TAME:

Trusted and Assured MicroElectronics

Working Groups Report

December 2019

The TAME Steering Committee would like to express their deepest gratitude to all volunteers for their hard work in the three committees over the past 18 months to develop these reports. The volunteers have been acknowledged at the beginning of each report.

Mark Tehranipoor
University of Florida

Waleed Khalil
Ohio State University

Matt Casto
OSD

Yousef Iskander
Cisco

Brian Dupaix
Air Force Research Laboratory (AFRL)

Ro Cammarota
Intel

Brian Cohen
Institute of Defense Analysis (IDA)

Table of Contents

Chapter 1: Hardware Assurance and Weakness Collaboration and Sharing (HAWCS)

Contributors:

- Jeremy Bellay (Lead) Battelle
- Domenic Forte (Scribe) University of Florida
- Bob Martin (Advisor) MITRE
- Jon Boyens (Advisor) NIST
- Mike Borza Synopsys
- Ron Perez Intel
- Yatin Hoskote ARM
- Sandip Ray University of Florida
- Ioannis Savidis Drexel University
- Fareena Saqib UNC Charlotte
- Whitney Batchelor Graf Research
- Lisa McIlrath University of Florida
- Khalil Maalouf Riverside Research
- Mark Temmen U.S. Army
- Marion Williams NIMBIS Services
- Vivian Kammler Sandia National Labs
- Melanie Berg NASA
- Vikram Rao Aerospace
- Wayne Reed Honeywell

1.1 Problem Statement

The field of computer and communications security has traditionally viewed hardware as an immutable root-of-trust. However, the last decade of hardware security research has led to the discovery of hardware-oriented weaknesses that can be exploited remotely, in the supply chain, through physical access, or through any combination of the three.

In particular, 2018 was a catastrophic year for many of the top processor vendors. Researchers uncovered multiple CPU hardware vulnerabilities – Meltdown, Spectre, Foreshadow, and even a new variant of Rowhammer to name a few [1]–[4] – and showed how they could be used to gain access to forbidden data and system privileges, even in trusted execution environments. Most exploits were 'baked into' the processor architectures through performance enhancing features such as branch prediction, speculative execution, and translation lookaside buffer. By and large, these vulnerabilities were complicated to mitigate and incurred significant performance degradation as well as other unintended side effects [5]. Those that could not be addressed by patches shall require system replacement.

In parallel, concerns are mounting about firmware and hardware-level attacks occurring in the supply chain. Semiconductor globalization has benefitted the industry by shortening time-to-market, lowering non-recurring engineering costs, and managing the escalating investment

required to build state-of-the-art integrated circuit (IC) manufacturing facilities and the years it takes them to achieve a high-yield level of production, but not without creating its share of new problems. In 2008, an IEEE Spectrum article [6] underscored the risks of electronic backdoors (i.e., hardware Trojans) inserted by contract foundries and third party intellectual property (IP) vendors. Further, supply chain interdiction, where items such as routers, flash drives, and other electronic devices are compromised while in transit to their final destination [7], has also been reported. Recently, there has been an increase in public awareness around the potential of deliberate backdoors in hardware including the (heavily disputed) "Big Hack" article published by Bloomberg [8], and the controversy surrounding Huawei's primary role in 5G hardware. Last but not least, counterfeit electronics are a longstanding threat that have only become worse with the increase in global e-waste [9], long-lived systems, and chip obsolescence[10].

Beyond simply offering an alternative attack surface beyond software, hardware has other attractions to attackers that will make it an increasingly common target. Since hardware is the foundation of computing and operates at the lowest abstraction layer, common software mitigations such as system updates and patches are inadequate. Moreover, while hardware attacks require physical access and have long been considered too expensive, the landscape is starting to change in the attacker's favor. As part of the emerging internet of things (IoT), the number of "smart things" is soon expected to dwarf mobile computing devices[11]. IoT products are often deployed in hostile environments where they can operate for years with minimal intervention. Those still practicing lax security practices (hard-coded cryptographic keys, unprotected JTAG, etc.) are in for an unpleasant surprise. In addition, improvements in the capabilities of failure analysis tools, especially semi-invasive ones, coupled with lower barriers to their access make on-chip assets, secrets, and IP more vulnerable to fault injection, probing, and reverse engineering than ever before[12].

We follow MITRE's definitions of vulnerabilities and weaknesses [1] : A vulnerability is an error in a particular information system that may allow exploitation of that system. Weaknesses, on the other hand, are the circumstances in design and manufacture that allow for the existence of a vulnerability. Extending MITRE's definitions to hardware, we define a hardware vulnerability/ weakness as one existing in the hardware or firmware of a system. Perhaps a functional definition is that the vulnerability or weakness can be potentially mitigated but not removed by a software changes.

Hardware is also evolving rapidly, and thus creating fundamentally new attack surfaces, many of which will probably never be entirely secured. The pressure of increasing performance and compactness in the face of a presumed physical limit on node size has led to new creative approaches that have increased performance but also opened new attack surfaces; both Rowhammer and Spectre/Meltdown can be seen to represent not individual vulnerabilities but entire attack surfaces. Several new technologies are promised in the next 10-20 years including 5nm and below node sizes, neuromorphic computing, Carbon-Nanotube FET, and quantum computing bringing with them unforeseen weaknesses.

Left unaddressed, vulnerabilities expose critical systems, their information, and the people who rely on them. But in order to be fixed, they must first be identified. The rise of viruses, malware, and remote attacks in the 80's, 90's, and early 2000's led to efforts aimed at supporting and

coordinating vulnerability education, discovery, advisories, and alleviation. These included a common language and taxonomies to describe software and software-related weaknesses [13], attack patterns [14], and vulnerabilities [15], [16], avenues for responsible disclosure of vulnerabilities and exposures [16], [17], entities to coordinate responses between researchers, vendors, and deployers, and approaches to prioritize mitigations [18].

Unfortunately, the unique challenges faced by semiconductor companies make it impractical to extend the above frameworks and management techniques in their entirety directly to hardware because hardware vulnerabilities differ in nature to their software counterparts. Current software vulnerability disclosure policies advise up to 90 days to develop and distribute a patch before public disclosure, but experience shows that mitigation of hardware vulnerabilities requires more time. This is because semiconductor companies need to work with companies across their supply chain to understand the implications of vulnerabilities and possible solutions at multiple levels – from IC to microcode to software. Patches must also be well-tested in order to maintain operation across diverse environments encountered by the chip; otherwise, serious performance and stability issues can arise from the patch. For critical real-time systems, which have already undergone years of certification, this is unacceptable. Sometimes, a costly hardware recall is the only solution. Beyond cost, however, tracking down the hardware assets of affected customers is so cumbersome that many vulnerable systems are likely to remain in the wild.

For these reasons, those who discover hardware vulnerabilities are often discouraged from publicly sharing their findings [19]. A similar attitude of squelching discussion about security vulnerabilities existed in the software arena in the late 90's, supposedly to protect the public interest by withholding critical information from attackers and to safeguard vendor reputation, and threats of lawsuits were prevalent. On the other hand, some consider the discouragement of sharing security research findings to be counterproductive. For software, and even more so hardware, prevention is likely less expensive and far more effective than mitigation. By allowing a community of researchers and practitioners to more openly discuss and share understandings about the problems that make hardware vulnerable, how to avoid, prevent, or recognize them before they make it into hardware implementations, etc., the entire ecosystem benefits. Lessons learned and best practices are needed so that the community can train design engineers and develop robust, tools and practices that can identify the weaknesses leading vulnerabilities and prevent their introduction and reoccurrences. Instead, it is more common today for fragile tools and checking regimes to be developed in-house, leaving external community of design engineers in the dark and destined to repeat the same mistakes in the systems of other vendors.

In summary, hardware weaknesses and vulnerabilities are a growing concern but the most effective way to deal with them is still an open problem. While one can look at past experience in software, there needs to be a candid discussion within the entire hardware community about how to properly balance the tradeoffs of disclosure on the technology ecosystem as well as society.

1.2 Objectives

The Trusted and Assured MicroElectronics (TAME) Forum's "Hardware Vulnerability Database" (HVD) working group was initially created to provide a roadmap for community dialogue and first steps towards investigation of these issues. While the working group's original focus (and therefore name) was on the creation of a HVD, it eventually become clear that the technologies

used to effectively characterize and share information regarding hardware vulnerabilities is inherently valuable to the community as well. Thus, the group also focused on the issues and technologies for effective sharing of information about the identification and prevention of hardware vulnerabilities, and its name was changed to "Hardware Assurance and Weakness Collaboration and Sharing" (HAWCS).

The real and perceived risks associated with hardware vulnerability sharing and disclosure necessitate a collaborative community effort in the development of effective common characterization, identification, and sharing schemes. The TAME HAWCS working group has representation from industry, government, and academia. This diversity allows the group to pursue three primary goals: First, it allows the identification of risks and benefits to the different sets of stakeholders; an oft cited hurdle to sharing hardware vulnerability details has been the perceived risk associated with vulnerability disclosure and the lack of quantifiable reward. Second, each community offers unique insight and perspective essential to the development of a common identification of vulnerabilities and the development of a common characterization of the underlying physical processes. Finally, the diversity of the working group allots for the integration and dissemination of information across the relevant communities; previous efforts have been limited by the difficulties of communication between groups of stakeholders.

The HAWCS working group has decided to address the following issues through its multi-disciplinary and multi-party perspective:

Identification of current vulnerability reporting and assessment structures. There are numerous ongoing efforts related to the characterization, storage, and reporting of hardware vulnerabilities. The most notable is the software security ecosystem that includes notable components such as Common Vulnerabilities and Exposures (CVE), the National Vulnerability Database (NVD), and agreed upon methods of responsible disclosure. However, there are other relevant efforts such as the Society for Automotive Engineers' Test Methods Standard [20], and the Accellera Systems Initiative[21][21][15]. The HAWCS working group has surveyed the current vulnerability sharing efforts and shall summarize their relevance to hardware vulnerabilities in this report.

Survey benefits and risk to stakeholders. The active participation of all three stakeholder groups – government, industry and academia -- is essential for the success of any vulnerability sharing scheme. The working group is leveraging the diversity of its members to identify the risks and benefits associated with a sharing hardware vulnerability information. This will help ensure that future efforts are meaningful to the participants and avoid exposure to unnecessary risks. Importantly, we anticipate that a shared vulnerability characterization scheme will help structure vulnerability comparison and discovery even for parties that are otherwise non-communicative with the rest of the community.

Survey of hardware security topics. There are many topics relevant to hardware security. The TAME HAWCS working group intends to use the breadth the participants' expertise to survey all the security and reliability concerns of each group member. While some concerns may not be of interest to all stakeholders, the underlying mechanism of the vulnerability (and its detection) may be relevant. For example, while industry may consider the risk of hardware Trojans to be relatively

low, tools developed for Trojan detection (e.g., side channel analysis) may offer inexpensive detection of information leakage and counterfeits.

Knowledge structures for vulnerability characterization. Knowledge structures used to characterize vulnerabilities are a key aspect to leveraging knowledge about vulnerabilities across the community. Even when specific vulnerabilities are not shared due to their sensitive nature, controlled characterization languages such as ontologies can be used to describe the mechanism of a class of vulnerabilities at a shareable, higher level of abstraction. Additionally, shared description languages can be used to describe best practices to avoid vulnerabilities. The development of such knowledge structures has been recognized as valuable even without a shared database as it facilitates communication about unshared vulnerabilities and can be used by parties to structure their own vulnerability cataloging efforts. Finally, a well-structured characterization that links vulnerabilities to their underlying mechanism is essential for the use of analytical methods to detect new vulnerabilities and predict their occurrence in design. A goal of the working group is to investigate what knowledge structures are available for the description of hardware vulnerabilities (e.g., CVE, CWE, CAPEC etc.), and how they need to be extended to describe the often physical nature of hardware vulnerabilities, as well as supporting and referencing the necessary data structures.

Integration with ongoing and future projects. In addition to surveying the current and ongoing efforts touching hardware vulnerabilities, the working group is considering how to best integrate and provide value to these ongoing efforts. The working group is already aware of several ongoing efforts including the CVE/NVD ecosystem, Accellera's standards creation efforts, and the efforts by the Society for Automotive Engineers. However, we anticipate the application of working group output to go beyond active collaborators. For example, a government agency may keep a database of vulnerabilities which is too sensitive for even highly controlled sharing between industry partners. However, external organizations may be able to benefit from the standards and ontology used in the highly sensitive database. The working group members will leverage their connections across different stakeholder groups to support these separate initiatives that are important for hardware security even if their sensitivity discourages active participation in a larger community.

Models for long term funding and sustainment. A key aspect of efforts around an HVD and ontology efforts is ensuring that the initial development is followed by indefinite sustainability. A key danger associated with any kind of knowledge structure development is rapid obsolescence. The working group plans to identify interested partners capable of funding early research and development of vulnerability sharing resources. While funding is always a key issue, leveraging the open source community for the maintenance and expansion of HVD resources must also be explored. Finally, we must also identify potential long term sustainment models such as subscriptions, incorporation into current government programs and support via consortium.

This report, which shall discuss the above topics in more detail, was developed in the multiple phases. This outline was prepared, presented, and discussed at the November 2018 TAME meeting in Columbus, Ohio. The initial draft was compiled and presented at the May 2019 TAME meeting at HOST 2019 in McLean, VA. The final draft was compiled for the TAME forum meeting December 2019 in Gainesville, FL. The final report will be published in Hardware and Systems Security (HaSS) Journal.

1.3 HAWCS Development Areas

1.3.1 Risks and Benefits to Stakeholders

1.3.1.1 Industry

Risks:

1. *IP disclosure*: Hardware weakness characterization could benefit from realistic test articles from industry, but industry stakeholders are very protective of their hardware IP and would be unwilling to divulge it, even for perceived public good. Disclosing details of vulnerabilities in a public database could give potential attackers hints for exploiting additional weaknesses that have not yet been discovered. Disclosures may also be less useful without explanation of design details that are proprietary.
2. *Premature vulnerability dissemination*: If a vulnerability becomes public through the HVD before patches or other mitigations have been deployed, the IP/system vendors could incur significant costs and other losses due to negative publicity and legal liabilities from system downtime and theft/corruption of customer data.
3. *Reputational damage*: By putting sensitive information in the public sphere before the details and extent of its impact has been fully analyzed, companies run the risk of being caught in embarrassing scenarios which could result in significant damage to their brand and reputation. For example, incomplete analysis may lead to attributing cause and responsibility incorrectly. Once reputational damage is done, it can be hard to undo.

Benefits:

1. *Standardization:* Use of a common language and taxonomy to communicate about hardware vulnerabilities would reduce the chance of misunderstanding and make engagements much more productive.
2. *Baseline:* Use the database as a baseline for internal security due diligence on all products before release. One could also use the database as the basis for threat modeling in implementation of a secure development lifecycle for products. There is no coverage metric for robustness. The database could be used as a proxy for a coverage metric.
3. *Collaboration:* A mechanism that facilitates sharing of information confidentially between stakeholders would enable responses to newly discovered vulnerabilities to be faster and more coordinated.
4. *Supply chain robustness:* Could potentially use certification against the database as a certificate of trust between suppliers and procurement departments to build a more robust supply chain.
5. *Tool development:* Provide a target database for development of electronic design automation (EDA) tools for finding vulnerabilities and for validating effectiveness of mitigations. Tools would be especially useful in verifying effectiveness of mitigations.
6. *Education and training:* Database can be used in universities for training new grads, as well as for training security engineers in industry.

7. *Severity metrics:* The database could be enhanced with additional information like severity metrics to help prioritize vulnerabilities and modulate the resources assigned to combat them. For example, a common understanding of the severity metrics would help IP providers and IP integrators to level set expectations on mitigation responsibilities.
8. *Mitigations:* The database could also include mitigations where possible, without divulging confidential information, to help the community respond more rapidly to known vulnerabilities.

1.3.1.2 Government

The risks and benefits of sharing technologies are similar for government and for industry. The major risks being the unintentional disclosure of IP and the weaponization of disclosed vulnerabilities. In addition, government is burdened with legacy devices that may contain hard to mitigate vulnerabilities that have been effectively dealt with in private industry. Additionally, other nation states are willing to spend the resources to develop hard to reach exploits to access government targets. On the other hand, the government has many siloed databases of known vulnerabilities and programs to investigate potential weaknesses. Public sharing technologies can aid these otherwise isolated efforts by providing external fodder and internal structure

Risks:

1. Exposure of vulnerabilities in legacy devices – Government maintains some systems long after their components have exited the commercial market. Thus even "obsolete" shared vulnerabilities may still pose a risk to government systems.
2. Exploitation of sophisticated vulnerabilities – There are many vulnerabilities that take high effort and expertise to exploit, which make them less tempting targets for monetarily motivated crimes. However, adversary nation states have the expertise and resources to exploit these vulnerabilities so as to subvert specific otherwise hardened targets.

Benefits:

1. Standardization in siloed data centers -- The development of vulnerability description and sharing technologies will benefit stakeholders within government agencies that possess unshareable vulnerability databases.
2. Defense strategies at the level of weaknesses – Government specific electronics are extensively red-teamed and databases of even abstract weaknesses can help red teams design realistic testing strategies.

1.3.1.3 Academia

Risks:

Academic researchers are bound to uphold their university mission statements which often involve "promoting the common good". However, the dual nature of security research makes this unwieldy. On the one hand, the public has the right to know about serious issues in the products they buy. Further, in the hands of vendors, this information is almost guaranteed to result in better products. On the other hand, the same information can be exploited by criminals and unethical hackers. Researchers are often left with two choices. Avoid applied research, and, subsequently, access to the vast amount of funding in this space. Or participate and be bound by ambiguous laws and protocols surrounding proper disclosure of information. For the second route, risks include:

1) *Legal threats and lawsuits:* Even in the case of proper disclosure, the inconsistency in handling disclosures between organizations combined with the open-ended interpretation of existing laws (e.g., definition of "good-faith" research) sometimes results in legal issues for ethical researchers working to improve security.
2) *Loss of academic licenses and donation programs:* Universities are reliant on access to commercial tools and products for education and research purposes. Even when lawsuits are not pursued against a researcher, organizations are still within their right to deny these resources to the researcher's institution, which can seriously harm their ability to deliver on existing projects, pursue research funding, and place students in the job market.
3) *Prolonged publication delays:* After proper disclose, embargoes are often used to ensure that there is enough time for the affected company to provide effective workarounds. However, the indeterminate timescale by which hardware vulnerabilities are resolved could result in significant publication delays for the discoverer. University researchers are under immense pressure to disseminate artifacts of their work in high-impact venues. Competition for research funding is fiercer than ever today, and publication is vital to the advancement of academic careers and the pursuit of research funding. By agreeing to an embargo, a researcher could risk proper recognition and rewards if a less ethical research group circumvents disclosure and prematurely disseminates the same vulnerability.

Benefits:
Similar benefits to industry, especially *Standardization*, *Collaboration,* and *Education and training*. In addition, academics would benefit from a more welcoming environment where the disclosure/mitigation process is less ambiguous and less treacherous.

1.3.2 Limitations of Existing Ecosystem and Integration with a HVD

The idea of a public national hardware vulnerability database is fairly new and comes two decades after the establishment of a public national vulnerability database that was initially focused on software issues. For context, in the late 1990's there was a rich and thriving set of groups trying to discuss and share information surrounding software vulnerabilities, how they occurred, how they were attacked, how to recognize them, what to do about them, and how to prioritize them. While thriving, these ecosystems were disjointed, duplicative, convoluted to use, and had no correlations amongst or between them. Against this backdrop, the MITRE Corporation, a not-for-profit corporation that runs Federally Funded Research and Development Centers (FFRDCs) for multiple[1] departments of the US Government, came up with the idea of a Common Enumeration of Vulnerabilities and presented a paper with that title[2] at the University of Purdue's 2nd Workshop on Research with Security Vulnerability Databases in January of 1999 to a mixed audience of industry, government, and academia. While there was immediate argument about what constituted a vulnerability, there was no disagreement on the potential simplification and communication improvements a simple correlation identifier could bring to the vulnerability communities.

Years after its September 1999 launch of CVE, the need for common terminology about what weaknesses in software created those vulnerabilities and how they could be attacked led MITRE

[1] MITRE – We Operate FFRDCs, MITRE, https://www.mitre.org/centers/we-operate-ffrdcs
[2] Towards a Common Enumeration of Vulnerabilities, MITRE, http://cve.mitre.org/docs/docs-2000/cerias.html

to launch the Common Weakness Enumeration and the Common Attack Pattern Enumeration and Classification efforts and NIST to establish the National Vulnerability Database[3].

1.3.2.1 Common Vulnerabilities and Exposures (CVE®)

The CVE website provides a publicly available dictionary of publicly known vulnerabilities in commercial and open source software that helps individuals and organizations correlate the numerous types of public and private information about such public vulnerabilities in applications and other cyber-enabled capabilities. The dictionary is organized in an intuitive manner based on known vantage points and abides by a concise schema for describing related public information and offering a mechanism for the correlating the shared information about them. CVE's origin is based on the concept of a simple agreed upon list of identifiers for publicly known vulnerabilities to help correlate those pieces that are referring to or discussing the same public issue [22].

CVE entries or CVEs are a publicly known issues that need to be patched or mitigated to address an exploitable vulnerability. CVEs are created through CVE Naming Authorities (CNAs) that represent a web of organizations sharing the responsibilities of consistently associating CVE identifiers with security issues. While MITRE is currently the root CNA, and CNA of last resort, there are nearly 100 other organizations that constitute security researchers, have specific products, product types, or other specified domains for which they have been identified as the preferred CNA which the community should work with to assign CVE identifiers. A CNA of last resort is a role where, if for some reason the CNA that should be involved with assigning a CVE for an issue cannot be reached or is not responsive, the party trying to coordinate a new vulnerability can go to the CNA of last resort and work with them to get the CVE.

A **vulnerability** is a specific type of weakness or weaknesses in a product that is exploitable by a threat. Specific vulnerabilities are concrete examples of items (hopefully) described in the catalog of Common Weakness Enumeration (CWE) items and attackable by the attack patterns captured in the Common Attack Pattern Enumeration and Classification (CAPEC) collection. By linking a public vulnerability in a specific product to the weakness and attack pattern collections, organizations can leverage those collections and the information in them in their assessment and investigation into newly discovered examples of vulnerabilities and also offer opportunities to examine their own code collections for the same type of vulnerability.

1.3.2.2 Common Weakness Enumeration (CWE™)

The CWE website provides a publicly available catalog of the weaknesses that occur in the software architecture, design, code, or deployment of the software that can result in exploitable vulnerabilities and is meant to help individuals and organizations understand how these weaknesses in applications and other cyber-enabled capabilities occur. The catalog of weaknesses is organized in an intuitive manner based on known vantage points and abides by a comprehensive schema for describing related information about the weaknesses and sharing information about them. CWE's origin is derived from working with the real-world examples of the types of vulnerabilities that appear in software applications and meant to generalize those real-world flaws into conceptual patterns of what make software exploitable so that people can learn to recognize

[3] A Framework for Modeling the Software Assurance Ecosystem: Insights from the Software Assurance Landscape Project, SEI, https://apps.dtic.mil/dtic/tr/fulltext/u2/a528427.pdf

them early in the lifecycle of the software and either avoid introducing them or finding them quickly and addressing them before the software is put into operation.

CWE entries or CWEs are a mixture of types of vulnerabilities found by exploiting them "in the wild" or through examination and testing of software by hackers, developers, and testers. CWEs are created by generalizing a specific vulnerability in a particular product or through examination of the architecture, design, code, or deployed software and finding constructs that allow someone to do something that was not intended.

A **weakness** is a description of the common attributes and susceptibilities of an architecture, design, code, or deployment construct that allows an adversary to exploit the weaknesses in cyber-enabled capabilities. Weaknesses define the opportunities that an adversary may leverage and how the builder/defender can go about finding and removing them. If an organization is concerned with specific weaknesses because of the possible consequences from a successful attack on them, the relationships between CWEs and CAPECs can be used to identify the likely CAPECs for a CWE and that knowledge can inform the defender on options for defense.

The following is a list of the properties of weaknesses in CWE. For definitions of each of these properties see https://cwe.mitre.org/documents/glossary/index.html.

- Description
- Extended Description
- Alternate Terms
- Background Details
- Modes of Introduction
- Applicable Platforms
- Common Consequences
- Likelihood of Exploit
- Observed Examples
- Potential Mitigations
- Detection Methods
- Memberships
- Notes
- Taxonomy Mappings
- Related Attack Patterns
- References

1.3.2.3 Common Attack Pattern Enumeration and Classification (CAPEC™)

The CAPEC website provides a publicly available catalog of common attack patterns that helps individuals and organizations understand how adversaries exploit weaknesses in applications and other cyber-enabled capabilities. The catalog is organized in an intuitive manner based on known vantage points and abides by a comprehensive schema for describing related attacks and sharing information about them. CAPEC's origin is based on the concept of software design patterns – templates for the design and implementation of commonly used software techniques.

CAPEC entries or CAPECs are a mixture of attack patterns actively seen "in the wild", through proof-of-concept, or based on research surrounding what is theoretically possible to do. CAPECs are created by generalizing adversary behaviors and can be thought of as an extrapolation of things that could happen or be observed. For instance, how does an adversary take advantage of a software weakness – what are the steps taken, what is the knowledge needed to follow those steps, how difficult is it to achieve, how likely is success? These are the types of questions that CAPEC aims to answer.

An **attack pattern** is a description of the common attributes and approaches employed by adversaries to exploit known weaknesses in cyber-enabled capabilities. Attack patterns define the challenges that an adversary may face and how they go about solving them. They derive from the concept of design patterns applied in a destructive rather than constructive context and are generated from in-depth analysis of specific real-world exploit examples, research, and

technologies. If an organization is concerned with attacks on specific weaknesses because of the possible consequences, the relationships between CWEs and CAPECs can be used to identify the likely CAPECs for a CWE and that knowledge can inform the defender on options for defense.

The following is a list of the properties of attack patterns in CAPEC. For definitions of each of these properties see https://capec.mitre.org/documents/schema/index.html.

- Description
- Alternate Terms
- Likelihood of Attack
- Typical Severity
- Related Attack Patterns
- Execution Flow
- Prerequisites
- Skills Required
- Resources Required
- Indicators
- Consequences
- Mitigations
- Example Instances
- Related Weaknesses
- Taxonomy Mappings
- References

Mechanisms of Attack. The high level categories of CAPEC attack patterns are organized by the mechanisms of attack:

- Engage in Deceptive Interactions
- Abuse Existing Functionality
- Manipulate Data Structures
- Manipulate System Resources
- Inject Unexpected Items
- Employ Probabilistic Techniques
- Manipulate Timing and State
- Collect and Analyze Information
- Subvert Access Control

These categories are not organized around the goals or consequences of an attack pattern, but the steps necessary to performing an attack. The important aspect of the pattern is "how" to achieve the goal. The pattern may be used to cause many different consequences, but the steps are relatively constant.

Domains of Attack. A second view is available related to the different domains where attack patterns are relevant.

- Software
- Hardware
- Communications
- Supply Chain
- Social Engineering
- Physical Security

Software developers can use this view to become aware of relevant attack patterns and can design and implement their software to make it more difficult to put these attack patterns into practice, thereby reducing their application's risk surface.

The Domain of Attack view also showcases attack patterns related to social engineering, supply chain, and physical attacks. Social Engineering CAPECS reveal many non-technical means by which adversaries act on their objectives. The list of Supply Chain CAPECS illustrate how the long chain of parts/processes that make up the final piece of hardware/software can be attacked at various points along the way and the specific weaknesses that are leveraged in such attacks need to be captured and shared across those communities. CAPECs related to physical security can assist those needing to know the various ways their physical facilities, devices, and locations can be breached.

1.3.2.4 CMU CERT/CC

The CERT Coordination Center (CERT/CC) is the coordination center of the computer emergency response team (CERT) for the Software Engineering Institute (SEI), a non-profit United States FFRDC. The CERT/CC researches software bugs that impact software and internet security, publishes research and information on its findings, and works with business and government to improve security of software and the internet as a whole.

The CERT/CC works directly with software vendors in the private sector as well as government agencies to address software vulnerabilities and provide fixes to the public. This process is known as coordination. The CERT/CC promotes a particular process of coordination known as *Responsible Coordinated Disclosure*. In this case, the CERT/CC works privately with the vendor to address the vulnerability before a public report is published, usually jointly with the vendor's own security advisory. In extreme cases when the vendor is unwilling to resolve the issue or cannot be contacted, the CERT/CC typically discloses information publicly after 45 days since first contact attempt [23].

Software vulnerabilities coordinated by the CERT/CC may come from internal research or from outside reporting. Vulnerabilities discovered by outside individuals or organizations may be reported to the CERT/CC. Depending on severity of the reported vulnerability, the CERT/CC may take further action to address the vulnerability and coordinate with the software vendor.

The CERT/CC coordinates information with US-CERT and other computer security incident response teams. In general, US-CERT handles cases that concern US national security, whereas CERT/CC handles more general cases, often internationally.

1.3.2.5 National Vulnerability Database (NVD)

The NVD is a comprehensive cybersecurity vulnerability database that allows the tracking of vulnerability trends over time. This trending service allows users to assess changes in vulnerability discovery rates within specific products or within specific types of vulnerabilities. NVD data is represented using Security Content Automation Protocol (SCAP), a suite of specifications that standardize the format and nomenclature by which software flaw and security configuration information is communicated, both to machines and humans (see table below). The NVD includes databases of security configuration checklists for the National Checklist Program, listings of publicly known software flaws, product names, and impact metrics

As of the end of September 2018, the NVD contained the following resources:

- Over 115,111 vulnerability advisories, with an average of 70 new vulnerabilities added daily;
- 181 SCAP-expressed checklists across 120 platforms containing thousands of low-level security configuration checks that can be used by SCAP-validated security products to perform automated evaluations of the system state;
- 359 non-SCAP security checklists (e.g., English prose guidance and configuration scripts);
- 249 U.S. Computer Emergency Readiness Team (US-CERT) alerts; 4,480 US-CERT vulnerability summaries; and 10,286 SCAP machine-readable software flaw checks; and
- A product dictionary with over 166,000 operating system, application, and hardware name entries; and over 94,000 vulnerability advisories translated into Spanish.

NVD is hosted and maintained by the National Institute of Standards and Technology (NIST) and is sponsored by the Department of Homeland Security's US-CERT. The use of SCAP data by commercial security products, deployed in thousands of organizations worldwide, has extended NVD's effective reach. Increasing demand for NVD XML data feeds (i.e., mechanisms that provide updated data from data sources) and SCAP-expressed content from the NVD website demonstrates an increased adoption of SCAP.

In the past year, the NVD continued to see a significant upward trend in the number of vulnerabilities received. To support this increased workload, several strategies are in progress, including longer-term efforts to focus on vulnerability ontology and natural language processing to manage future growth. Overall, the NVD continues to experience an average download growth rate of over 10% per month.

A key component of the NVD is the Common Vulnerability Scoring System (CVSS), used for measuring the relative severity of software flaws. In 2017, the NVD began providing CVSS base scores following the CVSS v3 specification within the data feeds. Currently, NIST is working with the vulnerability community to enable the automated analysis of metrics, such as CVSS, establishing a baseline of the minimum information needed to properly inform the vulnerability management process, and facilitating the sharing of vulnerability information across language barriers. To assist in this work, a public draft of *NISTIR 8138, Vulnerability Description Ontology (VDO): A Framework for Characterizing Vulnerabilities*, was created to foster a conversation and collect feedback on the best mechanisms to improve the degree of automation within vulnerability management processes. In FY 2019, NIST is planning to develop the VDO iteratively through collaboration with the security automation community on GitHub. This approach will allow participation from as many stakeholders in the vulnerability community as possible.

Table 1.1: SCAP 1.3 Specifications

SPECIFICATIONS	DESCRIPTION
Languages	
Extensible Configuration Checklist Description Format (XCCDF) 1.2	Used for authoring security checklists/benchmarks and for reporting the results of evaluating them
Open Vulnerability and Assessment Language (OVAL) 5.11.2	Used for representing system-configuration information, assessing machine state, and reporting assessment results
Open Checklist Interactive Language (OCIL) 2.0	Used for representing checks that collect information from people or from existing data stores populated by other data collection methods
Reporting Formats	
Asset Reporting Format (ARF) 1.1	Used to express information about assets and to define the relationships between assets and reports

Asset Identification 1.1	Used to uniquely identify assets based on known identifiers and other asset information
Identification Schemes	
Common Platform Enumeration (CPE) 2.3	A nomenclature and dictionary of hardware, operating systems, and applications; a method to identify the applicability to platforms
Software Identification (SWID) Tags 2015	A structured metadata format for describing a released software product
Common Configuration Enumeration (CCE) 5	A nomenclature and dictionary of software-security configurations
Common Vulnerabilities and Exposures (CVE)	A nomenclature and dictionary of security-related software flaws
Measurement and Scoring Systems	
Common Vulnerability Scoring System (CVSS) 3	Used for measuring the relative severity of software flaws
Common Configuration Scoring System (CCSS) 1.0	Used for measuring the relative severity of device security (mis-)configuration issues
Content and Result Integrity	
Trust Model for Security Automation Data (TMSAD) 1.0	Guidance for using digital signatures in a common trust model applied to security automation specifications

1.3.2.6 Trust-Hub

Trust-hub is a web portal (available at www.trust-hub.org) for the hardware security community to exchange of ideas, benchmarks, platforms, tools, and educational resources, which is led by the University of Florida and has been supported by funds from the National Science Foundation (NSF) for the last eight years. In 2018, Trust-hub has added a Vulnerability Database sub-portal, which currently consists of (1) a physical attack taxonomy that highlights the mechanisms used by attackers in IC/PCB reverse engineering, hardware Trojan insertion, fault injection, side channel analysis, etc.; and (2) a list of academic and commercial computer aided design (CAD) tools available for protecting hardware IP and assessing system-on-chip (SoC) susceptibility to information leakage, hardware Trojans, side channel attacks, fault injection, and probing.

While this sub-portal is a promising joint initiative from academia and industry that aligns well with the goals outlined in this document, it remains a work-in-progress. An additional SoC vulnerability database is mentioned as "coming soon". The physical attack taxonomy only includes attacks categorized in one manner. Further, it can be improved by adding short descriptions, execution steps, consequences, and details about the time, skills, and resources required for each attack. The CAD tool list provides just the right amount of information about each tool, but it's not yet exhaustive. Researchers and vendors are encouraged to send details to the Trust-hub Vulnerability Database organizers to gain exposure for their tools and so that community has a better idea of which tools still need to be developed.

1.3.2.7 Trusted Silicon Stratus

The Nimbis Services Trusted Silicon Stratus (TSS) Supply Chain Risk Management (SCRM) Cloud, is a secure cloud service for government agencies to design integrated circuits (ICs) in a

private community cloud. This novel cloud service will be the first authorized-to-operate (ATO) multi-tenancy, inter-organizational, collaborative, trusted microelectronics (i.e., ASICs, FPGAs, and SoCs) SCRM ecosystem that provisions a design-to-release, manufacturing-to-operational, Life-Cycle-Management cloud computing infrastructure and enterprise architecture. A novel ecosystem for EDA software, IC design, and manufacturer data has been implemented as a private community cloud computing workflow-as-a-service (WFaaS) for DoD research and development teams, contractors, and government research laboratories.

The TSS offers a secure user configurable Trusted Microelectronics chip design ecosystem hosted on Amazon.GOV. This cloud-based service enables the affordable multi-organizational chip design and verification platform required for today's complex microelectronics development cycle. The TSS provides:

- EDA Tools
- Scalable Computing Resources
- Life Cycle Management
- PDK & IP Libraries
- Metered Billing
- Tiered levels of security
- Archiving and version control

1.3.3 Hardware Differentiation

1.3.3.1 Overview of Hardware Security Taxonomies

As described above, there are several description frameworks available for characterizing computer vulnerabilities in the form of the CVE ecosystem (CVE, CVSS, CWE, CWSS, and CAPEC). CVE and CVSS have been used by the community to document well known vulnerabilities such as Spectre/Meltdown and Rowhammer on the NVD. All of the CVE ecosystem is capable of usefully characterizing hardware vulnerabilities. However, CWE and CAPEC often lack key hardware weakness concepts and hardware specific attacks.

Outside of the CVE ecosystem, there have been several efforts to create more detailed taxonomies for particular domains of hardware vulnerabilities. Taxonomies of hardware attacks and attackers have a long history [24], [25]. The most well-known current effort is Trust-Hub, (described in section 3.2.7) which contains taxonomies of physical vulnerabilities, hardware trojans, and hardware obfuscation. Guin, DiMase, and Tehranipoor [26] , have also published a taxonomy of counterfeit ICs. There have been a variety of specialized taxonomies around specific weaknesses including a taxonomy of attacks based on speculative execution (e.g. Spectre/Meltdown) [27] , general fault attacks [28], differential power analysis [29], and injection attacks [30] to name a few (see Table 2). In aggregate, these taxonomies give a detailed picture of the world of hardware security. However, with a few exceptions [31], the focus is on the definition of concepts for attacks, weaknesses, and attackers. The attributes of these attacks are often missing that would allow modeling and the development of best practices around each kind of attack. For example, while optical fault injection (OFI) attacks are well studied, no description framework that we know of has included the concepts necessary to describe *when* an OFI attack would be successful on *what* device.

Table 1.2: Examples of hardware security taxonomies

Hardware Topic	Reference
Physical Vulnerabilities	Trust-Hub
Hardware Trojans	Trust-Hub
Obfuscation	Trust-Hub
Fault Attacks	Karaklajic et al. 2013 [28]
Speculative Execution	Canella et al. 2019 [27]
Counterfeits	Guin et al. 2013 [26]
Biased-Fault Attacks	Farhardy et al. 2015 [32]
DPA analysis and countermeasures	Marzouqi and Salah 2013 [29]
Fault Injection and simulation	Piscitelli [33]
Signal Injection Attacks	Giechaskiel and Rasmussen 2019 [30]
Attackers	Anderson and Kuhn 1996 [24]
Hardware attacks and attackers	Rae and Wildman 2003 [34]
Hardware attacks and attackers	Keommerling and Kuhn 1999 [25]
Smartcard	Yahay and Omar 2010 [31]

1.3.3.2 Hardware Weakness and Attack Pattern Descriptions

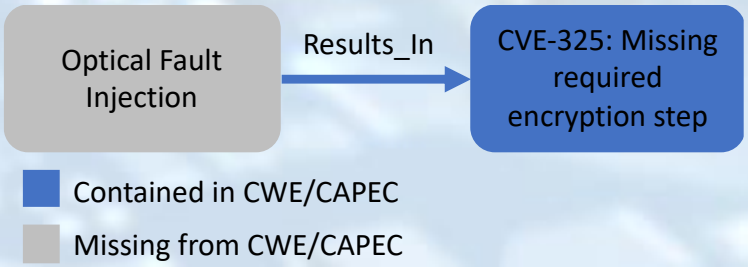
The concepts necessary to describe software weaknesses are well documented in the CWE description framework described in Section 3.2.2. In addition to software specific weaknesses, CWE includes many concepts required for computer security. Moreover, in addition to its taxonomies of concepts, it includes an ontology which allows for relations between those concepts and the concepts from CAPEC. This allows for the detailed description of hardware errors, even if hardware specific concepts are missing.

For example, the CWE concept "CWE-325: Missing Required Encryption Step[4]" is described as "The software does not implement a required step in a cryptographic algorithm, resulting in weaker encryption than advertised by that algorithm." While this effect could be caused by an inherent software weakness it is often the end result of hardware glitching attacks. The CWE ontology contains the relation "resultant" and CWE-325, in its definition, can be described by that relation. Thus CWE contains the concepts and relations to describe the action of a (for example) fault injection attack that causes an encryption algorithm to prematurely dump the plain text to a recoverable media.

CWE and CAPEC do not contain a concept for "Optical Fault Injection", though CAPEC does have a concept for "Fault Injection". From the hardware perspective, differentiating between types of fault injection is important since, for example, timing faults and optical faults will have wildly different mitigations. Many other notable attacks have no representative hardware concepts (e.g. Spectre/Meltdown, Rowhammer, power analysis etc.). However, as noted above, many of the key concepts have been described in some detail though the various taxonomic efforts over the past 20 years. A simple first step towards making the CVE ecosystem more

[4] https://cwe.mitre.org/data/definitions/325.html

relevant to hardware would be the incorporation of existing taxonomic concepts (e.g. those of Trust-Hub) into CWE and CAPEC.



1.3.3.3 Hardware Security Estimation

Estimating the security and risk of electronic devices is key to addressing hardware security concerns. They allow for a rational distribution of resources towards high impact security problems. Importantly, such estimations are necessary to incentivize the adoption of secure designs by industry. Security estimates must take several factors into account:

- Hardware security issues range from remotely accessible vulnerabilities such as Spectre/Meltdown, to counterfeit production, to subversion via fault injection. Generic security estimation is impossible and a threat model must be specified.
- Security estimates must take preventative measures into account such as secure designs, practices, and testing/verification.
- Hardware has a distinct and complex lifecycle beginning with multiple layers of design abstraction to fabrication and use in multiple environments.
- Hardware usually consists of integrated components beginning with integration of 3rd party IP at the level of IC design to printed circuit boards and large systems.

Indeed, integration poses an interesting problem for security estimation due to the combination of relatively secure components with other (possibly much less secure) components. This situation presents both risk and opportunity; restricting secure processes through trusted and secure components may allow for compensation for less secure components. However, an unsecure component may also serve as a "weakest link" for an otherwise secure system. Regardless, the ability to integrate security estimates across a system is essential to most applications.

The Dimensions of Security Estimation

There have been several approaches to establish measurements of security for microelectronics (see [35] for a review). Here we propose a trinary structure that roughly summarized security metrics:

- **Error Metrics (Impact)** – For a given identified error in a device, what is the severity of that error in terms of its damage, its accessibility, and the ability to mitigate that error. Error metrics usually measure "impact".
- **Trust Metrics (Reliability)** – An estimate as to whether the instantiated device is what you think it is in terms of design, production, chain of custody, and performance (to be free from defects). Trust metrics usually roughly measure the expected reliability of a device.
- **Subversion Resistance (Effort)** – An estimation of how secure the device is against deliberate attack. This can be measure roughly in terms of the effort that must be expended by the attacker to subvert the device. Though "effort" may also represent access to devices, the cost of the equipment necessary to attack the device, computer cycles required to break an encryption algorithm, etc. [36]

These scoring domains cover separate but closely related security descriptions. Error Metrics describe a defect that has been discovered or theorized. Trust Metrics can be informed by libraries of known Errors and their structure. Similarly, known errors inform the subversion resistance of a particular device. For example, a processors' vulnerability to the various attacks on speculative processing must be taken into account when estimating the effort required to subvert the processor in an attack.

Trust metrics are closely related to but differ from subversion resistance. A device may have a high trust score in the sense that it has a well-documented manufacturing process, and performs to specifications. However, it may be vulnerable to side-channel attacks and therefore have a relatively low subversion resistance. Similarly, a low trust score (which may indicate that the device was tampered with at some point during manufacture) will also indicate a low security resistance since the device may be more likely to contain a leverageable trojan. Finally, a device may be hardened against attacks but the design may be overly complex causing unexpected errors. Thus, it may have a high subversion estimate but low trust estimation.

Security Metrics in Hardware

Error Scoring: The scoring of errors in software has been key for the functioning of the software vulnerability patching infrastructure including vulnerability disclosure and mitigation strategy. The key efforts have been the Common Vulnerability Scoring System (CVSS, described in section 1.3.2.5)[5] and the Common Weakness Scoring System (CWSS)[6]. The CWSS is integrated with the Common Weakness Enumeration (CWE) so it's focused on software "weaknesses" which may or may not yet have been instantiated in actual software. CWSS is a more flexible than CVSS, being more abstract. CWSS also is more amenable to interaction with the Common Attack Pattern Enumeration and Classification (CAPEC) ontology.

There is no vulnerability scoring framework (e.g., an equivalent to CVSS) specifically for hardware, though there have been proposals in that direction [35]. CVSS has been used to characterize the severity of many vulnerabilities and weaknesses discovered in hardware and reported in the NVD. However, unlike most software patches, the mitigation of hardware vulnerabilities is often partial in its effectiveness; the attack surface still exists but may require a infeasibly high amount of effort to subvert.

Trust Metrics: A great deal of thought has gone into the general problem of establishing metrics of trust for software. For example, see Table 1 from Alberts et al. (2012) [37] for some of the metrics designed for different parts of the software lifecycle. These metrics reflect both verification of the software, but also the soundness of design and manufacture procedures. In hardware, trust has often been associated with reliability and industry has already produced a wide variety of tools to help designers and manufacturers improve yield, functionality, and SWaP (Size, Weight and Power). Thus, one view of trust metrics for a given device is the coherent combination of the various testing regimes the device goes through during design, manufacture and insertion. While there have been efforts to combine metrics [35], so far there is no standardized way to fuse design scores for a given threat model.

[5] https://www.first.org/cvss/specification-document
[6] https://cwe.mitre.org/cwss/cwss_v1.0.1.html

Table 1.3: Possible software security metrics from Alberts 2012

Life cycle phase	Example Security Measure
Requirements Engineering	• Percent security principles in design • Percent security requirements subjected to analysis • Percent of security requirements covered by attack patterns
Architecture and design	• Percent of design subject to attack surface measurement • Percent of design components subject to risk analysis
Coding	• Percent of software components subject to static and dynamic code analysis against known vulnerabilities and weaknesses • Percent of defects discovered during coding • Percent of components subject to code integrity and handling procedures
Testing	• Number of defects discovered during testing • Percent of components that demonstrate security requirements

Subversion Resistance: Software vulnerabilities tend to be viewed as binary - they are either exploitable or patched. This differs from many hardware attack surfaces (e.g., side-channel analysis) which can often be mitigated but are still available for exploitation to sufficient effort and expertise. "Bit-Security" in encryption is a good example of a non-binary measure of subversion resistance. Bit-security of n generally mean 2^n operations are required to break it. In the case of symmetric encryption this generally refers to the key length. Thus, AES symmetric encryption [38] with key length 128 also has bit-security of 128. Asymmetric public key encryption presents a different situation since there are always more efficient ways to attack an algorithm other than brute force methods. For example, 3072-bit RSA, which has key size of 3072, is estimated to have bit security of 128.

The problem of estimating the effort required to exploit a hardware vulnerability has been difficult and made assessing the effectiveness of hardware security features difficult. One approach for estimating subversion resistance is to appeal to "time to completion" approaches from project management. These can be used to produce a distribution of effort required for a successful attack [39]. Similarly, game theoretic approaches can be used to estimate the risk/benefits for attackers and defenders in an adversarial context [40].

Vulnerabilities in multiple lifecycles, data types, and integrated systems: The distinct lifecycle of hardware contains different phases at which vulnerabilities could be intentionally or unintentionally introduced. At the same time, these phases also present multiple data sources that can be used to provide assurance regarding security. For a given threat model, estimation of security may require a select set of data across different parts of the lifecycle. Hardware generally has a long (in the case of defense components a very long) life in comparison with software. Therefore, the use of a design may differ significantly in its later deployment than for what it was first intended.

1.3.3.4 Vulnerability Disclosure

Once a vulnerability in a public system has been discovered, the question is how to best alert relevant parties to produce a mitigation for that vulnerability before it can be exploited. When the vulnerability is discovered within the organization that produces a product, often the patch is developed and distributed without any public acknowledgement of the vulnerability. However, even internally discovered vulnerabilities are often given CVEs and publicly acknowledged.

Most cases of vulnerability disclosure arise when a 3rd party discovers the vulnerability. Then the reporting party must decide on how to proceed and whether to alert the product manufacturer or to disclose the vulnerability without input of the vendor (e.g. public disclosure). In the past, vulnerability disclosure was an extremely sensitive process with reporters being concerned about lawsuits brought by vendors.

In software, a formal vulnerability disclosure process has been largely recognized to be a benefit to both vendor and reporter [42]. This is not to say that disclosure is uncontroversial or without problems. The case has been made that disclosing vulnerabilities provides attackers with both a training ground and ammunition [19]. Moreover, due to the prestige associated with discovering major vulnerabilities, the exploitability of vulnerabilities can be exaggerated, and in some cases reported vulnerabilities turn out not to be true [43]. However, without a disclosure process, users may be unaware of unpatched vulnerabilities within their system. This not only puts them at risk, but any network into which they have privilege, as well as making them a target for botnet recruitment. Lack of disclosure also leaves system integrators at risk, as well as those who may use software without acknowledgement (often as a legally questionable practice). Several best practices have been published and collated in ISO/IEC report 29147 on Vulnerability Disclosure and the related report 30111 on Vulnerability handling.

Vulnerability disclosure can take one of several routes. The vender may discover the vulnerability and handle the disclosure and mitigation completely independently. A non-vender party may discover the vulnerability and report it to the vendor, who then handles the mitigation and disclosure independently. The discover may disclose the vulnerability without consulting the vender. Finally, the vender, discoverer and other stake holders may work together in mitigation and disclosure of the vulnerability in a process usually referred to as “Coordinated Disclosure”. Coordinated disclosure is of particular interest to the hardware case. Hardware vulnerabilities include not only the already complicated ecosystem required for software vulnerability mitigation, but also the original equipment manufacturers (OEMs) and their network or suppliers. Additionally, researchers who report a hardware vulnerability often have already expended extensive effort to discover and characterize the vulnerability and would like to receive credit through publicity or other reward. The coordinated disclosure process allows the reporter to continue to participate in the disclosure process.

The mitigation of a hardware vulnerability mitigation often requires the coordination between three types of technical parties: (1) OEMs, which are the companies that brand and distribute the products in which a vulnerability may occur, but might not manufacture all (or even most) of the component parts; (2) Hardware vendors manufacture the components but may have limited understanding of the larger functioning of the integrated product; and (3) Software distributers produce the operating systems (and often firmware) that utilize the device, and as such, they play a primary role in developing and testing software based mitigations. Because the process involves a continued coordination between the reporter, vendor, and mitigators it may be advantageous to involve a “coordinator” party. This may be a governmental entity (e.g., US CERT), and a commercial entity (bug bounty or commercial brokers), or other entities. CMU-CERT has a recent extensive report on coordinated disclosure [42]

IoT devices have been a focus for future disclosure policy. Although they may be in hard to reach places and nonstandard environments making updates or replacement difficult, they also play a role in safety-impacting systems. The constrained nature of IoT devices (sensors, programmable logic controllers, low power chips, embedded controllers, limited battery life, etc.) limit the inclusion of sophisticated security and the mitigation of discovered vulnerabilities. On the other hand, the devices (especially infrastructure supporting devices) have life expectancies of years or even decades. Finally, for smaller disposable devices sophisticated security features may be cost prohibitive.

Thus vulnerabilities or weaknesses discovered in IoT devices can carry large risks but at the same time may be difficult patch. This difficulty makes the treatment of vulnerabilities sensitive, and require a careful disclosure process. Unfortunately, many (even larger) IoT dependent companies do not have a defined vulnerability disclosure policies. A survey by the Internet of Things Security Foundation found that companies with IoT products showed that only 10% had any official disclosure policy. Out of this 10%, the actual disclosure policy varied radically:

- 41.9% disclosure policies gave no indication of the expected disclosure timeline.
- 0.9% disclosure policies had a hard deadline of 90 days for fixes to reported issues.
- 46.9% of policies also had a bug bounty program. Two of these programs were however by invitation only, so were not open for general contribution.
- 78.1% of policies supplied researchers with a public key for encryption to protect their communications and report details.
- 18.8% of policies utilized a proxy disclosure service (e.g., bugcrowd.com)

1.3.3.5 Summary of Gaps in the Hardware Vulnerability Description and Reporting Framework

Identification of vulnerabilities: CVE has been used successfully to identify and report hardware vulnerabilities in the NVD.

Description framework for weaknesses and attack patterns: The CWE and CAPEC description frameworks contain much of the necessary structure to describe hardware vulnerabilities. However, many hardware specific concepts are missing and those that do exist are at too high of a level to be useful (e.g., “fault injection”). Simultaneously, there have been many efforts to create taxonomies for hardware weaknesses and security issues. The inclusion of some of these into CVE and CAPEC could greatly extend their ability to describe hardware vulnerabilities adequately. On the other hand, very little has been done to systematically characterize hardware physical processes that would allow for detailed descriptions of weaknesses. This sort of detailed description framework would allow for the development of shard practices within the industry as a whole, as well within specific supply chains.

Scoring and hardware security estimates: Metrics for estimating hardware security appears to be a point of greatest differentiation between the software and hardware ecosystems. Scoring hardware vulnerabilities and weaknesses differ from software in three key ways. First, unlike most software vulnerabilities, many hardware vulnerabilities are not patchable. Second, if they can be mitigated, the mitigation is often partial – the attack surface persists though may

implausible to exploit. Third, because the attack surfaces persist, measuring the effort to subvert the device through these attack surfaces is a key metric. Finally, metrics that are robust to fusion across the lifecycle of the device and integration into higher level systems is key for useful risk estimates. These differences have further impacts in how hardware security metrics need developed:

- Any estimate needs to have a **defined threat model** (e.g. trojan, reverse engineering, remote activation, etc.) This is key for establishing what data from which parts of the lifecycle are relevant.
- Because hardware vulnerabilities are not patchable and persist in larger systems, there must be some way of **combining security estimates across the lifecycle** of a device. A system for countering fault injection may actually induce a different vulnerability [41]. Similarly, an emphasis on reduced accessibility during development may lead to a lack of robustness in a device when it is deployed. A lack of awareness of this sort of tradeoff will induce new weakest leaks in security development.
- The security estimates must be **comparable and integrable**. This is especially key for preventing weakest links in integrated systems. For example, a device may be secure for a copy machine but in a company network that device may form the weakest link and allow subversion the larger system.
- How **confident are we in our estimate**? A score of any kind by itself is insufficient. We need to have a methodology of assessing confidence in this score and how to improve the score [26]. A given score may be fully achievable (i.e. a functional verification on a relatively small device). Understanding the coverage of a given approach, and therefore our confidence in the result is essential. Finally, ideally a test framework could be established which allows automated data generation for increasing confidence. For example, as new Trojan models come to light, new tests can be generated to assure such an error is not present in already existing devices.
- There are two major perspectives to estimate risk in microelectronics. The first one is trust, that is the **risk that the component is already compromised**. This compromised state may be due to an unintentional error or defect, an intentional error (Trojan), or because the device is in fact a counterfeit. It is still important to think about the effort for an attacker to exploit this compromised state, however, the fundamental question is whether the device is and performs as expected. The second related estimate is the **effort of the attacker** to exploit a device. Indeed, with any device made by humankind, there is probably a way to exploit the device given enough resources. In the case of encryption, bit security has been a very good measure. However, many hardware security features need to be measured in expertise, person-hours/costly device time and accessibility. Effort required to compromise a device is of course closely related to whether the device is already compromised in terms of assurance; a device with a Trojan or known error with be easier to compromise.
- A final category that is of preeminence in software, is the **ability to mitigate future errors**. Mitigation in the case of hardware can be extremely difficult and may decrease performance or not be possible at all. However, thinking towards the future threats to a device is still key to security estimation. Even if there is not a way to generically mitigate future errors (as is the case in software), we can develop flexible testing regimes which

can be brought to bear dynamically and efficiently to at least determine the presence of future vulnerabilities.

Responsible disclosure and mitigation: The standards and practices around responsible disclosure have largely worked well with previous hardware vulnerability discoveries. However, very few IoT providers (~10 %) have a defined disclosure program.

1.3.3.6 Third Party IP Validation

Commercial FPGAs are in widespread use in military systems because they have a cost advantage over ASICs. Their programmable nature and system on chip (SOC) make them vulnerable to cyber malware and malicious insertion similar to microprocessors based systems. Since they are programmable, however, many of the security assurance tools there are in place for software based releases can be applied to FPGA IP releases. In particular,

- There is a need to establish a policy for trusted FPGA hardware and third party IP (3PIP)
- Creation of a database with IP issues related to performance and security
- IP authentication via hash marks
- Application of scoring system to FPGA 3PIP (Security, Maturity, ...)
- Engagement of manufacturer and 3PIP generators in a comprehensive independent validation and verification (IV & V) effort.
- Independent IV & V organization for FPGA 3PIP.

1.3.4 Funding and Sustainment

1.3.4.1 Design/Planning

The related DBs (database) CWE and CVD that are managed by the FFRDC (Federally Funded Research and Development Center) MITRE Corp., are of a similar or parallel type used in software whereas this HVD is concerned with the related hardware used with its programming software. Leveraging off of these existing DBs make the most sense for propel this HVD into existence.

Metrics is the ultimate and urgently needed goal. With HVD in place, a comparison or standard reference could be made as to whether certain things in hardware designs have or have not been accomplished. This would allow, for instance, a Joint Federated Assurance Center (JFAC) lab to compare and judge quantitatively the quality and authenticity of the design and hardware. Established metrics allow a conveyance of standards to other entities (other labs and programs) for a uniform horizontal metric of analysis. The HVD should be tailored from the start as a database for metrics to gauge a hardware instantiation.

Classification restrictions should be addressed at some point in the future, but at present the initial version of HVD should be unclassified. This will allow a much wider range of contributors to encompass a wider breath of ideas. A nominal list of experts could include academics, service JFAC labs, prime contractors, service acquisition programs, and other existing related resources such as CWE and CVD. Possibly controlled distribution D[7] or E[8] could be appropriate if necessary.

[7] DISTRIBUTION D. Distribution authorized to Department of Defense and U.S. DoD contractors only

[8] DISTRIBUTION E. Distribution authorized to DoD components only.

An estimate for the timeline of development in this working group's opinion is 18-24 months. The amount of funding for development of HVD could be approximately $3M per year with this mostly going to a MITRE type FFRDC, academics, and contractors. Government participation is somewhat covered in JFAC funding for development.

1.3.4.2 Prototype testing

DoD JFAC labs have the need to use the HVD database for verification and validation of designs the acquisition programs would be using. A JFAC lab would then be able to provide feedback and clarification to an acquisition program during the design phase of hardware development.

1.3.4.3 Production

Production can be the initial setup of the actual database with links provided through a cloud connection with the classification or distribution restrictions enforced appropriately. The actual distribution of the database would be potentially through the DoD JFAC portal links. JFAC labs working with program customers would allow links to the database based on need.

1.3.4.4 Long-term Management and Sustainability

Funding for the HVD would come from the Office of the Secretary of Defense (OSD) funding that is allocated to the DoD JFAC/TAM (Trusted and Assured Microelectronics) labs for trust and assurance efforts for each services' pertinent acquisition programs. There are 4 main labs in the US and several smaller specialized contributing labs. The funding vehicle could be the existing BAA structure managed by AFRL that is funded from OSD for any aspect of implementing trust and assurance in DoD microelectronics. The JFAC/TAM labs each contribute to the BAA funding, proposal reviews, and strategic direction. This structure allows small and large contributors to the database development.

For the long term management and sustainability of the database, this should be modeled after the CWE/CVD databases currently managed by the FFRDC MITRE. Using MITRE or a similar FFRDC to manage the database is appropriate with the funding being annual government MIPR direct to the entity. Annual budget for database management, availability, and upgrade sustainability could be $2-3M/year.

1.4 Conclusion and Future Directions

Hardware vulnerability and weakness sharing has been a controversial topic due to their hard-to-patch nature. However, we identify several fruitful directions for research and collaboration that can improve the hardware security landscape.

- The CVE ecosystem (CVE, CWE, CAPEC, etc.) has the underlying structure necessary for the reporting and high level description of hardware vulnerabilities, weakness, and attack patterns. However, the current concepts need to be expanded to better represent hardware specific weaknesses and vulnerabilities such as Spectre/Meltdown and physical attacks.
- A great deal of work has already been done in creating taxonomies of hardware weaknesses and even their mitigations. However, most are not well developed beyond the conceptual level. An ontology that not only represented the topics but also the underlying physical processes might allow for better policing of hardware vulnerabilities within value chains, as well as sharing across the industry.

- Hardware security scoring can be greatly improved to support better planning and allocation of security resources. Key for the hardware metrics are context dependent, graded measures of mitigation such as the effort to subversion. Additionally, due to the persistence of hardware vulnerabilities, models of how vulnerabilities threaten larger integrated systems are important.
- Responsible vulnerability disclosure has played a key role in the (largely patachable) world of software vulnerabilities. However, even unresolved weaknesses such as those associated with speculative execution have been successfully publicly disclosed. Disclosure is important for IoT devices, but most IoT firms have no disclosure policy.

Organization of future technical working groups: Note that the development of the technologies required to support hardware vulnerability ontology is beyond the scope of the current working group. However, the TAME HAWCS working group will offer the opportunity for a variety of researchers from various interested parties to organize working groups in support of technical objectives. So far, the HAWCS working group has identified four potential technical workgroups: shareable examples of public hardware vulnerabilities for research, hardware vulnerability ontology, vulnerability scoring and triage approaches, and procedures for an HVD.

References

[1] J. V. Bulck *et al.*, "Foreshadow: Extracting the Keys to the Intel {SGX} Kingdom with Transient Out-of-Order Execution," presented at the 27th {USENIX} Security Symposium ({USENIX} Security 18), 2018, pp. 991–1008.

[2] CTS Labs, "amdflaws_whitepaper.pdf," *Severe Security Advisory on AMD Processors*. [Online]. Available: https://safefirmware.com/amdflaws_whitepaper.pdf. [Accessed: 17-Oct-2019].

[3] P. Kocher *et al.*, "Spectre Attacks: Exploiting Speculative Execution," *ArXiv180101203 Cs*, Jan. 2018.

[4] M. Lipp *et al.*, "Meltdown: Reading Kernel Memory from User Space," presented at the 27th {USENIX} Security Symposium ({USENIX} Security 18), 2018, pp. 973–990.

[5] "The Hidden Toll of Fixing Meltdown and Spectre | WIRED." [Online]. Available: https://www.wired.com/story/meltdown-and-spectre-patches-take-toll/. [Accessed: 17-Oct-2019].

[6] S. Adee, "The Hunt For The Kill Switch - IEEE Spectrum," *IEEE Spectrum: Technology, Engineering, and Science News*. [Online]. Available: https://spectrum.ieee.org/semiconductors/design/the-hunt-for-the-kill-switch. [Accessed: 17-Oct-2019].

[7] P. Swierczynski, M. Fyrbiak, P. Koppe, A. Moradi, and C. Paar, "Interdiction in practice—Hardware Trojan against a high-security USB flash drive," *J. Cryptogr. Eng.*, vol. 7, no. 3, pp. 199–211, Sep. 2017.

[8] "China Used a Tiny Chip in a Hack That Infiltrated U.S. Companies," *Bloomberg.com*, 04-Oct-2018.

[9] M. H. Tehranipoor, *Counterfeit integrated circuits*. New York, NY: Springer Science+Business Media, LLC, 2015.

[10] R. C. Stogdill, "Dealing with Obsolete Parts," *IEEE Test*, vol. 16, no. 2, pp. 17–25, Apr. 1999.

[11] L. Columbus, "2018 Roundup Of Internet Of Things Forecasts And Market Estimates," *Forbes*. [Online]. Available: https://www.forbes.com/sites/louiscolumbus/2018/12/13/2018-roundup-of-internet-of-things-forecasts-and-market-estimates/. [Accessed: 14-Oct-2019].

[12] C. Boit, C. Helfmeier, and U. Kerst, "Security Risks Posed by Modern IC Debug and Diagnosis Tools," in *2013 Workshop on Fault Diagnosis and Tolerance in Cryptography*, 2013, pp. 3–11.

[13] "CWE - Common Weakness Enumeration." [Online]. Available: https://cwe.mitre.org/. [Accessed: 14-Oct-2019].

[14] "CAPEC - Common Attack Pattern Enumeration and Classification (CAPEC)." [Online]. Available: https://capec.mitre.org/. [Accessed: 14-Oct-2019].

[15] J. D. Howard and T. A. Longstaff, "A common language for computer security incidents," *Other Information: PBD: 1 Oct 1998*, 01-Oct-1998. [Online]. Available: https://digital.library.unt.edu/ark:/67531/metadc706351/. [Accessed: 14-Oct-2019].

[16] "CVE - Common Vulnerabilities and Exposures (CVE)." [Online]. Available: https://cve.mitre.org/. [Accessed: 21-Oct-2019].

[17] "Computer Emergency Response - An International Problem," *studylib.net*. [Online]. Available: https://studylib.net/doc/14324930/computer-emergency-response---an-international-problem. [Accessed: 14-Oct-2019].

[18] P. M. Mell, K. A. Scarfone, and S. Romanosky, “The Common Vulnerability Scoring System (CVSS) and its Applicability to Federal Agency Systems,” *NIST InteragencyInternal Rep. NISTIR - 7435*, Aug. 2007.
[19] G. Uht and 2019, “Let’s Keep it to Ourselves: Don’t Disclose Vulnerabilities,” *SIGARCH*, 31-Jan-2019. .
[20] G-19A Test Laboratory Standards Development Committee, “Test Methods Standard; General Requirements, Suspect/Counterfeit, Electrical, Electronic, and Electromechanical Parts,” SAE International.
[21] “Accellera Systems Initiative.” [Online]. Available: https://accellera.org/. [Accessed: 21-Jan-2019].
[22] “AS6171: Test Methods Standard; General Requirements, Suspect/Counterfeit, Electrical, Electronic, and Electromechanical Parts - SAE International.” [Online]. Available: https://www.sae.org/standards/content/as6171/. [Accessed: 14-Oct-2019].
[23] “Vulnerability Disclosure Policy.” [Online]. Available: https://vuls.cert.org/confluence/display/Wiki/Vulnerability+Disclosure+Policy. [Accessed: 18-Oct-2019].
[24] R. Anderson and M. Kuhn, *Tamper Resistance – a Cautionary Note*. 1996.
[25] O. Koemmerling and M. G. Kuhn, “Design Principles for Tamper-Resistant Smartcard Processors,” p. 12.
[26] U. Guin, D. DiMase, and M. Tehranipoor, “A Comprehensive Framework for Counterfeit Defect Coverage Analysis and Detection Assessment,” *J. Electron. Test.*, vol. 30, no. 1, pp. 25–40, Feb. 2014.
[27] C. Canella *et al.*, “A Systematic Evaluation of Transient Execution Attacks and Defenses,” *ArXiv181105441 Cs*, Nov. 2018.
[28] D. Karaklajić, J. Schmidt, and I. Verbauwhede, “Hardware Designer’s Guide to Fault Attacks,” *IEEE Trans. Very Large Scale Integr. VLSI Syst.*, vol. 21, no. 12, pp. 2295–2306, Dec. 2013.
[29] H. Marzouqi, M. Al-Qutayri, and K. Salah, “Review of gate-level differential power analysis and fault analysis countermeasures,” *IET Inf. Secur.*, vol. 8, no. 1, pp. 51–66, Jan. 2014.
[30] I. Giechaskiel and K. B. Rasmussen, “Taxonomy and Challenges of Out-of-Band Signal Injection Attacks and Defenses,” *ArXiv190106935 Cs*, Jan. 2019.
[31] S. Yahya and N. A. N. Omar, “Security validation of smartcard: MCOS,” 2010.
[32] N. F. G. Schaumont Bilgiday Yuce, Patrick, “Analyzing the Efficiency of Biased-Fault Based Attacks,” 663, 2015.
[33] R. Piscitelli, S. Bhasin, and F. Regazzoni, “Fault Attacks, Injection Techniques and Tools for Simulation,” in *Hardware Security and Trust: Design and Deployment of Integrated Circuits in a Threatened Environment*, N. Sklavos, R. Chaves, G. Di Natale, and F. Regazzoni, Eds. Cham: Springer International Publishing, 2017, pp. 27–47.
[34] A. Rae and L. Wildman, “A taxonomy of attacks on secure devices,” p. 14.
[35] A. Kimura, “Development of Trust Metrics for Quantifying Design Integrity and Error Implementation Cost,” PhD Thesis, The Ohio State University, 2017.
[36] S. Moein and F. Gebali, “A Formal Methodology for Quantifying Overt Hardware Attacks,” p. 8.
[37] C. Alberts, J. Allen, and R. Stoddard, *Risk-Based Measurement and Analysis: Application to Software Security*. 2012.

[38] “FIPS 197, Advanced Encryption Standard (AES),” p. 51.
[39] E. L. McCombs, M. E. Elam, and D. B. Pratt, “Estimating Task Duration in PERT using the Weibull Probability Distribution,” *J. Mod. Appl. Stat. Methods*, vol. 8, no. 1, pp. 282–288, May 2009.
[40] J. Graf, “Trust games: How game theory can guide the development of hardware Trojan detection methods,” in *2016 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2016, pp. 91–96.
[41] Y. Li, K. Sakiyama, S. Gomisawa, T. Fukunaga, J. Takahashi, and K. Ohta, “Fault Sensitivity Analysis,” in *Cryptographic Hardware and Embedded Systems, CHES 2010*, vol. 6225, S. Mangard and F.-X. Standaert, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 320–334.
[42] A. D. Householder, G. Wassermann, A. Manion, and C. King, “The CERT Guide to Coordinated Vulnerability Disclosure,” CARNEGIE-MELLON UNIV PITTSBURGH PA PITTSBURGH United States, 2017.
[43] “Software Vulnerability Disclosure Is a Real Mess | News & Opinion | PCMag.com.” [Online]. Available: https://www.pcmag.com/news/370074/software-vulnerability-disclosure-is-a-real-mess. [Accessed: 21-Oct-2019].

Chapter 2: Design for Security

Contributors:

- Michel A. Kinsy (Lead) Boston University
- Adam Kimura (Scribe) Battelle
- Brian Cohen (Advisor) IDA
- Ro Cammarota Intel AI
- Daniel J. Radack IDA
- Radu Teodorescu OSU
- Gang Qu University of Maryland - College Park
- Jia Di University of Arkansas
- Michael Mehlberg Cryptography Research, Inc.
- Steven McNeil Xilinx Inc
- Brandon Eames Sandia
- Eslam Tawfik OSU
- Joe Jarzombek Department of Homeland Security
- Robert A. Martin MITRE Corporation
- Antonio de la Serna DARPA

Design for Security (DFS) working group's focus areas are (1) formal set of definitions - the objective is to establish a set of definitions to guide the rest DFS work, (2) usage/user models - identify use scenarios and specify how the proposed DFS models fit into existing design flows, and finally (3) security metrics - both qualitative and quantitative set of metrics for classifying one system as more secure than another system.

2.1 Problem Statement

Heterogeneous system-on-chip (SoC) architectures have many advantages. They generally consist of highly specialized application-specific processing units and general-purpose cores. Their performance and power can be better optimized, and the specialization of computing units to different tasks promises greater energy/area efficiency. For example, authors in [1] show that a heterogeneous computer architecture can outperform a comparable-area homogeneous architecture by up to 63%. An equally important fact is that, the design of these systems and the development of associated kernels and applications are increasingly global. As shown by the Semiconductor Industry Association (SIA) report in Figure 1, the top participants of the semiconductor industry (manufacturing, fabrication, and packaging companies, etc.) currently come from more than 23 countries on three continents (Asia, North America, and Europe). A single semiconductor production process can involve at least four countries and trips around the world. As a result, the provenances of IPs become harder to establish.

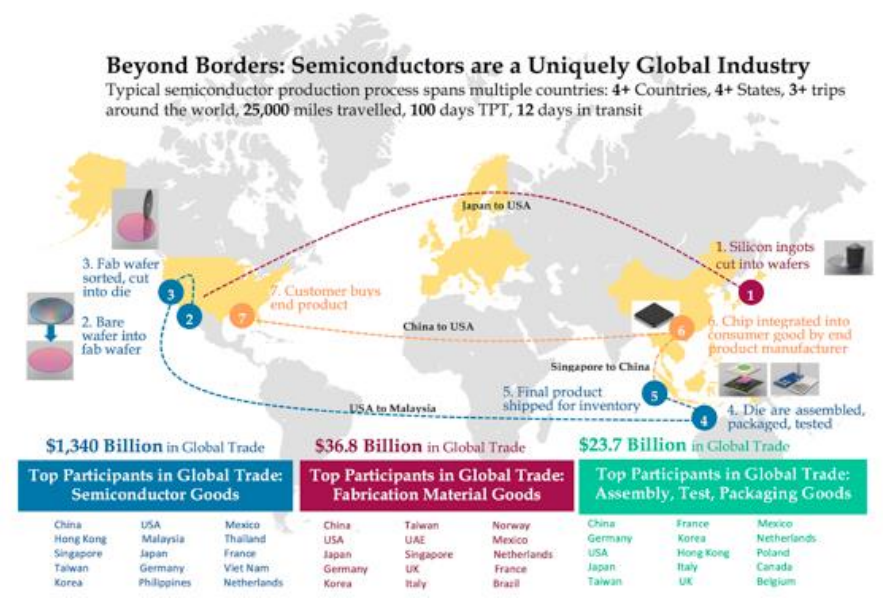


Figure 1: The Global Supply Chain Map from the Semiconductor Industry Association

In addition, the run-time interactions on those heterogeneous SoC systems maintains a high level of complexity. Processes may share processing elements, data, and I/O processing time. Similarly, processing elements/processors/cores themselves may share cache or memory structures, network-on-chip resources, and I/O modules [31].

In all, despite the many advantages of heterogeneous SoC systems, e.g., higher performance, better energy efficiency, lower communication latencies, they also give rise to a number of critical security challenges. For example, the globalization of the semiconductor industry has made IP provenance checking more challenging. Moreover, run-time interactions and resource sharing of processing elements have created a fertile attack ground and represents the vulnerable point of heterogeneous SoC architectures.

2.2 Security Challenges in Current Microelectronics

We identify two major security challenges in current heterogeneous SoCs, namely, supply chain trust issues and insufficient software-only protections.

2.2.1 Trust Issues in the Supply Chain

Most heterogeneous SoCs may consist of processing elements from different IP providers or manufacturers. In addition, the applications running on them may have varying levels of security and trust, all executing on the same compute platform while sharing resources with each other [2, 3]. For example, in a multicore smartphone chip, complex run-time interactions between processors running untrusted applications can sometimes circumvent the built-in security guards, in order to access memory blocks in protected sections of the phone. These systems are vulnerable to a variety of software-centric attacks, such as spyware, Trojans and viruses, as well as hardware-centric attacks such as side channel attacks and hardware Trojans [4].

There have been several efforts exposing the security issues in the global supply chain. As authors in [5] pointed out, in the semiconductor industry, the time-to-market window is made as short as possible in order to maximize the revenue. Therefore, design strategies based on intellectual property (IP) reuse and manufacture outsourcing are widely adopted. For this very reason,

vulnerabilities in the IPs and Trojans inserted in the manufacturing process (hardware design, fabrication, packaging, etc.) have turned into a growing concern. Worse, since most semiconductor companies are focusing on design only and becoming "fab-less", malicious hardware modifications and Trojans can be inserted at fabrication time by untrusted foundries [6]. For example, according to recent reports [7, 8], counterfeits and Trojans are the two major causes of hardware vulnerabilities in the U.S. military systems.

On the other hand, realizing that most semiconductor designers and users must live with an untrusted supply chain, there is a trend to move away from relying only on "trusted foundries", to "security by design" and "chain of custody" [9]. The U.S. Department of Defense (DoD) released their view on approaches to ensure trusted semiconductor products, as shown in Figure 2. This flow diagram covers a wide spectrum from design to operation and highlights the fact that malicious objects or behaviors could be injected at any stage of the design lifecycle.

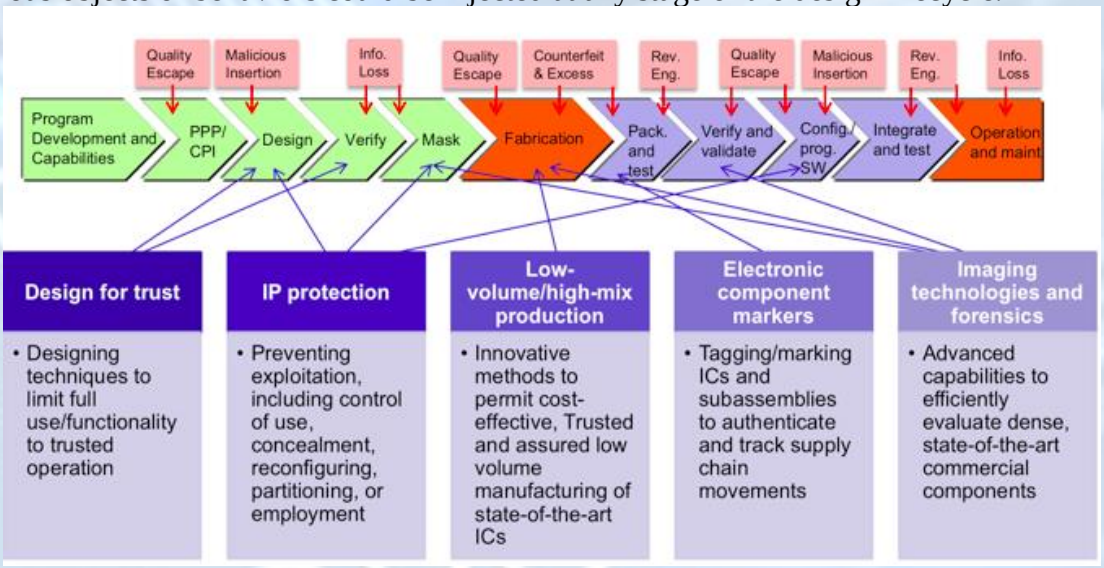


Figure 2: New Trust and Assurance Approaches. Source: DoD [9]

2.2.2 Insufficient Security by Software-only Protections

The consensus today is that conventional approaches and software-only add-on schemes have failed to provide enough security protections and trustworthiness. As Defense Advanced Research Projects Agency (DARPA) showed in their brief (Figure 3), it is increasingly expensive to defend by software-only solutions than to attack [10].

In addition, some hardware security issues are beyond the capability of software-based protections. In an analysis by IoT Inspector on over 4,000 embedded devices from 70 different hardware vendors, they found 580 cryptographic secret keys shared between a great number of devices [11]. For instance, not only have some manufacturers used identical keys among their own devices, but also there are keys shared amongst different vendors. The issue with key sharing often times comes because of constraints derived from market requirements on the product. Other times it is due to "laziness," e.g., malpractice or even ignorance.

Another alarming fact is that, when researchers scanned the Internet for those 580 keys, they found that at least 230 of them are actively used by more than 4 million IoT devices all over the world, with the United States being the top affected country (by owning 26.3% of those devices). These carelessly shared secret keys can be used to forge counterfeits, which can deceive authentication and breach secure systems. A similar report [12], the authors stressed that, since all these keys are hard-coded into the IoT devices, this issue cannot be resolved using software patches. In a report

titled *Is Computer Security Becoming a Hardware Problem?* [13] Kocher, the co-author of the SSL v3.0 protocol, stated that: "To make progress, we need another building block: simple, high-assurance hardware for secure computation."

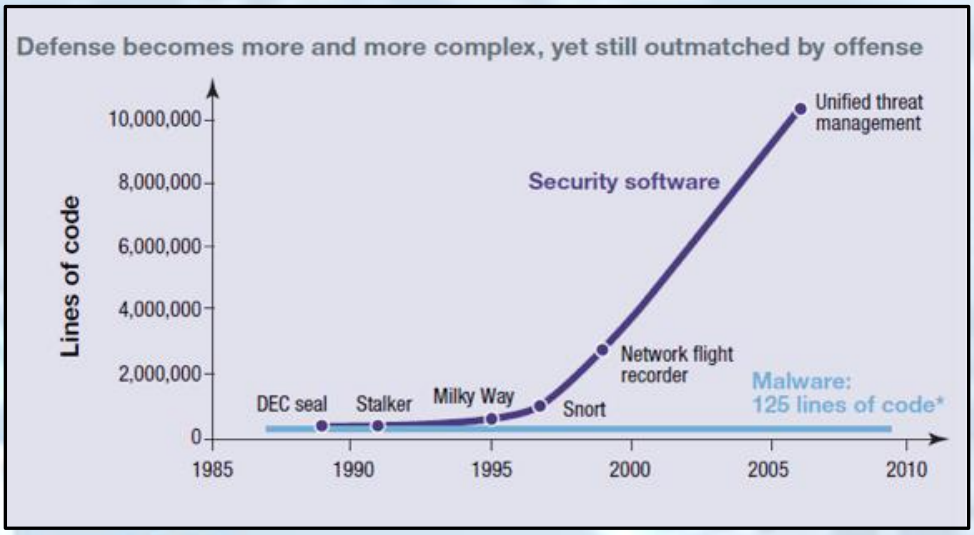


Figure 3: Defense Advanced Research Projects Agency (DARPA): Brief to Defense Science Board (DSB) Task Force (May 2011)

As a matter of fact, there have been increasing proposals to utilize hardware as the root of trust (RoT) for critical tasks such as authentication, key storage, and key transmission etc. The Computer Security Resource Center (CSRC) at the National Institute of Standards and Technology (NIST) launched a project to study the use of hardware RoT in securing computing systems [14]. Companies such as Synopsis and Intel have also been exploring this field [15, 16] even before NIST. The advantage of this approach is that it can provide functions that malware cannot tamper with. A system equipped with hardware RoT typically cannot be breached unless the attackers gain physical access to it.

2.3 Security Definitions

Security is traditionally concerned with maintaining properties prescribed by system designers. These policies are typically applied to items deemed of value and worth protecting under a given threat model.

2.3.1 Semantic Gap within Microelectronics System Design Flow

With the ever-increasing complexity of current SoCs, ad-hoc or independent security policy definitions or implementations have become unworkable. Considering security policies in isolation undoubtedly leads to lower risk assessment coverage, side-effects, and composition inconsistencies [2].

Computer security is the protection of computing systems and their data from malicious use. For the design of future secure computing systems and microelectronics, we must address how to implement multilevel user-defined security policies in these systems and how to optimally and securely share resources and data among processing elements. Design for security examines the set of computational or architectural methods applicable to systematic secure computer systems designs.

The notions of security and secure system design are highly contextual. Therefore, their specification, interpretation, implementation, and evaluation need to also be context based. In this context, “Design for Security” is a set of definitions and methods for (i) checking the integrity of computing processes, (ii) monitoring and controlling the access to system resources, and (iii) predicting computing services and behaviors.

Under this umbrella, “Security” is characterized by the capability to protect the system from malicious attempts, which either drive the system away from the accepted functional conditions or exploit the limitations and restrictions of the system [4]. These attempts can be either invasive or non-invasive.

For general-purpose computing microelectronics, 2018 and 2019 have not been kind to those harboring the illusion that their computers are secure. First came the news of Spectre, Meltdown, and their subsequent variants [17, 18]. Then came further attacks on the same model, so that making a computer secure – even at the hardware level – has become akin to the carnival game of whack-a-mole. Moreover, the mitigations to each of these attacks have been specific and piecemeal, often microcode patches rather than true solutions.

Broadly speaking, the steps in the microelectronics system design cycle are functional description, functional specification, functional and implementation modeling, and physical implementation. Figure 4 illustrates this design flow.

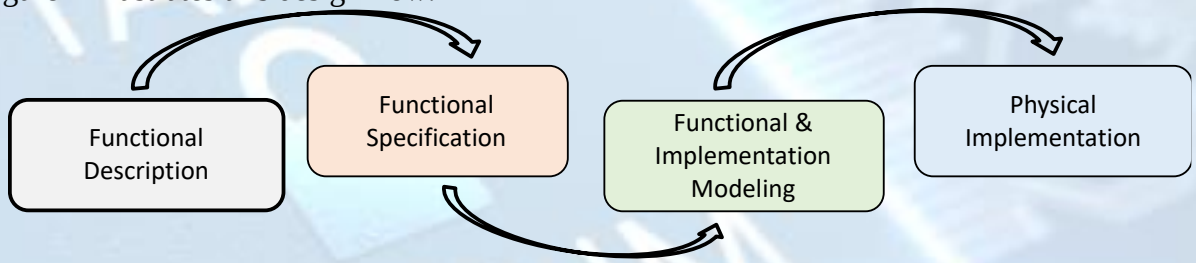


Figure 4: Generalized Microelectronics System Design Flow

For example, a functional specification is a set of requirements or features that a system must implement. From the functional implementation to the physical design implementation, the key design-verification focus areas are (i) functional correctness validation, (ii) performance and efficiency analysis, and (iii) reliability testing.

The partitioning of the computing system design flow into steps or layers originates from the need for abstraction. Abstraction-based design enables system designers to cope with the ever-increasing complexity of the system. One of the benefits that abstraction brings into the design fold is the principle of separation of concerns. This principle gives us the mechanisms to achieve modularity. We usually specify module internals separately from the interfaces between modules through which the communication flows in order to implement desired functionality of the whole system. Interfaces tend to be minimal and clear in order to allow for different internal implementations of adjacent layers. However, the abstraction has also failed to establish strong security guarantees across these levels [3]. In the process of defining the interfaces, some details of the design that are considered non-essential to the functionality of the system are abstracted out and are not necessarily preserved across the interfaces. This abstraction technique has traditionally worked. In fact, it is adopted across the full computing system stack, from instruction set architectures (ISAs) and memory subsystems to high-level application programming interfaces (APIs). However, this abstraction process sometimes removes or misses important contextual or security-related information. Essentially, the process may create semantic gaps or widen them.

Therefore, the key design challenge in microelectronics system security – both general purpose and embedded should be bridging the gap between high-level descriptions of the system and the system's physical implementation. Secure computer system design patterns should aim at eliminating or minimizing any semantic gaps between these design steps and implementing strong module and state isolation. Since both active side-channel attack (e.g., cache side-channel attack) and passive side-channel attack (e.g., thermal analysis) exploit gaps within the design steps. The ultimate push should be towards a greater semantic consistency between the different implementation steps to (a) avoid data leakage, (b) prevent an unauthorized party (users, processes, etc.) from accessing services, resources data or discovering the existence of a critical piece of data, (c) check the integrity of computing processes, (d) monitor and control access to system resources, or (e) have predictable computing behaviors [19].

Microelectronics system security requirements simply define what a system must do and be in order to perform securely. It can be functional or non-functional requirement. A functional security requirement is something that describes functional behavior that enforces security. Functional security requirements mostly have to do with access control, data integrity, and authentication. Non-functional requirements describe what a system must be.

It is very important to set the security goals at the inception of the system. Every system fits a need or a requirement. Setting these security requirements within the functional specification leads towards better design and implementation of the system. However, important questions need to be answered such as what kind of vulnerabilities one is looking to prevent? How will one measure whether a requirement is met? What measurements will one perform to ensure that a system module or sub-module is secure and does not have built-in vulnerabilities? When defining a design security requirement, what formalism should one use to express it? What specificity about the threat model for which the system needs protection should one provide? How can one validate the practicality or completeness of such a threat model? Hence, a security requirement should be built much like a functionality requirement. It should not be vague or unattainable.

In summary, bridging these design semantic gaps will require security to be a first-class focus during development. For instance, most ISAs do not incorporate a great level of security, except for a handful of security related instructions. Creating a richer contract between a hardware implementation and software applications could be a step towards a smaller semantic gap and fewer side channels. For instance, an ISA that specifies if/when an instruction can influence common side channels such as cache state would eliminate such side channels. However, this assumes that the ISA is correctly implemented which would require formal verification tools to ensure and is frequently difficult to do in practice.

2.3.2 Secure Enclaves Based Design

One of prevailing secure microelectronics or processor design techniques thus far has been the "secure enclave" approach. The National Institute of Standards and Technology defines an enclave as "a set of system resources that operate in the same security domain and that share the protection of a single, common, continuous security perimeter." Although there are many "secure enclave" definitions, implementations, and security guarantees, their core design principle is a mechanism to provide initial state attestation and to ensure the integrity and privacy of the execution from an adversarial entity [20 ,21, 22, 23].

Secure enclaves present two key challenges. On one hand, they can lead to a blanketed system partitioning where the enclaves act more like co-processors with substantial redundancies and overheads. On the other hand, attempts to closely integrate the enclave into the rest of the

computing system or to enable tightly coupled resources sharing, often result in the creation of security gaps. This is tension between performance/efficiency and process isolation is the root cause. In other words, in SoC architectures, run-time interactions can be exceedingly complex, and therefore it may be impossible for a designer to anticipate all of them. This can result in untrusted applications circumventing the chips' built-in security measures and accessing resources of the system that should, in theory, be off-limits to such applications.

2.3.3 Use of Micro-contracts to Design Secure Systems

One potential solution to close these semantic gaps may be the use of "formal micro-contracts" [24]. The "security micro-contract" or simply "micro-contract" paradigm is a secure computer systems design approach through minimal contracts (i.e., micro-contracts) between adjacent layers or modules of the architecture. These contracts have strict structures that contain security-relevant details of each connected layer and the secure properties that must be preserved to assure confidentiality, integrity, and availability of the data of interest. Where secure enclaves take a wholesale approach to processor security, micro-contracts provide a retail approach.

Micro-contracts are designed as a unified framework to tackle computing vulnerabilities in a formal and security-centric manner [24]. The solutions for known vulnerabilities normally mitigate each vulnerability using a relatively unique method. However, they do not describe security implications of the mitigations themselves. The solutions are normally demonstrated on minimal examples contained in vulnerability databases. Their implementation in complex systems usually require numerous other changes whose security implications are studied separately from the vulnerability being mitigated. Micro-contracts are designed as a framework for security analysis and policy enforcement in complex computing systems divided into layers. Policies are formed based on the concrete attacks being analyzed. A general overview of the micro-contracts framework is given in Figure 5. The framework provides a systematic approach for defining the security objects/ features/ invariants of the system under design and tracking their propagation through the system design flow.

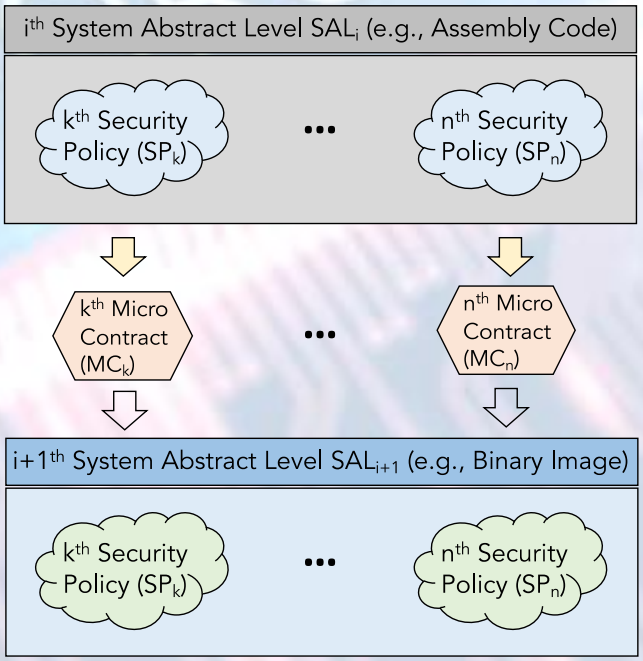


Figure 5: Overview of the Micro-contracts Framework.

Consequently, one can be assured that the implementation of these security objects or micro-contracts conform to their functional description, specification, or implementation (i.e., every step of the system design flow). To guarantee that security properties are maintained throughout the design flow or design step stack, micro-contracts express those properties and track their implementations or refinements across the design cycle and during execution.
To determine if a physical implementation of a functional specification and security definition is correct and complete, metrics must be used. Security metrics for a functional specification measure the likelihood of a non-secure state while metrics for a physical implementation try to detect non-secure states.

2.4 Security Metrics

Many of the existing metrics that have been proposed focus on quantifying hardware assurance for evaluating design integrity and supporting risk model frameworks [25]. As Design for Security (DFS) techniques and approaches are developed, DFS metrics must be established and play a critical role in assessing the effectiveness of various applied DFS strategies as well as provide insight into the cost/benefit tradeoffs associated with security. A portfolio of DFS security metrics will carry a range of use cases from identifying the areas of the design that are more susceptible to attack or exploitation to evaluating the effectiveness of a selected security mitigation to a known vulnerability. Furthermore, DFS metrics will quantify the value of sensitive assets as well as guide the amount of protection needed to achieve the desired level of security.

2.4.1 Salient Security Definition Questions

When implementing security, the essential questions that must be answered are: *Which assets of the system require the most security? How does one objectively decide that enough security has been implemented? What is the cost/benefit of adding more security?* and *What level of risk is now associated with the design and larger system after elevated amounts of security have been added?* Understanding how vulnerabilities of an asset propagate up to compromise the larger system and network is critical to understand. As such, security metrics could provide the necessary framework for performing microelectronics security evaluations as well as grant more observability into the cost, benefit, and risk trade-offs made when considering increased security for an asset. DFS metrics can also provide objective methods for evaluating the different mitigation solutions or improvements to asset security, thus establishing quantitative differentiation between one approach over another.

2.4.2 Asset Value Metrics

DFS metrics should constitute a portfolio of different metric types that span Functional Design, Design Implementation, and Run-time System States. In addition, DFS metrics should be specific to individual assets as they are considered in the overall security of the system they make up. Figure 6 displays a portfolio of generalized asset security metrics that must be developed. An asset can be generically defined as any instantiated hardware that holds or touches sensitive or valuable data. This could range from an encryption core to a data bus transferring sensitive data between two locations.
There are four primary metric domains that should be considered. The *Asset Value Metrics* domain quantifies the value of the asset based on the value of the data that flows through it. If the asset contains non-sensitive data only, the asset can be categorized as low value. Data that is highly sensitive, requiring highest protection, would be classified as a high value asset. The *Asset*

Vulnerability Metrics domain identifies areas of the asset that are unprotected or more susceptible to attack and need to be hardened against exploitation. These metrics quantify areas of the design that may have low reachability and observability that make assurance testing difficult to perform or evaluate. They identify the parts of the design that may be vulnerable to side channel attacks or locations that are easy to insert and hide malicious circuitry. The vulnerability assessments and metrics serve as a guide for directing where and how the security should be implemented. The *Level of Security Metrics* is a framework that evaluates the current level of security as well as the required level of security that is necessary for achieving the desired asset protection level. Finally, the *Cost of Increased Security* measures the resource cost of implementing additional security to increase the asset level of security. Each of these four domains relationally intersect with one another because each has a direct impact on the other. For example, if one considers a datapath that handles highly sensitive data, this can be considered a high value asset. A vulnerability assessment is performed on the datapath asset to identify vulnerabilities that could be exploited by an attacker. This assessment leverages external vulnerability databases and vulnerability discovery techniques. The security of the asset is evaluated in its current state to determine what level of inherent security the asset has with no additional security implemented. Based on this information and with an understanding of the inherent vulnerabilities, one determines that it is necessary to add more security to increase the level of protection for this asset. The vulnerability assessment guides the implementation of security solutions. If the current level of security is not high enough, the cost associated with the required security level is understood from the identified vulnerabilities. This process is repeated iteratively until an optimal security state for the asset is obtained. This is considered the optimal design security given the asset value and implemented security cost. Hardware Vulnerability Ontology databases and associated Vulnerability Mitigation databases will play a critical role in discovering and identifying the inherent vulnerabilities in the design as well as provide countermeasures and solutions for mitigating them [26].

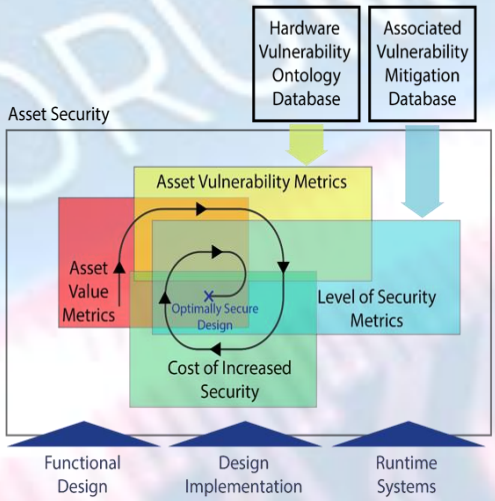


Figure 6: Spiral Flow for Realizing Optimally Secure Design for an Asset

Figure 7 expands on this concept by displaying four assets in a system that vary in asset value and required security. Each asset carries a value metric, V_{asset}, determined by the value of the data that flows through the asset. Based on the asset value, a level of required security, $S_{required}$, is determined to be necessary for protecting it. The required security may not be at the same level as the current state of security, expressed as S_{asset}. One must then decide if ΔS_{cost} is necessary to spend to improve current level of asset security. For example, Asset 4 in Figure 7 shows the

current security assessment as S_{asset4} = LOW, but based on the protection requirements of a high value asset, $S_{required}$, S_{asset4} should be HIGH. ΔS_{cost} therefore expresses the resource cost for moving from the lower tier of security to the highest tier. Since Assets 1 and 2 are both low value, there is minimal security required and design resources can be focused on implementing additional security for Asset 4, increasing S_{asset_4} from LOW to HIGH. Once the security for each asset has been determined, a higher abstracted system level metric, S_{system}, can be used to provide an overall assessment of the system security that encompasses all of the assets. Both Figure 6 and Figure 7 capture the essences of DFS metrics, but they will map uniquely into the three cross section domains of Functional Design, Design Implementation, and Run-time System States.

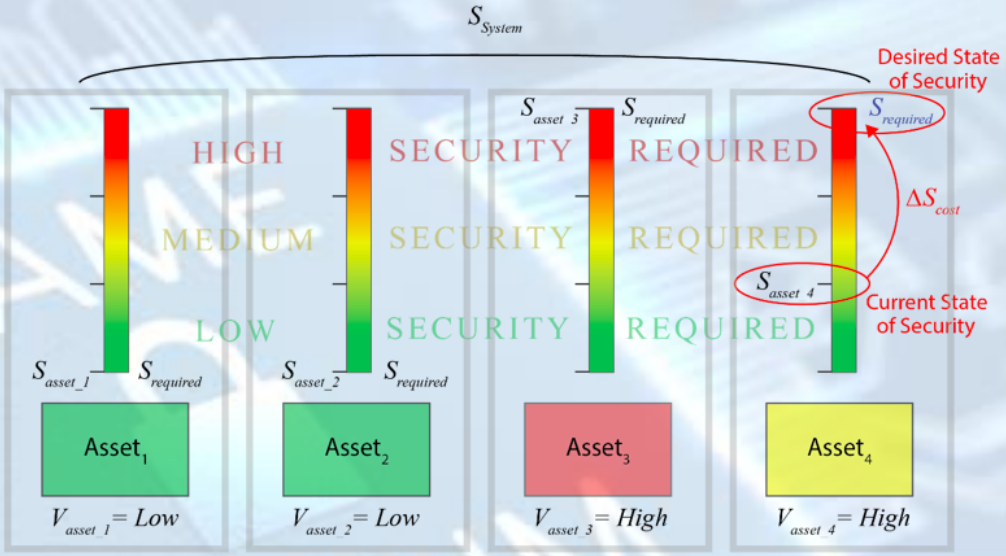


Figure 7: Security Metrics for Assets of a System

2.4.3 Asset Specific Functional Design Security Metrics

Functional Design, when discussed in the context of design for security, encompasses the design specification, verification, and validation at the functional level as it relates to security. Traditionally, at the specification level one is defining performance specs such as speed and power as well as determining the behavioral requirements of the design. By introducing security requirements at the specification phase, one now considers the sensitivity of data and the value of assets that need the most critical protection once the design has been deployed in the system. Security cannot be an afterthought, being "bolted on" late in the implementation or run-time phases. It should be considered as a parallel effort with the designing of the system itself. As such, security must be considered in the initial design specification development stage. In specification development, one is identifying the parts of the design that require more rigorous security and considering the trade-offs in the performance, power, and real estate domains that must be made for achieving the necessary levels of security.

Security at this level can be represented as properties [27] that enable security features or hardware properties that will be built into the fabric of the design. As one traverses the design path, the properties are eventually realized as additional circuits. Understanding the trade-offs that must be made is a critical piece to Functional Design DFS. For example, design real estate may be less important than high security, therefore, the designer determines to use more silicon area to

facilitate the implementation of additional circuitry that will mitigate certain vulnerabilities. It is also necessary to develop asset specific security metrics that grant more resolution into the various types of assets one may encounter in the system.

System components such as a Low Noise Amplifier (LNA), Central Processing Unit (CPU), or Analog to Digital Converter (ADC), for example, all have unique figures of merit that are specific to their design and performance. These figures of merit are often used as specification metrics that the final design must meet. For example, when designing an LNA, the design must meet certain metrics expressed in the design specification such as noise figure, output gain, and operational bandwidth. In contrast, when designing an ADC, the specification captures performance metrics such as sampling rate, signal-to-noise ratio (SNR), effective number of bits (ENOB), and dynamic range. As such, different figures of merit have been developed for different components that effectively capture and characterize their unique function and performance. It does not make sense to use ENOB as a performance metric for an LNA. A similar philosophy should be applied when considering security metrics for different assets in a system. Security metrics should be developed and tailored to specific asset classifications so that security is not lost in the nuances of different asset types. Different assets will have different vulnerabilities and thus require different mitigation approaches. To put it another way, optimum security for Asset A may look completely different than optimum security implemented for Asset B. As such, developing security metrics that are asset specific can be an effective means for getting a more accurate assessment for each asset's security.

2.5 Usage Model

As design for security techniques and practices mature, one of the main challenges to the adaptation of emerging security processes into modern design practices is a well-defined DFS usage model that captures the essence of different security use case scenarios. We define a usage model as the framework that directs the implementation of security processes into a specific application context. Usage models are viewed differently by different organizations and design teams, depending on the target system or run-time requirements. As such, the type of asset, the target system, and deployment context are the primary drivers for where and how the design needs to be secured. The DFS usage model framework should clearly map proposed DFS techniques and models into existing design flows and business processes. As new countermeasures and hardening techniques are developed, the DFS usage models must be flexible to adopt security advances to keep them relevant with the state-of-the-art. For each use case scenario, a unique threat model must be defined and analyzed. A design that is going into a GPS system for a military application will have a very different threat model than a microcontroller going into an IoT device for a commercial product. A unique threat model specific to the use case establishes the requirements for security and allows an associated DFS usage model to be leveraged as a blueprint for implementing the appropriate security strategy.

Figure 8 illustrates the process for arriving to a unique threat model and the general approach to apply DFS. There are three primary sectors that conduct microelectronics design: the government, commercial, and academic sectors. Due to varying goals and requirements, each sector has different focal interests based on the organization objectives and value structure. For example, the commercial sector generally values lowest possible cost, fastest time-to-market production, and protecting end user data and privacy. The government sector places high importance on reliability and system assurance to effectively carry out mission objectives. Cost and speed-to-market are of

much lower priority when compared to the prioritization structure of the commercial sector. Academia is primarily concerned with conducting novel research and extending fundamental science; therefore, reliability or data protection may be a much lower priority. Although each sector has a different primary focus, an overlap still exists between each sector that represents their common shared interests. For example, government and commercial entities both care about the reliability and assurance of parts; however, for commercial entities, it ranks significantly lower in priority when contrasted to the government due to the associated cost tradeoffs. In a similar fashion, cost is important to government entities, but does not generally have an equal level of prioritization as the assurance of a component to reliably function throughout the life of a critical system it is being deployed in. This difference in prioritization structures for each sector creates different threat concerns that cannot be addressed with a single prescriptive approach.

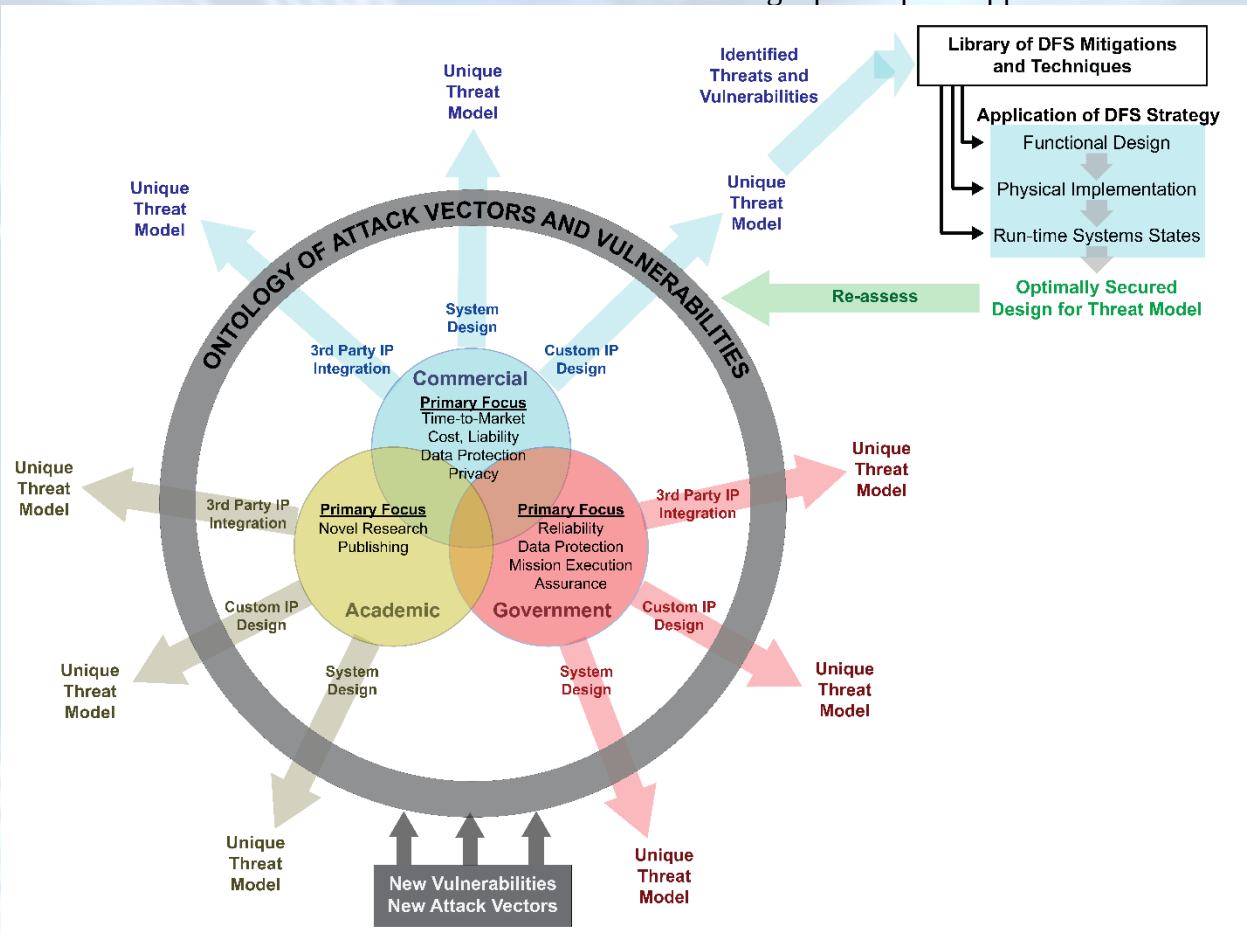


Figure 8: Design for Security Usage Model to Arrive at a Unique Threat Model to Guide Implementation of a Design for Security Strategy

A unique threat model must be developed for each of the three sectors discussed that account for the different ways designs are developed and integrated today. Across all sectors, there are different types of development and integration models, each carrying their own unique set of threats. Some threats engender higher concern for security and some lower, depending on the environment or design use case. The key to arriving to the appropriate security strategy begins with identifying which threats map to the specific design use case being evaluated. Once this is understood, the associated countermeasures can be instantiated against each identified threat for mitigation and vulnerability hardening. To this end, we consider this the DFS Usage Model.

Figure 8 presents the three sectors that microelectronics design is conducted in. Within each sector, there are three primary design development models that we will consider: Custom IP Design, 3rd Party IP (3PIP) Integration, and System Design. This can be extended to other more nuanced development and integration models; however, the approach for determining the unique threat model remains the same.

In Custom IP Design, the design is being developed from a requirements specification by a design team. Security concerns revolve around inherent design vulnerabilities and protecting the sensitive assets. The design team focuses on building security into the design by identifying appropriate vulnerability countermeasures that will mitigate known hardware vulnerability exploits or other discovered security vulnerabilities. Attention is also given to ensuring that sensitive assets are thoroughly protected. 3PIP presents the challenge of often being delivered in a black box format where the details of the IP are hidden from the system integrator. As such, security concerns arise with the 3PIP introducing new vulnerabilities that could compromise the security of the system it is embedded within. Furthermore, with the way IP is developed today, it is entirely possible that a purchased 3PIP design could contain unknown functionality that would exhibit malicious behavior if triggered [28]. Security around 3PIP would involve implementing external measures around the IP to monitor it or constrain its access to sensitive assets. System Design integrates both 3PIP and Custom IP Design components together and is concerned with the flow of sensitive data between IP blocks and the interaction of the different system components together with one another. Vulnerabilities at the system level must also be mitigated.

The ring around the three sectors represents an ontology and database of known attack vectors and hardware vulnerabilities that are continuously added to and updated as new ones are discovered, much like the Common Weakness Enumeration (CWE) of known software vulnerabilities and exploits [29]. When the threat model is being defined for a specific use case, the vulnerability ontology is the framework for the assessment process that determines the areas of the design that need to have countermeasures implemented or other DFS techniques applied. Once an assessment is generated, the DFS Usage Model guides the implementation of security measures to mitigate the vulnerabilities.

A generalized example of this process is shown in right corner of Figure 8. The design being developed is a Custom IP Design in the commercial sector by a company for a target product. The time-to-market, cost, and data protection are the highest priority for this use case. A specification has been developed along with a behavioral design. As the design “passes through” the Ontology of Attack Vectors and Vulnerabilities, the assessment is conducted that identifies all the vulnerabilities and exploits that are inherent to the design. Since data protection is a high priority, the threats and vulnerabilities surrounding the data leakage, protection, etc. can be prioritized based on vulnerability metrics [30]. The unique threat model for this design can now have the DFS strategy applied.

2.6 Future Directions

This paper captures the essence of Design for Security and discusses the core principles that should guide and be embedded in DFS frameworks as they are constructed. Looking forward, the techniques to generate the metrics for quantifying the different aspects of security discussed in this paper need to be developed and presented with tangible implementation use case examples. The mathematical frameworks and methods need to be standardized across platforms and abstraction levels to help gain traction in the community and reduce the trend of producing ad-hoc approaches that do not map well into other applications. A critical component to the standardization of

practical DFS usage models will be the establishment of a well-defined ontology database that catalogs known or observed hardware vulnerabilities from which designs can be evaluated against. Complementary mitigation databases must also be developed to assist in addressing discovered vulnerabilities.

References

[1] Kumar R, Tullsen DM, Ranganathan P, Jouppi NP, Farkas KI. Single-ISA heterogeneous multi-core architectures for multithreaded workload performance. InProceedings. 31st Annual International Symposium on Computer Architecture, 2004. 2004 Jun 23 (pp. 64-75). IEEE.

[2] Kinsy M, Bu L, Isakov M, Mark M. Designing secure heterogeneous multicore systems from untrusted components. Cryptography. 2018 Sep;2(3):12.

[3] Kinsy MA, Khadka S, Isakov M, Farrukh A. Hermes: Secure heterogeneous multicore architecture design. In 2017 IEEE International Symposium on Hardware Oriented Security and Trust (HOST) 2017 May 1 (pp. 14-20). IEEE.

[4] Tehranipoor M, Koushanfar F. A survey of hardware trojan taxonomy and detection. IEEE design & test of computers. 2010 Feb 5;27(1):10-25..

[5] Salmani H. The Global Integrated Circuit Supply Chain Flow and the Hardware Trojan Attack. In Trusted Digital Circuits 2018 (pp. 1-11). Springer, Cham.

[6] Forte D, Perez R, Kim Y, Bhunia S. Supply-Chain Security for Cyberinfrastructure [Guest editors' introduction]. Computer. 2016 Aug 15;49(8):12-6.

[7] IDST (2019). Threats to ICT supply chains including counterfeit electronic components and hardware Trojans present critical risk to military systems, researchers developing new innovations.

[8] Zorz, Z. (2018). Supply chain compromise: Adding undetectable hardware Trojans to integrated circuits.

[9] Lapedus, M. (2018). A crisis in DoDs trusted foundry program?

[10] Kaufman, D. (2011). DARPA: Cyber analytical framework.

[11] Khandelwal S. (2015) Millions of IoT devices using same hard-coded CRYPTO keys.

[12] Byrne, M. (2015). Internet of things encryption vulnerabilities show how often devs rip-off code.

[13] Byrne, M. (2016). Is computer security becoming a hardware problem?

[14] NIST (2018). Root of Trust.

[15] Elias, A. (2017). Understanding hardware Roots of Trust.

[16] Intel (2017). Intel security essentials.

[17] Trippel C, Lustig D, Martonosi M. Meltdownprime and spectreprime: Automatically-synthesized attacks exploiting invalidation-based coherence protocols. arXiv preprint arXiv:1802.03802. 2018 Feb 11.

[18] Weisse O, Neal I, Loughlin K, Wenisch TF, Kasikci B. NDA: Preventing Speculative Execution Attacks at Their Source. In Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture 2019 Oct 12 (pp. 572-586). ACM.

[19] Yellu P, Boskov N, Kinsy MA, Yu Q. Security Threats in Approximate Computing Systems. InProceedings of the 2019 on Great Lakes Symposium on VLSI 2019 May 13 (pp. 387-392). ACM.

[20] Aga S, Narayanasamy S. InvisiPage: oblivious demand paging for secure enclaves. InProceedings of the 46th International Symposium on Computer Architecture 2019 Jun 22 (pp. 372-384). ACM.

[21] Evtyushkin D, Elwell J, Ozsoy M, Ponomarev D, Ghazaleh NA, Riley R. Iso-x: A flexible architecture for hardware-managed isolated execution. InProceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture 2014 Dec 13 (pp. 190-202). IEEE Computer Society.

[22] Jang Y, Lee J, Lee S, Kim T. SGX-Bomb: Locking down the processor via Rowhammer attack. In Proceedings of the 2nd Workshop on System Software for Trusted Execution 2017 Oct 28 (p. 5). ACM.

[23] Kiriansky V, Lebedev I, Amarasinghe S, Devadas S, Emer J. DAWG: A defense against cache timing attacks in speculative execution processors. In2018 51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO) 2018 Oct 20 (pp. 974-987). IEEE.

[24] Kinsy MA, Boskov N. Secure Computing Systems Design Through Formal Micro-Contracts. InProceedings of the 2019 on Great Lakes Symposium on VLSI 2019 May 13 (pp. 537-542). ACM.

[25] A. Kimura, S. Bibyk, B. Dupaix, M. Casto and G. Creech, "Metrics for Analyzing Quantifiable Differentiation of Designs with Varying Integrity for Hardware Assurance," in Government Microcircuit Applications & Critical Technology Conference (GOMAC Tech), Reno, March 2017.

[26] J. Bellay, D. Forte and R. Martin, "Hardware Assurance and Weakness Collaboration and Sharing (HAWCS) ," 2019.

[27] Hu W, Althoff A, Ardeshiricham A, Kastner R. Towards Property Driven Hardware Security. In2016 17th International Workshop on Microprocessor and SOC Test and Verification (MTV) 2016 Dec 12 (pp. 51-56). IEEE.

[28] T. Reece, "Assessing and Detecting Malicious Hardware in Integrated Circuits," Vanderbilt University, Nashville, etd-12052014-031031, Dec. 2014.

[29] Mitre, "Common Weakness Enumeration," [Online]. Available: https://cwe.mitre.org/. [Accessed 3 November 2019].

[30] A. Kimura, "Development of Trust Metrics for Quantifying Design Integrity and Error Implementation Cost," Electronic Thesis or Dissertation, The Ohio State University, https://etd.ohiolink.edu/, 2017.

[31] Ray S., Chen W., and Cammarota, R. "Protecting the supply chain for automotives and IoTs". In Proceedings of the 55th Annual Design Automation Conference (DAC'18). ACM, New York, NY, USA, Article 89, 4 pages.

Chapter 3: Microelectronics Security and Trust-Grand Challenges

Contributors:

- Mark Tehranipoor (Lead) University of Florida (Lead)
- Yousef Iskander (Scribe) Cisco Systems
- Len Orlando (Advisor) AFRL
- Waleed Khalil Ohio State University
- Rosario Cammarota Intel Corporation
- Sohrab Aftabjahani Intel Corporation
- Greg Creech GLC Technologies
- Glen (Dave) Via AFRL
- Thomas Kent Battelle
- Swarup Bhunia University of Florida
- Saverio Fazzari Booz Allen Hamilton
- Pim Tuyls Intrinsic ID
- Vincent Mooney Georgia Tech
- Farimah Farahmandi University of Florida
- Jonathan Valamehr Tortuga Logic
- Brian Cohen Institute of Defense Analysis
- Navid Asadi University of Florida
- Matt French USC ISI
- Cayley Rice Leidos
- Brian Dupaix AFRL
- Mattew Casto OSD

3.1 Problem Statement

Computing devices have become indispensable parts of our daily lives. The expanding surface of end-user devices, such as smartphones, smartwatches, and smart home appliances constructing the internet of things (IoT) has made ubiquitous computing feasible and well within reach of the mass market. At the same time, industrial cyber-physical systems consisting of numerous industrial robots and controllers, a nationwide power-grid, communications networks, and transportation systems have rooted themselves to be essential parts of modern private and public infrastructure. The usage of electronic devices and systems in national defense and security applications, space programs, as well as in critical infrastructure is not far behind. This overwhelming usage of electronic devices and systems places us in a new era where the number of connected electronic devices exceeds the human population, and it is estimated that over 50 billion devices will be employed and mutually connected by the year 2020 [1].

While the benefits of ubiquitous computing in our lives are indisputable, it also creates several security and trust concerns from a multifaceted viewpoint – security and trust challenges faced during the device/system design and development processes, manufacturing and system

integration stages, as well as during distribution and deployment until the device's end of life, and beyond [44]. Over the years, hardware components and supply chains have been considered secure and trustworthy. However, recent reports on vulnerabilities and attacks against electronic hardware refute this assumption [3].

During the design phase, the primary trust issue arises from the use of third-party Intellectual Property (IP) blocks that often system-on-chip (SoC) integrators rely on to reduce the cost of development and speed the time to market [2]. In most cases, these IP cores are designed by several parties across the globe, and there are no comprehensive solutions that provide complete trust and transparency that the design is free from malicious insertions, leakage, or backdoors while also addressing piracy concerns [45].

At the same time, designing IP cores in trusted facilities still does not guarantee trust in the final device [46]. As the advancement in the semiconductor industry has continued to provide smaller technologies, more advanced and sophisticated fabrication facilities are required to achieve economic scales of production. Almost all market leading Integrated Circuit (IC) vendors are currently fabless, which means that the fabrication of their products is carried out overseas by a separate vendor. Different steps of chip manufacturing, such as design, integration, fabrication, and packaging can no longer be completed under the same roof. Hence, original IP owners no longer have complete control over the entire supply chain. In this case, IP vendors and design houses face the threat of piracy, tampering, and IC counterfeiting.

Another source of vulnerabilities is the computer-aided design (CAD) software used to design, test, and validate SOCs [38, 39]. Existing CAD tools can unintentionally introduce vulnerabilities in SoCs because they were not designed with security in mind; instead, they are driven primarily by conventional metrics such as area, timing, power, yield, and testability. Designers who overly rely on these tools can, therefore, fall victim to "lazy engineering" where the design is optimized without being aware of impacts on security. This can result in backdoors through which sensitive information can be leaked (i.e., violation of a confidentiality policy) or an attacker can gain control of a secured system (i.e., violation of integrity policy). For example, finite state machines (FSMs) often contain don't-care conditions in which a transition's next state or output is not specified. A synthesis tool will optimize the design by replacing don't-care conditions with deterministic states and transitions. A vulnerability will be introduced if a protected state (e.g., kernel mode) is illegally accessible by the newly introduced states/transitions [38-42,47].

Finally, many security vulnerabilities can be created unintentionally by design mistakes or a designer's lack of understanding of security problems [38]. Design engineers may not have sufficient knowledge in hardware and information security due to the high complexity of the designs and diversity of security problems. For instance, security is often in direct conflict with the intuition that engineers have developed for IC testing. Design-for-test and design-for-debug infrastructures can themselves provide backdoors if not properly designed.

The problem of IP theft or piracy exists even after the devices are delivered to end-users. A wide variety of deployed electronic devices in consumer, industrial, and military applications are targets of reverse-engineering (RE) in the field. The main motivation behind reverse-engineering

is to obtain access to stored secrets (assets) or deployed IP in the SoCs, which may provide financial benefits to commercial vendors as well as strategic information to competing parties. Deployed and consumed components may be reintroduced back into the supply chain even after their end-of-life. Components can be reclaimed from previously deployed systems by recycling or remarking them. Reused components are often unreliable, and remarked components can be insecure given the nature of the counterfeiting process and the untrusted supply chain, especially when they are used in critical infrastructure. The press regularly publishes dramatic reports of millions of dollars lost due to attacks of this nature, and these challenges are also far from resolution at the current state [4, 5].

Security and trust issues are not only the concerns of the industry for brand reputation and financial losses; the government can be greatly affected as well. Therefore, a great deal of attention must be paid by all to protect the integrity of systems, deployed secrets therein, and IP in electronic systems, as well as for ensuring a secure supply chain to preserve the reliability and trust of these devices.

3.1 Need for Major Initiatives to Address S&T Grand Challenges

In the past decade, hardware security and trust have received significant attention from research groups in government, industry, and academia to identify and address various limitations and challenges. The growing interest in this domain resulted in considerable growth in research and collaborative initiatives. This increased understanding of security and trust problems has been, in part, the result of extensive research and investment. Naturally, based on the explored problems, several security countermeasures and solutions have been proposed and designed by the research community to mitigate the weaknesses of the insecure supply chain and hardware systems.

While making tremendous progress toward feasible solutions to major security problems in the infrastructure lifecycle, such current initiatives will not be sufficient in the foreseeable future. In spite, the significant attention research and development (R&D) resources invested in hardware security and trust research, there is much still required to provide comprehensive solutions for not just existing but emerging challenges. Most importantly, we still lack a clear direction as to what the grand challenges facing the hardware security community are, especially with the current progress and emerging threats. The responses to the security threats remain reactive, rather than focusing on proactive or preventative solutions, and more robust technology. More specifically, there is a lack of support for systematic and interdisciplinary research methodologies. Therefore, the conducted research in the field of hardware security and trust has been ad-hoc. Commonly, research is built on unrealistic assumptions, i.e., inadequate models of attack and defense mechanisms, that do not represent real-world scenarios – which the TAME forum addressed in the past three years of activities.

To address these shortcomings, there is a need to outline grand challenges facing the research community, encourage increased investment in facilities and tools, and facilitate technology transfer – the transition of research results into practice.

3.2 Objectives

The primary goal for this living document is to identify major challenges in the field of security and trust and develop consensus within the community on important research problems as well as the investments which have to be made. To achieve this target, we focus on the following objectives for a complete understanding of the challenges in this domain and taking necessary actions:

1. Identifying the grand challenges that pose the major obstacles in thriving in the domain of electronic hardware security and trust.
2. Developing a comprehensive report that highlights all underlying components in terms of challenges and solutions for providing a roadmap to the community members and used by the stakeholders.

3. Identifying the existing and emerging threats and vulnerabilities for potential research direction for academia and to produce a well-educated and trained workforce for providing security and trust upfront.
4. Recognizing a set of actions to be undertaken by industry for easier technology transfer, workforce integration, tools development, metrics and policy development, and commercialization.
5. Integrating the knowledge and information from academia and industry to provide the government with existing and emerging threats, associated vulnerabilities, and security and trust requirements and practices for national defense and security.

Below, we briefly describe a number of grand challenges in the area of microelectronics S&T:

I. Workforce development and training.
II. University education.
III. Emerging research.
IV. Technology Transfer.
V. Standards and best practices.
VI. Development of electronic design automation (EDA) tools.
VII. Security and trust metrics development.
VIII. Development of policies.

The results from this report are to be used by the major stakeholders in the field, namely academia, industry, and the U.S. government and will allow them to take collaborative actions for ensuring security and trust for microelectronic devices and systems.

3.3 Major Challenges

Establishing strong security and trust in the microelectronic domain requires addressing multi-faceted challenges, as outlined in Section 2. Recognizing these challenges along with respective limitations and potentials, establishing proper roadmaps for academia and industry to identify and undertake necessary action items, and, most importantly, a collaborative effort and knowledge sharing by all members and stakeholders are extremely crucial to make a successful outcome in addressing these challenges.

Detailed scrutiny on these challenges provides interesting insights and executable, both short-term and long-term action items. For example, the current skill of a workforce under consideration and the projected skills to be required for addressing emerging attacks and vulnerabilities often exhibit significant differences between them and, therefore, strong workforce development is required for a comprehensive hardware assurance. Similarly, the development of microelectronic S&T-oriented curriculum for the university education and establishing associated research thrusts are also crucial to ensure a steady flow of the resource and workforce into the domain. Commercialization and transition to practice, along with the development of domain-specific standards and best practices, also aid to this context. Additionally, it is important that security-aware tools and automation process should be developed, both by the industry and academia, to fill any security gap in this domain since

current tools do not adequately support the creation of security subsystems. In other words, the current tools are often incapable of detecting security vulnerabilities. They may also introduce additional vulnerabilities to the designs since they are not developed with security in mind. Another crucial aspect is the development of S&T metrics and assessment schemes as they provide the foundation of CAD tool development and automation, as well as, development of solutions to the ever-growing emerging threats and vulnerabilities. As one can see, addressing these grand challenges require collaborative efforts and multi-directional approaches from academia, industry, and the government.

In the following sections, we provide detailed reports on each grand challenge.

3.3.1 Workforce Development

Microelectronics security and trust is a vital component of modern cybersecurity. Unfortunately, over the past three decades, the researchers in the field of cybersecurity assumed the hardware underlying the information system is secure and trustworthy. However, this notion has been challenged over and over again, especially with recent attacks and breaches. Hence, the developing workforce for microelectronic security fits well within the overall government mission in developing cybersecurity workforce who understand the hardware security challenges, supply chain issues, and more.

It is now widely recognized that there is an extreme demand for cybersecurity professionals and that demand is increasing at a very high rate. The United States government at the highest levels has identified cybersecurity as a national priority. All technologies are increasingly more sophisticated having an experienced, well-qualified workforce has never been more critical to the security of all organizations in both public and private sectors. The government, commercial industry, and academia are all creating new cyber jobs resulting in thousands of open positions for cyber professionals to protect networks and information systems. Microelectronic security, as a major component of cybersecurity, is no exception. There is a demand on a highly educated workforce who understands complex system designs and security.

Understanding the cyber threats, defining the requirements for, and developing a highly-skilled cybersecurity workforce has become such a national priority that the National Initiative for Cybersecurity Education (NICE) partnered with the Federal Chief Information Officer's (CIO) Council to develop and distribute the 2012 Information Technology Workforce Assessment for Cybersecurity (ITWAC). NICE evolved from the Comprehensive National Cybersecurity Initiative (CNCI) Initiative 8- Expand Cyber Education, to develop a technologically skilled and cyber-savvy workforce with the right knowledge and skills.

The ITWAC was developed to collect workforce data that would help identify the composition and capabilities of the federal civilian cybersecurity workforce [6]. The Department of Homeland Security (DHS) has leveraged this work to establish the National Initiative for Cybersecurity Careers and Studies (NICCS) [48]. The vision of NICCS is to provide the nation with the tools and resources necessary to ensure the Nation's workforce has the appropriate training and education in the cybersecurity field. The mission of NICCS is to be a national resource/hub for cybersecurity education, careers, and training and is the premier online resource for cybersecurity training. NICCS connects Government employees, students, educators, and industry with

cybersecurity training providers throughout the Nation [7]. Much effort has gone into the development of these resources, and most of the information presented in this Chapter comes from and is a summary of the expansive work previously performed, documented, and provided to the public to assist all organizations, large and small, to develop and maintain a highly-skilled cybersecurity workforce. While the focus of the TAME Forum is centered on Hardware Assurance (HwA), the information and processes provided here and at the referenced on-line websites, may be tailored to define the needs for the HwA workforce. In fact, where applicable, this chapter is focused on the development of the HwA workforce.

3.3.1.1 Hardware Cybersecurity Workforce Development

This section will focus on a challenge addressing two major thrust areas:

i. Identifying the HwA workforce skill requirements.
ii. Developing the training plan and identify the resources to educate, train, recruit, and retain the workforce with specific HwA cybersecurity skills.

3.3.1.1.1 Identifying HwA Workforce Skill Requirements

In order to protect and defend their critical assets such as information, systems, and networks from unauthorized access, one must have the right people, with the right skills, in the right place to address these issues. Organizations must understand the threats for which they are most vulnerable and develop and/or hire a workforce, which is highly skills in these areas. That is, one must customize their cybersecurity workforce. As mentioned previously, the DHS has taken the lead to strengthen the national workforce to make sure the government, industry, and academia have the tools and information needed to protect their organization and meet challenges in the future.

As outlined in [8], the DHS has the programs and capabilities to help an organization build a world-class cyber workforce:

- Identify and quantify the cybersecurity workforce.
- Understand workforce needs and skills gaps.
- Hire the right people for clearly defined roles.
- Enhance employee skills with training and professional development.
- Create programs and experiences to retain top talent.

These steps are depicted in the Workforce Development Lifecycle below [8]. Speaking broadly, the following guidance is given below.

3.3.1.1.2 How to Plan for the Cybersecurity Team

Planning for the development of a cybersecurity team starts with the following objectives and outcomes:

1. Determining the cybersecurity risk exposure and risk tolerance.
2. Inventorying the cybersecurity workforce.
3. Determining and addressing cybersecurity workforce gaps.

The DHS provides many interactive tools to help determine the information needed to support effective workforce planning for an organization. However, these tools are tailored towards the IT Cybersecurity expert. Therefore, the challenge to the TAME community is to address these tasks as they relate to Hardware Assurance.

3.3.1.2 Develop a Training Plan for the HwA Cybersecurity Workforce

Once an organization has analyzed the cybersecurity workforce needs, it can begin to develop a plan to educate, train, recruit, and retain a skilled workforce. This plan should be customized for specific organizational needs. Again, the DHS has provided a framework to assist in the planning process. Additionally, there has been significant activity in establishing the educational opportunities to train and develop this workforce. This includes K-12 outreach, certification programs, as well as two- and four-year college degree programs. The tools and information developed at the National Level are maintained and offered through the DHS and can be found at the sites referenced in this document. Brief summaries, taken from some of the referenced sites of some of the tools offered and educational opportunities are provided below.

3.3.1.2.1 NICE Cybersecurity Workforce Framework

The National Initiative for Cybersecurity Education (NICE) Cybersecurity Workforce Framework (NICE Framework) has an interactive tool to assist in Workforce Development training planning. The NICE Framework provides a blueprint to categorize, organize, and describe cybersecurity work into Categories, Specialty Areas, Work Roles, tasks, and knowledge, skills, and abilities (KSAs). The NICE Framework provides a common language to speak about cybersecurity roles and jobs.

3.3.1.2.2 Categories, Specialty Areas, Work Roles, and Capability Indicators

Within the NICE Framework, there are seven Categories, each comprising of several Specialty Areas. Additionally, within each Specialty Area, there are Work Roles. Each Work Role includes the tasks required to perform the role, as well as the KSAs required to perform those tasks. Additionally, Capability Indicators have been added to the NICE Cybersecurity Workforce Framework on the NICCS website. The Capability Indicators are a combination of education, certification, training, experiential learning, and continuous learning attributes that could indicate a greater likelihood in an individual's ability to perform a given Work Role. This organizing structure is based on extensive job analysis that group together work and activities that share common functions, regardless of job titles or other occupational terms.

The NICE Framework assists the user to establish the skills necessary to perform Specialty Area functions using a top-down approach, and also supports a bottom-up approach where one can use keywords to identify tasks, skills, knowledge, or abilities based on the keywords entered. Users can also link back to the relevant Work Role, Work ID, Category or Specialty Area for that search.

3.3.1.2.3 National Centers for Academic Excellence

The Department of Homeland Security (DHS) and the National Security Agency (NSA) jointly sponsor the National Centers of Academic Excellence (CAE) program. Over 200 top colleges and universities across 47 states, the District of Columbia, and the Commonwealth of Puerto Rico are designated as CAEs in Cybersecurity.

The website [49] lists specific two- and four-year colleges and universities designated as a CAE based on their robust degree programs and close alignment to specific cybersecurity-related knowledge units (KUs), validated by top subject matter experts in the field [9].

All regionally accredited two-year, four-year, and graduate-level institutions in the United States can apply for designation as an NSA/DHS CAE-CD. CAE designation is valid for five academic years, after which the school must successfully reapply in order to retain its CAE designation.

3.3.1.2.4 NICCS Education and Training Catalog

The DHS website also hosts the NICCS Education and Training Catalog. It provides a central location where cybersecurity professionals across the nation can find over 3,000 cybersecurity-related courses. Anyone can use the interactive map and filters to search for courses offered in their local area so they can add to their skill set, increase their level of expertise, earn a certification, or even transition into a new career.

All of the courses are aligned to the specialty areas of the National Cybersecurity Workforce Framework. Any organizations or academic institutions interested in listing courses may apply to have their courses included [10].

3.3.1.3 Challenge

So, what is the challenge? While researching material to include in this Chapter on Workforce Development, it became clear we are not the first organization to address this challenge. However, by far, the work performed by these National Initiatives to provided to the national community through the DHS website is the most comprehensive set of tools and best practices available. It seems apparent that millions of dollars and thousands of hours, directed by experienced Subject Matter Experts, have been expended to create this very valuable resource.

The challenge here is not to recreate the work that has been performed here, but rather, engage and collaborate with these institutions to expand to include the work roles and topics important to the HwA Cybersecurity field. This includes expanding the KSA to include the key skills and abilities one would require of HwA experts in the field and include hardware security-oriented educational opportunities needed to pursue a career in HwA. The framework and best practices are here for this community to leverage. The challenge is for this community to step up and define the necessary information, structured in a manner, to be incorporated within the NICE Framework.

3.3.2 University Education

Security and trust need to be integrated at all levels of the curriculum as first-class citizens in Electrical Engineering (EE), Computer Engineering (CmpE) and Computer Science (CS) degree programs. Similar to how low-power design in EE and CmpE moved from an optional topic twenty years ago to a required part of the curriculum today in the vast majority of programs in

the United States and many programs worldwide, so also do security and trust need to become a required part of the curriculum.

We will focus in this section on undergraduate education as graduate coursework tends to be easier to modify and change primarily due to far fewer constraints in terms of strict course sequence requirements as well as greater frequency and ease of introducing new coursework and topics.

The following topics should be considered to be present in the undergraduate coursework.

3.3.2.1 Authentication and Encryption

Central to any approach to security and trust in microelectronics are authentication and encryption. Authentication is required in order to make sure that entities communicating in a network are who they say they are. Encryption is required to prevent adversaries from discovering useful information just by eavesdropping. Without the availability of both authentication and encryption, many known attack vectors simply cannot be stopped. For example, a traditional man-in-the-middle (MiTM) attack relies on failed authentication or lack of a proper authentication sequence in a communication protocol.

Once it is agreed that authentication and encryption are key topics for inclusion in the curriculum, the question becomes how to integrate these topics.

3.3.2.2 Additional Topics

Of course, many additional topics merit inclusion. Chief among these for microelectronics hardware are side-channels including power analysis, timing and radio-frequency (RF) information leakage, fault injection, malicious change, counterfeit, etc. Power analysis, in particular, has a stunning array of powerful techniques such as simple power analysis (SPA) and differential power analysis (DPA), including second-, third- and higher-order techniques.

So-called physically unclonable functions or PUFs are now making their way into product lines across the semiconductor industry. A firm understanding of their various forms and instantiations on a chip, as well as their physical sources of entropy (i.e., randomness) are critical to their success or failure.

3.3.2.3 Concrete Steps

3.3.2.3.1 Update Existing Courses

The first step is to review existing course sequences and modify course syllabi appropriately. For example, when teaching computer architecture topics and caching, the use of caches as a timing side-channel could be added as a syllabus topic. In the class, a brief description of attacks such as the Meltdown attack can be a way to teach an example of this side timing channel.

3.3.2.3.2 Modify Core Course Sequences

After the first step of updating existing courses, the second step is to modify the existing course sequences to integrate security and trust better. As with any curricular change, this modification is likely to involve new course creation with the corresponding possible necessity of removing one or more courses in order not to overload the degree program with required coursework.

For example, an early (e.g., taught in the second or sophomore year of the undergraduate degree) course in security and trust concepts might meet this requirement.

3.3.2.3.3 Add Electives

A third and final step to modify the curriculum is to add elective courses. However, the business practices in most universities make this step the easiest to carry out; as a result, the temptation can be to perform this step first.

3.3.3 Emerging Research

In this section, we present a number of emerging research challenges in the field of microelectronics security and trust.

3.3.3.1 Role of Machine Learning in Hardware Security and Trust

The rise of applied machine learning inspires research with efficient and innovative solutions for different types of problems. Hardware security is not an exception! For example,
the ability of deep neural networks (DNNS) to capture complex input-output relationships makes them a very promising technology for hardware security.

In [11], researchers proposed a taxonomy of how machine learning is utilized in hardware security. According to this taxonomy, machine learning can be used for protecting and attacking hardware components. In protection, machine learning is widely applied for detecting hardware Trojans and malicious hardware activities. For example, presented research in [12] used an on-chip neural network that can be trained to detect untrusted chip activities that are performed by inserted Trojans.

Another application where machine learning can be applied is the detection of IC counterfeiting, such as in [13] and [14]. Researchers in [13] used machine learning to extract features from IC package images and trained their machine learning system to detect common defects, such as scratches, in counterfeited ICs. Work in [14] went further and constructed 3D images of the ICs using nondestructive X-ray based imaging, and trained their machine learning to detect die-face delamination in counterfeited ICs.

As the cat and mouse game continues between hardware security research and adversaries, researchers should keep an eye on applying machine learning in hardware attacks. For example, in side-channel attacks, an adversary collects statistical performance data such as power consumption and/or electromagnetic interference emitted by the chip. Machine learning can be applied to quickly and efficiently extract information from these gathered statistical data and help in revealing sensitive information, such as cryptographic keys. In general, side-channel attacks contains two phases: identification and exploitation.

The identification phase aims to define the type of leakage and building models to use it. In the exploitation phase, the adversary uses the developed models to reveal sensitive information. In this phase, the accuracy of the model greatly affects the success and needed time to compromise the system. Machine learning can be applied in this phase to increase the chance of the adversary breaching the system and to reduce the time required to complete the attack. In [15], researchers conducted analysis for different machine learning techniques, which can be used in a side-channel analysis application. It is clear that machine learning presents both opportunities and threats to hardware security as the body of research in this domain further being developed.

3.3.3.2 Quantum Leaps in Security

Quantum computing is expected to change the paradigms in processing, networks, and for sure, in security. Recently, researchers in University College London developed a secure method to communicate between multiple quantum devices. In their work [16], they proposed an approach that secures communication without the need to trust the manufacturer of the devices. "Security is ensured by the unique property that quantum correlations can only be shared between the devices that created them, ensuring no hacker can ever come to learn the key" [17].

On another dimension of using quantum physics for hardware security, NIST scientists showed in theory how the laws of quantum physics could allow for the construction of one-shot memory [18]. Quantum Key Distribution (QKD) is a way to produce and share random secret key between two communication parties where they can detect the presence of any third party that is trying to gain knowledge of the key [19]. Unconditional security of QKD is theoretically proven in many research works, such as in [20]. Discussion in [21] highlights different practical challenges that need to be addressed before being able to adopt this technique widely.

3.3.3.3 Security Implications of Emerging Devices

Emerging non-CMOS technologies is another promising direction where their unique physical properties can be leveraged to protect the hardware. IP protection, Physically Unclonable Functions (PUFs), and True Random Number Generators (TRNGs) are examples of potential applications.

Researchers in [22] analyzed three different circuit structures, including camouflaging gates, polymorphic gates, and power regulators using the interesting properties of silicon nanowire FETs and graphene Sym FET. A detailed discussion on how the unique I-V characteristics of post-CMOS transistors can be used in hardware security is presented in work in [23]. Phase-change memory and their stochastic variability at reset, Carbon Nanotube, and their manufacturing process

dependability can be used as a source of randomness in applications such as cryptography and random key generation [24].

3.3.3.4 Heterogeneous Integration in 3D/2.5D

3D/2.5D processes are another new technology that provides opportunities as well as concerns in hardware security. For example, 3D integration uses split-fabrication as a normal practice of its process. This adds an advantage from the security point of view through preventing potential IP theft, and chip overbuilding during the outsourced fabrication [25]. 3D stacking offers a natural defense against threats such as side-channel attacks and reverse engineering. In addition, with 3D heterogeneous technology integration, emerging non-CMOS security primitives can be integrated effectively into CMOS systems. Functional obfuscation can also benefit from 3D integration by hiding gates/wires in trusted-manufacturer dies which can be used to obfuscate the functionality of the dies fabricated at untrusted parties. Similarly, 3D integration presents some security concerns such as counterfeiting testing, and Trojan detection.

3.3.3.5 Hardware Trust Issues

Trust issues in microelectronics arising from the untrusted foundry and untrusted IPs, CAD tools, as well as design/verification stages, will create new challenges in trust verification as well as trusted design with untrusted components. Analysis of trust issues at the system level due to untrusted IP and tools will remain a significant problem. Particularly, design modifications, as well as apparently benign design artifacts (such as design for test/debug artifacts) exploited by software, will be an interesting and significant problem for researchers. Trust issues at the printed circuit board (PCB) level due to Trojans inserted either during the design or fabrication stage or in an untrusted supply chain, are an emerging area of trusted hardware design and verification.

3.3.3.6 CAD for Security and Trust

With the need to address security and issues for complex designs, the development of appropriate algorithms and tools for electronic design automation (EDA) would be mandatory. Automatic security synthesis that relies on a "secure by design" paradigm and an automatic security/trust verification framework can be a game-changer in the electronics industry since it alleviates the need for design/test engineers to have complete domain knowledge about security and trust issues and ways to effectively address them. Given the diversity of threat models that vary based on application and usage scenarios of a system and level of hardware abstractions, it is likely that a suite of such tools needs to be developed to meet the industry needs. Automatic CAD tools would essentially need to integrate appropriate security and trust metrics into their flow.

3.3.3.7 Hardware Security at Higher Levels of Abstractions

While the majority of hardware security research has focused on microelectronics, in particular, integrated circuit level security, complex security issues at the higher-level abstractions (e.g., printed circuit board, the network of heterogeneous components in a system of systems, such as an autonomous vehicle) emerge. These issues arise from the interaction of various components in the systems, both hardware and software, with varying levels of security to access and share common resources and security-critical assets across them. For example, in an automotive system, an infotainment component may need to use the same bus as the brake and steering control, hence

requiring appropriate access control protection. There will be a growing need to develop methodologies, architecture, and tools to enable efficient security design end-to-end verification, and flexibility to update security requirements during field operation.

3.3.4 Commercialization and Transition to Practice

Among the myriad of technical roadblocks that exist in the world of hardware security and trust, challenges exist in the effort to commercialize security solutions to a wide customer audience that is typically not considered at the research and development stages of security. The ideas of value proposition and marketing are not on top of a researcher's mind but become critical as attempts are made at commercializing any security solution. Also, challenges exist in transitioning a theoretical security solution into a product or service that has broad applicability and practicality, for several different reasons. In this section, we will outline the challenges in both commercialization and transitioning a security solution to practice, as well as provide guidance and suggestions to overcome these challenges and step towards more secure hardware being deployed.

3.3.4.1 Commercialization

Cybersecurity was a $151B market in 2018 and has grown rapidly year after year [11]. In the wake of Meltdown, Spectre, Foreshadow, and related hardware security attacks that were both financially costly and reputation damaging, one might imagine that the commercialization of hardware security solutions would be a trivial task. Unfortunately, there are many challenges in the commercialization process. One of the more pervasive challenges is the fact that many hardware vendors want to have a secure product but do not know how to justify its value and subsequently, its cost. Adding security to a system is intangible and abstract, as it is something that is necessary but is often not viewed as an additional feature or capability that the customer can utilize. Weak security is detrimental; of course, if there are security vulnerabilities that do not allow a customer to utilize their product fully, that is a showstopper. However, strong security is not something that enhances a product. It is simply seen as a necessary feature of the product. The end customer's experience and the utility of the product are typically not enhanced with strong security. With these customer perceptions being true for many organizations, it is hard to motivate hardware vendors to move forward with the purchase of additional security solutions, especially when they have solutions already. Some hardware vendors are more proactive, but this is not the case across the board. Without security being an enhancement rather than an avoidance of negative outcomes, security will be a hard sell.

Additionally, when attempting to commercialize a security solution, hardware vendors tend to desire a complete security solution, meaning that a solution must completely solve a given threat model or attack surface before it is considered for purchase. As an example, a hardware vendor may only need a single place and route tool to perform place and route. That place and route tool can be compared to other competing tools for runtime and quality, and a clear winner can be chosen. Unfortunately, security is a much more intractable problem that includes many different types of threats and malicious actors that need to be kept up with and combatted constantly. Also, security quality is hard to quantify, which makes it difficult to compare the utility of competing tools and IP. This sort of stalemate creates a situation where hardware vendors may be less likely to purchase and integrate security solutions because of the difficulty in knowing which solutions will solve their problems. As a result of that stalemate, hardware vendors may hesitate to move the needle in the right direction, even if it is the most beneficial thing to do.

Some security is strictly better than no security. In an attempt to create a complete solution, hardware vendors may purchase from several different security solution vendors, which makes the end solution more expensive and harder to coordinate. A higher expense will always slow down adoption.

Another clear challenge is that the hardware industry tends to be reactive rather than proactive when it comes to security (which is not unique to the hardware industry). Hardware vendors are much more quick to act when an exploit has been found on their hardware rather than looking for potential threats to patch proactively. It is difficult to pinpoint why this is the case, but it can be attributed to the idea that a security vulnerability only really becomes a threat once someone exploits it. The real "value-makers" are the hackers who expose problems and force designers to react, rather than designers being proactive in attempting to cover all holes before they are exposed.

One more challenging paradigm is that everyone tends to think that security is another person's problem. Even within the hardware world, oftentimes, a hardware design team will rely on security that is provided by IP cores that are provided by a third party and believe it is sufficient to lean on these IP cores for security rather than build their entire design to be secure from the ground up. While adding a security feature such as encryption or secure enclaves are necessary and can definitely help security, the security of the whole system needs to be tested and verified. This idea is sometimes lost of the hardware designer that wants to drop in a feature or core from another company's toolkit to eradicate security concerns.

An additional problem is that as an industry we need to explain and quantify the business case for implementing security. As one can extract from the previous paragraphs, this business case is complex, has many aspects, is hence not immediately clear, and can not be precisely computed. Many of the costs related to a successful attack are hidden and even its direct impact is often hard to quantify. Some recent analysis estimates a lower bound of the costs in case of a breach, and this in itself already shows that implementing the right countermeasures is an order of magnitude lower cost than fixing all aspects of a breach. A lot of work has to be done; however to make the total financial impact completely clear and make the purchase and implementation of security solutions an imperative.

Lastly, a large part of commercialization is building awareness about the problems and potential solutions. This can be challenging to educate the masses on the ever-growing field of security. New attacks and potential mitigations are released on a monthly basis through conferences, blogs, and other outlets. It can be very daunting for a hardware vendor and its designers to keep track of and digest all of these new findings. Also, categorizing the magnitude of the vulnerability and its likelihood of exploit is significantly difficult. Without these metrics, it becomes hard to know what to protect against and what can be ignored for the time being.

3.3.4.2 Transition to Practice

When attempting to create a commercial solution, the theory that is created in academia must be realized into an actual product that works with many practical challenges in the way. Also referred to as "implementation issues." Furthermore, the costs of the solutions have to be low, the solution has to fit within the resource constraints, it has to be fast (security cannot get in the way of the

actual application), and of course, it has to be perfectly secure. Clearly, in such a situation, trade-offs have to be made, and it is the art to make the correct trade-offs.

3.3.4.3 Suggestions

On the commercialization and adoption front, hardware vendors need to understand that security is an iterative process and that moving the needle towards a more secure product is a valiant effort, even if the final product is not 100% secure against all attacks. Also, the inclusion of security in end-products can absolutely be used as a product differentiator and a sales enabler if marketed the proper way. In this case, security can be used to enhance a product and increase a company's bottom line rather than hurt it.

To help with the problem of keeping track of the state of hardware security, hardware vendors can hire security architects or security researchers whose sole purpose is to stay up to date on the latest attacks and mitigations, and use their expertise to draw a line in the sand as to what needs to be protected and what can be skipped. Even a single security person can really demystify what needs to be done.

Overall, the hardware industry is moving in the right direction with lots of attention on hardware security garnered by many of the recently published attacks. With this momentum, the industry needs to re-evaluate the way security design and verification is done such that the efforts are proactive and iterative towards a more secure product, rather than the cat and mouse game that has been the norm over the past few decades.

3.3.5 Standards and Best Practices

Standards in the area of trusted and assured electronics is a broad area of active development across multiple fronts. The challenges of standardizing such practices for hardware are necessarily somewhat more complex, requiring deep collaboration between government, industry, and academia as well as between designers, fabs, packaging houses and system integrators. Robust practice for verification, validation and assurance of microelectronics and electronic systems require a combination of multimodal measurement hardware, with calibration data, signature data and others. In many cases, outcomes of these measurements are statistical in nature and therefore require risk frameworks that propagate inherent uncertainty throughout the assessment process are required.

In the late 1990's, security researchers started developing tools designed to detect, identify, and prevent software vulnerabilities. Unfortunately, during this period it was difficult to compare different researchers' tools or to compile information on a specific vulnerability because different researchers would use different terminology to refer to the same vulnerability. These mismatches in terminology resulted in significant duplication of effort and could have resulted in gaps in security coverage. In order to address this issue, MITRE released their Common Vulnerability Enumeration (CVE) framework in 1999. This framework standardized the language researchers used when discussing vulnerabilities, and helped to coordinate business efforts. In the proceeding years, MITRE went on to release standards for weakness enumeration (CWE), vulnerability scoring (CVSS), and attack pattern enumeration (CAPEC). Ontologies supporting frameworks must be modified to accommodate hardware centric weaknesses and vulnerabilities.

Standard approaches are required for specifying trust, assurance, threats, and risk. The security domain is exceptionally broad, spanning physical, functional, and behavioral domains. Current industry standard processes for identifying and mitigating risk at specific levels in electronics systems architecture requires abstraction. Ideally, a standardized verification process (a set of widely accepted best practices) would be developed integrating processes end-to-end across the electronic system lifecycle. Furthermore, the highest functional standard will take advantage of the quantitative natures of many assurance technologies, being able to specify calculated levels of trust, assurance and risk, providing a standard threat model.

3.3.5.1 Current Applicable StandardsSAE AS6171 outlines a standardized process for assessing the authenticity of electronic hardware. It utilizes a progression of knowledge acquisition and outcomes of various tests and then uses a Bayesian approach for combining estimates of the outcomes. However, like most risk frameworks, the approach utilizes binary answers of expert analysis and then functions as a means to weight multiple measurements in a decision process. Emerging second-order effects verification techniques require standards aware of their statistical nature which propagate uncertainty throughout the process. NIST SP800-53 outlines a series of controls (often referred to as the risk management framework or RMF) that has recently reshaped state, federal and large business information system security standards. RMF offers a robust process for access controls and data protection schemes for traditional information technology systems and networks. Revision and update of SP800-53 incorporating HwA techniques require further investigation.

The IEEE Standard for System, Software, and Hardware Verification and Validation (V&V) 1012-2012 (Revision of IEEE Std 1012-2004) formalizes the process of creating and defining verification and validation plans while providing a structure that facilitates concise and shareable process descriptions. This IEEE standard also establishes a minimum set of verification activities that should be performed to establish trust in a system. It also proposes a nested "system of systems" model that allows V&V principles and practices to be recursively applied to complex systems with computationally intractable numbers of potential states. While the IEEE standard identifies four targeted levels of system integrity and proposes minimum sets of activities required to achieve each level of integrity, it does not provide a framework for assessing the level of trust provided by a non-standard set of V&V activities.

The International Organization for Standardization (ISO) has released an enormous number of standards that relate to information security, many of which can be applied to hardware. These standards are important because they structure the different processes involved in producing IT products, and create checklists that can provide various levels of system trust when diligently completed. ISO standards cover a wide range of activities related to establishing trust. Some areas of inter-organizational coordination are described in multiple ISO standards. ISO/IEC 20243-1, 20243-2, and 20243-3 all cover the steps an organization should take to establish a trusted relationship with suppliers. ISO/IEC 29147 and 30111 cover best practices for handling and disclosing discovered vulnerabilities. Other ISO standards explore specific technological areas. For example, ISO 18257, ISO/IEC 17825, and ISO/IEC TS 30104 all cover processes used to build trust that specific properties are established. Then there are many ISO standards that outline broader types of problems, like ISO 22381, which discusses interoperable identification systems and ISO/IEC 20243-1, which describes a regimented approach to avoiding tainted products.

Although developing standards are highly recommended, the team also recommends ensuring the experts in the field are included, and the standard is made available to the public for potential assessment prior to being finalized for use in the field. As an example, the IEEE IP encryption standard (IEEE-P1735) was developed to protect the confidentiality of IPs and has been widely adopted by the semiconductor IP industry. However, limitations and vulnerabilities of these approaches have been highlighted in a recent CCS paper published by researchers from the University of Florida [3]. The team demonstrated the weaknesses in the IEEE-P1735 standard can be exploited to extract the entire RTL plaintext without the knowledge of the key [43].

3.3.6 Electronic Design Automation (EDA)

Although product security receives much attention these days, there are still significant gaps in security-aware toolsets and frameworks. Security design, validation engineers, and researchers in industry, academia, and government do not have access to a mature, security-aware toolset. Such a toolset would ideally analyze designs for various types of security vulnerabilities at different levels of abstraction to detect and fix the security issues or build security into designs efficiently

and easily from the offset. Despite such a demand, there does not exist an ecosystem of security-aware Electronic Design Automation (EDA), Computer-Aided Design (CAD) tools, since the commercial market for such security and validation tools are still in their infancy.

There are many research works that attempt to establish security analysis engines and provide solutions to address different classes of security issues. Such issues include data leakage, access control violation attacks, side-channel attacks, physical attacks, fault injection attacks, hardware Trojan insertions, physical tampering, reverse engineering, and chip counterfeiting. This section enumerates some of the challenges to realizing an ecosystem of security-aware CAD tools. It will also give an overview of a few recent attempts to address some of these challenges.

3.3.6.1 Problem Identification

There is a growing trend of placing more reliance on daily life on computational systems. This trend can be seen in the growing number of systems in domains such as finance, health, defense, and utility. This growth makes the security, integrity, and privacy of these systems even more critical. Users and operators seek trust, reliability, security, safety, and privacy-aware operation. Many of these systems have numerous parallel, concurrent, and distributed sub-systems interacting with each other and are typically built as a network of heterogeneous nodes. These systems can be edge nodes, gateways, bridges, routers, and servers, as the Internet of Things (IoT) era introduces typically isolated systems to the network.

Regardless of the type of nodes, all such systems comprise of components built as a various combination of hardware-, firmware-, and software-level subsystems to provide their intended functional capabilities and services. Hence, a security subsystem also needs to be built into many of these components, as well as their communications. The proper support for a security subsystem must be incorporated into a typical modern computing system product life cycle (PLC) [28], which typically includes planning, design, development, validation, manufacturing, testing, and support steps.

Although industries have already started adopting various Security Design Lifecycles (SDL) [28] and integrating those into the PLC with integrated EDA tools that also provide basic security services, the advanced support for comprehensive coverage through all the steps and levels has not been established yet. Formal security validation tools, for example, address the needs of some of the steps and abstraction levels, yet most of the tools currently used throughout the PLC are not security-aware. Hence, a new generation of EDA tools incorporating novel scalable approaches and methods is necessary. This next generation of tools must provide the level of security needed to be built into products, as well as the required level of security assurance. The tools should leverage the speed and accuracy possible by automation while having a minimum impact on time-to-market, cost, and efficiency for products.

Current tools do not adequately support the creation of security subsystems. It is, therefore, incumbent upon both industry and academia to fill this security gap through further research and development to bring to life a new generation of security-aware tools. The security-aware toolset should incorporate security through all levels and all phases of the product life cycle, including design and development and validation, testing, and verification of the following:

I. specification

II. architectural behavioral system model, including instruction set architectural models and the corresponding software/firmware which runs on it
III. register transfer model, covering micro-architectural models for hardware
IV. logical (gate or cell) level modelling
V. physical-level designs, such as placed-and-routed cell-level models augmented with clock, reset, and power networks, multi-layer layouts
VI. transistor-, device-, and circuit-level models
VII. manufactured silicon dies
VIII. packaged single- or multi-die chips

3.3.6.2 Potential research topics

3.3.6.2.1 Security specification, analysis, and validation [Spec.]

Research for creating extensible and open threat modeling paradigms that allow continuous and collaborative model updates as the new groups of threats have emerged.

3.3.6.2.2 Security architecture modelling, analysis, and validation [Arch.]

Wizards are needed to use the security specifications to generate or guide in the generation of alternative architectures for components of security subsystems and their communications as well as the corresponding interconnects and protocols for their communications. The assets should be somehow designated or tagged in the architectural model with the type of their security requirements for each of its interfaces. This makes it possible that the high-level synthesis tools preserve the security properties of assets and to its register-transfer-level design as well as validation of the architecture against the security requirements. Moreover, tools are needed for automatic and semi-automatic analysis of the architectures taking into account the level of security of each alternative versus other parameters such as estimated area, power, performance, and cost of lower-level implementations, validation, manufacturing, packaging, and testing. The analysis tools should generate reports and create the corresponding visualization aids to guide refinement of security architecture to reach the desired level of security. The tool should generate validation infrastructure and tests to make sure the chosen security architecture is valid against the security model defined in the specification and even robust considering lower-level implementation options if there exist lower-level requirements (e.g., requirements for physical attacks that could make an architecture unsuitable).

3.3.6.2.3 Design, analysis, and validation at register transfer level

Security-aware design and analysis frameworks are needed at register transfer level as well to make it possible to add security as a parameter for optimization in addition to the area, power, and performance parameters at this level to enable multi-objective optimizations. The security implemented in a design should be quantifiable as a continuous or discrete value to make it possible to measure the security level and compare it with the security level of the alternative designs. Hence, appropriate methods and models are required to assign a security level or value to a design. Since there are different solutions for various types of attacks such as software attacks, remote attacks, and physical attacks, including side-channel attacks (power, electromagnetic, etc.) and fault injection attacks (voltage, frequency, temperature, power, etc.). At this level, validation of security requirements can be done by simulation, emulation, and formal engines, while there is a

big need for automatic generation of the security tests based on the requirements. However, the increasing complexity and size of designs makes the usually exponential or even polynomial validation execution time prohibitive for exhaustive coverage by simulation and emulation tools whereas the formal engines suffer from the limitation of the size of the designs supported although they can use several solver engines in parallel each using its own set of heuristics to make the execution times manageable. Hence, novel scalable methods and algorithms are needed for analysis and verification engines. Moreover, multi-target optimization using security as an optimization parameter can be too expensive to be practical as security analysis engines should be used within the feedback loops of the optimizers.

3.3.6.2.4 Design, analysis, and validation at gate/cell level [Gate Level]

Methods are needed to transform gate lists by mapping the security sensitive subsystem to side-channel attack resistant (e.g., dual-rail logic cells), voltage attack resistant (e.g., cells with thick-oxide or stacked transistors), or reverse engineering resistant (e.g., by logic encryption and obfuscation). Analysis of gate-level models can make it possible to measure its level of vulnerability to side-channel and fault-injection attacks to detect Trojans inserted at gate level (e.g., with their triggering difficult to discover due to don't care states of logic optimized finite state machines or rare input combination or sequences). Although simulation- and emulation-based as well as formal security validation and automatic test (and testbench) generation can still be used for gate-level models, as the size of gate (cell) level models increases, tools for security validation face the same limitation of the size of the designs supported for exhaustive coverage as of the register transfer model. Gate-level models can also be utilized for quick and moderately accurate analysis for timing and power attacks as gate-level simulation can be used for a rough estimation of performance (timing) and dynamic power. In contrast, lower-level models provide more accurate results at the expense of higher cost and time. While the estimation of dynamic power consumption can be used for analysis and comparison of alternative mitigations for differential power attacks, the best timing estimation at this level uses Standard Delay Format (SDF) [29], but it is not as accurate as circuit-level delay to be used for the analysis of frequency attack.

3.3.6.2.5 Design, analysis and validation at physical level [Cell-Level]

At the physical level, layout information of the design is available, which captures the connectivity network as well as the 3-dimensional physical locations of all active and passive elements, wires, and isolators. Hence, the layout information makes it possible to determine physical distances among all of them and the design tools can use the information for making designs resistant to certain types of attacks such as tamper- and reverse-engineering- resistant, which are determined by threat analysis. Analysis tools are needed for trade-off analysis among different layout-level modification options for mitigation of attacks as well as validation of the selected option(s).

3.3.6.2.6 Design, analysis and validation at transistor-, device-, and circuit level model

Because a gate-level model is mapped into its equivalent cell-level model by technology mapping while each cell has already included the mapping to transistor-level models, a transistor-level model netlist is actually generated from the layout (by extracting the devices and their connections). Hence, a layout can be used to verify if a transistor-level model is equivalent to its cell-level model with respect to the logic implemented. Incorporating parasitics (modeled by the

network of resistors, capacitors, and inductors) into a transistor-level netlist makes the model accurate enough for timing analysis. Compact timing and power models for cells and interconnects can be generated using Spice simulation to speed up timing and power analysis by abstracting them at a higher level than the circuit level. Commercial timing and power analysis tools use such models for fast analysis with minimal impact on accuracy in comparison with circuit-level model accuracy because the heavy speed and memory requirements of circuit-level simulations make circuit-level simulation at system-level impractical. Design and analysis tools are needed to efficiently analyze the effects physical attacks such as (1) fault injection attacks by changing in voltage, frequency, and temperature of operation or applying laser beams or electromagnetic interference and (2) side-channel attacks based on optical, electromagnetic, electric (probing attacks), and power-based observations. While all can be theoretically done using circuit and electromagnetic simulators, the simulation time and cost is prohibitive due to the enormous size of the required circuit level model. Hence methods for efficient and intelligent hybrid-level simulations are highly desired. The EDA tools supporting such analysis are needed to provide guidance to users regarding the weakness of each design with respect to the list of attacks and providing feedback for mitigation to design tools at the same level (e.g., layout modification) or higher levels to mitigate the attacks. There should be analysis tools to be used for validation of the modified design to implement the mitigations to each of the selected attacks.

3.3.6.2.7 Design, analysis, and validation at chip level

Tools are needed to analyze the dies for possible fault injection, side-channel, tempering, reverse engineering, and Trojan attacks. Although it is possible to create automatic testbenches for experiments to help with analyzing each category of attack, it is highly desired to have general frameworks to be customized and/or configured for each die. Tools should be able to create 2-d and 3-d models (i.e., a 2-d image or a series of 2-d images to create 3-d images) of the electromagnetic, temperature, other parameters emitted from chips for further analysis. Efficient image processing methods should be devised and/or invented to abstract the information gathered for further processing to determine the level of vulnerability of the design to each of the attacks. Corresponding Computer-Aided Design (CAD) and/or Computer-Aided Engineering tools (CAE) tools should be created to make it possible to perform all the necessary analysis and guide the design tools at the same level or higher levels how to mitigate the threats if such tools are available. The creation of threat mitigation CAD tools is also necessary to use the feedback from the analysis engine. Creating methodologies on how to optimally use the tools in feedback loops of optimization tools (incorporating multi-target optimization engines) for all levels of the design will be required. The research area can be extended by considering the possibility of digital systems with integrated analog, optical, and micromechanical systems (MEMs).

3.3.6.2.8 Design, analysis, and validation at package level

Tools similar to that of (g) are needed to support packaged chips. Conceptually, the tools should have similar functionality with some enhancements regarding supporting packaging, pins, wiring among chips in a multi-chip design, and specific packaging technologies like active or passive metal shields embedded in packages or wires passing through packaging material to connect chips. Considering the effect of coexistence of advance techniques in packaging such as hydraulic cooling can be new venues for research.

3.3.6.3 Known Research Activities

There are numerous research efforts targeting various aspects of security. However, they are often dispersed in focus and not targeted for use in a general and comprehensive solution. While it is not possible to list all the relevant works here due to limited space, we encourage readers to visit the TrustHub Vulnerability DB CAD Solution [30] to view submitted work supporting the General Security Design, Analysis, and Validation Framework.

This framework is an attempt to demonstrate the feasibility of this approach in a limited scope for supporting SDL through PLC for a subset of the aforementioned attack types. Due to limited resources, a two-phase approach has been used to realize the framework. In the first phase, a general taxonomy of physical attacks [31] has been created along with available CAD solutions [32] from industry and academia. These attacks have been shared with security practitioners and researchers as guidance on TrustHub. In the second phase, a limited-scale framework was created with limited features through collaborative research among the authors with security researchers at the Florida Institute of Cyber Security (FICS) [33]. The goal of the second phase is to realize the framework is general and extensible with the collaborative support of many researchers from academia and industry. The extensible framework allows the addition of plug-ins to offer additional features and functionality. Plug-ins are expected to be created by industrial and academic partners.

For instance, the supported features can be in one of the following categories: (1) security design, analysis, and validation engines and (2) theories, methods, etc. to create such engines, and (3) research proposals to create theory, methods, etc. to make specific engines to realize the novel security primitives of the deliverables.

We anticipate that this ecosystem of security-aware CAD tools will provide superior, secure systems with near to ideal security coverage, require less effort, and make the products with built-in security more affordable.

3.3.7 Security and Trust Metrics and Assessment

Metrics are critical in all fields, as they help us measure where we are today, help us make decisions about what course of action needs to be taken, and help us identify if new paths represent an actual improvement over the state of practice. Good metrics produce data that informs an end-user about necessary adjustments and decisions. In hardware security, developing metrics to help us protect our systems is critical, and yet is challenging due to both the breadth of the attack vectors feasible and the cross-disciplinary nature of the problem. In surveying computer security metrics across both software and hardware [cite Howard and Longstaff] and experiences in multiple security research programs, a good metric has the following properties:

1. **Strategic**. To create effective performance metrics, the endpoint - the goals, objectives, or outcomes desired - needs to be considered first and then work backward. A good performance metric embodies a strategic objective.
2. **Mutually Exclusive**. A metric that classifies one category should exclude all others to enable system-level integration and reduce confusion.

3. **Exhaustive**. Taken together, the categories include all possibilities. For security this also includes extensibility to future systems with defensive upgrades or future attacks which downgrade defenses.
4. **Unambiguous**. Understandable, clear and precise so that classification is not uncertain, regardless of who is classifying. Users must know what is being measured, how it is calculated, what the targets are, and, more importantly, what they can do to affect the outcome in a positive direction.
5. **Repeatable & Comparative**. Repeated measurements are independently observable and result in the same classification, regardless of who is classifying. Measurements are comparative across technologies, approaches, and periods of time.
6. **Accepted.** Metrics must be logical and intuitive so that they become generally approved. They must be accepted by both the people measuring and the people accepting the results.
7. **Useful**. The metric could be used to gain insight into how the field behaves.

These properties can be difficult to achieve in most domains; however, once consensus is achieved, true progress can be realized. A key challenge is that security is a human concept, as opposed to a property of physics, and it can be challenging to develop metrics that anticipate human creativity or future use cases. The field of cyber software security has been around longer than hardware security, and while many of those metrics and defensive concepts can be translated and applied to the hardware domain, hardware security has its own challenges. Largely, the field is populated with "low-level" metrics that help to measure a particular aspect of a hardware system or its development process. However, there is no common agreement on what metrics to use even at the "low-level." Little work has been done at all in trying to bridge these "low-level" metrics to "high-level" metrics that help an end-user reason about the overall security performance level of their system.

An example is in the field of hardware obfuscation, where dozens of advanced hardware encryption techniques have been developed, yet each practitioner has developed their own metrics such as Hamming distance, Distinguishing Input Patterns, or formal verification anchor point failures. None of these “low-level” metrics directly answer the “high-level” strategic questions of “How long is my system protected in the field?” or “What do I need to change to my system to achieve 100 days of protection?”

The following sections provide a deeper look at some of the inherent challenges in metrics for hardware security and recommendations for future investigation.

3.3.7.1 Human Aspect of Hardware Security

Unlike most science and engineering, security is a human concept. Hardware can be well measured in terms of the power it draws, how fast the processor or bus clocks are, or how well they perform on a certain benchmark. But there is no such equivalent in hardware security.

There is a myriad of many engineering level metrics, which can measure the performance of a subcomponent, but these are often tied to how the security techniques were created. Some examples include Hamming distance, Discriminating Input Pattern, the number of traces until a

crypto unit is hacked, etc. Currently, it is not uncommon for developers of defensive techniques addressing the same problem to use completely different metrics to measure their effectiveness. Sadly, hardware security also rarely includes details on the conventional power, performance, area, and cost overhead metrics that could be easily measured. Common among the current state of practice in hardware security metrics is the inability to address the concerns of the end-user on items such as "How long will it take an adversary to unlock my obfuscation?" or "What is the risk that my system will be compromised if I do not include this security feature?"

In other words, little work has been done in translating engineering performance to human security concerns. There is a truism in that in having a hundred engineering level metrics, we, in fact, have no consensus on a single or useable reduced set of security metrics.

When trying to translate engineering performance to security-centric metrics, the unique aspects of hardware security come into play. First, not all systems are the same, and therefore, a one-size-fits-all solution is not appropriate. Like much of security, we need a framework which allows a mission to define what needs to be protected and measure of a solution's ability to protect it. The assumptions about a deployment scenario of a system are key to focusing the defenses in the manner that will provide the most cost to benefit ratio. At the same time, making incorrect assumptions about what security is required or what an attacker's abilities are, can lead to asymmetrical results. Insider attacks are often the least accounted for and most disruptive. This again points to the need for risk-aware based framework that can assess these trade-offs effectively.

3.3.7.2 Temporal Nature of Hardware Security

A huge challenge with hardware security is the temporal nature in which the game theoretics play out. In most cases, the defense moves first, and once fielded, an attacker has the lifetime of the platform to find a successful attack. Many systems, especially embedded systems, can have lifetimes of 15 years or longer. Our current design approaches require that a defensive designer account for an attack time of the full lifetime of the device, and also defend against attacks that may have never been seen before or even conceived of yet. This is in stark contrast to software security, where patches to the latest vulnerabilities can readily be deployed to fielded systems.

A chilling example is the advent of Differential Power Analysis attacks developed in the late 1990’s which correlate power signatures to the encryption algorithm running on a device in such a way as to determine which software loops were being executed and therefore determine what the key was. This attack had a disruptive effect, making all devices without DPA countermeasures effectively instantly broken.

Hardware security researchers and developers often bemoan protecting against such "unknown unknowns," but this phrase is unimaginative, and can be addressed using risk-based models that are common in many forecasting endeavors, including cybersecurity. This teaches us two things, first that new attacks can come out of human creativity, and we need to measure the risk that the assumptions we make during design time may not hold during the lifetime of the device. Second, where existing attack types may improve and get stronger, we need metrics which can be future-forward, and tell us the level of protection we should expect to have at the end of a platform's lifetime, and be able to compare the level of improvement between a device created with the state of the art security features in today's devices and one created in five years.

3.3.7.3 Independent Verification and Benchmarks

Another metrics related challenge in hardware security is independent verification and benchmarking. In traditional physics-based metrics, an end-user can easily use test equipment to verify the power consumption of a system or profile the throughput of their application, independent of the supplier of the technology, which builds openness and trust. This concept is non-existent in hardware security.

Additionally, many communities collectively agree on a useful set of benchmarks that can be used to measure performance. There have been attempts in some aspects of hardware security to develop benchmarks, such as TrustHub, however there can still be difficulties in that the attacks are publicly available, which can lead to the defense over-training on a set of attacks that are not fully representative of the entire set of attacks, the benchmark not evolving with the state of practice attacks, or the set of attacks even being suited for the particular deployment scenario of the device.

There are also several challenges on the assumptions or formulations of the problem that greatly affect the types of techniques that could be utilized and how to measure them. A current example is a debate about the efficacy or realism of techniques that require a golden reference example to use as a trusted baseline.

3.3.7.4 Recommended Areas of Investigation

A recommendation is that more research and development be done in the area of the interface from security engineering to program management. Actions need to be taken on how to translate engineering metrics into security metrics and provide guidance to decision makers on how best to deploy their resources. One starting place that software security has explored but hardware has not is in terms of defining attack surfaces and how much of an attack surface does a security approach addresses. Another approach can be explored by tying safety metrics and resilience data tie to security. This is also critical to address system-level security metrics in the integration of multiple security techniques. In addition, the metrics and benchmark development strategies should offer (if not embed) flexibility to account for attack types and attacker sophistication versus resources available.

3.3.8 Policy

The aspirations of policy surrounding Trusted and Assured Micro-Electronics are to balance access to leading-edge technologies and global markets with security and assurance of U.S. IP and systems. We are not the first to consider policy implications of securing microelectronics; recent reports include studies by the Defense Science Board [34], the Government Accountability Office [35], as well as the semiconductor industry recommendations in Winning the Future [36]. The discussion herein is intended to complement and build on those and other prior work, specifically with regards to the other topics addressed in this document. Three key challenges to realizing the desired balance of access and trust are the rapid evolution of technology, strategy to integrate assurance into acquisition policy, and a unified language and framework of assurance across governing agencies.

3.3.8.1 Rapid Evolution

A key challenge in policy regarding microelectronics is the relative pace of policy to that of technology. As discussed in Section [emerging research], new threats and countermeasures are continuously evolving. For U.S. interests to have access to global technologies and markets and to protect key investments and critical program information, the policy will need to be adaptable as technology changes. Hard thresholds for protected performance become obsolete rapidly and those protections impose an artificial hurdle to access, while new products would benefit from the rapid adoption of emerging standards and protections to address new threats. Adaptability is required in both the systems themselves and in the policies surrounding assurance to address new challenges and take advantage of new mitigations.

3.3.8.2 Integration into Acquisition Policy

To incorporate best practices and emergent technologies, DOD acquisition integrates hardware assurance into the Risk Managed Framework (RMF) and the Program Protection Plan (PPP). The Defense Science Board, in [34], points out a few current limitations of the PPP: security is addressed mostly after design, the PPP does not extend through fielded operation, and program personnel does not consistently have sufficient expertise and training in microelectronics assurance and security. Each of these weaknesses, and the challenges inherent in them is connected to another topic in this document. Overall, the integration into acquisition strategies is a subset of the transition to practice challenges addressed in Section [transition].

During the design phase, multiple hurdles inhibit the integration of assurance. Because the industry is global and foundry processes proprietary, building microelectronics only using DMEA Trusted Suppliers for IP, design services, and fabrication severely limits access to the most advanced technologies, thereby holding back the U.S. capabilities. Instead, an accepted set of standard practices to mitigate risks would enable access while reducing the threat to U.S. IP and systems. Challenges to developing such standards and best practices are discussed in Section [standards]. Further, a workforce – both private and public – that is more broadly educated in the criticality of assurance will promote the adoption of these security strategies early in the development process. Workforce training challenges are addressed in Section [workforce].

The Department of Defense Instruction 5200.44 [37] institutes policies to ensure trust/assurance, with specific objectives across the component lifecycle; as such, it addresses the DSB concerns over assurance of fielded systems. In order to adhere to this policy, a quantification of assurance is needed to assess risk and also the benefits of mitigations. Given our knowledge of the threat landscape, current testing procedures are impractical and incomplete. To assess the vulnerability in microelectronics requires the development of practical metrics, as well as tools to measure them, for each stage of lifecycle – design, fabrication, integration, and field systems. "A DARPA Approach to Trusted Microelectronics" [ref] lays out a starting point, but further metrics and quantification of assurance are required in response to evolving threats. The challenges to quantifying assurance are discussed further in Section [metrics].

3.3.8.3 Standardization Across Agencies

Microelectronics are foundational technologies used across industries, from medical devices to cars to nuclear submarines. Across these platforms, differing assurance needs and different

stakeholders have created a system of stovepiped and independent standards. Assurance policies, and related safety, reliability, and security policies, are defined independently by the authorities over each type of platform, e.g., FAA, FDA, DOD, NHTSA, etc., rather than through a unifying framework addressing microelectronics themselves.

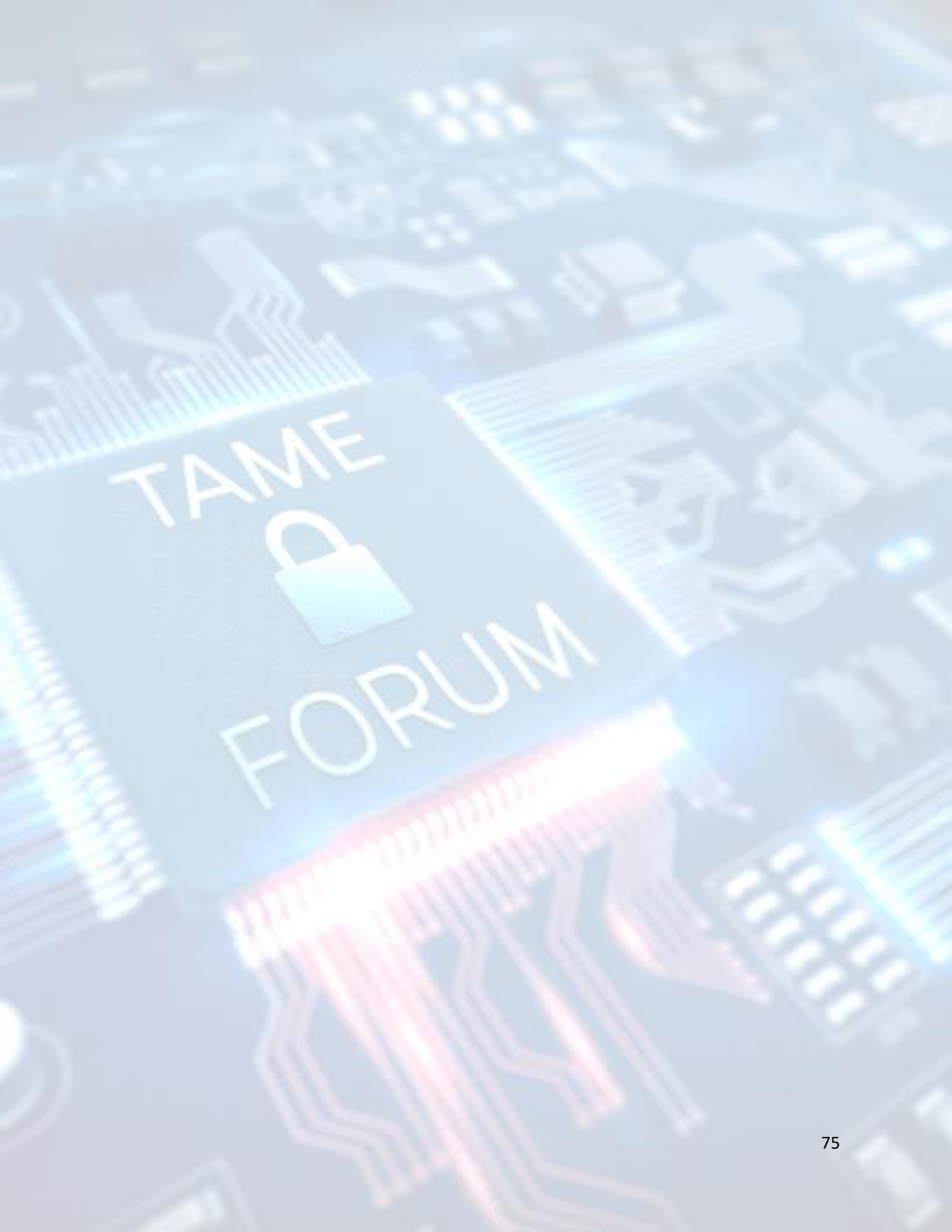
Because the needs of each application are different, the assurance requirements are unlikely to be the same; however, a consistent language and overarching structure, with varying levels or values for particular requirements, would enable broad specification of assurance in microelectronics and enable system engineers and integrators to select the most appropriate products for their end solutions. Further, this consistency would encourage investment in integrated circuits by broadening the market space for individual components through well-defined assurance criteria.

References

1. Evans, D.: Cisco White Paper on the Internet of Things: How the Next Evolution of the Internet is Changing Everything (2011).
2. Keating, M., Bricaud, P.: Reuse Methodology Manual for System-on-a-Chip Designs (1998).
3. Bhunia S., Agrawal, D., Nazhandali, L.: Guest Editor's Introduction: Trusted System-on-Chip with Untrusted Components. IEEE Des. Test Compute. (2013).
4. Tehranipoor, M.M., Guin, U., Bhunia, S.: Invasion of the Hardware Snatchers: Cloned Electronics Pollute the Market. IEEE Spectrum (2017).
5. Villasenor, J., Tehranipoor, M.: The Hidden Dangers of Chop-Shop Electronics: Clever counterfeiters sell old components as new threatening both military and commercial systems. IEEE Spectrum (cover story) (2013).
6. 2012 Information Technology Workforce Assessment for Cybersecurity (ITWAC) Summary Report March 14, 2013.
7. https://niccs.us-cert.gov/about-niccs/niccs
8. https://niccs.us-cert.gov/workforce-development
9. https://www.iad.gov/NIETP/reports/cae_designated_institutions.cfm
10. https://niccs.us-cert.gov/training/search
11. Elnaggar, R. & Chakrabarty, K., "Machine Learning for Hardware Security: Opportunities and Risks". J Electron Test (2018) 34: 183.
12. Y. Jin, D. Maliuk and Y. Makris, "Post-deployment trust evaluation in wireless cryptographic ICs," 2012 Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, 2012, pp. 965-970.
13. Asadizanjani, Navid, Mark Tehranipoor, and Domenic Forte. "Counterfeit Electronics Detection Using Image Processing and Machine Learning." Journal of Physics: Conference Series 787 (January 2017): 012023. doi:10.1088/1742-6596/787/1/012023.
14. Bahar Ahmadi, Bahram Javidi, Sina Shahbazmohamadi "Automated detection of counterfeit ICs using machine learning". 29th European Symposium on Reliability of Electron Devices, Failure Physics and Analysis (ESREF 2018)
15. S. Picek et al., "Side-channel analysis and machine learning: A practical perspective," 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, 2017, pp. 4095-4102.
16. Ciarán M. Lee and Matty J. Hoban "Towards Device-Independent Information Processing on General Quantum Networks". Phys. Rev. Lett. 120, 020504 –January 2018
17. Developing a secure, un-hackable net [online]. Available: https://www.ucl.ac.uk/news/2018/jan/developing-secure-un-hackable-net. [Accessed: March-6-2019].
18. Liu, Yi-Kai. "Single-Shot Security for One-Time Memories in the Isolated Qubits Model." In Advances in Cryptology – CRYPTO 2014, edited by Juan A. Garay and Rosario Gennaro, 19–36. Springer Berlin Heidelberg, 2014.

19. Quantum Key Distribution [online]. Available: https://en.wikipedia.org/wiki/Quantum_key_distribution. [Accessed: March-6-2019].
20. Kai Wen, Kiyoshi Tamaki, and Yoshihisa Yamamoto “Unconditional Security of Single-Photon Differential Phase Shift Quantum Key Distribution”. Phys. Rev. Lett. 103, 170503 –October 2009
21. Diamanti, Eleni, Hoi-Kwong Lo, Bing Qi, and Zhiliang Yuan. “Practical Challenges in Quantum Key Distribution.” Nature, Npj Quantum Information 2 (November 8, 2016): 16025.
22. Y. Bi, P. Gaillardon, X. S. Hu, M. Niemier, J. Yuan and Y. Jin, "Leveraging Emerging Technology for Hardware Security - Case Study on Silicon Nanowire FETs and Graphene SymFETs," 2014 IEEE 23rd Asian Test Symposium, Hangzhou, 2014, pp. 342-347.
23. A. Chen, X. S. Hu, Y. Jin, M. Niemier and X. Yin, "Using emerging technologies for hardware security beyond PUFs," 2016 Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, 2016, pp. 1544-1549.
24. F. Rahman, B. Shakya, X. Xu, D. Forte and M. Tehranipoor, "Security Beyond CMOS: Fundamentals, Applications, and Roadmap," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 25, no. 12, pp. 3420-3433, Dec. 2017.
25. Y. Xie, C. Bao, C. Serafy, T. Lu, A. Srivastava and M. Tehranipoor, "Security and Vulnerability Implications of 3D ICs," in IEEE Transactions on Multi-Scale Computing Systems, vol. 2, no. 2, pp. 108-122, 1 April-June 2016.
26. Xie, Yang, Chongxi Bao, and Ankur Srivastava. “3D/2.5D IC-Based Obfuscation.” In Hardware Protection through Obfuscation, edited by Domenic Forte, Swarup Bhunia, and Mark M. Tehranipoor, 291–314. Cham: Springer International Publishing, 2017.
27. https://www.statista.com/statistics/595182/worldwide-security-as-a-service-market-size/
28. H. Khattri, N. K. V. Mangipudi and S. Mandujano, "HSDL: A Security Development Lifecycle for hardware technologies," 2012 IEEE International Symposium on Hardware-Oriented Security and Trust, San Francisco, CA, 2012, pp. 116-121.
29. IEEE Standard for Standard Delay Format (SDF) for the Electronic Design Process," in IEEE Std 1497-2001 , vol., no., pp.0_1-, 2001.
30. Trust Hub Org, https://www.trust-hub.org
31. Trust Hub Vulnerability DB Physical Vulnerabilities, https://www.trust-hub.org/vulnerability-db/physical-vulnerabilities
32. Trust Hub Vulnerability DB CAD Solutions, https://www.trust-hub.org/vulnerability-db/cad-solutions
33. Florida Institute of Cyber Security, https://fics.institute.ufl.edu
34. Defense Science Board Task Force on Cyber Supply Chain Final Report, Office of the Undersecretary of Defense, Acquisition, Technology, and Logistics. 2017.
35. Mak, Marie A. "Trusted Defense Microelectronics: Future Access and Capabilities Are Uncertain." Testimony before the Subcommittee on Oversight and Investigations,

Committee on Armed Services, House of Representatives. 2015. https://www.gao.gov/assets/680/673401.pdf
36. Semiconductor Industry Association. "Winning the Future: A Blueprint for Sustained U.S. Leadership in Semiconductor Technology." 2019. https://www.semiconductors.org/winthefuture/
37. Department of Defense Instruction 5200.44 Incorporating change 3. October, 15 2018.
38. K. Xiao, A. Nahiyan and M. Tehranipoor, "Security Rule Checking in IC Design", in Computer (Supply Chain Security), 2016.
39. A. Nahiyan, K. Xiao, D. Forte, and M. Tehranipoor, "Security Rule Check," Hardware IP Security and Trust, Springer, 2017. 17–36.
40. C. Dunbar and G. Qu, "Designing Trusted Embedded Systems from Finite State Machines," ACM Transactions on Embedded Computing Systems (TECS), vol. 13, no. 5s, 2014.
41. Roy, Debapriya Basu, et al. "From theory to practice of private circuit: A cautionary note." Computer Design (ICCD), 2015 33rd IEEE International Conference on. IEEE, 2015.
42. A. Nahiyan, K. Xiao, K. Yang, Y. Jin, D. Forte and M. Tehranipoor, "AVFSM: a framework for identifying and mitigating vulnerabilities in fsms," DAC, 2016.
43. Animesh Chhotaray, Adib Nahiyan, Thomas Shrimpton, Domenic Forte, and Mark Tehranipoor. "Standardizing Bad Cryptographic Practice.", CCS 2017.
44. https://www.semiconductors.org/resources/semiconductor-research-opportunities-an-industry-vision-and-guide-2/sia-src-vision-report-3-30-17-2/
45. https://doi.org/10.1145/3195970.3199851
46. https://www.zdnet.com/article/at-t-employees-took-bribes-to-plant-malware-on-the-companys-network/
47. Nicole Fern ; Kwang-Ting Tim Cheng, "Evaluating Assertion Set Completeness to Expose Hardware Trojans and Verification Blindspots," Design, Automation & Test in Europe Conference & Exhibition (DATE), 2019.
48. https://niccs.us-cert.gov/about-niccs/niccs
49. https://www.iad.gov/NIETP/reports/cae_designated_institutions.cfm


TAME
FORUM