Healthcare/IoT Cybersecurity Testbed: Milestone 1

By: Justin Bower, Nathan Maloney, and Ipule Pipi

Instructors/Advisors: Dr. Sudhakaran and Dr. Aydeger

FLORIDA TECH

FLORIDA'S STEM UNIVERSITY

Milestone completion

Task	Completion %	Justin	Nathan	Ipule	To-Do
Tool Investigation	100%	33%	33%	33%	Nothing
Tooling Demos	66%	33%	33%	0%	Create demo web page
Tooling Integration	35%	10%	25%	0%	Integrate remaining tools from investigation
Custom Scripting	50%	50%	0%	0%	Integrate existing scripts and create more
Demo/Testing Environment Integration	50%	25%	25%	0%	Integrate container and web page to AWS

Tool Choices

- Cloud Provider: AWS
- API: API Gateway
- Front End: NextJs (React Framework)
- Database: NoSQL
- IDE: VSCode w/ DevContainer

Demo Progress

- AWS has been setup with a base level allowing infrastructure to be built and tested
- Containers have been created and run with tools included
- Front end is still in development and should be completed soon

Integration

- The beginning of integrating the various tools being used is in progress
- These will be integrated into existing work before being put together to begin the integration of each section together
- A single demo/test container has been successfully created in the cloud so far

Web Interface Progress

- **Static Next.js Export:** Compiling our UI into raw HTML/JS/CSS eliminates the need for a 24/7 Node.js server, slashing operating costs and removing server-side attack vectors.
- **S3 + CloudFront Hosting:** Replaces traditional VMs with serverless storage and global CDN distribution, ensuring the marketplace scales infinitely and loads instantly for students.
- **Native AWS Security (OAC):** By keeping the frontend within AWS, we use **Origin Access Control** to keep S3 buckets private—ensuring the source code is only accessible through our secure CloudFront gate.
- **Security & Gateway**
- **Integrate Cognito Auth:** Replace external auth middleware with **AWS Cognito**, providing a secure, serverless user directory that integrates natively with our API Gateway bouncer.
- **Unified Network Backbone:** Operating on a single cloud allows us to use **IAM Roles** instead of risky API keys, keeping all backend communication private and off the public internet..

Docker, Scripting, and RE Tooling Progress

- We are working to integrate multiple tools using Python as a common framework
- Ghidra as recently as last year (version 12) implemented python-based scripting so that is in progress as a means for RE
- More docker containers/environments will be tested to find the best overall environment
- Potential for functionality to switch between different container environment to optimize for different tasks
- Future additions to supported file formats may include a file-format property selection window, for custom file formats

Future Custom File Format Parameters

Endianness - Little endian versus Big endian byte order

Encoding - Other parameters like base 64, hex, and byte swapping

Integer v.s. Floating Point - Basid by which math is performed

Primarily Supported File Formats

.bin – Binary files are often used for firmware, raw data, or custom formats: common in embedded systems and low-level data storage.

.asm – Assembly language files: while these are human-readable, reverse engineering may involve analyzing the binary form of compiled code.

.s – Assembly source files: typically used in low-level programming and can be part of the reverse engineering process when disassembling executables.

.exe, **.dll**, **.so**, **.dylib** – Executable and library files: frequently reverse-engineered to understand behavior, extract functionality, or analyze malware.

Current Script Templates

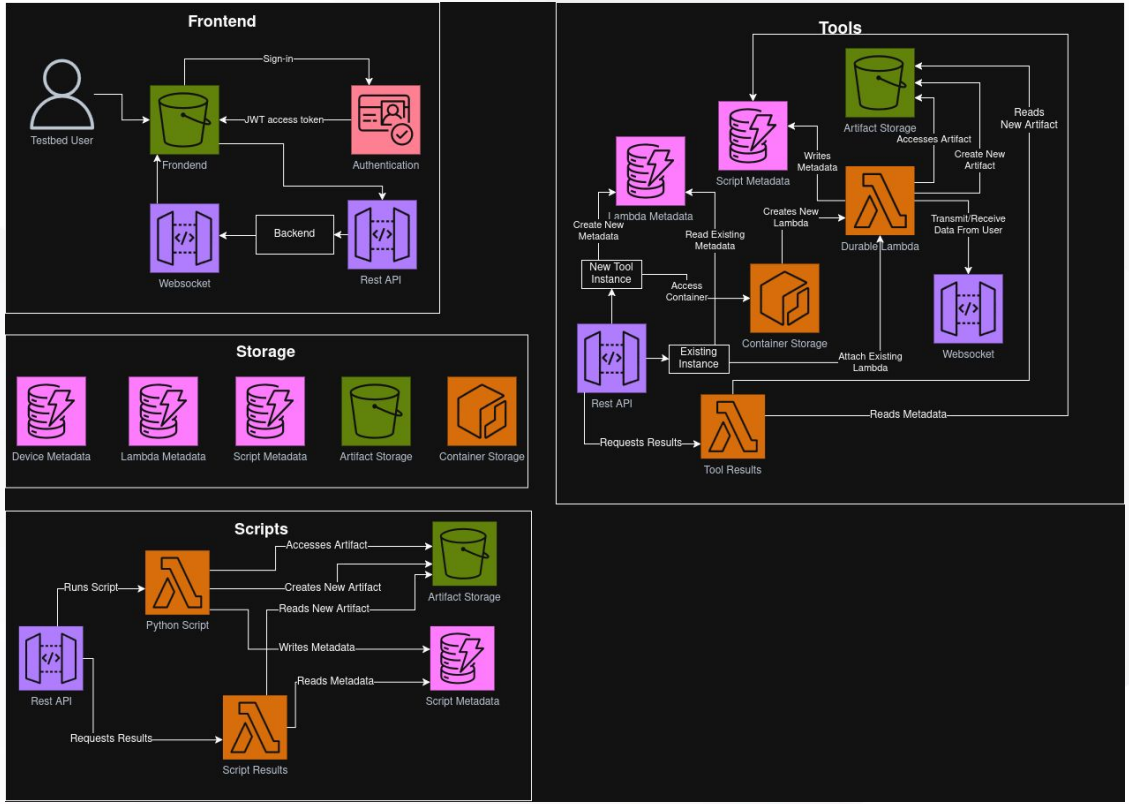
File Utilities

analyze_entropy.py - uses binwalk to examine file entropy
lambda_invoke.py - assists in lambda code execution
parse_firmware.py - parses firmware files over ssh or http

PyGhidra Utilities

unsanitized_text_entries_scanner.py - scans files for poor text entry handling for potential ret2* exploitations
buffer_overflow_scanner.py - scans for potential buffer overflow exploits
rop_gadgets_scanner.py - scans for ROP gadgets for ropper exploitation

Infrastructure Design



FLORIDA TECH
FLORIDA'S STEM UNIVERSITY

Milestone 2 plans

Task	Justin	Nathan	Ipule
Tooling Demos	Nothing	Nothing	Create web page demo using investigated tools
Integration Testing	Work to integrate container into AWS	Ensure cloud resources are properly running	Work to integrate frontend and backend with AWS
Device Examination	Examine chipsets, documentation, and plan device mounting	Nothing	Nothing
Tooling implementation	Migrate containers to AWS ECS to allow for deployment and access	Deploy and test database structures such as DynamoDB and S3	Add in support on front and backend for users to access currently running containers/services
Scripting	Examine interfaces to look for most common RE vectors	Investigate integrations of across multiple device interfaces	Create backend microservices
Infrastructure development	Assist in scripting hooks into infrastrcutures	Create scripts to automate infrastructure creation and security	Nothing

Thanks for listening!
Questions?

FLORIDA TECH
FLORIDA'S STEM UNIVERSITY