Integrity Protect Workloads with Mushroom

Outline

- Goals
- Demo
- High-level Architecture
- Kernel
- Supervisor
- VMM
- Non-Goals
- Attestation

whoami

- Tom Dohrmann
- Developer/Security Researcher/Reverse Engineer
- github.com/Freax13
- twitter.com/13erbse
- blog.freax13.de
- ❤️ Rust 🦀

Goals

- Run Linux programs.
- Transform an input file into an output file.
- Prevent tampering during processing.

Use case:

- Remote code compilation on an untrusted host

Demo

```
freax13@server:~/mushroom/host$ export SUPERVISOR="../tee/target/supervisor/supervisor-release/supervisor"
freax13@server:~/mushroom/host$ export KERNEL="../tee/target/x86_64-unknown-none/kernel/kernel"
freax13@server:~/mushroom/host$ export INIT="../tee/target/x86_64-unknown-linux-musl/release/init"
freax13@server:~/mushroom/host$ cat test.c
int main() {
        printf("Hello FOSDEM!\n");
}
freax13@server:~/mushroom/host$ mushroom run --input test.c --output test --attestation-report report.bin
2024-02-01T19:47:53.468192Z  INFO mushroom: launched num_launch_pages=16915 num_data_pages=16913 total_launch_du
ration=17.263863671s
2024-02-01T19:47:54.771201Z  INFO mushroom: finished
freax13@server:~/mushroom/host$ mushroom verify --input test.c --output test --attestation-report report.bin
Ok
freax13@server:~/mushroom/host$ chmod +x test
freax13@server:~/mushroom/host$ ./test
Hello FOSDEM!
```

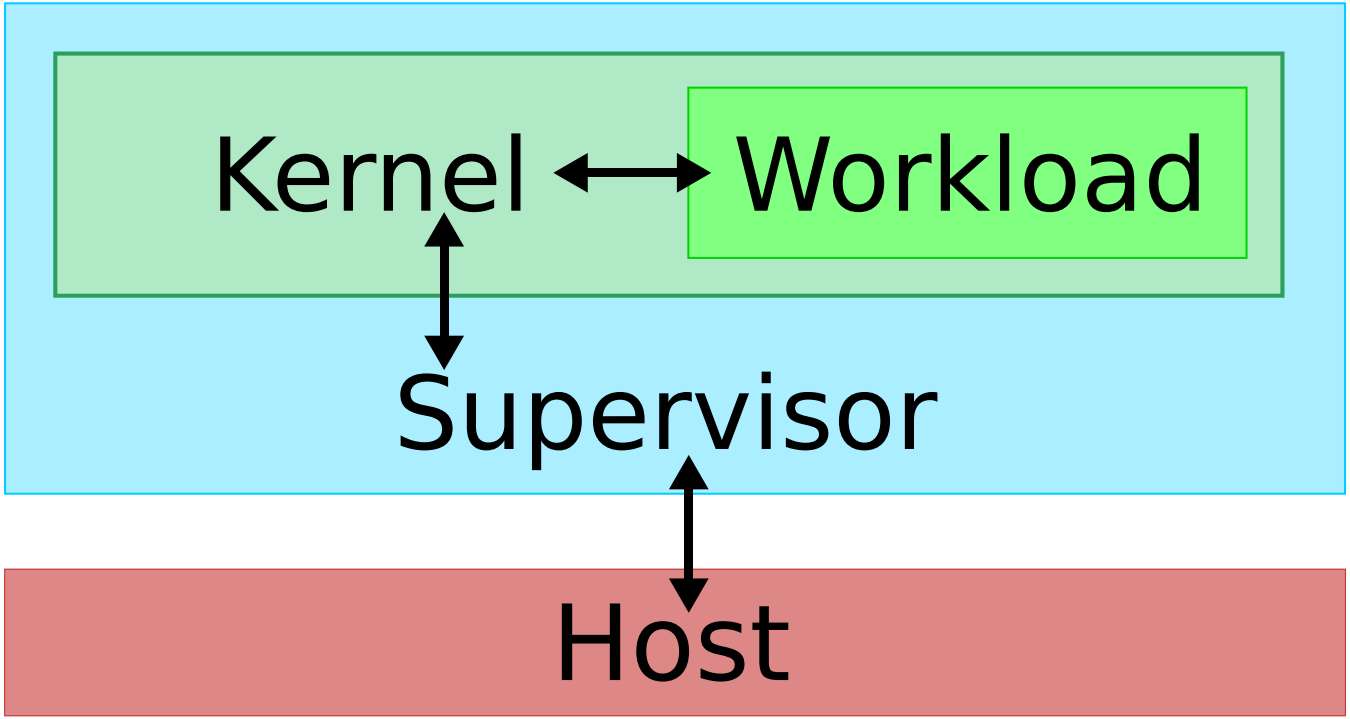
Architecture

Kernel

- ~15k LoC

Supervisor

- 2.5k LoC



Kernel

- Written in Rust
- Implements the Linux syscall interface.
- Currently implements 83 syscalls[1].
- Supports 32-bit and 64-bit binaries.
- Built for and optimized for minimal host interaction.
  - No real device drivers.

[1] Not all flags are implemented for all syscalls.

Supervisor

- Handles communication between host and kernel.
  - Loads workload input
  - Memory hot-plug
  - Scheduling
  - Emits workload output
- Host ↔ Kernel isolation is enforced using virtual top of memory (vTOM), #VC Reflect & virtual machine privilege levels (VMPL).
- The **only** security-critical component.

VMM

- Uses KVM API.
- Implemented in mushroom host binary.
- Very minimal implementation.
- Also supports running without SEV-SNP.

Non-Goals

- No I/O at runtime
  - No network.
  - No persistent storage.
- No complexity in the supervisor
  - No Multi-threading
  - No Heap

Attestation

- “Measurement” contains a hash of supervisor, kernel, and init binary.
  - We could be sign these in the future instead of comparing the hash directly.
- “HostData” contains a hash of input file.
  - The hash is verified by the supervisor when it loads the input.
- “ReportData” contains a hash of output file.
  - This is the only field that can be changed by the guest after it’s been launched.

Questions?

Thanks!

github.com/Freax13/mushroom