A Digital Twin System for Replaying Cyber Mission Data

Mark Petersen
Computer Science Department
Naval Postgraduate School
Monterey, CA USA
petersen.mark@icloud.com

Alan Shaffer
Computer Science Department
Naval Postgraduate School
Monterey, CA USA
alan.shaffer@nps.edu

Charles Prince
Computer Science Department
Naval Postgraduate School
Monterey, CA USA
cdprince@nps.edu

Gurminder Singh
Computer Science Department
Naval Postgraduate School
Monterey, CA USA
gsingh@nps.edu

***Abstract*—The Persistent Cyber Training Environment (PCTE), developed by the Department of Defense, provides a single training environment for cyberspace operations. PCTE offers a closed network that supports individual training as well as mission rehearsal and post-operation analysis. However, it does not have the ability to replay near real-time events as training scenarios, nor to ingest other network events as scenarios. Replaying cyber mission data on a digital twin network in PCTE would enable more realistic near real-time operator training, as well as supporting development and testing for cyberspace operations. This requires extracting target network specifications from a cyber mission data set. This research developed a prototype tool to extract the network specifications necessary to instantiate a digital twin network within PCTE from cyber mission data. A key contribution of this work is the ability, upon sending recorded network data to the tool, to create a high-fidelity network, and scenario for training purposes. This contribution enables a significant decrease in the time from detection of a problem on a live network to the creation of relevant training scenarios that address the detected problem.**

Keywords—digital twin, high-fidelity virtualization, cyber security, mission data replay, persistent cyber training environment

I. Introduction

Creating a virtual computing environment that is an exact duplicate, or "digital twin," of a physical computing environment can enable cybersecurity practitioners, researchers, and cyber operators to simulate real-world scenarios more realistically in controlled environments [1]. Such a digital twin is an example of a high-fidelity virtualization system, and can be used to replay cyber mission data (CMD) for the purposes of operator training, to practice defensive cyberspace operations (DCO), and to learn new strategies for offensive cyberspace operations (OCO) [2]. An environment that has such capability to host a digital twin system is the Persistent Cyber Training Environment (PCTE), recently developed by the Department of Defense (DOD) [3].

PCTE, developed as a joint force solution, provides a single training environment for cyberspace operations (CO). PCTE offers a closed network for Joint Cyberspace Operations Forces to utilize for training and enables a range of training solutions spanning individual sustainment training to cyber mission rehearsal. Being operational for less than two years, PCTE is still undergoing development to refine its capabilities, and currently cannot replay cyber mission events captured outside of the PCTE environment. Providing the capability to replay real-world cyber missions will enable a PCTE user to conduct more detailed after-action reviews, and to conduct more realistic and relevant training scenarios and post-analysis of CO. Replaying mission data can also introduce the capability to modify specific parameters within the mission data, enabling cost-benefit analysis of various systems and tactics employed during a cyber mission. This paper represents work in progress on automating the replay of CMD based on live network data.

The key contribution of this work is Automated Cyber Operations Mission Data Replay (ACOMDR), a prototype tool to replay cyber mission events in a dynamic environment and to provide enhanced visualization of transpired events within PCTE. ACOMDR is designed to increase a user's situational awareness of the cyber battlespace and integrate with current and future capabilities hosted within PCTE and its specific data repositories.

The rest of this paper is organized into four sections: related work, ACOMDR system design, system implementation and results, and future work.

II. Related Work

A. Digital Twin Networks

A digital twin network can be defined as a "complete virtual description of a physical product that is accurate to both micro and macro levels" [1]. This definition incorporates a broad perspective of the term where a virtual entity, or a digital representation of a physical construct, exists within a virtual environment, and the virtual entity and virtual environment duplicate that of their physical counterparts with the highest fidelity. While the physical environment corresponds to a "real-world" domain, the virtual environment exists within the digital domain. Furthermore, a two-way connection exists between the virtual and physical components that facilitates a twinning, or synchronization, of the two. This provides a continuous feedback cycle allowing for both the physical and virtual environments to interact and sustain accurate mirroring in near real-time. The fidelity of a digital twin is an important concept that can have a variety of impacts on a digital twin system. Here, fidelity is best described as "the number of parameters transferred between the physical and virtual entities, their accuracy, and their level of abstraction" [1]. A high degree of fidelity in a digital twin system is the desired end state; however, the degree of fidelity varies widely depending on the software and techniques used to generate the virtual components of the digital twin system.

2022 Resilience Week (RWS) | 978-1-6654-8819-8 /22/$31.00 USG | DOI: 10.1109/RWS55399.2022.9984020

B. Persistent Cyber Training Environment

The current state of the art in cyberspace training environments offers a range of capabilities and resources to replicate real-world scenarios, heighten situational awareness of the battlefield, and increase training and readiness of personnel through high-fidelity digital twin networks and systems. Key findings from these technologies emphasize the high degree of portability and API of the Java software platform, the value of graphical depictions of network actions for increased situational awareness, and consolidated training resources under a single host system. PCTE has the capacity to offer all these capabilities in a cohesive interoperable environment. Replaying CMD on a simulated operational network requires the establishment of the desired network in a virtual environment. Currently, PCTE offers automated solutions to build and configure virtual networks. There are a variety of interface methods with PCTE's networking tool that include a variety of scripting languages and API tools to process CMD and interface with PCTE.

The U.S. Cyber Command outlined a strategic demand for a cyber-architecture that provides a realistic and adaptable training environment for cyberspace operators to train at the individual, team, and force level [3], [4]. PCTE provides a joint training solution to these shortfalls and is advertised as a tool to "rapidly create 'digital twins' that replicate cyberspace operational environments in a virtualized platform for the Cyber Mission Force to execute realistic training and mission rehearsals" [5].

As stated in a 2019 Government Accountability Office report evaluating training and sustainment of the CMF, the goal of PCTE is to "provide on-demand access" to an environment that has been built to "enhance the quality, quantity, and standardization of phase three (collective) and phase four (sustainment) training and exercise events" [6]. PCTE can be accessed via a web-based client application from anywhere in the world and hosts a variety of applications for training courses, cyber ranges, and configurable virtual environments. It is currently hosted by the U.S. Army Project Manager for Cyber, Test, and Training under the umbrella of the Program Executive Officer for Simulation, Training, and Instrumentation. PCTE is composed of over 150 Remote Compute Storage servers installed at various locations across the U.S., facilitating disaggregate use by the Cyber Mission Force [7]. PCTE is still undergoing development to connect its current capabilities to "real-world physical security assets," such as industrial control systems that are not suitable for emulation [5]. By connecting PCTE to these assets, the DOD will have the ability to create a fully interoperable digital twin environment through PCTE [5].

C. Other Candidate Tools

Other CO tools and technologies were reviewed for this study but were found to have limitations as compared to PCTE. Among these, the Computer Emergency Response Team (CERT) STEPfwd (Simulation Training Exercise Platform) platform was developed as an experience and evaluation platform, while its FedVTE (Federal Virtual Training Environment) was developed for training and skills building [8]. Neither of the CERT platforms displayed the expandability seen in PCTE. As well, the Swedish Defense Research Agency developed the Cyber Range and Training Environment (CRATE) in 2008 as a platform for cyberspace exercises, competitions, and experiments. CRATE has the capability to deploy thousands of VMs in various configurations, and the system is configurable through a VirtualBox interface and, while full automation seems a clear goal of the researchers, human interaction is required through the duration of an exercise [9, 10].

III. System Design Methodology

ACOMDR is designed to be both mission agnostic and data-type agnostic, which renders it interoperable with both OCO and DCO missions using a broad range of software to collect the corresponding CMD. To be effective, an ACOMDR tool embedded within PCTE must work uniformly for current and future offensive, defensive, and training cyber mission environments, however, every cyber mission is inherently unique. Given the high degree of variability that surrounds every mission, it is safe to assume that both the data collection tools, and data types collected will vary. Additionally, it is assumed that the volume of data collected will also significantly vary from mission to mission. The requirements imposed by these assumptions introduce many issues that must be addressed in the design of the system, where a successful resolution of these issues results in an interoperable system that is relevant to cyber mission forces (CMF).

The collection of CMD in ACOMDR is a distributed process that is extremely circumstantial. Thus, it is necessary to establish repositories for collected CMD that can store the various data and data types collected. One such repository currently used by Marine Corps Cyber Forces is Big Data Platform-Cyber Hunt & Analytics Operation System (BDP-CHAOS). BDP-CHAOS is accessible via a web browser, much like PCTE, and is designed to host a broad spectrum of data for analysis and training purposes. To facilitate a non-discriminatory analysis of data, BDP-CHAOS uses a data normalization process during ingestion. The ingestion process normalizes data types by using the Elastic Common Schema (ECS), an open-source data specification that uses a common set of fields in order to normalize event data, while also incorporating the capability to introduce custom fields as necessary.

ACOMDR must have incoming data through the PCTE framework accessing BDP-CHAOS data. The application then parses the data and stores it in a database that categorizes it as data that can be used to create a digital twin network, or data used to create and run events on the digital twin network systems. The ACOMDR application consists of three modules and a SQL database running on a Java platform.

ACOMDR is designed to be hosted from within the PCTE infrastructure and allow users to process collected CMD within the PCTE environment. ACOMDR allows connections to external data repositories, which enables PCTE to process collected mission data for user visualization, training, and future mission rehearsals. Figure 1 provides a high-level overview of the foundational interactions and organization in ACOMDR.

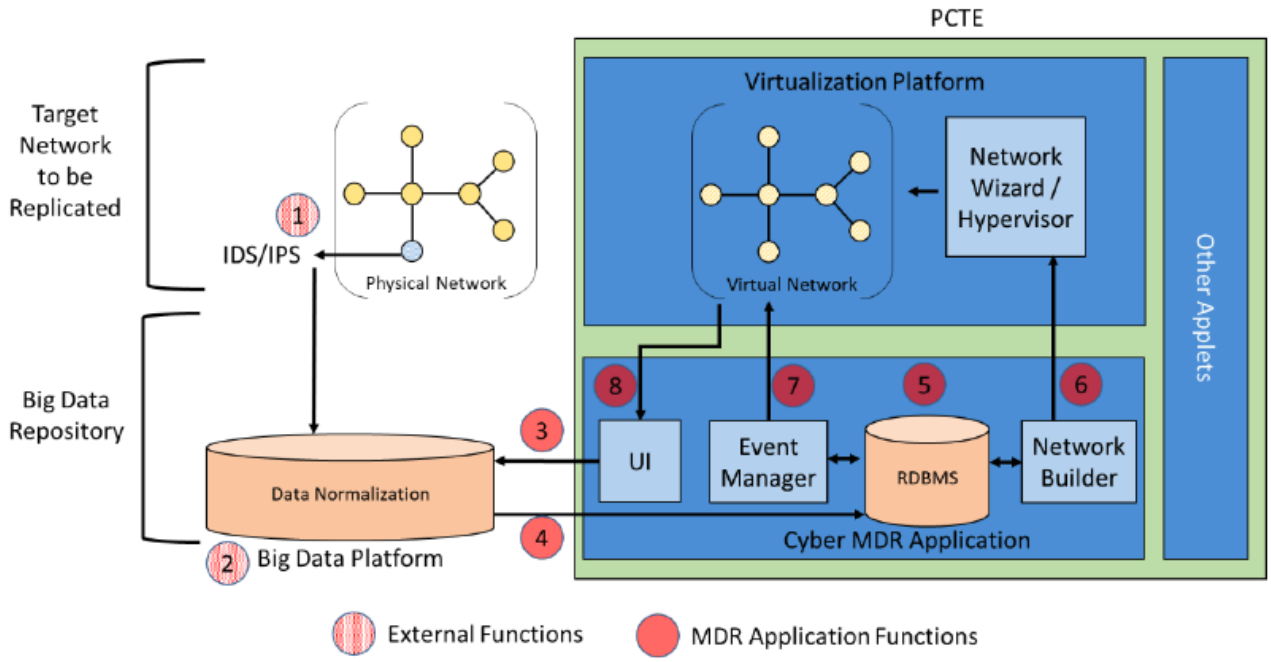

Fig. 1. ACOMDR Application Conceptual Model.

The conceptual model can be broken down into eight descriptive steps, with the Cyber MDR Application and its internal relationships labeled in dark red, and PCTE, along with its external event relationships, labeled in light red.

The first step in the ACOMDR conceptual model is to collect the CMD from the physical network (1). Careful consideration must be placed on how the input data is processed, as the data collection is an external function beyond the control of the ACOMDR application. There are a variety of software applications designed to monitor and capture network traffic, including IDS and IPS platforms such as Snort, Zeek, and Suricata. Each of these applications monitors or captures packets, or a portion of a packet, traversing a network to assimilate data for analysis. The collected network data is stored to BDP-CHAOS (2), which is designed to host a broad spectrum of data for analysis and training purposes. To facilitate a non-discriminatory analysis of data, BDP-CHAOS uses a data normalization process during ingestion.

The next step in the conceptual model depicts the ACOMDR user interface (3). Using the PCTE web portal, a user can navigate to the ACOMDR applet hosted on the PCTE portal and within the PCTE infrastructure. In this step, the user must provide a dataset for processing by either pointing to data housed within the PCTE infrastructure, or through an API call to an external data source. From here, the BDP-CHAOS target data is ingested into ACOMDR to be processed (4). The goal of processing the mission data is to extract the network configurations to build the digital twin network, and to build a chronological event matrix from the CMD.

The relational database used for storing and processing the mission data is depicted next in the model (5). For this we used MySQL relational database management system (RDBMS) since it is open-source, interoperable, and scalable, and it supports both relational and non-relational database structures. Figure 2 depicts a high-level model of the RDBMS schema used in ACOMDR. The *Connection* entity contains the Zeek *uid* attribute as the primary key. As each *uid* attribute describes a single connection, it can be referenced throughout multiple log entries as a point of continuity. We can use this key to quickly summarize the number of events and nodes within the scope of the dataset. The *Events* entity uses a timestamp as the primary key. Zeek structures logs in a manner whereby multiple timestamp records may exist for a single *uid*. Each *Events* record will compile all data relevant to that specific *uid*, *timestamp* pairing. Finally, the *Nodes* entity is used to compile all data that can be correlated with a specific node. In the event of duplicate IP address, and in the absence of a distinguishing MAC address, a duplicate node record will be made. From this database schema, we are able to identify all nodes that participate in a connection. Due to the nature of Zeek data, the ACOMDR application was designed to capture and build a network of endpoint devices. Identification and mapping of midpoint devices, such as routers and switches, requires processing additional forms of mission data.

Having defined the relational database schema in Figure 2, we return to step (4) of the model to describe the Zeek data extraction process in two phases. The raw Zeek data is initially extracted to obtain information on individual servers (*Nodes* in the RDBMS schema), and the connections that run on those servers as seen from external data flows (*Connections* in the RDBMS schema). This data is used to setup each system composing the network. Additionally, we extract data flows,

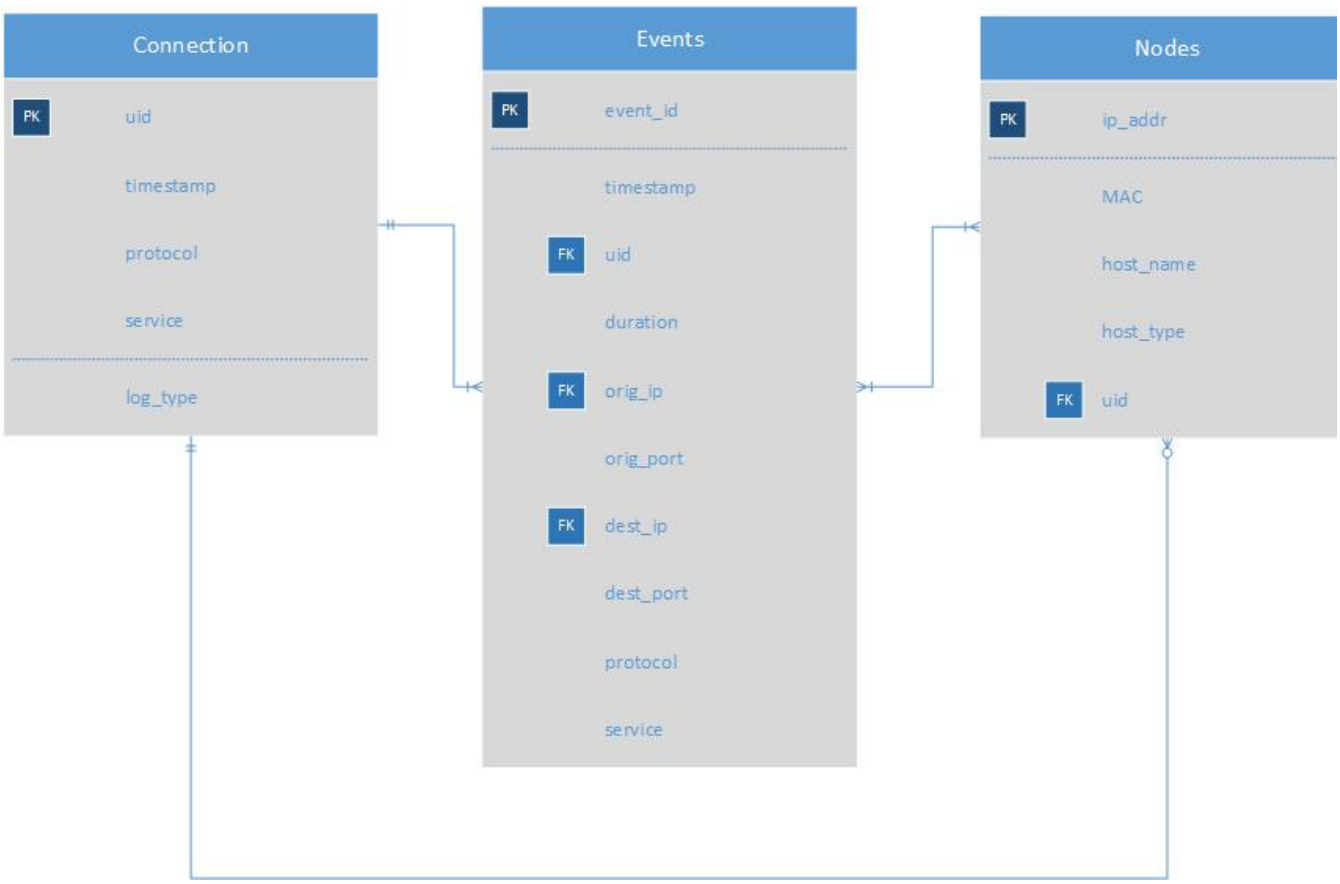

Fig. 2. ACOMDR Relational Database Management System Schema.

protocol, operating system version, etc., from TCP data tracked over time, with part of this timed data extracted from the executed scenario test data (*Events* in the RDBMS schema).

From there, the *Network Builder* module is used to create the digital twin network (6). ACOMDR can accomplish this through API calls to the virtualization infrastructure hosted within PCTE. This module is the primary component that orchestrates the processes necessary to interface with the PCTE hypervisor and ACOMDR relational database. From this model, the program iterates through traffic events on the newly established digital twin network. The *Event Manager* acts as the primary interface with the Puppet platform, which automates this process (7). Puppet is a software configuration management tool already established within the PCTE architecture. The *Event Manager* module leverages the appropriate Puppet module to automate the execution of events in the digital twin network.

Finally, ACOMDR displays the network traffic to the user (8). This step is at the crux of ACOMDR. There are two principal approaches to delivering a comprehensive and interactive perspective to users. The first approach leverages the existing capabilities intrinsic to PCTE. By means of APIs to established PCTE visualization infrastructure, there exists the possibility to display a logical network topology to the user of the digital twin network. From this topological view, it iterates through by highlighting the corresponding nodes and displaying information regarding the type of connection. The second approach leverages the developer's creativity and expands upon the ACOMDR UI module. While this may impose a larger footprint on PCTE infrastructure, it facilitates tailoring the UI to ACOMDR.

The ACOMDR implementation has been partially completed, and further work is needed to complete its full functionality, including instrumentation and interface to control remote systems in the digital twin network to replay events. Bolt, a tool within the Puppet open-source software project, has been identified as a potential tool to provide this functionality [11]. Puppet is already a component of the PCTE architecture, so integration of Bolt is not expected to be challenging. An interface between PCTE, ACOMDR, and Bolt must be created, as well as instrumentation to run the events. Additionally, ACOMDR needs to be instrumented to create the DTN, and interface calls need to be created to work with PCTE to create the DTN.

The existing ACOMDR prototype tool can more effectively create a digital twin network by improving the process it uses to collect the CMD, for example, by refining the Zeek log data extracted by the IDS for events and network data. In addition, the IDS data collected on the original network could be modified to extract more fine grain networking events that will lead to higher fidelity digital twin networks and events orchestrated by ACOMDR.

IV. System Implementation and Results

The previous section described the application model for replaying CMD through eight sequential steps. With our ACOMDR prototype, we implemented the first six of these steps, resulting in the extraction of network specifications for future implementation within PCTE.

The first step to replay CMD is to extract a network specification and use it to instantiate the digital twin network on PCTE. ACOMDR accomplishes this by utilizing two relational databases to process the CMD, producing a "YAML Ain't Markup Language" (YAML) network specification file for subsequent network instantiation.

ACOMDR was developed under the assumption that it will be hosted within the PCTE infrastructure. As such, it is sensible to design and implement the application in a manner that facilitates efficient incorporation within the PCTE architecture. Given that PCTE is an infrastructure composed of many commercial products, we found it prudent to implement a methodology that supports interoperability between many systems and platforms. The current design of PCTE suggests that developing the MDR application in Java was most beneficial for subsequent incorporation into PCTE, given Java's high level of portability.

ACODMR was developed using two disjoint MySQL databases hosted on the same local server, as depicted in Figure 3. The first database contains *raw CMD* in an unaltered state, as ingested into the MySQL database immediately after download from BDP-CHAOS. The second database contains *processed CMD* specifically selected and aggregated by ACOMDR. RDBMS 2 allows the program to quickly characterize two aspects of the corresponding CMD. First, the program needs to produce a network specification of the CMD. This was accomplished using tables that catalog the connections made and network nodes involved. Second, the program needs to replay the events that transpired in the CMD. This was accomplished using an *events* table.

ACOMDR was designed in Eclipse using the MySQL Connector/J, which is the Java Database Connectivity (JDBC) driver that implements the JDBC API. To enable ACOMDR to connect to the local MySQL server, we developed a `MySQLAccess` class within ACOMDR to connect to both CMD databases in order to extract necessary data to populate the ACOMDR database. Within the class we defined methods to process the raw CMD from the first MySQL database. User functionality was provided by a simple command-line interface to function as the main interface for ACOMDR.

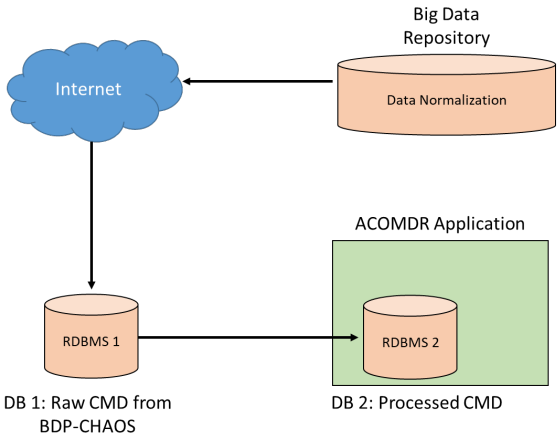

Fig. 3. ACOMDR Relational Database Conceptual Model.

To evaluate ACOMDR we developed a three-point testing scheme to evaluate efficiency and suitability. We first evaluated the time necessary to build the connection, events, and node tables from three increasingly larger datasets. Secondly, we evaluated the time required to build a network specification for varying scopes of data. And finally, we evaluated the suitability of the network specification produced. We did not consider the time necessary to download the CSV data from BDP-CHAOS as we assumed the final version of ACOMDR would query the data via API calls.

We found that network specifications can be generated from IDS Zeek data from BDP-Chaos, and a proof-of-concept tool, ACOMDR, was created to perform this task. We believe that a working system can be created to ingest IDS Zeek logs and use the data to create a representative network and control nodes on that network to simulate IDS incidents, but a working prototype for the system has yet to be created.

We plan to continue to develop a fully finished prototype of ACOMDR. Completing the ACOMDR to Puppet interface and API to replay events and to control nodes, as well as the interface with the PCTE automation tool, will be essential to completing the additional functionality to control the network nodes for attack replay. Additionally, the ability to search the BDP-CHAOS databases will result in increased functionality for the ACOMDR application. Creation of a graphical user interface to display views of digital twin network topology and cyber events over time would also enhance system capability. More work on inferring original network specifications based on Zeek data and translating it to the YAML specification needs to be performed. To support this, improving the Zeek data from the IDS will greatly enhance the tool's capabilities.

Our testing determined that the ACOMDR application can process CMD and produce a network specification document. We were also able to conclude that the prototype application produces a low-fidelity network specification and incurs significant delays in processing CMD due to the large amount of extraneous data contained within the database.

V. Conclusions and Future Work

This work in progress successfully demonstrated a prototype for processing Zeek data logs from a live network to produce a high-fidelity specification of that network. It also demonstrated the ability, upon sending recorded network data to the tool, to establish a training scenario in real time, and to create a higher-fidelity network for training purposes. This has enabled a significant decrease in the time from detection of a problem on a live network to creation of relevant training scenarios that address the detected problem. Following this work, ACOMDR will be extended to incorporate the use of live malware for cyber operations by using the Xen hypervisor with Drakvuf malware analysis tool to create high-fidelity virtualization modules,

adding another valuable method to the study and training of malware for PCTE [12].

Completion of a ACOMDR prototype will require completing the interface between the tool and Puppet, as well as the API to replay events and control nodes. For this, additional work on refining the extraction of network specification from the existing Zeek data to gather better network specifications to refine the process is needed. Other areas for further development of ACOMDR are discussed below.

- Scope of ingestion data. The proof-of-concept ACOMDR was developed using Zeek data, which captures header fields of select packets between end devices and stores them as log files. Using Zeek CMD to build a network specification introduces limitations to the scope of data and fidelity of the network specification. The scope of ingestion data could be expanded to include additional datatypes to research the possibility of increasing the fidelity of the induced network specification and subsequent digital twin network.
- BDP-CHAOS connectors. ACOMDR currently does not have any connectors to external data repositories. Establishing a connector to external data repositories, such as BDP-CHAOS, will significantly increase the capability of ACOMDR. For example, the tool would be much more efficient if it had the capability to query for specific fields of Zeek CMD in BDP-CHOAS, such as connection logs or DHCP logs.
- YAML Processing. ACOMDR currently produces the YAML network specification through a manual formatting process. It would be more efficient and productive to research additional methodologies to producing YAML output files. For example, reading a YAML network specification could allow the program to map Java objects to the YAML attributes. Subsequent programming could leverage the automated mapping to quickly produce a YAML file with the same mapping schema.
- Network Specification Fidelity. The fidelity of the network specification could be significantly increased through additional mapping and queries of the source data. For ACOMDR to make better assumptions, or to decrease the necessity to make assumptions, further processing of the source data needs to be accomplished. Such processing should include making additional correlations between source data fields and node profiles. A practical additional feature will be comparing incoming network specification to existing network specifications and correlating best fit in order to choose an existing pre-built network, saving time and space in the training system and saving time to scenario generation, with the tradeoff of less original network fidelity.
- ACOMDR Database Schema. The current ACOMDR database schema defines the live network Connection, Events, and Nodes tables. This schema could be expanded to better support defining the network specification by creating tables for the various node types, e.g., a table should be created for each of the various node types, such as web servers, mail servers, domain name servers, and user compute nodes.

REFERENCES

[1] D. Jones, C. Snider, A. Nassehi, J. Yon, and B. Hicks, "Characterising the digital twin: a systematic literature review," CIRP J. Manuf. Sci. Technol., vol. 29, pp. 36–52, May 2020 [Online]. Available: https://doi.org/10.1016/j.cirpj.2020.02.002.

[2] Norine, Shaffer, Singh, "Artifact Mitigation in High-Fidelity Hypervisors", Proceedings of the 54th Hawaii International Conference on System Sciences, 2021, URI: https://hdl.handle.net/10125/71466, 978-0-9981331-4-0.

[3] U.S. Army PEO STRI, "Persistent Cyber Training Environment (PCTE)." Accessed Oct. 30, 2020 [Online]. Available: https://www.peostri.army.mil/persistent-cyber-training-environment-pcte.

[4] U.S. Senate. 116th Congress, 1st Session. (2019, Feb. 14). *Statement of General Paul M. Nakasone Commander United States Cyber Command Before the Senate Committee on Armed Services*. [Online]. Available: https://www.armed-services.senate.gov/imo/media/doc/Nakasone_02-14-19.pdf.

[5] A. Kapadia, R. Osborne, and B. Vermillion, "Cyber training architecture, enabling digital twin environments," in I*T2EC*, 2020 [Online]. Available: https://www.itec.co.uk/__media/libraries/draft-abstracts--slides/42---Kapadia-&-Vermillion.pdf.

[6] J. Kirschbaum, "DOD training: U.S. Cyber Command and services should take actions to maintain a trained Cyber Mission Force," Government Accountability Office, Washington, DC, USA, GAO Report No. GAO-19-362, 2019.

[7] M. Schwartz, G. A. Martin, S. Sirigampola, S. Zielinski, and B. Caulkins, "Automated testing of a cyber training environment within an agile development process," in *MODSIM World*, 2020 [Online]. Available: http://www.modsimworld.org/papers/2020/MODSIM_2020_paper_34_.pdf.

[8] M. Baker, "State of cyber workforce development," Carnegie Mellon Univ Software Engineering Institute, Pittsburg, PA, USA, Rep. DM-0000571, 2013 [Online]. Available: https://resources.sei.cmu.edu/asset_files/WhitePaper/2013_019_001_83508.pdf.

[9] Swedish Defence Research Agency, "CRATE - Cyber Range and Training Environment." Accessed Nov. 11, 2020 [Online]. Available: https://www.foi.se/en/foi/resources/crate---cyber-range-and-training-environment.html.

[10] T. Sommestad, "Experimentation on operational cyber security in CRATE," NATO STO-MP-IST-133 Spec. Meet., Copenhagen, Denmark, 2015 [Online]. Available: http://www.sommestad.com/teodor/Filer/Sommestad-2015-Experimentationonoperationalcybersecurityin CRATE.pdf.

[11] Puppet, "Introduction to Puppet." Accessed Jul. 18, 2021 [Online]. Available: https://puppet.com/docs/puppet/7/puppet_overview.html.

[12] T. Lengyel, et el., "Scalability, Fidelity and Stealth in the DRAKVUF Dynamic Malware Analysis System", ACSAC 2014, Dec 8-12, 2014, New Orleans, LA, USA, ACM 978-1-4503-3005-3/14/12.