CyHELICS

Senior Design Team 28

Design Document

Dr. Gelli Ravikumar

Justin Templeton

Tyler Atkison

Kaya Zdan

Zach Hirst

Tommy Keeshan

Matt Nevin

sdmay24-28@iastate.edu

sdmay24-28.sd.ece.iastate.edu

December 1st, 2023 | Revision 1.0

Executive Summary

Development Standards & Practices Used

- HELICS and pandapower use an open source BSD-3 clause license.
- OpenDSS is open source, no listed license.
- MITRE ATT&CK Framework is an industry standard knowledge base for use in pentesting, gap assessments, threat intelligence/hunting, and more.
- OWASP Top 10 as a security guidance standard.
- Python is an industry standard interpreted scripting language.

Summary of Requirements

Functional Requirements:

- Use CyHELICS to combine multiple substream programs and run concurrently
- Include both power grid model analysis tools and cyber security focused programs.
- The simulation will be capable of handling multiple attack simulations, based on the OWASP top 10.
- Create a power grid with several transmission models that connect with several distribution models and demonstrate proper power flow.
- Power Grid will include multiple load types.
- The power grid interface will be able to simulate different grid set ups.
- The simulation will be tested in a VM environment.
- The simulation will be set up in a dockerized environment.

Nonfunctional Requirements:

- The interface must be easy to use for non technical users (city planners, grid designers).
- The user must be able to select how much of the grid they want to simulate an outage for, with specialized attacks for each one.
- The simulation must give feedback to the user about the state of the simulation (failed, complete, in progress etc.)

Applicable Courses from Iowa State University Curriculum

Cyber Security:	Electrical Engineering:
CPRE 230	EE 303
CPRE 231	EE 455
CPRE 308	EE 456
CPRE 489	EE 457

New Skills/Knowledge acquired that was not taught in courses

- Docker knowledge
- Flask usage
- Connecting multiple programs to accomplish one task (ie HELICS, OpenDSS, Pandapower)
- HELICS usage
- OpenDSS distribution grid design and analysis
- OpenDER electric vehicle and battery modeling and analysis
- Pandapower transmission grid design and analysis

Table of Contents

List of figures/tables/symbols/definitions (This should be the similar to the project plan)

- HELICS: a co-simulation tool that allows multiple simulators to run simultaneously and off of each others' results. This tool is necessary because otherwise, the simulators running by themselves would not accurately portray a whole and singular electric power grid.
- Pandapower: a tool that simulates the power generation and transmission of a power grid.
- DSS_python: a tool that simulates the power distribution and load characteristics of a power grid, python wrapper for OpenDSS.
- OpenDSS: an electric power distribution system simulator used to design distribution systems and simulate their usage.
- OpenDER: a package designed for OpenDSS that can model batteries and electric vehicles as load criteria for a distribution grid.
- Kali: a red team-oriented operating system that will help develop new attack methods and modules, as well as utilize pre-existing ones.
- Docker: Containerized solution to run programs on many platforms easily. Simple to create, tear down, and connect environments.
- Virtualization: Running a guest operating system on a host machine, available to Iowa State Students to safely run their programs.
- Flask: A basic python web application package that can be used with both simple and complex applications alike. It allows many other packages to be used on the python platform to run complex methods.
- Ubuntu: the most popular and flushed out linux distribution that is one of the two industry standards besides for Red Hat linux.

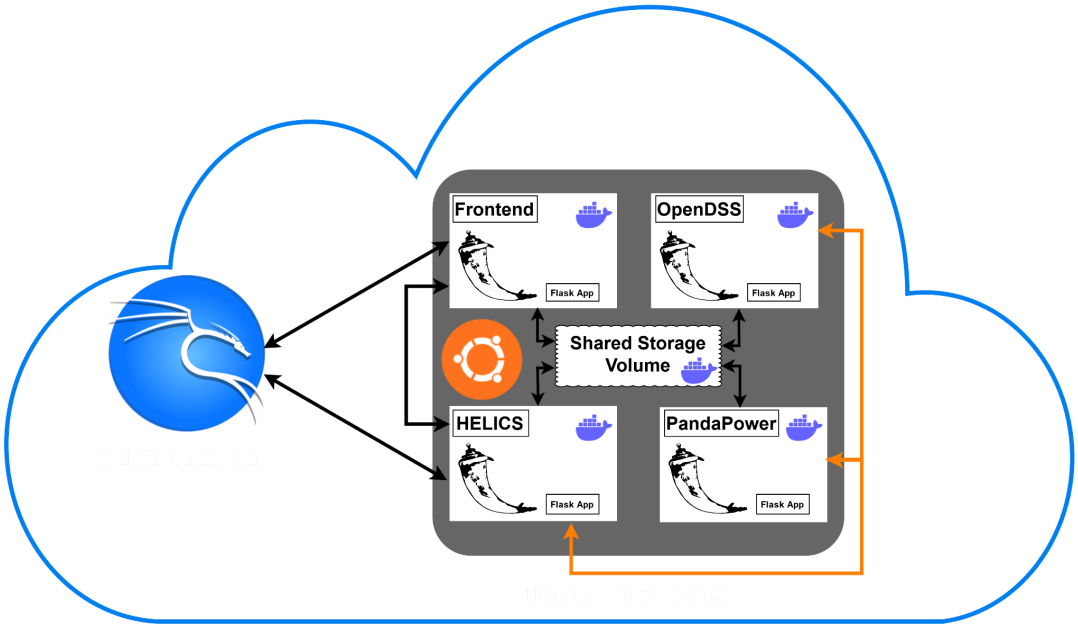


Figure 1: Basic Overview of Project

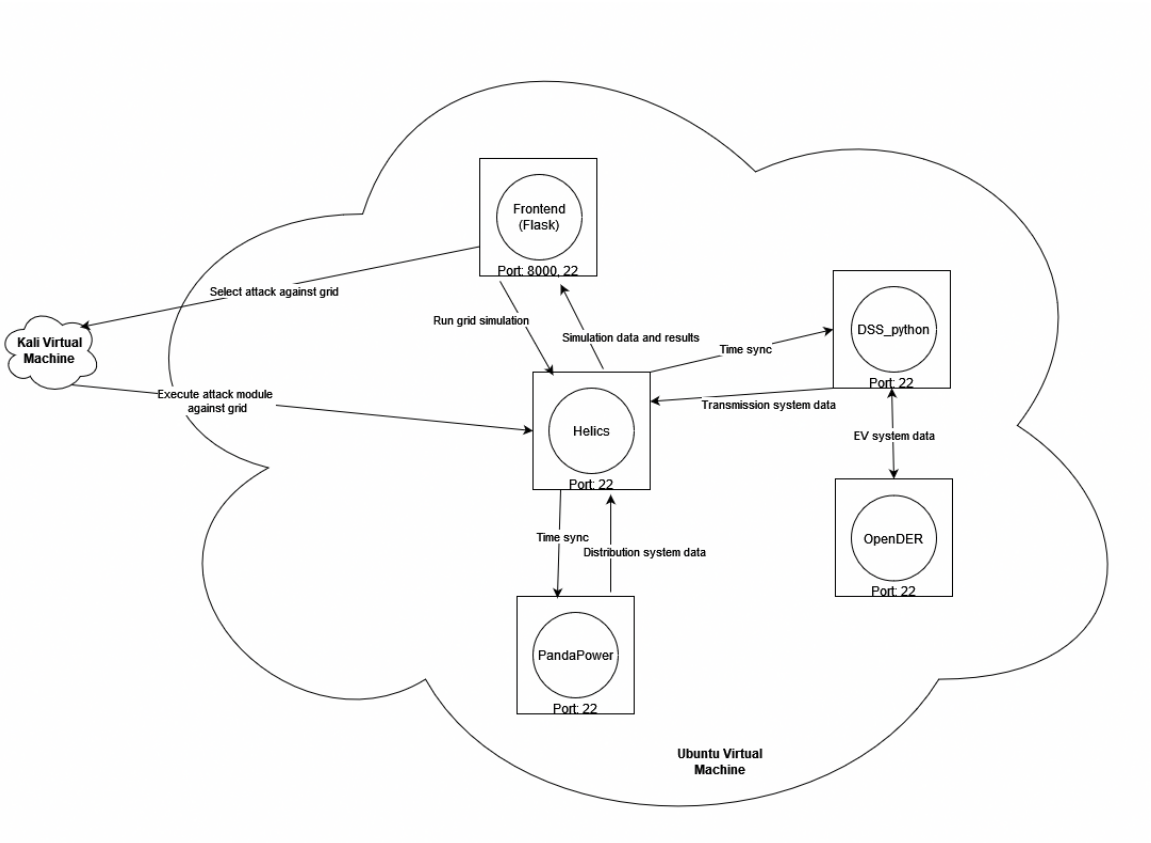


Figure 2: Complete Flowchart of Operation

1 Team, Problem Statement, Requirements, and Engineering Standards

1.1 Team Member

Justin Templeton

Tyler Atkison

Kaya Zdan

Zach Hirst

Tommy Keeshan

Matt Nevin

1.2 Required Skill Sets for Your Project

Coding proficiency

Power grid analysis and creation

Analysis of cyber attacks

HELICS proficiency

OpenDSS proficiency

OpenDSS proficiency

Generation of cyber attacks

Web Development (frontend and backend)

Docker

OpenDER

PandaPower

1.3 Skill Sets covered by the Team

Coding proficiency (All)

Power grid analysis and creation (Matthew, Thomas)

Analysis of cyber attacks (Kaya, Tyler, Zachary, Justin)

Generation of cyber attacks (Kaya, Tyler, Zachary, Justin)

Web Development (frontend and backend) (Kaya, Tyler, Zachary, Justin)

HELICS proficiency (Kaya, Justin)

OpenDSS proficiency (Matthew, Thomas)

OpenDER proficiency (Matthew, Thomas)

Pandapower proficiency (Matthew, Thomas, Kaya)

Communication (All)

Docker (Justin

1.4 Project Management Style Adopted by the team

Agile - Scrum (sprint based)

1.5 Initial Project Management Roles

Justin - DevOps Manager and Scrum Master

Kaya - HELICS Connection Manager

Zachary - Cyber Attack Simulation Manager

Tyler - Cyber Attack Generation Manager

Matthew - Powergrid Analysis Manager

Thomas - Powergrid Creation Manager

Name	Role	Contributions
Justin	DevOps Manager and Scrum Master	Created development Docker instances for all members virtual machines and set up running environment for project

Kaya	HELICS Connection Manager	Experimented and researched using HELICS, Pandapower, and DSS_python, and setting up connecting the three tools together.
Zachary	Cyber Attack Simulation Manager	Helped generate the list of preconfigured attacks against the grid. Developed network/machine configuration. Also developed the attack deployment website used to configure new and initiate preconfigured attacks.
Tyler	Cyber Attack Generation Manager	Helped create a list of attack vectors and attacks to be used. Also helped research attacks and how they could be used in our environment.
Matthew	Powergrid Analysis Manager	Ran simulations of existing grid examples for both distribution and transmission models through Pandapower and OpenDSS.
Thomas	Powergrid Creation Manager	Created and ran simulations of distribution models along with distributed energy resources for loads and generation using OpenDER and OpenDSS.

2 Requirements and Engineering Standards

2.1 Problem Statement

Cyber attacks against the power grid are a growing concern. Our group is creating a virtual distribution & transmission power grid that we can simulate cyber attacks against to help showcase potential attack and defense scenarios. It ensures that the companies running their power grid have substantial protection against attacks.

2.2 Requirements & Constraints

Functional Requirements:

- Use CyHELICS to combine multiple substream programs and run concurrently
- Include both power grid model analysis tools and cyber security focused programs.
- The simulation will be capable of handling multiple attack simulations, based on the OWASP top 10.
- Create a power grid with several transmission models that connect with several distribution models and demonstrate proper power flow.
- Power Grid will include multiple load types.
- The power grid interface will be able to simulate different grid set ups.
- The simulation will be tested in a VM environment.
- The simulation will be set up in a dockerized environment.
- The user must be able to select how much of the grid they want to simulate an outage for in the flask frontend, with specialized attacks to take out the varying percentages of the grind. This will purely interact with the Kali box to send the various attacks.

Nonfunctional Requirements:

- The interface must be easy to use for non technical users (city planners, grid designers).
- The user must be able to select how much of the grid they want to simulate an outage for, with specialized attacks for each one.
- The simulation must give feedback to the user about the state of the simulation (failed, complete, in progress etc.)

2.3 Engineering Standards

- HELICS and PandaPower use an open-source BSD-3 clause license.
- OpenDSS is open source, with no listed license.
- MITRE ATT&CK Framework is an industry-standard knowledge base for pentesting, gap assessments, threat intelligence/hunting, and more.
- OWASP Top 10 as a security guidance standard.
- Python is an industry-standard interpreted scripting language.

2.4 Intended Users and Uses

The people who benefit from the results of our project are the power grid companies, the city, and the general population. It ensures that the companies running their power grid have substantial protection against attacks. Those who will be directly interacting with the software or its productions are as follows: Power grid companies, local utilities, city planners, maintenance companies, city politicians, city citizens, and researchers. This output data will be used to find weaknesses in the input grids and to test future expansions of the grids, checking for errors along the way.

3 Project Plan

3.1 Task Decomposition

Our project will have a structured sequence of events. Beginning with the setup of virtual machines within a dockerized environment. We then establish connections between HELICS, Pandapower, and OpenDSS using the virtual machines we have already created. Followed by crafting an electric grid diagram for visualization. Going deeper into electrical grid design, we will create smaller transmission and distribution grid sections. Once they are set up we will then simulate these with an emphasis on how they interact with each other. The security aspects of our project are addressed through a Kali purple box in the company with an implementation of Security Onion for defense. Finally, we will implement a front-end website to enhance user experience.

3.2 Project Management/Tracking Procedures

Our team decided upon an Agile Project Management approach. This gives us the flexibility to accommodate multiple adjustment periods in our project. The adjustment periods ensure a smooth transition for our team to acclimate to the proposed software. Agile development allows for small incremental parts of our project to be developed and tested. As goals and tasks become more advanced, the agile management structure allows us to break these tasks down into smaller attainable goals. To streamline our Agile management style, we will be using Gitlab for version control and centralized project management.

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Our team has identified milestones that we deem fit for evaluation criteria. First, in the preliminary grid phase, we simulate pre-existing transmission, distribution, and load models, ensuring they simulate seamlessly through the HELICS software. Moving forward into the grid design phase, we dive into designing multiple transmission models, ensuring proper power flow, and conducting analyses encompassing power flow, fault simulations, harmonics, and unbalanced power flow. We also will make distribution models, capable of accommodating dynamic loads, employing both linear and nonlinear models. A dynamic load profile is created, emulating real-world scenarios, including electric vehicle usage and residential neighborhood power consumption, with fluctuations based on actual data. In the simulation setup phase, each simulation is dockerized individually. The attack modules phase entails setting up a Kali box for executing targeted attacks against the simulated grid and designing a user-friendly frontend interface to streamline attack execution. Lastly, in the defense phase, we deploy Security Onion to detect and counteract grid attacks, implement automated defense mechanisms, centralize logs in a database, and establish rules to swiftly identify and address potential issues. These milestones collectively shape our project's progression, ensuring its success and security at each stage.

3.4 Project Timeline/Schedule

	1/22-29	2/5-12	2/19-26	3/4-11	3/18-25	4/1-8	4/15-22	4/29-5/6	5/13-20
Analyze pre-existing transmission and distribution models	Deliver by 1/29								
Analyze the power flow and design of models		Deliver by 2/12							
Co-Simulation between transmission and distribution models		Deliver by 2/12							
Working simulation				Deliver by 3/11					
Set up VM and Dockerized Environments	Deliver by 1/29								
Create and run attacks							Deliver by 4/22		
Frontend for attack modules									Deliver by 5/20
Integrating security onion						Deliver by 4/8			
Using security onion to detect the attacks									Deliver by 5/20

3.5 Risks And Risk Management/Mitigation

As our entire project will be done within a dockerized VM environment, there will not be significant risks associated with this project.

Risks:

- Losing our VMs and our progress - 0.3
    - Mitigation: Create regular backups and VM snapshots
- Using too many resources, causing the department to be mad at us - 0.3
    - Mitigation: Being aware of our resources and making sure nothing runs out of hand

3.6 Personnel Effort Requirements

Task	Justin Hrs	Kaya Hrs	Matt Hrs	Tommy Hrs	Tyler Hrs	Zach Hrs	Total Hrs
Set up and connect HELICS with pandapower and OpenDSS	20	50	15	15	5	5	110
Analyze the power flow and design of models	0	0	5	5	0	0	10
Create smaller sections of the transmission and distribution grid	5	5	30	30	5	5	80
Co-Simulation between transmission and distribution models	2	2	2	2	2	2	12
Create a working simulation	20	20	10	10	20	20	100
Create a Kali/red team box that can perform attacks against the power grid	2	2	0	0	2	2	8
Set up security onion to defend against the attacks against the power grid	0	0	0	0	20	0	20
Frontend for attack modules	5	5	0	0	5	5	20

3.7 Other Resource Requirements

Our team has identified a few other resource requirements for our project to function effectively. At a base, we will need access to the university's High-Performance Compute Cluster, which serves as our backbone for testing complex power designs. Additionally, our team will need a multitude of virtual machines. We have identified a few types of machines we will need. The first is a Kali Linux purple pentesting box for heading the cyber attack section of our project. We will need a few Ubuntu and Windows 10 machines for hosting HELICS, Panda Power, & Open DSS. Finally, we will need a larger virtual machine to host Security Onion for network monitoring.

4 Design

4.1 Design Content

Our design content includes the design and architecture of our project. This includes the software architecture of the interworking components, as well as the design of our simulated electrical grid.

For the software architecture, we need to establish the connections between our various open source software systems, HELICS, pandapower, pyDSS, OpenDER, establish the docker containers, and connect this system to a user interface, where the client will be able to launch the attack modules and run the simulation of the grid.

As for the electrical design, we need to design an integrated power grid that includes various transmission lines on PandaPower with varying voltages integrated with OpenDSS for distribution.

4.2 Design Complexity

1. Simulation
    a. We will be using a Dockerized environment for our simulation to function on many different types of ecosystems with minimal adjustments being made. These Docker containers will utilize a frontend, HELICS, PandaPower, and Python-DSS instances to properly simulate a power grid.

2. Grid Creation

    a. Our Solution will come out of the box with a premade power grid that users can experiment with. We also would like to allow users to input their grids, but this feasibility needs to be tested.

3. Attack Vectors

    a. We will be utilizing the MITRE ATT&CK framework to determine a list of common attack vectors to be launched against the simulated grid. These attack vectors are:

        i. Command and control backdoor that allows us to manipulate the grid by changing electrical values

        ii. Keystroke injection attack that will be carried out against a machine jumped onto the network, which will then be infected to change/mess with the grid

        iii. SYN flood attack to introduce false network traffic & bog down the network

        iv. Or other DDoS variations

        v. Malware downloaded on the network through a what-if scenario where a user has successfully phished

vi. "Time Bomb"

vii. Solarwinds (infecting the top of the supply chain e.g adding in infected code through the web app)

viii. Man in the middle

4.3 Modern Engineering Tools

- HELICS: a co-simulation tool that allows multiple simulators to run simultaneously and off of each others' results by allowing each software to speak the same language. This tool is necessary because otherwise, the simulators running by themselves would not accurately portray a whole and singular electric grid.
- Pandapower: a tool that simulates a transmission grid.
- DSS_python: a tool that portrays a distribution grid, python wrapper for OpenDSS.
- OpenDER: a tool that portrays batteries connected to the grid. Can be used to simulate EV's.
- Kali: a red team-oriented operating system that will help develop new attack methods and modules, as well as utilize pre-existing ones.
- Docker: Containerized solution to run programs on many platforms easily. Simple to create, tear down, and connect environments.
- Virtualization: Running a guest operating system on a host machine, available to Iowa State Students to safely run their programs.
- Flask: A basic python web application package that can be used with both simple and complex applications alike. It allows many other packages to be used on the python platform to run complex methods.
- Ubuntu: the most popular and flushed out linux distribution that is one of the two industry standards besides for Red Hat linux.

4.4 Design Context

Area	Description	Examples
Public health, safety, and welfare	Many health and safety solutions rely on the power grid to function, from hospitals to homes.	Increased power grid reliability due to decreased chance of cyber-attacks impacting users.
Global, cultural, and social	People of all groups, within cities and rural areas, expect electricity to be delivered at all times if needed.	Less power outages caused by cyberattacks. More efficient designs can be created.
Environmental	Reduced attacks on power grids will lead to less power waste. This will also help with the overall design of the grid to be safer and more efficient.	Protecting power grids from cyber attacks will decrease the need for generators. Using the principle of economies of scale, this will waste less fuel.
Economic	If the electric grid gets taken down or disabled by attackers, it will result in financial losses for the power companies and any companies using that grid.	Protecting against cyber attacks will allow less outages and economic loss.

4.5 Prior Work/Solutions

There are a couple papers that did similar projects to ours that our client suggested we look into. They are:

- HELICSAuto: Automating the Development of Cyber-Physical Co-Simulation Framework for Smart Grids
    - This paper delves into automating the HELICS API, and tested it's usage by simulating Pandapower, PowerWorld, OpalRT, and PyDNP3 with Helics.
    - This shows us that Pandapower can be utilized with HELICS, as well as gives examples of other similar programs, but it is not implementing a full electric grid nor simulating cyber attacks on it.
- Defense-in-Depth Framework for Power Transmission System against Cyber-Induced Substation Outages
    - This paper takes an IEEE 14 bus system and uses it to evaluate cyber attacks. This perspective of this paper focuses on the defensive side, and what can be done to protect an electric grid against a cyber attack, as well as what portions of the grid need to be particularly paid attention to. The IEEE 14 bus system was simulated using MatLab 2019a.
    - This paper is similar to our project in the sense that we are both simulating attacks against a power grid to check for weaknesses, however our project will go into

specific types of attacks, as well as using tools such as HELICS, Pandapower, and DSS_python to simulate a more accurate and large scale power grid.

- Next-Generation CPS Testbed-based Grid Exercise - Synthetic Grid, Attack, and Defense Modeling
    - This paper focuses on creating a test-bed environment for the industry to practice incident response for power grid cyber attacks. This paper is the most similar to our project, however still has some differences.
    - The program in the paper focuses on how specific parts of the power grid will go offline in the events of an attack, to help simulate a real time attack against a power grid. Our project will focus more on how specific attacks will fare against the power grid, as well as allowing different types of power grid models to be tested against these attacks.

4.6 Design Decisions

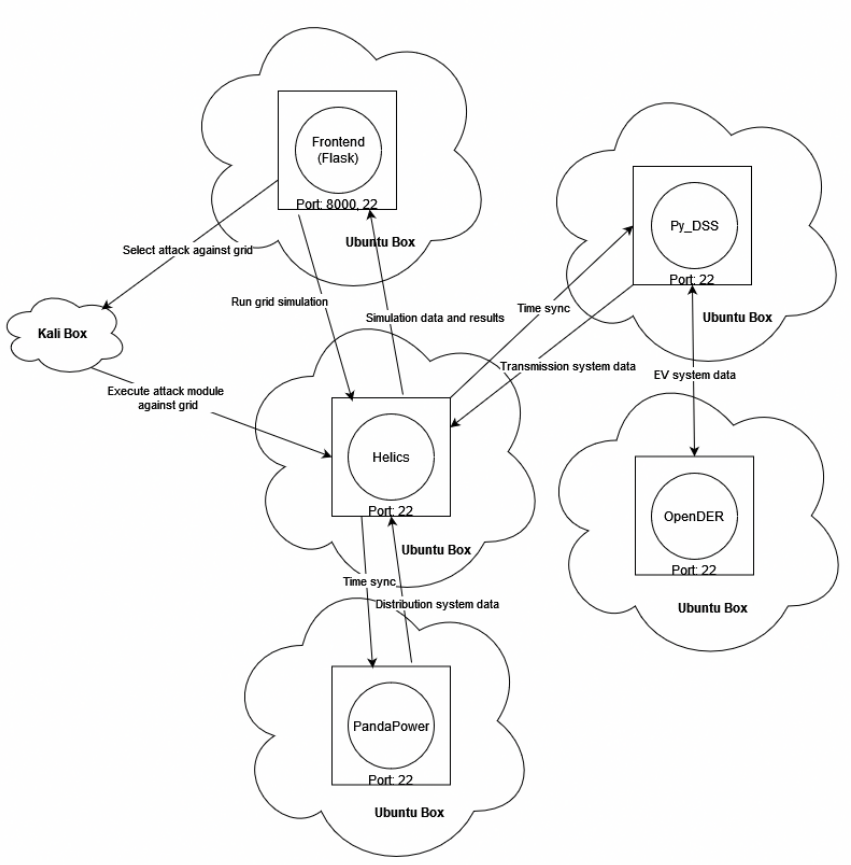
- Run each program in its own Docker container in order to keep different packages separate. Keeping the transmission and distribution grids in separate Docker environments also makes the simulation more accurate as these in reality would be separate machines.
- Run the same category of programs in their own virtual machines, because it will be easier to perform attacks on the virtual machines, and simulate disengaging certain parts of the grid.
- We've decided on the open source simulators that we did, since it was suggested by our advisor, many research papers have used the same tooling, and since they're open source, they incur no fees.

4.7 Proposed Design

- We have experimented using HELICS, pandapower, dss_python (python wrapper for openDSS), and OpenDSS.
- We have dockerized containers for HELICS, pandapower and dss_python and are continuing to make improvements.
- We are starting experimentation with connecting HELICS, pandapower and dss_python to get a fully functioning grid.
- We are currently designing a basic grid layout.
- We are identifying potential attack vectors.
- We got access to the client provided virtual machines and are setting them up.

4.7.1 Design 0 (Initial Design)

Design Visual and Description

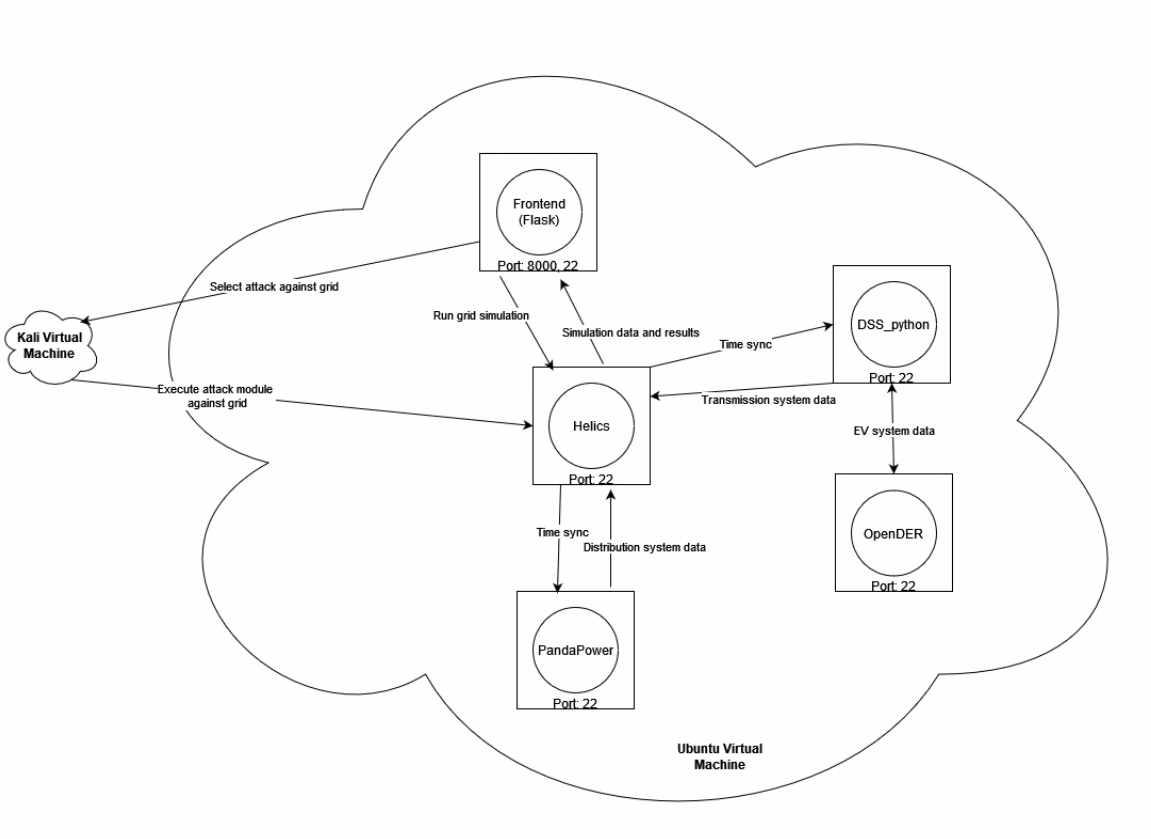


This initial design included sectioning off each program into its own virtual machine Ubuntu box. dss_python is used to simulate our distribution systems, and is used in conjunction with pandapower, a simulator for our transmission system. The transmission system is how the high voltage energy moves across the grid, whereas the distribution system is how the voltage gets converted to lower voltages at substations to allow for distribution to all of the end users on the grid. These systems communicate with each other through HELICS, which facilitates the communication and timing of the simulation results. OpenDER provides dss_python with electric vehicle information to add more of a strain on the power grid, as well as mimic a real life scenario now that EV charging is becoming more widespread. This will only communicate with the distribution simulation (dss_python) because it is one of the end "users" that are on the grid and only needs to interact with the distribution simulation. HELICS communicates the results of the simulation with our flask frontend, which will be our user interface that allows clients to run attack modules against the grid simulation. The "Time Sync" arrows are information that HELICS sends out to the individual simulations to make sure they are stepping in the correct time increments and making sure all simulations are at the same time step. The time will be stepped by default in milliseconds. This is important as this is what will help it simulate an energy grid in real time, with data mimicking real world use. When an attack module is chosen and executed, this command will be sent to a Kali box that will run this command against the grid simulation.

Functionality

This current design satisfies all of the outlined requirements.

4.7.2 Design 1 (Design Iteration)



The major change made was modifying the virtual machine structure from including each program in its own virtual machine, to having each program included in one virtual machine.

This change was made for:

- Simplicity, since it's easier to manage one virtual machine with all of the programs in it, rather than tons of virtual machines all talking to each other.
- This will also result in easier connections, since we won't have to manage the communication between several virtual machines. All of the connections will be done over the localhost.

4.8 Technology Considerations

- Using Docker instances makes the overall design much more complex, but makes it easy to deploy on a wide range of platforms, from servers of different architectures or operating systems.
- Using a Docker shared volume comes with a few downsides when it comes to how each instance of the program gets pushed to the Docker instances, but this makes it easy to keep network traffic clean and only related to the simulation.
- Having the network traffic occur over localhost allows for it to be sniffed on the network - Docker does have a way to do network traffic inside of the Docker Daemon, but this would make the simulation awfully unrealistic and cyber attacks near impossible.

4.9 Design Analysis

Our first design uses multiple VMs, which would work as well, but would add a lot of complexity, setup time, and confusion to the overall design of the system. This unneeded complexity would make the project much harder to implement and explain both to other developers and outside observers attempting to understand it. Using Dockers will have the connections made to the same host, but on different ports that each App is running on.

Our design will work with the existing softwares. We have HELICS sending the results from its Docker instance with Frontend and displaying graphs and other data on said Frontend. There are research papers that have used HELICS alongside PandaPower and OpenDSS, so they have worked together in the past using older versions and wrappers; we want it to work in a Dockerized environment which will be much more adaptable for others to use. We know how to use PandaPower and OpenDSS. We are currently trying to figure out how to use HELICS itself in order to connect the two - which is its main purpose.

5 Testing

The CyHELICS project focuses on simulation integration between several softwares in real time. Our project proposes a few unique software complications that will need to be tested extensively. Due to the Dockerized environment we will have predominantly unit testing, integration testing, and system testing. Unique challenges that we will face during the testing phase of our design will include making sure that data is properly and efficiently transferred between softwares, making sure that each software correctly interprets that data transferred, and that our system works correctly as an entire system.

5.1 Unit Testing

Unit Testing will be done by making sure that all software packages can return their versions. Checking for errors at this step will make sure that all software packages are installed correctly on their respective Docker containers. Making sure that Docker properly sets up the shared backend "/app" between the Dockers is also critical, but easy to do with checking the Docker shared volume directory. Ensuring that all Docker containers are able to talk to one another is vital, this can be done through a series of pings on localhost ports to ensure that they are opened by the Docker daemon. We can achieve this by a bash or python script whose results can be shown on the front end.

5.2 Interface Testing

We need to keep close track of what information is being sent between the Docker containers in order to make sure that there are no mistranslations in the system. We will set up the expected values to be sent out of the containers and make sure that valuesF are the correct type (integers are not strings, etc.). These tests can be run using packages such as pytest.

5.3 Integration Testing

We will test the integration through testing HELICS, as HELICS requires all software (pandapower, python_dss, Docker, Flask frontend) to be correctly integrated with each other to work. Since HELICS is a bridge software, if one of the software components does not work, HELICS will not function properly. These tests will be run through a series of basic calculations to ensure correct output. The results of HELICS will be displayed on a Flask frontend application, which will further display the proper connection between HELICS and the frontend.

5.4 System Testing

5.5 Regression Testing

We are using Docker to ensure that new additions and features can be added at any point. It is rather easy to add a new Docker container to the existing nodes that we have set up. Altering the

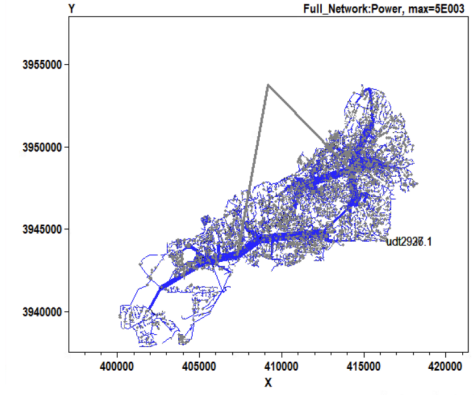
code will be easy to add new features with Docker as its connections will be made over localhost. We will test the localhost ports for the connections being established.

We need to ensure that the existing features are not broken by new Docker containers being added, but due to the nature of the hardcoded port values, unless the new addition is misconfigured, it will run as intended.

5.6 Acceptance Testing

Our client has been working with us throughout the semester to ensure that we are using the correct tools in order to complete the tasks. Building the correct framework out of the correct libraries is critical to getting the job done. We will test whether the requirements are met to the client's satisfaction and see if the pre planned attacks fulfill the planned goals. This will be done by setting a baseline to achieve success using a small grid. The small grid in question is called "123Bus," it's a small scale electric grid model that has 123 bus connections. 123Bus is a small example model provided by OpenDSS.

Our tests ensure that our product can continue to be developed and help users determine the security of their power grids, regardless of their system specifications or tech stack. For compliance with the project, we must make sure that all parts of the system are able to function individually and communicate with each other - which will lead us to a proper simulation. We can test this through a myriad of ways and present it to the user when a test is successful at affecting the grid. For example, when a certain test is established the diagrams produced by PandaPower, OpenDSS, and Open DER will adjust to those conditions so we will see changes in the model with line thickness (shown on the first figure) and the data results on each individual lines (shown on the second figure).




```
Values for:  LINE.L(R:UDT10535-UHS7_1247)
Bus1= udt10535-uhs7_1247x.1.2.3
Bus2= udt10535.1.2.3

Distance from Meter:   0.000 kM (  0.000 mi)
Power Flow:          3031.622 kW + j  509.173 kvar

Phase Currents:       138.211 123.168 156.702 Amps
Phase Voltages(L-N):    7.361   7.347   7.358 kV
Per Unit Voltages:      1.022   1.021   1.022

Losses for this Element:
TotalLosses:   22.486 kW + j  33.612 kvar

Number of Customers: 0
Total Customers downline: 0
```

5.7 Results

Our testing will ensure that upon creation - our project will automatically test itself to ensure that it will be set up for success and will be able to serve those using it.

6 Implementation

We have started experimenting with connecting HELICS, OpenDSS, Pandapower, the Flask frontend together, and running it all within Docker. We are working on getting Pandapower and OpenDSS to communicate with each other with HELICS as the communication facilitator. We have started experimenting with some basic attacks that we could run against the grid, specifically a SYN flood.

7 Professionalism

This discussion is with respect to the paper titled "Contextualizing Professionalism in Capstone Projects Using the IDEALS Professional Responsibility Assessment", *International Journal of Engineering Education* Vol. 28, No. 2, pp. 416–424, 2012

7.1 Areas of Responsibility

Area of Responsibility	**Definition**	**NSPE Canon**	**IEEE**
Work Competence	Perform work of high quality, integrity, timeliness, and professional competence.	Perform services only in areas of their competence; Avoid deceptive acts.	Point 6 states that we must not only maintain but also improve our technical competence, and only complete tasks if qualified by training, experience, or have disclosed all limitations.
Financial Responsibility	Deliver products and services of realizable value and at reasonable costs.	Act for each employer or client as faithful agents or trustees.	Point 5 states that we must be honest and realistic in our claims and estimates, and adds that we should fairly credit the contributions of others.
Communication Honesty	Report work truthfully, without deception, and understandable to stakeholders	Issue public statements only in an objective and truthful manner; Avoid deceptive acts.	Point 9 states that we should avoid injuring others' property, reputation, or employment with our actions, rumors, or any other forms of abuse.
Health, Safety, Well-Being	Minimize risks to safety, health, and well-being of stakeholders.	Hold paramount the safety, health, and welfare of the public.	Point 1 states that we should hold paramount the safety, health, and welfare of the public .
Property Ownership	Respect property, ideas, and information of clients and others.	Act for each employer or client as faithful agents or trustees.	Point 5 includes that we should fairly credit the contributions of others.
Sustainability	Protect environment and natural resources locally and globally.		Point 1 states that we should comply with ethical design and sustainable practices, not

			only to the environment, but also to the public and their privacy.
Social Responsibility	Produce products and services that benefit society and communities.	Conduct themselves honorably, responsibly, ethically, and lawfully so as to enhance the honor, reputation, and usefulness of the profession.	Point 2 states that we should improve the understanding of individuals and society of the capabilities and implications of new technologies, including intelligent systems.

7.2 Project Specific Professional Responsibility Areas

Responsibility	Application	Performance
Work Competence	Yes	Medium
Financial Responsibility	No	N/A
Communication Honesty	Yes	High
Health, Safety, Well-Being	Yes	High
Property Ownership	No	N/A
Sustainability	Yes	High
Social Responsibility	Yes	High

- Work Competence
    - Our project must not be deceptive in its topics, we must all have knowledge about what types of grids we are working with and how the code connects them.
    - A lot of our project so far has been learning about the project itself, hence this only deserves a medium.
- Financial Responsibility
    - This project uses a provided testbed for virtualization using VMWare. There are no costs besides the processing resources used by this project, hence it is not applicable.

- Communication Honesty
    - While we do not have any stakeholders, we are reporting to our advisor and client every week on what work we have performed and what will be done in the future. We also get guidance on how to better achieve the goal of the project. Therefore, we must do well in our communication with the client for the project to be successful.
    - We have done a good job with our weekly reports to our client and advisor as well as meeting with him outside of the regularly scheduled meeting time.
- Health, Safety, Well-Being
    - The power grid has many different uses when it comes to medicine, safety, and modern civilization. Our project may be used to help develop newer power grids or modify existing ones in order to make them more secure. We must be understanding and serious about the changes this could cause and make sure we emphasize that this project is not the be-all-end-all when it comes to power grid security checking.
    - We have not really had to show this yet, as we have not gotten to the stage of full product implementation, but we are aware of the weight this holds.
- Property Ownership
    - Our project does not have any property required to own, therefore this does not apply.
- Sustainability
    - In relation to the Financial Responsibility of the project, we have to have the sustainability of the project in mind to make sure that no simulation is kept running on the virtual machines and wasting needless amounts of energy.
    - We have been keeping tabs on the virtual machines and making sure that there are no active running programs when we are not working.
- Social Responsibility
    - Similar to Health, Safety, Well-Being, we must be cognizant that the product of this project will be used to help develop power grids. This means that we must do our best to be mindful of the impact that this will have on power grid developments and the social impact and weight that holds.
    - We have not really had to show this yet, as we have not gotten to the stage of full product implementation, but we are aware of the weight this holds.

7.3 Most Applicable Professional Responsibility Area

Health, Safety, Well-Being and Social Responsibility

This is because the purpose of our project is to benefit the health, safety, and well being of society. Our project's goal is to provide a simulator that can help protect the current power grid against cyber attacks, as well as test potential future designs. Health, safety, well-being and social responsibility is the backbone of our project.

8 Closing Material

8.1 Discussion

This semester we did a lot of research and familiarized ourselves with the programs that we will be using for our project, as well as setting up our environment.

We:

- Got our virtual machine environments set up and configured with the programs we will be using.
- Set up our Dockerfiles so that our programs will be containerized and run with a Docker compose file.
- Familiarized ourselves with HELICS, PandaPower, OpenDER, and OpenDSS by watching YouTube tutorials, reading the documentation, and running basic examples and experiments.
- Researched and developed a list of potential attacks to utilize.
- Attempted preliminary experiments with running a SYN flood against the Docker containers.
- Set up a basic Flask frontend and configured it to display the HELICS results.

8.2 Conclusion

This semester we have done a lot of investigative and experimental research regarding the programs we will be using in our project (HELICS, Pandapower, OpenDSS, OpenDER), and our goal was to set up a basic simulation as a proof of concept. Our plan of action for the future will be to use what we have learned this semester as well as the pre-existing architecture that has been setup to hit the ground running with our implementation plan to get a functioning complex simulation that meets our requirements working and then begin testing cyber attacks.

8.3 References

- Docker:

  [1] T. Donohue, "How To Communicate Between Docker Containers," *Tutorial Works*, Nov. 06, 2020. https://www.tutorialworks.com/container-networking/ (accessed Oct. 11, 2023).

  [2] "How To Share Data between Docker Containers | DigitalOcean," *www.digitalocean.com*. https://www.digitalocean.com/community/tutorials/how-to-share-data-between-docker-containers (accessed Oct. 09, 2023).

- HELICS:

[1] "HELICS documentation — HELICS documentation," docs.helics.org. https://docs.helics.org/en/latest/index.html (accessed Sept. 18, 2023).

[1] "GMLC-TDC/HELICS," GitHub, Nov. 14, 2023. https://github.com/GMLC-TDC/HELICS (accessed Sept. 18, 2023).

[1] "HELICS-Examples," GitHub, Oct. 31, 2023. https://github.com/GMLC-TDC/HELICS-Examples (accessed Sept. 19, 2023).

[1] "Docker," hub.docker.com. https://hub.docker.com/r/HELICS/HELICS# (accessed Oct. 04, 2023).

- OpenDSS

[1] "DSS-Python's API reference — dss_python 0.14.0.dev documentation," dss-extensions.org. https://dss-extensions.org/dss_python/ (accessed Nov. 11, 2023).

[1] "DSS-Python: Extended bindings for an alternative implementation of EPRI's OpenDSS," GitHub, Nov. 27, 2023. https://github.com/dss-extensions/dss_python (accessed Nov. 11, 2023).

- OpenDER

[1] "epri-dev/OpenDER," GitHub, Oct. 29, 2023. https://github.com/epri-dev/opender (accessed Nov. 15, 2023).

[1] Y. M. Anandan Wei Ren, Paulo Radatz, Jithendar, "opender: Open-source Distributed Energy Resources (DER) Model that represents IEEE Standard 1547-2018 requirements for steady-state and dynamic analyses," PyPI. https://pypi.org/project/opender/ (accessed Nov. 15, 2023).

[1] "EPRI Home," www.epri.com. https://www.epri.com/opender (accessed Nov. 15, 2023).

- PandaPower

[1] "pandapower," pandapower. https://www.pandapower.org/ (accessed Sept. 15, 2023).

[1] L. T. Scheidler Alexander, "pandapower: An easy to use open source tool for power system modeling, analysis and optimization with a high degree of automation.," PyPI. https://pypi.org/project/pandapower/ (accessed Sept. 15, 2023).

[1] "e2nIEE/pandapower," GitHub, Dec. 02, 2023. https://github.com/e2nIEE/pandapower (accessed Sept. 15, 2023).

8.4 Appendices

8.5 Team Contract

Team Members:

1) Tyler Atkinson
2) Zach Hirst
3) Thomas Keeshan
4) Matthew Nevin
5) Justin Templeton
6) Kaya Zdan

Team Procedures

1. Day, time, and location (face-to-face or virtual) for regular team meetings:
- Wednesdays @ 7:30 - 8:30

2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):
- Preferred method of communication is the team discord channel

3. Decision-making policy (e.g., consensus, majority vote):
- Consensus

4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived):
- One team member will take notes as needed and publish them with other team members in the Discord Channel and the Google Drive.

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:
- Expected to attend every meeting unless there is notice given to the team on standard channels for their absence
- If abused for what is deemed not adequate reasoning will be treated like any other infraction

2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:
- Expected to fulfill their assignments to a satisfactory level determined by the team, by the assigned deadlines.
- If the deadline is not met, the team member is expected to have a justifiable reason, and if determined to be excessively abused, will be treated as any other infraction.

3. Expected level of communication with other team members:
- Instant communication is not a requirement in this team. However, timely communication to complete the assignment / project on time before the deadline is required.

4. Expected level of commitment to team decisions and tasks:
- Expected level of commitment to tasks is equal to the amount of time and effort required to complete the task assigned to the team member. Members should make an effort to contribute to team decisions, but not all members are required to contribute to all decisions.

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction, individual component design, testing, etc.):
- These roles should be the equal responsibility of everyone so as to have everyone have the responsibility of keeping everyone else on track throughout the project.
- Client interaction should be done on a regular schedule or on a as needed basis.

2. Strategies for supporting and guiding the work of all team members:
- Regular interactions in discord when outside of meetings to help teammates with problems that have arisen and/or questions that may arise. Also if someone is lacking due to a justifiable reason, comments can be made in the discord asking for help.

3. Strategies for recognizing the contributions of all team members:
- Shoutouts in the team discord and/or team meetings.
- Emails if needed (TA, advisor, professors, etc.)

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the team.
- Tyler: I have work experience as a SOC Analyst (so Tier-1 investigative actions), I have gotten my Security+ certification, and I am skilled in investigations on attack vectors, how attacks work, and their consequences. I also gather threat intelligence so I am up to date with modern attacks and zero days.
- Tommy: I have work experience involving the power grid as a substation engineer. I am specializing in power systems for electrical engineering and have some experience in various coding languages.
- Matthew: I have interned for two summers with a substation design firm for a company specializing in grid solutions. My skills are power systems (grid analysis, substation design, t-line modeling), experience with C and the Linux operating system. (currently taking a java course).
- Justin: I have done a lot of security work through my two internships with Principal Financial Group. My skills are SIEM rule writing, Pentesting, Linux commands and coding (Java, Python, C, Assembly, Terraform).
- Kaya: I interned with PwC's red team over the summer and really enjoy the penetration testing side of cybersecurity. My skills are comfortable with Linux, VMs, various Kali and cybersecurity tools. Some C, Java, Python experience.
- Zach: My skills are Java, C, Python, VHDL programming. Windows and Unix system programming. Cyber security network design. Cryptography analysis and programming.

2. Strategies for encouraging and support contributions and ideas from all team members:
- Fostering an open and supportive team environment. This includes being friendly and open to ideas from other team members, encouraging other team members, and creating an effort to actively include members who may seem more reserved.

3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will a team member inform the team that the team environment is obstructing their opportunity or ability to contribute?)
- If the team member is comfortable, they should post a message in the team discord channel, or directly with the person in question discussing the environment and how it makes them feel.
- If the team member is not comfortable posting in the team-wide discord channel, the member can bring the matter to the attention of either the team lead, or a team member they feel the most comfortable with, and the team lead or other member can post in the discord channel to create the discussion point with the team.

Goal-Setting, Planning, and Execution

1. Team goals for this semester:
- Ensure assignments/due dates are handled efficiently and before deadlines.
- Have a standard of communication that is met or exceeded throughout the semesters.
- Get the project done to a point that the team feels comfortable with.

2. Strategies for planning and assigning individual and team work:
- Dish out weekly and multi week assignments at team weekly meetings.
- Briefly discuss deadlines at least a week before the due date and assign people to tasks

Consequences for Not Adhering to Team Contract

- In weekly team meetings, discuss infractions and add them to a three-strike policy.
- Three-strike policy includes 2 warnings and an escalation to teaching staff.
- On the third strike escalate to the professor and/or TA asking for assistance.

****

a) *I participated in formulating the standards, roles, and procedures as stated in this contract.*
b) *I understand that I am obligated to abide by these terms and conditions.*
c) *I understand that if I do not abide by these terms and conditions, I will suffer the consequences as stated in this contract.*

1) Zach Hirst	DATE 9/6/23
2) Justin Templeton	DATE 9/8/2023
3) Tyler Atkinson	DATE 9/8/2023
4) Kaya Zdan	DATE 9/7/2023
5) Thomas Keeshan	DATE 9/10/2023
6) Matthew Nevin	DATE 9/10/2023