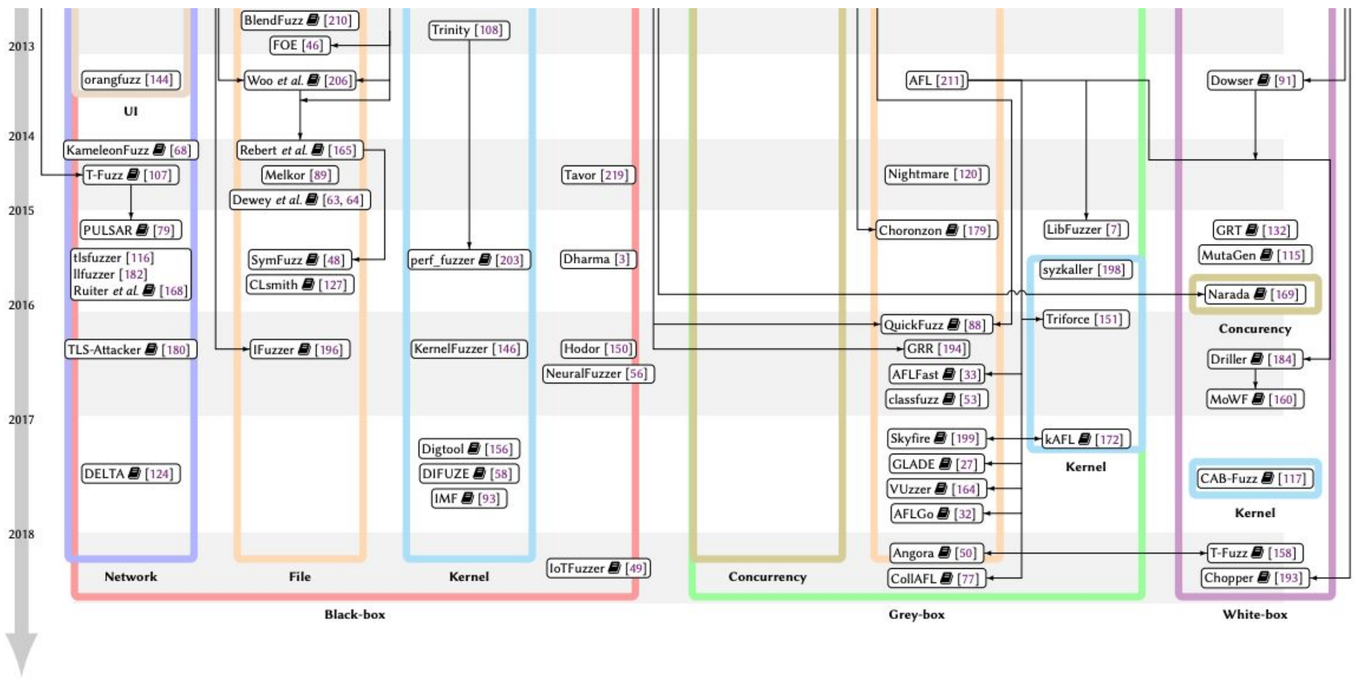
RSAC Security Scholar

Taint-Driven Embedded Software Fuzzing

RSAC Security Scholar Melisa Savich
New York University

NYU

Problem Statement and Goals



[1] Genealogy of recent fuzzers, traced back to Miller et al.'s work.

Problem

- Embedded systems are becoming increasingly difficult to test due to the range of complexity and purposes of these devices.
- To the best of our knowledge, an efficient, correct, and easy approach to testing the security of embedded devices has yet to be explored.

Goal

Fuzzing is one technique used to evaluate the correctness of software. We apply the techniques utilized in popular software fuzzers (i.e. BuzzFuzz, Driller, VUzzer, AFLGo, Angora) to fuzz embedded systems.

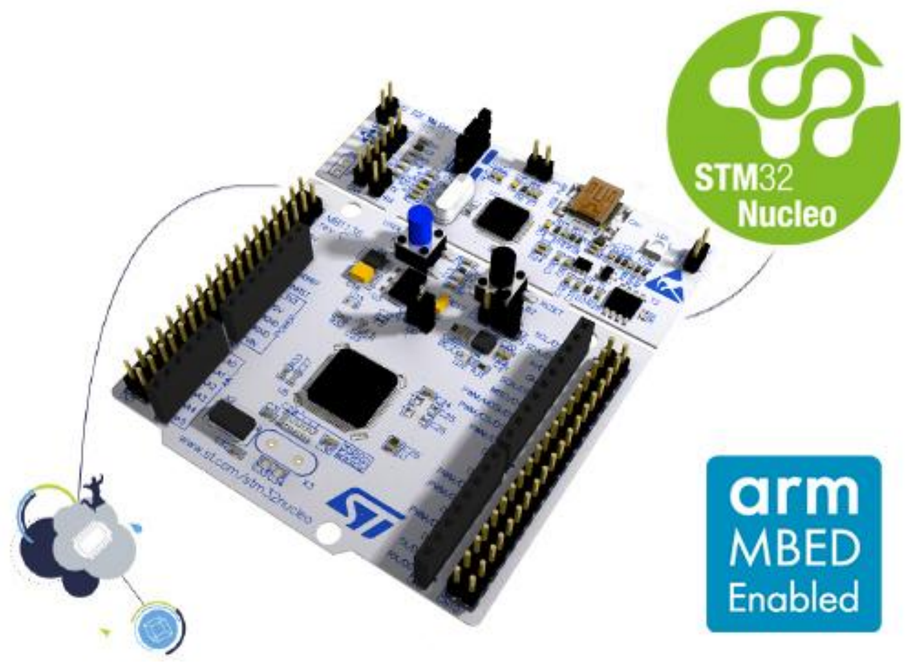
Approach

Taint Analysis

1. Write a taint analysis PANDA plugin.
2. Use the taint plugin during PANDA replays to gather taint propagation information.

Fuzzing

3. Use gathered taint information to make informed decisions on which areas of code to fuzz.
4. Find deep and meaningful bugs.



[2] Currently testing with the STM32 Nucleo board.

Results

[3] Wago PFC100 Controller.

As of this writing, we have developed the PANDA plugin responsible for taint. During a replay, the plugin applies taint to specified addresses of memory, and returns a log of meaningful taint points. These taint points will be used to fuzz interesting parts of memory. We plan on further developing and testing this fuzzer on other embedded devices, such as the Wago PFC100 Controller.

Acknowledgements: Brendan Dolan-Gavitt

References:

[1] https://arxiv.org/pdf/1812.00140.pdf

[2] https://www.st.com/en/evaluation-tools/nucleo-l152re.html

[3] https://www.wago.com/us/discover-plcs/pfc100