Toward a Cyber-Physical Topology Language:

Applications to NERC CIP Audit

Gabriel A. Weaver[†], Carmen Cheh[†], Edmond J. Rogers[†], William H. Sanders[†] and Dennis Gammel[‡]

[†]University of Illinois at Urbana-Champaign, {*gweaver,cheh2,ejrogers,whs*}*@illinois.edu*
[‡]Schweitzer Engineering Laboratories, *Dennis_Gammel@selinc.com*

ABSTRACT

Our Cyber-Physical Topology Language (CPTL) provides a language that utilities can use to programmatically analyze current and future cyber-physical architectures. The motivation for our research emerged from the importance and limitations of several audit scenarios: account management, vulnerability assessment, and configuration management. Those scenarios occur in the context of the North American Electric Reliability Corporation's Critical Infrastructure Protection (NERC CIP) audits. The NERC CIP standards define security controls by which utilities must be audited. Although the standards were designed to make power control networks less vulnerable to cyber attack and to decrease the chance of outages, the audit process is manual and costly. In order to save utilities and auditors time and money, we used the limitations of those audit scenarios in formally specifying and implementing CPTL, which consists of both a representation of cyber-physical assets and operations upon that representation. First, CPTL uses graph theory to represent a network of cyber-physical assets; we currently implement this representation in GraphML. Second, CPTL defines operations upon that representation. In this paper, we introduce operators to process attributes by expanding and contracting components of a network, and implement these operations using the Boost Graph Library (BGL). In order to demonstrate the potential for CPTL to save auditors and utilities time and money, we provide a detailed example of how CPTL could help with vulnerability assessment and discuss additional applications beyond the audit scenarios mentioned above. We describe current approaches to those scenarios and argue that CPTL improves upon both the state-of-the-art and current practice. In fact, we intend CPTL to enable a broad range of new research on realistic cyber-physical architectures by giving utilities, auditors, managers, and researchers a common language with which to communicate and analyze those architectures.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations—*network management*; G.2.3 [**Discrete Mathematics**]: Applications

Keywords

NERC CIP; graph theory; audit

1. INTRODUCTION

The power grid has been called the "world's biggest machine" by engineers and is part of the "greatest engineering achievement of the 20th century" [32]. The smart grid promises to increase the overall reliability of the power grid. However, if the number of computational devices in the grid increases, then the attack surface of the grid may also increase. For example, the NISTIR Guidelines for Smart-Grid Cyber Security state that "increasing the complexity of the grid could introduce vulnerabilities and increase exposure to potential attackers and unintentional errors" [24].

Even a high-level view of today's smart grid shows a great deal of complexity [24]. Just one Investor Owned Utility (IOU) in 2009 was responsible for 200 transmission substations and 1 million residential customers [3].

In the future, the complexity of the smart grid will likely increase. In 2010, there were approximately 160 million residences in the U.S. [40], but by 2012 only 18 million smart meters were deployed [2]. The discrepancy between the number of residences and the number of smart meters indicates that many more smart meters will be deployed in the future. Furthermore, there were 250 million registered cars in 2010 [31], and as more cars become electric, we will see an increasing number of devices on the smart grid. For example, car batteries may be used to store electricity, and the power grid will need to coordinate vehicle charging with other electrical loads.

Our Contribution: In this paper, we define and implement a Cyber-Physical Topology Langauge (CPTL) so that utilities can systematically prepare for and demonstrate compliance during an audit. In addition, our tools might also be useful in the context of the Identify Assess and Correct (IAC) program in NERC CIP v5 or NERC's Reliability Assurance Initiative (RAI) program. Both of those programs encourage utilities to have robust internal security and reliability practices that they can demonstrate to NERC. De-

pending upon those programs, a utility's audit schedule and scope may be modified.[1]

Although we envision a wide variety of applications (which we will discuss in Section 6), in this paper, we focus on the following "audit scenarios": account management, vulnerability assessment, and configuration management.[2]

Audit Scenarios: In the context of this paper, an **audit scenario** is a set of NERC CIP standards related by a common topic. Utilities are audited against the North American Electric Reliability Corporation's Critical Infrastructure Protection (NERC CIP) standards (which are treated as if they are regulations). Those standards define basic and sound security controls and help improve the reliability of the Bulk Electric System (BES) [34].

As the smart grid matures and more computational devices are deployed in the power grid, there will be a larger attack surface in electrical power networks. Current approaches to audit are manual and labor-intensive. Furthermore, as smart grid infrastructures evolve, utilities and auditors alike need to be able to understand how architectural changes affect auditing requirements. For example, one area of active research is how the migration of substation functionality to the cloud could affect security and compliance [17].

We argue that we can improve the scalability and cost of audit across a wide variety of smart grid architectures if we can address two limitations of current audit procedures. First, we need to be able to integrate and process information from a wide variety of data sources. Second, we need to be able to operate on different views of networked cyber assets at multiple levels of granularity. Both of those limitations were recognized in the *Roadmap to Achieve Energy Delivery Systems Cybersecurity*, a strategy framework for smart grid cybersecurity published in 2011 by the Department of Energy (DOE) [5].[3]

The Cyber-Physical Topology Language: We designed CPTL so that utilities can systematically operate on relations between networked resources found in a variety of smart grid architectures. CPTL is a domain-specific language for operating upon and analyzing cyber-physical networks relative to high-level compliance goals or low-level architectural details.

In contrast to a static network topology diagram, which only provides a visualization of a network, CPTL enables utilities to operate systematically upon networked assets. CPTL's network and attribute contraction and expansion operations allow practitioners to compute and view networked cyber-physical assets at different levels of granularity.

We use graph theory to formalize the already-common practice of using a network architecture diagram to reason about a network. Graph theory provides an additional benefit; practitioners can relate constructs from a variety of different languages and formats currently used to describe grid devices. Those languages and data formats include the Common Information Model (CIM) [16], Cisco IOS, Energy Systems Provider Interface XML (ESPI-XML) [20], the Substation Configuration Language (SCL) [26], DNP3 [19], and the Inter-Control Center Communications Protocol (ICCP) [25]. Furthermore, our approach allows us to relate higher-level information, such as threat models or natural-language security documents, to a representation of the underlying cyber-physical architecture.

The remainder of this paper is organized as follows. In Section 2 we outline the importance and limitations of NERC CIP audit as currently practiced. In Section 3 we explain how those limitations motivated the design, formalization, and ongoing implementation of CPTL. In Section 4 we describe CPTL's implementation and apply CPTL to account management, vulnerability assessment, and configuration management. In Section 5 we briefly describe related work. Finally, Section 6 outlines potential future work, and Section 7 concludes.

2. AUDIT SCENARIOS

We claim that CPTL can streamline several aspects of NERC CIP audit. In this section, we describe NERC CIP audits and their importance with respect to account management, vulnerability assessment, and configuration management. Finally, we will discuss the limitations of NERC CIP audit as currently practiced and explain how these limitations motivated CPTL.

During a NERC CIP audit of an electrical power substation, network administrators must produce evidence that their substation networks satisfy various standards. These standards define basic security controls, and failure to meet these controls has severe consequences. In general, the consequences of failing to fulfill NERC CIP standards are severe and may include fines that scale up to $1 million for every day that the violation occurred [15]. In addition to having financial consequences, a lack of sound security controls could make a control network vulnerable to cyber attacks and their consequences, including power outages.

Table 1 enumerates the NERC CIP standards for each of our audit scenarios.[4] We now briefly describe the importance of each of the audit scenarios, the measures currently used to demonstrate compliance, and the limitations of these approaches.

Account Management: The account management audit scenario requires utilities to inventory, verify, and revoke user accounts and user account groups. Furthermore, utilities must also have a procedure for identifying who has access to which accounts, and for managing passwords. This scenario is important to ensure that practitioners understand who has access to the substation network and in what capacity.

Current approaches to this auditing scenario are largely manual. Example evidence enumerated within the NERC CIP standard includes a dated list of all accounts and groups or roles in the system, a summary of privileges associated with each group, a list of people associated with shared accounts, and records that verify that default passwords have been changed [34].

[1] At the time of publication, we are uncertain whether the Federal Energy Regulatory Commission (FERC) will agree with those programs.

[2] Within the scope of NERC CIP, several of those terms have very specific meanings (e.g., Configuration Management). In our discussion, those terms may take on a broader scope than that currently defined by the NERC CIP standards.

[3] For the remainder of this paper, we will refer to this framework as the *Roadmap*.

[4] Although we use version 5 of the NERC CIP standards, at the time this paper was written, they were not yet approved by the Federal Energy Regulatory Commission (FERC).

account management	
standard	**description**
CIP-004-5, Requirement R4, Part 4.3	For electronic access, verify at least once every 15 calendar months that all user accounts, user account groups, or user role categories and their specific, associated privileges are correct and are those that the Responsible Entity determines are necessary.
CIP-007-5, Requirement R5, Part 5.2	Identify and inventory all known enabled default or other generic account types, either by system, by groups of systems, by location, or by system type(s).
CIP-007-5, Requirement R5, Part 5.3	Identify individuals who have authorized access to shared accounts.
CIP 007-5, Requirement R5, Part 5.4	Change known default passwords, per Cyber Asset capability.

vulnerability assessment	
standard	**description**
CIP-010-1, Requirement R3, Part 3.1	At least once every 15 calendar months, conduct a paper or active vulnerability assessment.

configuration management	
standard	**description**
CIP-007-5, Requirement R1, Part 1.1	Where technically feasible, enable only logical network-accessible ports that have been determined to be needed by the RA, including port ranges or services where needed to handle dynamic ports. If a device has no provision for disabling or restricting logical ports on the device, then those ports that are open are deemed needed.
CIP-007-5, Requirement R1, Part 1.2	Protect against the use of unnecessary physical input/output ports used for network connectivity, console commands, or removable media.
CIP-010-1, Requirement R1, Part 1.1	Develop a baseline configuration, individually or by group, which shall include the following items: 1.1.1 Operating system(s) (including version) or firmware where no independent operating system exists; 1.1.2 Any commercially available open-source application software (including version) intentionally installed; 1.1.3 Any custom software installed; 1.1.4 Any logical network accessible ports; and 1.1.5 Any security patches applied.
CIP-010-1, Requirement R1, Part 1.2	Authorize and document changes that deviate from the existing baseline configuration.

Table 1: Audit scenarios that cover a wide variety of NERC CIP standards [34].

Vulnerability Assessment: Vulnerability assessments are important to ensure that security controls are properly implemented within a substation network. As noted in CIP-010-5, Requirement R3, Part 3.1, a utility must perform these checks every 15 calendar months (even though the utility may be on a three or six-year audit cycle) [34].

The current approach to vulnerability assessment is to perform a review of the cyber-security controls and policies described in the compliance documentation. The review includes verification that ports and services described in compliance artifacts are accurate. The verification is often performed by contracted independent analysts. The utilities then document the assessment results, the controls evaluated for each system, and the output of any tools used to perform the assessment [34].

Configuration Management: The NERC CIP regulations that pertain to configuration management require utilities to document logical network ports and services as well as baseline configurations for devices on a substation network.

Currently, many utilities demonstrate compliance for this audit scenario with manual documentation. For ports and services, evidence of compliance may include written documentation about the ports enabled on network devices and firewall configuration files. For baseline configurations, practitioners may use a spreadsheet or a ticketing system to store the relevant information (e.g., OS, software installed, logical network ports, and patches). Unfortunately, those approaches are largely manual and will not scale when more devices are added or audits become more frequent.

2.1 Current Limitations

As mentioned before, NERC CIP audits, though required and likely to become more frequent, are both manual and costly. During an audit, utilities commonly present different types of information to demonstrate compliance. Auditors must decide whether the information presented constitutes adequate evidence of compliance, and this process can be time-consuming.

Also NERC CIP audits can be costly in terms of both time and money. A conservative estimate suggests that audits consume 30 man-days of work per day of audit, and this estimate does not include ramp-up time for the auditors. In terms of money, audits may cost large IOUs from hundreds of thousands to millions of dollars. Currently, smaller utilities with a less widescale reliability impact are on a six-year audit cycle, while larger utilities are on a three-year audit cycle. There is demand for more frequent audits, and utilities will play a larger role in this process in the future. In fact, NERC is currently working on programs in which utilities are expected to have robust internal security and reliability practices that they can demonstrate to NERC. Utilities' performance on those internal practices could adjust audit schedules and scopes [30].

We can improve the scalability and cost of audit if we can address limitations recognized in the 2011 *Roadmap*. The *Roadmap* states that in order to assess and monitor risk in the smart grid, capabilities need to be developed that provide "actionable information about security posture from vast quantities of disparate data from a variety of sources and levels of granularity" [5].

Thus, practitioners need tools that can process data from a variety of sources. Configuration files for individual devices (firewall rules and Windows Registries), as well as high-level descriptions of the network architecture itself (SCL), are all potential sources of information for configuration management. We can process that information to understand how a substation control network evolves.

Practitioners also need the ability to operate on a substation network architecture at a variety of levels of granularity. For example, the Schweitzer Engineering Laboratories (SEL) 3354 Intelligent Electronic Device (IED) runs Active Directory Certification Services in order to create, distribute, and revoke substation PKI certificates. Practitioners may want to document and analyze not only the SEL 3354 as a network device, but also the processes that provide these services.

3. CPTL

In this section, we define the Cyber-Physical Topology Language (CPTL). CPTL formalizes cyber-physical network topologies as a "datatype." A *datatype* consists of a set of strings and a set of operations upon those strings [38]. In CPTL, the set of strings corresponds to a graph-theoretic representation for a cyber-physical network topology. In addition, we define two kinds of CPTL operations: network expansion and contraction, and attribute expansion and contraction. In the subsections that follow, we use graph theory to define CPTL primitives that represent cyber-physical networks, and operations upon those primitives.

3.1 Primitives

Practitioners traditionally present computer networks as diagrams in which network devices are connected via lines that reflect communication paths. The devices may be annotated with information such as their IP addresses or services. CPTL formalizes a computer network as a *target system network*, a graph whose vertices are a set of network devices and whose edges capture a communication path between those devices.

DEFINITION 1. *A* **target system network** *is a graph $G = (V, E)$ in which the vertices $V(G)$ represent networked cyber-physical assets and the edges $E(G)$ relate a pair of those assets.*

CPTL extends a target system network with *vertex and edge attributes*. Vertex and edge attributes give practitioners a CPTL primitive for relating arbitrary data structures to a target system network. The data structures may represent constructs in high-level natural-language documents (e.g., pertinent NERC CIP provisions) or lower-level information, such as a device IP address or configuration information. For example, the output of the *nmap* tool relates a network device to an IP address, a service, or a possible OS running on that device. Even some SCADA threat model surveys relate information to an underlying network architecture [54].

DEFINITION 2. *A* **vertex attribute** *is a function $a_v : V(G) \rightarrow S_V$ in which $V(G)$ is the set of vertices in a target system network and S_V is a set of structures.*

For example, we could represent the output of the *nmap* command over a computer network as a target system network and several vertex attributes. The vertices of the target system network represent the set of network devices found while scanning. The edges of the target system graph correspond to communication paths between devices. Finally, we could define three vertex attribute functions that map the vertex set to a set of IP addresses, a set of OSes, and a set of services.

DEFINITION 3. *An* **edge attribute** *is a function $a_e : E(G) \rightarrow S_E$, where $E(G)$ is the set of edges in a target system network and S_E is a set of structures.*

For example, edge attributes may be used to indicate which protocol (such as DNP3 or ICCP) is used on a given communication link.

3.2 Operations

We want to support a variety of operations that utilities can use on target system networks to encode analyses that streamline NERC CIP audits. Although we are considering other operations, we focus on network and attribute contraction and expansion.

Practitioners need to operate on a cyber-physical topology at a variety of levels of granularity. Therefore, we formally define network/attribute contraction and expansion in order to satisfy that requirement. Network contraction and expansion operate upon the target system network. Attribute contraction and expansion operate upon the attributes associated with vertices and edges in a target system network. Both operations are useful in practice. For example, auditors might want to assess both the "vulnerability" of a particular host on a network and the overall vulnerability of all hosts in a given subnet.

Network and Attribute Contraction: Since there are a variety of ways to contract a graph, we provide a very general formalization of network contraction. The contraction operation collapses one or more nodes in a network of cyber-physical assets to provide a higher-level view of those assets.

DEFINITION 4. *A* **target system network contraction** *is a set of "vertex contractions" that results in another target system network whose vertices contain one or more "supernodes."*

DEFINITION 5. *A* **vertex contraction** *of vertices $V_C \subseteq V(G)$ produces a graph G' whose vertices and edges are identical to G with the following exception. Vertices V_C are replaced with a single vertex $v \in V(G')$. Edges are modified such that v is adjacent to the union of the neighborhood of the vertices in V_C. The vertex v is called a* **supernode**.

Although target system network contraction allows us to operate on a topology at multiple levels of granularity, it does not give us a mechanism to aggregate attributes associated with contracted vertices. We therefore formalize *vertex attribute contraction*.

DEFINITION 6. *An* **attribute contraction** *updates vertex and edge attributes in a target system network that results from a target system network contraction. Given a target system with an original vertex attribute of $A : V(G) \rightarrow S_V$, the new target system network has a vertex attribute of $A' : V(G') \rightarrow S'_V$, where $V(G')$ are the vertices in the new target system network.*

DEFINITION 7. **Vertex attribute contraction** *is a function $c_v : T_V \rightarrow S_V$ where $T_V \subseteq S_V$ such that T_V is the set of structures associated with the contracted vertices by the vertex attribute. Let $V_C \subseteq V(G)$. Then $T_V = \{s | s = a_v(v), v \in V_C\}$.*

Vertex attribute contractions are practically useful. For example, vertex contraction allows administrators to summarize a property of a subnet by contracting all vertices within that subnet to a supernode. In contrast, multiple relays implementing a single service may not all be directly connected, but it may still be useful to apply vertex attribute contraction to reveal an aggregate property of relays (e.g., an average relay protection threshold).

Network and Attribute Expansion: We define the increase in granularity of a view as a "target system network expansion operation." The expansion operation provides a lower-level view of networked resources than the original view does,

because a single vertex is expanded to multiple vertices and edges. For example, a practitioner who is interested in a file system on a given device could expand the vertex for that device into the file system tree. That could be useful during an audit in looking for deviations from a baseline configuration. Alternatively, a network engineer could expand a vertex that corresponds to a state estimation service to view the underlying devices that support that service (whether deployed on the cloud or in a substation).

DEFINITION 8. *A* **target system network expansion** *expands a single vertex $v \in V(G)$ in the target system graph G to a subgraph G_v. For each edge between v and another vertex v' in $V(G)$, there is at least one edge between a vertex in G_v and v'.*

The above definition allows us to construct a multi-view target system network in which views with higher granularity are built upon views with lower granularity. We note, however, that the topological expansion defined above does not specify how to distribute the value of the attribute among the nodes in G_v. We would therefore have to store all of G_v and the desired attributes prior to expansion. Thus we are not going to consider attribute expansion in this paper unless we have a representation of G_v and its attributes in storage.

4. IMPLEMENTATION AND APPLICATIONS OF CPTL

We now describe the implementation of CPTL and its applications to the NERC CIP audit scenarios. First, we describe our current implementation of the CPTL primitives and operations. We then apply this implementation of CPTL to our audit scenarios and provide an in-depth example of how to use CPTL to present evidence for vulnerability assessment.

4.1 Implementation

We organized the CPTL library according to a set of namespaces so that developers may define and operate upon a variety of different networks at multiple levels of granularity. For example, our CPTL core namespace (`cptl::core`) models and operates on networks purely in terms of topological information. Therefore, the CPTL core library namespace contains procedures for target system network contraction, and target system network expansion. In contrast, our CPTL enterprise namespace (`cptl::enterprise`) incorporates more detailed information about devices on a network via vertex attributes such as IP address. CPTL namespaces are implemented as C++ namespaces within our library.

The remainder of our discussion focuses on primitives and operations within CPTL's core library namespace. We now describe the implementation of our core primitives and operations and provide a small but realistic running example to demonstrate these concepts.

4.1.1 Primitives

CPTL's core primitives—the target system network, vertex attribute, and edge attribute—give practitioners a way to represent networks of cyber-physical assets and to organize disparate sources of data relative to networked assets. The target system network (see DEFINITION 1) as well as its vertex attributes (see DEFINITION 2) and edge attributes (see DEFINITION 3) have two representations within GraphML [8] and the Boost Graph Library (BGL) [46].

The GraphML representation of CPTL core primitives gives practitioners a human-readable, machine-actionable syntax to encode networks of cyber-physical assets. We represent vertex and edge attributes as GraphML vertex and edge attributes. That syntax may then be operated upon by our CPTL library or loaded into a graphical editor, such as yEd [52].

We currently use the GraphML schema to ensure that target system networks are valid GraphML. In the future, however, we plan to use XML namespaces to ensure that a target system network is valid relative to a graph defined within a specific CPTL namespace.

Since we implement CPTL operations using C++ and the Boost Graph Library, we use the GraphML representation of a target system network to initialize an undirected Boost graph whose internal properties correspond to vertex and edge attributes. The intent of the BGL representation is to allow us to implement a variety of analyses that leverage BGL's algorithms.

Figure 1 illustrates an enterprise computer network, a target system network that encodes basic topology, and the corresponding GraphML representation. The computer network topology was inferred with the NetAPT tool from firewall rules very similar to those found in a major IOU [35].

4.1.2 Operations

The intent of CPTL's operations is to allow practitioners to operate upon networks of cyber-physical assets at multiple levels of granularity. In order to address that design requirement, CPTL defines target system network contraction, target system network expansion, and attribute contraction. The CPTL core namespace implements the first two of those operations. Due to space limitations, we will only discuss the implementation of network contraction. Network expansion is implemented using the same approach as network contraction. In Section 4.2, we describe how we implemented an attribute contraction routine to present evidence for our NERC CIP audit scenarios.

Target System Network Contraction: CPTL defines target system network contraction (see DEFINITION 4) as a set of vertex contraction operations (see DEFINITION 5). We implement target system network contraction and vertex contraction operations using a *hierarchy tree*.

DEFINITION 9. *A rooted tree T is a* **hierarchy tree** *of G if $L(T) = V(G)$ where $L(T)$ denotes the set of leaves of T. For clarity, in the remainder of this discussion, we refer to elements of $V(G)$ as* **vertices** *of G and to elements of $V(T)$ as* **nodes** *of T [9].*

Figure 2 shows a hierarchy tree for the graph induced by vertices that correspond to hosts in the target system network of Figure 1. In addition to a node identifier (shown in italicized blue), each node within the hierarchy tree is marked with a vertex index (shown in green) that relates that node to a vertex in the target system graph. Currently, hierarchy trees are specified in GraphML by the user and can be viewed as a hierarchical clustering of vertices in the target system graph. In some cases, however, it may be possible to generate hierarchy trees directly from the target system graph.

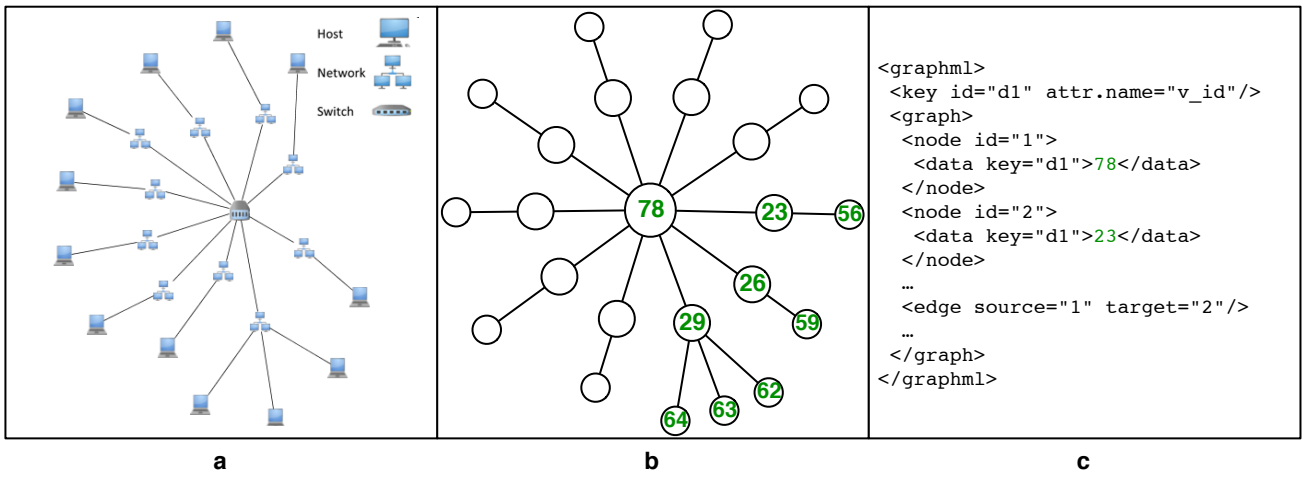

Figure 1: Given the artificial but realistic network from a major IOU in (a), we represent the network as a CPTL target system network in (b) and encode this representation with GraphML in (c)

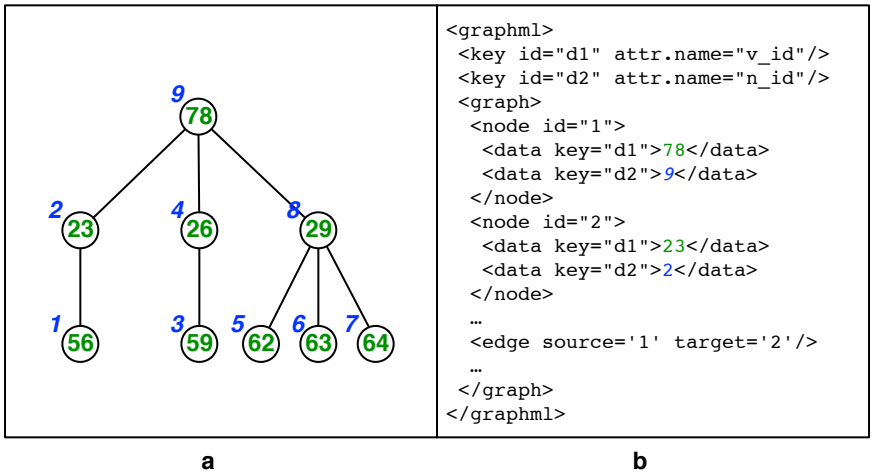

Figure 2: We can specify groupings of cyber-physical assets via a hierarchy tree. Given the target system network from Figure 1(b), we use the hierarchy tree shown in (a) to group hosts by the network to which they belong. We encode the hierarchy tree with GraphML in (b).

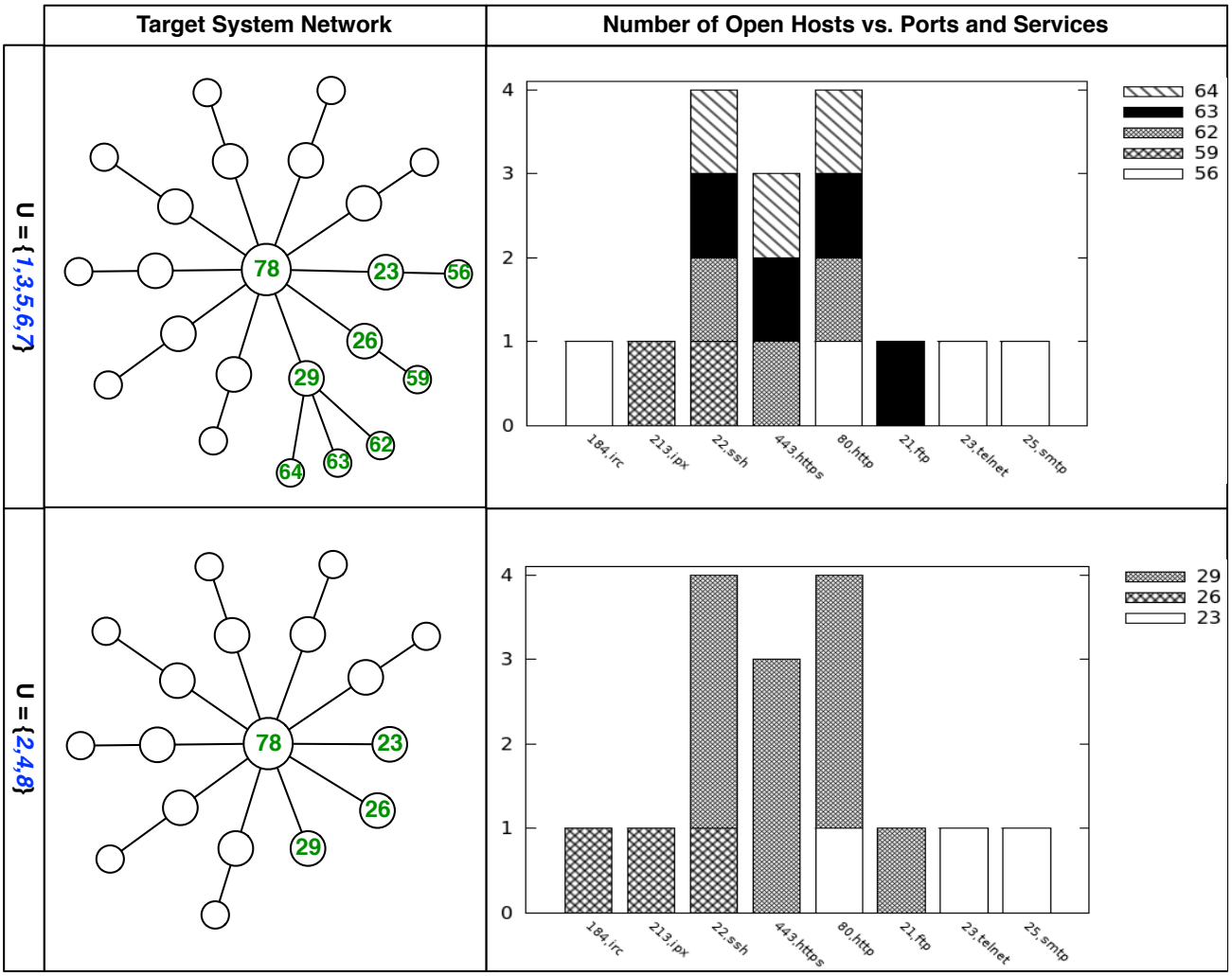

Figure 3: We can use attribute contraction to generate different views of open ports and services within a network. Given the view in the first row, we can generate a target system network to view each host and compute a histogram of ports and services open on hosts 56, 59, 62, 63, and 64. In the second row, we use vertex contraction combined with attribute contraction to generate a collapsed target-system network and histogram that shows open ports and services with respect to networks 23, 29, and 26.

Each node v in $V(T)$ can be used to specify a vertex contraction, because the leaves of the subtree rooted at v define a subset V_C that may be contracted to a supernode. Therefore, CPTL specifies a vertex contraction operation in terms of a node in a hierarchy tree T. For example, the subtree rooted at node 8 in Figure 2 contains leaf nodes 5, 6, and 7. Thus, a vertex contraction of those leaf nodes results in a supernode whose node identifier is 8 and whose vertex index is 29.

Since each node v can be used to specify a vertex contraction, we can use a set of nodes in $U \subset V(T)$ to specify a target system network contraction. That subset U must be a *view*.

Definition 10. *A subset U of $V(T)$ is a* **view** *of G if the set $\{leaves(v)|v \in U\}$ partitions $V(G)$.*

We can perform target system network contractions (and expansions) in terms of a view U. For example, given a target system network graph G, a hierarchy tree T, a view U, and a node within the tree x, we can contract node x by removing the children of x from U and adding x to U. Target system network expansion consists of the same arguments but removes x from U and adds the children of x to U. We then use the naive algorithm by Buschbaum and Westbrook to generate the target system graph induced by the view U [9].

4.2 Applications

We now apply our CPTL operations to the vulnerability assessment audit scenario discussed in Section 2. Specifically, we have implemented a simple attribute aggregation function within a CPTL CIP namespace (`cptl::cip`) that computes a histogram of ports and services over hosts, net-

works, and a subnet. That example is relevant to vulnerability assessment in which documented ports and services must be verified as accurate. We conclude by describing how that and other CPTL-based analyses may be applied to account and configuration management.

Vulnerability Assessment: As mentioned in Section 2, the current approach to vulnerability assessment includes verification that only ports and services described in compliance artifacts are accurate. In Section 3 we observed that the *nmap* tool relates a network device to an IP address or service. We have implemented a simple attribute aggregation operation that allows utilities to compute open ports and services at a variety of levels of granularity by employing attribute contraction. Figure 3 shows how we can use views of our hierarchy tree (see DEFINITION 9) to generate port and service histograms at the host and network levels within our running example from Figures 1 and 2.

In general, we implement attribute contraction (see DEFINITION 6) as a postorder traversal of hierarchy subtrees rooted at each of the nodes in a view U. Just as we use a view to generate a target system graph, so do we use a view to specify the level of granularity at which to aggregate data via attribute contraction. For example, the first row in Figure 3 uses the view $U = \{1, 3, 5, 6, 7\}$ to generate a histogram of the number of hosts that have various ports and services open. In contrast, the second row of the same figure uses the view $U = \{2, 4, 8\}$ to generate a histogram of the number of hosts *per network* that have various ports and services open.

Specifically, we implemented an attribute contraction function to compute a histogram of ports and services associated with each host in a target system network. Given a hierarchy tree T, a view U, and a histogram H that maps a `std::string` to a count, our method performs a postorder traversal of the subtrees rooted at each view node to generate a histogram. For the example, we used artificial data about the services enabled on each host in order to generate our plots.

Account Management: The account management audit scenario requires utilities to inventory, verify, and revoke user accounts and account groups. Current approaches are largely manual; they produce a dated list of accounts and groups, privileges, and people associated with accounts. CPTL will allow practitioners to programatically generate those and other kinds of measures to demonstrate compliance.

Consider the substation system network of Figure 4 and its representation as a target system network (see DEFINITION 1). We can construct a vertex attribute that corresponds to the number of user accounts and/or groups defined on each machine (see DEFINITION 2). In the worst-case scenario, that would be a manual process analogous to the current approaches to demonstrate compliance to the standard. In fact, SEL has begun to deploy devices with embedded SELinux, and embedded devices are beginning to implement the Lightweight Directory Access Protocol (LDAP) [45] and the Remote Access Dial In User Service (RADIUS) [44]. Our attribute contraction method discussed above could use the information provided by those services to generate a histogram of accounts and groups that apply to each of the network devices at a variety of levels of granularity.

Configuration Management: Configuration management is essential to understanding how a substation is configured

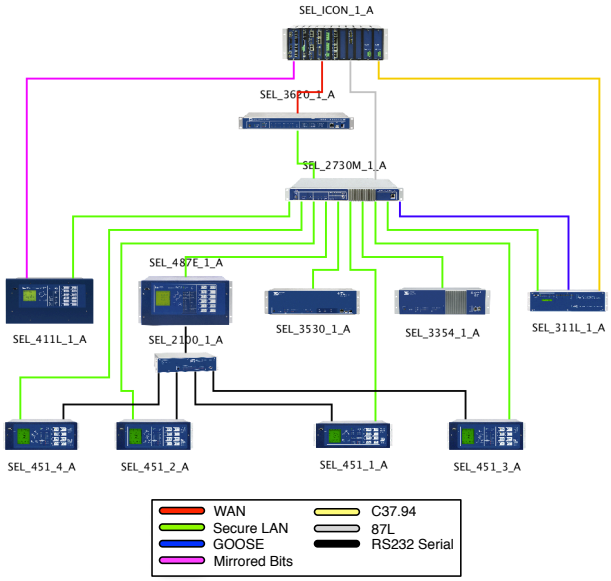

Figure 4: A transmission substation network architecture as defined by SEL best practices [33]. The legend illustrates the properties of links between devices.

and how this configuration evolves. As mentioned before, much of the evidence for compliance is manually collected.

We could use CPTL to understand the OS patch level and versions that run across an entire network. For example, if a utility administrator is interested in the OSes of various devices within a network or subnetwork, he or she could use the attribute contraction method discussed above to generate a histogram of the number of hosts per network that run a given OS. In order to perform the analysis, we would need to collect the OS and version level of each machine.

5. RELATED WORK

We now briefly summarize State-of-the-Practice (SOTP) and State-of-the-Art (SOTA) tools and techniques that are relevant to evaluating compliance within our audit scenarios. We also discuss other network analysis and visualization tools that are used by practitioners and compare their features and operations with CPTL's.

Account Management: Current approaches to user account management may range from manual creation of user accounts to configuration of access, protocols such as LDAP and RADIUS, and Public Key Infrastructure (PKI) certificates. Many of those services may be used to allow practitioners to authenticate only once and subsequently access a variety of devices. As mentioned earlier, LDAP and RADIUS are increasingly being implemented in embedded systems in IEDs. However, PKI is also currently used to authenticate users within substations. For example, the SEL 3354 uses PKI to authenticate users for a variety of devices within substations [33].

Within the academic community, the field of identity management focuses on ways to create and revoke digital identities in a scalable, expressive manner. For example, researchers have proposed a variety of schemes to express an

authentication and authorization policies ranging from XML (e.g. SAML [10] and XACML [22]) to ASN.1 [7, 29, 23]. Furthermore, much research has focused on how to determine trust across organizations through digital identities. For example, PKI bridges exist to mediate trust in the pharmaceutical industry, the U.S. Federal government, the aerospace and defense industry, and higher education [1].

Vulnerability Assessment: During vulnerability assessment, for example, practitioners may use the Advanced Network Toolkit for Assessments and Remote Mapping (ANTFARM) to identify a substation's electronic security perimeter. Like CPTL, ANTFARM is a passive network-mapping application that fuses a variety of information (e.g., configuration files and traffic sniffs) into a network map. However, ANTFARM is primarily a network visualization tool and does not allow practitioners to operate easily upon the maps it produces. In contrast, we designed CPTL to provide a representation that was grounded in theory and could be used to generate analyses that support audit requirements. For industrial control system networks, ANTFARM is a powerful tool from which we could export CPTL for further processing.

Practitioners are also starting to adopt the Network Access Policy Tool (NetAPT). The NetAPT tool performs a security policy analysis to identify global access policy violations. In fact, the tool is built upon ANTFARM and incorporates firewall and OS-based access control rules into a network map. NetAPT demonstrates the power of performing a specialized analysis relative to a network architecture. The intent of CPTL is to enable practitioners to create and disseminate their own analyses that operate on standard representations of cyber-physical systems, complementing what NetAPT does.

Finally, Hierarchical Holographic Models (HMM) and Interoperability Input-Output Models (IMM) are two risk assessment techniques [13] explicitly developed for Industrial Automation Control Systems (IACS). Although both techniques were developed primarily for calculating risk, like CPTL, they model the system at several layers of abstraction and support different operations upon this model. For example, HMMs define specific procedures to combine different views of a system that allow one to capture all sources of risk. IMMs overcome some of the limitations of HMMs on systems with complex interactions among components [13]. In contrast, we designed CPTL to encode the kinds of analyses that practitioners manually perform on networks as well as more formal methods.

Configuration Management: There are a variety of tools that help system administrators better manage their systems. For example, tools such as Puppet [41] and CFEngine [11] give administrators in traditional enterprise networks the ability to declare and deploy machine configurations. CFEngine in particular allows administrators to continually verify and maintain properties, called *promises*, of different machines. Like CFEngine, CPTL is also a language to represent properties of networked resources. In contrast, the focus of the CPTL representation is not on configuration management, but rather on operation of a variety of sources of data relative to a system architecture. If, however, we wanted to use CPTL to instantiate networks, then it might prove useful to translate our representation into the languages used by CFEngine or Puppet. In addition, although CFEngine and Puppet allow one to specify a set of resources running on a device, they do not allow one to operate on relations between those resources, as CPTL does (since it adopts a graph-theoretic approach).

Network administrators also use the Really Awesome New Cisco config Differ (RANCID) to track which parts of a network configuration have changed [42]. Whenever an administrator changes a router configuration, RANCID automatically saves the old configuration, and emails the changes (along with administrator notes) to all other administrators for that network. Although RANCID is helpful, the way in which it records changes prevents practitioners from seeing the big picture.

Utilities also use change management products such as ChangeGear [12], Remedy [43], and TripWire [37]. ChangeGear and Remedy both provide a ticketing system to document changes and are only as good as the change documentation that an administrator writes. TripWire monitors changes to general-purpose computers on a network; TripWire stores hashes of software to be monitored and reports when the stored hash and periodically recomputed hash differ. CPTL rests somewhere between TripWire and ticketing systems; the output of our analyses in Section 4 are higher-level than comparisons of hash values, but lower-level (though possibly better-quality) than manually written change logs. In fact, in previous work, we found that the quality of change logs written by practitioners may be quite poor [51].

Finally, within the network configuration management literature, researchers have proposed several different ways to summarize and measure change. Two approaches to summarizing and measuring change include longitudinal studies [14, 39, 47, 48, 27] and metrics on router configuration files [6]. The intent of CPTL is to give researchers a standard platform through which they can make such cutting-edge techniques practically available by writing modules that operate on a simple, graph-theoretic representation.

Network Analysis and Visualization Tools: Representing computer networks as annotated graphs is not novel. Most network analysis and visualization tools (e.g., Pajek [4], Commetrix [49], and GraphTool [28]) offer clustering and partitioning as operations on the graph. Furthermore, tools already exist to annotate graph structures with data (e.g., JUNG [36], graph-tool [28], and COSBI [50]). CPTL differs from those tools both in how clustering is defined and in its focus on analysis over visualization.

First, the clustering and partitioning operations provided by several of those tools are based on the connectedness of vertices. In other words, one may only contract vertices connected by edges. A few tools, such as JUNG and COSBI, do allow one to group vertices according to vertex properties. (That is equivalent to CPTL's vertex contraction.) None of the tools surveyed thus far, however, provide an operation to aggregate properties of clustered vertices that is equivalent to CPTL's attribute contraction.

Second, the primary intent of CPTL is not to help utilities *visualize* cyber-physical networks, but rather to help them process properties of those assets at multiple levels of abstraction to help them generate evidence for compliance audits. Certainly, many network analysis and visualization tools provide typical graph algorithms, such as shortest path. CPTL, however, uses such algorithms to implement higher-level analyses that utilities can use for internal compliance programs and NERC CIP audits.

6. FUTURE WORK

CPTL sets the stage for utilities and auditors to perform more efficient NERC CIP audits. However, CPTL represents ongoing research and there is more work left to do. We now describe that work in terms of the definition of CPTL (its representation and operations) as well as applications of CPTL.

6.1 CPTL Definition

First, we discuss future work related to the definition and implementation of CPTL.

Representation: In Section 3, we defined CPTL's representation of network architecture as a graph that captures pairwise relations among network resources. However, not all relations between assets may be pairwise. For example, an Intrustion Detection System (IDS) may "cover" many different network resources. We may also want to allow for multiple edges between the same sets of vertices. For example, two devices in a substation may communicate using two different protocols over two distinct logical channels.

We want to investigate simplicial complexes as another way to formalize our target system network that captures higher-dimensional relations. Simplicial complexes have been used to model social dynamics as well as to prove "coverage" of monitors over $\mathbb{R}^2$ in the context of sensor networks [18]. We can leverage those applications of simplicial complexes. Finally, we want to note that the operations we have defined in this paper (topological expansion and contraction) would have analogues if we used simplicial complexes.

Operations: CPTL currently defines network contraction and expansion on a single graph. One possible additional operation would be to merge two graphs. For example, that would allow CPTL to merge two role/object hierarchies of two different Role-Based Access Control (RBAC) policies [21] defined on a network.

We also want to measure the variation of a function (such as relative risk) on each edge. The variation can be formalized via a **graph gradient** [53]. We also want to think of composing different graph gradients to derive a graph gradient that describes the overall relation between entities.

In the future, we may want to extend our notion of attribute contraction. An edge attribute could be used to map edges that represent communication links to the protocols used on that link. Therefore, we could employ edge attribute contraction to operate upon protocols employed by a set of communication links.

Finally, we want practitioners to be able to make logical assertions about the topology and attributes of cyber-physical assets at multiple levels of abstraction over a set of networked resources. Our CPTL-based assertions would also include a way to evaluate the *degree of compliance*, or percentage of resources covered by the associated assertion. That mechanism could enable utilities and auditors to systematically evaluate a network against a suite of compliance tests.

6.2 Applications

Although we focused on power substation audits in Section 2, we envision a wide variety of applications for CPTL, both in electrical power and in other domains. Each of the applications would have its own corresponding namespace within the CPTL library.

Within the power domain, we can apply CPTL across a wide variety of additional power system architectures, such as Advanced Metering Infrastructures (AMI) as well as smart grid architectures that integrate renewable energy sources.

CPTL can also be used to describe enterprise and cloud architectures. In the latter case, we may be able to compare trade-offs between implementing power system services (such as state estimation) in traditional substations or in cloud architectures (as some researchers propose [17]).

We can also apply CPTL to other scenarios, such as attacker modeling and monitoring, that are relevant to multiple domains, including power.

We want to research the extent to which CPTL can be used as a model for security analysis. Unlike any of the current models, our graph-theoretic approach provides a modular formalization that could allow analysts to compare and analyze system architectures relative to a threat model at multiple levels of granularity. We can also make use of assertions to make claims about the resiliency of a system under attack.

As we hinted to earlier in this section, we also want to use CPTL to express relations between network monitors(this includes IDSes). We want to formalize and model the "coverage" of monitoring services over a system architecture. CPTL could also be used to express interactions between monitors as well as to place monitoring services at multiple layers within an architecture.

7. CONCLUSIONS

We designed and implemented CPTL so that utilities and auditors can systematically analyze cyber-physical architectures. Although there are a variety of potential applications of our tool, this paper explains and evaluates our language in the context of NERC CIP audit scenarios. NERC CIP audits, though necessary to ensure that basic and sound security controls are in place, are also costly and time-consuming. We claim that CPTL could form the basis of a standard format by which utilities can express a compliance argument to auditors and thereby save time and money.

More broadly, however, we argue that CPTL could provide a common language by which cyber-physical architectures may be encoded and analyzed. System and network administrators, auditors, and managers all can reason about a cyber-physical architecture by viewing CPTL in a graph viewer. Furthermore, since CPTL is also machine-actionable, it enables researchers both to encode current analyses and to design new analyses that can be vetted by practitioners. CPTL is an admittedly ambitious attempt to explicitly realize the goal of a common, human-readable, machine-actionable language to study, evaluate, and compare analyses of realistic cyber-physical systems.

8. ACKNOWLEDGMENTS

The authors would like to thank Rakesh B. Bobba for his conversations and support. In addition, the authors would like to thank Scott Mix for his feedback, and Jenny Applequist for her suggestions.

The material presented in this paper is based upon work supported by the Department of Energy under Award Number DE-OE0000097. This report was prepared as an account of work sponsored by an agency of the United States Government (DOE). Neither the United States Government nor

any agency therefore, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

9. REFERENCES

[1] 4BF: Four Bridges Forum, 2009.

[2] G. Arnold. NIST Smart Grid Program Overview, March 2012. http://www.nist.gov/smartgrid/upload/Smart_Grid_Program_Review_overview_-_arnold_-_draft1.pdf.

[3] K. Barnes and B. Johnson. National SCADA Test Bed Substation Automation Evaluation Report. Technical Report 15321, Idaho National Laboratory (INL), INL Critical Infrastructure Protection/Resilience Center, Idaho Falls, Idaho, October 2009.

[4] V. Batagelj and A. Mrvar. PAJEK: Program for Large Network Analysis. *Connections*, 21(2):47–57, 1998. http://vlado.fmf.uni-lj.si/pub/networks/pajek/.

[5] D. Batz, J. Brenton, D. Dunn, G. Williams, P. Clark, S. Elwart, E. Goff, B. Harrell, C. Hawk, M. Henrie, H. Kenchington, D. Maughan, L. Kaiser, and D. Norton. Roadmap to Achieve Energy Delivery Systems Cybersecurity. Technical report, Department of Homeland Security, Cyber Security R&D Center, Menlo Park, CA, September 2011.

[6] T. Benson, A. Akella, and D. Maltz. Unraveling the Complexity of Network Management. In *Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation (NSDI '09)*, pages 335–348. USENIX Association, April 2009.

[7] M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized Trust Management. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy (SP 1996)*, pages 164–173. IEEE, May 1996.

[8] U. Brandes, M. Eiglsperger, J. Lerner, and C. Pich. Graph Markup Language (GraphML). In R. Tamassia, editor, *Handbook of Graph Drawing and Visualization (Discrete Mathematics and Its Applications)*. Chapman & Hall/CRC, Boca Raton, FL, 2010.

[9] A. L. Buchsbaum and J. R. Westbrook. Maintaining Hierarchical Graph Views. In *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '00)*, pages 566–575, Philadelphia, PA, USA, 2000. Society for Industrial and Applied Mathematics.

[10] Assertions and Protocols for the OASIS Security Assertion Mark up Language (SAML V2.0), March 2005.

[11] Configuration Management for Agile System Administrators: CFEngine. http://cfengine.com/.

[12] SunView Software: ChangeGear. http://www.sunviewsoftware.com/.

[13] M. Cheminod, L. Durante, and A. Valenzano. Review of Security Issues in Industrial Networks. *IEEE Trans. Industrial Informatics*, 9(1):277–293, 2013.

[14] X. Chen, Z. M. Mao, and J. Van der Merwe. Towards Automated Network Management: Network Operations Using Dynamic Views. In *Proceedings of the 2007 SIGCOMM Workshop on Internet Network Management (INM '07)*, pages 242–247. ACM, August 2007.

[15] Civil Penalty Authority, Title 15: Commerce and Trade. http://www.law.cornell.edu/uscode/text/15.

[16] Common Information Model (CIM): Energy Management. Technical Report 61970-1, International Electrotechnical Commission IEC, December 2007. Available on November 23, 2012 from http://webstore.iec.ch/webstore/webstore.nsf/ArtNum_PK/35316.

[17] G. Dan, R. B. Bobba, G. Gross, and R. H. Campbell. Cloud Computing for the Power Grid: From Service Composition to Assured Clouds. In *Proceedings of the 5th USENIX Workshop on Hot Topics in Cloud Computing*. USENIX Association, June 2013.

[18] V. De Silva and R. Ghrist. Homological Sensor Networks. *Notices of the American Mathematical Society*, 54(1):10–17, 2007.

[19] IEEE Standard for Electric Power Systems Communications: Distributed Network Protocol (DNP3). Technical Report 1815-2012, IEEE, 2012. standards.ieee.org/findstds/standard/1815-2012.html.

[20] Energy Services Provider Interface (ESPI). Technical Report REQ.21, North American Energy Standards Board (NAESB), 2010. http://www.naesb.org/ESPI_Standards.asp.

[21] D. Ferraiolo, J. Cugini, and D. R. Kuhn. Role-Based Access Control (RBAC): Features and Motivations. In *Proceedings of 11th Annual Computer Security Application Conference*, pages 241–248, 1995.

[22] S. Godik and T. Moses. eXtensible Access Control Markup Language (XACML), February 2005.

[23] R. Grimm and T. Hetschold. Security Policies in OSI: Management Experiences from the DeTeBerkom Project BMSec. In *Proceedings of JENC6: Bringing the World to the Desktop*, page unknown. unknown, May 1995.

[24] Guidelines for Smart Grid Cyber Security. Technical Report 7628, National Institute of Standards and Technology (NIST), Gaithersburg, MD, August 2010.

[25] Telecontrol Equipment and Systems. Technical Report 60870-6-503, International Electrotechnical Commission IEC, 2002. http://webstore.iec.ch/preview/info_iec60870-6-503\{ed1.0\}b.pdf.

[26] Substation Automation. Technical Report 61850-1, International Electrotechnical Commission IEC, April 2003. Available on November 23, 2012 from http://webstore.iec.ch/webstore/webstore.nsf/ArtNum_PK/30525.

[27] H. Kim, T. Benson, A. Akella, and N. Feamster. The Evolution of Network Configuration: A Tale of Two Campuses. In *Proceedings of the Internet*

Measurement Conference (IMC 2011), pages 499–512. ACM, November 2011.

[28] V. J. Leung, M. B. Dillencourt, and A. L. Bliss. GraphTool: A Tool for Interactive Design and Manipulation of Graphs and Graph Algorithms. In N. Dean and G. E. Shannon, editors, *Computational Support for Discrete Mathematics: DIMACS Workshop*, volume 15, pages 269–278, March 1992.

[29] S. Mendes and C. Huitema. A New Approach to the X.509 Framework: Allowing a Global Authentication Infrastructure Without a Global Trust Model. In *Proceedings of the 1st Symposium on Network and Distributed System Security (NDSS '95)*, pages 172–189. ISOC, February 1995.

[30] S. Mix. Conversations with Scott Mix, NERC CIP Technical Manager, 2013.

[31] W. H. Moore. National Transportation Statistics. Technical report, U.S. Department of Transportation, April 2012. Section B, Table 11-1. `http://www.bts.gov/publications/national_transportation_statistics/html/table_01_11.html`.

[32] Greatest Engineering Achievements of the 20th Century, 2003. `http://www.greatachievements.org/`.

[33] S. Neary and T. Mullis. Network Architecture Best Practice. Technical report, Schweitzer Engineering Laboratories, July 2012.

[34] NERC CIP v5. Technical report, North American Electric Reliability Corporation, 2013. `www.nerc.com/pa/stand/ReliabilityStandards/`.

[35] NetAPT Access Policy Tool, 2012. `https://www.perform.csl.illinois.edu/netapt/index.html`.

[36] J. O'Madadhain, D. Fisher, P. Smyth, S. White, and Y.-B. Boey. Analysis and Visualization of Network Data Using JUNG. *Journal of Statistical Software*, 10(2):1–35, 2005.

[37] Open Source Tripwire. Retrieved February 3, 2012 from `http://sourceforge.net/projects/tripwire/`.

[38] D. L. Parnas, J. E. Shore, and D. Weiss. Abstract Types Defined as Classes of Variables. *ACM SIGMOD Record*, 8(2):149–154, 1976.

[39] D. Plonka and A. J. Tack. An Analysis of Network Configuration Artifacts. In *Proceedings of The 23rd Conference on Large Installation System Administration (LISA '09)*. USENIX Association, November 2009.

[40] Public Power Annual Directory & Statistical Report. Technical report, American Public Power Association (APPA), May 2012. `http://www.publicpower.org/files/PDFs/USElectricUtilityIndustryStatistics.pdf`.

[41] Puppet—Overview—Puppet Labs. `http://projects.puppetlabs.com/projects/puppet`.

[42] RANCID - Really Awesone New Cisco Config Differ, 2010. Retrieved December 1, 2010 from `http://www.shrubbery.net/rancid/`.

[43] BMC Remedy IT Service Management. Retrieved February 3, 2012 from `http://www.bmc.com/solutions/itsm/it-service-management.html`.

[44] C. Rigney, A. Rubens, W. Simpson, and S. Willens. RFC 2865: Remote Authentication Dial In User Service (RADIUS). Technical report, IETF, June 2000.

[45] J. Sermersheim. RFC 4511: Lightweight Directory Access Protocol (LDAP): The Protocol. Technical report, IETF, June 2006.

[46] J. G. Siek, L.-Q. Lee, and A. Lumsdaine. *The Boost Graph Library: User Guide and Reference Manual*. Addison-Wesley Longman, Boston, MA, USA, 2002.

[47] X. Sun, Y. W. Sung, S. Krothapalli, and S. Rao. A Systematic Approach for Evolving VLAN Designs. In *Proceedings of the 29th IEEE Conference on Computer Communications (INFOCOM 2010)*, pages 1–9. IEEE Computer Society, March 2010.

[48] Y.-w. E. Sung, S. Rao, S. Sen, and S. Leggett. Extracting Network-Wide Correlated Changes from Longitudinal Configuration Data. In *Proceedings of the 10th Passive and Active Measurement Conference (PAM 2009)*, pages 111–121, April 2009.

[49] M. Trier. Commetrix: Project Overview, 2005.

[50] R. Valentini and F. Jordán. CoSBiLab Graph: The Network Analysis Module of CoSBiLab. *Environmental Modelling & Software*, 25(7):886–888, 2010.

[51] G. A. Weaver, N. Foti, S. Bratus, D. Rockmore, and S. W. Smith. Using Hierarchical Change Mining to Manage Network Security Policy Evolution. In *Proceedings of the 11th USENIX Conference on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services (HotICE 2011)*. USENIX Association, March–April 2011.

[52] yWorks. yED, 2013. `http://www.yworks.com/`.

[53] D. Zhou and B. Schölkopf. Regularization on Discrete Spaces. *Lecture Notes in Computer Science (LNCS)*, 3363:361–368, 2005.

[54] B. Zhu, A. Joseph, and S. Sastry. A Taxonomy of Cyber Attacks on SCADA Systems. In *Proceedings of the 2011 International Conference on the Internet of Things (iThings/CPSCom) and the 4th International Conference on Cyber, Physical, and Social Computing*, pages 380–388. IEEE, 2011.