|galois|

FireSim in High-Profile Action—FETT: DARPA's First Ever Bug Bounty Program

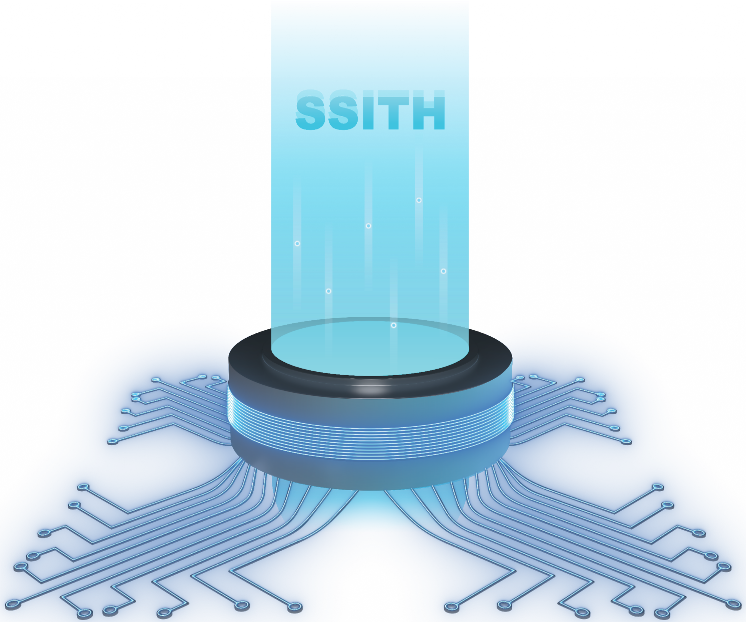
Joseph R. Kiniry

The First FireSim Workshop, March 2023

This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA). The views, opinions, and/or findings expressed are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

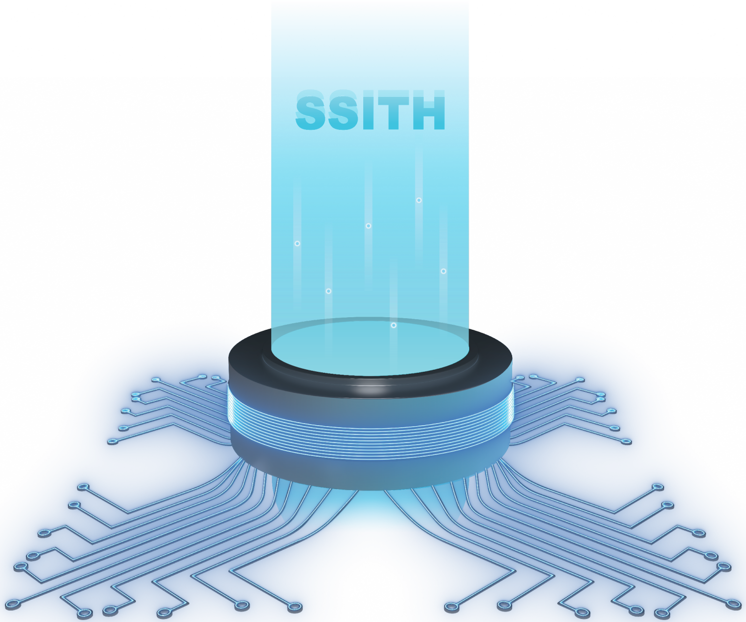
The SSITH Program:
System Security Integrated through Hardware and Firmware

- DARPA MTO program
- started in Dec 2017, ~4 year POP
- goals:
  - to develop hardware architectures to actively prevent security exploitation of badly written software (TA-1)
  - to develop tools/techniques for measuring the security impact of TA-1 architectures (TA-2)



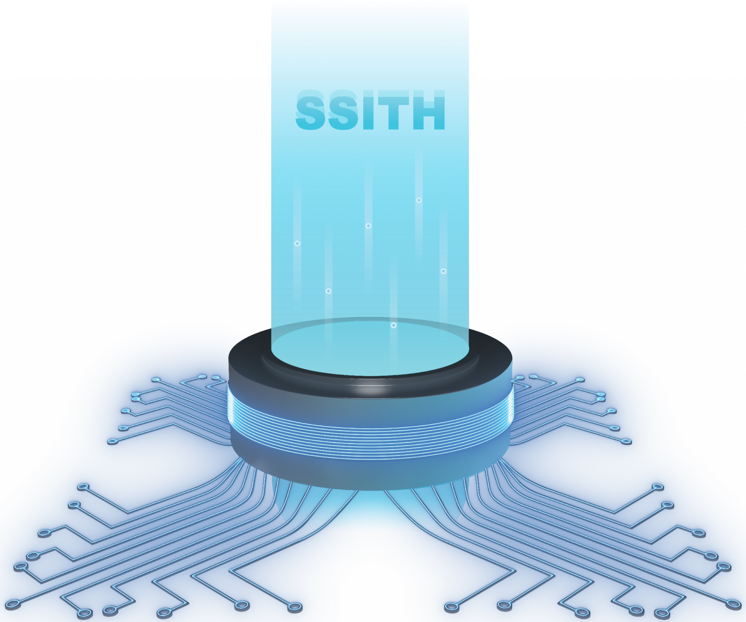
The SSITH Program: System Security Integrated through Hardware and Firmware

- TA-2 (Galois) also responsible for developing:
  - government-furnished baseline, unsecured SoCs for TA-1 use
  - demonstration applications for SSITH technology
  - a way to enable larger-scale security research on SSITH hardware architectures



The SSITH Program: System Security Integrated through Hardware and Firmware

- TA-2 (Galois) also responsible for developing:
    - government-furnished baseline, unsecured SoCs for TA-1 use
    - demonstration applications for SSITH technology
    - a way to enable larger-scale security research on SSITH hardware architectures



Outline of this Talk

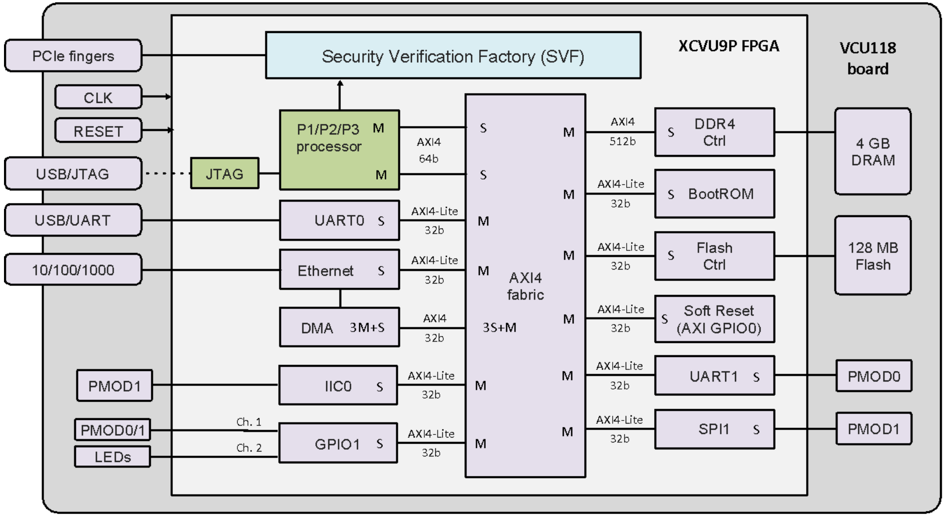
- Overview of SSITH GFE
- Moving GFE to the Cloud
  - AWSteria
  - ***FireSim***
- FETT Bug Bounty
  - Infrastructure
  - Summer 2020 Results
- Future Directions
- Current R&D at Galois

The SSITH GFE Systems-on-Chip

- program requirements: 3 different “classes” of RISC-V CPU represented in SoCs
  - 32-bit microcontroller for embedded applications
  - 64-bit Unix-capable processor supporting only in-order execution
  - 64-bit Unix-capable processor supporting out-of-order execution
- SSITH TA-1 performers use two hardware description languages for their work: *Bluespec SystemVerilog (BSV)* and *Chisel*
- therefore, 6 different SoCs—one of each class with each HDL

The SSITH GFE Systems-on-Chip

- originally designed to run in emulation on a Xilinx VCU118
- UltraScale+, 4 GB RAM, 128MB Flash, Ethernet, PCIe, other I/O
- 32-bit SoCs support FreeRTOS and bare metal
- 64-bit SoCs support Linux and FreeBSD
- but… VCU118s are very expensive (~$8000), making larger-scale SSITH security research impractical



Moving SSITH SoCs to the Cloud

solution: a *cloud-hosted* version of the GFE — *CloudGFE*

- available on-demand, from anywhere, without expensive dedicated hardware
- scalable to many simultaneous researchers and SSITH hardware implementations

implementation on Amazon Web Services EC2 F1

- exactly the same UltraScale+ FPGA as the VCU118
- but… designed for FPGA acceleration of computations running on AWS, *not* for emulating entire systems

CloudGFE Challenges

- to function properly on F1, CloudGFE uses virtualized peripherals within AWS infrastructure rather than physical ones on VCU118
- five main peripherals needed: UART, Ethernet, filesystem, random number generator, debugger interface
- different approaches required for Chisel/BSV hardware designs:
  - Berkeley FireSim-based for Chisel
  - Connectal and VirtIO for BSV=>AWSteria
- additional coherency challenges in the F1 infrastructure, compared to the VCU118 development board's dedicated memory

FETT: Finding Exploits to Thwart Tampering

- one motivation for CloudGFE:
  run a bug bounty program
- challenge: how to let security researchers who are not associated with the SSITH program attempt to exploit SSITH-protected systems…
  - in a controlled environment where useful data can be gathered
  - while keeping researcher identities private as much as possible

https://fett.darpa.mil/

FETT: Finding Exploits to Thwart Tampering

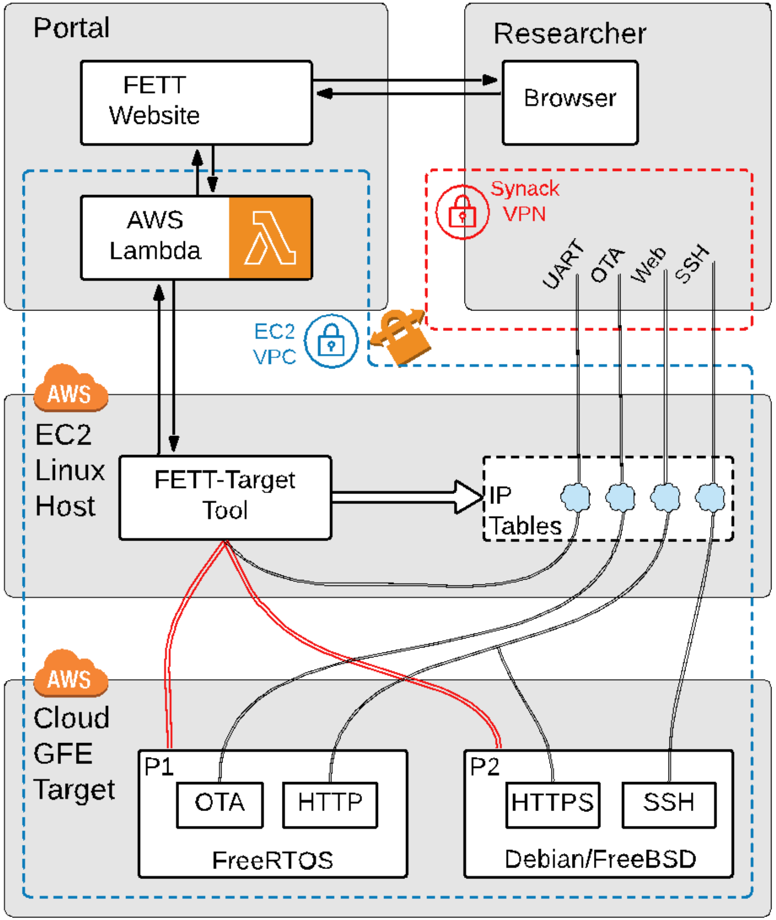
- **DARPA's first ever bug bounty program**
- Crowd-Sourced Red Team: stress-test SSITH using real world cyber exploits executed by white hat hackers
- FETT's goal was to validate
  - security IP in hardware (TA-1)
  - approach to security analyses (TA-2)

Final Results	
587 researchers	13,171 hours spent attacking
10 vulnerabilities found	983 instances launched

Lockheed-Martin 32-bit Microcontroller Instance	• **IoT-inspired over-the-air update client running on FreeRTOS**
University of Michigan 32-bit Microcontroller Instance	• **COVID-19 medical records database server running on FreeRtos**
Lockheed-Martin 64-bit CPU Instance	• **Voter registration system** • **Debian Linux with assorted userland applications**
MIT 64-bit CPU Instance	• **AES engine in secure enclave** • **Password authentication module in secure enclave** • **Debian Linux distribution**
SRI/Cambridge 64-bit CPU Instances	• **Voter registration system** • **FreeBSD with assorted userland applications**

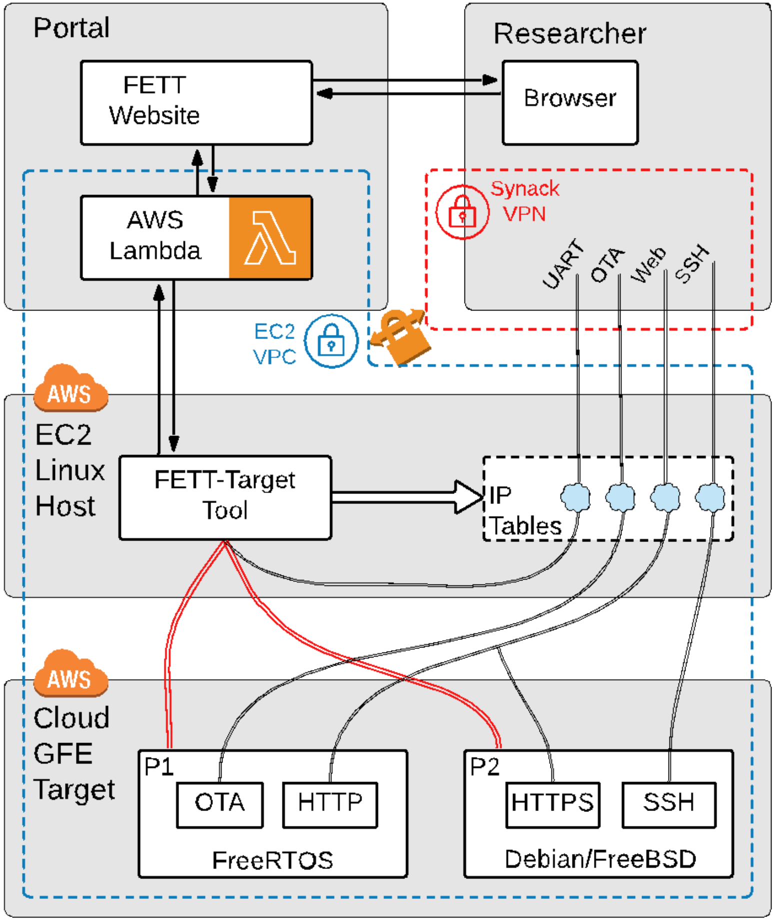
FETT Infrastructure

- FETT system architecture controls access to SSITH hardware using AWS Virtual Private Cloud
- Synack, a reputable crowdsourced security company, handles researcher registration and vulnerability reporting: the “Synack Red Team” (SRT)
- Synack provides SRT researchers a VPN, which we connect to AWS VPC to access CloudGFE targets



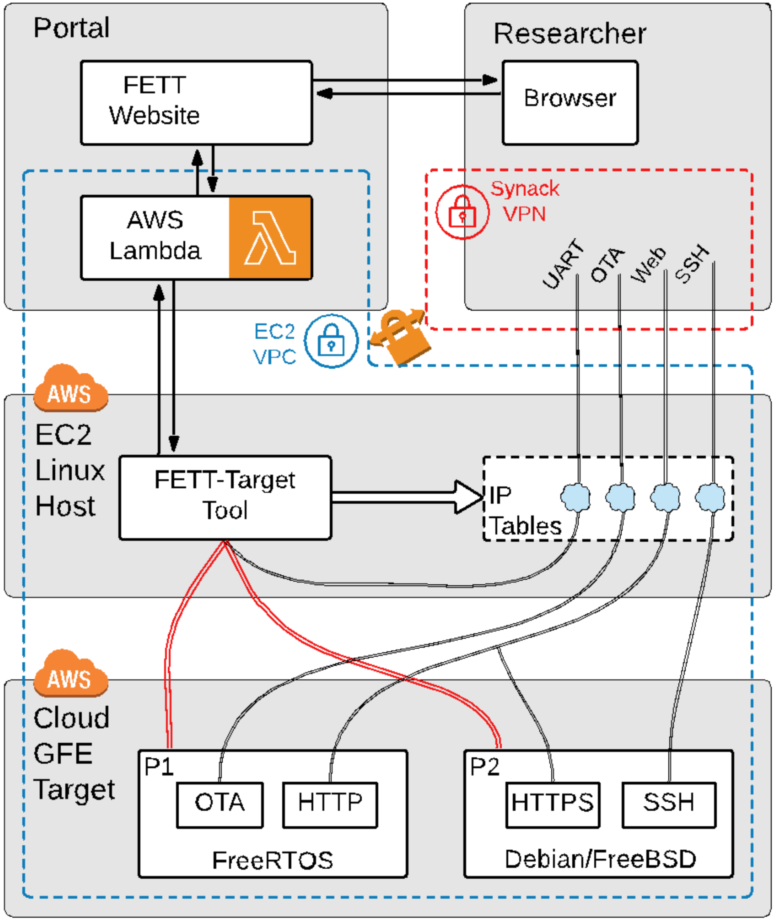
FETT Infrastructure

- to spin up a new CloudGFE target, the researcher…
  - logs into the FETT website with credentials provided by Synack
  - chooses a target type, which spins up the FETT-Target tool (or FETT tool) on a Linux host
- FETT tool handles all the details of spinning up the CloudGFE target and its network configuration



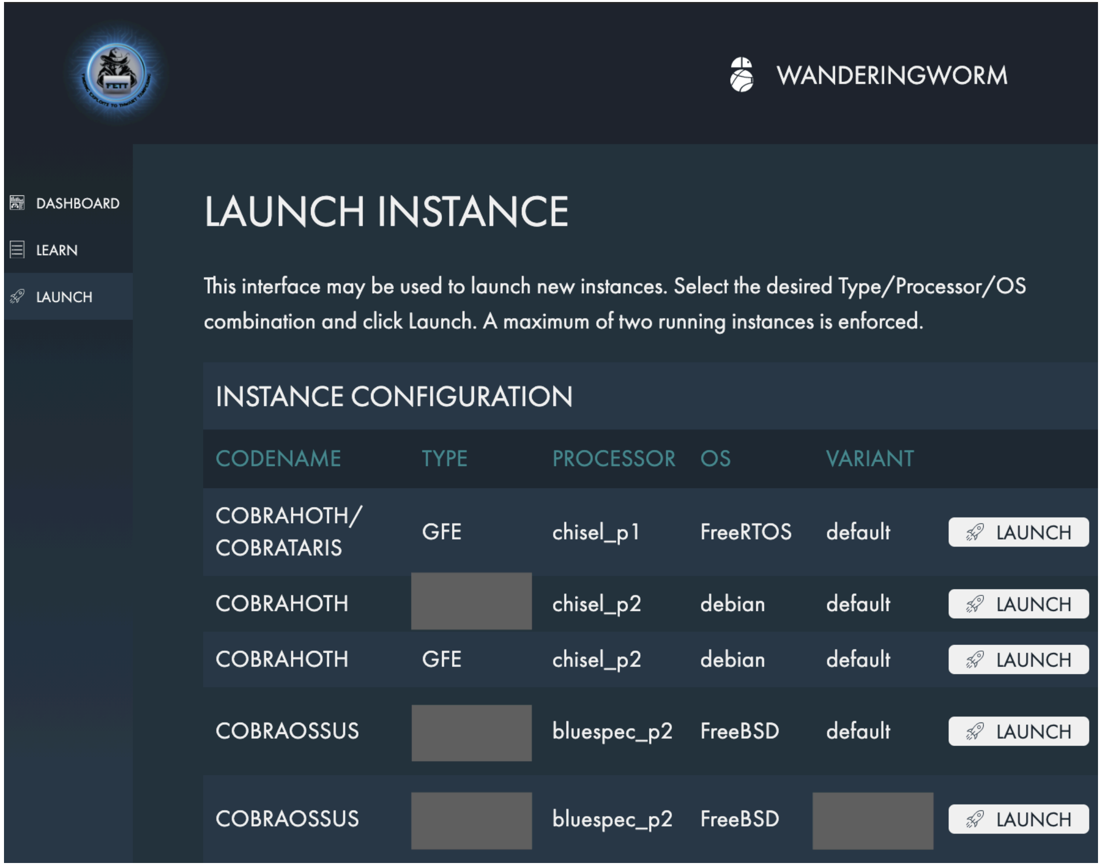
FETT Infrastructure

- available access points for the researcher depend on choice of 32-bit (P1) or 64-bit (P2) target
    - P1: UART, OTA, Web
    - P2: UART, Web, SSH
- ssh credentials were provided for the P2 target



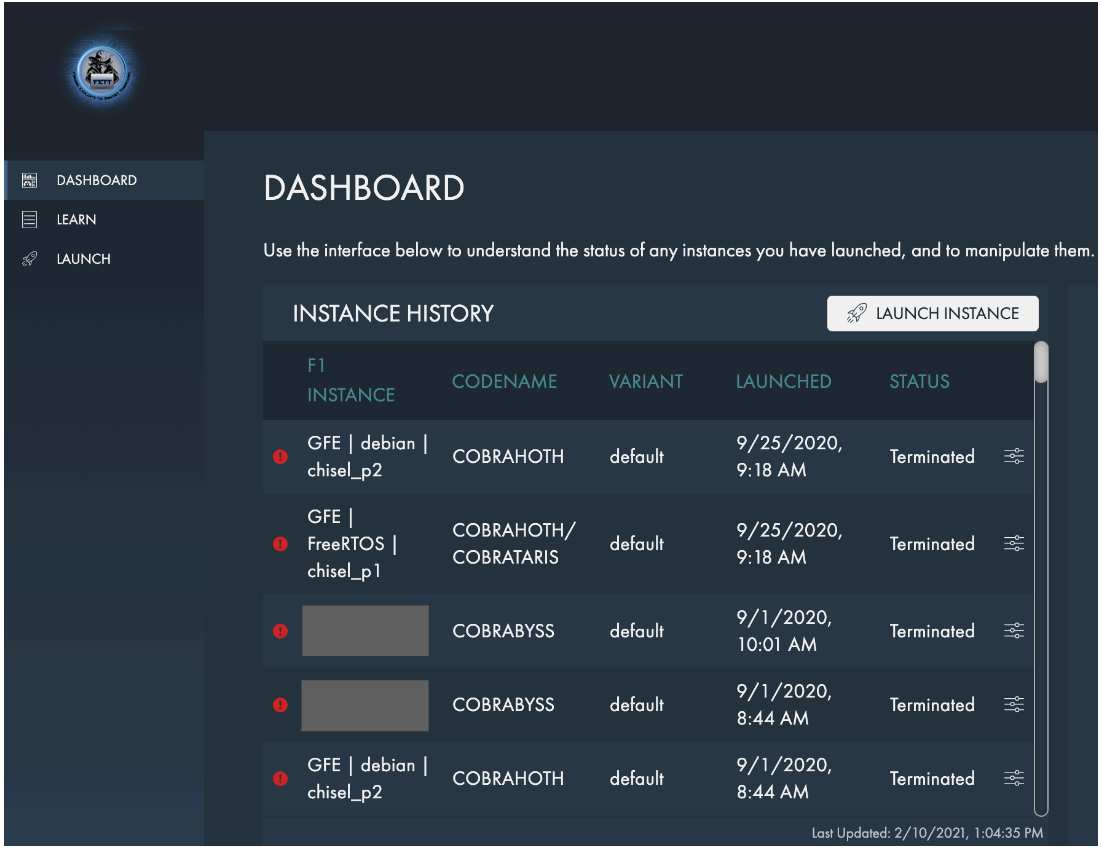
FETT Portal

- “launch” view on the FETT portal for researchers
- note the researcher’s “alias” (upper right) and the instance code names (lower left)
- some instance types/ variants redacted



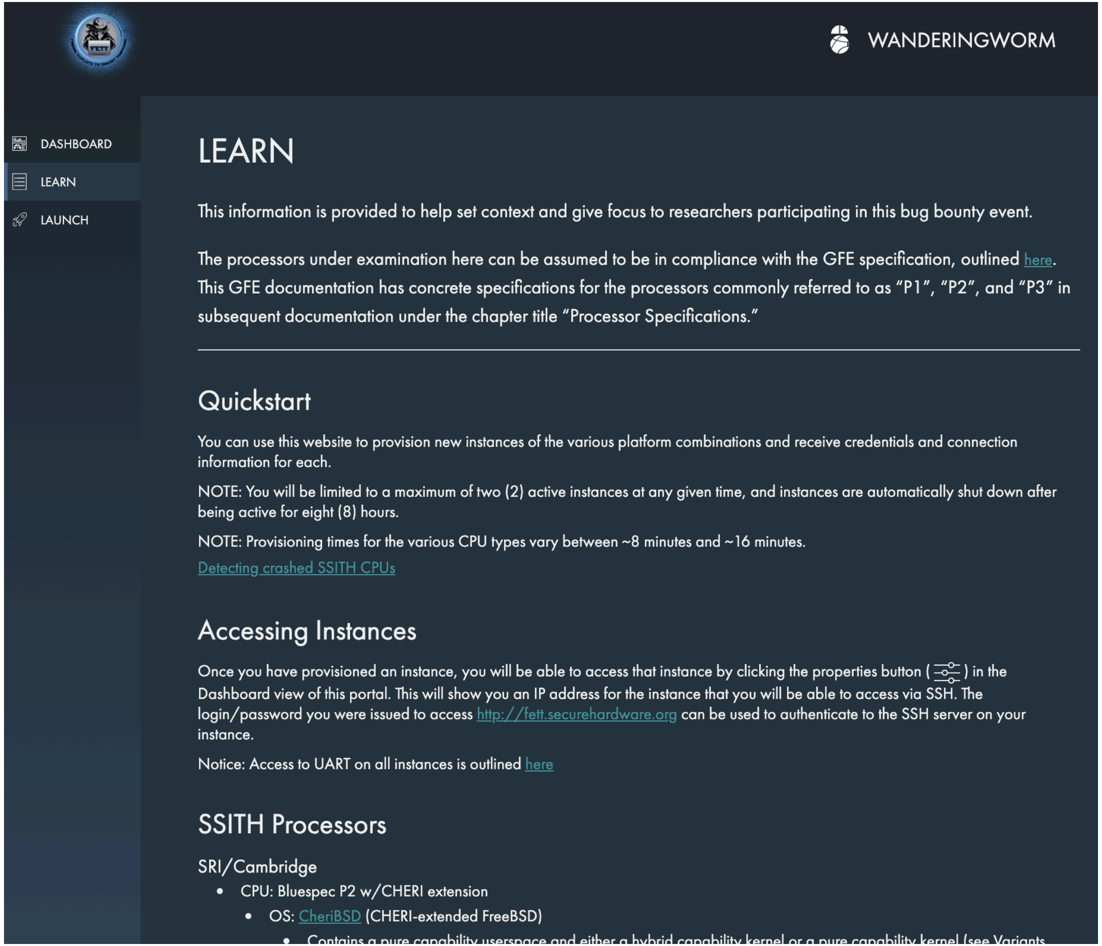
FETT Portal

- “dashboard” view gives an overview of the researcher’s running and terminated instances
- some instance types/ variants redacted



FETT Portal

- “learn” view gives information about the GFE and the various SSITH processors
- helps researchers focus their efforts
- most of the target were completely open source, including design documents, technical reports, and published papers

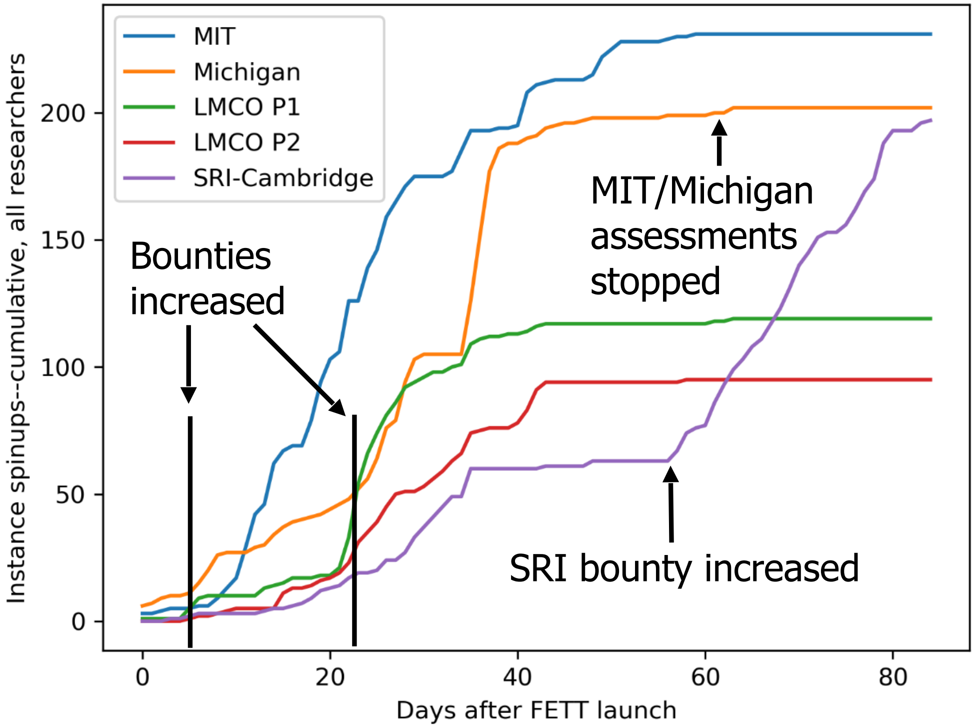


Summer 2020 FETT Results

- contest ran 3 months, from mid-July through mid-October
- included SSITH processors from 4 research teams—Lockheed Martin, MIT, SRI/Cambridge, and the University of Michigan
- more than 580 cybersecurity researchers participated
- nearly 1,000 individual SSITH processor instances started
- over 13,000 aggregate hours of hacking and analysis
- SRT disclosed 10 valid vulnerabilities: 7 “critical” and 3 “high”
- most vulnerabilities had to do with weaknesses in interactions between SSITH hardware, firmware, and software
- 3 vulnerabilities patched (and patches verified) during the contest

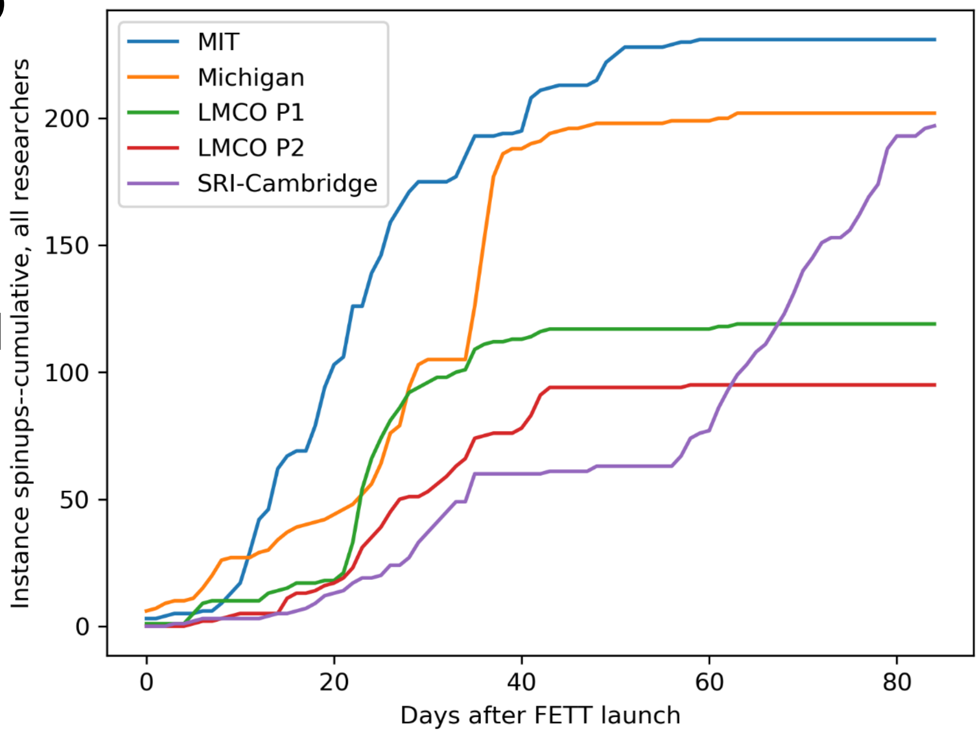
Researcher Participation Timeline—Bounties

- Day 0: bounties set at 2x value of typical ‘high vulnerability’ payout
- Day 7: any in-scope vulnerability rewarded at ‘critical vulnerability’ payout level; plus publicly announced $50,000 bonus for a vulnerability on all 4 architectures
- Day 23: bounties set at 3x value of typical ‘high vulnerability’ payout
- Day 55: bounty on SRI/Cambridge instances increased to 8x value of typical ‘high vulnerability’ payout
- began to narrow focus
- Day 61: Michigan and MIT assessments stopped



Researcher Participation Timeline—Missions

- Day 22: Mission targeting LMCO P1 TFTP server
- Day 23: Missions targeting LMCO P2 protections
- Day 28: 'High-value missions' campaign launched against all targets
- Day 35: SQLi-to-RCE launched against Michigan P1 instance
- Day 57: Start of multiple campaigns targeting SRI/ Cambridge protections



Final FETT Results

10 valid bugs were reported:

1. MIT: Security Monitor fails to validate thread owner
2. MIT: Security Monitor integer overflow when calculating enclave memory requirements
3. MIT: Security Monitor fails to sanitize memory at exit
4. LM: Failure to activate HARD pipelines at program start
5. LM: HARD pipeline fails to protect against global array overwrites
6. LM: HARD pipeline global array protections reliant on GP-relative addressing
7. MIT: Security Monitor allows manipulation of enclave handler
8. SRI/Cambridge: Capabilities not cleared with dlmalloc realloc()
9. SRI/Cambridge: vm_fault_quick_hold_pages() does not check capability bounds
10. SRI/Cambridge: varargs protections not implemented for CHERI RISC-V

Hardware vs. Software Security Researchers

- Synack researcher skillsets heavily skewed toward software evaluation
- typical engagement for bug bounty is a red-team evaluation of a network-facing application stack

- 7 out of 10 reported bugs were flaws in security-critical software that used security resources provided by hardware
- the affected TA-1 teams provided full design information for both hardware and software
- research included formal verification of hardware components; hardware may have been a harder target

Hardware vs. Software Security Researchers

- 3 out of 10 bugs were in the hardware, but were described in terms of software behavior
- from the standpoint of the bug reports, hardware was a mysterious black box that influenced the software operation

- researchers struggled to find meaningful attacks for microcontroller-class processors
- microcontroller interaction somewhat constrained in the virtualized environment
- multi-process operating system on the application processors provided a more flexible evaluation environment

Future Directions: FETT Infrastructure and Tools

- the FETT tool can be extended beyond the SSITH program, and even beyond RISC-V
    - abstraction in the tool makes adding new target hardware, OS, processor architecture straightforward
    - current tool supports not just VCU118 and AWS F1, but also QEMU for emulation without dedicated hardware
    - extensions to manage multiple targets simultaneously, for future demonstration applications
- open source release will provide a good starting point for anyone to run a hardware bug bounty
- the entire FETT infrastructure was also open sourced

Future Directions: Hardware R&D at Galois

- Galois continues to pursue hardware-related R&D for the USG
- our current projects include:
  - HIPERSPACE: developing a high-assurance, ultra-high performance cryptographic accelerator for space-based applications
  - HARDENS: developing a high-assurance nuclear power plant protection system that is FPGA-based and includes a formal assurance case that spanning hardware and software
  - SAW: developing a hardware backend for our SAW tool to reason about hardware designs (thus, now SAW can reason about LLVM, JVM, VHDL, Verilog, SystemVerilog, and Bluespec SystemVerilog)
  - BASILISC: developing a high-assurance fully homomorphic encryption ASIC accelerator