What is Zephyr

“The Zephyr™ Project is a scalable real-time operating system (RTOS) supporting multiple hardware architectures, optimized for resource constrained devices, and built with safety and security in mind.”

- https://www.zephyrproject.org/
- https://docs.zephyrproject.org/
- https://github.com/zephyrproject-rtos/zephyr/
Zephyr Security Subcommittee

- One person from each platinum member, silver members by invitation
- A security Chair elected by the subcommittee
  - Responsible for running bi-weekly meeting
  - Sets agenda and takes notes
- A security architect elected by the subcommittee
  - Responsible for overall project security
  - Maintainers to seek signoff for significant changes to LTS codebase
What is done

- Have security subcommittee, meets bi-weekly
- Created secure coding practices documentation
- Zephyr registered as a CVE Numbering Authority (CNA) with MITRE
- Subcommittee has vulnerability response criteria publicly documented (and has addressed issues)
- Passing CII Best Practices gold
- Using automation to prevent regressions
Code repositories: LTS

● (Long Term Stable) It is:
  ○ Product Focused
  ○ Current with latest Security Updates
  ○ Compatible with New Hardware: We will make point releases throughout the development cycle to provide support for new hardware
  ○ More tested: Shorten the development window and extend the beta cycle to allow for more testing and bug fixing

● It is not:
  ○ A feature-based release: focus on hardening functionality of existing feature, versus introducing new ones
  ○ Cutting edge
Code repositories: Auditable

- Stable branches off of LTS
- Subset of the code
- Frozen in time
- Intended to be "certifiable"

A starting point for a product that needs certification
Ongoing work

- Described in project documents
- Code guidelines
- How to report vulnerabilities
- Process for a security bug
- JIRA instance to manage bugs during embargo
Ongoing: Code guideline

- A coding guideline tailored for Zephyr
  - MIRSA C:2012, with Amendment 1 as a baseline
  - SEI CERT C and JPL used as reference
- Help to address security concerns across the project
- Code easier to maintain and review
- Can be enforced using static analysis tools
Ongoing example: randomness

- Open PR updating entropy/random framework
- Addresses multiple security issues
- Discussed within subcommittee meeting
- In this case, worked on by someone in team
- Goal: clearer API and docs so it is easier to do things right
Aims: Crypto Drivers

● Same API for different implementations
  ○ Provided by hardware, e.g. Atmel ATAES132A
  ○ Provided by software
    ■ TinyCrypt small footprint
    ■ mbed TLS feature rich

● Ongoing discussions, we are evaluating possibilities
Aims: FIPS 140-2/3

● Common for cryptographic modules
● Generally, certifies products
● But, certification of auditable helps that process
● Focus is on crypto operations

● FIPS 140-3 is effective Sept 22, 2019, too late to be in the slides, but I should have something to say 😊
Aims: Secure Boot

● Today:
  ○ **MCUboot** supported by Zephyr
    ■ Bootloader with revertible upgrades
    ■ Sign images against public key in ROM
    ■ Used by Trusted Firmware M (TF-M) as part of its secure boot story

● Future
  ○ Upgrade story
  ○ **SUIT**
  ○ Richer key infrastructure
Aims: Fuzzing

- Most fuzzing work done on bigger systems than Zephyr targets
- Research into QEMU-based fuzzer
- Other possibilities:
  - Existing fuzzers often assume lots of memory
  - Posix Native port can help with some areas
- Open area for research
Aims: Documentation

● Have existing threat models
● We’ve learned since then

● Future
  ○ Reorganize existing threat model to read more like a thread model
  ○ Define threat models for other contexts/applications
Thank you

Join Linaro to accelerate deployment of your Arm-based solutions through collaboration

contactus@linaro.org