Simulating the effect of cyber attacks on a Power Grid

Final Report

Sdmay23-02

Benjamin Blakely

Team Members

Jake Stanerson

Noah Peake

Cole Medgaarden

Conner Spainhower

Michael Gierek

Hrijul Balayar

https://sdmay23-02.sd.ece.iastate.edu

1 Revised Project Design

1.1 How the project has evolved

Our project focuses on simulating the effects of a False Data Injection and IoT attack. We based the design of our grid to create a makeshift version of Iowa State's campus and test the attacks on a larger scale.

1.2 Summary of Requirements

Functional requirements:

- Program needs to output results of attacks: increased power consumption, blackout, etc.
- Shell environment to replicate real grid in code
- Run attacks individually or in large quantities on grid
- Initialize a functioning power grid to be tested on

Resource requirements:

- Github connection is reliable for submitting code
- Connectivity to a python IDE is reliable
- Panda Power implementation (modeling tool)
- Python scripts for Cyber Attacks (attack anomaly)
- Other associated python libraries needing to be installed (i.e. numpy)
- Being able to pull, push, & commit code from other team members
- Can be ran on basic laptops without need for expensive hardware / computers

Aesthetic requirements:

- The output should be able to be formed into different visual representations including: tables, charts and diagrams.
- Accessibility to be able to find and read the results provided from the project
- Accessibility for information regarding specifics related to the cyber attacks

1.3 Development Standards & Practices Used

MITRE Framework:

The MITRE framework gives us a good overview of how industrial control systems can be vulnerable to specific types of attacks. These are specific standards from this framework that are related to our project:

Command & Control- by mimicking normal or expected traffic, it is possible for hackers to obtain access to your system within a network. Once they achieve administration privileges, they are then able to take control of the system, or in our case the power grid.

Privilege Escalation- after already achieving access to the system, attackers will then follow up by trying to obtain the highest level of privileges. By escalating their privileges to an administration level they could then have the ability to make permanent changes (i.e. delete user accounts, delete/transfer files, or upload malware to the servers) to a system.

Discovery- this tactic is used to gather information about a network or system, and is used as a means of reconnaissance. Attackers use this technique to obtain specific information about a network or system to find out if there are any potential vulnerabilities that could be exploited.

Credential Access- this is typically gathered through phishing attempts and is used to obtain user credentials and access at any means. Once a hacker is able to achieve access to some credentials and user access, they could then begin their mission to achieving higher privileges.

Execution- is performed when the attacker is able to access the network through methods listed above. Execution results from techniques used that would allow for an attacker to run code on local or remote systems, devices or other assets.

IEEE Standards:

IEEE 1547 & 2030 - Distributed energy resources interconnection & interoperability with electrical grid.

1.4 Engineering Constraints

For our project we were only required to work inside of one constraint. Our constraint was that the project had to be completed within the Python coding language.

1.5 Security Concerns and Countermeasures

In order to better prove our attack scripts will produce realistic results we chose to test it on a realistic grid. We were able to get information about the design of the Iowa State University power grid. Our grid was based on this information, with some modifications, in order to make our project more realistic. Due to the sensitive nature of this information it has been limited to only the members of our group. No information, which is not publicly available, is present in our reports or presentation. In addition, modifications were made to the grid so that it is not an exact representation of the University's grid.

2 Implementation details

We started by configuring a functional power grid, the next step was to begin adapting our layout to one similar to that of Iowa State's power grid. After successfully simulating a power grid using PandaPower, we then began incorporating cyber threats to be exploited against the grid. PandaPower allowed for us to manually configure the distributions of power through the grid, and this helped us be able to exploit False Data Injections and IoT attacks. The first exploit will change and edit the distributions of power throughout each of the transformers used by the grid. The IoT attack targets different buses located on the grid and attempts to manipulate other surrounding buses.

3 Testing Process and Results

The essential parts for testing the system as a whole is ultimately making sure that the fundamental components are working properly. Since the only components we use are Python, PandaPower, and GitHub, this makes things easy. For Python, we just need a simple test program that uses the libraries we will need to verify they work. PandaPower is a library in Python, so this can be tested in conjunction with Python. GitHub is also quite easy to test as we can do simple push and pull requests to see if our code uploads and downloads respectively. The critical integration paths in our design are verifying if instances of attacks work, and their correctness. Verifying these tests will include running an attack script on our grid and then simulating it in PandaPower. Following this if the grid does not converge we will need to verify a blackout occurred. Otherwise we will need to verify the attack was successful and the outputs are reasonable. The tools used for this will be our attack scripts, PandaPower, and our grid.py

The results of these conducted tests helped to further improve and enhance our procedure for completing the project and design. The results ensured compliance through proving and providing visualizations of the outcomes, as well as the realism of the output. This ensures individuals are able to physically see the different outcomes from the attack vectors. Once the results are created for the designated user, they will then be able to implement any countermeasures as desired. Our intended use for the final product is for companies to be able to take the information that is simulated from our project and use that to become more aware of ongoing threats.

4 Related Products

GridAttackSim- Cyber Attack Simulation Framework for Smart Grids

With an increase in cyber attacks against smart grids in more recent years, there have been efforts made to help educate people of the dangers and potential outcomes of these attacks. GridAttackSim is a co-simulation framework that helps to facilitate simulations of various customized smart grid systems that involve both power grid and network components. The simulation provides a GUI for users to interact with and execute attacks without the need for programming knowledge. Our project is similar to aspects of this product through the visual representations of different attacks that a power grid is subject to. We also provide an interactive shell for users to be able to generate a grid for themselves without needing to have programming experience for grid formations.

Appendix I

Operation Manual

To be able to use this product you must have an operating system with access to a terminal to be able to run the program. It is optional for you to have an IDE that can compile Python to be able to customize your own grid. If you choose to design your own grid, it is recommended that you utilize PandaPower resources for development of the personalized power grid.

Grid Creation Shell:

1. First open terminal and cd into src
2. Run "./gridshell.sh"
3. Enter either 1 to create a new file or 2 to edit an existing file
4. If you choose option 1 you will have to enter a file to create, if you choose option 2 you will have to type a file to edit. If you enter an incorrect file, the shell will exit with an error message and will have to be restarted at step 2. Ensure that the filename you entered has a .py extension
5. If you have created a new file you will be prompted with entering a network name. The network name must not contain any spaces. Once you have entered the network name you will be asked if you want to create any more networks. Simply type "yes" or "no" to keep making networks or continue.
6. After creating the network, you will then be presented to start creating the buses, lines, loads, static generators, external grids, transformers, or generators. From here you may develop the grid as you wish. A functional grid requires an external grid for power source and must include at minimum a bus, line, and load.

Main Program:

1. First you will want to open your terminal and locate the file named "MainProgram.sh".

2. After starting the program you will then be prompted to choose from two attack vectors including a False Data Injection attack and an IoT attack.
3. Once you have selected your desired intrusion method, you will then be prompted to input a ".p" file to execute against the selected attack.
4. After the file has been imputed, you will then be able to choose how many attacks you wish to run against that grid. If the number of attacks exceeds the amount of buses/transformers within the grid, you will have to re-input a new number of attacks.
5. Once the attack is executed it will then generate and open an html file that is used for visual representation of the selected attack. Additionally, data tables are printed within the terminal so the user has a list of the individual buses and their respective load distributions.
6. Located on this html file, you will have an interactive view of the affected cables and buses. This view allows for the user to observe the influxes in power distributions across the grid and have visual representation of which areas were targeted and the differences in load values at each bus and cable.

Appendix II

Alternative versions of our design

Pydgrid Package

We had the option to use any python package in order to complete the project. One consideration we had along with PandaPower was Pydgrid. Pydgrid also allows users to create a simulated power grid in python. It includes similar features such as, power flow calculations and plotting results. We ultimately decided to use PandaPower for two reasons. Reason one, our advisor Dr. Blakely had some previous experience with PandaPower which would help us get started with the project. Reason two, PandaPower is built on top of the pandas library which we found would be a great tool to help simulate our attacks.

GUI vs Shell

For our project we wanted to make sure the end user had an interface to interact with. This would reduce the amount of knowledge they would need to make use of our project. An interface like this can be done in a shell or GUI. We found the only additional benefit of a GUI would be the aesthetic aspect of the interface. Since a shell interface would still provide all the functional requirements for the project, we decided to choose it over a GUI and put more resources elsewhere.

Manually built grid vs generated grid

For our testing purposes we decided to use a manually built grid on a larger scale and adapted to that of Iowa State's infrastructure, but in the final version of our project we include a grid shell that is able to generate grids automatically for end users. The generated grid is intended for users that may not be as familiar with some of the components that go into creating a personalized grid.

Appendix III

False Data Injection:

Foundation

Our simulation was created in order to show realistic results this type of attack would produce if executed on the power grid. A false data injection takes advantage of a system by injecting false and malicious data into a system, causing some sort of harm. We worked to develop a script that would replicate this behavior. Making use of pandas, the underlying library of PandaPower, we are able to manipulate data tables of any grid and change any value of individual parts of the grid; including lines, buses, and substations. For this attack the most realistic point of injection is at substations, as shown in the example below (Ukraine Blackout 2015). By manipulating values of substations of the grid we feel we have replicated the effects of this type of attack.

How it Works

Our attack simulation first takes input for a power grid file to run the attack against. The user is then asked to input the number of attacks they want performed on the grid. In this case it is the number of substations. The script then does error checking to make sure the grid is viable. Next the attack script targets a random substation, by searching for transformers in the grid, and injects a modified voltage value. Substations are only attacked once and the number attacked is based on the user's previous 'attacks' input or until the grid has no more valid targets. Finally, the resulting values are printed in a table and a visual graph is plotted and saved as an html file.

Real World Connection

Ukraine Blackout (2015)

This attack shows that even with certain countermeasures in place, a FDIA is possible on a power grid. In late 2015 attackers were able to cause a blackout in

Ukraine's power grid. This attack affected over 200,000 customers. Malware was installed on employee machines. The malware used to complete the attack was able to bypass mechanisms that were in place to detect malicious data; which led to 30 substations being disconnected and the subsequent blackout.

IoT Attack:

Foundation

This simulation is designed to show the effects of a large scale IoT device attack and its impact on a power grid. A large-scale IoT attack would cause a noticeable increase in voltage levels than what would be normal in the area it occurred. The script was created using pandas, the underlying library in PandaPower. This attack was designed to mostly target buses within a specified area, because it best replicates how an actual large IoT attack would be shown on a grid. We will do that by manipulating the values of the bus to replicate this attack and present the results.

How it Works

Our attack simulation first takes input for a power grid file to run the attack against. The user is then asked to input the number of attacks they want to be performed on the grid. The script then does error checking to make sure the grid is viable. Next, the attack script randomly picks a bus to create a target area around. The targeted bus and those around it are then manipulated to represent an attack. Finally, the resulting values are printed in a table, and a visual graph is plotted and saved as an HTML file.

Real World Connection

Unsecured University IoT (2017)

This example is of an unknown university where the network was overrun with Domain Name Service (DNS) queries for seafood eateries is given in Verizon's Data Breach Digest 2017 report. Although it appeared to be a student joke, 5,000 IoT devices,

including vending machines and lighting systems, were used in the outside hack. A brute force attack was used to carry out the hack, taking advantage of the university's network's vulnerability to malware deployment.

Appendix IV

Source Code:

Grid Creation Shell:

```
#!/bin/bash

	echo "Entered shell interface"
while :

do
	valid=0 #valid bit for actually editing the file
	echo ""
	echo "For more information type \"--help\" or \'h\'"
	echo "Press 'ctrl+c' to exit shell"
	echo ""

	#prompt create new or update existing
	echo "Create new (1) or edit existing file (2)?"
	read -p "1 or 2: " opt
	if [ $opt != "1" ] && [ $opt != "2" ] && [ ${opt,,} != "--help" ] && [ ${opt,,} != "-h" ]
	then
		echo "Incorrect input"
	fi

 #FILE CREATION CASE
	if [ $opt = "1" ]
	then
		read -p "Enter file name: " filename
		echo "Creating file $filename"
   #create file here, also generate header. Not sure what the header will look like but I think you can figure out a good one
   #something along the lines of "This was automatically generated using gridshell.sh with a timestamp
		header="This was automatically generated using gridshell.sh"
		timestamp=$(date +"%H:%M:%S")
		touch "$filename"
		echo "#### $header ####" > $filename
```

```
                echo "#### $timestamp ####" >> $filename
  echo "import pandapower" >> $filename
                valid=1
        fi

 #EXISTING FILE CASE
        if [ $opt = "2" ]
        then
                read -p "Enter file name: " filename
                echo "Editing file $filename"
                cat "$filename"
  status=$?
  if [ $status -ne 0 ]
  then
        echo "No file with the given name exists"
        exit $1
  fi
                opt=0
                valid=1
        fi

 #${opt,,} <<<< this here just converts string to lowercase. pseudo ignore case, helps with
usability

 #USAGE STATEMENT CASE
        if [ ${opt,,} = "--help" ] || [ ${opt,,} = "-h" ]
        then
           echo "Enter a network name you would like to add. Afterwards, you can add
components such as network, bus, line, load, static generator, external grid, transformer, or
generator or line"
        fi

        while [ $valid -eq 1 ]
        do
           if [ $opt = "1" ]
           then
        read -p "Please enter name for Network: " input
        #grab name, get other options for empty network, append to input, use sed command or
other inline command to put in file
        #sed -i '$ a '"$input" "$filename"
```

```
        echo "$input = pandapower.create_empty_network()" >> $filename
          read -p "Would you like to use another network?" question
          while [ $question = "yes" ]
          do
                    read -p "Please enter name for Network: " input
                    #grab name, get other options for empty network, append to input, use sed
command or other inline command to put in file
                    echo "$input = pandapower.create_empty_network()" >> $filename
                    read -p "Would you like to use another network?" question
          done
             opt=0
             fi
                  valid=0
          done


   read -p "Please enter component type to create. To delete a component, type \"delete\". To list
all components in file, type \"list\". The available commands you can add are: network, bus, line,
load, static generator, external grid, transformer, or generator: " input
   #use input and check val to determine what was typed. If input doesn't match any of the
commands possible, report error
   while [ $input != "exit" ]
   do

          if [ $input = "network" ]
          then
             read -p "What is the name of the network you would like to add?" network
             echo "$network = pandapower.create_empty_network()" >> "$filename"
          fi

          if [ ${input,,} = "bus" ]
          then
             read -p "What is the name of the bus you would like to add?" bus
          read -p "What is the variable name of the bus?" var
          read -p "What network do you want this bus on?" network
          read -p "What is the bus's vn_kv?" vnkv
          echo "$var = pandapower.create_bus($network, vn_kv=$vnkv, name=\"$bus\")" >>
$filename
          fi
```

```
        if [ ${input,,} = "line" ]
        then
           read -p "What is the name of the line you would like to add?" line
        read -p "What type of line is it? Types of lines can be found at
https://pandapower.readthedocs.io/en/v2.11.1/index.html" linetype
        read -p "What network is this line on?" network
        read -p "From what bus is the line connected to? (Variable name)" from
        read -p "To what bus is this line connected to? (Variable name)" to
        read -p "How long is this line in kilometers?" lengthkm
        echo "pandapower.create_line($network, from_bus=$from, to_bus=$to,
length_km=$lengthkm, std_type=\"$linetype\")" >> $filename
        fi

    if [ ${input,,} = "load" ]
    then
        read -p "What is the name of the load you want to add?" load
        read -p "What network do you want it to be on?" network
        read -p "What bus is this load on? (Variable name)" bus
        read -p "What is the active power of the load?" pmw
        read -p "What is the reactive power of the load?" mvar
        echo "pandapower.create_load($network, bus=$bus, p_mw=$pmw, q_mvar=$mvar,
name=\"$load\")" >> $filename

    fi

    if [ ${input,,} = "static generator" ]
    then
        read -p "What is the name of the static generator?" sgen
        read -p "What network is this on?" network
        read -p "What bus is this static generator connected to? (variable name)" bus
        read -p "What is the active power of the static generator?" pmw
        read -p "What is the reactive power of the static generator?" mvar
        echo "pandapower.create_sgen($network, bus=$bus, $pmw, q_mvar=$mvar,
name=\"$sgen\")" >> $filename

    fi

    if [ ${input,,} = "external grid" ]
    then
        read -p "What is the name of the external grid?" name
```

```
        read -p "What network is this on?" net
        read -p "What bus is this connected to? (variable name)" bus
        read -p "What is the voltage at the slack node in per unit?" vmpu
        read -p "What is the voltage angel at the slack node in degrees?" vadegree
        echo "pandapower.create_ext_grid($net, bus=$bus, vm_pu=$vmpu,
va_degree=$vadegree, name=\"$name\")" >> $filename
    fi

    if [ ${input,,} = "transformer" ]
    then
        read -p "What is the name of the transformer?" name
        read -p "What is the variable name?" var
        read -p "What network is this connected to"? net
        read -p "What is the bus on the high-voltage side? (variable name)" hvbus
        read -p "What is the bus on the low-voltage side? (variable name)" lvbus
        read -p "What is the type of the transformer? Types can be found at
https://pandapower.readthedocs.io/en/v2.11.1/index.html" stdtype
        echo "$var = pandapower.create_transformer($net, hv_bus=$hvbus, lv_bus=$lvbus,
std_type=\"$stdtype\", name=\"$name\")" >> $filename
    fi

    if [ ${input,,} = "generator" ]
    then
        read -p "What is the name of the generator?" name
        read -p "What network is this on?" net
        read -p "What bus is this connected to? (Variable name)" bus
        read -p "What is the active power?" pmw
        echo "pandapower.create_gen($net, bus=$bus, p_mw=$pmw, name=\"$name\")" >>
$filename
    fi

        if [ ${input,,} = "list" ]
        then
           cat "$filename"
        fi

        if [ ${input,,} = "delete" ]
        then
           cat "$filename"
           read -p "What line would you like to delete?" linenum
```

```
            sed -i $linenum'd' $filename
        fi

        if [ ${input,,} = "--help" ]
    then
        echo "Here are the available commands you can use to get started: network, bus, line,
load, static generator, external grid, transformer, or generator"
    fi


    echo "Components you can instantiate are network, bus, line, load, static generator, external
grid, transformer, or generator."
            read -p "Please enter component type to create. To delete a component, type \"delete\".
To list all components in file, type \"list\": " input




        done

done
```

Main Program:

```
#!/bin/bash

while [ 1 ]
do
    echo "Select an attack to run"
    echo "IOT hack or False Data Injection"

    read input

    if [ "${input,,}" = "false data injection" ]
    then
        python ./attackScripts/falseDataInjection.py
            `rm temp-plot.html`
    fi

    if [ "${input,,}" = "iot hack" ]
    then
```

```
        python ./attackScripts/IOTattack.py
            `rm temp-plot.html`
    fi


done
```

False Data Injection Script:

```
import pandapower as pp
import numpy as np
import pandapower.create
import pandapower.plotting
import pandapower.timeseries
from pandapower.timeseries.output_writer import OutputWriter
import pandas as pd
import random as rand
import shutil

gridFilePath = input ('Please enter the filepath for the grid you wish to attack. File type must be ".p", generated by running "python gridfilenamehere.py"\n: ')

net = pp.from_pickle(filename=gridFilePath, convert=True)

transfo_arr = [0] * len(net.trafo)

totalCompleted = 0;
# while loop for how many things we want to change



numAttacks = input ('How many attacks do you want to simulate on this grid? NOTE this attack cannot attack a transformer that has already been attacked. Any attacks over the number of transformers in the grid will not be simulated.\n: ')
a = 0
while (a < int(numAttacks)) & (totalCompleted < len(net.trafo)):
    if len(net.trafo) == 0:
        raise Exception("Given grid does not contain any transformers")
    if len(net.trafo) == 1:
        i = 0
```

```
    else:
        i = rand.randrange(0, len(net.trafo)-1)
    while transfo_arr[i] != 0:
        i = rand.randrange(0, len(net.trafo) - 1)
    transfo_arr[i] += 1

    net.trafo.loc[[i], "vn_hv_kv"] = net.trafo.at[i, "vn_hv_kv"] * 3

    a += 1
    totalCompleted += 1

pp.runpp(net)

ow = OutputWriter(net, output_path="timeseriesout/FDIout", output_file_type=".xlsx")
ow.log_variable('res_line', 'loading_percent')
pp.timeseries.run_time_series.run_timeseries(net)

print(net.trafo)

pp.plotting.plotly.pf_res_plotly(net, aspectratio=(1.2, 1), figsize=1.2)

shutil.copyfile('temp-plot.html', 'html/FDIplot.html')
```

IoT Attack Script:

```
import pandapower as pp
import numpy as np
import pandapower.create
import pandapower.plotting
import pandapower.timeseries
from pandapower.timeseries.output_writer import OutputWriter
import pandas as pd
import random as rand
import shutil
import sys

sys.stdout.write('Please enter the filepath for the grid you wish to attack. File type must be ".p",
generated by running "python gridfilenamehere.py"\n:')

gridFilePath = input(' ')
```

```
net = pp.from_pickle(filename=gridFilePath, convert=True)




bus_arr = [0] * len(net.bus)

aoe =  rand.randrange(1, 2)

if aoe >= len(net.bus)-1:
    aoe = len(net.bus)-1

numAttacks = input('How many attacks do you want to simulate on this grid?\n: ')
a = 0
while a < int(numAttacks):
    # check if the grid has been filled
    if 0 in bus_arr:
        # randomize a bus number to create an "area" effect around
        centre_bus = rand.randrange(0, len(net.bus)-1)
        while bus_arr[centre_bus] != 0:
            centre_bus = rand.randrange(0, len(net.bus) - 1)

        # X above and center
        for x in range(centre_bus, centre_bus + aoe + 1):
            # check if its been attacked already
            # only compute if it wont go OB
            if x < len(net.bus):
                if bus_arr[x] == 0:
                    net.bus.loc[[x], "vn_kv"] = net.bus.at[x, "vn_kv"] * 1.5
                    bus_arr[x] += 1

        # X below
        for x in range(centre_bus - aoe, centre_bus):
            # check if its been attacked already
            # only compute if it wont go OB
            if x >= 0:
                if bus_arr[x] == 0:
                    net.bus.loc[[x], "vn_kv"] = net.bus.at[x, "vn_kv"] * 1.5
                    bus_arr[x] += 1
```

```
    a += 1

pp.runpp(net)

ow = OutputWriter(net, output_path="timeseriesout/IOTout", output_file_type=".xlsx")
ow.log_variable('res_line', 'loading_percent')
pp.timeseries.run_time_series.run_timeseries(net)


print(centre_bus)
print(net.bus)

pp.plotting.plotly.pf_res_plotly(net, aspectratio=(1.2, 1), figsize=1.2)

shutil.copyfile('temp-plot.html', 'html/IOTplot.html')
```

IowaState.py:

```
import pandapower
import pandapower as pp

net = pp.create_empty_network()

# Ames External Power *MAKE GENERATOR*
# city_of_ames_source = pp.create_bus(net, vn_kv=13.8, name="City_of_Ames_Source")

# Hawthorn Bay 1-4
hawthorn_sub = pp.create_bus(net, vn_kv=13.8, name="Hawthorn_Sub")

# load for substation
pp.create_load(net, bus=hawthorn_sub, p_mw=0.1, q_mvar=0.05, name="Hawthorn_Sub")

# busses
SWGR_ES120 = pp.create_bus(net, vn_kv=13.8, name="SWGR_ES120")
SWGR_ES123 = pp.create_bus(net, vn_kv=13.8, name="SWGR_ES123")
SWGR_ES126 = pp.create_bus(net, vn_kv=13.8, name="SWGR_ES126")

# loads
pp.create_load(net, bus=SWGR_ES120, p_mw=0.1, q_mvar=0.05, name="SWGR_ES120")
```

```
pp.create_load(net, bus=SWGR_ES123, p_mw=0.1, q_mvar=0.05, name="SWGR_ES123")
pp.create_load(net, bus=SWGR_ES126, p_mw=0.1, q_mvar=0.05, name="SWGR_ES126")

# lines
cable_205 = pp.create_line(net, from_bus=hawthorn_sub, to_bus=SWGR_ES120,
length_km=0.2, std_type="NAYY 4x50 SE",
                    name="cable_205")
cable_207 = pp.create_line(net, from_bus=hawthorn_sub, to_bus=SWGR_ES123,
length_km=0.1, std_type="NAYY 4x50 SE",
                    name="cable_207")
cable_7 = pp.create_line(net, from_bus=hawthorn_sub, to_bus=SWGR_ES126, length_km=0.1,
std_type="NAYY 4x50 SE",
                   name="cable_7")
# Switches
sw_ES120 = pp.create_switch(net, element=cable_205, et="l", bus=SWGR_ES120)

# SWGR_ES120 Bay 1-2
# busses
SWGR_ES121 = pp.create_bus(net, vn_kv=13.8, name="SWGR_ES121")
SWGR_ES116 = pp.create_bus(net, vn_kv=13.8, name="SWGR_ES116")

# loads for above buses
pp.create_load(net, bus=SWGR_ES121, p_mw=0.1, q_mvar=0.05, name="SWGR_ES121")
pp.create_load(net, bus=SWGR_ES116, p_mw=0.1, q_mvar=0.05, name="SWGR_ES116")

# Static Generator and Load and External Grid
# City of Ames power source
city_of_ames_source = pp.create_sgen(net, bus=SWGR_ES116, p_mw=13.8,
name="City_of_Ames_Source")
source_power_load = pp.create_load(net, bus=SWGR_ES116, p_mw=13.8)
stupid_external_grid = pp.create_ext_grid(net, bus=SWGR_ES116, name="dumbass")

# lines
cable_17 = pp.create_line(net, from_bus=SWGR_ES120, to_bus=SWGR_ES121,
length_km=0.1, std_type="NAYY 4x50 SE",
                    name="cable_17")
cable_83 = pp.create_line(net, from_bus=SWGR_ES120, to_bus=SWGR_ES116,
length_km=0.4, std_type="NAYY 4x50 SE",
                    name="cable_83")
# switches
```

```
sw_ES120_2 = pp.create_switch(net, element=cable_83, et="l", bus=SWGR_ES120)

# SWGR_ES116 Bay 1-4
# busses
SWGR_ES117 = pp.create_bus(net, vn_kv=13.8, name="SWGR_ES117")
SWGR_ES118 = pp.create_bus(net, vn_kv=13.8, name="SWGR_ES118")

# loads
pp.create_load(net, bus=SWGR_ES117, p_mw=0.1, q_mvar=0.05, name="SWGR_ES117")
pp.create_load(net, bus=SWGR_ES118, p_mw=0.1, q_mvar=0.05, name="SWGR_ES118")

# lines
cable_27 = pp.create_line(net, from_bus=SWGR_ES116, to_bus=city_of_ames_source,
length_km=0.1, std_type="NAYY 4x50 SE",
                  name="cable_27")
cable_29 = pp.create_line(net, from_bus=SWGR_ES116, to_bus=SWGR_ES118,
length_km=0.08, std_type="NAYY 4x50 SE",
                  name="cable_29")
cable_37 = pp.create_line(net, from_bus=SWGR_ES116, to_bus=SWGR_ES117,
length_km=0.3, std_type="NAYY 4x50 SE",
                  name="cable_37")
# switches
sw_ES116 = pp.create_switch(net, element=cable_27, et="l", bus=SWGR_ES116)
sw_ES116_2 = pp.create_switch(net, element=cable_37, et="l", bus=SWGR_ES116)
sw_ES118 = pp.create_switch(net, element=cable_29, et="l", bus=SWGR_ES118)
sw_ES117 = pp.create_switch(net, element=cable_37, et="l", bus=SWGR_ES117)

# Adding in more sections to grid
# NCS sub 4.16
NCS_sub = pp.create_bus(net, vn_kv=4.16, name="NCS_substation")

# Load for NCS sub
pp.create_load(net, bus=NCS_sub, p_mw=0.1, q_mvar=0.05, name="NCS_substation")

# Transformer should be added here to change from substations

pp.create_transformer(net, hv_bus=SWGR_ES117, lv_bus=NCS_sub, name="Hawth_NCS",
std_type="40 MVA 110/20 kV")

# Buses apart of NCS substation
```

```
atrb_vista = pp.create_bus(net, vn_kv=4.16, name="ATRB_Vista")
mol_bio = pp.create_bus(net, vn_kv=4.16, name="Molecular_bio")
insectary = pp.create_bus(net, vn_kv=4.16, name="Insectary")
printing = pp.create_bus(net, vn_kv=4.16, name="Printing")
swine_res = pp.create_bus(net, vn_kv=4.16, name="Swine_research")
carver_co = pp.create_bus(net, vn_kv=4.16, name="Carver_colab")

# loads for above buses
pp.create_load(net, bus=atrb_vista, p_mw=0.1, q_mvar=0.05, name="ATRB_Vista")
pp.create_load(net, bus=mol_bio, p_mw=0.1, q_mvar=0.05, name="Molecular_bio")
pp.create_load(net, bus=insectary, p_mw=0.1, q_mvar=0.05, name="Insectary")
pp.create_load(net, bus=printing, p_mw=0.1, q_mvar=0.05, name="Printing")
pp.create_load(net, bus=swine_res, p_mw=0.1, q_mvar=0.05, name="Swine_research")
pp.create_load(net, bus=carver_co, p_mw=0.1, q_mvar=0.05, name="Carver_colab")

# Lines for NCS sub
cable_21 = pp.create_line(net, from_bus=SWGR_ES117, to_bus=NCS_sub, length_km=0.4,
std_type="NAYY 4x50 SE",
                  name="Cable21")
cable_12 = pp.create_line(net, from_bus=NCS_sub, to_bus=atrb_vista, length_km=0.1,
std_type="NAYY 4x50 SE",
                  name="Cable12")
cable_129 = pp.create_line(net, from_bus=NCS_sub, to_bus=mol_bio, length_km=0.1,
std_type="NAYY 4x50 SE",
                   name="Cable129")
cable_131 = pp.create_line(net, from_bus=NCS_sub, to_bus=insectary, length_km=0.1,
std_type="NAYY 4x50 SE",
                   name="Cable131")
cable_133 = pp.create_line(net, from_bus=NCS_sub, to_bus=printing, length_km=0.1,
std_type="NAYY 4x50 SE",
                   name="Cable133")
cable_134 = pp.create_line(net, from_bus=NCS_sub, to_bus=swine_res, length_km=0.1,
std_type="NAYY 4x50 SE",
                   name="Cable134")
cable_136 = pp.create_line(net, from_bus=NCS_sub, to_bus=carver_co, length_km=0.1,
std_type="NAYY 4x50 SE",
                   name="Cable136")

# Switches for each cable
sw108 = pp.create_switch(net, bus=NCS_sub, element=cable_12, et="l")
```

```
sw4 = pp.create_switch(net, bus=atrb_vista, element=cable_12, et="l")

sw109 = pp.create_switch(net, bus=NCS_sub, element=cable_129, et="l")
sw72 = pp.create_switch(net, bus=mol_bio, element=cable_129, et="l")

sw110 = pp.create_switch(net, bus=NCS_sub, element=cable_131, et="l")
sw77 = pp.create_switch(net, bus=insectary, element=cable_131, et="l")

sw111 = pp.create_switch(net, bus=NCS_sub, element=cable_133, et="l")
sw80 = pp.create_switch(net, bus=printing, element=cable_133, et="l")

sw112 = pp.create_switch(net, bus=NCS_sub, element=cable_134, et="l")
sw81 = pp.create_switch(net, bus=swine_res, element=cable_134, et="l")

sw113 = pp.create_switch(net, bus=NCS_sub, element=cable_136, et="l")
sw83 = pp.create_switch(net, bus=carver_co, element=cable_136, et="l")

# Building up from NCS Sub

bus43 = pp.create_bus(net, vn_kv=4.16, name="Bus43")
bus42 = pp.create_bus(net, vn_kv=4.16, name="Bus42")

# loads
pp.create_load(net, bus=bus43, p_mw=0.1, q_mvar=0.05, name="Bus43")
pp.create_load(net, bus=bus42, p_mw=0.1, q_mvar=0.05, name="Bus42")

# NCS Sub 13.8 East
NCS_subE = pp.create_bus(net, vn_kv=13.8, name="NCS_subE")

# Load for NCS sub E
pp.create_load(net, bus=NCS_subE, p_mw=0.1, q_mvar=0.05, name="NCS_subE")

# I believe another transformer needs to go here

pp.create_transformer(net, hv_bus=NCS_subE, lv_bus=bus42,
name="NCS_Subs_Transformers", std_type="40 MVA 110/20 kV")

# NCS Sub 13.8 West
NCS_subW = pp.create_bus(net, vn_kv=13.8, name="NCS_subW")
```

```
# Load for NCS sub W
pp.create_load(net, bus=NCS_subW, p_mw=0.1, q_mvar=0.05, name="NCS_subW")

# lines to Sub Bay 4
cable_76 = pp.create_line(net, from_bus=bus43, to_bus=NCS_sub, length_km=0.1,
std_type="NAYY 4x50 SE",
                          name="Cable76")
cable_75 = pp.create_line(net, from_bus=bus43, to_bus=NCS_sub, length_km=0.1,
std_type="NAYY 4x50 SE",
                          name="Cable75")
ncs_sub3 = pp.create_line(net, from_bus=NCS_subW, to_bus=NCS_subE, length_km=0.1,
std_type="NAYY 4x50 SE",
                          name="ncs sub bay 3")

# switch between NCS sub's

sw_ncs_sub3 = pp.create_switch(net, bus=NCS_subW, element=ncs_sub3, et="l")

# Branching off towards MWL buildings
mapleHall = pp.create_bus(net, vn_kv=4.16, name="mapleHall")

hixson_lied = pp.create_bus(net, vn_kv=4.16, name="hixson-lied")

mwl416 = pp.create_bus(net, vn_kv=4.16, name="MWL4.16")

# loads for buses above
pp.create_load(net, bus=mapleHall, p_mw=0.1, q_mvar=0.05, name="mapleHall")
pp.create_load(net, bus=hixson_lied, p_mw=0.1, q_mvar=0.05, name="hixon-lied")
pp.create_load(net, bus=mwl416, p_mw=0.1, q_mvar=0.05, name="MWL4.16")

# cable connecting between MWL and substations
cable_72 = pp.create_line(net, from_bus=mwl416, to_bus=NCS_subW, length_km=2.1,
std_type="NAYY 4x50 SE",
                          name="line connecting MWL to NCS stations")

# Yet another transformer for changes in voltages

pp.create_transformer(net, hv_bus=NCS_subW, lv_bus=mwl416, name="Hawth_NCS",
std_type="40 MVA 110/20 kV")
```

```
# MWL 4.16 substation lines
cable_200 = pp.create_line(net, from_bus=mwl416, to_bus=mapleHall, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable200")

cable_203 = pp.create_line(net, from_bus=mwl416, to_bus=hixson_lied, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable203")

# Switches for cables 200/203
sw175 = pp.create_switch(net, bus=mwl416, element=cable_200, et="l")
sw176 = pp.create_switch(net, bus=mapleHall, element=cable_200, et="l")

sw179 = pp.create_switch(net, bus=mwl416, element=cable_203, et="l")
sw180 = pp.create_switch(net, bus=hixson_lied, element=cable_203, et="l")

# building up from MWL 4.16 substation
###May need another transformer here
mwl138 = pp.create_bus(net, vn_kv=13.8, name="MWL 13.8")

# load for MWL 13.8
pp.create_load(net, bus=NCS_subE, p_mw=0.8, q_mvar=0.35, name="MWL 13.8")

cable_197 = pp.create_line(net, from_bus=mwl138, to_bus=mwl416, length_km=2.1,
std_type="NAYY 4x50 SE",
                    name="Line connecting MWL substations together")

sw172 = pp.create_switch(net, bus=mwl416, element=cable_197, et="l")

# Building over to the commons and alumni center
larchHall = pp.create_bus(net, vn_kv=13.8, name="Larch Hall")

alumniCenter = pp.create_bus(net, vn_kv=13.8, name="Alumni Center")

# loads for above buses
pp.create_load(net, bus=larchHall, p_mw=0.1, q_mvar=0.05, name="Larch Hall")
pp.create_load(net, bus=alumniCenter, p_mw=0.1, q_mvar=0.05, name="Alumni Center")

# Line connecting from MWL power source to Larch hall
```

```
cable_208 = pp.create_line(net, from_bus=mwl138, to_bus=larchHall, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable208")

# Switches for cable 208
sw182 = pp.create_switch(net, bus=mwl138, element=cable_208, et="l")
sw184 = pp.create_switch(net, bus=larchHall, element=cable_208, et="l")

# branching down from larch hall
cable_220 = pp.create_line(net, from_bus=larchHall, to_bus=alumniCenter, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable220")

sw190 = pp.create_switch(net, bus=alumniCenter, element=cable_220, et="l")

# EE Substation
EE_sub = pp.create_bus(net, vn_kv=4.16, name="EE_sub")

# load for EE substation
pp.create_load(net, bus=EE_sub, p_mw=0.1, q_mvar=0.05, name="EE sub")

# transformer for EE substation
pp.create_transformer(net, hv_bus=alumniCenter, lv_bus=EE_sub, name="alumni_EEsub",
std_type="40 MVA 110/20 kV")

# Teleporting to Durham
Durham = pp.create_bus(net, vn_kv=4.16, name="Durham")
Coover = pp.create_bus(net, vn_kv=4.16, name="Coover")
Snedecor = pp.create_bus(net, vn_kv=4.16, name="Snedecor")
Davidson = pp.create_bus(net, vn_kv=4.16, name="Davidson")
DesignCenter = pp.create_bus(net, vn_kv=4.16, name="Design Center")
Sweeney = pp.create_bus(net, vn_kv=4.16, name="Sweeney")

# loads for buildings above
pp.create_load(net, bus=Durham, p_mw=0.2, q_mvar=0.3, name="Durham")
pp.create_load(net, bus=Coover, p_mw=0.3, q_mvar=0.55, name="Coover")
pp.create_load(net, bus=Snedecor, p_mw=0.3, q_mvar=0.55, name="Snedecor")
pp.create_load(net, bus=Davidson, p_mw=0.1, q_mvar=0.05, name="Davidson")
pp.create_load(net, bus=DesignCenter, p_mw=0.25, q_mvar=0.25, name="Design Center")
pp.create_load(net, bus=Sweeney, p_mw=0.15, q_mvar=0.05, name="Sweeny")
```

```
# lines connecting to EE substation
cable_15 = pp.create_line(net, from_bus=alumniCenter, to_bus=EE_sub, length_km=0.7,
std_type="NAYY 4x50 SE",
                          name="Cable15")
# lines for each building
cable_101 = pp.create_line(net, from_bus=EE_sub, to_bus=Durham, length_km=0.1,
std_type="NAYY 4x50 SE",
                           name="Cable101")
cable_104 = pp.create_line(net, from_bus=EE_sub, to_bus=Coover, length_km=0.1,
std_type="NAYY 4x50 SE",
                           name="Cable104")
cable_106 = pp.create_line(net, from_bus=EE_sub, to_bus=Snedecor, length_km=0.1,
std_type="NAYY 4x50 SE",
                           name="Cable106")
cable_108 = pp.create_line(net, from_bus=EE_sub, to_bus=Davidson, length_km=0.1,
std_type="NAYY 4x50 SE",
                           name="Cable108")
cable_110 = pp.create_line(net, from_bus=EE_sub, to_bus=DesignCenter, length_km=0.1,
std_type="NAYY 4x50 SE",
                           name="Cable110")
cable_113 = pp.create_line(net, from_bus=EE_sub, to_bus=Sweeney, length_km=0.1,
std_type="NAYY 4x50 SE",
                           name="Cable113")

# switches on each cable
sw90 = pp.create_switch(net, bus=Durham, element=cable_101, et="l")
sw48 = pp.create_switch(net, bus=Durham, element=cable_101, et="l")

sw93 = pp.create_switch(net, bus=Coover, element=cable_104, et="l")
sw49 = pp.create_switch(net, bus=Coover, element=cable_104, et="l")

sw94 = pp.create_switch(net, bus=Snedecor, element=cable_106, et="l")
sw57 = pp.create_switch(net, bus=Snedecor, element=cable_106, et="l")

sw95 = pp.create_switch(net, bus=Davidson, element=cable_108, et="l")
sw58 = pp.create_switch(net, bus=Davidson, element=cable_108, et="l")

sw96 = pp.create_switch(net, bus=DesignCenter, element=cable_110, et="l")
sw59 = pp.create_switch(net, bus=DesignCenter, element=cable_110, et="l")
```

```
sw97 = pp.create_switch(net, bus=Sweeney, element=cable_113, et="l")
sw60 = pp.create_switch(net, bus=Sweeney, element=cable_113, et="l")

# Building up from EE Sub 4.16
EE_sub13_8 = pp.create_bus(net, vn_kv=13.8, name="EE_sub13_8")

# load for EE sub 13.8
pp.create_load(net, bus=EE_sub13_8, p_mw=0.9, q_mvar=0.35, name="EE sub 13.8")

# transformer between EE substations
pp.create_transformer(net, hv_bus=EE_sub13_8, lv_bus=EE_sub, name="T4-EE Sub",
std_type="40 MVA 110/20 kV")

# Cables between substations
cable_14 = pp.create_line(net, from_bus=EE_sub, to_bus=EE_sub13_8, length_km=0.3,
std_type="NAYY 4x50 SE",
                    name="Cable14")

# switch for cable between EE subs
sw12 = pp.create_switch(net, bus=EE_sub, element=cable_14, et="l")

# Moving to Pearson halls substations
pearsonSub_138 = pp.create_bus(net, vn_kv=13.8, name="PearsonSub_13.8")
pearsonSub_416 = pp.create_bus(net, vn_kv=4.16, name="PearsonSub_4.16")

# loads for Pearson substations
pp.create_load(net, bus=pearsonSub_138, p_mw=0.7, q_mvar=0.35, name="Pearson 13.8")
pp.create_load(net, bus=pearsonSub_416, p_mw=0.1, q_mvar=0.05, name="Pearson 4.16")

# transformer between Pearson substations
pp.create_transformer(net, hv_bus=pearsonSub_138, lv_bus=pearsonSub_416,
name="T1-Pearson Sub",
                  std_type="40 MVA 110/20 kV")

# Cables connecting substation to Pearson
cable_242 = pp.create_line(net, from_bus=EE_sub13_8, to_bus=pearsonSub_138,
length_km=0.5, std_type="NAYY 4x50 SE",
                    name="Cable242")
```

```
cable_144 = pp.create_line(net, from_bus=pearsonSub_138, to_bus=pearsonSub_416,
length_km=0.3, std_type="NAYY 4x50 SE",
                           name="Cable144")

# Switch to Pearson substaion 4.16
sw75 = pp.create_switch(net, bus=pearsonSub_416, element=cable_144, et="l")

# Buses apart of Pearson Sub 4.16
Carver = pp.create_bus(net, vn_kv=4.16, name="Carver")
Beardshear = pp.create_bus(net, vn_kv=4.16, name="BeardShear")
StudentServices = pp.create_bus(net, vn_kv=4.16, name="Student Services")
Pearson = pp.create_bus(net, vn_kv=4.16, name="Pearson")

# loads for buildings above
pp.create_load(net, bus=Carver, p_mw=2.5, q_mvar=0.5, name="Carver")
pp.create_load(net, bus=Beardshear, p_mw=1.5, q_mvar=0.25, name="Beardshear")
pp.create_load(net, bus=StudentServices, p_mw=1.7, q_mvar=0.35, name="Student Services")
pp.create_load(net, bus=Pearson, p_mw=2.5, q_mvar=0.5, name="Pearson")

# Cables for each bus on Pearson sub
cable_68 = pp.create_line(net, from_bus=pearsonSub_416, to_bus=Carver, length_km=0.15,
std_type="NAYY 4x50 SE",
                          name="Cable68")
cable_94 = pp.create_line(net, from_bus=pearsonSub_416, to_bus=Beardshear, length_km=0.1,
std_type="NAYY 4x50 SE",
                          name="Cable94")
cable_2_1 = pp.create_line(net, from_bus=pearsonSub_416, to_bus=StudentServices,
length_km=0.1, std_type="NAYY 4x50 SE",
                           name="Cable2_1")
cable_40 = pp.create_line(net, from_bus=pearsonSub_416, to_bus=Pearson, length_km=0.1,
std_type="NAYY 4x50 SE",
                          name="Cable40")

# Switches for each of the above cables
sw33 = pp.create_switch(net, bus=Carver, element=cable_68, et="l")

sw36 = pp.create_switch(net, bus=Beardshear, element=cable_94, et="l")

sw41 = pp.create_switch(net, bus=StudentServices, element=cable_2_1, et="l")
```

```
sw45 = pp.create_switch(net, bus=Pearson, element=cable_40, et="l")

# Student services has 2 buses built off of it
Bus51 = pp.create_bus(net, vn_kv=0.208, name="Bus51")
Bus74 = pp.create_bus(net, vn_kv=0.208, name="Bus74")

# loads for buses above
pp.create_load(net, bus=Bus51, p_mw=0.1, q_mvar=0.25, name="Bus51")
pp.create_load(net, bus=Bus74, p_mw=0.1, q_mvar=0.25, name="Bus74")

# Transformers for each of the sub buses
pp.create_transformer(net, hv_bus=StudentServices, lv_bus=Bus51, name="Lab Mech",
std_type="40 MVA 110/20 kV")

pp.create_transformer(net, hv_bus=StudentServices, lv_bus=Bus74, name="Student 208",
std_type="40 MVA 110/20 kV")

# Cables for sub buses of student services
cable_301 = pp.create_line(net, from_bus=StudentServices, to_bus=Bus51, length_km=0.1,
std_type="NAYY 4x50 SE",
                     name="Cable31")
cable_46 = pp.create_line(net, from_bus=StudentServices, to_bus=Bus74, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable46")
# No switches on the above sub buses

# Building over from Pearson 13.8 to ME Sub station
MEsub_138 = pp.create_bus(net, vn_kv=13.8, name="ME Sub_13.8")

# load for ME sub 13.8
pp.create_load(net, bus=MEsub_138, p_mw=0.9, q_mvar=0.35, name="ME sub 13.8")

# Connecting cable between sub stations
cable_1 = pp.create_line(net, from_bus=EE_sub13_8, to_bus=MEsub_138, length_km=0.4,
std_type="NAYY 4x50 SE",
                    name="Cable1")

# Building substation and buses inside of ME sub
MEsub_416 = pp.create_bus(net, vn_kv=4.16, name="ME Sub_4.16")
SIC = pp.create_bus(net, vn_kv=4.16, name="SIC")
```

```
StudentHealth = pp.create_bus(net, vn_kv=4.16, name="Student Health")
BlackEngr = pp.create_bus(net, vn_kv=4.16, name="Black Engr")
StateGym = pp.create_bus(net, vn_kv=4.16, name="State Gym")

# loads for buildings above
pp.create_load(net, bus=MEsub_416, p_mw=0.1, q_mvar=0.05, name="ME sub 4.16")
pp.create_load(net, bus=SIC, p_mw=0.1, q_mvar=0.05, name="SIC")
pp.create_load(net, bus=StudentHealth, p_mw=0.1, q_mvar=0.05, name="Student Health")
pp.create_load(net, bus=BlackEngr, p_mw=0.1, q_mvar=0.05, name="Black Engr")
pp.create_load(net, bus=StateGym, p_mw=0.1, q_mvar=0.05, name="State Gym")

# Transformer for ME substations
pp.create_transformer(net, hv_bus=MEsub_138, lv_bus=MEsub_416, name="T1-ME Sub",
std_type="40 MVA 110/20 kV")

# Cables connecting subs and buses
cable_02 = pp.create_line(net, from_bus=MEsub_138, to_bus=MEsub_416, length_km=0.1,
std_type="NAYY 4x50 SE",
                          name="Cable2")
cable_115 = pp.create_line(net, from_bus=MEsub_416, to_bus=SIC, length_km=0.1,
std_type="NAYY 4x50 SE",
                           name="Cable115")
cable_117 = pp.create_line(net, from_bus=MEsub_416, to_bus=StudentHealth, length_km=0.1,
std_type="NAYY 4x50 SE",
                           name="Cable117")
cable_120 = pp.create_line(net, from_bus=MEsub_416, to_bus=BlackEngr, length_km=0.1,
std_type="NAYY 4x50 SE",
                           name="Cable120")
cable_122 = pp.create_line(net, from_bus=MEsub_416, to_bus=StateGym, length_km=0.1,
std_type="NAYY 4x50 SE",
                           name="Cable122")

# switches apart of ME 4.16 substation
sw19 = pp.create_switch(net, bus=SIC, element=cable_115, et="l")

sw23 = pp.create_switch(net, bus=StudentHealth, element=cable_117, et="l")

sw24 = pp.create_switch(net, bus=BlackEngr, element=cable_120, et="l")

sw32 = pp.create_switch(net, bus=StateGym, element=cable_122, et="l")
```

```
# Cable connecting Pearson 4.16 sub to ME 4.16 sub
cable_18 = pp.create_line(net, from_bus=pearsonSub_416, to_bus=MEsub_416, length_km=0.2,
std_type="NAYY 4x50 SE",
                          name="Cable18")

# Switch for above cable
sw09 = pp.create_switch(net, bus=MEsub_416, element=cable_18, et="l")

# Building over to VAC Clad substations
VacClad_138 = pp.create_bus(net, vn_kv=13.8, name="Vac Clad 13.8 sub")

# load for VAC sub
pp.create_load(net, bus=VacClad_138, p_mw=0.9, q_mvar=0.35, name="VAC sub 13.8")

# Connecting cable from ME sub
cable_24 = pp.create_line(net, from_bus=MEsub_138, to_bus=VacClad_138, length_km=1.2,
std_type="NAYY 4x50 SE",
                          name="Cable24")

# Building Vac clad 4.16 sub and buses
VacClad_416 = pp.create_bus(net, vn_kv=4.16, name="Vac Clad 4.16 sub")
GeneralServices = pp.create_bus(net, vn_kv=4.16, name="General Services")
HeatingPlant = pp.create_bus(net, vn_kv=4.16, name="Heating Plant")
LandscapeArch = pp.create_bus(net, vn_kv=4.16, name="Landscape Arch(B)")
HamilonHall = pp.create_bus(net, vn_kv=4.16, name="Hamilton Hall")
Forker = pp.create_bus(net, vn_kv=4.16, name="Forker")
BesseyVISTA = pp.create_bus(net, vn_kv=4.16, name="Bessey VISTA")

# Transformer between VAC subs
pp.create_transformer(net, hv_bus=VacClad_138, lv_bus=VacClad_416, name="T1-VAC Sub",
std_type="40 MVA 110/20 kV")

# loads for buildings above
pp.create_load(net, bus=VacClad_416, p_mw=0.5, q_mvar=4.16, name="Vac sub 4.16")
pp.create_load(net, bus=GeneralServices, p_mw=0.2, q_mvar=1.15, name="General Services")
pp.create_load(net, bus=HeatingPlant, p_mw=0.2, q_mvar=1.15, name="Heating Plant")
pp.create_load(net, bus=LandscapeArch, p_mw=0.2, q_mvar=1.15, name="Landscape
Arch(B)")
pp.create_load(net, bus=HamilonHall, p_mw=0.2, q_mvar=1.15, name="Hamilton Hall")
```

```
pp.create_load(net, bus=Forker, p_mw=0.2, q_mvar=1.15, name="Forker")
pp.create_load(net, bus=BesseyVISTA, p_mw=0.2, q_mvar=1.15, name="Bessey VISTA")

# Feeding into smaller substation
SEFeeder = pp.create_bus(net, vn_kv=4.16, name="SE Feeder")

# load for SE Feeder
pp.create_load(net, bus=SEFeeder, p_mw=0.2, q_mvar=2.2, name="SE Feeder")

# Cables associated with VAC 4.16 sub
cable_173 = pp.create_line(net, from_bus=VacClad_416, to_bus=GeneralServices,
length_km=0.1, std_type="NAYY 4x50 SE",
                    name="Cable173")
cable_175 = pp.create_line(net, from_bus=VacClad_416, to_bus=HeatingPlant, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable175")
cable_177 = pp.create_line(net, from_bus=VacClad_416, to_bus=LandscapeArch,
length_km=0.1, std_type="NAYY 4x50 SE",
                    name="Cable177")
cable_181 = pp.create_line(net, from_bus=VacClad_416, to_bus=HamilonHall, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable181")
cable_183 = pp.create_line(net, from_bus=VacClad_416, to_bus=Forker, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable183")
cable_28 = pp.create_line(net, from_bus=VacClad_416, to_bus=BesseyVISTA, length_km=0.1,
std_type="NAYY 4x50 SE",
                   name="Cable28")
cable_036 = pp.create_line(net, from_bus=VacClad_416, to_bus=SEFeeder, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable36")

# Switches on cables listed above
sw149 = pp.create_switch(net, bus=GeneralServices, element=cable_173, et="l")

sw153 = pp.create_switch(net, bus=HeatingPlant, element=cable_175, et="l")

sw157 = pp.create_switch(net, bus=LandscapeArch, element=cable_177, et="l")

sw161 = pp.create_switch(net, bus=HamilonHall, element=cable_181, et="l")
```

```
sw165 = pp.create_switch(net, bus=Forker, element=cable_183, et="l")

sw016 = pp.create_switch(net, bus=BesseyVISTA, element=cable_28, et="l")

sw37 = pp.create_switch(net, bus=SEFeeder, element=cable_036, et="l")

# SE Feeder buildings
Forker_PMH = pp.create_bus(net, vn_kv=4.16, name="Forker PMH 5")
ES11A = pp.create_bus(net, vn_kv=4.16, name="ES11A")

# loads for buses above
pp.create_load(net, bus=Forker_PMH, p_mw=0.2, q_mvar=1.15, name="Forker PMH")
pp.create_load(net, bus=ES11A, p_mw=0.2, q_mvar=1.15, name="ES11A")

# Cables for buildings and substation
cable_39 = pp.create_line(net, from_bus=SEFeeder, to_bus=Forker_PMH, length_km=0.1,
std_type="NAYY 4x50 SE",
                          name="Cable39")
cable_41 = pp.create_line(net, from_bus=SEFeeder, to_bus=ES11A, length_km=0.1,
std_type="NAYY 4x50 SE",
                          name="Cable41")

# Switches for each cable
sw40 = pp.create_switch(net, bus=SEFeeder, element=cable_39, et="l")

sw39 = pp.create_switch(net, bus=Forker_PMH, element=cable_39, et="l")

sw43 = pp.create_switch(net, bus=SEFeeder, element=cable_41, et="l")

sw44 = pp.create_switch(net, bus=ES11A, element=cable_41, et="l")

# bus inside of ES11A
HeadyHall = pp.create_bus(net, vn_kv=0.208, name="Heady Hall")

# load for Heady Hall
pp.create_load(net, bus=HeadyHall, p_mw=0.1, q_mvar=0.15, name="ES11A")

# Transformer between ES11A and Heady hall
```

```
pp.create_transformer(net, hv_bus=ES11A, lv_bus=HeadyHall, name="Heady Hall
transformer", std_type="40 MVA 110/20 kV")

# cable for Heady hall
cable_45 = pp.create_line(net, from_bus=ES11A, to_bus=HeadyHall, length_km=0.15,
std_type="NAYY 4x50 SE",
                          name="Cable45")

# switch for Heady hall- always open
sw46 = pp.create_switch(net, bus=HeadyHall, element=cable_45, et="l")

# Building out to another substation from VAC 4.16
PPscSwgrSouth = pp.create_bus(net, vn_kv=4.16, name="PP S&C Swgr South")
BesseyVISTA2 = pp.create_bus(net, vn_kv=4.16, name="Bessey VISTA2")
EastAgronomy = pp.create_bus(net, vn_kv=4.16, name="East Agronomy")
LandscapeArchA = pp.create_bus(net, vn_kv=4.16, name="Landscape Arch(A)")
Troxel = pp.create_bus(net, vn_kv=4.16, name="Troxel")
JischkeHonors = pp.create_bus(net, vn_kv=4.16, name="Jischke Honors")
SouthKildee = pp.create_bus(net, vn_kv=4.16, name="South Kildee")

# loads for buildings above
pp.create_load(net, bus=PPscSwgrSouth, p_mw=0.25, q_mvar=1.4, name="ES11A")
pp.create_load(net, bus=BesseyVISTA2, p_mw=0.25, q_mvar=1.4, name="ES11A")
pp.create_load(net, bus=EastAgronomy, p_mw=0.25, q_mvar=1.4, name="ES11A")
pp.create_load(net, bus=LandscapeArchA, p_mw=0.25, q_mvar=1.4, name="ES11A")
pp.create_load(net, bus=Troxel, p_mw=0.3, q_mvar=2.5, name="ES11A")
pp.create_load(net, bus=JischkeHonors, p_mw=0.1, q_mvar=1.15, name="ES11A")
pp.create_load(net, bus=SouthKildee, p_mw=0.15, q_mvar=0.75, name="ES11A")

# cables for each building
cable_259 = pp.create_line(net, from_bus=VacClad_416, to_bus=PPscSwgrSouth,
length_km=1.2, std_type="NAYY 4x50 SE",
                           name="Cable259")
cable_32 = pp.create_line(net, from_bus=PPscSwgrSouth, to_bus=BesseyVISTA2,
length_km=0.1, std_type="NAYY 4x50 SE",
                          name="Cable32")
cable_167 = pp.create_line(net, from_bus=PPscSwgrSouth, to_bus=EastAgronomy,
length_km=0.1, std_type="NAYY 4x50 SE",
                           name="Cable167")
```

```
cable_165 = pp.create_line(net, from_bus=PPscSwgrSouth, to_bus=LandscapeArchA,
length_km=0.1, std_type="NAYY 4x50 SE",
                    name="Cable165")
cable_163 = pp.create_line(net, from_bus=PPscSwgrSouth, to_bus=Troxel, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable163")
cable_169 = pp.create_line(net, from_bus=PPscSwgrSouth, to_bus=JischkeHonors,
length_km=0.1, std_type="NAYY 4x50 SE",
                    name="Cable169")
cable_171 = pp.create_line(net, from_bus=PPscSwgrSouth, to_bus=SouthKildee,
length_km=0.1, std_type="NAYY 4x50 SE",
                    name="Cable171")
# switches for each bus
sw25 = pp.create_switch(net, bus=PPscSwgrSouth, element=cable_32, et="l")
sw31 = pp.create_switch(net, bus=BesseyVISTA2, element=cable_32, et="l")

sw142 = pp.create_switch(net, bus=PPscSwgrSouth, element=cable_167, et="l")
sw143 = pp.create_switch(net, bus=EastAgronomy, element=cable_167, et="l")

sw140 = pp.create_switch(net, bus=PPscSwgrSouth, element=cable_165, et="l")
sw141 = pp.create_switch(net, bus=LandscapeArchA, element=cable_165, et="l")

sw138 = pp.create_switch(net, bus=PPscSwgrSouth, element=cable_163, et="l")
sw139 = pp.create_switch(net, bus=Troxel, element=cable_163, et="l")

sw144 = pp.create_switch(net, bus=PPscSwgrSouth, element=cable_169, et="l")
sw145 = pp.create_switch(net, bus=JischkeHonors, element=cable_169, et="l")

sw146 = pp.create_switch(net, bus=PPscSwgrSouth, element=cable_171, et="l")
sw147 = pp.create_switch(net, bus=SouthKildee, element=cable_171, et="l")

# Branching off from Vac Clad 13.8 to go to NW sub 13.8
NWsub138 = pp.create_bus(net, vn_kv=13.8, name="NW sub 13.8")

# load for NW sub
pp.create_load(net, bus=NWsub138, p_mw=3.5, q_mvar=10.5, name="ES11A")

# cable connecting Vac to NW sub
cable_030 = pp.create_line(net, from_bus=VacClad_138, to_bus=NWsub138, length_km=1.7,
std_type="NAYY 4x50 SE",
```

```
                    name="Cable030")
# switch between Vac Clad and NW sub
sw29 = pp.create_switch(net, bus=NWsub138, element=cable_030, et="l")

# Branching off to NW sub 4.16
NWsub416 = pp.create_bus(net, vn_kv=4.16, name="NW sub 4.16")
BRLEast = pp.create_bus(net, vn_kv=4.16, name="BRL East")
HachHall = pp.create_bus(net, vn_kv=4.16, name="Hach Hall")
TownEngr = pp.create_bus(net, vn_kv=4.16, name="Town Engr")
Armory = pp.create_bus(net, vn_kv=4.16, name="Armory")

# loads for above buildings
pp.create_load(net, bus=NWsub416, p_mw=0.25, q_mvar=1.4, name="ES11A")
pp.create_load(net, bus=BRLEast, p_mw=0.25, q_mvar=1.4, name="ES11A")
pp.create_load(net, bus=HachHall, p_mw=0.25, q_mvar=1.4, name="ES11A")
pp.create_load(net, bus=TownEngr, p_mw=0.25, q_mvar=1.4, name="ES11A")
pp.create_load(net, bus=Armory, p_mw=0.15, q_mvar=1.1, name="ES11A")

# transformer for NW subs
pp.create_transformer(net, hv_bus=NWsub138, lv_bus=NWsub416, name="NW substation
transformer",
                      std_type="40 MVA 110/20 kV")

# cables for buses above
cable_62 = pp.create_line(net, from_bus=NWsub138, to_bus=NWsub416, length_km=0.5,
std_type="NAYY 4x50 SE",
                          name="Cable62")
cable_125 = pp.create_line(net, from_bus=NWsub416, to_bus=BRLEast, length_km=0.2,
std_type="NAYY 4x50 SE",
                           name="Cable125")
cable_127 = pp.create_line(net, from_bus=NWsub416, to_bus=HachHall, length_km=0.3,
std_type="NAYY 4x50 SE",
                           name="Cable127")
cable_88 = pp.create_line(net, from_bus=NWsub416, to_bus=TownEngr, length_km=0.4,
std_type="NAYY 4x50 SE",
                          name="Cable88")
cable_82 = pp.create_line(net, from_bus=NWsub416, to_bus=Armory, length_km=0.3,
std_type="NAYY 4x50 SE",
                          name="Cable82")
```

```
# Switches on each cable
sw106 = pp.create_switch(net, bus=BRLEast, element=cable_125, et="l")

sw107 = pp.create_switch(net, bus=HachHall, element=cable_127, et="l")

sw66 = pp.create_switch(net, bus=TownEngr, element=cable_88, et="l")

sw68 = pp.create_switch(net, bus=Armory, element=cable_82, et="l")

# building off of NCS sub W
AdminSerBuild = pp.create_bus(net, vn_kv=13.8, name="Administrative Services Building")
AdvanTeachResBuild = pp.create_bus(net, vn_kv=13.8, name="Advanced Teaching and
Research Building")
Sub138 = pp.create_bus(net, vn_kv=13.8, name="Substation for additional Halls")

# Loads for the buildings and sub
pp.create_load(net, bus=AdminSerBuild, p_mw=1278, q_mvar=1.1, name="Admin Services
Building")
pp.create_load(net, bus=AdvanTeachResBuild, p_mw=2640, q_mvar=2.3, name="Advanced
Teaching and Research Building")
pp.create_load(net, bus=Sub138, p_mw=524, q_mvar=0.7, name="Sub 13.8")

# lines for each building
cable_121 = pp.create_line(net, from_bus=NCS_subW, to_bus=AdminSerBuild, length_km=2.7,
std_type="NAYY 4x50 SE",
                           name="Cable121")
cable_57 = pp.create_line(net, from_bus=AdminSerBuild, to_bus=AdvanTeachResBuild,
length_km=0.7,
                          std_type="NAYY 4x50 SE",
                          name="Cable57")
cable_58 = pp.create_line(net, from_bus=AdvanTeachResBuild, to_bus=Sub138,
length_km=1.3, std_type="NAYY 4x50 SE",
                          name="Cable58")
# switches on each cable
sw79 = pp.create_switch(net, bus=AdminSerBuild, element=cable_121, et="l")

sw368 = pp.create_switch(net, bus=AdvanTeachResBuild, element=cable_57, et="l")

sw268 = pp.create_switch(net, bus=Sub138, element=cable_58, et="l")
```

```
# Buildings apart of Substation 13.8
Sub416 = pp.create_bus(net, vn_kv=4.16, name="Sub 4.16")
Atanasoff = pp.create_bus(net, vn_kv=4.16, name="Atanasoff Hall")
Beyer = pp.create_bus(net, vn_kv=4.16, name="Beyer Hall")
BartonRes = pp.create_bus(net, vn_kv=4.16, name="Barton Residence Hall")
Catt = pp.create_bus(net, vn_kv=4.16, name="Catt Hall")
Curtis = pp.create_bus(net, vn_kv=4.16, name="Curtis Hall")

# transformer for Subs above
pp.create_transformer(net, hv_bus=Sub138, lv_bus=Sub416, name="13.8 - 4.16kv sub
transformer",
                      std_type="40 MVA 110/20 kV")

# Loads for above buildings
pp.create_load(net, bus=Sub416, p_mw=381, q_mvar=0.6, name="Sub 4.16")
pp.create_load(net, bus=Atanasoff, p_mw=489, q_mvar=0.7, name="Atanasoff")
pp.create_load(net, bus=Beyer, p_mw=497, q_mvar=0.7, name="Beyer")
pp.create_load(net, bus=BartonRes, p_mw=779, q_mvar=1.1, name="Barton Residence Hall")
pp.create_load(net, bus=Catt, p_mw=626, q_mvar=0.8, name="Catt Hall")
pp.create_load(net, bus=Curtis, p_mw=523, q_mvar=0.6, name="Curtis Hall")

# lines for each building
cable_01 = pp.create_line(net, from_bus=Sub138, to_bus=Sub416, length_km=0.7,
std_type="NAYY 4x50 SE",
                          name="Cable01")
cable_010 = pp.create_line(net, from_bus=Sub416, to_bus=Atanasoff, length_km=0.5,
std_type="NAYY 4x50 SE",
                           name="Cable010")
cable_011 = pp.create_line(net, from_bus=Sub416, to_bus=Beyer, length_km=0.5,
std_type="NAYY 4x50 SE",
                           name="Cable011")
cable_012 = pp.create_line(net, from_bus=Sub416, to_bus=BartonRes, length_km=0.5,
std_type="NAYY 4x50 SE",
                           name="Cable012")
cable_013 = pp.create_line(net, from_bus=Sub416, to_bus=Catt, length_km=0.5,
std_type="NAYY 4x50 SE",
                           name="Cable013")
cable_014 = pp.create_line(net, from_bus=Sub416, to_bus=Curtis, length_km=0.5,
std_type="NAYY 4x50 SE",
                           name="Cable014")
```

```
# Switches for each cable
sw01 = pp.create_switch(net, bus=Sub416, element=cable_01, et="l")

sw010 = pp.create_switch(net, bus=Atanasoff, element=cable_010, et="l")
sw73 = pp.create_switch(net, bus=Atanasoff, element=cable_010, et="l")

sw011 = pp.create_switch(net, bus=Beyer, element=cable_011, et="l")
sw76 = pp.create_switch(net, bus=Beyer, element=cable_011, et="l")

sw012 = pp.create_switch(net, bus=BartonRes, element=cable_012, et="l")
sw74 = pp.create_switch(net, bus=BartonRes, element=cable_012, et="l")

sw013 = pp.create_switch(net, bus=Catt, element=cable_013, et="l")
sw71 = pp.create_switch(net, bus=Catt, element=cable_013, et="l")

sw014 = pp.create_switch(net, bus=Curtis, element=cable_014, et="l")
sw74 = pp.create_switch(net, bus=Curtis, element=cable_014, et="l")

# Building off from Curtis
Ross = pp.create_bus(net, vn_kv=4.16, name="Ross Hall")
ForestryGH = pp.create_bus(net, vn_kv=4.16, name="Forestry Greenhouse")

# Loads
pp.create_load(net, bus=Ross, p_mw=408, q_mvar=0.6, name="Ross Hall")
pp.create_load(net, bus=ForestryGH, p_mw=922, q_mvar=1.2, name="Forestry Greenhouse")

# lines for connecting buildings
cable_015 = pp.create_line(net, from_bus=Curtis, to_bus=Ross, length_km=0.3,
std_type="NAYY 4x50 SE",
                    name="Cable015")
cable_016 = pp.create_line(net, from_bus=Ross, to_bus=ForestryGH, length_km=0.7,
std_type="NAYY 4x50 SE",
                    name="Cable016")

# switches for each cable
sw015 = pp.create_switch(net, bus=Ross, element=cable_015, et="l")

sw016 = pp.create_switch(net, bus=ForestryGH, element=cable_016, et="l")
```

```
# Substation connecting to Forestry Greenhouse
FoodScienceSub = pp.create_bus(net, vn_kv=4.16, name="Food Sciences sub")
Martin = pp.create_bus(net, vn_kv=4.16, name="Martin Hall")
Palmer = pp.create_bus(net, vn_kv=4.16, name="Palmer Building")
Morrill = pp.create_bus(net, vn_kv=4.16, name="Morrill Hall")
MacKay = pp.create_bus(net, vn_kv=4.16, name="MacKay Hall")
Kildee = pp.create_bus(net, vn_kv=4.16, name="Kildee Hall")
Linden = pp.create_bus(net, vn_kv=4.16, name="Linden Hall")

# Loads for each building
pp.create_load(net, bus=FoodScienceSub, p_mw=867, q_mvar=1.1, name="Food Science
Substations")
pp.create_load(net, bus=Martin, p_mw=610, q_mvar=0.8, name="Martin Hall")
pp.create_load(net, bus=Palmer, p_mw=646, q_mvar=0.8, name="Palmer Building")
pp.create_load(net, bus=Morrill, p_mw=449, q_mvar=0.6, name="Morrill Hall")
pp.create_load(net, bus=MacKay, p_mw=558, q_mvar=0.7, name="MacKay Hall")
pp.create_load(net, bus=Kildee, p_mw=717, q_mvar=0.8, name="Kildee Hall")
pp.create_load(net, bus=Linden, p_mw=837, q_mvar=0.9, name="Linden Hall")

# Lines connecting from Food Science Substation
cable_017 = pp.create_line(net, from_bus=ForestryGH, to_bus=FoodScienceSub,
length_km=0.7, std_type="NAYY 4x50 SE",
                           name="Cable017")
cable_018 = pp.create_line(net, from_bus=FoodScienceSub, to_bus=Martin, length_km=0.5,
std_type="NAYY 4x50 SE",
                           name="Cable018")
cable_019 = pp.create_line(net, from_bus=FoodScienceSub, to_bus=Palmer, length_km=0.5,
std_type="NAYY 4x50 SE",
                           name="Cable019")
cable_020 = pp.create_line(net, from_bus=FoodScienceSub, to_bus=Morrill, length_km=0.2,
std_type="NAYY 4x50 SE",
                           name="Cable020")
cable_021 = pp.create_line(net, from_bus=FoodScienceSub, to_bus=MacKay, length_km=0.5,
std_type="NAYY 4x50 SE",
                           name="Cable021")
cable_022 = pp.create_line(net, from_bus=FoodScienceSub, to_bus=Kildee, length_km=0.9,
std_type="NAYY 4x50 SE",
                           name="Cable022")
cable_023 = pp.create_line(net, from_bus=FoodScienceSub, to_bus=Linden, length_km=0.7,
std_type="NAYY 4x50 SE",
```

```
                    name="Cable023")

# Switches on each cable
sw015 = pp.create_switch(net, bus=FoodScienceSub, element=cable_017, et="l")

sw016 = pp.create_switch(net, bus=Martin, element=cable_018, et="l")
sw017 = pp.create_switch(net, bus=Martin, element=cable_018, et="l")

sw018 = pp.create_switch(net, bus=Palmer, element=cable_019, et="l")
sw019 = pp.create_switch(net, bus=Palmer, element=cable_019, et="l")

sw020 = pp.create_switch(net, bus=Morrill, element=cable_020, et="l")
sw021 = pp.create_switch(net, bus=Morrill, element=cable_020, et="l")

sw022 = pp.create_switch(net, bus=MacKay, element=cable_021, et="l")
sw023 = pp.create_switch(net, bus=MacKay, element=cable_021, et="l")

sw024 = pp.create_switch(net, bus=Kildee, element=cable_022, et="l")
sw025 = pp.create_switch(net, bus=Kildee, element=cable_022, et="l")

sw026 = pp.create_switch(net, bus=Linden, element=cable_023, et="l")
sw027 = pp.create_switch(net, bus=Linden, element=cable_023, et="l")
```

IowaState_timeseries.py:

```
import pandapower as pp
import numpy as np
import pandas as pd

import pandapower.control as control
import pandapower.networks as nw
import pandapower.timeseries as timeseries
from pandapower.timeseries.data_sources.frame_data import DFData

import matplotlib.pyplot as plt

net = pp.create_empty_network()

# Ames External Power *MAKE GENERATOR*
#city_of_ames_source = pp.create_bus(net, vn_kv=13.8, name="City_of_Ames_Source")
```

```
# Hawthorn Bay 1-4
hawthorn_sub = pp.create_bus(net, vn_kv=13.8, name="Hawthorn_Sub")
# busses
SWGR_ES120 = pp.create_bus(net, vn_kv=13.8, name="SWGR_ES120")
SWGR_ES123 = pp.create_bus(net, vn_kv=13.8, name="SWGR_ES123")
SWGR_ES126 = pp.create_bus(net, vn_kv=13.8, name="SWGR_ES126")
# lines
cable_205 = pp.create_line(net, from_bus=hawthorn_sub, to_bus=SWGR_ES120,
length_km=0.2, std_type="NAYY 4x50 SE",
                     name="cable_205")
cable_207 = pp.create_line(net, from_bus=hawthorn_sub, to_bus=SWGR_ES123,
length_km=0.1, std_type="NAYY 4x50 SE",
                     name="cable_207")
cable_7 = pp.create_line(net, from_bus=hawthorn_sub, to_bus=SWGR_ES126, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="cable_7")
# Switches
sw_ES120 = pp.create_switch(net, element=cable_205, et="l", bus=SWGR_ES120)


# SWGR_ES120 Bay 1-2
# busses
SWGR_ES121 = pp.create_bus(net, vn_kv=13.8, name="SWGR_ES121")
SWGR_ES116 = pp.create_bus(net, vn_kv=13.8, name="SWGR_ES116")
# lines
cable_17 = pp.create_line(net, from_bus=SWGR_ES120, to_bus=SWGR_ES121,
length_km=0.1, std_type="NAYY 4x50 SE",
                     name="cable_17")
cable_83 = pp.create_line(net, from_bus=SWGR_ES120, to_bus=SWGR_ES116,
length_km=0.4, std_type="NAYY 4x50 SE",
                     name="cable_83")
# switches
sw_ES120_2 = pp.create_switch(net, element=cable_83, et="l", bus=SWGR_ES120)

# SWGR_ES116 Bay 1-4
# busses
SWGR_ES117 = pp.create_bus(net, vn_kv=13.8, name="SWGR_ES117")
SWGR_ES118 = pp.create_bus(net, vn_kv=13.8, name="SWGR_ES118")
```

```
# Static Generator and Load and External Grid
city_of_ames_source = pp.create_sgen(net, bus=SWGR_ES116, p_mw=13.8,
name="City_of_Ames_Source")
source_power_load = pp.create_load(net, bus=SWGR_ES116, p_mw=13.8)
stupid_external_grid = pp.create_ext_grid(net, bus=SWGR_ES116, name="dumbass")

# lines
cable_27 = pp.create_line(net, from_bus=SWGR_ES116, to_bus=city_of_ames_source,
length_km=0.1, std_type="NAYY 4x50 SE",
                  name="cable_27")
cable_29 = pp.create_line(net, from_bus=SWGR_ES116, to_bus=SWGR_ES118,
length_km=0.08, std_type="NAYY 4x50 SE",
                  name="cable_29")
cable_37 = pp.create_line(net, from_bus=SWGR_ES116, to_bus=SWGR_ES117,
length_km=0.3, std_type="NAYY 4x50 SE",
                  name="cable_37")
# switches
sw_ES116 = pp.create_switch(net, element=cable_27, et="l", bus=SWGR_ES116)
sw_ES116_2 = pp.create_switch(net, element=cable_37, et="l", bus=SWGR_ES116)
sw_ES118 = pp.create_switch(net, element=cable_29, et="l", bus=SWGR_ES118)
sw_ES117 = pp.create_switch(net, element=cable_37, et="l", bus=SWGR_ES117)

# Adding in more sections to grid
# NCS sub 4.16
NCS_sub = pp.create_bus(net, vn_kv=4.16, name="NCS_substation")

#Adding in first transformer
pp.create_transformer(net, hv_bus=SWGR_ES117, lv_bus=NCS_sub, name="Hawth_NCS",
std_type="40 MVA 110/20 kV")

# Buses apart of NCS substation
atrb_vista = pp.create_bus(net, vn_kv=4.16, name="ATRB_Vista")
mol_bio = pp.create_bus(net, vn_kv=4.16, name="Molecular_bio")
insectary = pp.create_bus(net, vn_kv=4.16, name="Insectary")
printing = pp.create_bus(net, vn_kv=4.16, name="Printing")
swine_res = pp.create_bus(net, vn_kv=4.16, name="Swine_research")
carver_co = pp.create_bus(net, vn_kv=4.16, name="Carver_colab")


# Lines for NCS sub
```

```
cable_21 = pp.create_line(net, from_bus=SWGR_ES117, to_bus=NCS_sub, length_km=0.4,
std_type="NAYY 4x50 SE",
                    name="Cable21")
cable_12 = pp.create_line(net, from_bus=NCS_sub, to_bus=atrb_vista, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable12")
cable_129 = pp.create_line(net, from_bus=NCS_sub, to_bus=mol_bio, length_km=0.1,
std_type="NAYY 4x50 SE",
                     name="Cable129")
cable_131 = pp.create_line(net, from_bus=NCS_sub, to_bus=insectary, length_km=0.1,
std_type="NAYY 4x50 SE",
                     name="Cable131")
cable_133 = pp.create_line(net, from_bus=NCS_sub, to_bus=printing, length_km=0.1,
std_type="NAYY 4x50 SE",
                     name="Cable133")
cable_134 = pp.create_line(net, from_bus=NCS_sub, to_bus=swine_res, length_km=0.1,
std_type="NAYY 4x50 SE",
                     name="Cable134")
cable_136 = pp.create_line(net, from_bus=NCS_sub, to_bus=carver_co, length_km=0.1,
std_type="NAYY 4x50 SE",
                     name="Cable136")

# Switches for each cable
sw108 = pp.create_switch(net, bus=NCS_sub, element=cable_12, et="l")
sw4 = pp.create_switch(net, bus=atrb_vista, element=cable_12, et="l")

sw109 = pp.create_switch(net, bus=NCS_sub, element=cable_129, et="l")
sw72 = pp.create_switch(net, bus=mol_bio, element=cable_129, et="l")

sw110 = pp.create_switch(net, bus=NCS_sub, element=cable_131, et="l")
sw77 = pp.create_switch(net, bus=insectary, element=cable_131, et="l")

sw111 = pp.create_switch(net, bus=NCS_sub, element=cable_133, et="l")
sw80 = pp.create_switch(net, bus=printing, element=cable_133, et="l")

sw112 = pp.create_switch(net, bus=NCS_sub, element=cable_134, et="l")
sw81 = pp.create_switch(net, bus=swine_res, element=cable_134, et="l")

sw113 = pp.create_switch(net, bus=NCS_sub, element=cable_136, et="l")
sw83 = pp.create_switch(net, bus=carver_co, element=cable_136, et="l")
```

```
#Building up from NCS Sub

bus43 = pp.create_bus(net, vn_kv=4.16, name="Bus43")
bus42 = pp.create_bus(net, vn_kv=4.16, name="Bus42")

#NCS Sub 13.8 East
NCS_subE = pp.create_bus(net, vn_kv=13.8, name="NCS_subE")

#Transformer between substations
pp.create_transformer(net, hv_bus=NCS_subE, lv_bus=bus42,
name="NCS_Subs_Transformers", std_type="40 MVA 110/20 kV")

#NCS Sub 13.8 West
NCS_subW = pp.create_bus(net, vn_kv=13.8, name="NCS_subW")

#lines to Sub Bay 4
cable_76 = pp.create_line(net, from_bus=bus43, to_bus=NCS_sub, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable76")
cable_75 = pp.create_line(net, from_bus=bus43, to_bus=NCS_subE, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable75")
ncs_sub3 = pp.create_line(net, from_bus=NCS_subW, to_bus=NCS_subE, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="ncs sub bay 3")

#switch between NCS sub's

sw_ncs_sub3 = pp.create_switch(net, bus=NCS_subW, element=ncs_sub3, et="l")


#Branching off towards MWL buildings
mapleHall = pp.create_bus(net, vn_kv=4.16, name="mapleHall")

hixson_lied = pp.create_bus(net, vn_kv=4.16, name="hixson-lied")

mwl416 = pp.create_bus(net, vn_kv=4.16, name="MWL4.16")
```

```
cable_72 = pp.create_line(net, from_bus=mwl416, to_bus=NCS_subW, length_km=2.1,
std_type="NAYY 4x50 SE",
                          name="line connecting MWL to NCS stations")

#Yet another transformer for changes in voltages

pp.create_transformer(net, hv_bus=NCS_subW, lv_bus=mwl416, name="Hawth_NCS",
std_type="40 MVA 110/20 kV")

#MWL 4.16 substation lines
cable_200 = pp.create_line(net, from_bus=mwl416, to_bus=mapleHall, length_km=0.1,
std_type="NAYY 4x50 SE",
                          name="Cable200")

cable_203 = pp.create_line(net, from_bus=mwl416, to_bus=hixson_lied, length_km=0.1,
std_type="NAYY 4x50 SE",
                          name="Cable203")

#Switches for cables 200/203
sw175 = pp.create_switch(net, bus=mwl416, element=cable_200, et="l")
sw176 = pp.create_switch(net, bus=mapleHall, element=cable_200, et="l")

sw179 = pp.create_switch(net, bus=mwl416, element=cable_203, et="l")
sw180 = pp.create_switch(net, bus=hixson_lied, element=cable_203, et="l")

#building up from MWL 4.16 substation
mwl138 = pp.create_bus(net, vn_kv=13.8, name="MWL 13.8")

cable_197 = pp.create_line(net, from_bus=mwl138, to_bus=mwl416, length_km=2.1,
std_type="NAYY 4x50 SE",
                          name="Line connecting MWL substations together")

sw172 = pp.create_switch(net, bus=mwl416, element=cable_197, et="l")

#Building over to the commons and alumni center
larchHall = pp.create_bus(net, vn_kv=13.8, name="Larch Hall")

alumniCenter = pp.create_bus(net, vn_kv=13.8, name="Alumni Center")
```

```
cable_208 = pp.create_line(net, from_bus=mwl138, to_bus=larchHall, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable208")

#Switches for cable 208
sw182 = pp.create_switch(net, bus=mwl138, element=cable_208, et="l")
sw184 = pp.create_switch(net, bus=larchHall, element=cable_208, et="l")


#branching down from larch hall
cable_220 = pp.create_line(net, from_bus=larchHall, to_bus=alumniCenter, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable220")

sw190 = pp.create_switch(net, bus=alumniCenter, element=cable_220, et="l")

#EE Substation
EE_sub = pp.create_bus(net, vn_kv=4.16, name="EE_sub")

#transformer for EE substation
pp.create_transformer(net, hv_bus=alumniCenter, lv_bus=EE_sub, name="alumni_EEsub",
std_type="40 MVA 110/20 kV")

#Teleporting to Durham
Durham = pp.create_bus(net, vn_kv=4.16, name="Durham")
Coover = pp.create_bus(net, vn_kv=4.16, name="Coover")
Snedecor = pp.create_bus(net, vn_kv=4.16, name="Snedecor")
Davidson = pp.create_bus(net, vn_kv=4.16, name="Davidson")
DesignCenter = pp.create_bus(net, vn_kv=4.16, name="Design Center")
Sweeney = pp.create_bus(net, vn_kv=4.16, name="Sweeney")

#lines connecting to EE substation
cable_15 = pp.create_line(net, from_bus=alumniCenter, to_bus=EE_sub, length_km=0.7,
std_type="NAYY 4x50 SE",
                    name="Cable15")
#lines for each building
cable_101 = pp.create_line(net, from_bus=EE_sub, to_bus=Durham, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable101")
```

```
cable_104 = pp.create_line(net, from_bus=EE_sub, to_bus=Coover, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable104")
cable_106 = pp.create_line(net, from_bus=EE_sub, to_bus=Snedecor, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable106")
cable_108 = pp.create_line(net, from_bus=EE_sub, to_bus=Davidson, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable108")
cable_110 = pp.create_line(net, from_bus=EE_sub, to_bus=DesignCenter, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable110")
cable_113 = pp.create_line(net, from_bus=EE_sub, to_bus=Sweeney, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable113")

#switches on each cable
sw90 = pp.create_switch(net, bus=Durham, element=cable_101, et="l")
sw48 = pp.create_switch(net, bus=Durham, element=cable_101, et="l")

sw93 = pp.create_switch(net, bus=Coover, element=cable_104, et="l")
sw49 = pp.create_switch(net, bus=Coover, element=cable_104, et="l")

sw94 = pp.create_switch(net, bus=Snedecor, element=cable_106, et="l")
sw57 = pp.create_switch(net, bus=Snedecor, element=cable_106, et="l")

sw95 = pp.create_switch(net, bus=Davidson, element=cable_108, et="l")
sw58 = pp.create_switch(net, bus=Davidson, element=cable_108, et="l")

sw96 = pp.create_switch(net, bus=DesignCenter, element=cable_110, et="l")
sw59 = pp.create_switch(net, bus=DesignCenter, element=cable_110, et="l")

sw97 = pp.create_switch(net, bus=Sweeney, element=cable_113, et="l")
sw60 = pp.create_switch(net, bus=Sweeney, element=cable_113, et="l")

#Building up from EE Sub 4.16
EE_sub13_8 = pp.create_bus(net, vn_kv=13.8, name="EE_sub13_8")

#transformer between EE substations
```

```
pp.create_transformer(net, hv_bus=EE_sub13_8, lv_bus=EE_sub, name="T4-EE Sub",
std_type="40 MVA 110/20 kV")

#Cables between substations
cable_14 = pp.create_line(net, from_bus=EE_sub, to_bus=EE_sub13_8, length_km=0.3,
std_type="NAYY 4x50 SE",
                          name="Cable14")

#switch for cable between EE subs
sw12 = pp.create_switch(net, bus=EE_sub, element=cable_14, et="l")

#Moving to Pearson halls substations
pearsonSub_138 = pp.create_bus(net, vn_kv=13.8, name="PearsonSub_13.8")
pearsonSub_416 = pp.create_bus(net, vn_kv=4.16, name="PearsonSub_4.16")

#transformer between Pearson substations
pp.create_transformer(net, hv_bus=pearsonSub_138, lv_bus=pearsonSub_416,
name="T1-Pearson Sub", std_type="40 MVA 110/20 kV")

#Cables connecting substation to Pearson
cable_242 = pp.create_line(net, from_bus=EE_sub13_8, to_bus=pearsonSub_138,
length_km=0.5, std_type="NAYY 4x50 SE",
                          name="Cable242")

cable_144 = pp.create_line(net, from_bus=pearsonSub_138, to_bus=pearsonSub_416,
length_km=0.3, std_type="NAYY 4x50 SE",
                          name="Cable144")

#Switch to Pearson substaion 4.16
sw75 = pp.create_switch(net, bus=pearsonSub_416, element=cable_144, et="l")

#Buses apart of Pearson Sub 4.16
Carver = pp.create_bus(net, vn_kv=4.16, name="Carver")
Beardshear = pp.create_bus(net, vn_kv=4.16, name="BeardShear")
StudentServices = pp.create_bus(net, vn_kv=4.16, name="Student Services")
Pearson = pp.create_bus(net, vn_kv=4.16, name="Pearson")

#Cables for each bus on Pearson sub
cable_68 = pp.create_line(net, from_bus=pearsonSub_416, to_bus=Carver, length_km=0.15,
std_type="NAYY 4x50 SE",
```

```
                    name="Cable68")
cable_94 = pp.create_line(net, from_bus=pearsonSub_416, to_bus=Beardshear, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable94")
cable_2_1 = pp.create_line(net, from_bus=pearsonSub_416, to_bus=StudentServices,
length_km=0.1, std_type="NAYY 4x50 SE",
                    name="Cable2_1")
cable_40 = pp.create_line(net, from_bus=pearsonSub_416, to_bus=Pearson, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable40")

#Switches for each of the above cables
sw33 = pp.create_switch(net, bus=Carver, element=cable_68, et="l")

sw36 = pp.create_switch(net, bus=Beardshear, element=cable_94, et="l")

sw41 = pp.create_switch(net, bus=StudentServices, element=cable_2_1, et="l")

sw45 = pp.create_switch(net, bus=Pearson, element=cable_40, et="l")

#Student services has 2 buses built off of it
Bus51 = pp.create_bus(net, vn_kv=0.208, name="Bus51")
Bus74 = pp.create_bus(net, vn_kv=0.208, name="Bus74")

#Transformers for each of the sub buses
pp.create_transformer(net, hv_bus=StudentServices, lv_bus=Bus51, name="Lab Mech",
std_type="40 MVA 110/20 kV")

pp.create_transformer(net, hv_bus=StudentServices, lv_bus=Bus74, name="Student 208",
std_type="40 MVA 110/20 kV")

#Cables for sub buses of student services
cable_301 = pp.create_line(net, from_bus=StudentServices, to_bus=Bus51, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable31")
cable_46 = pp.create_line(net, from_bus=StudentServices, to_bus=Bus74, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable46")
#No switches on the above sub buses
```

```
#Building over from Pearson 13.8 to ME Sub station
MEsub_138 = pp.create_bus(net, vn_kv=13.8, name="ME Sub_13.8")

#Connecting cable between sub stations
cable_1 = pp.create_line(net, from_bus=EE_sub13_8, to_bus=MEsub_138, length_km=0.4,
std_type="NAYY 4x50 SE",
                         name="Cable1")

#Building substation and buses inside of ME sub
MEsub_416 = pp.create_bus(net, vn_kv=4.16, name="ME Sub_4.16")
SIC = pp.create_bus(net, vn_kv=4.16, name="SIC")
StudentHealth = pp.create_bus(net, vn_kv=4.16, name="Student Health")
BlackEngr = pp.create_bus(net, vn_kv=4.16, name="Black Engr")
StateGym = pp.create_bus(net, vn_kv=4.16, name="State Gym")

#Transformer for ME substations
pp.create_transformer(net, hv_bus=MEsub_138, lv_bus=MEsub_416, name="T1-ME Sub",
std_type="40 MVA 110/20 kV")

#Cables connecting subs and buses
cable_02 = pp.create_line(net, from_bus=MEsub_138, to_bus=MEsub_416, length_km=0.1,
std_type="NAYY 4x50 SE",
                         name="Cable2")
cable_115 = pp.create_line(net, from_bus=MEsub_416, to_bus=SIC, length_km=0.1,
std_type="NAYY 4x50 SE",
                         name="Cable115")
cable_117 = pp.create_line(net, from_bus=MEsub_416, to_bus=StudentHealth, length_km=0.1,
std_type="NAYY 4x50 SE",
                         name="Cable117")
cable_120 = pp.create_line(net, from_bus=MEsub_416, to_bus=BlackEngr, length_km=0.1,
std_type="NAYY 4x50 SE",
                         name="Cable120")
cable_122 = pp.create_line(net, from_bus=MEsub_416, to_bus=StateGym, length_km=0.1,
std_type="NAYY 4x50 SE",
                         name="Cable122")

#switches apart of ME 4.16 substation
sw19 = pp.create_switch(net, bus=SIC, element=cable_115, et="l")

sw23 = pp.create_switch(net, bus=StudentHealth, element=cable_117, et="l")
```

```
sw24 = pp.create_switch(net, bus=BlackEngr, element=cable_120, et="l")

sw32 = pp.create_switch(net, bus=StateGym, element=cable_122, et="l")

#Cable connecting Pearson 4.16 sub to ME 4.16 sub
cable_18 = pp.create_line(net, from_bus=pearsonSub_416, to_bus=MEsub_416, length_km=0.2,
std_type="NAYY 4x50 SE",
                          name="Cable18")

#Switch for above cable
sw09 = pp.create_switch(net, bus=MEsub_416, element=cable_18, et="l")

#Building over to VAC Clad substations
VacClad_138 = pp.create_bus(net, vn_kv=13.8, name="Vac Clad 13.8 sub")

#Connecting cable from ME sub
cable_24 = pp.create_line(net, from_bus=MEsub_138, to_bus=VacClad_138, length_km=1.2,
std_type="NAYY 4x50 SE",
                          name="Cable24")

#Building Vac clad 4.16 sub and buses
VacClad_416 = pp.create_bus(net, vn_kv=4.16, name="Vac Clad 4.16 sub")
GeneralServices = pp.create_bus(net, vn_kv=4.16, name="General Services")
HeatingPlant = pp.create_bus(net, vn_kv=4.16, name="Heating Plant")
LandscapeArch = pp.create_bus(net, vn_kv=4.16, name="Landscape Arch(B)")
HamilonHall = pp.create_bus(net, vn_kv=4.16, name="Hamilton Hall")
Forker = pp.create_bus(net, vn_kv=4.16, name="Forker")
BesseyVISTA = pp.create_bus(net, vn_kv=4.16, name="Bessey VISTA")

#Feeding into smaller substation
SEFeeder = pp.create_bus(net, vn_kv=4.16, name="SE Feeder")

#Transformer between VAC subs
pp.create_transformer(net, hv_bus=VacClad_138, lv_bus=VacClad_416, name="T1-VAC Sub",
std_type="40 MVA 110/20 kV")

#Cables associated with VAC 4.16 sub
cable_173 = pp.create_line(net, from_bus=VacClad_416, to_bus=GeneralServices,
length_km=0.1, std_type="NAYY 4x50 SE",
```

```
                    name="Cable173")
cable_175 = pp.create_line(net, from_bus=VacClad_416, to_bus=HeatingPlant, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable175")
cable_177 = pp.create_line(net, from_bus=VacClad_416, to_bus=LandscapeArch,
length_km=0.1, std_type="NAYY 4x50 SE",
                    name="Cable177")
cable_181 = pp.create_line(net, from_bus=VacClad_416, to_bus=HamilonHall, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable181")
cable_183 = pp.create_line(net, from_bus=VacClad_416, to_bus=Forker, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable183")
cable_28 = pp.create_line(net, from_bus=VacClad_416, to_bus=BesseyVISTA, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable28")
cable_036 = pp.create_line(net, from_bus=VacClad_416, to_bus=SEFeeder, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable36")

#Switches on cables listed above
sw149 = pp.create_switch(net, bus=GeneralServices, element=cable_173, et="l")

sw153 = pp.create_switch(net, bus=HeatingPlant, element=cable_175, et="l")

sw157 = pp.create_switch(net, bus=LandscapeArch, element=cable_177, et="l")

sw161 = pp.create_switch(net, bus=HamilonHall, element=cable_181, et="l")

sw165 = pp.create_switch(net, bus=Forker, element=cable_183, et="l")

sw016 = pp.create_switch(net, bus=BesseyVISTA, element=cable_28, et="l")

sw37 = pp.create_switch(net, bus=SEFeeder, element=cable_036, et="l")

# SE Feeder buildings
Forker_PMH = pp.create_bus(net, vn_kv=4.16, name="Forker PMH 5")
ES11A = pp.create_bus(net, vn_kv=4.16, name="ES11A")

# Cables for buildings and substation
```

```
cable_39 = pp.create_line(net, from_bus=SEFeeder, to_bus=Forker_PMH, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable39")
cable_41 = pp.create_line(net, from_bus=SEFeeder, to_bus=ES11A, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable41")

# Switches for each cable
sw40 = pp.create_switch(net, bus=SEFeeder, element=cable_39, et="l")

sw39 = pp.create_switch(net, bus=Forker_PMH, element=cable_39, et="l")

sw43 = pp.create_switch(net, bus=SEFeeder, element=cable_41, et="l")

sw44 = pp.create_switch(net, bus=ES11A, element=cable_41, et="l")

# bus inside of ES11A
HeadyHall = pp.create_bus(net, vn_kv=0.208, name="Heady Hall")

# Transformer between ES11A and Heady hall
pp.create_transformer(net, hv_bus=ES11A, lv_bus=HeadyHall, name="Heady Hall
transformer", std_type="40 MVA 110/20 kV")

# cable for Heady hall
cable_45 = pp.create_line(net, from_bus=ES11A, to_bus=HeadyHall, length_km=0.15,
std_type="NAYY 4x50 SE",
                    name="Cable45")

# switch for Heady hall- always open
sw46 = pp.create_switch(net, bus=HeadyHall, element=cable_45, et="l")

# Building out to another substation from VAC 4.16
PPscSwgrSouth = pp.create_bus(net, vn_kv=4.16, name="PP S&C Swgr South")
BesseyVISTA2 = pp.create_bus(net, vn_kv=4.16, name="Bessey VISTA2")
EastAgronomy = pp.create_bus(net, vn_kv=4.16, name="East Agronomy")
LandscapeArchA = pp.create_bus(net, vn_kv=4.16, name="Landscape Arch(A)")
Troxel = pp.create_bus(net, vn_kv=4.16, name="Troxel")
JischkeHonors = pp.create_bus(net, vn_kv=4.16, name="Jischke Honors")
SouthKildee = pp.create_bus(net, vn_kv=4.16, name="South Kildee")
```

```
# cables for each building
cable_259 = pp.create_line(net, from_bus=VacClad_416, to_bus=PPscSwgrSouth,
length_km=1.2, std_type="NAYY 4x50 SE",
                    name="Cable259")
cable_32 = pp.create_line(net, from_bus=PPscSwgrSouth, to_bus=BesseyVISTA2,
length_km=0.1, std_type="NAYY 4x50 SE",
                    name="Cable32")
cable_167 = pp.create_line(net, from_bus=PPscSwgrSouth, to_bus=EastAgronomy,
length_km=0.1, std_type="NAYY 4x50 SE",
                    name="Cable167")
cable_165 = pp.create_line(net, from_bus=PPscSwgrSouth, to_bus=LandscapeArchA,
length_km=0.1, std_type="NAYY 4x50 SE",
                    name="Cable165")
cable_163 = pp.create_line(net, from_bus=PPscSwgrSouth, to_bus=Troxel, length_km=0.1,
std_type="NAYY 4x50 SE",
                    name="Cable163")
cable_169 = pp.create_line(net, from_bus=PPscSwgrSouth, to_bus=JischkeHonors,
length_km=0.1, std_type="NAYY 4x50 SE",
                    name="Cable169")
cable_171 = pp.create_line(net, from_bus=PPscSwgrSouth, to_bus=SouthKildee,
length_km=0.1, std_type="NAYY 4x50 SE",
                    name="Cable171")
#switches for each bus
sw25 = pp.create_switch(net, bus=PPscSwgrSouth, element=cable_32, et="l")
sw31 = pp.create_switch(net, bus=BesseyVISTA2, element=cable_32, et="l")

sw142 = pp.create_switch(net, bus=PPscSwgrSouth, element=cable_167, et="l")
sw143 = pp.create_switch(net, bus=EastAgronomy, element=cable_167, et="l")

sw140 = pp.create_switch(net, bus=PPscSwgrSouth, element=cable_165, et="l")
sw141 = pp.create_switch(net, bus=LandscapeArchA, element=cable_165, et="l")

sw138 = pp.create_switch(net, bus=PPscSwgrSouth, element=cable_163, et="l")
sw139 = pp.create_switch(net, bus=Troxel, element=cable_163, et="l")

sw144 = pp.create_switch(net, bus=PPscSwgrSouth, element=cable_169, et="l")
sw145 = pp.create_switch(net, bus=JischkeHonors, element=cable_169, et="l")

sw146 = pp.create_switch(net, bus=PPscSwgrSouth, element=cable_171, et="l")
sw147 = pp.create_switch(net, bus=SouthKildee, element=cable_171, et="l")
```

```
#Branching off from Vac Clad 13.8 to go to NW sub 13.8
NWsub138 = pp.create_bus(net, vn_kv=13.8, name="NW sub 13.8")

#cable connecting Vac to NW sub
cable_030 = pp.create_line(net, from_bus=VacClad_138, to_bus=NWsub138, length_km=1.7,
std_type="NAYY 4x50 SE",
                    name="Cable030")
#switch between Vac Clad and NW sub
sw29 = pp.create_switch(net, bus=NWsub138, element=cable_030, et="l")

#Branching off to NW sub 4.16
NWsub416 = pp.create_bus(net, vn_kv=4.16, name="NW sub 4.16")
BRLEast = pp.create_bus(net, vn_kv=4.16, name="BRL East")
HachHall = pp.create_bus(net, vn_kv=4.16, name="Hach Hall")
TownEngr = pp.create_bus(net, vn_kv=4.16, name="Town Engr")
Armory = pp.create_bus(net, vn_kv=4.16, name="Armory")

#transformer for NW subs
pp.create_transformer(net, hv_bus=NWsub138, lv_bus=NWsub416, name="NW substation
transformer", std_type="40 MVA 110/20 kV")

#cables for buses above
cable_62 = pp.create_line(net, from_bus=NWsub138, to_bus=NWsub416, length_km=0.5,
std_type="NAYY 4x50 SE",
                    name="Cable62")
cable_125 = pp.create_line(net, from_bus=NWsub416, to_bus=BRLEast, length_km=0.2,
std_type="NAYY 4x50 SE",
                    name="Cable125")
cable_127 = pp.create_line(net, from_bus=NWsub416, to_bus=HachHall, length_km=0.3,
std_type="NAYY 4x50 SE",
                    name="Cable127")
cable_88 = pp.create_line(net, from_bus=NWsub416, to_bus=TownEngr, length_km=0.4,
std_type="NAYY 4x50 SE",
                    name="Cable88")
cable_82 = pp.create_line(net, from_bus=NWsub416, to_bus=Armory, length_km=0.3,
std_type="NAYY 4x50 SE",
                    name="Cable82")

#Switches on each cable
```

```
sw106 = pp.create_switch(net, bus=BRLEast, element=cable_125, et="l")

sw107 = pp.create_switch(net, bus=HachHall, element=cable_127, et="l")

sw66 = pp.create_switch(net, bus=TownEngr, element=cable_88, et="l")

sw68 = pp.create_switch(net, bus=Armory, element=cable_82, et="l")

# building off of NCS sub W
AdminSerBuild = pp.create_bus(net, vn_kv=13.8, name="Administrative Services Building")
AdvanTeachResBuild = pp.create_bus(net, vn_kv=13.8, name="Advanced Teaching and
Research Building")
Sub138 = pp.create_bus(net, vn_kv=13.8, name="Substation for additional Halls")

# lines for each building
cable_121 = pp.create_line(net, from_bus=NCS_subW, to_bus=AdminSerBuild, length_km=2.7,
std_type="NAYY 4x50 SE",
                    name="Cable121")
cable_57 = pp.create_line(net, from_bus=AdminSerBuild, to_bus=AdvanTeachResBuild,
length_km=0.7,
                    std_type="NAYY 4x50 SE",
                    name="Cable57")
cable_58 = pp.create_line(net, from_bus=AdvanTeachResBuild, to_bus=Sub138,
length_km=1.3, std_type="NAYY 4x50 SE",
                    name="Cable58")
# switches on each cable
sw79 = pp.create_switch(net, bus=AdminSerBuild, element=cable_121, et="l")

sw368 = pp.create_switch(net, bus=AdvanTeachResBuild, element=cable_57, et="l")

sw268 = pp.create_switch(net, bus=Sub138, element=cable_58, et="l")

# Buildings apart of Substation 13.8
Sub416 = pp.create_bus(net, vn_kv=4.16, name="Sub 4.16")
Atanasoff = pp.create_bus(net, vn_kv=4.16, name="Atanasoff Hall")
Beyer = pp.create_bus(net, vn_kv=4.16, name="Beyer Hall")
BartonRes = pp.create_bus(net, vn_kv=4.16, name="Barton Residence Hall")
Catt = pp.create_bus(net, vn_kv=4.16, name="Catt Hall")
Curtis = pp.create_bus(net, vn_kv=4.16, name="Curtis Hall")
```

```
# transformer for Subs above
pp.create_transformer(net, hv_bus=Sub138, lv_bus=Sub416, name="13.8 - 4.16kv sub
transformer",
                      std_type="40 MVA 110/20 kV")

# lines for each building
cable_01 = pp.create_line(net, from_bus=Sub138, to_bus=Sub416, length_km=0.7,
std_type="NAYY 4x50 SE",
                          name="Cable01")
cable_010 = pp.create_line(net, from_bus=Sub416, to_bus=Atanasoff, length_km=0.5,
std_type="NAYY 4x50 SE",
                           name="Cable010")
cable_011 = pp.create_line(net, from_bus=Sub416, to_bus=Beyer, length_km=0.5,
std_type="NAYY 4x50 SE",
                           name="Cable011")
cable_012 = pp.create_line(net, from_bus=Sub416, to_bus=BartonRes, length_km=0.5,
std_type="NAYY 4x50 SE",
                           name="Cable012")
cable_013 = pp.create_line(net, from_bus=Sub416, to_bus=Catt, length_km=0.5,
std_type="NAYY 4x50 SE",
                           name="Cable013")
cable_014 = pp.create_line(net, from_bus=Sub416, to_bus=Curtis, length_km=0.5,
std_type="NAYY 4x50 SE",
                           name="Cable014")

# Switches for each cable
sw01 = pp.create_switch(net, bus=Sub416, element=cable_01, et="l")

sw010 = pp.create_switch(net, bus=Atanasoff, element=cable_010, et="l")
sw73 = pp.create_switch(net, bus=Atanasoff, element=cable_010, et="l")

sw011 = pp.create_switch(net, bus=Beyer, element=cable_011, et="l")
sw76 = pp.create_switch(net, bus=Beyer, element=cable_011, et="l")

sw012 = pp.create_switch(net, bus=BartonRes, element=cable_012, et="l")
sw74 = pp.create_switch(net, bus=BartonRes, element=cable_012, et="l")

sw013 = pp.create_switch(net, bus=Catt, element=cable_013, et="l")
sw71 = pp.create_switch(net, bus=Catt, element=cable_013, et="l")
```

```
sw014 = pp.create_switch(net, bus=Curtis, element=cable_014, et="l")
sw74 = pp.create_switch(net, bus=Curtis, element=cable_014, et="l")

# Building off from Curtis
Ross = pp.create_bus(net, vn_kv=4.16, name="Ross Hall")
ForestryGH = pp.create_bus(net, vn_kv=4.16, name="Forestry Greenhouse")

# lines for connecting buildings
cable_015 = pp.create_line(net, from_bus=Curtis, to_bus=Ross, length_km=0.3,
std_type="NAYY 4x50 SE",
                    name="Cable015")
cable_016 = pp.create_line(net, from_bus=Ross, to_bus=ForestryGH, length_km=0.7,
std_type="NAYY 4x50 SE",
                    name="Cable016")

# switches for each cable
sw015 = pp.create_switch(net, bus=Ross, element=cable_015, et="l")

sw016 = pp.create_switch(net, bus=ForestryGH, element=cable_016, et="l")

# Substation connecting to Forestry Greenhouse
FoodScienceSub = pp.create_bus(net, vn_kv=4.16, name="Food Sciences sub")
Martin = pp.create_bus(net, vn_kv=4.16, name="Martin Hall")
Palmer = pp.create_bus(net, vn_kv=4.16, name="Palmer Building")
Morrill = pp.create_bus(net, vn_kv=4.16, name="Morrill Hall")
MacKay = pp.create_bus(net, vn_kv=4.16, name="MacKay Hall")
Kildee = pp.create_bus(net, vn_kv=4.16, name="Kildee Hall")
Linden = pp.create_bus(net, vn_kv=4.16, name="Linden Hall")

# Lines connecting from Food Science Substation
cable_017 = pp.create_line(net, from_bus=ForestryGH, to_bus=FoodScienceSub,
length_km=0.7, std_type="NAYY 4x50 SE",
                    name="Cable017")
cable_018 = pp.create_line(net, from_bus=FoodScienceSub, to_bus=Martin, length_km=0.5,
std_type="NAYY 4x50 SE",
                    name="Cable018")
cable_019 = pp.create_line(net, from_bus=FoodScienceSub, to_bus=Palmer, length_km=0.5,
std_type="NAYY 4x50 SE",
                    name="Cable019")
```

```
cable_020 = pp.create_line(net, from_bus=FoodScienceSub, to_bus=Morrill, length_km=0.2,
std_type="NAYY 4x50 SE",
                    name="Cable020")
cable_021 = pp.create_line(net, from_bus=FoodScienceSub, to_bus=MacKay, length_km=0.5,
std_type="NAYY 4x50 SE",
                    name="Cable021")
cable_022 = pp.create_line(net, from_bus=FoodScienceSub, to_bus=Kildee, length_km=0.9,
std_type="NAYY 4x50 SE",
                    name="Cable022")
cable_023 = pp.create_line(net, from_bus=FoodScienceSub, to_bus=Linden, length_km=0.7,
std_type="NAYY 4x50 SE",
                    name="Cable023")

# Switches on each cable
sw015 = pp.create_switch(net, bus=FoodScienceSub, element=cable_017, et="l")

sw016 = pp.create_switch(net, bus=Martin, element=cable_018, et="l")
sw017 = pp.create_switch(net, bus=Martin, element=cable_018, et="l")

sw018 = pp.create_switch(net, bus=Palmer, element=cable_019, et="l")
sw019 = pp.create_switch(net, bus=Palmer, element=cable_019, et="l")

sw020 = pp.create_switch(net, bus=Morrill, element=cable_020, et="l")
sw021 = pp.create_switch(net, bus=Morrill, element=cable_020, et="l")

sw022 = pp.create_switch(net, bus=MacKay, element=cable_021, et="l")
sw023 = pp.create_switch(net, bus=MacKay, element=cable_021, et="l")

sw024 = pp.create_switch(net, bus=Kildee, element=cable_022, et="l")
sw025 = pp.create_switch(net, bus=Kildee, element=cable_022, et="l")

sw026 = pp.create_switch(net, bus=Linden, element=cable_023, et="l")
sw027 = pp.create_switch(net, bus=Linden, element=cable_023, et="l")

# number of time switches
nts = 95

# this might be broken right now but this is supposed to be the generator range for time series
(hopefully)
```

```
df = pd.DataFrame(np.random.normal(0.5, 0.1, size=(nts, len(net.sgen.index))),
index=list(range(nts)), columns=net.sgen.index) * net.sgen.p_mw.values

# make datasource
ds = DFData(df)

const_sgen = control.ConstControl(net, element='sgen', element_index=net.sgen.index,
variable='p_mw', data_source=ds, profile_name=net.sgen.index)

# same thing here, just load
df = pd.DataFrame(np.random.normal(0.5, 0.1, size=(nts, len(net.sgen.index))),
index=list(range(nts)), columns=net.load.index) * net.load.p_mw.values

ds = DFData(df)

const_load = control.ConstControl(net, element='load', element_index=net.load.index,
variable='p_mw', data_source=ds, profile_name=net.load.index)


# ow = timeseries.OutputWriter(net, output_path="./", output_file_type=".xlsx")

timeseries.run_timeseries(net)

df.loc[:, net.sgen.index].plot()
plt.show()
# THIS NEEDS TO BE CHANGED FOR COMPUTERS INDIVIDUALLY (SAME AS IN
HawthornShim_attacktest)
pp.to_pickle(net, filename=r'./IowaState_timeseries.p')
```