Rough outline of the project:

**Question: ‘How effective is isolation at preventing malware from propagation compared against a non-isolated honeypot?’**

1. **Honeypot containment framework for IoT: measuring the benefits of sandboxing/segmentation effectiveness vs vanilla**
   1. Create a honeypot that provides segmentation and sandboxing to prevent malware escape
      1. **Segmentation** - isolating a honeypot from the network
      2. **Sandboxing** - Ensure the honeypot is in a secure, contained environment
   2. Docker, Firejail, custom VM?
      1. **Docker** - platform to build containment environments
      2. **Firejail** - Linux based
   3. Why?
      1. Many honeypots (especially large ones like IoTPOT) attract a lot of Telnet-based bot behaviour; however a large number of honeypots lack containment to avoid real-world harm. They detect and act, but don’t contain. [link](https://www.researchgate.net/publication/303181974_IoTPOT_A_novel_honeypot_for_revealing_current_IoT_threats)
      2. Recent work on cloud deployed honeypots, providing extra layer of security when paired with segmentation [link](https://arxiv.org/pdf/2301.00710)
      3. Many existing honeypots in modern IDS’ require ‘malware escape’ and segmentation. They highlight strategies, but don’t provide implementations
   4. Methods:
      1. Contained honeypot: Inside string Linux based isolation (strict seccomp/AppArmor + Firejail or dedicated VMs + segmented virtual network and strict egress firewalling)
      2. Vanilla: Same honeypot but different device profile
      3. Run active tests, deploy both privately and publicly
   5. Tools: Docker, FireJail, QEMU/KVM VMs, nftables)
   6. Crowrie for the Honeypot

* Vanilla honeypot: Low-interaction honeypot deployed on the virtual network (inside VM) segment with no container
  + Used to mimic an IoT device to capture attackers [What is a honeypot? How honeypots help security](https://www.kaspersky.com/resource-center/threats/what-is-a-honeypot)
* Sandboxed honeypot: Identical to a vanilla honeypot, but rather deployed inside a container for high-interaction
  + Used to thoroughly and safely analyse malicious risks

Problem:

Low interaction vanilla honeypots may be considered weak in nature due to only simulating basic IoT devices, with zero in-depth analysis measures – in essence, weak simulation.

On the contrast, high-interaction honeypots (within a sandboxed environment) provide much greater security by creating an isolated environment for malicious attacks [honeypot vs sandbox](https://tssc.mx/EN/honey.php) where such are able to be evaluated and thoroughly processed. It permits for OS resources to be monitored.

The purpose of comparing is to evaluate how vanilla honeypots do not have any overhead and aren’t as undetectable by attackers.