1.1 Requirements & Constraints

1.1.1 Functional Requirements

The software our team will develop will assess the robustness of security systems that attempt to detect microarchitecture attacks. The robustness will be measured by its ability to detect microarchitecture attacks specially designed to evade detection. The software will generate these evasive adversary attacks by inserting artificial noise into the attack instructions to mimic benign power signatures and exploit the security system's underlying machine-learning model.

Five microarchitecture attack codes will be provided, and all five attacks must be able to execute without detection and without significantly slowing down the attack. The security system cannot report any higher than 20% detection certainty for the attack to be undetected. The power usage should not exceed 2x normal activity, and the attack should not surpass a 5x slower data leak rate than its non-evasive counterpart.

1.1.2 UI Requirements

Our team's program will feature a simple and user-friendly graphical user interface (GUI). The interface must allow users to do the following tasks:

1. Define the attack type
2. Include the data sets they used.
3. Upload the application's source code used to train the model
4. Upload attack source code
5. Select between different detection models
6. Run evasive power-mimicking attacks

After running an attack, the GUI must display statistics about the attack, including the data leak rate in bits per second and the security systems malware detection certainty for the selected model as a percentage.

1.1.3 Resource Requirements

The project will require specialized hardware to pull the necessary CPU power consumption data and achieve the performance needed to run the AI-based microarchitecture attack detector. Our team will be provided and required to use the following experimental setup:

- Intel Comet Lake Microarchitecture
    - CPU Model: Intel(R) Core (TM) i7-10610U CPU @ 1.80GHz
    - OS: Ubuntu 20.04 LTS
    - Linux Kernel: 5.11.0-46-generic
- Server Information
    - Nvidia GeForce RTX 3090 GPU
    - CPU Model: Intel(R) Xeon(R) Gold 5218 CPU @ 2.30GHz

1.2 Engineering Standards

Software Development Life Cycle

- **IEEE 12207:** This standard provides processes that can be employed for defining, controlling, and improving software life cycle processes within an organization or a project.

- **IEEE 1074:** This standard provides a process for creating a software project life cycle process (SPLCP) and it is primarily directed at the process architect for a given software project.

Software Testing

- **IEEE 29119:** This is a series of five standards for software testing. They define vocabulary, processes, documentation, techniques, and a process assessment model for testing that can be used within any software development lifecycle.