Special Session: The Future of IoT Security

Sibin Mohan*, Mikael Asplund†, Gedare Bloom‡, Ahmad-Reza Sadeghi¶,
Ahmad Ibrahim¶, Negin Salajageh‖, Paul Griffioen** and Bruno Sinopoli**
*University of Illinois at Urbana-Champaign, †Linköping University, ‡Howard University,
¶Technische Universität Darmstadt, ‖Visa Research, **Carnegie Mellon University
{*sibin@illinois.edu, †mikael.asplund@liu.se, ‡gedare@scs.howard.edu, ¶ahmad.sadeghi@trust.tu-darmstadt.de,
¶ahmad.ibrahim@trust.tu-darmstadt.de, ‖msalajeg@visa.com, **pgriffi1@andrew.cmu.edu, **brunos@ece.cmu.edu}

***Abstract*—The Internet-of-Things (IoT) is a large and complex domain. These systems are often constructed using a very diverse set of hardware, software and protocols. This, combined with the ever increasing number of IoT solutions/services that are rushed to market means that most such systems are rife with security holes. Recent incidents (*e.g.,* the Mirai botnet) further highlight such security issues.**

With emerging technologies such as blockchain and software-defined networks (SDNs), new security solutions are possible in the IoT domain. In this paper we will explore future trends in IoT security: *(a)* the use of blockchains in IoT security, *(b)* data provenance for sensor information, *(c)* reliable and secure transport mechanisms using SDNs *(d)* scalable authentication and remote attestation mechanisms for IoT devices and *(e)* threat modeling and risk/maturity assessment frameworks for the domain.

I. Introduction

It is estimated that the number of sensors and corresponding devices in the Internet-of-Things (IoT) domain is projected to grow beyond 1 trillion devices [1] by 2030 (and beyond). While this opens up immense economic opportunities and improved services or even significant changes to society, the proliferation of such devices can also raise significant security concerns. In fact, many recent incidents have either targeted the IoT systems [2] or used them as gateways to launch attacks against other systems [3], [4]. Security experts believe that such attacks and incidents will increase in quantity and efficacy in the near future. With the advent of 'fog' and 'edge' computing [5] as well as the increased use of new technologies such as blockchains [6], the problems are only expected to worsen.

As mentioned earlier, IoT systems generate a lot of data and hence security solutions need to focus on not just the computational nodes but also the network that carries the data. In addition, new techniques are required that allows users and/or designers to check the provenance of the data that is generated. Establishing trust for the computational nodes in the IoT system will also improve the trust in the data that is generated; so does the increased resiliency in the underlying networks that carry this information (either between local nodes or even across larger geographical distances).

Hence, improving the security for IoT systems and devices will require efforts along multiple lines – from fundamental new theories on using blockchains and related technologies for improved logging and tracking (Section II) to better attestation (Section VI) and authentication (Section V) mechanisms for IoT devices. Understanding the threats faced by such systems and developing methods to mitigate them are important (Section VII). Ensuring the integrity and trustworthiness of sensor data is important for maintaining the integrity of IoT systems (Section III) and so is the reliable transportation of such data (Section IV).

In this paper we explore these topics with the perspective of building secure IoT systems for the future. We identify some of the important issues and present initial thoughts and approaches to solving them. We hope that this will provide useful hints and directions to designers (and users) of such systems.

II. Blockchains for IoT security

Blockchain technology has become one of the more prevalent concepts in the area of distributed computing. However, it is sometimes difficult to judge how well this technology is able to meet the needs and requirements of a particular application. Fueled by a speculation economy, anti-establishment sentiments as well as media and marketing logic, blockchains have been hailed as the solution to all sorts of problems, including that of IoT security. But is this interest warranted?

In its most pure form, a blockchain is simply a data structure composed of data blocks where each block links to the previous block by containing a hash of the former's contents. Moreover, this data structure is typically distributed among a set of nodes thus forming a distributed database. The key property of this database compared to traditional database systems is the guaranteed absence of modifications to the data by a minority of the participants in the system.

There are a number of use cases for such storage mechanisms that are applicable to IoT security. For example, logging device behaviour in a blockchain over time allows security monitoring as well as ensuring that system resources are fairly distributed over time. Consider the case of vehicular platooning where the second vehicle in a platoon gains much more speed than the leading vehicle. Having a global record of vehicle behaviour can be used to enforce some level of fairness in such a system. Another interesting use case for blockchains is to keep track of device status information. Basic information such as firmware version, or more detailed information such as current configuration parameters or activity levels stored over time can be used to detect anomalies in the device's operation.

So what is stopping blockchains from already being widely adopted as a basic IoT technology and potentially for security in such systems? One of the main challenges to deploy blockchains in the context of IoT devices is to control the creation of blocks in the blockchain (which is sometimes inaccurately referred to as a consensus layer). So far, there are three main approaches to solve this problem – proof-of-work (PoW), proof-of-stake (PoS) and permissioned systems. Each of these approaches come in many flavours and they differ greatly in terms of timing performance, scalability, resource efficiency and architectural requirements.

For IoT devices, PoW is clearly not feasible since it consumes huge amounts of resources. PoS is primarily adapted to financial applications and can be difficult to apply in many IoT domains. This leaves us with the approach of using a closed, permissioned system (*e.g.,* the Hyperledger Fabric). Essentially, such an approach brings us back to a more traditional view of distributed computing where shared state is managed through a consensus protocol such as PBFT.

However, byzantine fault tolerance is still hampered by considerable performance issues since every transaction is subject to a voting process among the participating entities in the consensus algorithm. While there are lots of ways this process can be optimized, we lose some of the elegance of the probabilistic PoW mechanism. The so-called Nakamoto consensus used in many cryptocurrencies allows as much as 50% malicious nodes in an open environment (not considering selfish miners), whereas deterministic byzantine fault tolerance can only tolerate one-third of the nodes being intruders.

One potential way forward to solve this conundrum is to continue exploring the probabilistic approach to leader election, but adapted to an IoT setting. This will require solving several sub-problems, including *(i)* ensuring proper group membership information that prevents sybil attacks, *(ii)* limiting the amount of damage that can be caused by malicious nodes that are temporarily in control of block creation and *(iii)* modelling the level of probabilistic security achieved under different parameter settings, allowing proper choices to be made in the system design.

III. Trustworthy Sensor Data

A major challenge in IoT security is ensuring the integrity and trustworthiness of sensor data streaming from the edge. Sensor data integrity is necessary to prevent or detect sensor attacks. Securing the integrity of sensors in the IoT is challenging because of the fundamental difference in the design patterns of IoT applications in contrast to general-purpose computing [7]. Sensor data integrity can be achieved through one of three accepted approaches: (1) cryptographic integrity, (2) Byzantine agreement, and (3) data provenance. Each approach has benefits and drawbacks.

Cryptographic integrity is widely used in general-purpose computing with cryptographic hash functions and digital signatures. Unfortunately, cryptographic algorithms tend to incur high resource costs in both time, space, and power. The resource constraints of IoT devices makes adoption of traditional cryptography difficult. An active research area to explore in addressing this difficulty is lightweight cryptography [8].

Byzantine agreement solves the Byzantine Generals Problem, which is to achieve consensus in a distributed system among honest nodes despite a fractional number of Byzantine nodes. Byzantine fault tolerant systems can handle up to one-third of the nodes of a system acting arbitrarily (malicious), but these solutions are tailored for large-scale distributed systems, and are not suitable for resource-constrained IoT edge devices. Relaxing the consensus guarantees leads to approximate Byzantine agreement algorithms, which show promise for IoT sensor fusion [9].

Data provenance provides a complete history of transformations made to data. This history provides traceability and transparency for the source and modifications of data, and can establish the data's trustworthiness and integrity [10]. Two provenance data models focused on IoT are PAIoT [11], [12] and ProvThings [13]. The main problem with provenance is that maintaining a complete history easily leads to exhaustion of available storage space. Log files of events consume linear space, a fixed cost per log entry, while provenance graphs derived from log files may consume polynomial space based on the graph size. Existing work addresses this problem through compression [14], [15] and policy-based pruning [16]. It is unclear whether or to what extent the prior art is applicable for IoT edge devices, which (1) have storage constraints that even (lossless) compression may not satisfy, and (2) do not have any uniform policy to leverage for guiding pruning algorithms. Other aspects of IoT provenance that require further investigation include ensuring security and privacy of the provenance itself [17].

Solutions for sensor data integrity and trustworthiness may also combine multiple approaches. For example, blockchains use both cryptographic integrity, by way of Merkle trees, and Byzantine agreement using Nakamoto Consensus. More work is needed in all three solution areas for building trust and ensuring integrity of sensor data in ways that adhere to the low-cost, resource-constrained requirements of IoT devices.

IV. Reliable Networks for IoT Systems

IoT systems often carry critical data that must be transferred across the network (either local networks or even across the Internet). This could include critical sensor information, financial data (*e.g.,* payment transactions), important audio/video feeds, control commands, *etc.* Oftentimes, such data streams/network flows will require isolation and performance guarantees to ensure that *(a)* any design constraints (*e.g.,* end-to-end QoS guarantees such as maximum time for a payment system to respond to a user action) are met and *(b)* malicious actors are unable to either peer into the data streams or even disrupt them. There have been multiple instances of IoT networks being targeted by attackers – from distributed denial of service (DDoS) aimed at IoT networks [18] to IoT devices being used at botnets to launch other attacks (*e.g.,* Mirai [3]). Such indiscriminate attacks or worse, targeted attacks that aim to destabilize critical flows/information on the IoT network, can result in devastating consequences to the IoT systems, the operators, or even other, connected, systems. In this section, we will discuss techniques to use software defined networks (SDNs) to manage IoT networks – *(a)* from improving the management of the network flows in such systems to *(b)* better isolation and hence, security of flows (especially the *critical flows*) in such systems and *(c)* even improved resiliency to failure/attacks for critical flows.

Software-defined networks [19] have become increasingly popular for managing networks of all sizes – from cloud and data computing systems (*e.g.,* [20], [21]) to embedded and real-time systems [22]. SDNs allow for global visibility into the network and also for better management and control. This is achieved by separating the control plane from the data plane – a (logically) centralized *controller* (*e.g.,* RYU, OpenDaylight, *etc.* [23]) decides what actions must be taken for each flow (or even each packet of every flow in the system) SDN switches just implement the actions/rules that are given to them by the controllers.

SDNs have been proposed for use in IoT systems in the past (*e.g.,* [24]). There also exists work in improving network security using SDNs in particular targeting the DDoS attacks [25], [26] and even network monitoring [27]. There also exists some work on applying security techniques to IoT networks using SDNs (*e.g.,* [28]). None of this work considers the criticality of certain flows in the IoT network. Such flows will require not just isolation guarantees (so that adversaries cannot tamper with these flows/packets) but also improved *resiliency* – if there is an attack of a link/node failure, then these flows must still be delivered to their destinations *while still meeting their end-to-end QoS guarantees*.

Hence, we believe that SDNs can be used to improve the resiliency of critical flows in IoT systems. This can be achieved using a three-pronged approach:

1) *Using global visibility for better management*. The critical flows (*e.g.,* payment/financial information, control commands, *etc.*) will often come with design constraints such as quality of service (QoS) guarantees. One example could be that control commands in industrial internet of things (IIoT) or power systems be delivered to their destinations (often robots, conveyors, generators, *etc.*) within a *predetermined amount of time*. Failure to do so could result in catastrophic failures in such systems. Hence, we intend to use the fact that the SDN controller has detailed knowledge about the state of the system (the amount of data on the network, the link capacities, *etc.*) to *find routes through the network that meet the given QoS guarantees*.
2) *Improved security via isolation*. If malicious actors are able to identify the critical flows in the network then there is a possibility that such attackers could tamper with the packets of such flows. Hence, there is a need to *improve the isolation* for such packets – essentially make sure that even in the presence of attacks (*e.g.,* DoS), these packets get through the network unmolested. The controller might need to carve out special resources (links, queues on switches, *etc.*) specifically for these packets for the lifetime of the flow. The controller will also need to track what other flows are entering the network and ensure that they do not affect the behavior of the critical flows – i.e. they don't impinge on the resources allocated to the latter.
3) *Resiliency against attacks/failures*. If links/switches either get congested (either due to increased activity or even DoS attacks) or fail then there is a need to ensure that the critical flows as not affected. Hence, such flows must *automatically be re-routed* so that they still meet their QoS guarantees. During this process of re-routing, new paths must be calculated and resources allocated, on the fly, as mentioned above.

While there exists initial work on providing end-to-end QoS guarantees [22], this early solution is limited to real-time systems with stringent timing requirements and isolated/fully contained networks. In contrast, network flows in IoT systems may have to traverse across multiple networks and even the Internet. Hence, we need to adapt such solutions to more heterogeneous, open, networks.

One of the challenges to calculating end-to-end routes for critical flows in larger networks (or even across the Internet) is that a single SDN controller may not have visibility into the state of the entire network. Hence, all of the above requirements might be hard to guarantee. One way to get around this problem could be to build upon recent research in *distributed SDN controllers* (*e.g.,* [29]–[31]). Using such solutions, we can then break the problem into smaller problems – *(a)* finding routes, checking for malicious behavior and finding alternate paths (during failures/attacks) in smaller networks and *(b)* *composing* the solutions to solve the end-to-end problem for the critical flows.

V. Scalable Authentication for IoT Devices

While authenticating users has been very well studied and established in the security field, authentication of embedded devices to each other and to web services is still an open challenge. Whether embedded devices should be managed by private or public certificate authority or even certificates are the answer is not clear yet. Specially, in the case of smart homes and everyday IoT devices that lack secure elements, have limited user interface, and suffer from hardware/ software limitations, designing a secure, scalable, and user-friendly (or better yet user-free) authentication system is challenging.

The paradigm of something "you know, you have, or you are" has been practiced for the user authentication, however, the situation is quite different for the embedded devices. In many cases, there is no user interface on a device to enter a password (something you know) or measure biometrics (something you are). Similarly, there might not be a port to input a USB key (something you have). The idea of storing a secret key on the device (as a measure of what you know) is not always reliable since there is no secure element (a trusted environment) present in most cases.

Compromising a user identity is considered a threat in security, however, physical threats are less often considered for users and are out of the scope of the threat model. The same principle does not hold for embedded devices. A phone can be stolen, reverse engineered (SW and HW), probed for keys using memory attacks. If a device is "borrowed" for a few hours, an attacker could infer data about your device via power analysis. Such physical threats make embedded devices more prone to compromise, and impersonation attacks, and as a result, it makes authentication more challenging.

Another challenge with devices, for example in a smart home setting or a smart city scenario, is that they are part of a high-churn network. As the user goes from home to work, the devices around his phone will change. As a result, any authentication scheme that relies on a particular device (e.g. home router) as a root of trust would not work. Therefore, any distributed authentication protocol for smart devices must account for changes in the network and for new devices being added to the network. For example, a key sharing scheme with high tolerance for small threshold would be quite suitable. On the other hand, identity management is not a major problem as the devices are not added or removed that often.

Accounting for all these challenges, we would argue that any authentication method designed for embedded systems in the heterogeneous IoT settings could be secure and effective if it achieves the following properties: 1) scalable to the number of devices, and type of devices; 2) decentralized so that devices are not tied to one specific device for authentication; 3) risk-aware and risk-tolerant so that one compromised device would not jeopardize the whole system. To achieve these goals, one avenue to explore is the multi-device nature of IoT settings, where secret sharing schemes, and secure multi-party computation techniques could strengthen the system. Moreover, multi-device algorithms bring with themselves redundancy and avoid single point of failure.

VI. Remote Attestation for IoT devices

Establishing trust in a world full of IoT devices is extremely critical, as attacks targeting these devices are becoming more prevalent. Remote attestation is a security service that allows a remote, potentially infected, device (denoted by prover) to send an authentic status report to a trusted party, called verifier, to demonstrate that it is in a known and, thus trustworthy, state. Conventional attestation schemes [32]–[35] rely on the existence of a secure co-processor (e.g., a TPM [36]) to ensure the authenticity of the attestation report. Such schemes are most suitable for advanced computing platforms, such as smartphones, personal computers, and servers. However, they are not applicable to embedded devices that do not support such complex and expensive hardware setup.

Another approach would be to apply software-based attestation that requires neither secure hardware nor cryptographic secrets [37]–[40]. The security of software-based attestation is rather based on the limited computational power of the prover and a strict estimation of the time required to generate a valid attestation report. Unfortunately, software-based attestation methods rely on strong assumptions, such as the adversary being passive while the attestation protocol is executed and optimality of the attestation algorithm and its implementation. Such assumptions are hard to achieve in many realistic settings [41]. Consequently, approaches for practical remote attestation that are based a lightweight hardware trust anchor were devised.

Recent research efforts proposed multiple lightweight hardware security architectures for low-end embedded devices based on minimal hardware features such as a read-only memory (ROM) and a simple memory protection unit (MPU), e.g., SMART [42] and TrustLite [43]. These architectures present the trust anchor that allows secure remote attestation for many embedded devices.

However, conventional attestation is static in general, it provides the verifier with an authentic measurement (typically, a hash) of the software binaries, i.e., it provides a proof that the correct software was loaded on the prover for execution. However, it does not capture runtime attacks that hijack the program's control-flow (e.g., return-oriented programming – ROP [44]). Recently progress has been made in tackling the static character of conventional remote attestation through the development of control-flow attestation [45]–[47]. Control-flow attestation allows a prover to attest the exact control-flow path of an executed program to a remote verifier. Thus, enabling the detection of runtime attacks that lead to an unintended program execution, e.g., non-control-data attacks.

A second problem is that remote attestation solutions are geared for a single-prover setting, i.e., do not securely scale to large swarms of embeded devices such as autonomous systems (e.g., drones, vehicles). Several recent efforts [48]–[51] yielded attestation methods for swarms of devices providing secure swarm attestation. Swarm attestation allows attesting a million-device swarm in order of seconds [48], provides resiliency against denial of service (DoS) attacks [49], and is capable of detecting physical attacks that are more relevant in device swarms [50], [51].

In the following we will briefly consider each of these aspects. In particular, we describe lightweight security architecture for embedded devices, and explain control-flow attestation as well as swarm attestation.

A. Lightweight Security Architectures

Diversity of embedded devices presents a challenge for designing hardware security solutions for such devices. Such hardware security aims at establishing trust in embedded devices and guarantee that a device is in a known and trustworthy state by enabling secure remote attestation.

SMART [42] is a security architecture for low-end embedded devices which allows establishing a dynamic root of trust on a remote device. SMART enables secure remote attestation based on two simple components: (1) a read-only memory (ROM), which stores program code used for attestation and the attestation key, and (2) a simple memory protection unit (MPU), which controls access to ROM where the key is stored. Both ROM and MPU are considered minimal hardware requirement for remote attestation and they are easy and inexpensive to realize. The concept of SMART is that program code in ROM cannot be altered by any software running on the device, which ensures integrity of attestation code. On the other hand, MPU grants access to the key only to ROM code by checking whether the program counter is within the address space of ROM whenever the key is accessed.

Based on Intel's Siskiyou Peak research platform, TrustLite [43] enables isolation of code executing on embedded devices. Isolated code chunks on TrustLite are called trustlets. The isolation of trustlets is ensured by the *Execution-Aware Memory Protection Unit* (EA-MPU), which can be seen as a generalization of SMART's MPU. The main difference is that memory access control rules of EA-MPU in TrustLite can be programmed as required by trustlets in contrast to memory access control rules of SMART's MPU which are static. Authenticity and confidentiality of both code and data of trustlets are ensured via secure boot.

B. Towards Runtime Attestation

Static attestation typically computes the hash of binary code to be attested, thus only ensuring the integrity of binaries and not the runtime behavior of the underlying code.

C-FLAT [45] presents the first step towards runtime attestation on embedded devices. In addition to static attestation, C-FLAT captures the runtime behavior of a program by measuring its control-flow during execution. It allows the prover to efficiently compute an aggregated authenticator of the program's control-flow and report it to the verifier, which can then determine whether application's control-flow has been compromised. LO-FAT [46] aims at improving the performance of C-FLAT on the prover side by leveraging hardware assistance to track control-flow events and performs hash calculations parallel to program execution. LO-FAT supports control-flow attestation of legacy code since binary instrumentation is not required. Finally, in addition to detecting control-flow attacks, ATRIUM [47] is capable of detecting physical time-of-check-time-of-use (TOCTOU) on attestation that allow a prover to execute a different code than that reported to the verifier. ATRIUM [47] detects ROP and TOCTOU attacks by performing both static and control-flow attestation concurrently at runtime.

C. Swarm Attestation

Based on lightweight hardware security architecture for low-end embedded devices such as SMART [42] and TrustLite [43], swarm attestation enables secure and efficient attestation of large swarms of embedded devices.

SEDA [48] is the first efficient attestation protocol for large swarms. It distribute attestation burden across the network, allowing neighbors to attest each other and aggregate the attestation results. To attest a swarm, the verifier chooses a device as initiator and sends it a random challenge, which is then flooded in the network forming a spanning tree rooted at the initiator. Each device generates an attestation report and sends it to its parent for verification. This process ends when the initiator forwards the aggregated attestation report of the entire swarm to the verifier. SANA [49] presents a novel cryptographic primitive, called Optimistic Aggregate Signatures (OAS). Based on OAS, SANA proposes an attestation protocol for swarms of embedded devices, which is more resilient to physical and denial of service (DoS) attacks than SEDA. In order to mitigate DoS attacks on the network, SANA requires the verifier to possess a secure attestation token. LISA [52] presents two concrete implementations of SEDA. It aims at realizing swarm attestation in real networks and investigating its applicability. LISA focuses particularly on the construction of the spanning tree for attestation.

DARPA [50] aims at detecting both software manipulations and physical attacks in swarms of embedded devices. It extends swarm attestation with an absence detection protocol based on periodic network-wide heartbeats. Based on absence detection, it enables devices in the swarm to detect physical attacks on their peers and report it to the verifier. SCAPI [53] overcomes the limitations of DARPA by devising an efficient mean of detecting physical attacks, which is based the frequent update of a session key that is shared between devices and used for swarm attestation. Finally, US-AID [51] combines continuous attestation of neighbors and periodic local heartbeats with a key exchange mechanism to detect both software and physical attacks in autonomous systems with minimal cost.

D. Challenges

Establishing trust in remote IoT devices is becoming increasing crucial. A lot of research has been done on secure remote attestation. Nevertheless, several challenges in emerging IoT scenarios remain open requiring further research. Examples include securing remote attestation under stronger adversary models and the detection of more sophisticated runtime attacks such as data-oriented programming (DOP) attacks. Another challenge facing attestation is its applicability to autonomous networks where extremely mobile embedded devices need to act as both provers and verifiers.

VII. Threat, Risk, and Maturity Assessment Frameworks for the Internet of Things

An increasing number of everyday devices are being connected to the Internet, creating a network of physical devices, home appliances, and other items embedded with unique sets of electronics, software, sensors, and actuators. This rapid growth in the Internet of Things (IoT) creates opportunities for more direct integration of the physical world into computer-based systems but also introduces a plethora of risks. IoT devices, whether they be part of smart transportation systems, thermostats that adapt to daily lifestyles, or medical devices that can monitor a patient in real time, are not always built with security in mind, leaving them extremely vulnerable to attacks.

A. Threats and Risks in IoT Environments

An example of a pervasive IoT attack was the Mirai botnet, where an attacker gained unauthorized access to numerous IoT devices including IP cameras and older routers. These infected devices were used to make much of the Internet unavailable by overwhelming Dyn, a domain name system (DNS) provider. The malicious code took advantage of devices running out-of-date versions of the Linux kernel, relying on the fact that most users do not change the default usernames and passwords on their devices [54].

Due to the ease with which large numbers of IoT devices may be compromised and their direct relationship to physical processes, this attack among many others motivates the need for organizations to assess threats and risks in IoT environments, regardless of the role organizations play within an IoT ecosystem. Possible roles include IoT consumers, IoT producers, and/or platform operators that provide software that customers connect their devices to. Each role comes with its own set of security and privacy risks, affecting the actions or controls an organization takes to mitigate those risks.

B. Current Threat Modeling and Risk Assessment Frameworks

Risk is a measure of the extent to which an entity is threatened by a circumstance or event, and it is a function of the likelihood of the event and the adverse impact caused by the event. Risk assessment is commonly seen as a means to identify, estimate, and prioritize risk to national, organizational, and individual operations and assets. The traditional approach to risk assessment has been to identify relevant threats to organizations, internal and external vulnerabilities to organizations, the impact or harm to organizations that may occur if vulnerabilities are exploited, and the likelihood that harm will occur due to a successful attack [55]. After the identified risks have been prioritized, appropriate and effective actions and controls are chosen that mitigate those risks. Several popular and well-regarded approaches to threat modeling and risk assessment include STRIDE [56], PASTA [57], Trike [58], NIST SP800-30 [55], ISO/IEC 27005 [59], IEC/FDIS 31010 [60], CRAMM [61], FRAP [62], COBRA [63], CORAS [64], and OCTAVE [65].

Since existing threat modeling and risk assessment methodologies were established prior to the development of the IoT, they do not always cater to the complexity and pervasiveness of IoT ecosystems and the new risks they pose. These inadequacies present themselves in a few ways. Firstly, these approaches are currently based on periodic assessment, assuming that systems do not change significantly in a short period of time. However, this assumption does not hold well in the IoT where there is much variability in system scale, dynamics, and coupling. Secondly, current risk assessment approaches require a large amount of knowledge about organizational assets, threats, likelihoods, and impacts which is difficult to obtain in IoT ecosystems due to limited system knowledge and history of attacks. Thirdly, these approaches fail to thoroughly consider the relationships, processes, and couplings between IoT devices, focusing instead on individual assets, devices, and communication platforms. Lastly, current risk assessments simply view organizational assets as things of value as opposed to platforms from which an attack may be launched as in the case of the Mirai botnet [3].

C. IoT Threat Modeling and Risk Assessment

Due to these inadequate approaches, it is important to establish new threat modeling and risk assessment frameworks that analyze security and privacy risks unique to IoT ecosystems. For instance, to design and leverage an IoT attack taxonomy as the basis for threat modeling and risk assessment. The IoT attack taxonomy could be comprised of a comprehensive list of atomic attacks – where any IoT incident is composed of a series of atomic attacks. Any atomic attack can be broken down into four parts, each of which is associated with a particular dimension of the IoT attack taxonomy: an attacker with a specific set of 1) *assets* carries out a particular 2) *action* to exploit one or more 3) *vulnerabilities*, compromising a particular set of 4) *properties*.

By using an IoT attack taxonomy as the foundation for threat modeling and risk assessment, we can comprehensively characterize and break down the entire known IoT attack space. Furthermore, the IoT attack taxonomy can be consistently updated as IoT ecosystems evolve, allowing threat modeling and risk assessment to be modular and responsive to changes in IoT technology while at the same time retaining a similar high-level approach and methodology of existing threat modeling and risk assessment frameworks. In addition, such an approach considers all the relationships and couplings between elements of the IoT taxonomy to allow for a comprehensive vulnerability and device-centered evaluation of threats and risks. Lastly, this approach to risk and maturity assessments is quantitative in nature, providing a more robust and less subjective analysis of risk by using historical data and expert opinion as inputs to detailed quantitative risk functions. Furthermore, a quantitative approach to maturity assessment allows consultants to conduct sensitivity analyses, giving them a more detailed view and understanding of the controls that need to be implemented to maximally decrease an organization's residual risk.

By establishing and designing new threat modeling, risk, and maturity assessment frameworks for IoT ecosystems, organizations will be able to receive accurate information about which actions and controls to implement as safeguards or countermeasures to prevent and protect against attacks. This information allows organizations to make informed decisions as to where regulatory, financial, and time investments should be made, the result of which may protect consumers and producers of IoT devices from the plethora of attacks that have the potential to cause catastrophic physical harm and damage.

VIII. Conclusion

Security and privacy of IoT devices and services is already of significant concern. Designers of such systems need to bake in security from design time. An ever increasing rush to release new products into the market leaves multiple security holes in IoT devices. This is only expected to increase in the near future. An understanding of some of the requirements and trends in IoT security should aid designers and users to develop/use such systems in a better fashion.

Acknowledgements

The material in this paper is based upon work supported in part by the U.S. Department of Energy (DoE) awards DE-OE0000679 and DE-OE0000780 and National Science Foundation (NSF) grants NSF-CPS-1544901 and NSF SaTC 1718952; NSF CNS-1646317 and the U.S. Department of Homeland Security grant 2017-ST-062-000003. The researchers gratefully acknowledge the support of the Risk and Regulatory Services Innovation Center at Carnegie Mellon University sponsored by PwC. Any findings, opinions, recommendations or conclusions expressed in the paper are those of the authors and do not necessarily reflect the views of sponsors.

References

[1] S. Kolozali, M. Bermudez-Edo, D. Puschmann, F. Ganz, and P. Barnaghi, "A knowledge-based approach for real-time iot data stream annotation and processing," in *Internet of Things (iThings), 2014 IEEE International Conference on, and Green Computing and Communications (GreenCom), IEEE and Cyber, Physical and Social Computing (CPSCom), IEEE*. IEEE, 2014, pp. 215–222.

[2] M. M. Hossain, M. Fotouhi, and R. Hasan, "Towards an analysis of security issues, challenges, and open problems in the internet of things," in *Services (SERVICES), 2015 IEEE World Congress on*. IEEE, 2015, pp. 21–28.

[3] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "Ddos in the iot: Mirai and other botnets," *Computer*, vol. 50, no. 7, pp. 80–84, 2017.

[4] E. Bertino and N. Islam, "Botnets and internet of things security," *Computer*, no. 2, pp. 76–79, 2017.

[5] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1125–1142, 2017.

[6] S. Huh, S. Cho, and S. Kim, "Managing iot devices using blockchain platform," in *Advanced Communication Technology (ICACT), 2017 19th International Conference on*. IEEE, 2017, pp. 464–467.

[7] G. Bloom, B. Alsulami, E. Nwafor, and I. C. Bertolotti, "Design patterns for the industrial Internet of Things," in *2018 14th IEEE International Workshop on Factory Communication Systems (WFCS)*, Jun. 2018, pp. 1–10.

[8] "NIST Lightweight Cryptography." [Online]. Available: https://csrc.nist.gov/Projects/Lightweight-Cryptography

[9] B. Ao, Y. Wang, L. Yu, R. R. Brooks, and S. S. Iyengar, "On Precision Bound of Distributed Fault-Tolerant Sensor Fusion Algorithms," *ACM Comput. Surv.*, vol. 49, no. 1, pp. 5:1–5:23, May 2016. [Online]. Available: http://doi.acm.org/10.1145/2898984

[10] E. Bertino, "Data Trustworthiness–Approaches and Research Challenges," in *Data Privacy Management, Autonomous Spontaneous Security, and Security Assurance*, ser. Lecture Notes in Computer Science. Springer, Cham, Sep. 2014, pp. 17–25. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-17016-9_2

[11] E. Nwafor, A. Campbell, and G. Bloom, "Anomaly-based Intrusion Detection of IoT Device Sensor Data using Provenance Graphs," in *1st International Workshop on Security and Privacy for the Internet-of-Things*, Orlando, Florida, USA, Apr. 2018.

[12] E. Nwafor, D. Hill, A. Campbell, and G. Bloom, "Towards a Provenance Aware Framework for Internet of Things Devices," in *Proceedings of the 14th International Conference on Ubiquitous Intelligence and Computing*, ser. UIC '17. San Fransisco, CA, USA: IEEE Computer Society, 2017.

[13] Q. Wang, W. U. Hassan, A. J. Bates, and C. Gunter, "Fear and Logging in the Internet of Things," in *Proceedings of the NDSS Symposium*, 2017.

[14] C. Wang, S. R. Hussain, and E. Bertino, "Dictionary Based Secure Provenance Compression for Wireless Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 405–418, Feb. 2016.

[15] Y. Xie, K.-K. Muniswamy-Reddy, D. D. E. Long, A. Amer, D. Feng, and Z. Tan, "Compressing Provenance Graphs," in *3rd USENIX Workshop on the Theory and Practice of Provenance*, Jun. 2011.

[16] A. Bates, K. R. B. Butler, and T. Moyer, "Take Only What You Need: Leveraging Mandatory Access Control Policy to Reduce Provenance Storage Costs," in *Proceedings of the 7th USENIX Conference on Theory and Practice of Provenance*, ser. TaPP'15. Berkeley, CA, USA: USENIX Association, 2015, pp. 7–7. [Online]. Available: http://dl.acm.org/citation.cfm?id=2814579.2814586

[17] R. Hasan, R. Sion, and M. Winslett, "Introducing Secure Provenance: Problems and Challenges," in *Proceedings of the 2007 ACM Workshop on Storage Security and Survivability*, ser. StorageSS '07. New York, NY, USA: ACM, 2007, pp. 13–18. [Online]. Available: http://doi.acm.org/10.1145/1314313.1314318

[18] C. Zhang and R. Green, "Communication security in internet of thing: preventive measure and avoid ddos attack over iot network," in *Proceedings of the 18th Symposium on Communications & Networking*. Society for Computer Simulation International, 2015, pp. 8–15.

[19] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[20] R. Jain and S. Paul, "Network virtualization and software defined networking for cloud computing: a survey," *IEEE Communications Magazine*, vol. 51, no. 11, pp. 24–31, 2013.

[21] D. Levin, M. Canini, S. Schmid, and A. Feldmann, "Incremental sdn deployment in enterprise networks," in *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4. ACM, 2013, pp. 473–474.

[22] R. Kumar, M. Hasan, S. Padhy, K. Evchenko, L. Piramanayagam, S. Mohan, and R. B. Bobba, "End-to-end network delay guarantees for real-time systems using sdn," in *Real-Time Systems Symposium (RTSS), 2017 IEEE*. IEEE, 2017, pp. 231–242.

[23] "List of sdn controller software," https://en.wikipedia.org/wiki/List_of_SDN_controller_software, [Wikipedia, Online, accessed Juy 30 2018.].

[24] Z. Qin, G. Denker, C. Giannelli, P. Bellavista, and N. Venkatasubramanian, "A software defined networking architecture for the internet-of-things," in *Network Operations and Management Symposium (NOMS), 2014 IEEE*. IEEE, 2014, pp. 1–9.

[25] Q. Yan, F. R. Yu, Q. Gong, and J. Li, "Software-defined networking (sdn) and distributed denial of service (ddos) attacks in cloud computing environments: A survey, some research issues, and challenges," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 602–622, 2016.

[26] S. Lim, J. Ha, H. Kim, Y. Kim, and S. Yang, "A sdn-oriented ddos blocking scheme for botnet-based attacks," in *Ubiquitous and Future Networks (ICUFN), 2014 Sixth International Conf on*. IEEE, 2014, pp. 63–68.

[27] A. Zaalouk, R. Khondoker, R. Marx, and K. Bayarou, "Orchsec: An orchestrator-based architecture for enhancing network-security using network monitoring and sdn control functions," in *Network Operations and Management Symposium (NOMS), 2014 IEEE*. IEEE, 2014, pp. 1–9.

[28] F. Olivier, G. Carlos, and N. Florent, "New security architecture for iot network," *Procedia Computer Science*, vol. 52, pp. 1028–1033, 2015.

[29] K. Phemius, M. Bouet, and J. Leguay, "Disco: Distributed multi-domain sdn controllers," in *Network Operations and Management Symposium (NOMS), 2014 IEEE*. IEEE, 2014, pp. 1–4.

[30] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow *et al.*, "Onos: towards an open, distributed sdn os," in *Proceedings of the third workshop on Hot topics in software defined networking*. ACM, 2014, pp. 1–6.

[31] A. Dixit, F. Hao, S. Mukherjee, T. Lakshman, and R. Kompella, "Towards an elastic distributed sdn controller," in *ACM SIGCOMM computer communication review*, vol. 43, no. 4. ACM, 2013, pp. 7–12.

[32] X. Kovah, C. Kallenberg, C. Weathers, A. Herzog, M. Albin, and J. Butterworth, "New results for timing-based attestation," in *IEEE Symposium on Security and Privacy*, 2012, pp. 239–253.

[33] D. Schellekens, B. Wyseur, and B. Preneel, "Remote attestation on legacy operating systems with trusted platform modules," *Science of Computer Programming*, vol. 74, no. 1, pp. 13–22, 2008.

[34] R. Sailer, X. Zhang, T. Jaeger, and L. Van Doorn, "Design and implementation of a TCG-based integrity measurement architecture." in *Proceedings of the 13th USENIX Security Symposium*, 2004, pp. 223–238.

[35] J. M. McCune, Y. Li, N. Qu, Z. Zhou, A. Datta, V. Gligor, and A. Perrig, "TrustVisor: Efficient TCB reduction and attestation," in *Proceedings of the 2010 IEEE Symposium on Security & Privacy*, ser. S&P '10, 2010, pp. 143–158.

[36] Trusted Computing Group (TCG), "Website," http://www.trustedcomputinggroup.org, 2015.

[37] A. Seshadri, M. Luk, A. Perrig, L. van Doorn, and P. Khosla, "SCUBA: Secure code update by attestation in sensor networks," in *ACM Workshop on Wireless Security*, 2006.

[38] A. Seshadri, M. Luk, and A. Perrig, "SAKE: Software attestation for key establishment in sensor networks," in *Distributed Computing in Sensor Systems*, 2008.

[39] R. Gardner, S. Garera, and A. Rubin, "Detecting code alteration by creating a temporary memory bottleneck," *IEEE Transactions on Information Forensics and Security*, 2009.

[40] Y. Li, J. M. McCune, and A. Perrig, "VIPER: Verifying the integrity of peripherals' firmware," in *ACM Conference on Computer and Communications Security*, 2011.

[41] F. Armknecht, A.-R. Sadeghi, S. Schulz, and C. Wachsmann, "A security framework for the analysis and design of software attestation," in *ACM Conference on Computer and Communications Security*, 2013.

[42] K. Eldefrawy, G. Tsudik, A. Francillon, and D. Perito, "SMART: Secure and Minimal Architecture for (Establishing a Dynamic) Root of Trust," in *Network and Distributed System Security Symposium*, 2012.

[43] P. Koeberl, S. Schulz, A.-R. Sadeghi, and V. Varadharajan, "TrustLite: A Security Architecture for Tiny Embedded Devices," in *European Conference on Computer Systems*, 2014.

[44] H. Shacham, "The geometry of innocent flesh on the bone: Return-into-libc without function calls (on the x86)," in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, ser. CCS '07. New York, NY, USA: ACM, 2007, pp. 552–561. [Online]. Available: http://doi.acm.org/10.1145/1315245.1315313

[45] T. Abera, N. Asokan, L. Davi, J.-E. Ekberg, T. Nyman, A. Paverd, A.-R. Sadeghi, and G. Tsudik, "C-flat: Control-flow attestation for embedded systems software," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16, 2016.

[46] G. Dessouky, S. Zeitouni, T. Nyman, A. Paverd, L. Davi, P. Koeberl, N. Asokan, and A.-R. Sadeghi, "Lo-fat: Low-overhead control flow attestation in hardware," in *54th Design Automation Conference (DAC'17)*, Jun. 2017.

[47] S. Zeitouni, G. Dessouky, O. Arias, D. Sullivan, A. Ibrahim, Y. Jin, and A.-R. Sadeghi, "Atrium: Runtime attestation resilient under memory attacks," in *2017 International Conference On Computer Aided Design (ICCAD'17)*, Nov. 2017.

[48] N. Asokan, F. Brasser, A. Ibrahim, A.-R. Sadeghi, M. Schunter, G. Tsudik, and C. Wachsmann, "SEDA: Scalable Embedded Device Attestation," in *Proceedings of the 22nd ACM Conference on Computer & Communications Security*, ser. CCS '15, 2015, pp. 964–975.

[49] M. Ambrosin, M. Conti, A. Ibrahim, G. Neven, A.-R. Sadeghi, and M. Schunter, "SANA: Secure and Scalable Aggregate Network Attestation," in *Proceedings of the 23rd ACM Conference on Computer & Communications Security*, ser. CCS '16, 2016.

[50] A. Ibrahim, A.-R. Sadeghi, and G. Tsudik, "DARPA: Device Attestation Resilient against Physical Attacks," in *Proceedings of the 9th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, ser. WiSec '16, 2016.

[51] ——, "Us-aid: Unattended scalable attestation of iot devices," in *Proceedings of the 37th IEEE International Symposium on Reliable Distributed Systems*, ser. SRDS '18, 2018.

[52] X. Carpent, K. ElDefrawy, N. Rattanavipanon, and G. Tsudik, "Lightweight swarm attestation: A tale of two lisa-s," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ser. ASIA CCS '17, 2017.

[53] F. Kohnhäuser, N. Büscher, S. Gabmeyer, and S. Katzenbeisser, "Scapi: A scalable attestation protocol to detect software and physical attacks," in *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, ser. WiSec '17, 2017.

[54] M. A. et al., "Understanding the mirai botnet," in *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, 2017, pp. 1093–1110.

[55] "Guide for conducting risk assessments," NIST, Tech. Rep. SP-800-30 - Revision 1, September 2012.

[56] The stride threat model. [Online]. Available: https://msdn.microsoft.com/en-us/library/ee823878(v=cs.20).aspx

[57] T. Ucedavélez and M. M. Morana, *Risk Centric Threat Modeling: Process for Attack Simulation and Threat Analysis*. John Wiley & Sons, 2015.

[58] B. L. Eleanor Saitta and M. Eddington, "Trike v.1 methodology document," July 2015.

[59] "Information technology - security techniques - information security risk management," ISO, Tech. Rep. ISO/IEC 27005:2011, June 2011.

[60] "Risk management - risk assessment techniques," ISO, Tech. Rep. IEC/FDIS 31010:2009, November 2009.

[61] *CRAMM user guide, risk analysis and management method*, United Kingdom CCTA, 2001.

[62] T. R. Peltier, *Information Security Risk Analysis*. Auerbach Publications, 2010.

[63] Consultative, objective and bi-functional risk analysis (cobra). [Online]. Available: http://www.security-risk-analysis.com/introcob.htm

[64] F. den Braber et al., "Model-based security analysis in seven steps - a guided tour to the coras method," *BT Technology Journal*, vol. 25, no. 1, pp. 101–117, 2007.

[65] C. J. Alberts and A. J. Dorofee, *Managing Information Security Risks: The OCTAVE Approach*. Addison-Wesley Professional, 2002.