NREL's Cyber-Energy Emulation Platform for Research and System Visualization

Adarsh Hasandka, Joshua Rivera, and Joshua Van Natta

National Renewable Energy Laboratory

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Contract No. DE-AC36-08GO28308

Technical Report
NREL/TP-5R00-74142
May 2020

NREL's Cyber-Energy Emulation Platform for Research and System Visualization

Adarsh Hasandka, Joshua Rivera, and Joshua Van Natta

National Renewable Energy Laboratory

Suggested Citation
Hasandka, Adarsh, Joshua Rivera, and Joshua Van Natta. 2020. *NREL's Cyber-Energy Emulation Platform for Research and System Visualization*. Golden, CO: National Renewable Energy Laboratory. NREL/TP-5R00-74142.
https://www.nrel.gov/docs/fy20osti/74142.pdf.

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Contract No. DE-AC36-08GO28308

Technical Report
NREL/TP-5R00-74142
May 2020

National Renewable Energy Laboratory
15013 Denver West Parkway
Golden, CO 80401
303-275-3000 • www.nrel.gov

NOTICE

This work was authored by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. This work was supported by the Laboratory Directed Research and Development (LDRD) Program at NREL. The views expressed herein do not necessarily represent the views of the DOE or the U.S. Government.

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

U.S. Department of Energy (DOE) reports produced after 1991 and a growing number of pre-1991 documents are available free via www.OSTI.gov.

Cover Photos by Dennis Schroeder: (clockwise, left to right) NREL 51934, NREL 45897, NREL 42160, NREL 45891, NREL 48097, NREL 46526.

NREL prints on paper that contains recycled content

List of Acronyms

CEE	Cyber-Energy Emulation
IDS	intrusion detection system
KVM	kernel-based virtual machine
NREL	National Renewable Energy Laboratory
OpenDSS	Open Distribution System Simulator

Executive Summary

This paper presents the National Renewable Energy Laboratory's Cyber-Energy Emulation (CEE) Platform. This is a novel emulation platform for achieving real-time visualization of large-scale environments involving cyber-physical devices. It allows for the environment to include real, physical hardware along with emulated devices communicating with each other as part of the same system. The CEE Platform is also capable of streaming, collecting, storing, transporting, and visualizing all data within the emulated environment. This capability enables high-fidelity visualization of events to be performed in real time as well forensic analysis based on historical data. The paper presents the design of the CEE Platform as well as several potential use cases and applications. It also highlights the potential for this tool to be used for broad research agendas and for cyber-physical system education.

Table of Contents

List of Figures

Introduction

One of the most challenging problems for cybersecurity researchers is making sense of the immense amount of data captured during an experiment and knowing where to begin looking for abnormalities or areas for improvement and optimization. During only a few minutes of operation, a co-simulation engine can generate gigabytes of system logs, packet captures, and power values for a system with approximately 100 devices. This means that after the data have been captured, they must be aggregated, compared with the results of any leveraged intrusion detection systems (IDSs) against available system logs, and cross-referenced with the logged power values at any given time stamp. Considering the amount of data, this process, even for a relatively short run time, can take days of concerted effort to produce readable results.

The ability to combine currently available system and network virtualization, power simulation, real-time data streaming, and power hardware-in-the-loop technologies to build a comprehensive, real-time visualization tool is a desirable capability for many organizations, especially those related to research and education. For that reason, the National Renewable Energy Laboratory (NREL) has directed resources into building the Cyber-Energy Emulation (CEE) Platform. One goal of this platform is understanding how such a capability could be used to produce detailed visualizations of complex energy systems that enable researchers to observe and analyze the electric grid of the future. This capability requires multiple varied skill sets. In particular, the development team at NREL has noted the importance of (1) personnel with skills in virtualization and networking, (2) scripting and operational development, and (3) full-stack development for visualization. Our team found that these three capabilities are the minimum skills necessary to put such a platform together.

A main objective of the CEE Platform is to build a virtual environment in which users can view a simulated cyber event from multiple perspectives. The ability to discuss unfolding events through this virtual model will provide users significant insights into cybersecurity for energy systems. These advances in creating a virtual environment with simultaneous observation and control led to the team considering the development of a web-based application and front-end interface for this platform.

The goal of this research effort is to develop a 3D platform that would, through a single web interface, provide users with a vantage from which to observe cybersecurity experiments unfold, quickly highlight anomalies and attacks, and interact with the system for in-depth analysis. The research team can quickly set up an environment, from small-scale microgrids with a few subnets to city-size grids with thousands of networks, either on local hardware or remotely on a server. The team uses this environment to run experiments or execute scenarios—for example, to help determine an optimal, highly resilient security posture.

Previous research highlights the benefits of the insights and perspectives provided by visualizations of complex systems in many different fields and in individual and group settings (Bresciani and Eppler 2009). Specifically, in the field of networking and information security, there is significant advantage in having a way to visually analyze different events as they impact different cyber-physical layers parallel to the network (Fisk et al. 2003). There have been multiple previous attempts to build a tool to visualize cyber-physical events; however, they have been for a specific use case or for visualizing individual events (Bosch et al. 2000; Huffer et al.

2017; Rossey et al. 2002). This application aims to fill this gap in capability by providing a platform for users to build virtual worlds by incorporating multiple cyber-physical systems interacting with an emulated network in real time. Users could then interact with this system to initiate and observe events, as well as extreme system states, without having to compromise and risk damage to real hardware. The user has the option to link in real devices to the emulated network and add them into the experimental environment. For example, Figure 1 depicts a live reconnaissance probe inserted into an experiment. This allows for the evaluation of physical hardware in many different scenarios without requiring the build-out of large, physical test beds. The CEE Platform was built by integrating many different tools that individually provide the following core features:

- Cyber emulation layer
- Power co-simulation framework
- Data collection and storage
- Data parsing and transport
- Real-time visualization
- Binding and control scripts.

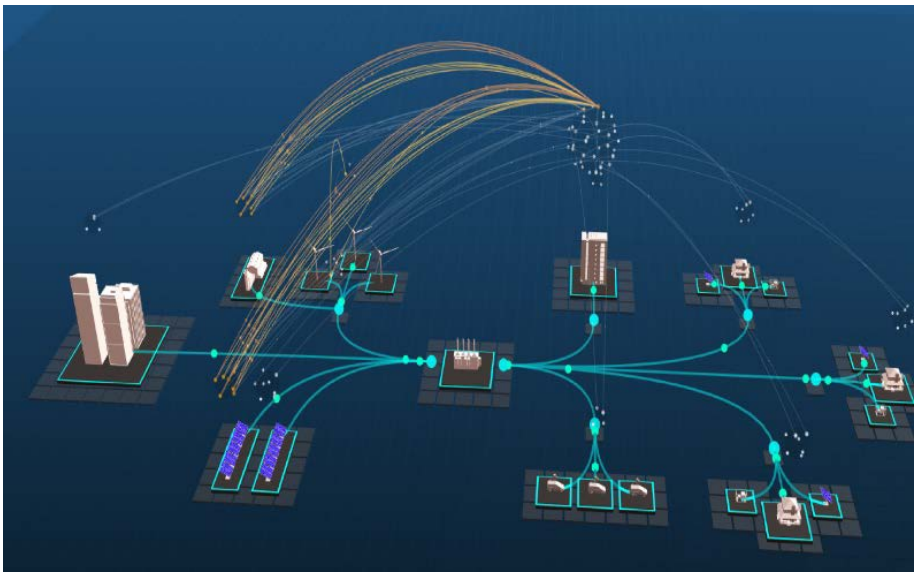

Figure 1. Screenshot of the CEE Platform in action depicting a live reconnaissance probe using the network-mapping application, Nmap, launched by a remote attacker within the environment

1 Core Components

1.1 Emulation

Cyber emulation using virtualization has proved to be valuable for research, development, testing, validation, and modeling. For this reason, Linux kernel-based virtualization, in parallel with Open vSwitch, has been recognized as the virtualization technology that provides all the necessary elements, along with open-source licensing, for creating a virtual environment that provides a medium for linking cyber-physical devices.

1.1.1 minimega

minimega[1] is the main virtualization engine used to enable the visualization of the CEE Platform. minimega provides all the essential functions to create as well as autonomously deploy virtual systems, containers, switches, and more. minimega provides functions for the autonomous provisioning and deployment of systems and network infrastructure, enabling the system to quickly generate and disassemble large environments on demand. Additional functions include command and control, device monitoring, and traffic generation.

Written in Golang, minimega binds Quick Emulator, a kernel-based virtual machine (KVM), and Open vSwitch into a single entity that provides an autonomous functionality to generate virtual machines on local hardware while creating virtual local area networks that provide network segmentation (minimega 2019). The robust functions built into the minimega framework are then essential for controlling and monitoring the spawned virtual machines. The binding of KVM and Open vSwitch through minimega is key to emulating systems and networks to support large-scale comprehensive analysis and to provide autonomous provisioning and deployment of large-scale system and network infrastructure. minimega also creates a local server, called a miniweb, that provides monitoring and control mechanisms through either a web interface or an external application programming interface. Figure 2 depicts the output of this client.

[1] minimega is an open source, distributed, virtual machine management tool developed by Sandia National Laboratories. It is accessible at http://minimega.org/.

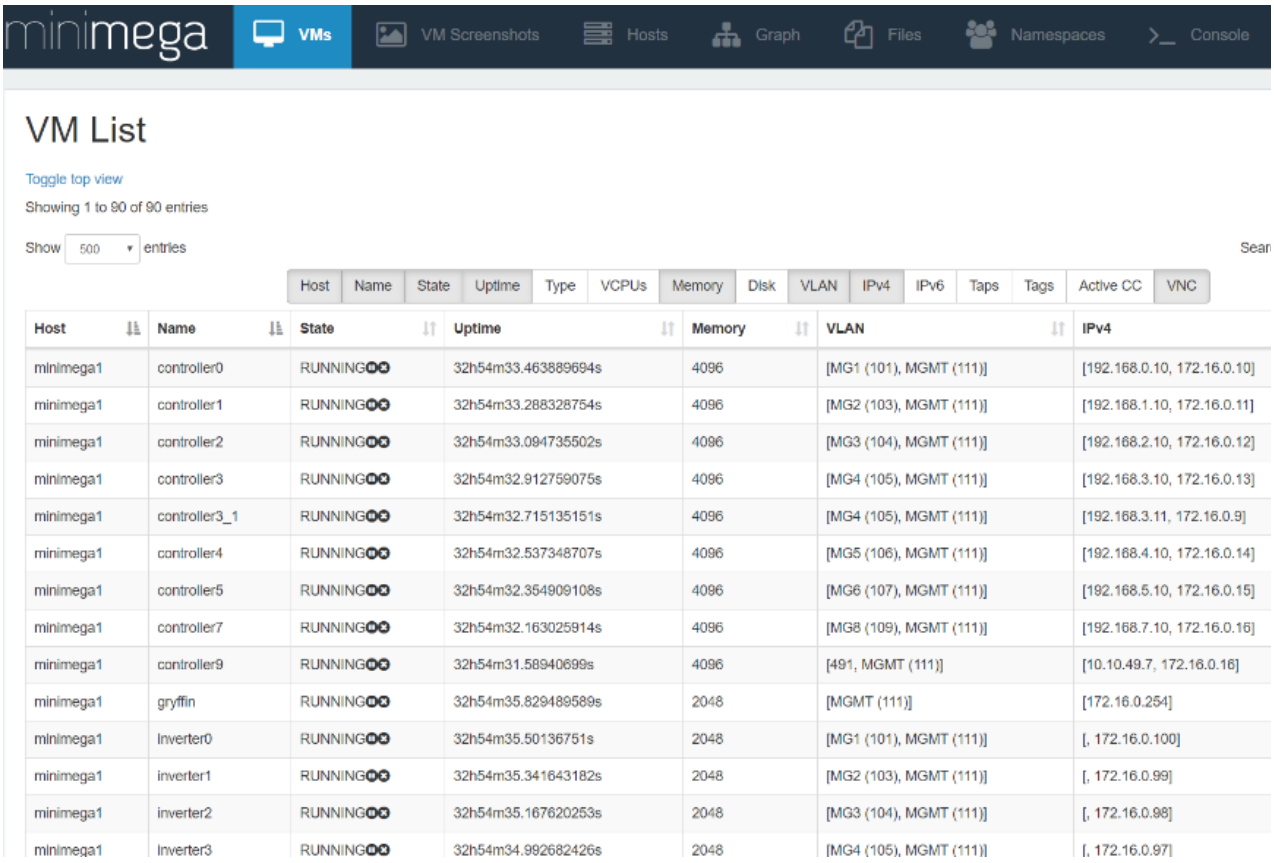

Figure 2. The minimega console listing output data on all emulated virtual machines

1.1.2 Kernel-Based Virtual Machine

Through the application of minimega, a KVM is used as the core system virtualization capability. A KVM is a Linux subsystem that leverages the available virtualization extensions inbuilt in any modern Linux kernel to allow for the operating system to provide a virtual machine monitor or hypervisor capability. Using KVM, one can create and run multiple virtual machines on the same Linux hardware (Kivity et al. 2007). These newly created virtual machines appear entirely as normal Linux processes and integrate seamlessly with the rest of the system while sharing all available resources according to allocations. Figure 3 shows the live screenshots of all emulated virtual machines. This interface could be used to access and pass user input into the virtual machines. In fact, this interface is leveraged in this manner to provide the virtual machine live view and control features of the CEE Platform. This web page is accessed via the miniweb interface available in minimega.

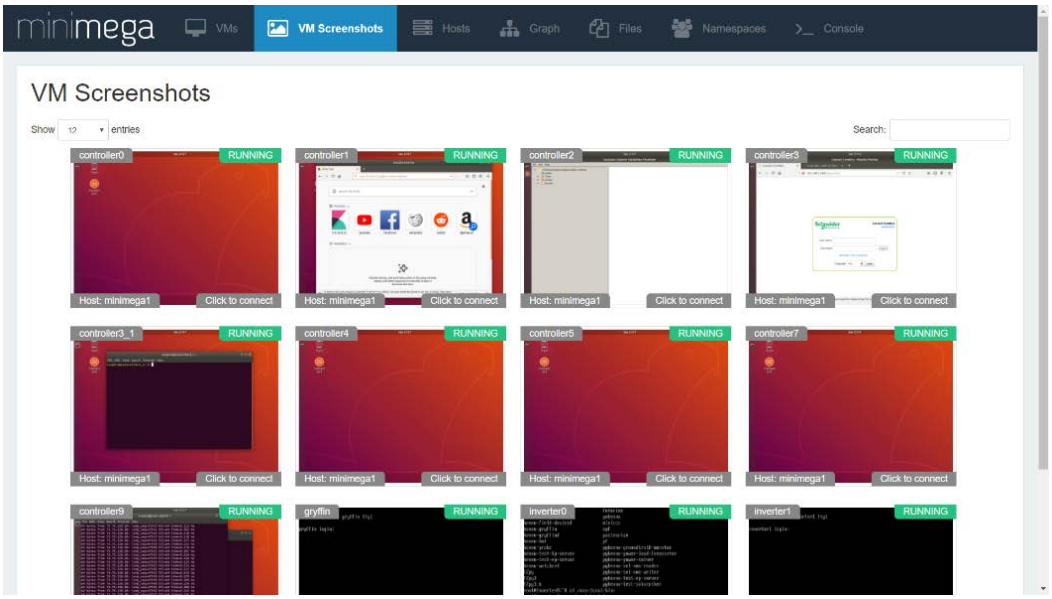

Figure 3. Live screenshot displays of KVMs for emulated devices

1.1.3 Open vSwitch

Through the application of minimega, Open vSwitch is used as the core switching capability. Open vSwitch enables the application of security, monitoring, quality of service, and automated control, which facilitate substantial network automation (Open vSwitch 2016). Open vSwitch allows for a series of virtual local area networks to be created as well as for the mirroring of traffic across each virtual local area networks to be achieved. The collection, transport, storage, and analysis of network traffic across the experimental network enables visibility into each network for analysis. Figure 4 shows the 2D network graph of the emulated network spun up by minimega using Open vSwitch. This web page is accessed via the miniweb interface available in minimega.

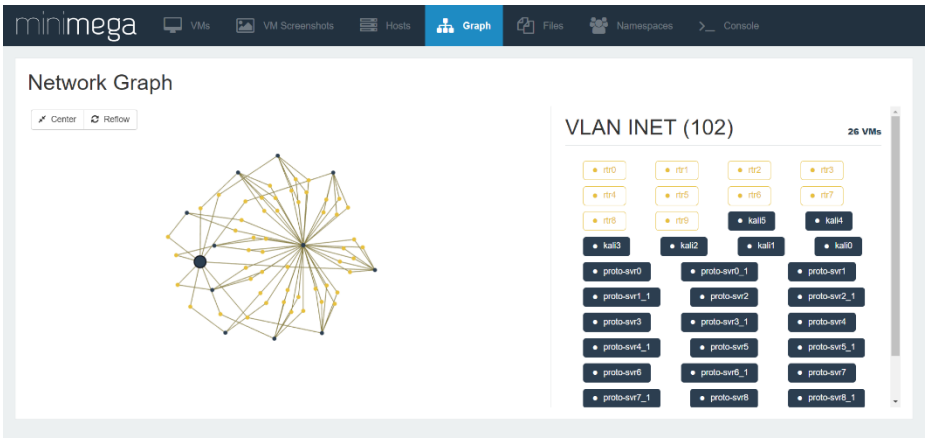

Figure 4. 2D graph of Open vSwitch emulated network in minimega

1.2 Co-simulation

1.2.1 SCEPTRE

SCEPTRE is a co-simulation framework developed by Sandia National Laboratories. "SCEPTRE is an application that uses an underlying network emulation and analytics platform (Emulytics) to model, simulate, emulate, test, and validate control system security and process simulations" (Jacobs and Johnson 2018). Emulytics uses minimega to virtualize different simulation tools to handle different layers in the cyber-physical system. SCEPTRE builds on this virtualization to integrate additional simulation tools by spinning up a tightly integrated image, which implements the needed simulation tool. As an example, the Open Distribution System Simulator (OpenDSS) is currently used to provide static power flow data; however, SCEPTRE also supports PowerWorld and PYPOWER. Looking beyond SCEPTRE, by focusing efforts on binding co-simulation engines, such as the Hierarchical Engine for Large-scale Infrastructure Co-Simulation (Palmintier et al. 2017), this platform is also capable of integrating more detailed simulators, such as GridLAB-D (Chassin, Schneider, and Gerkensmeyer 2008) or OPAL-RT's ePHASORSIM (Jalili-Marandi et al. 2013). Figure 5 shows an example of the live view of detailed power flow data in the power properties window, which is available to the user as a result of integrating this software.

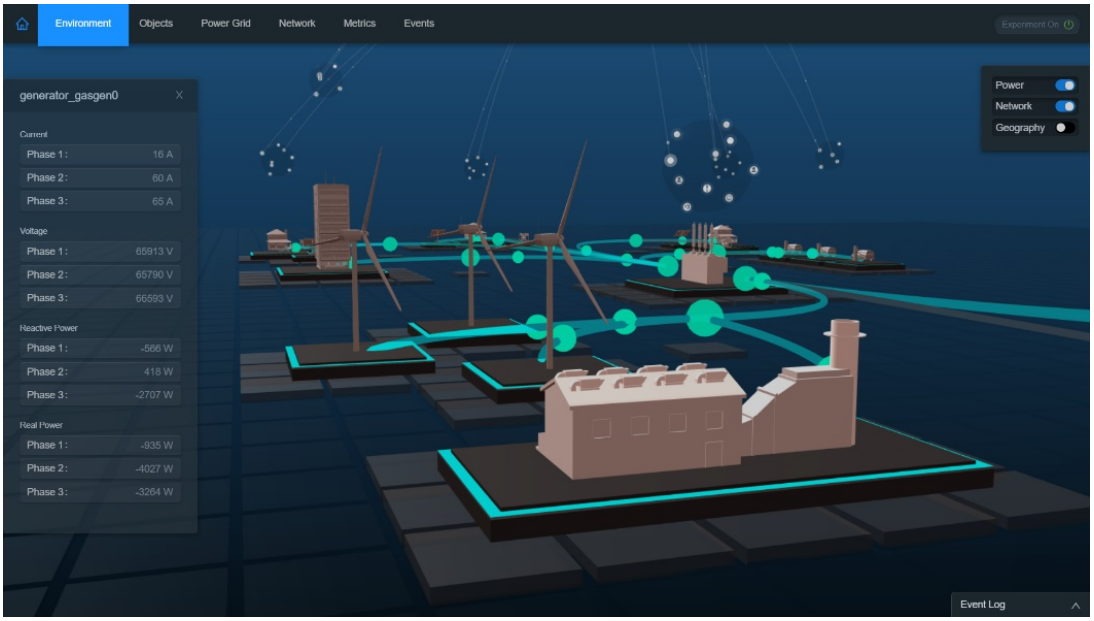

Figure 5. Screenshot of detailed power system data from OpenDSS via SCEPTRE

1.2.2 Open Distribution System Simulator

OpenDSS is an electric power system simulation tool designed mainly to simulate power distribution systems at utility scale. It is primarily used for sinusoidal-steady-state analyses of these types of systems (Ramachandran 2011). It also provides support for smart grid and renewable energy power system device models for research. Most importantly, these attributes meet the requirements for a power systems layer simulator needed in the CEE Platform. As support for additional power simulators develops, the reliance on OpenDSS solely for power systems data can be expected to decrease. The scenario depicted in this paper includes many

different types of generation sources, such as wind turbines, natural gas power plants, and utility-scale solar power plants, as shown in Figure 6. Similarly, Figure 7 shows the commercial building load, electric vehicle chargers, and smart homes, including local solar generation and diesel backup, to support a local residential load, all included in this scenario.

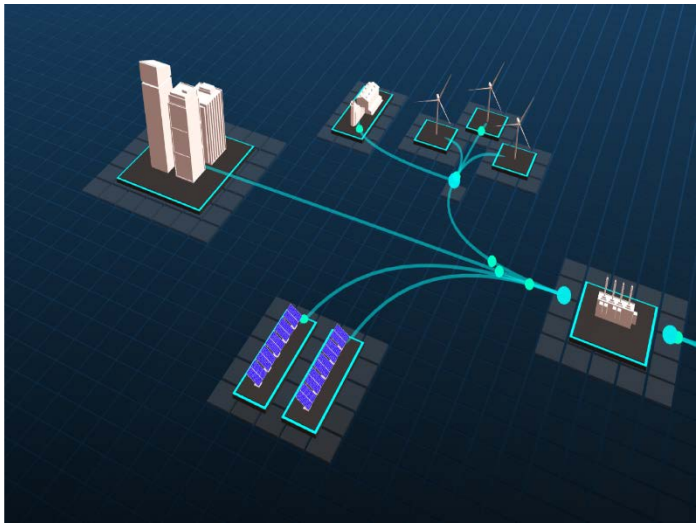

Figure 6. Major generation elements in the power layer, including wind turbines, natural gas power plants, and utility-scale solar power plants

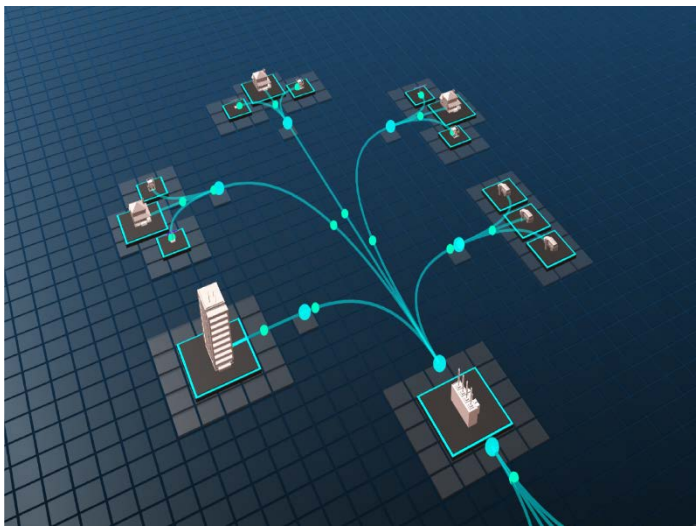

Figure 7. Distribution-side elements in the power layer, including commercial building load, electric vehicle chargers, and smart homes

1.3 Processing and Storage

1.3.1 Elasticsearch

Elasticsearch, a search and analytics engine, provides the data structure and querying functions necessary to ensure that data can be pulled into the visualization front end. Elasticsearch is sent data via the built-in HyperText Transfer Protocol interface. Data can be indexed into the database either individually or in bulk, depending on timing or other constraints. Upon running

the scenario depicted in this paper for a 24-hour period, with the current level of fidelity of data and the currently implemented power and network and system-log layers, 0.1 TB of storage space was used. This disk space size of data does not account for any redundancy requirements. This approximation of data storage will need to be reconsidered if any additional layers or data streams are enabled in the future or if the intensity of the emulated traffic is increased.

1.3.2 Event Manager

The event manager script, named *EventMgr* and written in Go (Go 2019), is designed to be the centralized data handler and experimental controller for the system. This script leverages the strong multithreading available in the Go language (Tang 2019) to start individual goroutines to handle concurrent processes, and it uses channels to pipe data between these processes, as shown in Figure 8. The event manager runs parallel processes to collect and parse data from each individual data stream, to send out buffered updates of data to the database, to send out separately buffered updates of data to the visualization server, and, finally, to handle experimental control signals from the user. Because the data streams are in real time, the volume and density of data prove to be too intense to be able to send the stream directly to the Elasticsearch database. To better handle this data transfer asynchronously, a separate thread is used as a buffer that collects all the individual database updates and periodically flushes the buffer by writing the collected data in bulk to the database. A similar issue occurs with the visualization server. Because the real-time visualization application can handle higher throughput data streams using ZeroMQ than the Elasticsearch database can using HyperText Transfer Protocol, a similar thread is used as a buffer that collects all the individual visual updates and periodically flushes the buffer in faster intervals by publishing bulk updates on a ZeroMQ PUB socket. This is done for each individual data stream; however, the update intervals and selected streams are adjusted depending on the expected data intensity in each stream.

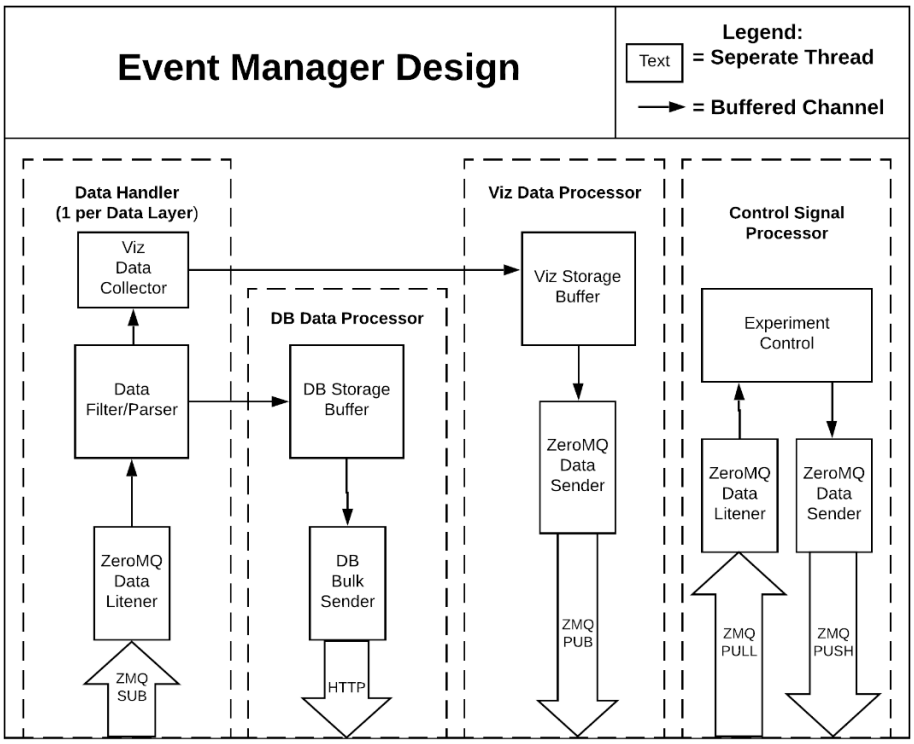

Figure 8. Event manager script with individual goroutines to manage concurrent processes and channels to pipe data between processes

1.4 Data Parsing and Transport

1.4.1 Go Parser Scripts

Several scripts have been written in Go that collect, parse, and send out data in an ingestible format for the event manager. The parser scripts run on the cluster node, which runs the experimental environment. Depending on which data the parser is collecting, a different interface could be used for data collection. Once the parser script has collected a new data point, it uses a ZeroMQ PUB socket to send the data out to the event manager and, via the PUB-SUB connection, to any other client that is subscribed to this real-time data stream.

1.4.2 ZeroMQ

ZeroMQ is a message-queuing system that allows devices to pass messages between each other or even between threads in a single process. The ZeroMQ project includes several built-in bindings for many commonly used languages, such as C++ and Python. Specifically, this project makes use of the Go language bindings. ZeroMQ is leveraged to send low-latency data streams to different components of the CEE Platform. It makes use of the PUB-SUB and PUSH-PULL socket models in ZeroMQ to achieve both data streams and control interfaces, respectively. This software allows for binding multiple different systems with each other in real time. ZeroMQ is also heavily leveraged to enable co-simulation within SCEPTRE (Jacobs and Johnson 2018).

1.5 Visualization

1.5.1 React+ThreeJS

The visual front-end web application was developed using the ThreeJS JavaScript library and React 360 framework. This combination allows for the development of highly customizable, highly accessible, and high-performance 3D visualizations.

1.6 Binding and Control

1.6.1 Commander Script

The commander script runs on the head node and allows the CEE Platform to control its own operations as well as enable user control. The script listens for commands on a ZeroMQ PULL socket and individually handles each command received in a sequential manner. After fully executing the command, the script sends a response to the original source of the request indicating success or failure. The design of this script is very similar to the design of the control signal processor subprocess within the event manager design shown in Figure 9.

2 System Architecture

The CEE system architecture shown in Figure 9 depicts a high-level perspective of how each major system component links together. Recognized as complex, this system binds all core components in a manner that enables real-time visualization of network traffic, security alerts, and power system state and allows historical data sets of network flow, packet capture, system logs, and power system state to be queried for analysis.

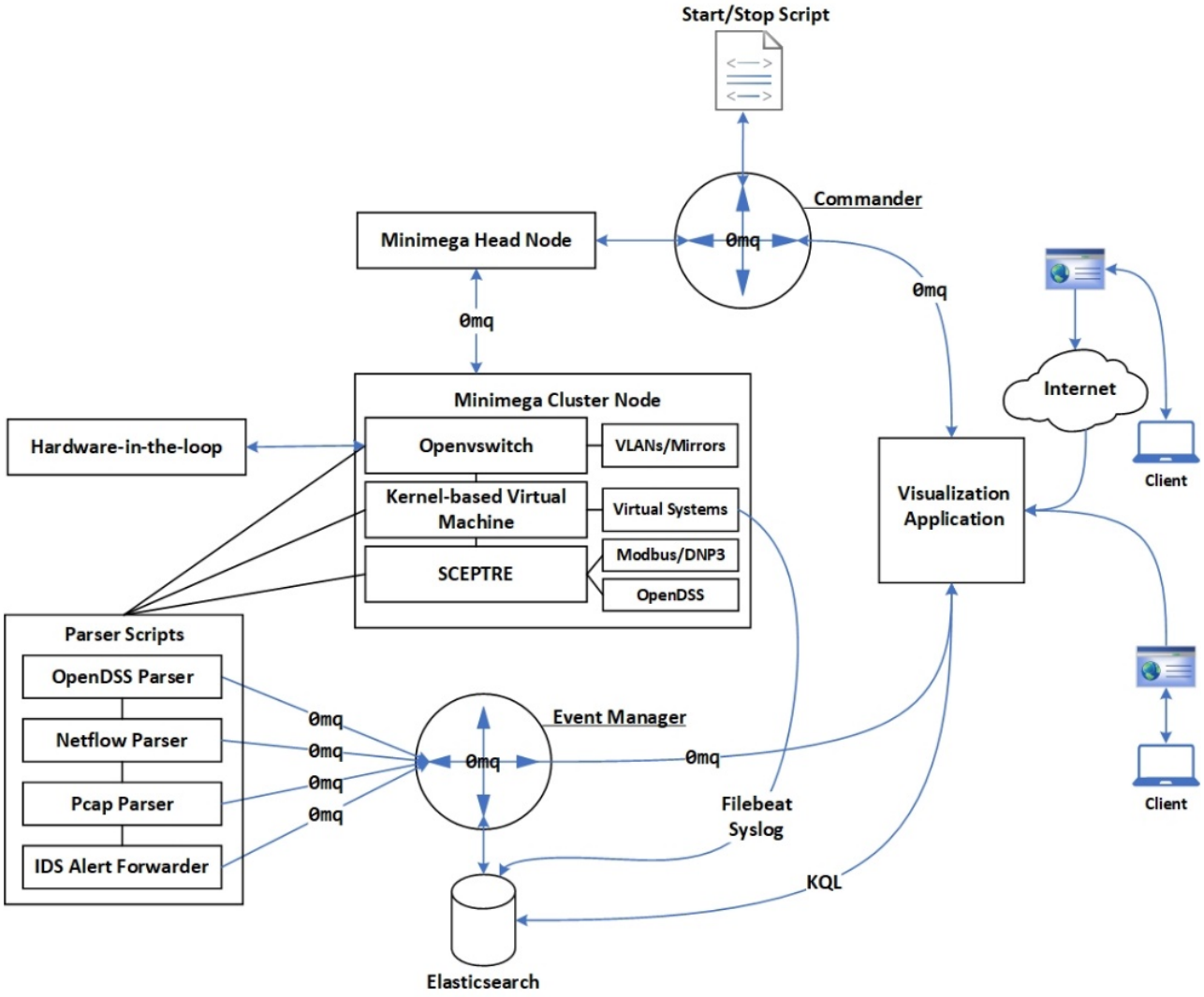

Figure 9. Major components of CEE Platform system architecture

2.1 Multilayered 3D Environment

To support a multitude of potential research objectives, the CEE Platform needs to logically visualize the subject electric grid, including a multitude of electric devices; the geographic context for reference; and the communications network with its full diversity of networked components. Additionally, the platform needs to include exploration and evaluation of pathways toward resilience and potential financial impacts. This variety of sources and behavior, coupled

with access to best-in-class tool sets, would provide researchers with a depth of awareness currently unavailable in any commercially available technology.

This wide variety of data sets presents an immediate problem, however: how to ensure that everything remains clear and understandable to the user. Including too many elements on the screen at once could become visually complicated and could fail to convey useful information. Including too few elements could cause the view to become too abstract to be usable. Each element needs to communicate clearly what it represents with as little superfluous data as possible.

The solution to this problem is to focus narrowly on one data set and ensure that the correlated elements are easy to identify. For example, power and network elements should be distinct both from each other and from the underlying geography. Then, once the research team is satisfied with an isolated view, the views are overlaid, and each layer is ensured to be distinct and not confuse the overall visual. If a view cannot be included in the combined context, or if the data set does not appear to have a clear relationship to X and Y coordinates, this instance is branched off as a distinct 3D render, as shown in Figure 10.

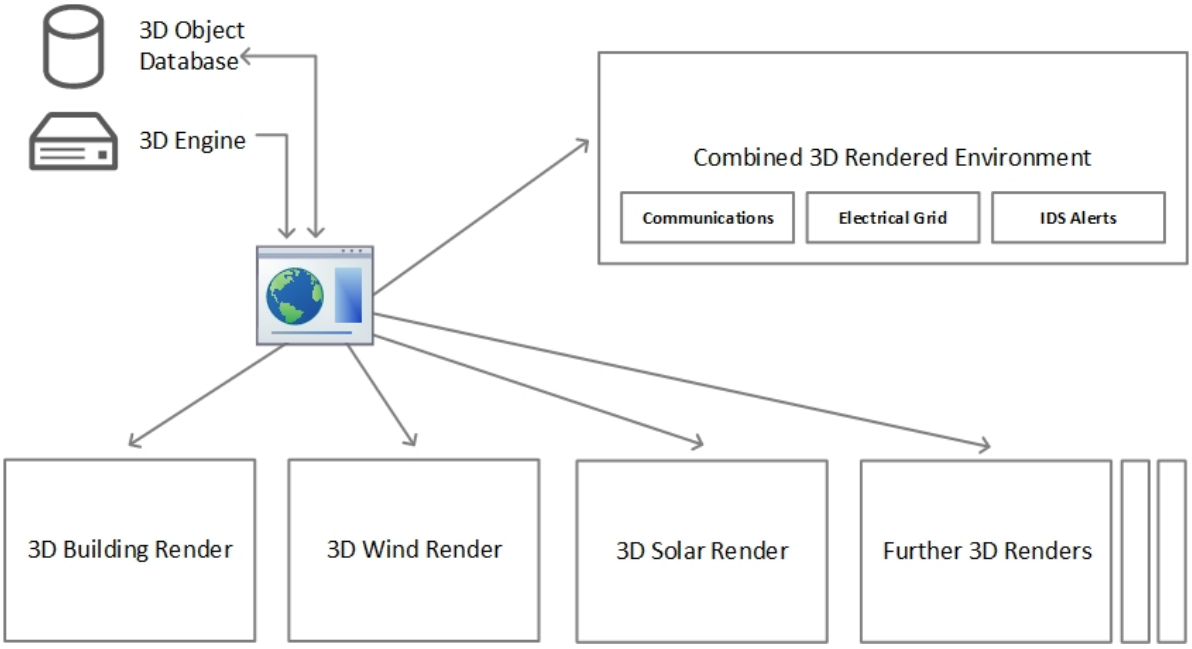

Figure 10. Multiple 3D renders of power and network data sets

The dynamic and multivariate nature of this build-out demands a 3D engine and framework capable of replication and of binding seamlessly to ReactJS, which composes the user interface. Because the visualization is built on web browsers and no such engine exists, the research team built a unique engine, referencing extant engines such as Unity and Unreal for organizational structure. ThreeJS provides a direct way to import and manufacture 3D objects, allowing the development to focus on elements unique to the CEE Platform. This engine permits as many 3D instances as necessary to be built and separated from the main context while adhering to a strict set of guidelines.

Because of the dynamic nature of the application, a formal way to place each element onload or edit is required, and an isometric grid, like those in real-time strategy games, offers a utilitarian structure for all elements to adhere to and speak a common placement language. Instance-rendering was leveraged to allow this grid to support any size of experiment (further explored in Section 3.3). Each tile in this 2D array was passed as a reference to any item located atop it.

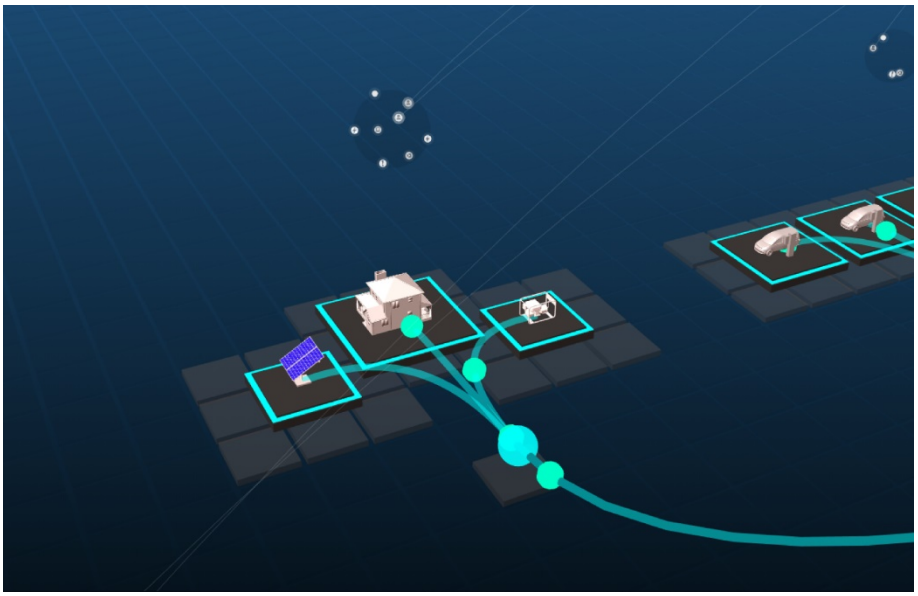

Figure 11. Residential load scaled to its maximum power consumption on the CEE Platform

For instance, each power object sits atop a tile on the grid with given X and Y coordinates, and its platform scales to its maximum power consumption or production to communicate relative sizes in the grid. Once an object is placed, other corelated elements are passed its location on the grid as a reference, as shown in Figure 11, with the residential network.

2.2 Visual Structure and Behavior

The most difficult layer to conceptualize was the network, because it does not neatly align with the coordinate system. Often, networking devices are compressed together, making this a possibly confusing visual.

The solution was to build a relational network first, without placing it onto the underlying grid. The team observed several useful networking properties, enabling us to build an easy-to-understand visual. Each node on a network is part of a subnet—for instance, 192.168.0.1 is part of network 192.168.0.0/24—so, by passing the range (24) and identity (192.168.0.0) to the CEE Platform, the network could easily be divided into smaller subnets. Each subnet hovers above an assigned power partner, whereas unassigned subnets are located by default in the outside Internet, hovering in the center of the grid. Assigned and unassigned subnets divide the network into two distinct network divisions: the experimental network and outside Internet.

To build this structure, each node is placed uniformly on a larger sphere by building a Fibonacci spiral along one polar coordinate and passing each coordinate to a network node. By extending this pattern, other network elements can be bound together, such as subnets to reference entire networks (the Internet) or reference networks to their respective parent. Interestingly, after

constructing this relationship, the team found similarities between the electric grid and network by comparing their respective parent-to-child relationships. This relationship is depicted in Figure 12.

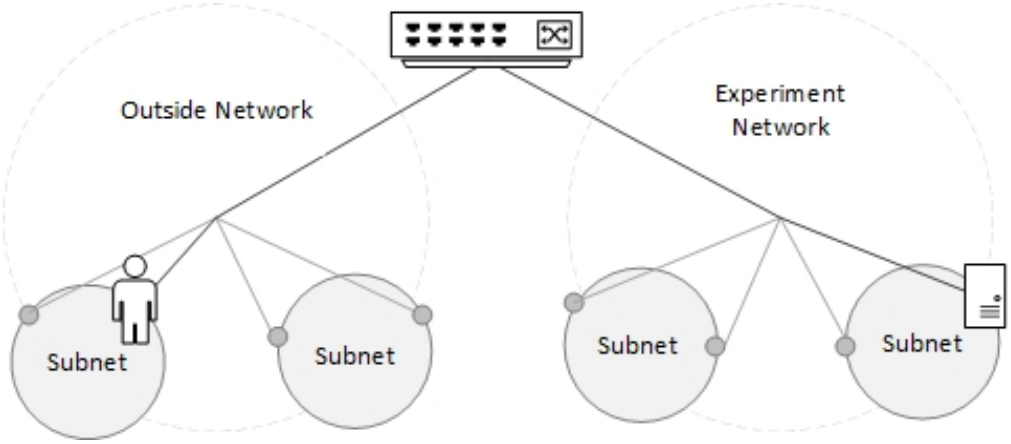

Figure 12. Network connections from parent to child showing similarities to the electric grid

To explain the power visual, all power objects are connected with angled lines, which, when active, change from a gray, transparent color to a fully opaque turquoise and show a moving electron. Although using red to indicate an energized power line might seem like the obvious choice to an electrical engineer, the team chose turquoise instead because other elements in the application use red to indicate a warning state. This is only superficially different from network connections, which contain a moving packet jumping from its source straight to its destination, changing arc height and color if an IDS detects abnormal behavior, as shown in Figure 13.

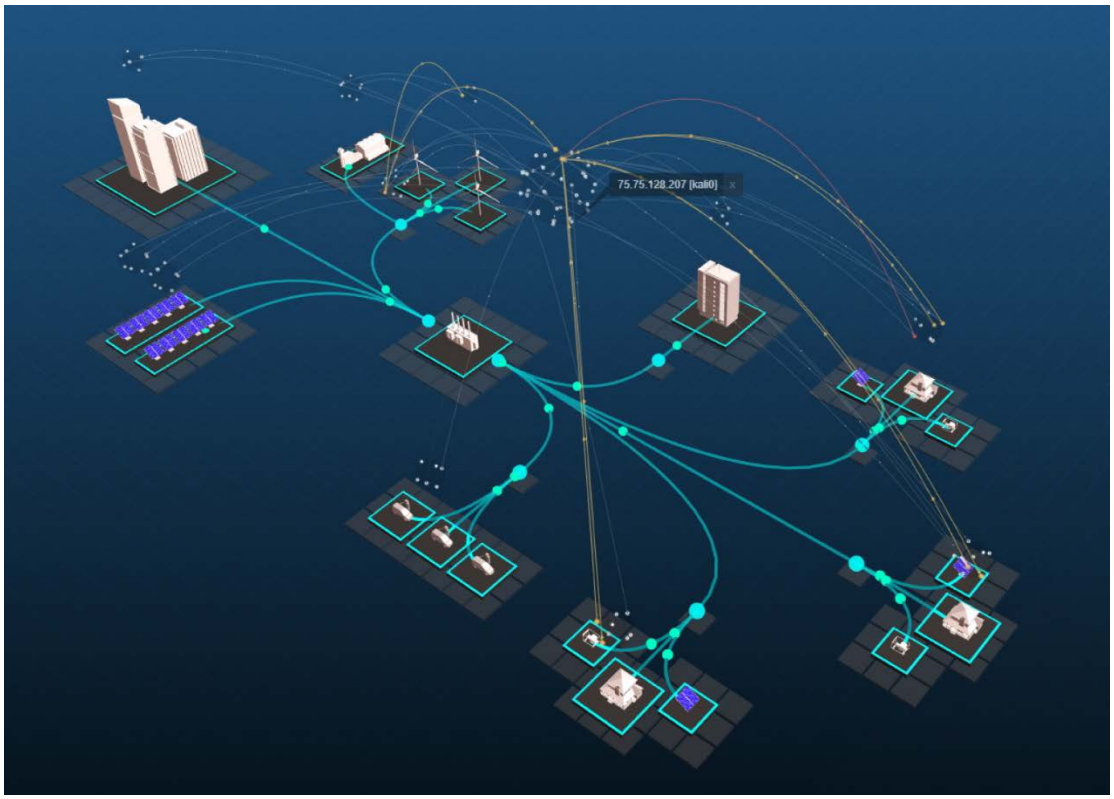

Figure 13. Network view depicting a remote brute-force alert in a residential visualization

This connection pathway permits different visual effects to be applied to communicate different aspects of the connection behavior over time. From OpenDSS, a set of dynamic power properties is received for each line—its directionality, voltage, real power, phase, and so on. Using this, it is possible to build a library of visuals that mirror these properties and alter the behavior of a power line—for instance, wavelike behavior propagating down the line could show wavelength and frequency. Each visual alteration can easily be swapped out during run time without affecting the pathway itself. This ability to visualize different properties of connections will help users understand how different network security postures can affect different aspects of the grid.

2.3 Supporting Lab-Wide Research

The research capability of the CEE Platform is designed to support multiple NREL lab-wide research efforts while being able to run smaller contained experiments. This intent, as well as the capability of the back-end platform, requires a flexible and substantial front-end infrastructure. It needs to be secure and needs to allow control and access to multiple researchers or research teams simultaneously while performing quickly enough to support most desktop platforms. Several desktop 3D development platforms were tested—Unity3D, CRYENGINE, and Unreal Engine 3, for instance—but surprisingly, the ideal platform was the web browser. Distributed desktop applications were found to be less ideal than web applications when connecting to the rest of the experimental infrastructure. This—compounded with unknowns about the desktops that might be running the CEE Platform (the advantage of worrying about only browser support narrowed these variables considerably)—made the decision straightforward. But this design choice came at a cost. Desktop processing, for the moment, outperforms the browser by a significant factor, which means that significant effort was necessarily directed toward designing

the application to optimize performance for every client. Additionally, preloaded assets needed to be of lower polycount to keep load times manageable.

One method used to optimize performance is instance-rendering (shown in Figure 14), a process that uses a single 3D mesh or texture as a "flyweight" to leverage replicating large numbers of objects onto the screen at a reduced cost. This method was used in several places throughout the application, notably on the isometric grid. The entirety of the tiles is only one tile instance, and it is rendered at different X and Y coordinates and passed to the other objects as reference locations. This is a good solution for the isometric grid, where most tiles are static, but this technique is not ideal for interactive elements. To interact with the grid, three consecutive actions are quickly performed: if the user hovers over one tile instance, on the next render, that instance is subtracted from the grid and replaced with a real, interactive tile model at the same location.

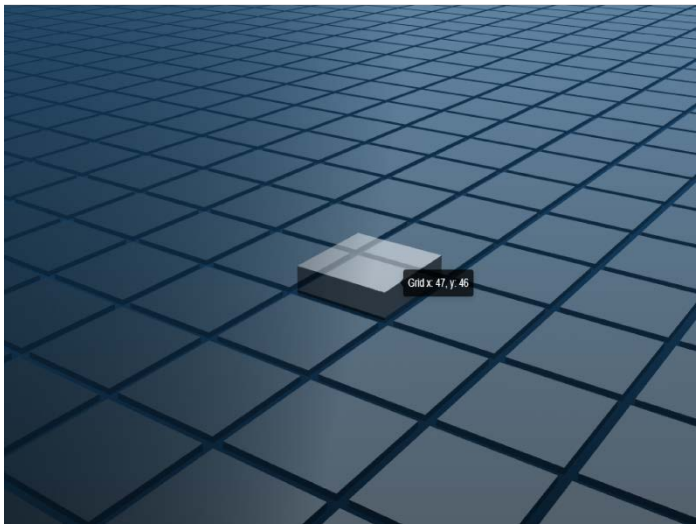

Figure 14. Instance-rendered grid, created with a single 3D mesh or texture as a "flyweight" to leverage replicating large numbers of objects onto the screen

Another performance optimization implemented was reducing how often a new model is re-rendered. This impacted network communications. During network stress tests, the application would grind almost to a standstill. The team observed that this slowdown was caused by network updates that were being generated at approximately 30 times the average rate; however, the number of packets drawn on the visual did not increase by the same factor. Researchers discovered that these packets were being redrawn in the exact location, directly on top of each other, visually indistinct from a single packet. Adding a minimum delta for redraws increased performance by more than 20 frames per second on average.

To serve the client applications, the team built a complete NodeJS server with fully featured authentication, session handling, and server-side rendering capabilities. Building both the client and server applications in TypeScript allowed the front end and back end to share resources, type files, and a selection of libraries, reducing build times and development effort.

2.4 Handling Real-Time Data

A major consideration is ensuring that the server and web clients are in sync with the CEE Platform. Each experiment runs in real time, and each client (3D application) observes as it

changes, receiving all data sets at a constant interval. This is useful only if the latency between the emulator and application is nearly nonexistent and the data sent are useful and manageable. If an experiment is modified, torn down, or rebuilt, all web clients need to be informed.

The communications pathway from the CEE Platform (SCEPTRE and minimega) to a web client is as follows:

1. OpenDSS solves the next series of data.
2. The OpenDSS provider publishes data to the SCEPTRE broker.
3. Updates are serialized as protocol buffer objects and published via a ZeroMQ PUB socket.
4. The event manager, subscribed to broker, receives power values.
5. The event manager stores updates in Elasticsearch with time stamps.
6. The event manager fills a buffer with updates.
7. The event manager sends a filled power updates buffer to the visualization server at a fixed interval.
8. The visualization server sends updates to clients on a web socket.

This pathway illustrates the process for power data. The event manager buffers each data stream at different intervals to the level the 3D visualization can handle. Note that even though data can be sent with very low latency, with an average delay of less than 1 millisecond, the visualization requirements demand this decimation. In fact, the animations are slowed considerably to be visible at all, because network communications happen in a fraction of a second. The ability to query Elasticsearch for the exact time-stamped data makes up for this slowdown and ensures that the research data are still accurate.

To ensure that all clients are keyed into what is happening in their emulation instance, the visualization server is running a session management service. This service maps the ZeroMQ socket to the experimental instances and the experimental instance to clients, and it stores this information in a database, MongoDB. The service is constantly determining whether each experiment is active, powered down, stopping, or starting. Once a client is interacting with a visualization, they control the zoom level of the camera and, therefore, the zoom level of the data itself. The visualization has certain break points in zoom level to colocate multiple objects (for instance, a set of buildings) into a single object (such as a city).

This means that the visualization server needs to know which 3D instance a client is viewing at any time and the level to which each client is magnified to send the right granularity of data to each client. A client viewing the buildings along with individual transmission lines and transformers needs every power update, but at the city level it needs only approximate values. A filter is set between packaging the data and the web socket, which essentially works like a valve on a fire hydrant, controlling the stream to avoid flooding the client with useless data.

3 Capabilities

3.1 Real-Time Visual Analysis

As the primary driver of this capability, the research team designed the CEE Platform with the intention to achieve real-time visualization analysis. Using ZeroMQ PUB-SUB sockets, the application achieves very low-latency data streams as well as high-throughput message passing between individual processes. This communications fabric using ZeroMQ makes possible the current level of responsiveness and accuracy of the experiment in this system. Another feature available in the PUB-SUB model is the simultaneous availability of streaming real-time data to any application subscribing to the right PUB socket. In case a third-party application connected to the CEE Platform data streams requires the stream to be filtered or parsed, a custom script can be used to handle the data before relaying it out to any dependent applications. Data are serialized for transport on ZeroMQ sockets using protocol buffers. Protocol buffers, also known as protobuf, are a light and simple method for serializing structured data usable for a variety of different applications (Protocol Buffers 2019, Popić et al. 2016).

3.2 Historical Data Analysis

All data streams are passed to indexes in the Elasticsearch database. Each data point is associated with a time stamp; thus, it is possible to query the database for historical events in the simulation and to go back in time to replay the simulation from the available data. By storing the experimental data in a database, any interested party could be granted network access to the database and, hence, query any data needed for analysis. Because this database stores all experimental data, it is necessary to have enough storage space to accommodate large or long-running experiments. Currently, the system is configured with 8 TB of storage space, with the option to expand in the future if necessary. Figure 15 shows a view in Kibana of the data in the Elasticsearch database. Although this screenshot shows that the database is currently accessible from any device within the environment, this feature might be restricted or disabled entirely for experiments with stricter security requirements.

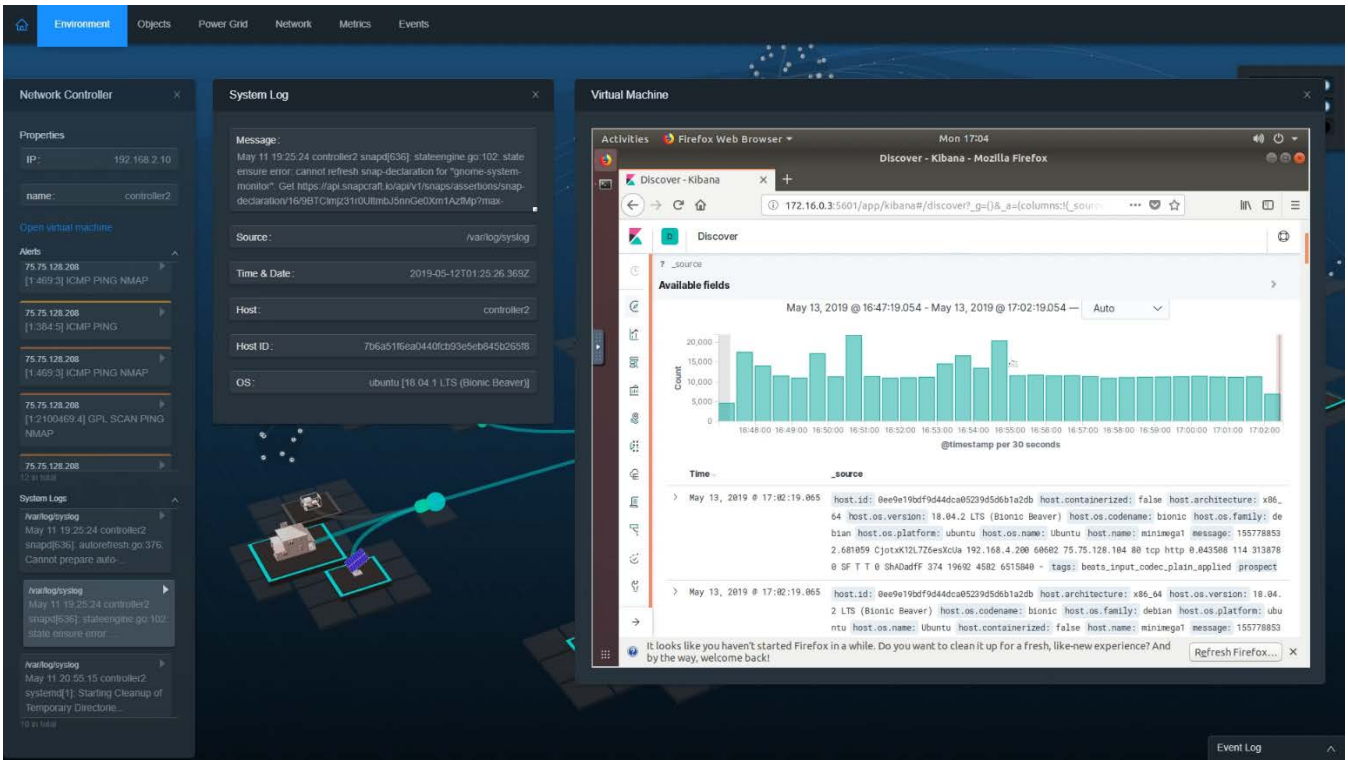

Figure 15. Kibana view of syslog data in the Elasticsearch database

3.3 Integrated Intrusion Detection and Prevention Systems

The CEE Platform provides the capability to perform intrusion detection and prevention across experimental systems and networks. As a method for achieving a proof of concept for security-alerting visualization, intrusion detection and prevention systems have been integrated in a manner that allows for systems and networks to perform system and network security functions and to generate system logs that are collected and stored in the Elasticsearch database using the built-in Filebeat service. This allows the system to integrate multiple different IDSs simultaneously and provide detailed logs of the output of these systems for the duration of the experiment. This means that the system could be used to run comparative analyses of IDSs, both those currently available and any developed in the future. Because the network is almost entirely software-defined, intrusion prevention measures and moving-target defense strategies might be deployed, whether using a stand-alone intrusion prevention system or an automated response mechanism. Figure 16 shows the visual depiction as well as the detailed log of a real-time alert generated by Suricata IDS, which was the IDS deployed in this instance.

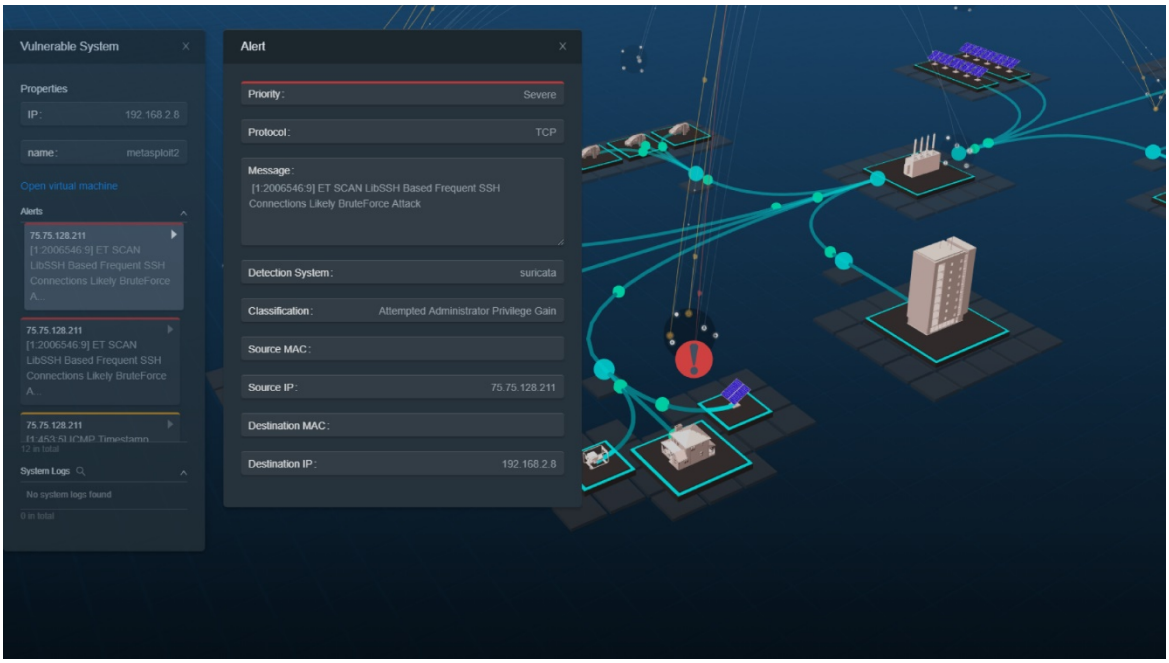

Figure 16. Visualized Suricata of an IDS alert with detailed log

3.4 Hardware-in-the-Loop Capability

Researchers at NREL recognize that the ability to link cyber-physical systems with the emulated devices in an experimental environment is a primary capability for research, development, testing, and validation for cyber-physical systems. By integrating real physical networks with the CEE Platform through Open vSwitch, any real device could be brought into this environment. If the network traffic passes through the emulated network, it will be captured, logged, and visualized by the environment. Through the user interface, Figure 17 shows the output of a real, physical webcam pointed at a rack of hardware devices, all of which are connected within the CEE Platform environment.

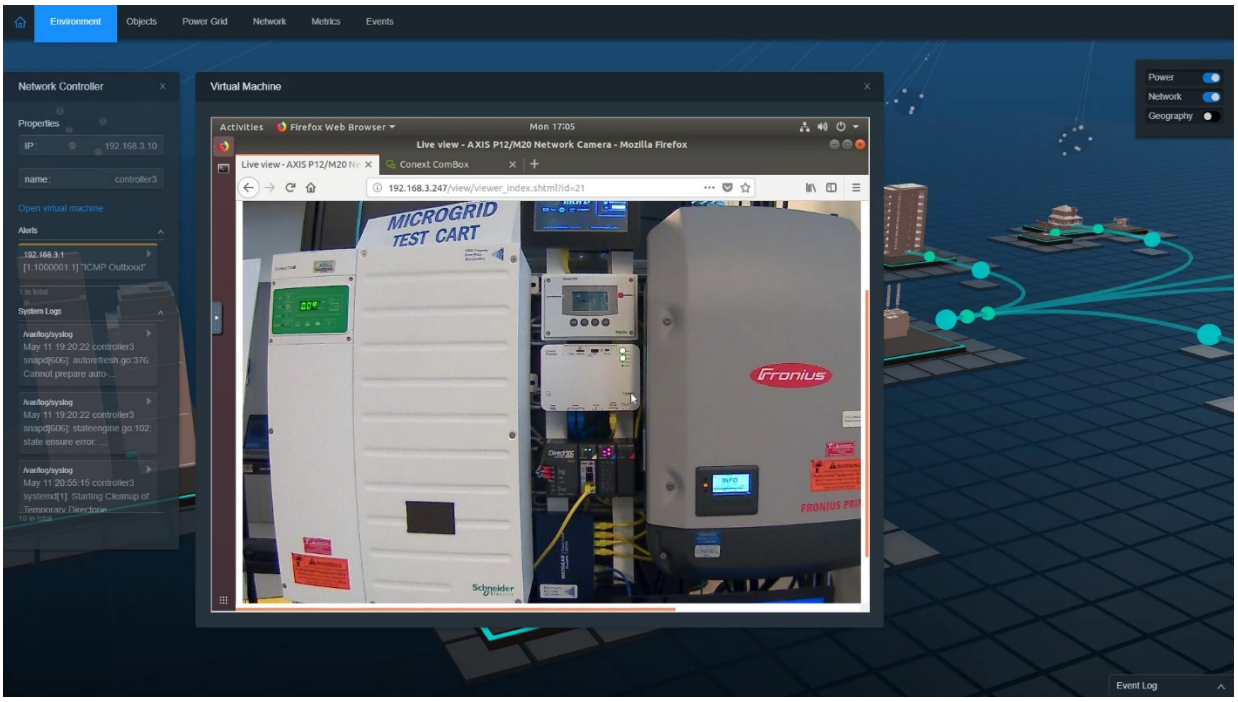

Figure 17. Connected webcam viewed from inside the CEE Platform environment

3.5 Robust Data Transport and Storage

Leveraging Go, ZeroMQ, Filebeat, Logstash, and Elasticsearch, this capability can achieve robust data transport and storage. This is done by the simultaneous availability of real-time data streams on ZeroMQ PUB sockets as well as all historical data collected via the Filebeat or Logstash services and custom Go scripts on the Elasticsearch database. The integrity of the database is maintained using the backup and redundancy features available in Elasticsearch.

4 Applications

4.1 Incident Handling and Response

By running several different security incident scenarios multiple times with controlled settings, security incident responses can be enacted, evaluated, and improved. This enables significant application in creating new incident response procedures and performing continuous evaluation of any conceivable type of incident, even if it normally occurs infrequently, because of the highly repeatable nature of a controlled experiment in this environment. By including the entire cyber-physical system of the environment in the experiment, users can consider the effects of the taken actions on the surrounding connected systems while enacting the developed response strategy. The additional situational awareness provided by the CEE Platform also serves to further inform users and improve their overall incident-handling and response capacity.

4.2 Computer Forensics

The CEE Platform includes functionality to simultaneously visualize alerts from a series of different IDSs. Alert data are also stored in a database; therefore, a user can query the central database for alert data as well as system logs associated with incidents or related events. Both real-time and historical data could be supplied to forensic analysis tools.

The ability to clearly understand and convey information about cybersecurity incidents presents a serious challenge. The CEE Platform demonstrates an attempt to link all system, network, security, and electric grid state information to a central location for time-synchronized analysis and visual presentation of data to help answer the core questions following any incident: namely, who, what, when, where, why, and how?

4.3 Building Training Sets for Machine Learning

As the relationship between machine learning and resilience becomes more evident, so does the potential for the array of data sets collected across experiments to be used to build training sets for machine-learning algorithms that, in turn, can support autonomous response and recovery of systems, networks, and applications. The CEE Platform hardware must ensure it has allocated enough resources for the collection and storage of these high-fidelity data sets that can be used for machine learning.

4.4 Hardware Evaluation

This cyber-power capability provides an environment in which various kinds of Internet Protocol-based hardware can be linked and evaluated. Hardware evaluation can explore systems security as well as the state of resilience. Hardware could also be evaluated from the perspective of interoperability and optimization. Further, the CEE Platform has great potential for automated firmware testing.

4.5 Education and Training

Previous research proves the benefit of visualization tools in improving the efficiency and effectiveness of training and education programs (Baecker 1998). Given its real-time and historic visualization capabilities, the CEE Platform can support the education of cybersecurity and power systems researchers in an immersive environment. By guiding users through a scenario in

detail as it is being played out visually, the CEE Platform can convey important information about the incident in an understandable manner for diverse audiences. Clearly, there is great potential for the CEE Platform to be used as a tool to accelerate education about cybersecurity and interconnected cyber-physical systems.

5 Future Work

Future research capabilities for cyber-physical systems at NREL will be robust because of the CEE Platform's dynamic nature and sturdy overall functionality. The overall system, network, and application design and flexibility of these interlinkages demonstrate forethought for potential future simulation data layers that connect needed areas of focus in research and development. Future work currently planned for the CEE Platform is focused on the resilience of the future electric grid, distributed generation, and machine-learning efforts that link incident and/or event identification and autonomous response through secure command and control. On the other end of the spectrum, more specific to education and training, the CEE Platform demonstrates the potential to support workforce development efforts "…to maximize the cybersecurity talents and capabilities" highlighted by Executive Order 13870 on America's Cybersecurity Workforce (Exec. Order 13870 2019). In addition, the capability directly aligns with the U.S. Department of Energy's *Multiyear Plan for Energy Sector Cybersecurity*, which seeks to complete a "...prototype of a tool or technology for next-generation energy delivery systems to isolate, encapsulate, and reject data or algorithms subjected to malicious compromise" (U.S. Department of Energy Office of Energy Efficiency and Renewable Energy 2018).

5.1 Resilience

Because the CEE Platform provides all the features of software-defined networking and KVM, these features could be leveraged to study and exercise the level of resilience of the system, network, and/or any running applications. Depending on which cyber-physical layer is being studied, different factors might affect resilience. Specifically relating to the power system, the CEE Platform could be used to develop and visually represent resilience metrics for individual power systems or microgrids. Looking forward, by integrating this resilience metric calculation as a layer in the environment, the values of the developed resilience metrics could be displayed for each element in real time.

5.2 Machine Learning

Future plans for CEE Platform development include leveraging artificial intelligence. As an emulation and co-simulation platform, it generates in real time a large number of data sets, including power system states, communications, security alerts, and system logs. This capability could be used to generate data sets for machine-learning research. The CEE Platform has two main ways of supporting machine learning: (1) it can directly integrate tools, leveraging machine-learning approaches such as an anomaly-based IDS; or (2) it can provide the generated environment data to a machine-learning platform for analysis or as a training set.

5.3 Workforce Development

This current application could be readily translated to a version focused on workforce development. Because the CEE Platform allows users to visualize multiple data layers, effectively providing high-fidelity visualizations of system events across different cyber-physical domains, it could have significant application in this area.

Because multiple users can work simultaneously in the same emulated environment, the CEE Platform can be used to set up detailed, long-running scenarios in complex, large-scale environments. Leveraging this, instructors could play out specific training scenarios for

individuals to demonstrate, develop, and evaluate users' ability to handle these scenarios. By using the CEE Platform to replay key moments of the scenario, instructors can compare expected responses and actions with those performed by users during a live scenario.

6 Conclusion

This paper presented the core components, system architecture, developmental lessons learned, and selected potential applications of the CEE Platform. In demonstrating these details about the platform, and because of the open-source nature of the underlying tools, it is possible to build a system with similar capabilities by integrating the core tools as outlined or by using equivalent components in a comparable architecture. Because nearly all the tools used to develop the CEE Platform are open source, this emulation framework can leverage the robust, open-source support community available for each tool. With its potential for accelerating cybersecurity research as well as education and training, the development team sees the CEE Platform as an emulation tool for pushing the frontiers of our current understanding of complex and interconnected systems at scale. Currently, the CEE Platform is in active development at NREL and is in a closed alpha development stage. This platform has been awarded U.S. Provisional Patent Application No. 62/913232.

References

Baecker, Ronald. 1998. "Sorting Out Sorting: A Case Study of Software Visualization for Teaching Computer Science." In *Software Visualization: Programming as a Multimedia Experience*, edited by John T. Stasko, Marc H. Brown, John B. Domingue, and Blaine A. Price, 369–381. Cambridge: MIT Press.

Bosch, Robert, Chris Stolte, Diane Tang, John Gerth, Mendel Rosenblum, and Pat Hanrahan. 2000. "Rivet: A Flexible Environment for Computer Systems Visualization." *ACM SIGGRAPH Computer Graphics* 34, no. 1: 68–73.

Bresciani, Sabrina, and Martin J. Eppler. 2009. "The Benefits of Synchronous Collaborative Information Visualization: Evidence from an Experimental Evaluation." *IEEE Transactions on Visualization and Computer Graphics* 15 no. 6: 1073–1080. https://ieeexplore.ieee.org/document/5290714.

Chassin, David P., Kevin Schneider, and Clint Gerkensmeyer. 2008. "GridLAB-D: An Open-Source Power Systems Modeling and Simulation Environment." *Proceedings of the 2008 IEEE/PES Transmission and Distribution Conference and Exposition*. https://ieeexplore.ieee.org/document/4517260.

Executive Order 13870 of May 2, 2019, on America's Cybersecurity Workforce. 2019. Code of Federal Regulations. https://www.whitehouse.gov/presidential-actions/executive-order-americas-cybersecurity-workforce/.

Fisk, Mike, Steven A. Smith, Paul Weber, Satyam Kothapally, and Thomas Caudell. 2003. "Immersive Network Monitoring." *Proceedings of the 2003 Passive and Active Measurement Workshop*. http://citeseerx.ist.psu.edu/viewdoc/citations?doi=10.1.1.13.6374.

Go. 2019. "Download Go." Accessed December 9, 2019. https://golang.org.

Huffer, Kelly, John Goodall, Maria Vincent, Michael Iannacone, and Robert Bridges. 2017. *Stucco System* (ORNL/SPR-2017/527). Oak Ridge, TN: Oak Ridge National Laboratory. https://info.ornl.gov/sites/publications/Files/Pub104227.pdf.

Jacobs, Nicholas, and Jay Tillay Johnson. 2018. *SCEPTRE: Power System and Networking Co-simulation Environment* (SAND2018-5328PE). Albuquerque, NM: Sandia National Laboratories. https://www.osti.gov/servlets/purl/1515324.

Jalili-Marandi, Vahid, Fabio Jose Ayres, Esmaeil Ghahremani, Jean Belanger, and Vincent Lapointe. 2013. "A Real-Time Dynamic Simulation Tool for Transmission and Distribution Power Systems." *Proceedings of the 2013 IEEE Power and Energy Society General Meeting*. https://ieeexplore.ieee.org/document/6672734.

Kivity, Avi, Yaniv Kamay, Dor Laor, Uri Lublin, and Anthony Liguori. 2007. "kvm: The Linux Virtual Machine Monitor." *Proceedings of the Linux Symposium*. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.488.2278&rep=rep1&type=pdf.

minimega. 2019. “minimega: A Distributed VM Management Tool.” https://minimega.org/.

Open vSwitch. 2016. “Production Quality, Multilayer Open Virtual Switch.” Accessed January 8, 2020. https://www.openvswitch.org/.

Palmintier, Brian, Dheepak Krishnamurthy, Philip Top, Steve Smith, Jeff Daily, and Jason Fuller. 2017. “Design of the HELICS High-Performance Transmission-Distribution-Communication-Market Co-Simulation Framework.” *Proceedings of the 2017 Workshop on Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES)*. https://ieeexplore.ieee.org/document/8064542.

Popić, Srđan, Dražen Pezer, Bojan Mrazovac, and Nikola Teslić. “Performance Evaluation of Using Protocol Buffers in the Internet of Things Communication.” *Proceedings of the 2016 International Conference on Smart Systems and Technologies (SST)*. https://ieeexplore.ieee.org/document/7765670.

Protocol Buffers. 2019. “Developer Guide.” Accessed December 9, 2019. https://developers.google.com/protocol-buffers/docs/overview.

U.S. House of Representatives. Committee on Homeland Security. Subcommittee on Cybersecurity and Infrastructure Protection. *Challenges of Recruiting and Retaining a Cybersecurity Work Force*. 115th Cong., 1st sess., 2017. Committee Print 1-54. https://www.govinfo.gov/content/pkg/CHRG-115hhrg28415/pdf/CHRG-115hhrg28415.pdf.

Ramachandran, Vaidyanath. 2011. “Modeling of Utility Distribution Feeder in OpenDSS with Steady State Impact Analysis of Distributed Generation.” Master’s thesis, West Virginia University. https://researchrepository.wvu.edu/etd/371/.

Rossey, Lee M., Robert K. Cunningham, David J. Fried, Jesse C. Rabek, Richard P. Lippmann, Josh W. Haines, and Marc A. Zissman. 2002. “LARIAT: Lincoln Adaptable Real-Time Information Assurance Testbed. *Proceedings of the IEEE Aerospace Conference*. https://ieeexplore.ieee.org/document/1036158.

Tang, Peiyi. 2019. “Multi-Core Parallel Programming in Go.” *Proceedings of the First International Conference on Advanced Computing and Communications*, 64–69. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.467.9137.

U.S. Department of Energy Office of Electricity Delivery and Energy Reliability. 2018. *Multiyear Plan for Energy Sector Cybersecurity*. Washington, D.C. https://www.energy.gov/sites/prod/files/2018/05/f51/DOE%20Multiyear%20Plan%20for%20Energy%20Sector%20Cybersecurity%20_0.pdf.