

System Security Integrated Through Hardware and Firmware (SSITH)

Proposers Day Overview

Develop hardware design tools to provide inherent security against hardware vulnerabilities that are exploited through software in DoD and commercial electronic systems.



Linton Salmon

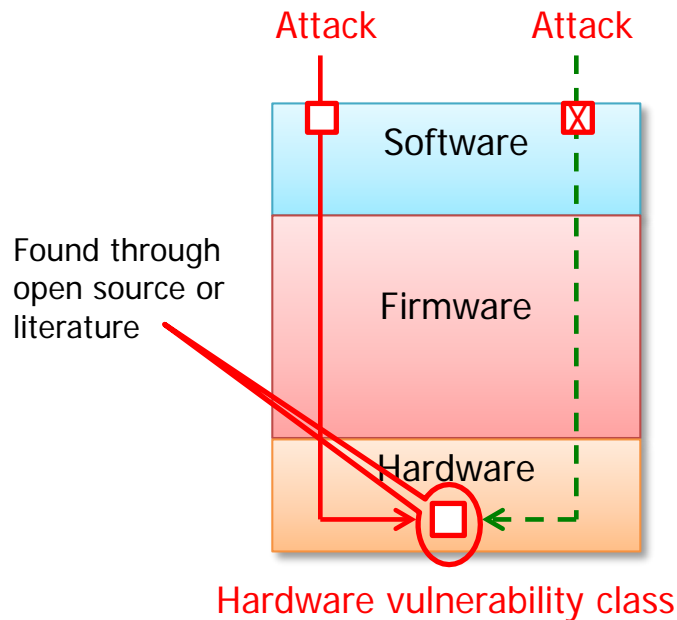
April 21, 2017



Electronic Systems Need Better Hardware Protection

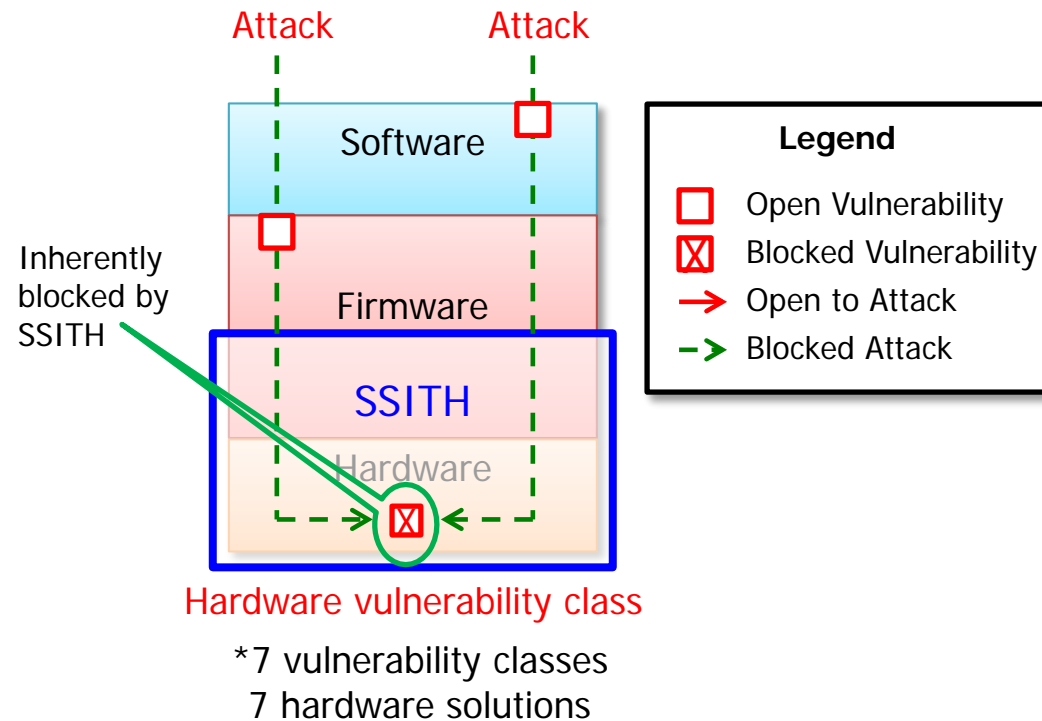
Today: Patch and Pray

*2800 vulnerability instances
2800 software patches



Future: SSITH

SSITH will protect against all 7 hardware classes



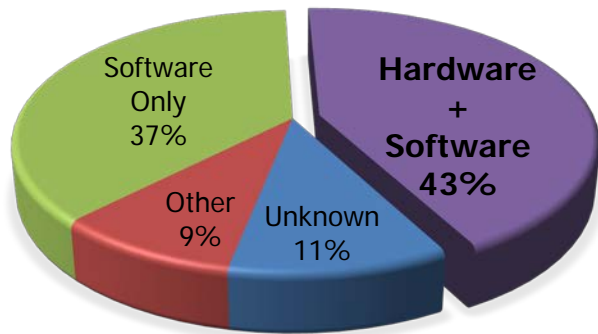
SSITH addresses hardware vulnerabilities at their source and will address current and future vulnerabilities



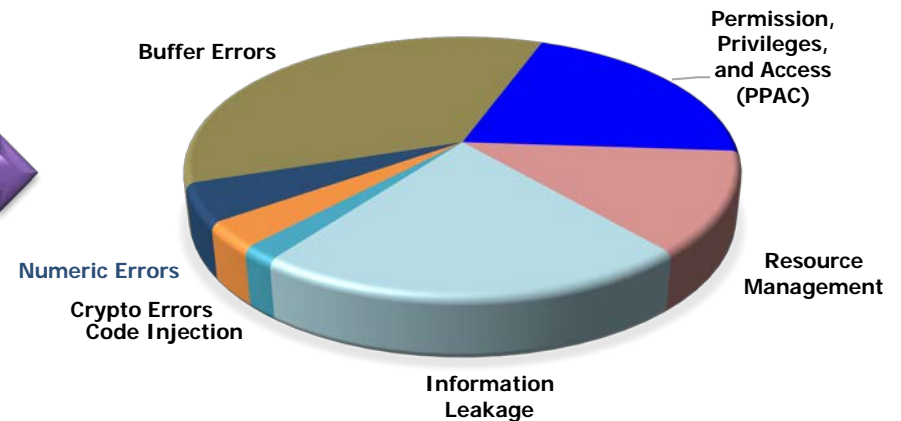
Software-Assisted Attacks on Hardware Are Important

A significant portion of vulnerabilities recorded are software attacks on hardware.

Electronic System Vulnerabilities



Hardware+ Software Vulnerabilities



Data from MITRE/NIST CVE website

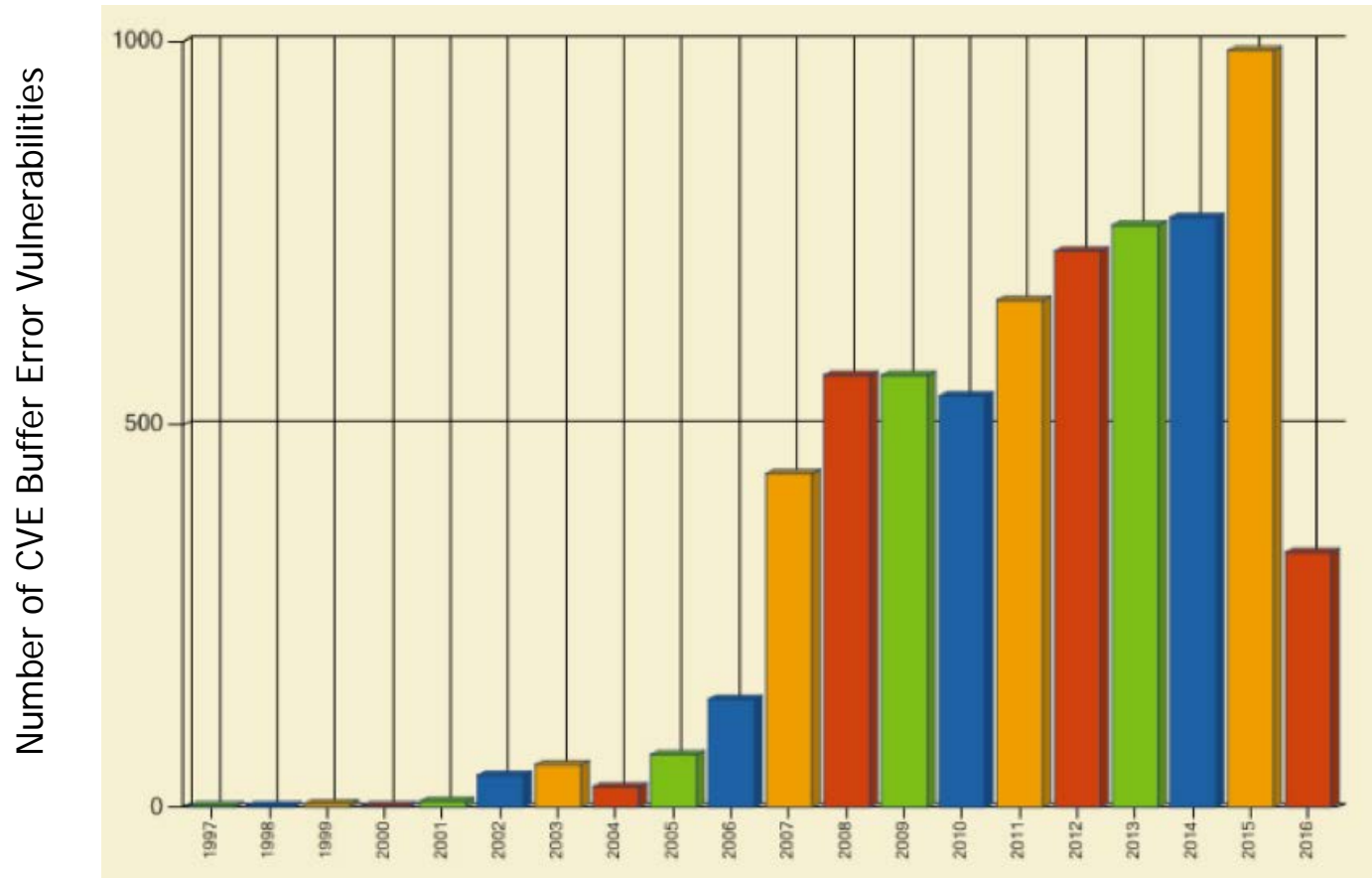
- Common Vulnerability Exposure (CVE-MITRE) commercial and DoD entries in 2015:
 - 6,488 recorded vulnerabilities in 2015
 - **43% were software-assisted hardware vulnerabilities**
- Projected to be addressed by SSITH:
 - 31% of the recorded vulnerabilities would have been prevented by SSITH
 - 12% of the recorded vulnerability prevention would have been addressed by SSITH
 - SSITH is not concerned with software only attacks (*e.g.* script errors) or anti-tamper (*e.g.* packaging)

Protection against software exploitation of hardware vulnerabilities is essential



Buffer Error Vulnerabilities Are Still Increasing

The number of Buffer Error vulnerabilities recorded in the CVE continues to increase year on year and reached a peak of almost 1,000 cases in 2015. We need solutions to this class of vulnerabilities, not just instances.



Data from MITRE/NIST CVE website

Year



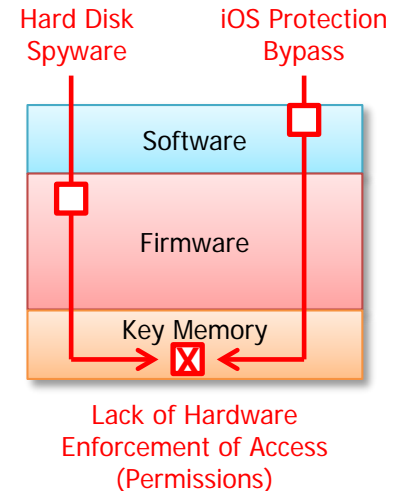
Classes of Security Vulnerabilities Can be Addressed in Hardware

Hardware vulnerability classes: *Permissions

Definition: "Management of permissions, privileges, and other security features that are used to perform access control" (*CWE definitions-MITRE*)

Vulnerability: Inability for hardware to restrict malicious requests for access to secure memory and operations

Potential SSITH solution: Establish hardware methods to constrain access to legitimate software sources



Example 1: Storage device (QNAP®) spyware instance (CVE-2009-3200)

Vulnerability: Attackers were able to use malicious firmware to define hard disk encryption keys in memory and use those keys to inappropriately encrypt/decrypt the hard drive.

Current solution: Software rewritten to block loading of malicious firmware

Potential SSITH solution: Tag encryption key memory, tag instructions allowed to access those keys, and implement rules to require both tags for decryption

Example 2: Apple iOS (pre 9.3 version) protection bypass instance (CVE 2016-1751)

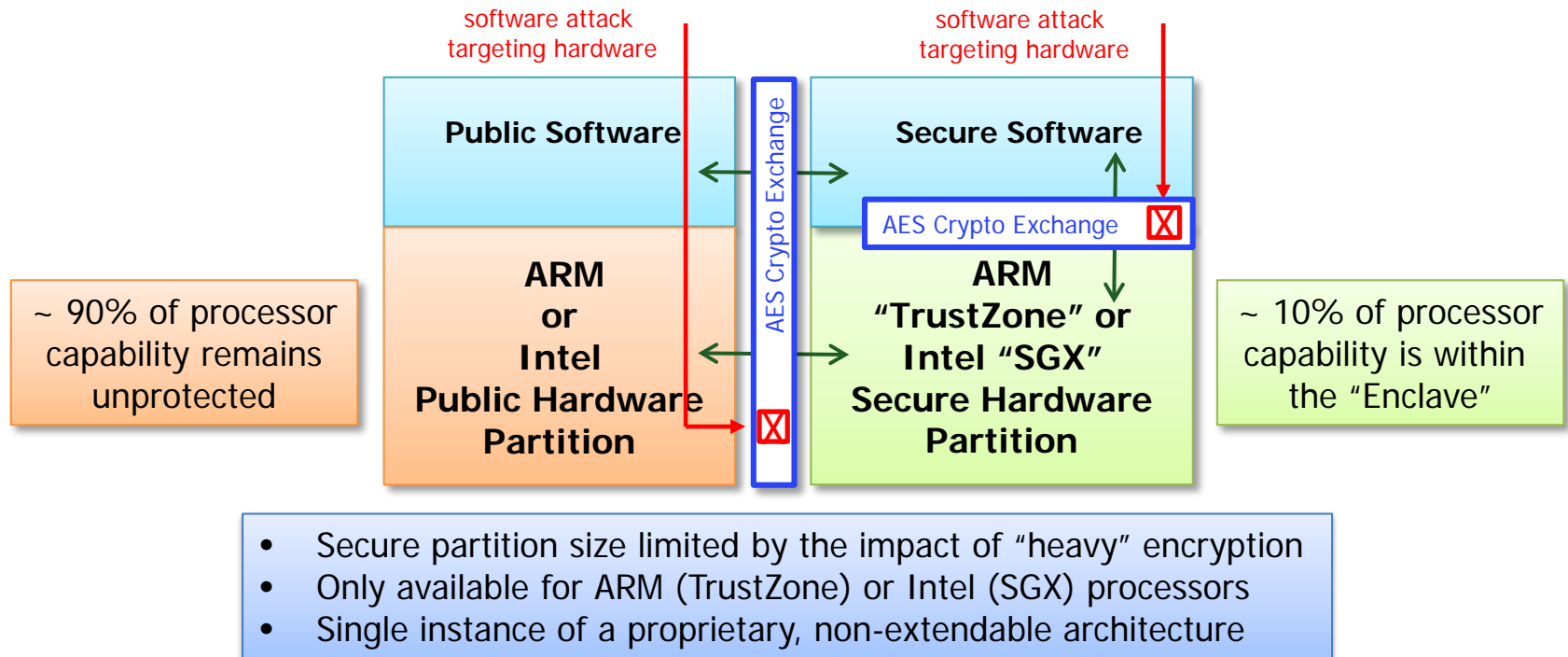
Vulnerability: Attackers were able to exploit an error in the operating system (OS) to bypass OS code-signing protection and inappropriately access hardware.

Current solution: iOS was rewritten to further restrict execution permission

Potential SSITH solution: Tag signing key memory, tag instructions allowed to access those keys, and implement rules to require both tags for access

Current commercial hardware methods are limited to items companies can “monetize”

Example: ARM/Intel “Secure Enclaves” are limited to a single section of their processor



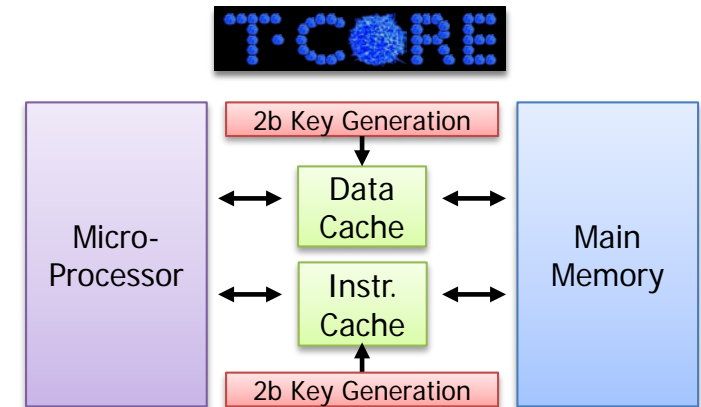
State-of-the-art commercial approaches don’t meet DoD requirements for hardware security across a broad range of applications and SoCs.

Unlike commercial efforts, SSITH will provide design tools that will be open and extendible to enable hardware security across DoD and commercial systems.



Current DoD hardware security approaches focus on narrow program needs.

- **Efforts to build secure micro-processors**
 - Innovation focused on a specific application of interest and not broadly applicable
 - Example: unclassified AFRL T-CORE processor
 - Most other efforts are classified
 - Protection limited to memory security
- **Protection of the electronics in use**
 - Reverse engineering
 - Software protection
 - Supply chain protection
- **Discrete hardware elements**
 - Physically Unclonable Functions (PUF)
 - Random Number Generators (RNG)

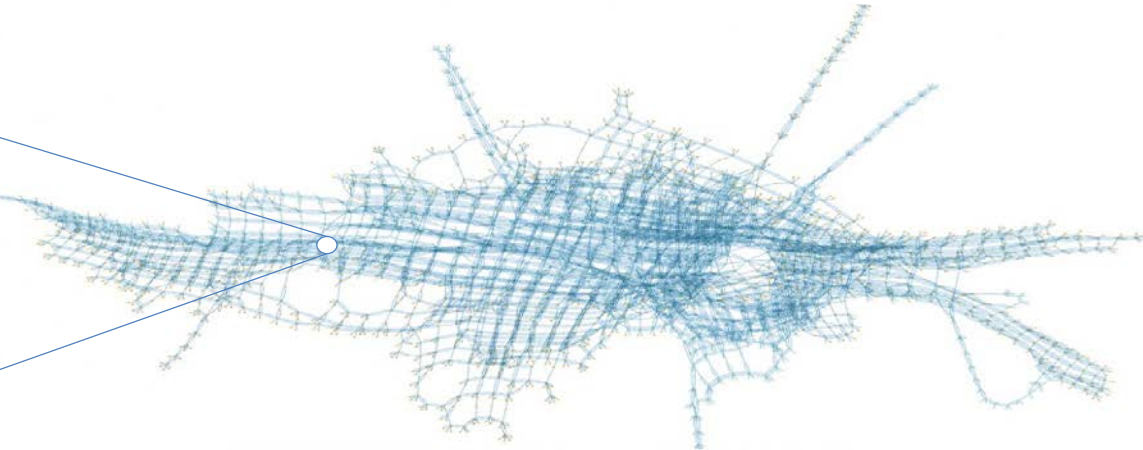


SSITH will develop concepts and design flow elements to secure a broad range of DoD systems against software exploitation of hardware vulnerabilities.




Approach: Limit the Hardware to Allowable States

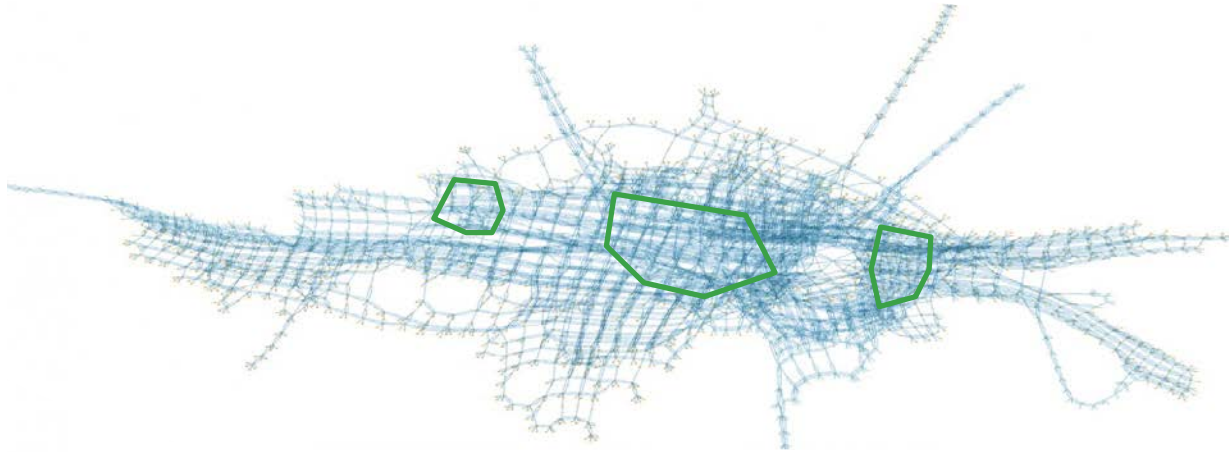
<i>TA</i>	<i>TB</i>	<i>TC</i>	<i>TD</i>	<i>TE</i>
0	0	0	0	1
0	0	0	1	1
0	0	0	0	1
0	0	1	1	1
0	0	0	0	1



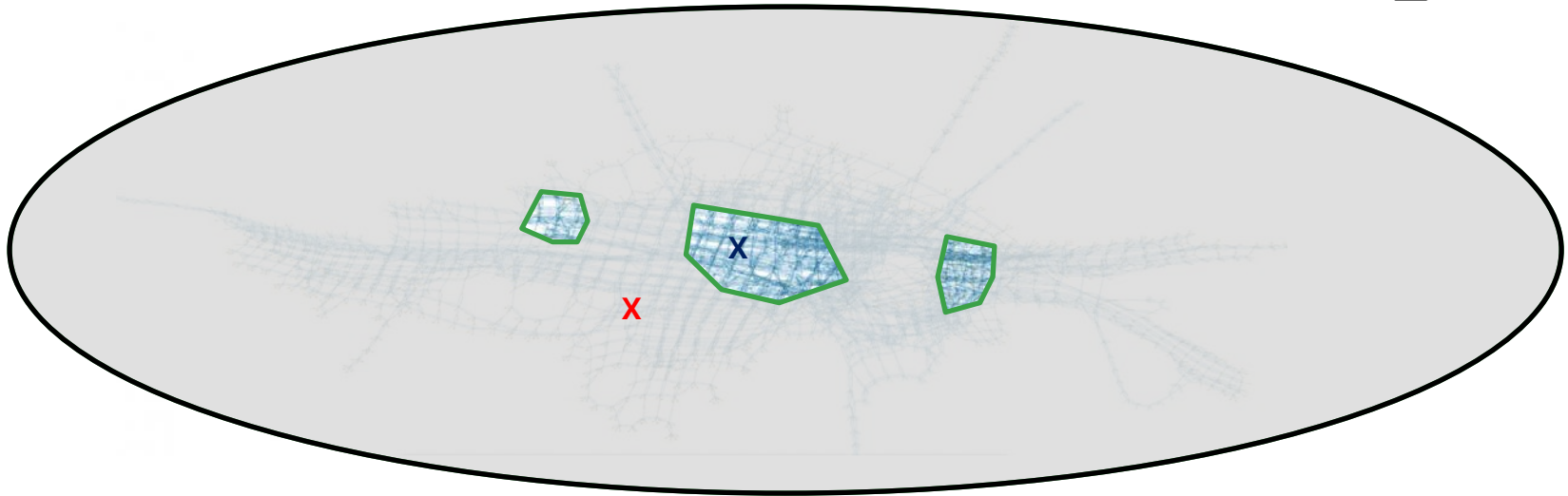


Approach: Limit the Hardware to Allowable States

 Allowed states



- Allowed states
- Not-allowed states



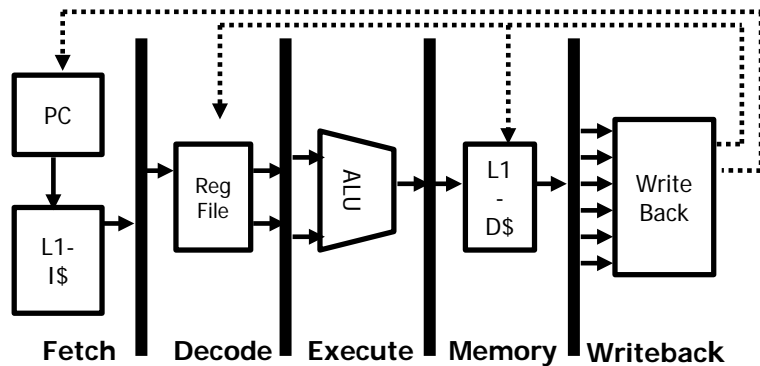
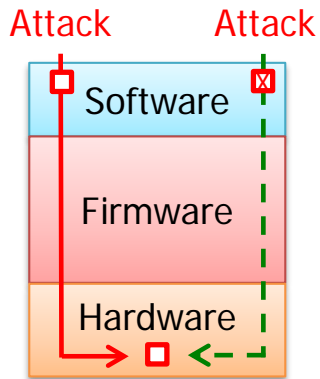
Limiting the allowable state space makes securing the system much easier

- Restriction of allowed hardware states permits verification through exhaustive techniques such as formal methods
- It is critical to restrict the state space while maintaining system performance

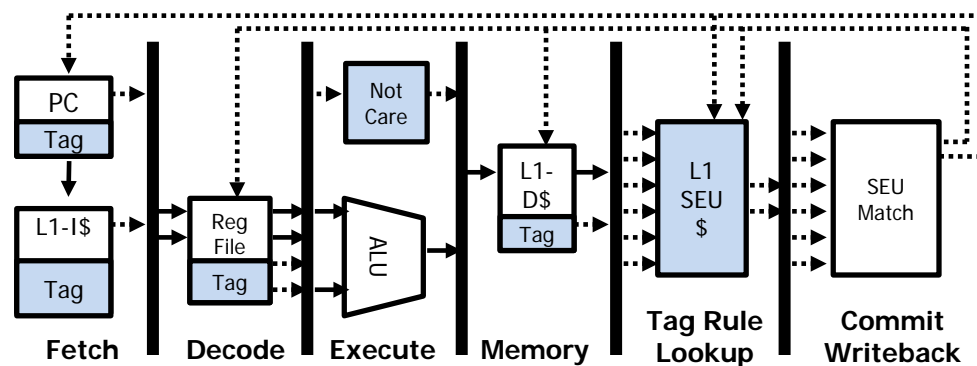
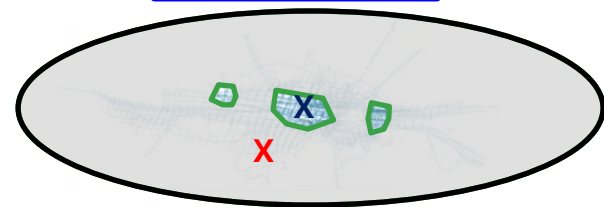
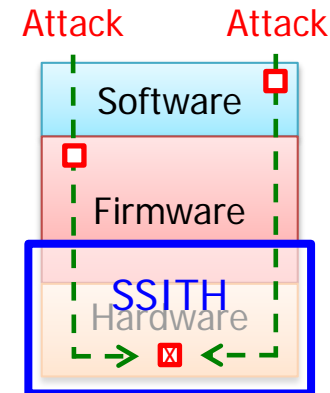
Potential methods to instantiate this approach:

- Meta-data tagging: Allowed states restricted through data/instruction tagging/rules
- Verified state matching: Allowed states restricted to design-verified vectors
- Anomalous state detection: Machine learning identification of normal and anomalous states

Today



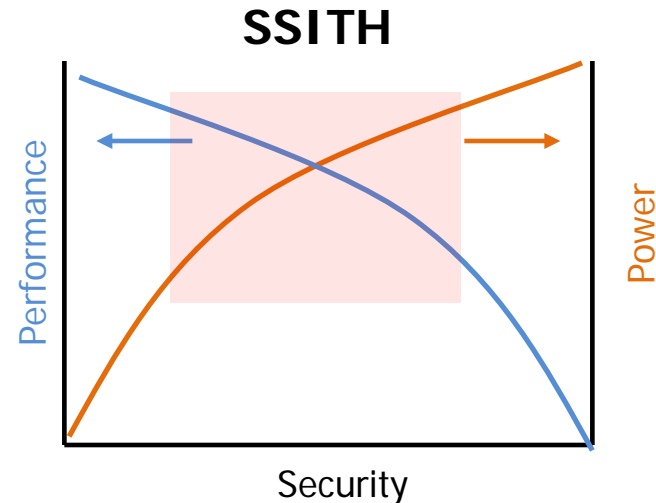
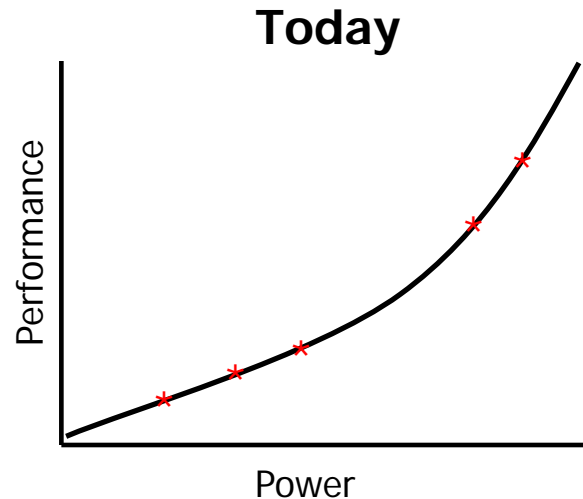
SSITH



SSITH will appropriately constrain the hardware state space to address security vulnerabilities while imposing acceptable increases in CSWaP



Security Can be Implemented with Acceptable CSWaP Impact



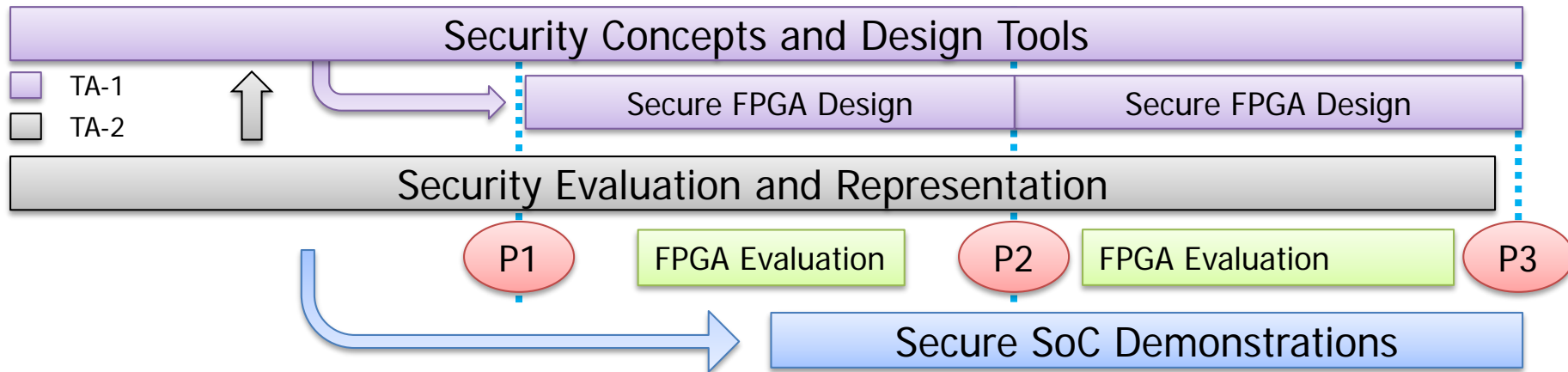
- **Simulated system (unoptimized) overhead for a tagging implementation** (data from BAE Systems)
 - Impact on performance: < 10%
 - Impact on power consumption: < 10%
 - Impact on area: ~ 100% increase
 - Assumptions: 64-bit tag, 4 diverse policies, RISC microprocessor, SPEC 2006 benchmark
- **Implementation in hardware allows more security capability** (data from ARM Holdings)
 - RSA public key operation: 2,310% better performance at power (3,850 OPS* vs 167 OPS)
 - ECC public key operation: 1,320% better performance at power (60.5 OPS vs 4.57 OPS)

SSITH will leverage recent technology advancements to find/develop more efficient ways to implement electronic system security in hardware as measured by CSWaP.



SSITH Program Plan

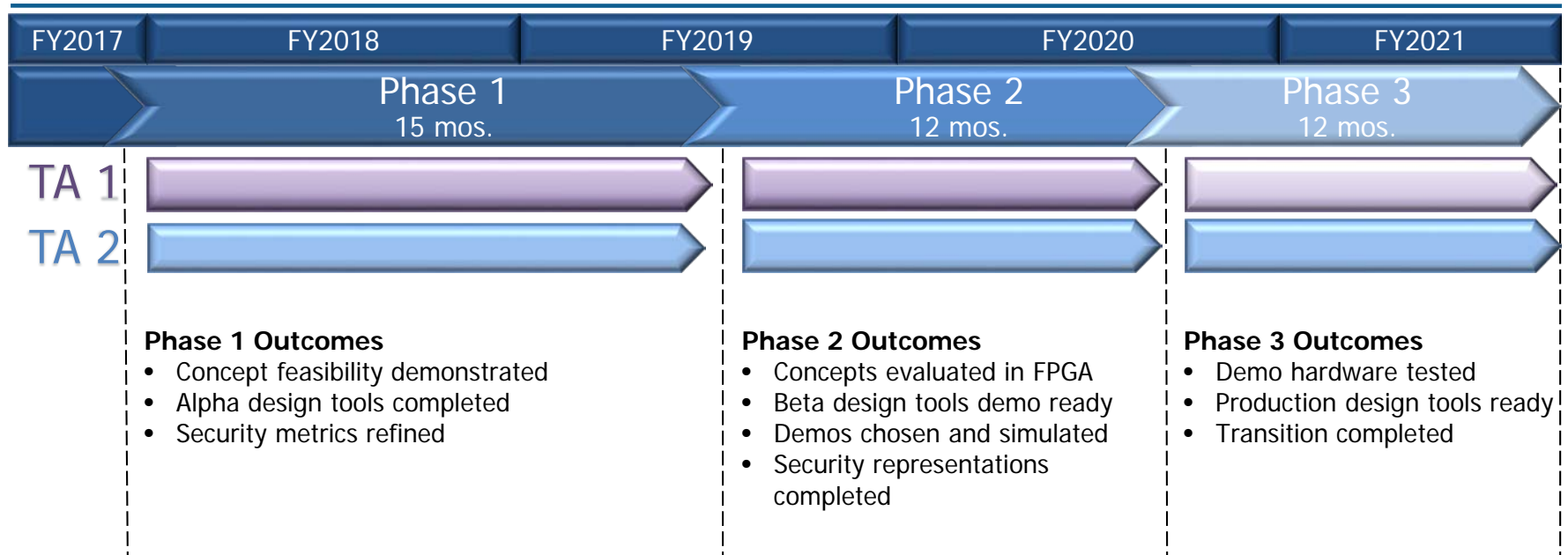
- **TA-1: Novel hardware security architecture and design tool development**
 - Develop security architectures that provide protection against classes of hardware vulnerabilities
 - Develop design tools that implement the security architectures being explored
- **TA-2: Security Evaluation Methodologies and Metrics**
 - Create and implement a security evaluation methodology for the program
 - Define quantitative security metrics for hardware/firmware security
 - Establish a security representation framework for hardware/firmware security
- **Potential future BAA**
 - Incorporate TA-1 security architecture(s) in a full SoC design
 - Submit the fabricated SoC for security evaluation



SSITH will develop and refine hardware security concepts, instantiate those concepts in a set of SoC design tools, and demonstrate their effectiveness in hardware.



SSITH Program Schedule



- TA-1 Novel hardware security architecture/design tools
 - Investigation of novel hardware/firmware concepts
 - Creation of IC design flows to implement concepts
 - Core IP design and proof in FPGA
- TA-3 Security evaluation methodologies and metrics
 - Define quantitative hardware/firmware security metrics
 - Establish a security representation framework for hardware/firmware security

Metric	Goal
Performance Impact	<10%
Power Impact	=0%
Area Impact	<30%
Increased Security	Proven against 7 classes
Scalability	Coverage with 3 SoC platforms



Technical Area 1 (TA-1)

Novel hardware security architecture and design tool development



SSITH Rationale

The problem is:

Current security solutions against software-assisted exploitations of hardware vulnerabilities use software patchwork barrier approaches rather than intrinsic barrier approaches and as a result are either flexible/scalable or hard to hack, but not both

Image from Wikimedia Commons

```
streampos begin,e  
ifstream myfile (  
begin = myfile.te  
myfile.seekg (0,  
end = myfile.tell  
myfile.close();
```

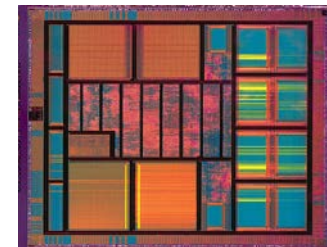
Software

Flexible
Scalable
Easy to hack
Patch solutions
Open architecture



SSITH

Flexible
Scalable
Hard to hack
Intrinsic solutions
Integrated architecture



Hardware

Fixed
Not scalable
Hard to hack
Patch solutions
Closed architecture

A goal of SSITH is to:

Develop architectures and design flows that use the combination of hardware and firmware to provide flexible and scalable intrinsic security for DoD electronic systems



SSITH Technical Program Characteristics

SSITH will provide IP and design flows that will enable design of GOTS parts that provide greatly enhanced security for DoD electronic systems.

- SSITH will provide flexible and scalable protection
 - Scalability of the solution at design or dynamically adjustable
 - Tradeoff of security and loss of performance/power/area
 - Tradeoff of security vs consequences
 - Flexibility through standard approaches and flexible implementation
 - Similar to biological systems that use antibodies as an approach, but the antibody changes in response to a threat
- SSITH will provide inherently secure protection
 - Security is a fundamental part of the logic architecture
 - The architecture drives security implementation
 - Inherent nature of architecture drives asymmetry in favor of the defender
- SSITH will provide security that limits any successful attack
 - Approach reduces size of the protected volume (# gates/code/etc.)
 - The level/cost of security increases as protected volume is reduced
 - Restrict the impact of a hack to a single system through the use of hardware primitives such as PUFs



Examples of State-Aware Architectures

Architectural Concept	High-level Description	Approach				
		Intrinsic	Scalable	Flexible	Low-Power Impact	Low-Performance Impact
Meta-Data Tagging	Establishes multi-bit tags on all data and operations; defines acceptable use cases for each via policies.	✓	✓	✓	✓	?
Verified State Matching	Defines trusted hardware pathways through circuit verification and run other pathways in slower/more secure mode.	✓	✓	✓	✓	?
Anomalous State Detection	Implements traffic monitors on the circuit wide network or bus (NOC) and identify normal and abnormal behavior.	✓	✓	✓	✓	?
Multi-Party Computing	Partitions logical function onto several separate circuits and cryptographically recombines to ensure security.	✓	?	?	?	✓
Semi Homomorphic Computing	Utilizes a mathematically proven method of homomorphic encryption in hardware in a form that limits the hardware overhead.	✓	✓	?	?	?

SSITH will examine these and other architectures and develop the best to meet program goals

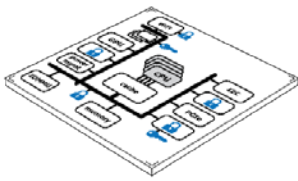

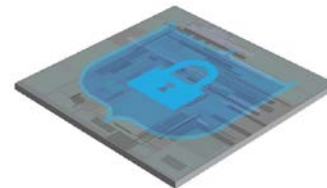


TA-1 Key Elements: Tasks (from SSITH BAA pg. 8)

- ***Security architectures:*** develop and demonstrate one or more security architectures that can be used to protect electronics systems from software-assisted attacks that exploit the 7 CWE hardware vulnerability classes. TA-1 teams must show how the security architecture will secure designs, and how it would be implemented.
- ***Design tool development:*** develop design tools required to implement the chosen security architectures in arbitrary circuit designs. The design tools may include methods or techniques which utilize new EDA software developments and/or modifications to existing EDA software that enable other design teams to utilize the security architecture to secure future circuit designs. Proposals should include details about how the design tools developed in TA-1 would insert security at the hardware level into circuit elements, circuit blocks and hardware architectures.
- ***Impact of security implementation:*** evaluate the impact of the security architecture implementation on key circuit metrics as described in section 3, and demonstrate the impact on circuit metrics through simulation and custom circuit emulation.



TA-1 Key Elements: Tasks by Phase (from SSITH BAA pg. 10)

Phase 1	Phase 2	Phase 3
		
<u>Security architectures</u> 1. Prove feasibility of security architectures that provide protection against the seven CWE classes of hardware vulnerabilities	<u>Security architectures</u> 1. Implement security architectures in circuit designs.	<u>Security architectures</u> 1. Implement security architectures in circuit designs.
<u>Design tool development</u> 2. Develop <i>alpha</i> * design tools that implement security architectures	<u>Design tool development</u> 2. Develop <i>beta</i> ** design tools that implement security architectures 3. Implement security architectures on the three GFE-provided FPGA designs using design tools.	<u>Design tool development</u> 2. Develop <i>production</i> *** design tools to implement security architectures 3. Implement a second version of the security architectures on the three GFE-provided FPGA designs using design tools.
<u>Impact of security on metrics</u> 3. Establish, by simulation, impact of security architectures on PPASS: <ul style="list-style-type: none"> • Power/performance • Design area/complexity • Security • Software compatibility 	<u>Impact of security on metrics</u> 4. Establish by simulation and FPGA demo, impact of security architecture on PPASS 5. Support implementation of security architectures	<u>Impact of security on metrics</u> 4. Establish, by and simulation and FPGA demo, impact of security architecture on PPASS 5. Support implementation of security architectures

* Alpha: Usable by the developing team

** Beta: Usable by other design teams with significant interaction with the developing team

*** Production: Sufficiently robust and documented for use by other design teams without support of the developing team



TA-1 Key Elements: Characteristics (from SSITH BAA pg. 8)

- ***Scalability:*** demonstrate that implementation of the security architecture enables scaling of security across a wide range of system parameters, such as power, performance, and complexity. Demonstrate that scalability will enable use of security architectures across a wide range of applications (small to large).
- ***Flexibility:*** demonstrate that the selected security architecture can be used to upgrade hardware to protect against newly found vulnerabilities without requiring redesign of the hardware.
- ***Adaptability:*** demonstrate that the selected security architecture can adapt system characteristics to respond to detected known attacks on the electronic system without reprogramming or firmware modification. Demonstrations will show the ability of the architecture to detect and adaptively respond to classes of attacks in an appropriate manner. For example, the security architecture could detect a request for inappropriate permission access through an IO by lowering the IO transfer rate and restricting data exchange to known safe pathways.



Need for Scalability

The broad range of DoD systems drive the need for a scalable security approach.



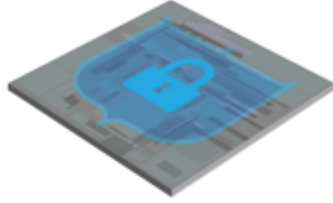
System Type:	Large, fixed	Portable	Mobile
Power	kW	W	mW
Performance	T-OPS	G-OPS	M-OPS
Area/Cost	Expensive	Cost Sensitive	Low Cost
Security Sensitivity	ConOp/Product Driven		
Public Domain Vehicle	BOOM-2	Rocket	Z-scale

- Scalability in power/performance/area required to meet broad range of DoD security needs
 - Electronic power impact limits are higher for a fixed system than for a mobile system
 - Security requirements are higher for fighter jet avionics than for a observation UAV
 - Cost requirements are lower for a ground-based radar than for a squad radio
- SSITH scaling span example:
 - 4bit vs 128bit tag length
 - Security: 10 vs 10^{30} span in computational complexity to break encryption
 - Performance/Power: Used area to drive to ~ 10% in both cases
 - Area: 12% vs 215% span in increased area of SoC
 - Scaling implemented through use of SSITH design tools to design different SoCs



TA-1 Key Elements: Metrics (from SSITH BAA pg. 9)

- ***Security***: demonstrate that selected security architectures effectively secure electronic systems against attacks on all seven (7) CWE hardware vulnerability classes described in Appendix 3. TA-1 teams must address
 - a theoretical evaluation of protection;
 - evaluation of the protection architecture against security metrics established by TA-2 performers; and
 - resistance of the security architecture against a 'red team' attack on the architecture as instantiated in hardware.
- ***Performance at power***: accurately quantify the *impact of implementing security on key circuit parameters* such as circuit performance, power, robustness, and reliability. TA-1 teams must demonstrate the ability to trade-off these parameters against each other and with respect to security and area/complexity metrics.
- ***Area/complexity***: accurately quantify the *impact of implementing security on circuit area and on design complexity*. TA-1 teams must demonstrate the ability to trade-off these parameters against each other and with respect to metrics a (*security*) and b (*performance at power*).
- ***Software compatibility***: ensure that existing application software will run on hardware secured with SSITH and minimize the amount of software modifications required to implement all of the SSITH security features.

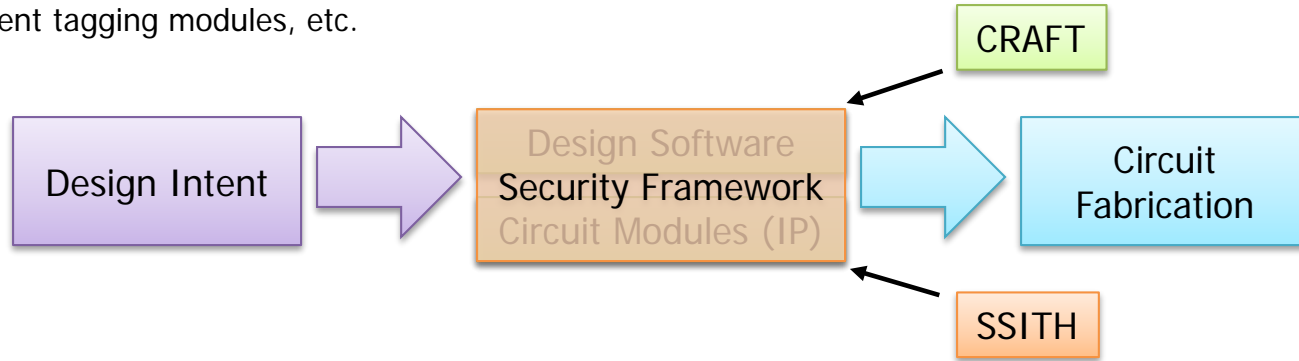
Phase 1	Phase 2	Phase 3
		
1. Provide 100% protection against at least three (3) CWE (reference) classes of the defined hardware vulnerabilities exploited by software attacks.	1. Provide 100% protection against at least five (5) CWE (reference) classes of the defined hardware vulnerabilities exploited by software attacks.	1. Provide 100% protection against all seven (7) CWE (reference) classes of the defined hardware vulnerabilities exploited by software attacks.
2. Prove security protection <i>against the three (3) CWE classes</i> through use of simulation and theoretical justification derived from performer-selected design.	2. Prove security protection <i>against the five (5) CWE classes</i> through red-team attack on the first pass FPGA instantiation of the three Government-provided GFE processor designs.	2. Prove security protection <i>against the seven (7) CWE classes</i> through red-team attack on the second pass FPGA instantiation of the three Government-provided GFE processor designs.
3. Provide quantitative simulation data indicating that improvement on system power, performance, and area (PPA) overhead meets Phase 1 metrics while maintaining software compatibility: -Performance impact < 20% -Power impact = 0% -Area impact < 50%	3. Provide quantitative simulation and FPGA data indicating that improvement on system power, performance, and area (PPA) overhead meets Phase 2 metrics while maintaining software compatibility: -Performance impact < 15% -Power impact = 0% -Area impact < 40%	3. Provide quantitative simulation and FPGA data indicating that improvement on system power, performance, and area (PPA) overhead meets Phase 3 metrics while maintaining software compatibility: -Performance impact < 10% -Power impact = 0% -Area impact < 30%
4. Provide theoretical or empirical evidence of the ability to respond to new hardware vulnerabilities without modifying hardware.	4. Demonstrate the ability to respond to new hardware vulnerabilities without modifying FPGA hardware.	4. Demonstrate the ability to respond to new hardware vulnerabilities without modifying FPGA hardware.



Security Should be an Innate Part of the Design Flow

Security becomes easier with SSITH frameworks integrated into the design flow.

- Sections of the design flow
 - Software tools: software to design security-aware buses, tagging protocols, etc.
 - Verification methods: new EDA tools, logic partitioning verification, etc.
- Circuit modules (IP)
 - Efficient cryptography modules
 - Efficient tagging modules, etc.



SSITH provides security ideas and frameworks that will revolutionize hardware design for security by:

- Enabling broad use of key security concepts across the DoD and the commercial sector
- Addressing the major software-assisted hardware attack categories through unified frameworks
- Permitting scaled use of security concepts across the wide range of system PPASS requirements



Technical Area 2 (TA-2)

Security Evaluation Methodologies and Metrics



TA-2 Focus (from SSITH BAA pp. 11-12)

- “The focus of TA-2 is to develop a methodology and metrics by which to measure secure electronic systems. Specifically, TA-2 teams must develop quantitative metrics required to evaluate trade-offs in security, performance, power, area and other standard circuit metrics. In addition, TA-2 teams must establish a framework that enables representation of hardware/firmware security properties to overall system designers.”
- Quantitative metrics
 - Generally lacking in the community
 - Required to enable PPAS tradeoffs
 - Will probably require a strong theoretical foundation
 - Will augment “Red Team” evaluation in SSITH



TA-2 Tasks by Phase (SSITH BAA pp. 11-12)

- ***Definition of quantitative security metrics for hardware security:*** These metrics must be measureable and must enable trade-off decisions with respect to other circuit parameters such as performance, power, and area. The metrics must correlate with both the attack vector (e.g., software, IO port) and the protection surface (e.g., software intrusion, IO intrusion).
- ***Establishment of a framework for hardware/firmware security:*** This framework will permit overall evaluation of system security. The framework must have a theoretical and/or empirical foundation. It must enable a common basis on which to communicate and evaluate security properties.

Phase 1	Phase 2	Phase 3
1. Define initial quantitative metrics for the hardware/firmware security and coordinate use of these metrics to evaluate TA-1 performers.	1. Finalize the security metrics and support their use to drive engineering tradeoffs in the first version of the FPGA designs.	1. Formalize security metrics and support their use to drive engineering tradeoffs in the second version of the FPGA designs.
2. Establish an initial theoretical security representation framework.	2. Refine the security representation framework established in Phase 1 and support use of the framework by TA-1 teams to represent their hardware security architectures	2. Collaborate with TA-1 design teams to represent their security properties in the established security representation framework.



Security Representation and Verification

Commercial Approach to Security

Software

Individual application security evaluation
Overall system security is evaluated heuristically

Firmware

Security a minor consideration
Firmware security frequently independent from software or hardware

Hardware

Major security effort is in circuit verification
Verification is focused on functionality
Malicious intent to create errors is often ignored
Security communicated through specification sheets

DARPA Formal Method Approach

Software

Assertion:
System is secure against software attacks
Assumption:
OS executes microcode as instructed

Firmware

Assertion:
OS executes microcode as instructed
Assumption:
SoC performs only specified operations

Hardware

Assertion:
SoC performs only specified operations with hardware vulnerability protection
Assumption:
IP performs only as per spec
Transistors perform as per models

HACMS

SSITH

Develop a security "contract" between hardware and software

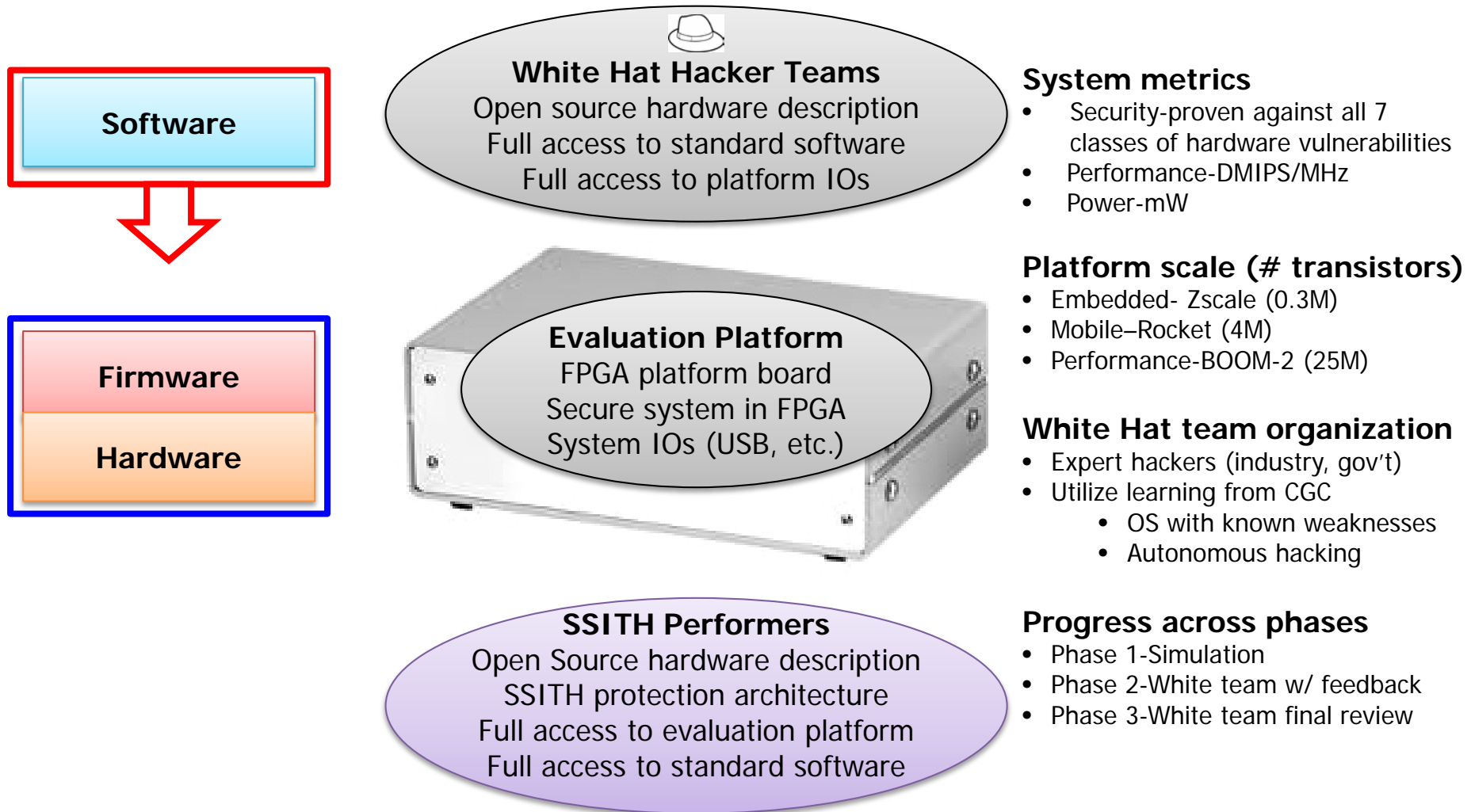
Hardware security must be rigorously evaluated and appropriately represented to software and system designers in order to secure the entire system.



Other BAA Comments



FPGA Evaluation in Phases 2 & 3



Demonstrate system protection against classes of hardware vulnerabilities that can be exploited by software.



- An FPGA board with IOs and 2X the capacity required to implement the most complex RISC-V baseline circuit. The FPGA board will be mounted in a chassis containing the FPGA board and accessible IOs.
- RTL and FPGA bitstream for three different sizes of RISC-V processor designs:
 - A small, reduced-feature version of the Rocket processor,
 - A full-featured, single threaded version of the Rocket processor, and
 - A full-featured, multi-threaded, out of order execution RISC-V processor.
- Versions of an Operating System that can be run on each of the three processors.



Out of Scope Technical Areas (from SSITH BAA pg. 7)

- The SSITH BAA will not focus on attacks that are not mediated through software access to the hardware. Although other areas of security are important, SSITH will focus on hardware vulnerabilities that are exploited through software to define achievable goals in a limited, but critical, part of the overall cybersecurity enterprise.
- Examples of out of scope topics are:
 - Development of physical elements of hardware security such as Physically Unclonable Functions (PUF) and Random Number Generators (RNG). Physical elements can be used as a part of a SSITH proposal, but SSITH will not fund their development.
 - Protection against hardware-only vulnerabilities such as EM side-channel attacks or insertion of hardware Trojans during design and/or fabrication.
 - Vulnerabilities that occur exclusively in the software domain, such as insecure interaction between software components or cross-site request forgeries.



Other Proposal Rules

- Submission requirements
 - "Individual proposals should address one of the two technical areas; organizations wishing to propose to both technical areas must submit a separate proposal for each." (BAA pg 7)
- Classified submissions
 - Classified submissions will be accepted, as appropriate
 - Review the BAA (particularly pp 28-30) for the rules regarding submission
- Proposal submission date
 - "Full proposals must be submitted to DARPA/MTO on or before 1:00 PM, Eastern Time, 22 May 2017 in order to be considered during the single round of selections. Proposals received after this deadline will not be reviewed." (BAA pg 32)



Transition Plan

DoD transition:

- Goal is to transition the IP and design methods through the CRAFT repository and/or other DoD repositories
 - Software tools will be integrated with a reference design flow
 - IP will be available using the CRAFT IP format specification
- Goal is to have the backbone of the approach unclassified
 - General methodology, design reference flow, selected IP
 - At least one implementation example
- Other portions may be classified
 - Specific applications
 - Application-specific or unique security IP



Commercial involvement:

- Critical to convince companies that implementing security is in their best interest
 - Value to their customers (SSITH: security metrics and demonstration of effectiveness)
 - Low risk/cost to implement (SSITH: development of design tools including simulators)
 - Scalable to meet their customer needs (SSITH: demonstration from embedded to fixed)
- Requires corporate knowledge/involvement in SSITH program
 - Already seeking their participation: Intel, Xilinx, TI, NxP, Medtronic, others
 - Emphasis on engagement in program



www.darpa.mil