Pre-Fuzzing Analysis of Embedded System binaries with Ghidra SRE

Student: Ron Ng Jian Ying
Supervisor: Professor Liu Yang

**PROBLEM**

Only two solutions currently exist in the market for coverage guided fuzzing of embedded systems – inline instrumentation which requires source codes or fuzzing within a virtual machine a la QEMU.

Fuzzing in a virtual machine is slow and inline instrumentation without source code is complex and inefficient.

**MOTIVATION**

Detour/Hooking instrumentation can be used instead. Suitable portions of instructions in code can be identified as relocatable.

This allows instructions to be copied into a memory segment with instrumentation code added. Original instruction flow remains undisturbed.

**SOLUTION**

Ghidra was used to analyse binaries and its API used to create a Python script that retrieves a list of compatible address locations for hooking. These are known as gadgets.

The gadget list is then passed on to BSFuzz which is a coverage-guided embedded fuzzer developed by NTU CSL. BSFuzz uses the list of gadget addresses to know which function to hook at runtime.