off-the-shelf software components," *IEEE Computer*, 31(6): 53-59, June 1998. (Translated into Japanese and reprinted in Nikkei Computer magazine)

[48] J. Voas, "The Software Quality Certification Triangle," *Crosstalk*, 11(11), pp. 12-14, November 1998.

[49] J. Voas, “Certifying software for high assurance environments,” *IEEE Software*, 16(4), pp. 48-54, July 1999.

[50] J. Voas & J. Payne, “Dependability certification of software components,” *Journal of Systems and Software*, Vol. 52, p. 165-172, 2000.

[51] J. Voas, “Toward a Usage-Based Software Certification Process,” *IEEE Computer*, 33(8), pp. 32-37, August 2000.

[52] J. Voas, “Software’s secret sauce: The `ilities’,” Quality Time Column, *IEEE Software*, 21(6), pp. 2-3, November 2004.

[53] J. Voas & G. Hurlburt, “Third-party software’s trust quagmire”, IEEE *Computer*, December 2015.

[54] J. Voas & P. Laplante, “The IoT Blame Game,” *IEEE Computer*, 2017.

[55] J. Voas, R. Kuhn, & P. Laplante, “IoT Metrology,” *IEEE IT Pro*, May 2018.

[56] J. Voas & P. Laplante, “IoT’s certification quagmire,” *IEEE Computer*, April 2018.

[57] Weaver, N, “The Internet of Things Cybersecurity Improvement Act: A good start on IoT security,” August 2017. https://www.lawfareblog.com/internet-things-cybersecurity-improvement-act-good-start-iot-security [accessed 7/21/2018].

[58] Weiser, M. “The computer for the 21st century,”. *Scientific American*, *265*(3), pp. 94-105, September 1991.

[59] J. Winter, “Algorithmic discrimination: Big data analytics and the future of the Internet,” *The Future Internet: Alternative Visions*, pp. 127, 2015. http://tinyurl.com/zjqh9gc.

[60] J. Yang, H. Zhang, & J. Fu, “A fuzzing framework based on symbolic execution and combinatorial testing,” *Green Computing and Communications IEEE International Conference*, pp. 2076-2080, 2013.

Appendix E—Abbreviations

AI	Artificial Intelligence
BBC	British Broadcasting Corporation
BLE	Bluetooth Low Energy
COTS	Commercial Off-the-Shelf
DECT ULE	Digital Enhanced Cordless Telecommunications Ultra Low Energy
ENISA	European Union Agency for Network and Information Security
FTC	Federal Trade Commission
GPS	Global Positioning System
HTML	Hypertext Markup Language
HTTPS	Hypertext Transfers Protocol Secure
IETF	Internet Engineering Task Force
IIOT	Industrial Internet of Things
IoT	Internet of Things
IT	Information Technology
LPWAN	Low Power Wide Area Network
MUD	Manufacturer Usage Description
NHTSA	National Highway Traffic Safety Administration
NIST	National Institute of Standards and Technology
NoT	Network of Things
PC	Personal Computer
RFID	Radio Frequency identification
SLOC	Source Lines of Code
TCP/IP	Transmission Control Protocol / Internet Protocol