2021-09

AUTOMATED CYBER OPERATIONS MISSION DATA REPLAY

Petersen, Mark N.

Monterey, CA; Naval Postgraduate School

NAVAL
POSTGRADUATE
SCHOOL

MONTEREY, CALIFORNIA

THESIS

AUTOMATED CYBER OPERATIONS MISSION DATA REPLAY

by

Mark N. Petersen

September 2021

Thesis Advisor:	Alan B. Shaffer
Co-Advisors:	Gurminder Singh
	Charles D. Prince

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC, 20503.			
1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** September 2021	**3. REPORT TYPE AND DATES COVERED** Master's thesis	
4. TITLE AND SUBTITLE AUTOMATED CYBER OPERATIONS MISSION DATA REPLAY			**5. FUNDING NUMBERS**
6. AUTHOR(S) Mark N. Petersen			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.			**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

The Persistent Cyber Training Environment (PCTE) has been developed as the joint force solution to provide a single training environment for cyberspace operations. PCTE offers a closed network for Joint Cyberspace Operations Forces, which provides a range of training solutions from individual sustainment training to mission rehearsal and post-operation analysis. Currently, PCTE does not have the ability to replay previously executed training scenarios or external scenarios. Replaying cyber mission data on a digital twin virtual network within PCTE would support operator training as well as enable development and testing of new strategies for offensive and defensive cyberspace operations. A necessary first step in developing such a tool is to acquire network specifications for a target network, or to extract network specifications from a cyber mission data set. This research developed a program design and proof-of-concept tool, Automated Cyber Operations Mission Data Replay (ACOMDR), to extract a portion of the network specifications necessary to instantiate a digital twin network within PCTE from cyber mission data. From this research, we were able to identify key areas for future work to increase the fidelity of the network specification and replay cyber events within PCTE.

14. SUBJECT TERMS digital twins, virtualization, cyber, security, cybersecurity, Persistent Cyber Training Environment, PCTE, data replay, mission data replay, Automated Cyber Operations Mission Data Replay, ACOMDR			**15. NUMBER OF PAGES** 83
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

AUTOMATED CYBER OPERATIONS MISSION DATA REPLAY

Mark N. Petersen
Captain, United States Marine Corps
BS, San Diego State University, 2012
MS, University of Maryland College Park, 2017

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN CYBER SYSTEMS AND OPERATIONS

from the

NAVAL POSTGRADUATE SCHOOL
September 2021

Approved by: Alan B. Shaffer
Advisor

Gurminder Singh
Co-Advisor

Charles D. Prince
Co-Advisor

Alex Bordetsky
Chair, Department of Information Sciences

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The Persistent Cyber Training Environment (PCTE) has been developed as the joint force solution to provide a single training environment for cyberspace operations. PCTE offers a closed network for Joint Cyberspace Operations Forces, which provides a range of training solutions from individual sustainment training to mission rehearsal and post-operation analysis. Currently, PCTE does not have the ability to replay previously executed training scenarios or external scenarios. Replaying cyber mission data on a digital twin virtual network within PCTE would support operator training as well as enable development and testing of new strategies for offensive and defensive cyberspace operations. A necessary first step in developing such a tool is to acquire network specifications for a target network, or to extract network specifications from a cyber mission data set. This research developed a program design and proof-of-concept tool, Automated Cyber Operations Mission Data Replay (ACOMDR), to extract a portion of the network specifications necessary to instantiate a digital twin network within PCTE from cyber mission data. From this research, we were able to identify key areas for future work to increase the fidelity of the network specification and replay cyber events within PCTE.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ACAS	Assured Compliance Assessment Solution
ACOMDR	automated cyber operation mission data replay
API	application programming interface
BDP	Big Data Platform
CCDCOE	Cooperative Cyber Defense Centre of Excellence
CEC	CRATE Exercise Control
CERT	Computer Emergency Response Team
CLI	command line interface
CMD	cyber mission data
CMF	Cyber Mission Force
CO	cyberspace operations
COCOM	combatant command
COTS	commercial off-the-shelf
CPT	Combat Protection Team
CSV	comma-separated value
CRATE	Cyber Range and Training Environment
DCO	defensive cyberspace operations
DHS	Department of Homeland Security
DISA	Defense Information Systems Agency
DOD	Department of Defense
ECS	Elastic Common Schema
EVE & ADAM	Events Visualization Environment & Advanced Data Aggregation Module
GAO	Government Accountability Office
GHOSTS	(G)eneral HOSTS
HBSS	Host Based Security System
HTML	HyperText Markup Language
IDE	integrated development environment
IDS	intrusion detection software
IP	internet protocol

IPS	intrusion prevention software
JCWA	Joint Cyber Warfighting Architecture
JDBC	Java Database Connectivity
JDK	Java Development Kit
JRSS	Joint Regional Security Stack
JS	Java Script
JVM	Java Virtual Machine
MAC	media access control
MARFORCYBER	Marine Corps Forces Cyberspace Command
MAST	Malicious Activity Simulation Tool
MDR	mission data replay
NATO	North Atlantic Treaty Organization
NIC	network interface card
NPC	non-player character
NPS	Naval Postgraduate School
OCO	offensive cyberspace operations
OS	operating system
OUI	organizationally unique identifier
PCTE	Persistent Cyber Training Environment
PEO-STRI	Program Executive Officer for Simulation, Training and Instrumentation
PHP	PHP: Hypertext Preprocessor
PM CT2	Project Manager for Cyber, Test, and Training
RCS	Remote Computer Storage
RTE	runtime environment
SEI	Software Engineering Institute
STEPfwd	Simulation Training Exercise Platform forward
TCP	transmission control protocol
TOSCA	Topology and Orchestration Specification for Cloud Applications
UI	user interface
UK	United Kingdom
U.S.	United States

USCYBERCOM	United States Cyberspace Command
VM	virtual machine
XML	extensible markup language
YAML	YAML Ain't Markup Language

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank my thesis advisors, Dr. Alan Shaffer, Dr. Gurminder Singh, and Charles Prince for their support and guidance during the development of my thesis.

I would also like to thank Brigadier General Arthur Pasagian, Major Taylor Paul, Captain Daniel Salazar, and Robert Parrish. Your collective willingness and assistance made this project possible.

Most importantly, I would like to thank my family, Neena, Kaia, and Felix. Your love and patience were both greatly needed and appreciated during my time at Naval Postgraduate School.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Creating a virtual computing environment that is an exact duplicate, or "digital twin," of a physical computing environment enables researchers, cyber operators, and trainees to simulate real-world scenarios in controlled environments. Such a digital twin 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). A recently developed environment that has the capability to host a digital twin system is the Persistent Cyber Training Environment (PCTE).

PCTE has been developed as the joint force solution to provide a single training environment for cyberspace operations (CO). The 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 mission rehearsal. Having been operational for approximately one year, the PCTE is currently undergoing further development to refine its available capabilities. Currently, the PCTE does not have the capability to replay cyber mission events captured outside of the PCTE environment. Introducing the capability to replay real-world cyber missions will enable operators to conduct more detailed after-action reviews, and to conduct more realistic and relevant training scenarios and analysis of CO. Replaying mission data can also introduce the capability to modify specific parameters within the mission data, which would allow for cost-benefit analysis of different software, hardware, or tactics experienced during the cyber mission.

The purpose of this research is to begin the development of an embedded tool that will allow users to replay cyber mission events in a dynamic fashion and provide enhanced visualization of transpired events within PCTE. This tool is designed to increase a user's situational awareness of the cyber battlespace and integrate with current and future capabilities hosted within the PCTE and service specific data repositories. The tool will leverage the PCTE's ongoing development to allow for rapid deployment to the joint cyberspace operating forces and future expansion of capabilities.

A. PROBLEM STATEMENT

This study addresses the following research questions:

1. Primary Question

How can the relevant network specifications be extracted from log data to instantiate a digital twin network?

2. Secondary Questions

How can log data be converted to scripts to be executed on a digital twin network?

What are the desirable features of a run-time environment to control the execution of cyber mission scripts?

B. SCOPE

This thesis reviews previous cyberspace training solutions to develop a proof-of-concept tool to extract network specifications from CMD and to set conditions to replay CMD on a digital twin network. The focus of this study is to analyze CMD logs provided by Marine Corps Forces Cyberspace Command (MARFORCYBER) and develop a software environment to process and extract network specifications to instantiate a digital twin network on the PCTE.

The project was divided into two phases. The first phase consisted of researching, collecting, and analyzing CO training environments and CMD logs to extract network configuration data to instantiate a digital twin network within the PCTE infrastructure. The second phase consisted of developing software to process the CMD logs and construct a network specification from the CMD. The tool built in this thesis allows for future development and integration with both existing PCTE capabilities and external data repositories.

C. BENEFITS OF STUDY

One of the foundations of winning battles is maintaining comprehensive situational awareness of the battlespace. Cyberspace is a uniquely challenging battlespace in that cyberspace operations are conducted in abstract digital environments, which introduces new complexities to developing and maintaining situational awareness. This research has begun the development of new capabilities to enhance visualization of CO by instantiating a digital twin of an operational network for replaying the events that transpired during a mission. Since the PCTE is still undergoing regular updates, the possibility of near-term incorporation into the operational PCTE is high. Creating a digital twin network and replaying mission data would support operator training as well as developing and testing new strategies for OCO and DCO.

D. THESIS OUTLINE

1. Chapter II: Background

Chapter II provides a general background to military cyberspace operations. This chapter describes the development and capabilities of the PCTE and covers applicable terms relating to digital twin network environments. Chapter II concludes with a summary of current cyberspace training solutions and current examples of CMD collection tools.

2. Chapter III: Methodology and System Design

Chapter III defines the system design of the proof-of-concept tool developed in this thesis. The system design is broken down into eight steps: data collection, data sourcing, user interface, data processing, database design, network builder module, event manager module, and finally the CMD replay visualization component of the tool.

3. Chapter IV: Test System Implementation and Results

Chapter IV describes the development, testing, and results of the proof-of-concept CMD replay tool. This chapter begins with a step-by-step approach to implement the current application and a description of the programming environment. The chapter concludes with a description of the various tests conducted and an analysis of the output of the tool.

4. Chapter V: Conclusions and Future Work

Chapter V summarizes the work of this thesis, examines the results of the proof-of-concept tool, and makes recommendations for future research and design modifications to increase the fidelity of CMD replay within the PCTE.

II. BACKGROUND

Cyberspace embodies the interconnectedness of global digital technology, ranging from personal computing devices and vast data centers to processers and controllers across the Internet. Cyberspace is unique in that it transcends the physical domain and has both logical and social characteristics that add layers of complexity to CO. Our dependence on cyberspace continues to increase with the growing use of smart devices, the ease of access to the Internet of Things, and the interweaving of digital capabilities in industrial, commercial, government, and private sectors. This reliance on digital technologies is increasing the importance of governing and securing cyberspace.

Military operations have greatly expanded with their integration of and dependence on cyberspace. It is becoming more difficult for commanders to develop situational awareness of the operating environment as cyberspace plays a more significant role across the range of military operations. Commanders must evaluate their unit's cyberspace security and integrity as they are becoming more reliant on cyberspace to control operations in both the physical and cyberspace domains. Commanders are also becoming progressively more vulnerable to cyberspace exploitation, making cyberspace a critical vulnerability.

In this chapter, we assess the growing significance of U.S. CO and the necessity for advanced training and modeling solutions in cyberspace and we review present-day CO training and modeling frameworks.

A. NECESSITY OF CYBERSPACE OPERATIONS

The U.S. Department of Defense (DOD) defined cyberspace as an operational domain in July 2011, marking a shift in national military strategy that placed increased emphasis on CO [1]. As U.S. military operations in cyberspace continued to grow in significance, U.S. Cyber Command (USCYBERCOM) was elevated to the level of a unified combatant command (COCOM) in August 2017 [2]. As a COCOM, USCYBERCOM now plays a larger role in characterizing the cyberspace operating environment. In USCYBERCOM's 2018 command vision, adversaries are credited with molding a “new normal” of operations below the threshold of armed conflict that capitalizes on U.S. constraints and vulnerabilities [3]. USCYBERCOM recognizes the asymmetric nature of CO, which gives disproportionate capabilities to traditionally under-resourced actors. USCYBERCOM’s command vision defines a methodology for how the U.S. plans to combat near-peer and asymmetric threats, highlighting the concepts of superiority through persistence and defending forward, where the metric for success is measured by decreased adversarial aggression and increased options for commands [3]. USCYBERCOM and researchers recognize that nation states, both adversary and ally, continue to increase their investments in CO, signifying a growing role in states’ national security strategies [3], [4].

The three primary missions of USCYBERCOM are defending military networks, supporting the joint force with CO, and defending the nation from cyberspace attacks [5]. Their framework for accomplishing these missions is the Joint Cyber Warfighting Architecture (JCWA), designed to provide “a comprehensive, integrated cyberspace architecture to achieve and sustain the insight, agility, and lethality necessary for maintaining a competitive advantage against near-peer adversaries” [6]. The JCWA is composed of five components:

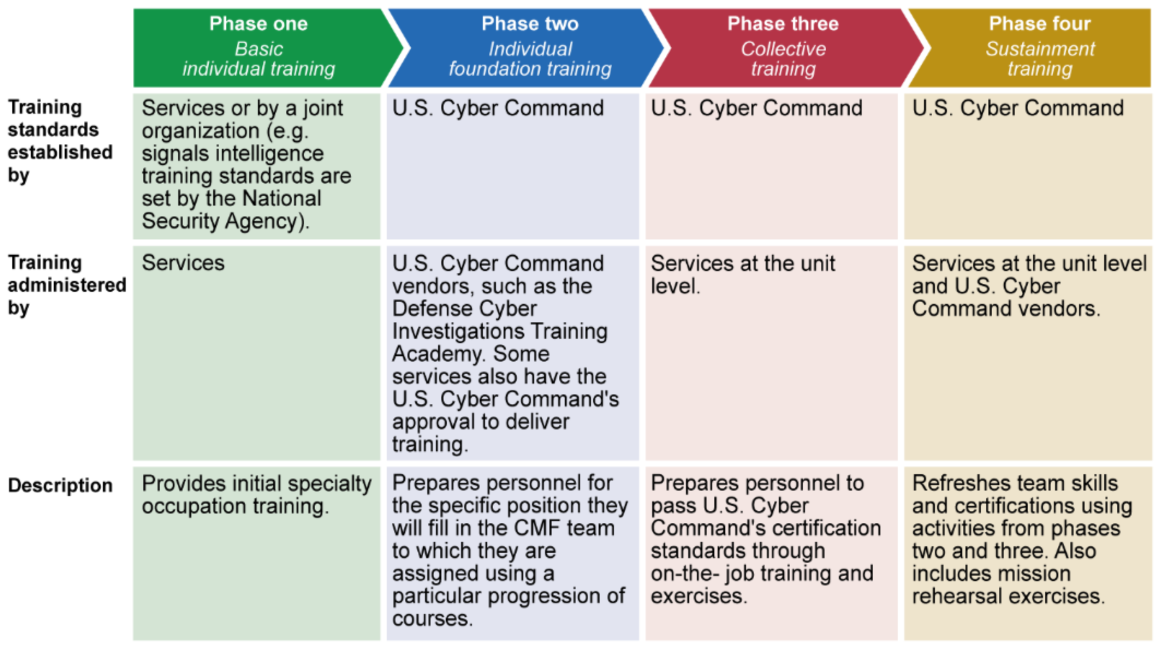
- common firing platforms at [USCYBERCOM’s] four cyber operating locations (each operated and employed by Service cyber components) using a comprehensive suite of cyber tools;
- a “Unified Platform” for integrating and analyzing data from both offensive and defensive cyber operations with intelligence and partners (including the private sector);
- joint command and control mechanisms for situational awareness and battle management at the strategic, operational and tactical levels;
- sensors that support defense of the network and drive operational decisions;
- and a Persistent Cyber Training Environment where teams can train and even rehearse missions under realistic conditions. [7]

The PCTE is designed to be interoperable with all of the JCWA components and to facilitate training and rehearsals of CO using the available tools and capabilities [6]. The purpose of this thesis research was to develop additional capabilities within PCTE to enhance modelling, simulation, and operator training in a virtual environment, a domain identified as being critical to the advancement of CO [4].

B. PERSISTENT CYBER TRAINING ENVIRONMENT

As outlined by USCYBERCOM, there is 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 [6], [7]. Previously, service components operated independently and provided disjoint training solutions that often resulted in training scenarios, data, or resources that were redundant between services or did not persist between training evolutions. 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 (CMF)] to execute realistic training and mission rehearsals” [8].

PCTE is addressed in a Government Accountability Office (GAO) report evaluating the training and sustainment of the CMF, which states that 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” [9]. Four training phases, depicted in Figure 1, characterize the sequencing of CMF training. Phase one is largely generic, being established and administered by individual services, and only provides general military knowledge. Phases two through four were developed by USCYBERCOM to specifically develop the desired skillsets of a CMF operator, reinforcing joint training concepts across the range of capabilities in the cyberspace domain.



	Phase one *Basic individual training*	Phase two *Individual foundation training*	Phase three *Collective training*	Phase four *Sustainment training*
Training standards established by	Services or by a joint organization (e.g. signals intelligence training standards are set by the National Security Agency).	U.S. Cyber Command	U.S. Cyber Command	U.S. Cyber Command
Training administered by	Services	U.S. Cyber Command vendors, such as the Defense Cyber Investigations Training Academy. Some services also have the U.S. Cyber Command's approval to deliver training.	Services at the unit level.	Services at the unit level and U.S. Cyber Command vendors.
Description	Provides initial specialty occupation training.	Prepares personnel for the specific position they will fill in the CMF team to which they are assigned using a particular progression of courses.	Prepares personnel to pass U.S. Cyber Command's certification standards through on-the- job training and exercises.	Refreshes team skills and certifications using activities from phases two and three. Also includes mission rehearsal exercises.

Figure 1. Cyber Mission Force training model phases. Source: [9].

The development of PCTE is ongoing as designers have adopted the agile software development process with new versions released approximately every six months [10]. While this process means PCTE is undergoing frequent changes, it allows for regular user feedback to be incorporated into future releases. Given the volatility of CO, the agile development process allows for a high degree of adaptability and relevancy to the evolving cyberspace environment.

PCTE is accessed via a web application from anywhere in the world and hosts a variety of web-based applications consisting of training courses, cyber ranges, and configurable virtual environments. It is currently hosted by the Army Project Manager for Cyber, Test, and Training (PM CT2) under the umbrella of the Program Executive Officer for Simulation, Training and Instrumentation (PEO-STRI). PCTE is composed of over 150 virtual machines (VM) and Remote Compute Storage (RCS) servers installed at various locations across the U.S., facilitating disaggregate use by the CMF [10].

Traditional cyber training environments have largely relied on cyber ranges for research, development, and exercises. Cyber ranges are designed to simulate networks and systems in an isolated and controlled environment; however, they potentially lack the ability to integrate function as a "complex cyber-physical environment with unexpected dependencies and consequences" [11]. This novel concept has been defined as a Cyber Arena, an environment that facilitates extensive modeling of the cyber ecosystem, and is composed of the following requirements:

1. realism;
2. isolated and controlled environment;
3. Internet simulation;
4. user and network traffic generation;
5. attack execution and simulation;
6. organizations' infrastructures;
7. collaboration;
8. and planning, executing, monitoring, and analyzing [11].

PCTE is currently undergoing development using an "evolutionary architecture" to connect its current capabilities to "real world physical security assets," such as industrial control systems, that are not suitable for emulation [8]. By connecting PCTE to these assets, the DOD will have the ability to create a fully interoperable digital twin environment through PCTE and created a Cyber Arena [8].

C. DIGITAL TWIN

Although the *digital twin* concept was first introduced in the early 2010s, there is a degree of ambiguity that is still associated with the term [12]. For this thesis, we define the term digital twin as well as important concepts surrounding its use.

1. Definition

A digital twin can be defined as a "complete virtual description of a physical product that is accurate to both micro and macro level" [13]. 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 [13]. Furthermore, there is a two-way connection between the virtual and physical components that facilitates a twinning, or synchronization, process.

This provides a continuous feedback cycle that allows for both the physical and virtual environments to interact and sustain accurate mirroring in near real-time.

2. Fidelity

The fidelity of a digital twin is an important concept that can have a variety of impacts on a digital twin system. Fidelity is best described as "the number of parameters transferred between the physical and virtual entities, their accuracy, and their level of abstraction" [13]. 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.

3. Emulation vs. Simulation

Virtualization is "the application of the layering principle through enforced modularity, whereby the exposed virtual resource is identical to the underlying physical resource being virtualized" [14]. Virtualization is the result of multiplexing, aggregation, and emulation, the latter defined as a "level of indirection in software to expose a virtual resource or device that corresponds to a physical device" [14].

A virtual machine is "a complete compute environment with its own isolated processing capabilities, memory, and communication channels" [14]. A virtual machine is characterized as one of three disjoint abstractions: a language-based virtual machine such as Java Virtual Machine (JVM) or Microsoft Common Language Runtime; a lightweight virtual machine such as Docker or Denali; or a system-level virtual machine that will run as a fully isolated instance [14]. System-level virtual machines are run on either a hypervisor or a machine simulator platform, where machine simulation is defined as being "implemented as a normal user-level application, with the goal of providing an accurate simulation of the virtualized architecture" [14]. It is important to identify the distinction between simulation and emulation with respect to virtualization, as the different methodologies can have a significant impact on modeling speed and system design.

D. CURRENT CYBER OPERATIONS TRAINING TECHNOLOGIES AND LIMITATIONS

The tools available to train and evaluate cyber operators have evolved along with the expansion of this domain. There is a range of private and government training solutions and learning environments, which span the range from traditional computer-based training to immersive and adaptive digital environments. This section aims to summarize the current state of training solutions offered for cyberspace professionals.

1. CERT

The Computer Emergency Response Team (CERT), a division under the Software Engineering Institute (SEI) at Carnegie Mellon University, is a federally funded research and development center that, through public and private sponsorship, seeks to advance cyberspace security, training, and analysis capabilities. Investigators at CERT have implemented a continuous four-phase workforce development model designed to improve upon traditional classroom-based training models. The four phases – Knowledge Building, Skill Building, Experience Building, and Evaluation – leverage virtual environments to deliver adaptable and relevant training scenarios [15]. CERT efforts to develop simulation and training capabilities for the purpose of developing the cyberspace workforce have resulted in two web-based platforms: the STEPfwd (Simulation Training Exercise Platform) platform, formerly known as XNET, developed as an experience and evaluation platform; and FedVTE (Federal Virtual Training Environment), developed for training and skills building [16].

The STEPfwd platform is capable of running network simulations with thousands of nodes and is designed to simulate a real-world environment [16]. STEPfwd is designed to support phases three (Experience Building) and four (Evaluation) of the CERT workforce development model [16]. Researchers at CERT are continuing to pursue increased realism within training and simulation platforms, as demonstrated by the development and implementation of CERT's GHOSTS (General Hosts) framework. The GHOSTS framework produces network traffic from multiple instances of simulated-human non-player characters (NPCs), each with its own unique cyberspace related goal

[17]. The GHOSTS framework was deployed on STEPfwd in a case study with three Cyber Protection Teams (CPT) between December of 2017 and April of 2018, and demonstrated the correlation between increased realism and increased training value [17].

The FedVTE platform is a training and education platform that delivers cyber-based training courses and labs in information assurance and cybersecurity, and is advertised as a cost saving system, providing convenient access to federal employees to requisite training [16], [18]. It was developed by SEI in 2005 to resolve DOD information security training and capability shortcomings and is currently managed by the Department of Homeland Security (DHS) with support from Defense Information Systems Agency (DISA) [16]. FedVTE is designed to support phases one (Knowledge Building), two (Skill Building), and three (Experience Building) of the CERT workforce development model [16].

2. MAST

The Malicious Activity Simulation Tool (MAST) is a system developed by researchers at Naval Postgraduate School (NPS) to simulate malware on military networks for network operator training. MAST is not a standalone training environment, but is a program designed to be deployed on a target network. MAST was developed using a modular configuration composed of the Scenario Execution Server, the Scenario Generation Server, and the MAST client [19]. MAST is structured as a client-server model, where clients use a Java program to run modules at the direction of the Scenario Execution Server [19], [20]. A custom scenario scripting language and custom interpreter for the Scenario Execution Server was developed to facilitate continued remote development and deployment of scenarios [19].

MAST was designed to deploy on any network configuration by building in a layer of abstraction between client workstations and the authored scenario, which removes the necessity for mapping the network [19]. As a training tool, MAST leverages real-world networks to deploy scenarios that test network operators' skill and knowledge, workstation configurations, such as antivirus settings, and network vulnerabilities, such as port scanning. MAST is currently configured to facilitate defensive training against simulated malware, termed *simware,* however the modularity of MAST introduces many possibilities for developing DCO or OCO scenarios.

3. UK Network Attack Simulation Environment

Researchers from the United Kingdom (UK) developed a tool in 2017 that creates a configurable emulated network environment for DCO. Their (unnamed) tool generates a user-defined network of VMs using Netkit and replays network traffic to obfuscate malicious network traffic in training exercises [21]. Netkit is an open source network emulator utilizing a kernel derived from the Debian Linux Operating System (OS), which uses start-up scripts to configure the network interfaces and the lab environment.

The UK researchers captured a range of network traffic while connected to the Internet using *tcpdump*, a command line packet analyzer. From this capture, they generated a categorized directional repository of all packets in PCAP format [21]. Using *TCPReplay*, an open source suite to edit and replay captured transmission control protocol (TCP) traffic, the researchers modified the PCAP files to make it appear that their origin was legitimate, and then replayed them at random using bash scripts [21].

The UK researchers developed a Python program that interacts with a user via a command line interface (CLI) to configure the network and subsequently interact with Netkit. While the authors do not evaluate the efficacy of their system in a DCO or OCO context, the system was designed for that specific purpose. Testing and evaluating the functionality of their tool was performed and deemed successful [21].

4. CRATE

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 is composed of approximately 800 servers and has the capability to deploy thousands of VMs in various configurations [22]. Experimenters have implemented the open source VirtualBox hypervisor, developed by OraclePrior, on CRATE's servers to manage scenario development and execution [23]. The system is configurable through the

VirtualBox interface and, while full automation is a clear goal of the researchers, human interaction is required through the duration of the exercise [23].

To enhance the training value of CRATE, in 2015, researchers developed and tested the CRATE Exercise Control (CEC) tool, an exercise management and support tool to enhance planning, execution, and evaluation of cyber defense exercises by interfacing with CRATE [24]. CEC interfaces with CRATE through reports submitted by players in CRATE, which in turn create new event threads within CEC; additionally, exercise administrators correlate reports to injects that serve to document and generate new event actions [24]. The developers of CEC emphasize the value of exercise- and experience-based learning, which they attribute to enhanced learning outcomes.

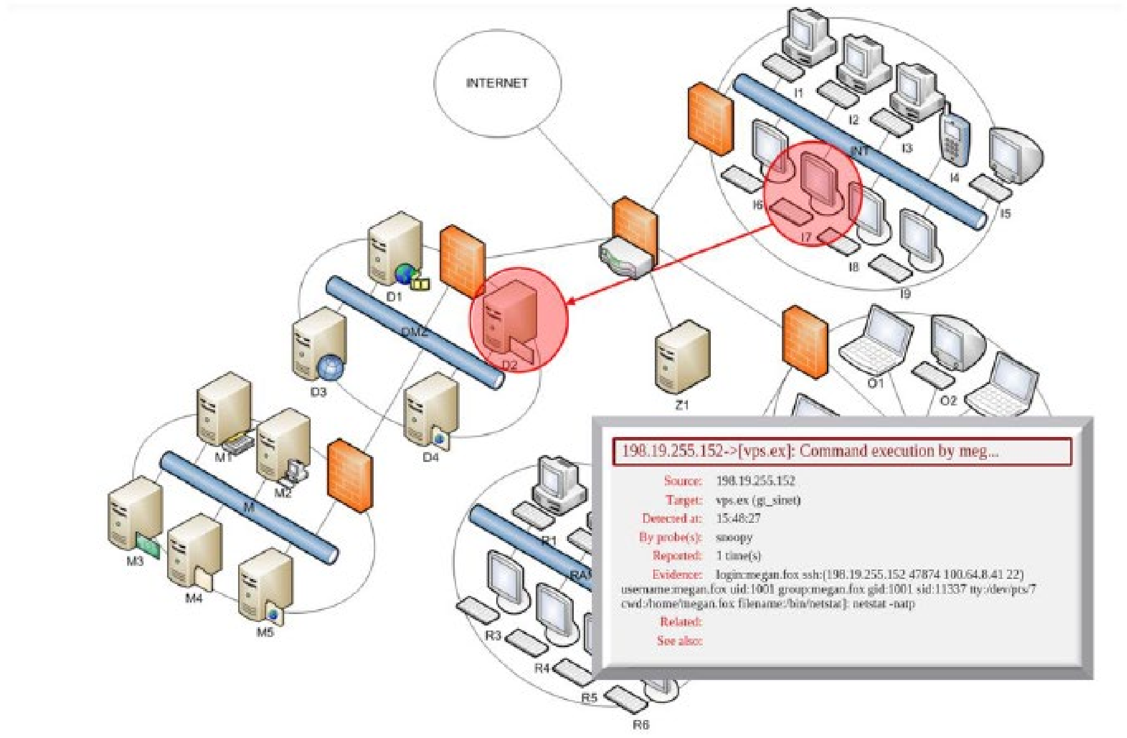
5. EVE and ADAM

European researchers affiliated with North Atlantic Treaty Organization (NATO) Cooperative Cyber Defense Centre of Excellence (CCDCOE) have developed an application designed to improve situational awareness through visualization of the network topology and associated security alerts. The application, Events Visualization Environment (EVE), and a corresponding internal events aggregator module, the Advanced Data Aggregation Module (ADAM), were designed for and tested in two NATO CCDCOE exercises conducted in 2018 [25]. While not a standalone training environment, EVE and ADAM serve together as a toolset to visualize the configuration of networks from sensors composed of intrusion detection and protection systems, system logs monitoring tools, and deception techniques and tools [25].

The EVE application follows a modular configuration composed of five modules:

- the Input Module, which receives incoming data from network sensors;
- The ADAM module, which acts as an event correlator between events and alerts;
- The Network Definition Module, which stores a static map of the network configured before running EVE;
- The Graphical Interface Module, which displays the network map and alerts to users;
- The Internal Control Module, which listens, transmits events to ADAM, maintains the event-alert correlation database, and transmits alerts to the Graphical Interface Module [25].

The EVE application was developed using a Java Web Application, which allows for dynamic interaction with users through a static website, and capitalizes on HyperText Markup Language (HTML) version 5 to push alerts to the JavaScript application programming interface (API). Figure 2 depicts a notional representation of the EVE application depicting an alert of an attack on a network map. Once an alert is received, circles and arrows are drawn depicting the source and target of an attack as well as an informative popup. In Figure 2, node D2 is attacked by node 17 and the corresponding informative popup is displayed.



```
198.19.255.152->[vps.ex]: Command execution by meg...
       Source:  198.19.255.152
       Target:  vps.ex (gi_sinet)
  Detected at:  15:48:27
  By probe(s):  snoopy
     Reported:  1 time(s)
     Evidence:  login:megan.fox ssh:(198.19.255.152 47874 100.64.8.41 22)
username:megan.fox uid:1001 group:megan.fox gid:1001 sid:11337 tty:/dev/pts/7
cwd:/home/megan.fox filename:/bin/netstat]: netstat -natp
      Related:
     See also:
```

Figure 2. Simplified representation of EVE user interface. Source: [25].

The EVE and ADAM applications underpin the importance of visualization in CO to increase situational awareness. The authors recognize further improvements to EVE and ADAM could be made to introduce more detail in alerts and the resulting visualization. Additional concerns addressing the network topologies were also addressed by the researchers.

6. Comparison of CO Technologies

In this section, we have compared five different cyber training environments that have varying degrees of suitability for the training environments set forth by USCYBERCOM. Table 1 provides an evaluation and summary of the five cyber training environments relative to USCYBERCOM's requirements of on-demand access, Phase III enhancement, and phase four training and exercise events.

Table 1. Summary of cyber training environments

Environment	On-Demand Access	Training Standards	Phase III Enhancement	Phase IV Enhancement	System Interoperability	Network Configurability
CERT	Moderate. FedVTE and STEPfwd are web-based platforms; some evaluations incorporated hands-on labs.	DoD	High. FedVTE was designed to meet DoD standards through labs and classes.	Moderate. STEPfwd can incorporates orchestrated NPC's provide a higher degree of real-time interaction in cyber range.	Low. Exercises conducted in disjoint ranges (CERT Private Cyber Training Cloud, STEPfwd, and VTE).	Unknown. Cyber ranges conducted in proprietary range environment.
MAST	Low. Standalone software not accessible by web interface.	N/A	None.	Low. Program allows for test and evaluation of users and networks against specific malware.	Low. Program can run on multiple operating systems, but no API for external connections.	None. Program runs on live DoD network. Suitability for digital twin network unknown.
UK Network Attack Simulator	Low. Standalone software not accessible by web interface.	UK	Moderate. Similar intent, but program not designed to meet DoD standards.	Low. Program does not offer multiple user interactions.	Unknown. System appears to be prototype version / proof-of-concept; likely no API.	High. Program captures network traffic with Netkit and offers users high degree of configurability.
CRATE	Unknown. Web-browser interfaces for server configuration; unknown .	Swedish	Moderate. Similar intent, but program not designed to meet DoD standards.	High. Similar intent, but program not designed to meet DoD standards.	Unknown. System does not appear to offer interfacing with other capabilities.	High. Capable of deploying thousands of VM's in a highly configurable environment.
EVE and ADAM	Low. Standalone software not accessible by web interface.	NATO / CCDCOE	Moderate. Similar intent, but program not designed to meet DoD standards.	High. Similar intent, but program not designed to meet DoD standards.	Low. EVE and ADAM are disjoint tools; unknown interoperability with other systems.	High. Capable of deploying thousands of VM's in a highly configurable environment.

E. CYBER MISSION DATA OVERVIEW

The term *cyber mission data* is inherently vague. One could generally classify all data collected during a cyber mission as being CMD, but this fails to characterize the type, scope, and purpose of the data. In general, CMD is typically structured data as it is commonly text output from software packages that are designed to monitor and capture varying degrees of network traffic. While certain CMD may be more unstructured in nature, such as tools that assemble packets into picture, audio, or video files, we focused on software packages that produce structured CMD. These software packages are commonly advertised as intrusion detection software (IDS) and intrusion prevention software (IPS). In this section, we introduce three software solutions that are currently used in the DOD.

1. Assured Compliance Assessment Solution

The Assured Compliance Assessment Solution (ACAS) is a DISA contract awarded to Perspecta (formerly HP Enterprise Services) and Tenable in 2012 to assess DOD networks against DOD standards while identifying known vulnerabilities [26]. According to a 2018 Tenable Whitepaper, the current ACAS solution includes the following commercial off-the-shelf (COTS) products :

- Tenable.sc (formerly SecurityCenter) for vulnerability, compliance, and event management; Tenable.sc functions as a client in a client server model in conjunction with Nessus Scanners,
- Nessus Scanners function as a server that is controlled by the Tenable.sc platform,
- Nessus Network Monitor (formerly Passive Vulnerability Scanner) is a packet level network discovery and vulnerability analysis tool designed to determine network topology, services, and vulnerabilities,
- Nessus Agents allow for increased client deployment to enable expanded scan coverage of assets,

- Log Correlation Engine aggregates, normalizes, correlates, and analyzes enterprise log data by passing them, as well as IPS/IDS events, to Tenable.sc [26], [27].

ACAS is primarily structured in a defensive posture designed to identify threats through passive monitoring and active scanning.

2. Host Based Security System

The Host Based Security System (HBSS) is COTS software suite developed by McAfee that is utilized by the DOD. For HBSS to function, DOD workstations must run a host IPS that also connects with and allows system administrators to monitor threat alerts through the ePolicy Orchestrator. The HBSS suite also allows for the establishment of a performance baseline, enabling administrators to detect anomalous behavior as well as known malicious program signatures [28].

3. Joint Regional Security Stack

The Joint Regional Security Stack (JRSS) is a DOD solution intended to provide a broad range of network security capabilities by means of centralizing and standardizing network security architectures [29]. These capabilities are sourced from 36 different Original Equipment Manufacturers that offer services such as firewalls, IPS and IDS, enterprise management, and virtual routing and forwarding [29]. Relevant software solutions to capture CMD exist within the JRSS, such as Zeek.

Zeek, formerly Bro until 2018, is a software package that is better described as a sensor vice an IPS or IDS [30]. Zeek observes network traffic and produces many different descriptive, and customizable, log files for subsequent analysis. Zeek is often employed in a manner that does not capture the payload of a packet, but instead extracts relevant information from packet headers to process into log files. Zeek logs such as *conn.log*, *http.log*, and *dhcp.log* are of particular value as they provide detailed information on established connections and network specifications. Zeek log file types and descriptions are summarized in Appendix A.

F. SUMMARY

In this chapter, we reviewed the current state of CO in the U.S., the development and implementation of the PCTE, and the current state of the art in cyberspace training solutions. 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 twins. 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.

The PCTE has the capacity to offer all of 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. However, there are a variety of mechanisms available to interface with PCTE's networking tool. Such mechanisms include a variety of scripting languages and API tools available to process the CMD and interface with PCTE. The next chapter discusses the requirements and methodology to develop a runtime environment that will interface with PCTE to automate CMD replay.

THIS PAGE INTENTIONALLY LEFT BLANK

III. METHODOLOGY AND SYSTEM DESIGN

A. OVERVIEW

One of the principal goals of our automated cyber operation mission data replay (ACOMDR) application is to be both mission agnostic and data-type agnostic, which renders the ACOMDR application an interoperable tool with applicability to 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, we can safely assume that both the data collection tools and data types collected will vary. Additionally, we can assume 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 the entire CMF.

In this chapter, we outline the design of our ACOMDR application that replays network traffic in a digital twin environment. Key components of the application include data structure and processing, database management, an event manager, virtualization infrastructure, and runtime environment (RTE). In the following sections, we discuss each of these components in detail. Finally, we will conclude by outlining the key requirements necessary to replay cyber mission data (CMD) in a simulated network environment.

B. ACOMDR APPLICATION DESIGN

The conceptual model of the ACOMDR application, depicted in Figure 3, provides a high-level overview of the foundational interactions and organization of the application. The intent is to host the ACOMDR application within the PCTE infrastructure, rendering the ACOMDR application available to all the CMF. The conceptual model can be broken down into eight descriptive steps that are discussed in detail.

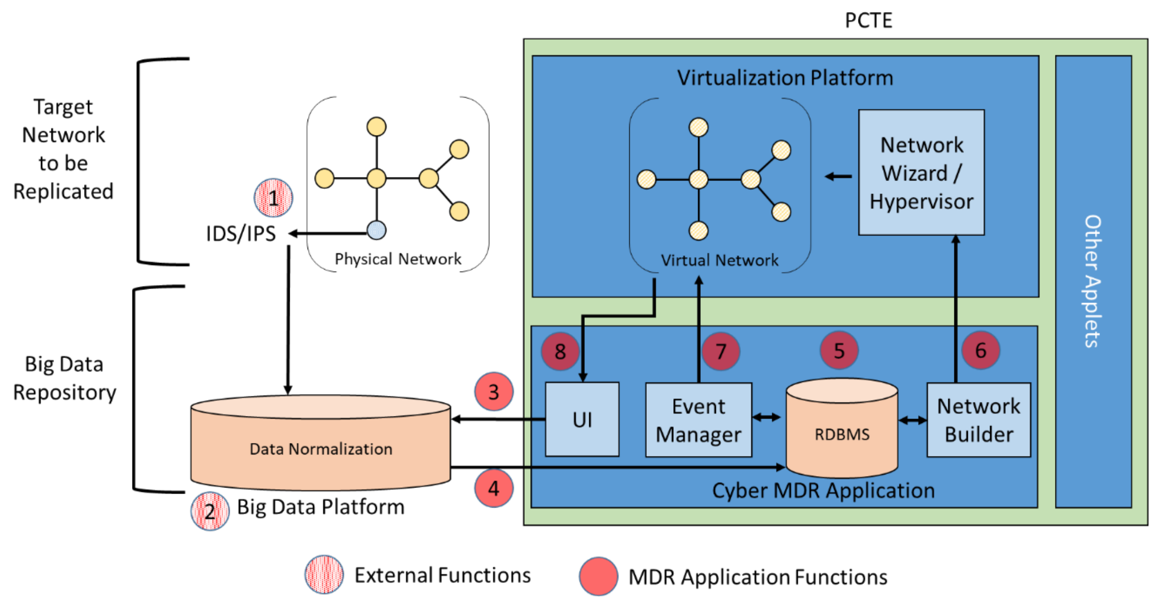


Figure 3. ACOMDR application conceptual model

1. Cyber Mission Data Collection

The first step in the ACOMDR conceptual model is to collect the CMD. The intent is not to collect mission data with this model, but to build an application that processes already collected mission data. The collection of mission data is not trivial and needs to be considered during the design of the ACOMDR application. Just as the performance of an automobile relies on the quality of the fuel, the performance of the ACOMDR application relies on the quality of the input log data. Careful consideration must be placed on the manner in which the input data is processed, as the data collection is an external function beyond the control of the ACOMDR application. Given the assumption that there will be no live connection between the virtual and physical network components, as the ACOMDR tool is designed to function with data that has already been collected and stored in a data repository, there will be no twinning process or feedback cycle readily available between our two abstractions. Nevertheless, we desire as much as possible, a digital twin instance of the target network within our virtual infrastructure. Therefore, the fidelity of our digital environment is a variable dependent on the quality and type of collected log data.

In an ideal scenario, network specifications would be provided for the corresponding CMD. This scenario is advantageous in that the fidelity of the digital twin network is independent of the type and quality of the CMD. However, network specifications are not always available (as with OCO operating outside of friendly networks) or provided. Given a scenario where network specifications are not available, we can retain the capability of replaying CMD by developing functionality to extract network specifications form the provided CMD in order to instantiate a digital twin network.

Given the end state of a high-fidelity virtual representation of a physical network environment which will process the data, the design of our software needs to maintain a degree of modularity, flexibility, and scalability to process increasing amounts and types of log data to support the needed fidelity of the virtual environment. In other words, the application needs to produce the highest fidelity replay, independently of the data quality. In certain circumstances, assumptions can be made about missing information within a dataset in order to produce sufficient results, nevertheless high quality CMD should be pursued to the fullest possibility.

As the ACOMDR application is designed to replay network traffic in a digital twin network, we can narrow the scope of desired mission data to network traffic log data captured during a cyber mission. There are currently 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.

2. Cyber Mission Data Sourcing

The collection of CMD in the second step of the ACOMDR model is a distributed process that is extremely circumstantial. Thus, it is necessary to establish repositories for collected CMD that can store the various data collected throughout the CMF. One such repository utilized by MARFORCYBER 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.

Given the variability in network monitoring applications, and to increase the degree of interoperability of the ACOMDR application, we use mission data that has been ingested and stored in BDP-CHAOS under the ECS. Using a normalized data schema for our ACOMDR application facilitates a highly versatile ACOMDR tool capable of supporting multiple different data types and origins. This capability requires increased mapping prior to data ingestion to be successful. For our initial program development, we opted to use Zeek log data that has undergone ingestion into an ECS database. Zeek logs such as *conn.log*, *http.log*, and *dhcp.log* enable extraction of information such as network configurations, hosts, and event information, which can then be used to build our digital twin network and replay event traffic. This yields an initial capability to replay CMD captured using the Zeek application. Incorporation of subsequent data captured from other (non-Zeek) IDS/IPS applications requires additional front-end mapping and validation.

3. ACOMDR UI

The third step in the ACOMDR conceptual model consists of the application user interface (UI). Via the PCTE web portal, a user will be able to navigate to the ACOMDR applet hosted on the PCTE portal and within the PCTE infrastructure. In step three, a user must provide a dataset for processing. There are two primary approaches to accomplishing how the application will process the data: pointing to data housed within the PCTE infrastructure, or an API call to an external source. Utilizing a pointer within PCTE would require that the mission data be already stored on PCTE resources. In order to have a minimal impact on the PCTE resources, decrease frequency of large data transfers over multiple networks, and increase interoperability, we anticipate developing the program to facilitate making API calls to separate external data repositories.

4. Data Processing

The fourth step in the ACOMDR conceptual model is to process the targeted data. The goal of processing the mission data is twofold: extract the network configurations to build the digital twin network, and build a chronological event matrix. A foundational Zeek log type that enables correlating events across log types is the connection log, or *conn.log*. This grouping of data fields is designed to act as an initial overview and is more beneficial when combined with additional Zeek logs. However, it is useful to establish a chronological record of connections to build our event matrix and identify nodes. Some key data fields from the Zeek *conn.log* include:

- “ts” – time of the first packet
- “uid” – unique identifier for the specific connection
- “id.XXXX_X” – connections 4-tuple
- “proto” – transport layer protocol
- “service” – application protocol
- “conn_state” – 13 possible variables defining state of connection

We can build a more complete picture of the network and corresponding traffic when the *conn.log* data fields are further refined in conjunction with subsequent Zeek logs. The “uid” data field is an artifact of the Zeek application that helps to identify a single connection.

a. Network Configuration

Extracting the network configurations will primarily rely on information tabulated from the Zeek *dhcp.log* and *dns.log* data fields. The *dhcp.log* enables us to correlate media access control (MAC) addresses to dynamically and statically assigned internet protocol (IP) addresses. Identification of a MAC address is particularly useful as it is composed of two key pieces of information: the organizationally unique identifier (OUI), which identifies the card’s manufacturer, and the network interface card (NIC), which is unique to a specific device. Additionally, extracting the MAC address of a device is desirable as it enables us to identify possible IP number reuse and make assumptions on devices on the target network. The *dhcp.log* also enables us to identify the gateway routers and infer the corresponding subnet mask. Additional information from the *dhcp.log* can include machine hostname and connection state.

Another useful grouping of data fields originates from the Zeek *dns.log*. The *dns.log* helps to provide an additional layer of clarity by correlating IP addresses to domain names and record types (AAAA, A, MX, etc.). The *dns.log* can help to distinguish between the various end devices by identifying a device as a specific type of server.

b. Events

In addition to extracting network specifications, the CMD must be processed to extract events that transpired between the various network nodes. Given that the purpose of the ACOMDR application is to replay the events of a cyber mission, to enable CMF personnel to better visualize the events that took place, we need to first compile a list of events, and then aggregate additional information about each event to better depict what happened. For example, if node A visited a website hosted by server B, we want to categorize that as an *event* or even multiple events depending on the traffic generated. We can leverage the Zeek *uid* data field as a primary key during log processing, allowing us to distinguish between a new event and additional information pertaining to an existing event.

5. Relational Database

The fifth step of the ACOMDR conceptual model represents its data store. Many relational—and potentially non-relational—database models can be used with the ACOMDR application. Given the high probability of CMD being well-structured, it is natural to gravitate towards a relational database, with its highly structured schema. Common SQL databases such as MySQL, PostgreSQL, and SQLite are mature products that all have a high degree of compatibility with common programming languages. We elected to utilize MySQL as it is open source, interoperable, and scalable, and it supports both relational and non-relational database structures [31].

Figure 4 depicts a high-level conceptual model of the relational database schema we incorporated into the ACOMDR application. The *Connection* entity in Figure 4 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 will 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.

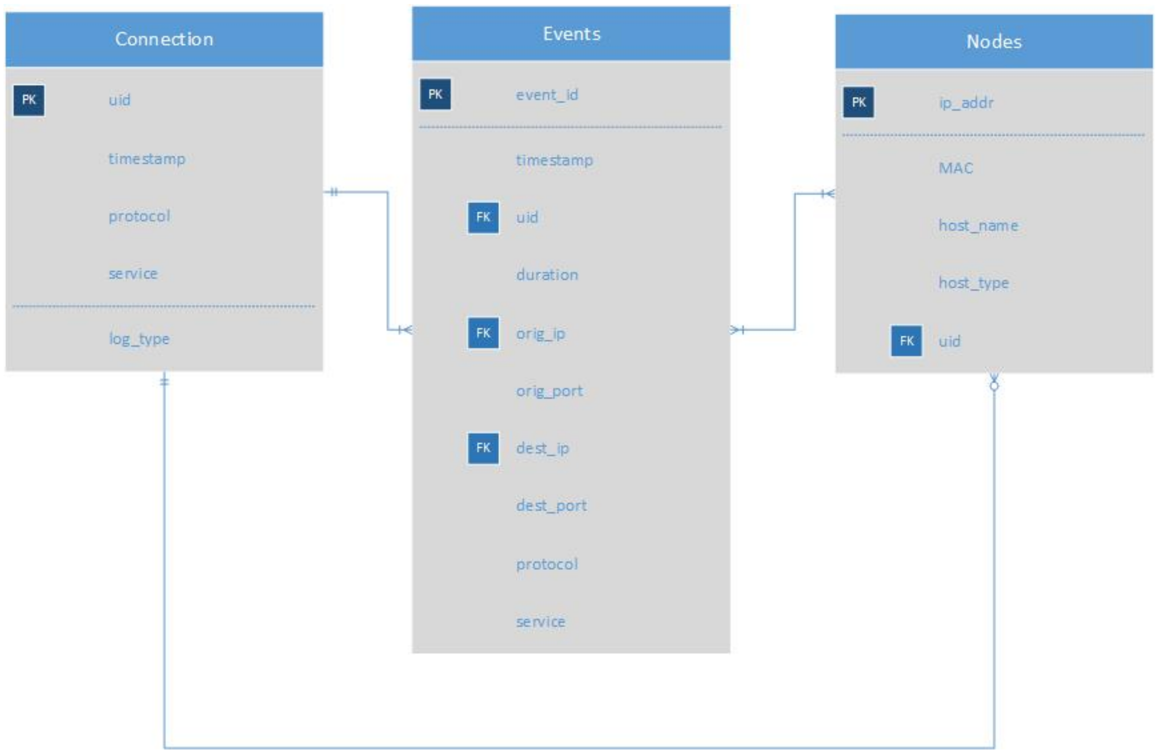


Figure 4. Conceptual relational database schematic

From this database schema, we will be 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.

6. Network Builder Module

The sixth step of the ACOMDR conceptual model represents the *Network Builder*, which builds the digital twin network. The ACOMDR application can accomplish this through API calls to the virtualization infrastructure hosted within PCTE. As previously discussed, PCTE hosts a robust virtualization infrastructure capable of running a broad variety of VMs.

The *Network Builder* module of the ACOMDR application will be the primary component that orchestrates the processes necessary to interface with the PCTE hypervisor and ACOMDR relational database. With regard to virtual network construction, the *Network Builder* must pass configuration parameters for each node to the hypervisor for node creation. Additionally, the *Network Builder* must pass information on the logical topology of the network.

7. Event Manager Module

In the seventh step of the ACOMDR conceptual model, the program iterates through traffic events on the newly established digital twin network. The *Event Manager* module of the ACOMDR application acts as the primary interface with the Puppet platform, which automates this process. Puppet is a software configuration management tool already established within the PCTE architecture [32]. An open source orchestration tool developed by Puppet, Bolt, enables automation of deployed VMs [33]. The *Event Manager* module leverages the appropriate Puppet module to automate the execution of events in the digital twin network.

8. Replay Visualization

The final step of the ACOMDR conceptual model is for the program to display the network traffic to the user. This step is at the crux of the ACOMDR application. An application that perfectly executes the functions described in the earlier steps, but then fails to deliver a clear visual depiction of those events, provides little value to the CMF user. Additionally, this step offers the developers the most flexibility and creativity within the

ACOMDR project. As such, we will define both the target end state of the application and the iterative methodology we intend to adopt to achieve those goals.

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, we can then iterate through the events by highlighting the corresponding nodes and displaying information regarding the type of connection. The second approach would be to develop a visualization infrastructure specifically for the ACOMDR application. The second approach leverages the developer's creativity and expands upon the ACOMDR application's UI module. While this may impose a larger footprint on PCTE infrastructure, it will facilitate tailoring the UI to the ACOMDR application. Development of the UI via the second approach can be effectively accomplished by means of JavaScript (JS) or PHP: Hypertext Preprocessor (PHP).

JS is a lightweight, interpreted, object-oriented scripting language commonly implemented on the client side of web browser environments. Because JS is an interpreted language, it does not need to be compiled at runtime. JS is attractive for website developers as it permits a degree of user interactivity with the website that traditional HTML cannot provide, however, without the Node.js RTE, JS will require a user to have the appropriate plug-ins for their web browser [34]. The Node.js RTE allows for the web server to process the JS code, as opposed to the client's browser, and removes the dependency for different languages between the client and server. An important distinction between JS and Java is that JS is not part of the Java platform; they are disjoint entities [35], [36].

PHP is an open-source HTML-embedded interpreted scripting language commonly employed for web development. In website applications, PHP is processed on the server and the resulting HTML is returned to a client's browser to create dynamic webpage content. Additionally, PHP can run on common operating systems rendering it platform-independent [37].

For the scope of this thesis, we focus on the development of steps 1–6. Therefore, we opted to adopt a simple CLI for the UI in order to diagnose and validate the ACOMDR application. However, by planning for a more robust UI during the development of steps 1–6, we can set conditions for corresponding future work on the ACOMDR application.

C. SUMMARY

This chapter presented the conceptual model for the design of the ACOMDR application. The ACOMDR application consists of three modules and a database developed using the Java platform. The ACOMDR application is designed to be hosted from within the PCTE infrastructure and allow users to process already collected mission data within the PCTE environment. The ACOMDR application will enable connections to external data repositories, which enables the PCTE to process collected mission data for user visualization, training, and future mission rehearsals.

In the next chapter, we discuss the implementation of steps one through six for the ACOMDR application. The objective was to source Zeek CMD from BDP-CHAOS in order to extract a network spec and instantiate a network within PCTE.

IV. TEST SYSTEM IMPLEMENTATION AND RESULTS

Chapter III described the MDR application framework for replaying CMD by accomplishing eight sequential steps. This chapter describes a proof-of-concept implementation of steps 1-6 of the framework, culminating in the extraction of network specifications for subsequent future implementation within PCTE. The goal is to outline the step-by-step approach we implemented to validate this proof-of-concept model.

A. OVERVIEW

If CMD does not have a corresponding network specification, as is the case with the CMD used for the development of this program, the first step to replay the CMD is to extract a network specification and use it to instantiate the twin network on PCTE. The proof-of-concept ACOMDR program we developed accomplishes this task by utilizing two relational databases to process the CMD and produces a "YAML Ain't Markup Language" (YAML) network specification file for subsequent network instantiation (YAML will be described in further detail in Section D of this chapter).

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

B. ENVIRONMENT

The application design uses the following environment:

- OS: Windows 10 Home Version 21H1

- Integrated development environment (IDE): Eclipse IDE for Java Developers Version 2021-06
- Java: Version 15.0.2
- Java Development Kit (JDK): Version 15.0.2
- SnakeYAML: Version 1.25
- MySQL Connector/J: Version 8.0.25
- MySQL: Version 8.0.25

C. DATABASE

Our Automated Cyber Operations Mission Data Replay (ACOMDR) application was developed using two disjoint MySQL databases hosted on the same local server. The first database simply contains CMD in an unaltered state, as it was ingested into the MySQL database immediately after download from BDP-CHAOS. We will refer to CMD in the first database as raw CMD. The second database contains CMD that is specifically selected and aggregated by the ACOMDR application, or processed CMD. Figure 5 provides a high-level depiction of the data processing sequence used in production of the ACOMDR application.

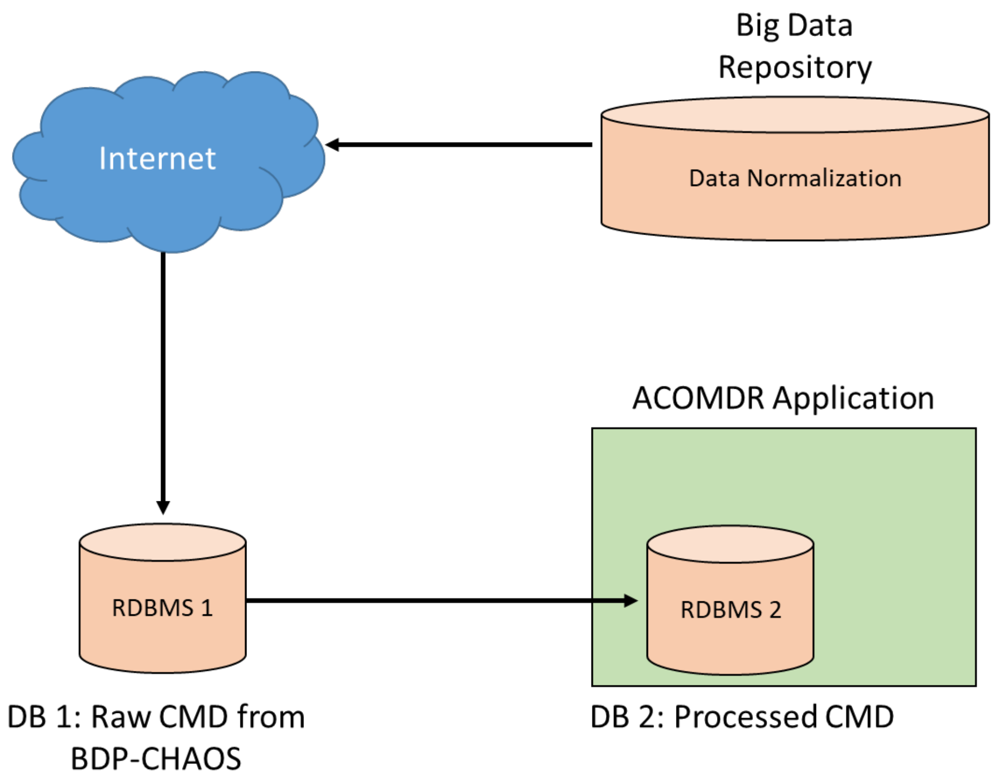


Figure 5. ACOMDR development using two databases

The ACOMDR application database, depicted as RDBMS 2, was developed to allow the program to quickly characterize two aspects of the corresponding CMD. First, the program needs to produce a network specification from the CMD. This was accomplished using tables that catalog the connections made and nodes involved. Second, the program needs to replay the events that transpired in the CMD. This was accomplished using an events table.

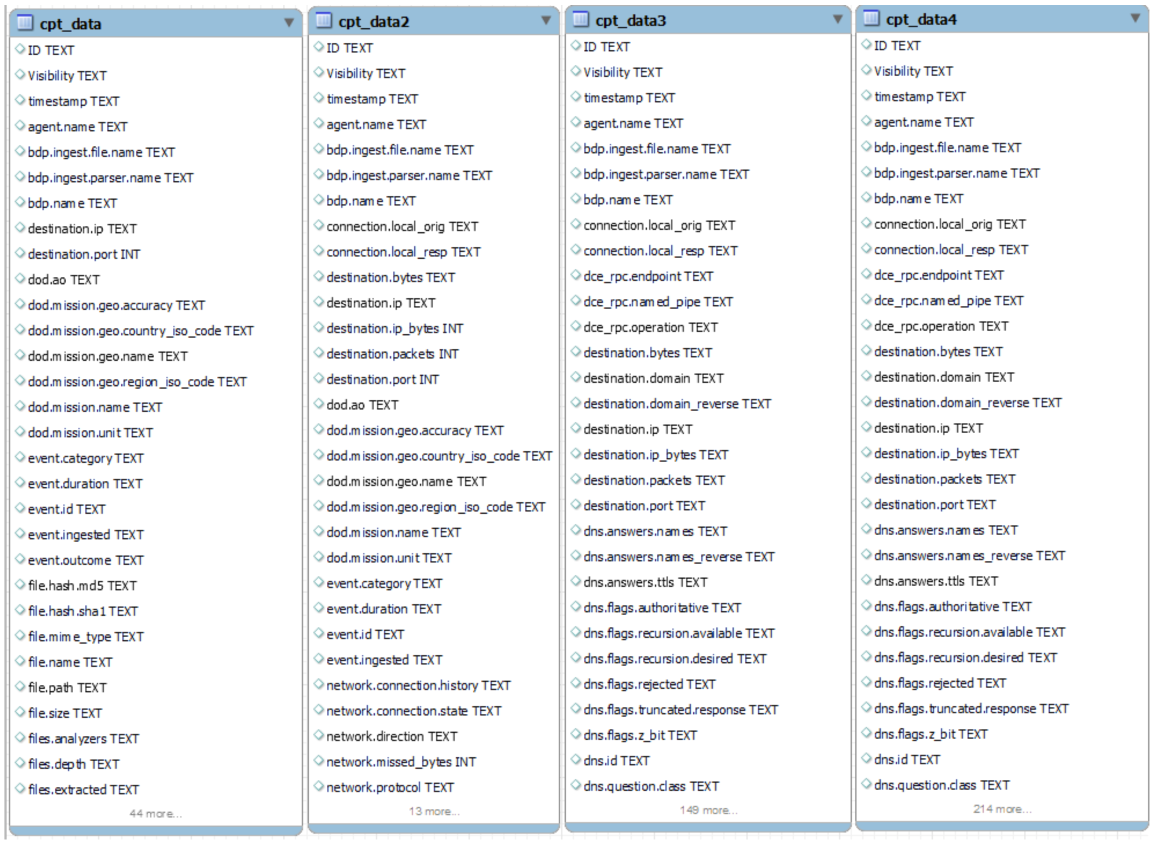
1. Database 1

As previously discussed, BDP-CHAOS is a MARFORCYBER data repository that houses data from the Marine Corps Enterprise Network. The BDP-CHAOS platform provides users with a capability to visualize data from many IPS/IDS sources as well as to query various datatypes and timeframes. As such, we were able to make queries for a unique CMD set for varying timeframes. For developmental purposes, we downloaded four CMD datasets, each with different dates and durations. As BDP-CHAOS uses ECS, the fields of each dataset vary depending on the corresponding original Zeek logs. A summary of major Zeek logs found within each dataset is given in Table 2.

Table 2. Primary Zeek logs in datasets 1-4

	Dataset 1	Dataset 2	Dataset 3 and 4
Zeek Log Types	smb_files	conn.log	conn.log
	weird.log		dns.log
	ssl.log		files.log
	files.log		snmp.log
			ssl.log
			rpc.log

The purpose of Database 1 is to maintain tables for each of the four datasets from BDP-CHAOS, downloaded as comma-separated value (CSV) files. MySQL Workbench offers an efficient *Table Data Import Wizard* tool that enables users to quickly create tables within a MySQL database from local CSV files. All fields in the datasets were imported into MySQL as text to prevent formatting errors between MySQL's import wizard and the cell values. Figure 6 depicts the four datasets imported into our local MySQL database where each dataset exists as a table containing the various fields.



cpt_data

ID TEXT
Visibility TEXT
timestamp TEXT
agent.name TEXT
bdp.ingest.file.name TEXT
bdp.ingest.parser.name TEXT
bdp.name TEXT
destination.ip TEXT
destination.port INT
dod.ao TEXT
dod.mission.geo.accuracy TEXT
dod.mission.geo.country_iso_code TEXT
dod.mission.geo.name TEXT
dod.mission.geo.region_iso_code TEXT
dod.mission.name TEXT
dod.mission.unit TEXT
event.category TEXT
event.duration TEXT
event.id TEXT
event.ingested TEXT
event.outcome TEXT
file.hash.md5 TEXT
file.hash.sha1 TEXT
file.mime_type TEXT
file.name TEXT
file.path TEXT
file.size TEXT
files.analyzers TEXT
files.depth TEXT
files.extracted TEXT
44 more...

cpt_data2

ID TEXT
Visibility TEXT
timestamp TEXT
agent.name TEXT
bdp.ingest.file.name TEXT
bdp.ingest.parser.name TEXT
bdp.name TEXT
connection.local_orig TEXT
connection.local_resp TEXT
destination.bytes TEXT
destination.ip TEXT
destination.ip_bytes INT
destination.packets INT
destination.port INT
dod.ao TEXT
dod.mission.geo.accuracy TEXT
dod.mission.geo.country_iso_code TEXT
dod.mission.geo.name TEXT
dod.mission.geo.region_iso_code TEXT
dod.mission.name TEXT
dod.mission.unit TEXT
event.category TEXT
event.duration TEXT
event.id TEXT
event.ingested TEXT
network.connection.history TEXT
network.connection.state TEXT
network.direction TEXT
network.missed_bytes INT
network.protocol TEXT
13 more...

cpt_data3

ID TEXT
Visibility TEXT
timestamp TEXT
agent.name TEXT
bdp.ingest.file.name TEXT
bdp.ingest.parser.name TEXT
bdp.name TEXT
connection.local_orig TEXT
connection.local_resp TEXT
dce_rpc.endpoint TEXT
dce_rpc.named_pipe TEXT
dce_rpc.operation TEXT
destination.bytes TEXT
destination.domain TEXT
destination.domain_reverse TEXT
destination.ip TEXT
destination.ip_bytes TEXT
destination.packets TEXT
destination.port TEXT
dns.answers.names TEXT
dns.answers.names_reverse TEXT
dns.answers.ttls TEXT
dns.flags.authoritative TEXT
dns.flags.recursion.available TEXT
dns.flags.recursion.desired TEXT
dns.flags.rejected TEXT
dns.flags.truncated.response TEXT
dns.flags.z_bit TEXT
dns.id TEXT
dns.question.class TEXT
149 more...

cpt_data4

ID TEXT
Visibility TEXT
timestamp TEXT
agent.name TEXT
bdp.ingest.file.name TEXT
bdp.ingest.parser.name TEXT
bdp.name TEXT
connection.local_orig TEXT
connection.local_resp TEXT
dce_rpc.endpoint TEXT
dce_rpc.named_pipe TEXT
dce_rpc.operation TEXT
destination.bytes TEXT
destination.domain TEXT
destination.domain_reverse TEXT
destination.ip TEXT
destination.ip_bytes TEXT
destination.packets TEXT
destination.port TEXT
dns.answers.names TEXT
dns.answers.names_reverse TEXT
dns.answers.ttls TEXT
dns.flags.authoritative TEXT
dns.flags.recursion.available TEXT
dns.flags.recursion.desired TEXT
dns.flags.rejected TEXT
dns.flags.truncated.response TEXT
dns.flags.z_bit TEXT
dns.id TEXT
dns.question.class TEXT
214 more...

Figure 6. Datasets downloaded and imported into MySQL

2. Database 2

The second database exists within the ACOMDR application. When ACOMDR is deployed within the PCTE infrastructure, the application will require a local MySQL server to process and house the necessary data to replay the CMD. This second database contains the connection table, the nodes table, and the events table, whose entities were defined in Chapter III.

The **connection table** aggregates the *uid*, timestamp, protocol, transport layer, service, and log type of every connection made within the CMD dataset. Null values are allowed in all imported fields and each connection is indexed using a local auto-incremented connection number field.

The **nodes table** aggregates the IP address, MAC address, host name, and host type from the CMD. Null values are allowed in all imported fields and each node is indexed using a local auto-incremented node number field.

The **events table** aggregates the timestamp, *uid*, connection duration, originating IP address, originating port, destination IP address, destination port, application layer protocol, and transport layer from the CMD. Null values are allowed in all imported fields and each event is indexed using a local auto-incremented event number field. The tables necessary for the ACOMDR application database are depicted in Figure 7 with character spaces detailed within the parenthesis.

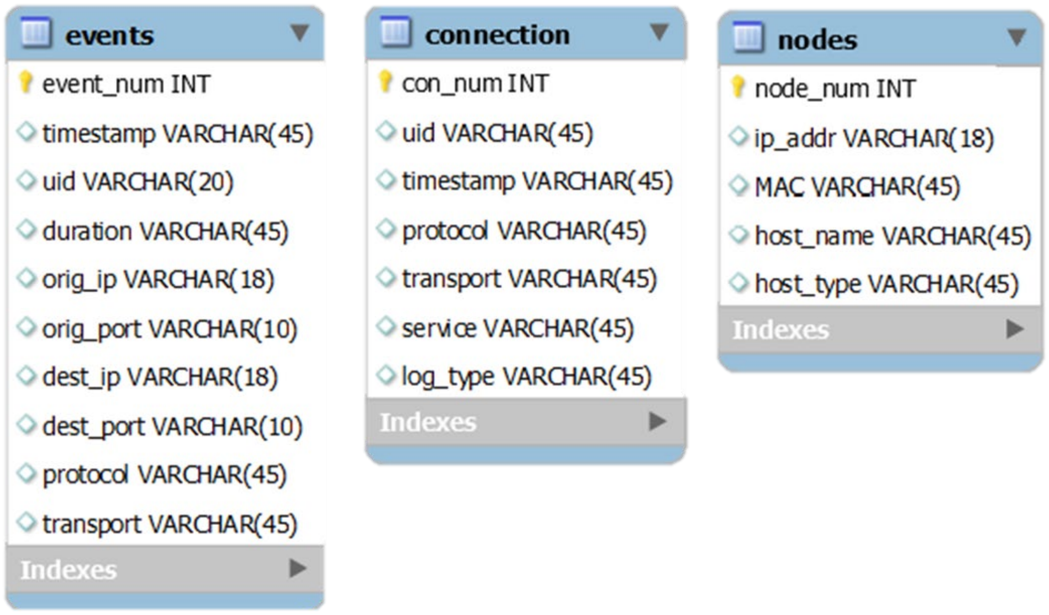


Figure 7. ACOMDR database tables

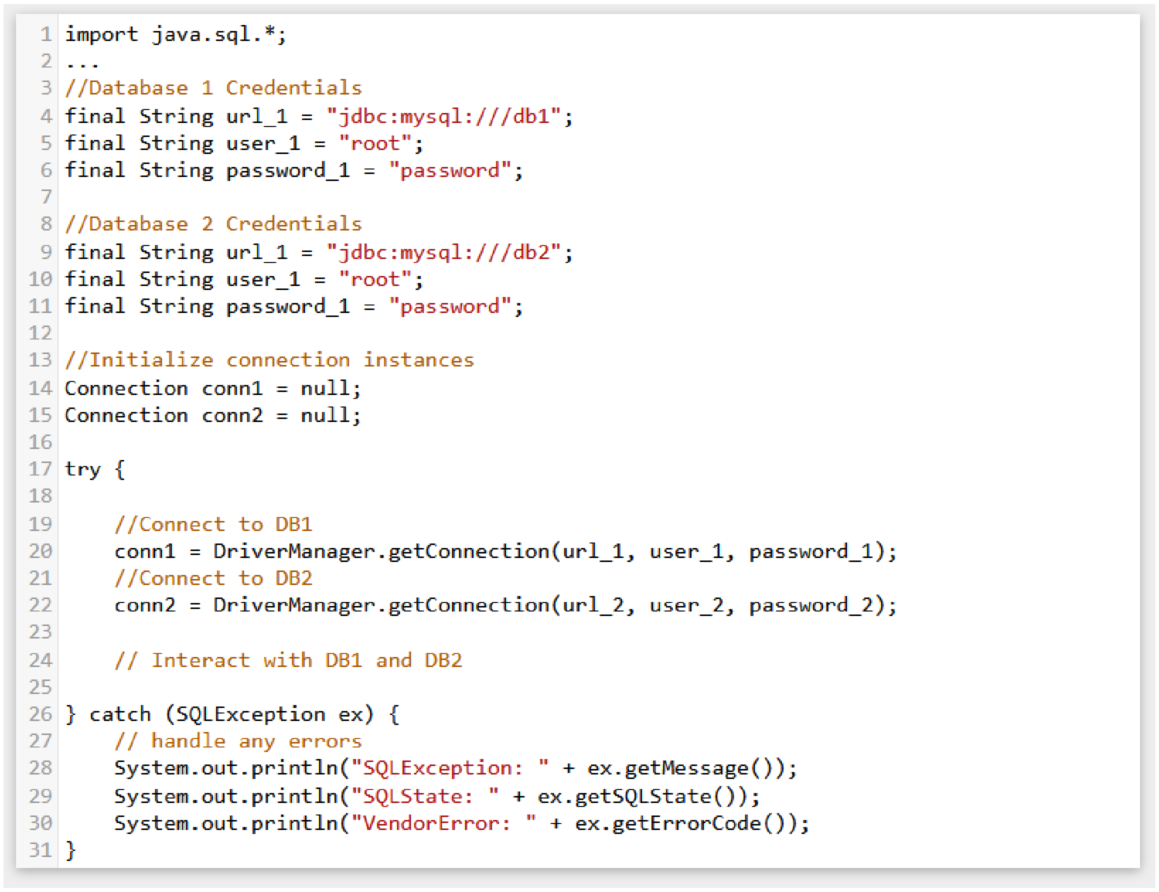
D. PROGRAM

The proof-of-concept ACOMDR application was designed in Eclipse using the MySQL Connector/J, which is the Java Database Connectivity (JDBC) driver that implements the JDBC API. The following sub-sections describe the major components of our Java application, database interface, and resulting output.

1. MySQL Connection

To have the ACOMDR application connect to the local MySQL server, we developed a MySQLAccess class with the ACOMDR program. This class connects to both the raw CMD database (Database 1) and the ACOMDR database (Database 2) to extract the data necessary to populate the three tables in the ACOMDR database.

For the `MySQLAccess` class to access the databases, we first had to download the latest version of the Connector/J JAR file. Once downloaded, we mapped the Java Build Path to the external Connector/J JAR file. The JDBC API allows for quick and efficient connections to MySQL databases by obtaining a *Connection* instance from the *DriverManager*. A modified summary of our `MySQLAccess` class connection methodology is depicted in Figure 8. We connected to each database by declaring unique connection variables (server location, username, and password) for each database, depicted as lines 4-6 and 9-11 in Figure 8, as well as declaring two null *connection* instances outside of our methods, as was done in lines 14 and 15 in Figure 8. However, unlike Figure 8, a connection to a database is only established within a method and is closed before returning from the method in order to manage the connections. Therefore, each method begins with a `try` block establishing the connection, as depicted in line 17 of Figure 8, followed by prepared statements to interact with the MySQL databases.



```java
import java.sql.*;
...
//Database 1 Credentials
final String url_1 = "jdbc:mysql:///db1";
final String user_1 = "root";
final String password_1 = "password";

//Database 2 Credentials
final String url_1 = "jdbc:mysql:///db2";
final String user_1 = "root";
final String password_1 = "password";

//Initialize connection instances
Connection conn1 = null;
Connection conn2 = null;

try {

    //Connect to DB1
    conn1 = DriverManager.getConnection(url_1, user_1, password_1);
    //Connect to DB2
    conn2 = DriverManager.getConnection(url_2, user_2, password_2);

    // Interact with DB1 and DB2

} catch (SQLException ex) {
    // handle any errors
    System.out.println("SQLException: " + ex.getMessage());
    System.out.println("SQLState: " + ex.getSQLState());
    System.out.println("VendorError: " + ex.getErrorCode());
}
```

Figure 8. Connection to MySQL databases 1 and 2

2. Processing ACOMDR Data

To process the raw CMD stored in Database 1, we defined seven methods in the `MySQLAccess` class. These public methods are called from the UI class and each performs a unique function:

- `ArrayList<String> query_tables()`: takes no parameters and returns an *ArrayList* of strings. This method returns a dynamic array of the tables in Database 1. As the tables in Database 2 are fixed, the method does not need to establish a connection or interact with Database 2.

- `void wipeTbl(String table)`: takes a Database 2 table name (string) as a parameter and has the return voided. This method has the capability to truncate tables in Database 2, which retains the table fields but deletes all rows in the table.

- `void build_con_tbl(String table)`: takes a Database 1 table name (string) as a parameter and has the return voided. This method populates the connection table in Database 2 with values from the desired raw CMD in Database 1.

- `void build_node_tbl(String table)`: takes a Database 1 table name (string) as a parameter and has the return voided. This method populates the node table in Database 2 with values from the desired raw CMD in Database 1.

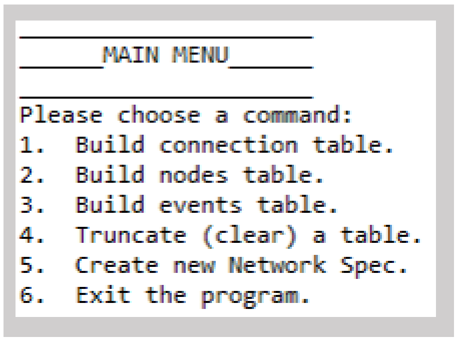
- `void build_events_tbl(String table)`: takes a Database 1 table name (string) as a parameter and has the return voided. This method populates the events table in Database 2 with values from the desired raw CMD in Database 1.

- `void sql_close()`: takes no parameters and has the return voided. This method closes any open connections and is used internally within the class so is private.

- `ResultSet sql_pull(String port)`: takes a port number as a parameter and returns the `ResultSet`. This method queries Database 1 for information necessary to build the network specification document and returns a `ResultSet`, which is the table of data that is generated from executing prepared statements or statements.

3. User Interface

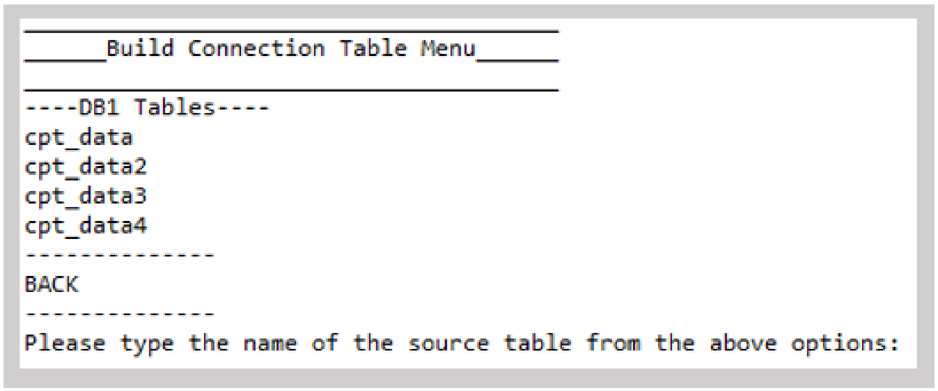
Next we developed a simple CLI to function as the UI for the ACOMDR program. We accomplished this by developing a UI class within our ACOMDR Java project. The UI initially presents users with a main menu listing six functional options, as depicted in Figure 9.



```
_______________________
______MAIN MENU______
_______________________
Please choose a command:
1.  Build connection table.
2.  Build nodes table.
3.  Build events table.
4.  Truncate (clear) a table.
5.  Create new Network Spec.
6.  Exit the program.
```

Figure 9. ACOMDR UI main menu

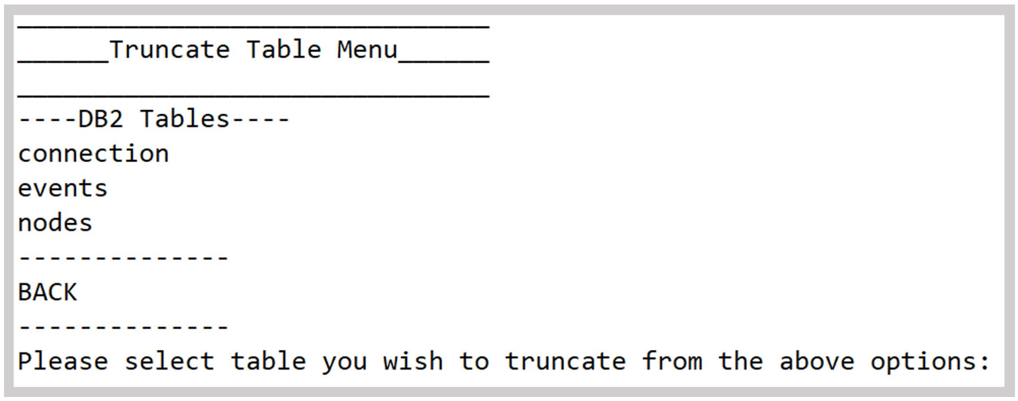
From the main menu, users are directed to one of five sub-menus based on their choice (the sixth option exits the program). From these sub-menus, users are asked to make various additional selections or confirm their selections. Options 1-3 of the main menu direct the user to identify a source table from Database 1 that the application will use to populate the tables in Database 2. The sub-menu for the first option to build the connection table is depicted in Figure 10 (the sub-menus for the second and third options of the main menu are similar to this one).



```
_______________________________________
______Build Connection Table Menu______
_______________________________________
----DB1 Tables----
cpt_data
cpt_data2
cpt_data3
cpt_data4
--------------
BACK
--------------
Please type the name of the source table from the above options:
```

Figure 10. ACOMDR UI build connection table sub-menu

The sub-menu for option 4 to truncate a table directs the user to identify a table to truncate from Database 2, as depicted in Figure 11. We have restricted the ability of users to only truncate tables within Database 2 to avoid any accidental erasing of raw CMD. Future implementation of this method could be automated and called once a user exits the application.



```
_______________________________
______Truncate Table Menu______
_______________________________
----DB2 Tables----
connection
events
nodes
--------------
BACK
--------------
Please select table you wish to truncate from the above options:
```

Figure 11. ACOMDR UI truncate table sub-menu

All user input commands are verified prior to passing the commands to any other methods or classes. In the event a user inputs an incorrect command, such as a letter character vice an integer, they are either redirected to the main menu or asked to try again. In the event a user inputs the incorrect name for one of the various tables, we follow a similar procedure of verifying the input by comparing it to queried table names from Database 1 or hard-coded table names for Database 2.

4. Formatting YAML Output File

The network specification is written to a YAML file by means of the SnakeYAML package. YAML is a readable markup language that is commonly used in place of extensible markup language (XML) for configuration files. The basic structure of YAML is key-value pairing in a standardized format.

The format of the YAML network specification file is configured in accordance with the Oasis Topology and Orchestration Specification for Cloud Applications (TOSCA) Simple Profile in YAML Version 1.2 standard. We have developed a `YAML_Writer` class, a custom representer class, and three additional object classes that collectively build the YAML network specification file. Additionally, we leveraged the `sql_pull` method in the `MySQLAccess` class that takes a Database 2 table name as a string parameter and returns the `ResultSet` from the MySQL query. From this `ResultSet`, we are able to iterate through the database table information to populate the network specification objects. A summary of the YAML output architecture is depicted in Figure 12.

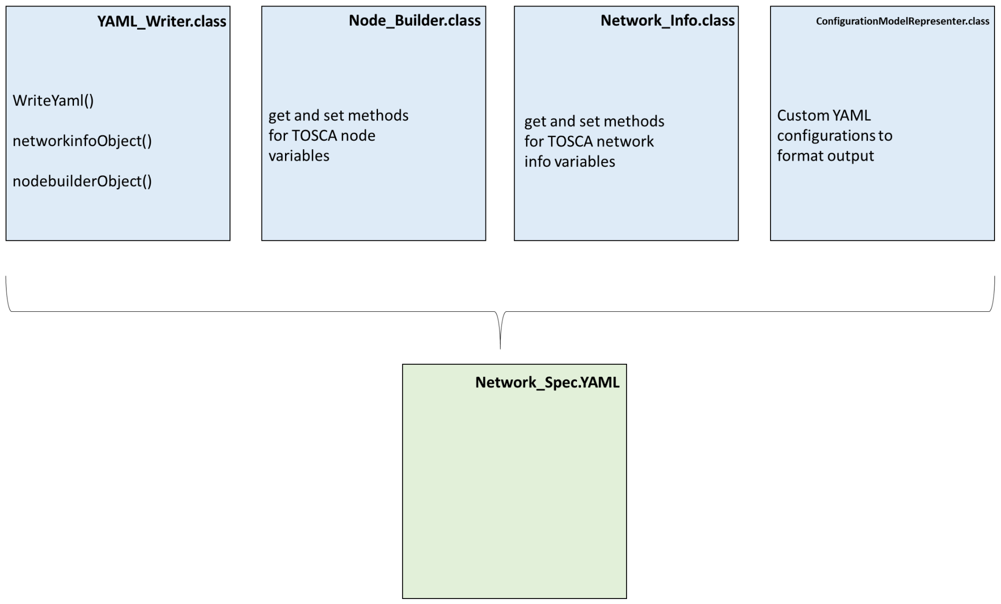
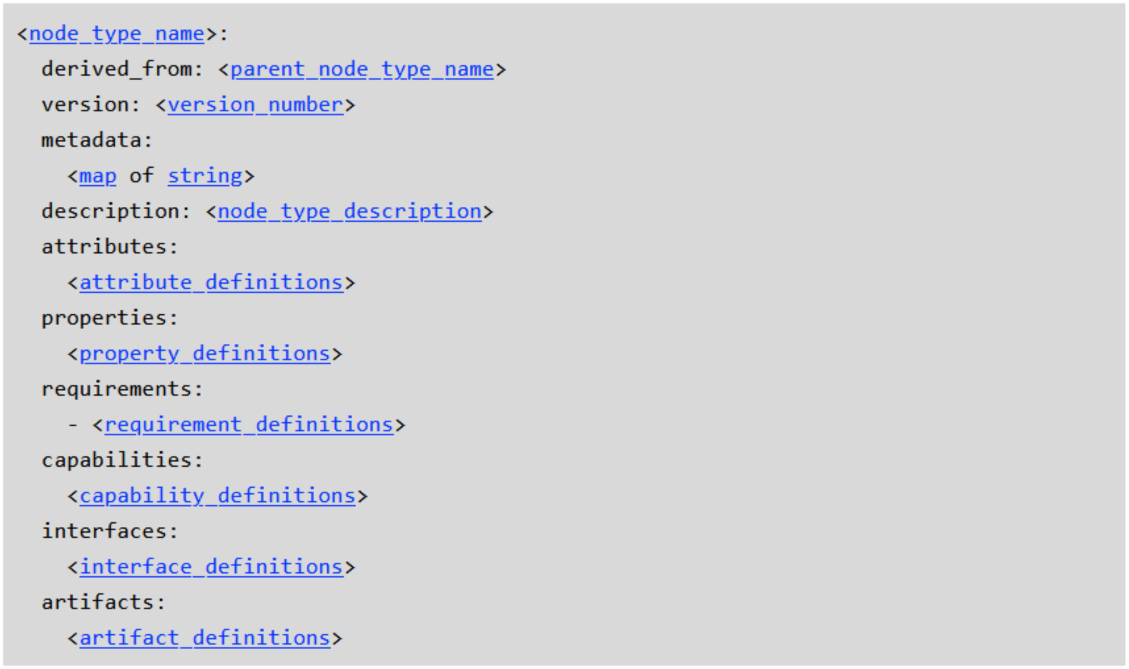

YAML_Writer.class
WriteYaml()
networkinfoObject()
nodebuilderObject()
Node_Builder.class
get and set methods for TOSCA node variables
Network_Info.class
get and set methods for TOSCA network info variables
ConfigurationModelRepresenter.class
Custom YAML configurations to format output
Network_Spec.YAML

Figure 12. ACOMDR YAML output architecture

We have defined three methods within the YAML writer class that allow us to configure and produce the network specification file. The `WriteYaml` method allows us to set certain snakeYAML dumper options when writing the YAML file. This method also allows us to define the path for the resulting YAML output file and makes individual method calls to the two other private methods to write the YAML file. The remaining two methods are a network object method and a node object method. Each of these methods make calls to `MySQLAccess` class in order to query Database 2. From the returned `ResultSet`, these methods call the network specification object classes and set the objects in accordance with the database.

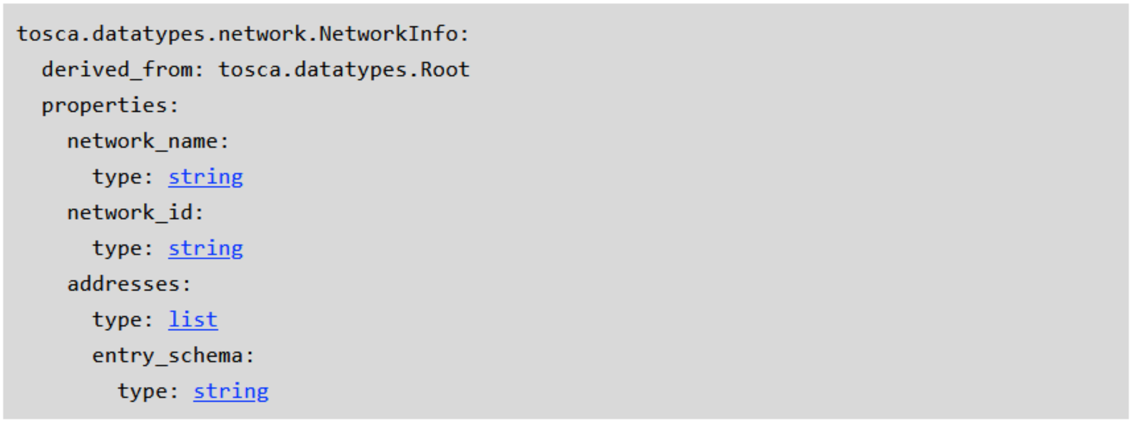
The custom representer class, which extends the SnakeYAML representer class, allows us to configure specific formatting options regarding the writing YAML file. We have configured the custom representer to omit null values, class tags, and adopt the class order of objects.

The remaining three object classes contain get and set for methods for all the private variables defined within the three classes. The variables correlate to the TOSCA standard for nodes and network information. The node and network information variables are depicted in Figure 13 and Figure 14, respectively.



```
<node_type_name>:
  derived_from: <parent_node_type_name>
  version: <version_number>
  metadata:
    <map of string>
  description: <node_type_description>
  attributes:
    <attribute_definitions>
  properties:
    <property_definitions>
  requirements:
    - <requirement_definitions>
  capabilities:
    <capability_definitions>
  interfaces:
    <interface_definitions>
  artifacts:
    <artifact_definitions>
```

Figure 13. TOSCA node type grammar. Source: [38].



```
tosca.datatypes.network.NetworkInfo:
  derived_from: tosca.datatypes.Root
  properties:
    network_name:
      type: string
    network_id:
      type: string
    addresses:
      type: list
      entry_schema:
        type: string
```

Figure 14. TOSCA network info grammar. Source: [38].

E. RESULTS AND ANALYSIS

In order to evaluate the ACOMDR application we developed a three-point testing scheme that we used to evaluate the efficiency and suitability of the application. 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 the ACOMDR application would query the data via API calls.

1. Database Table Population

The time necessary to build the connection, node, and events tables in the ACOMDR database (Database 2) varied significantly depending on the size of the source data tables (Database 1). Table 3 depicts the sizes in megabytes of three source tables that were tested by the application, the average build times of each table in seconds, and the ratio of the build time by size and build time by number of rows.

Table 3. Size of source data in Database 1

Source Data	Columns	Rows	Table size (MB)	Build Time (sec)	sec / size	sec / row
cpt_data	74	6,899	5.50	29.33	5.333	0.004251824
cpt_data2	43	48,372	26.60	300.67	11.303	0.006215717
cpt_data3	179	360,885	267.00	4313.67	16.156	0.011953023

We saw an exponential increase in the time required to process the source data as it increased in size. The time to populate each table is depicted in the left graph of Figure 15, and the relative time to populate each table (seconds divided by number of rows) is depicted in the right graph of Figure 15.

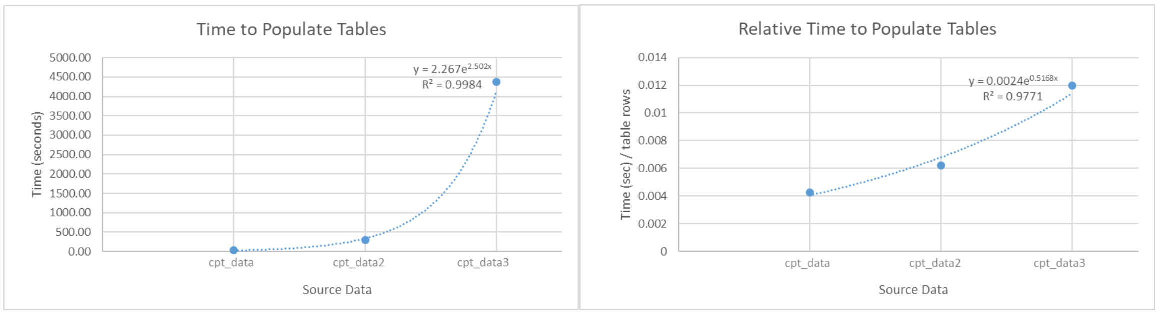


Figure 15. Time to populate data in Database 2 (left) and time divided by table rows (right)

Our results found that the current structure of processing entire source datasets to extract the necessary information is inefficient. Much of the source data, retained in Database 1, is superfluous and currently only serves to inhibit the processing of CMD.

2. Network Specification Population

We evaluated the creation of the network specification file using a varying number of nodes to build the document. Figure 16 depicts the near linear correlation between the number of nodes used to build the network specification document and the time required in seconds to build a network specification document from the data. From our analysis, we projected that the ACOMDR application would take approximately 11.7 hours to process all 38,180 unique node instances within our various datasets if directed to process all nodes.

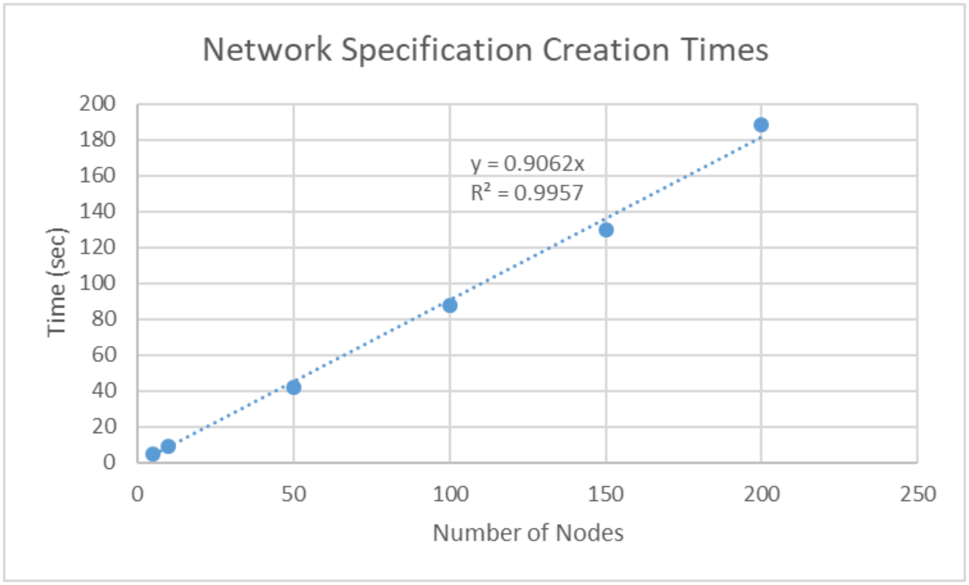


Figure 16. Time to build network specification document

The time necessary to develop a comprehensive network specification is certainly significant. However, additional data processing is necessary to evaluate the node types, interfaces, and connections.

3. Network Specification Suitability

The suitability of the network specification is currently untested. An ideal evaluation would consist of using the ACOMDR YAML network specification file as the source file for the PCTE to instantiate a virtual network. However, we were unable to complete such an evaluation due to time constraints. In lieu of an evaluation within PCTE, we evaluated the network specification from the TOSCA standards and assumptions we made in developing the ACOMDR application.

The TOSCA standards are very broad and incorporate a high degree of flexibility and variability. As such, it is likely that the network specification produced by the ACOMDR application lacks sufficient mapping to the variables utilized within the PCTE infrastructure.

The current network specification produced by the ACOMDR application is limited in scope to port 80 traffic. We hardcoded SQL queries to identify traffic destined for a port 80 IP address. We inferred that these connections were made from a compute node to a web server, and can identify IP address for each. From that inference, we populated the network specification to detail web servers and the nodes connecting to the web servers. From the CLI, users can specify the number of connections to use in building the network specification file. Our program then iterated through the appropriate tables in Database 2 via a MySQL query method available in the `MySQLAccess` class and built the network specification from the results.

While extremely limited in scope, this network specification serves to validate that we can extract some network information from Zeek CMD. Further correlations and mapping are necessary to develop a more robust network specification.

F. SUMMARY

This chapter described the virtual environment, development, results, and analysis of the proof-of-concept ACOMDR application. Our testing determined that the ACOMDR application can process CMD and produce a network specification document. Our testing also concluded that the current version of the ACOMDR application produces a low fidelity network specification and incurs significant delays in processing CMD due to the large amount of superfluous data contained within the database. The current ACOMDR application is a successful proof-of-concept that requires additional development and refinements to increase the fidelity of network specifications and a more efficient data processing mechanism. The next chapter discusses the overall conclusions of this research as well as potential areas for future work.

THIS PAGE INTENTIONALLY LEFT BLANK

V. CONCLUSIONS AND FUTURE WORK

A. SUMMARY

The primary goal of this research was to explore the capability of replaying cyber mission data (CMD) within the PCTE infrastructure through the development of an application. We developed a proof-of-concept tool called the Automated Cyber Operations Mission Data Replay (ACOMDR) application that can process Zeek CMD and produce a network specification. We were able to identify that the primary limitation to replaying CMD within PCTE is the lack of readily available network specifications that can correlate to CMD. A crucial first step in the development of the ACOMDR application was to be able to process the CMD and produce a network specification of the highest possible fidelity.

The ACOMDR application was designed to integrate within the continually evolving PCTE infrastructure and to provide enhanced visualization of cyberspace operations to the Cyber Mission Force (CMF).

B. CONCLUSIONS

The current ACOMDR application serves as a proof-of-concept for the processing of CMD to extract network specifications. Additional data processing or other types of CMD are necessary to establish more correlations between the CMD and node or network parameters to build network specifications of higher fidelity. In the following sections we provide answers to our specific research questions.

1. Primary Research Question

How can the relevant network specifications be extracted from log data to instantiate a digital twin network?

We have successfully developed a proof-of-concept tool that processes CMD to extract network specifications for web servers and nodes attempting to connect to web servers. We have determined that, in order to extract network specifications from original Zeek CMD, it is necessary to develop a list of assumptions regarding node characteristics.

For example, if a node is sending traffic from an ephemeral port to a well-known port, we can assume that the sending node is a user computer node, and the receiving node is a type that corresponds to the well-known port number. Additional assumptions will need to be made about each node as development continues, such as the hardware characteristics, operating system, and other capabilities, properties, and attributes in accordance with the TOSCA standards.

2. Secondary Research Questions

How can log data be converted to scripts to be executed on a digital twin network?

The processing of CMD logs can be accomplished with a software solution, which, in turn, will interface with existing automation in place within the PCTE to control the replay of events on the digital twin network. The software solution we developed successfully processes CMD and produces a network specification, however further development of the ACOMDR application is required for proper to interfacing with the PCTE's existing capabilities, such as the Puppet module or event visualization service. Our current software solution is structured in a manner that will enable further investigation and development to produce a software tool that has the capability to interface with the existing automation tool within the PCTE to replay cyber mission data.

What are the desirable features of a run-time environment to control the execution of cyber mission scripts?

With respect to implementation within PCTE, we have determined that the Puppet module is the appropriate feature to control the execution of events within the digital twin network. Further development of a software solution to interface with Puppet is necessary to facilitate the replay of events within a PCTE network. Additional research is also recommended to determine the feasibility of utilizing Puppet outside of the PCTE to control the execution of events within a network.

C. RECOMMENDATIONS FOR FUTURE RESEARCH

The research we have conducted has identified multiple areas where further research would serve to improve the fidelity and capability of our ACOMDR application.

The areas described below are not all-inclusive, but will answer necessary questions regarding the scope and capability of future development.

1. Scope of Ingestion Data

The proof-of-concept ACOMDR application 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.

We recommend expanding the scope of ingestion data to include additional datatypes to research the possibility of increasing the fidelity of the induced network specification and subsequent digital twin network. BDP-CHAOS hosts other datatypes that may better describe network boundaries and traffic beyond a local LAN.

2. BDP-CHAOS Connectors

The ACOMDR application 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 the ACOMDR application.

A connection to BDP-CHAOS, and modification to the UI, can allow users to directly query BDP-CHAOS for CMD. Most importantly, a connection to BDP-CHAOS would allow the ACOMDR application to pursue the specific attributes of CMD that build higher fidelity network specifications. For example, the ACOMDR application would be much more efficient if the application had the capability to query for specific fields of Zeek CMD in BDP-CHOAS, such as connection logs or DHCP logs. This would decrease the amount of superfluous CMD ingested and streamline the development of network specifications that would allow us to drop the first database (Database 1) housing raw CMD, decreasing the overhead of the application.

Furthermore, it would be useful to establish additional connectors to other BDP's within the CMF community. Additional connectors would increase the level of interoperability and functionality of PCTE capabilities in accordance with the JCWA.

3. UI

The ACOMDR application currently has a simple CLI. The application could be further refined by developing and integrating a robust GUI within the application to display dynamic views of digital twin network topology and iterate through cyber events that occur on the digital twin network. An ideal GUI would include an interactive display that allows for customizable/interactive replay of cyber mission. PCTE currently has the capabilities to produce user visualizations of events that have occurred within the PCTE infrastructure.

4. YAML Processing

The ACOMDR application 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.

5. Network Specification Fidelity

The fidelity of the network specification could be significantly increased through additional mapping and queries of the source data. In order for the ACOMDR application 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.

6. ACOMDR Database Schema

The current ACOMDR database schema defines the Connection, Events, and Nodes tables. This schema could be modified to better support defining the network specification by creating tables for the various node types. A table should be created for each of the various node types, such has web servers, mail servers, domain name servers, and user compute nodes. Data processing should include querying the CMD for well-known port numbers and ephemeral port numbers to populate the various node type tables.

This schema would better support making inferences on the node types and network environment to build a higher fidelity network specification.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX. ZEEK LOG TYPES AND ECS MAPPING

Table 4.   Zeek log types and ECS mapping

Zeek Log	Description
capture_loss.log	analysis of missing traffic
conn.log	stateful (TCP) and stateless (UDP) protocols
dhcp.log	Dynamic Host Configuration Protocol activity
dns.log	Domain Name System activity
dpd.log	Dynamic protocol detection for protocols on ports beyond those used as standard services
files.log	record of observed files
ftp.log	FTP activity
http.log	clear-text HTTP or exposed HTTPS
irc.log	Internet Relay Chat activity
known_certs.log	SSL/TLS certificate information
known_hosts.log	entries for when a new system joins the local network
known_services.log	entries for when a system offers a new service on the local network
notice.log	inspection-worthy situations
ntp.log	Network Time Protocol activity
pe.log	portable executable
rdp.log	Remote Desktop Protocol activity
reporter.log	internal warnings and errors
SMB Logs	commonly associated with Microsoft Windows

smtp.log	email traffic using ports 25, 465, and 587 TCP
software.log	details on applications operated by the hosts
ssh.log	details about SSH sessions
ssl.log	parsed TLS traffic (typically includes HTTPS)
traceroute.log	script that tries to identify traceroute activity
tunnel.log	identify encapsulated traffic (IPv6 within IPv4)
weird.log	anomalies and unknown records
x509.log	certificates exchange details from TLS negotiations

LIST OF REFERENCES

[1] Department of Defense, "Department of Defense strategy for operating in cyberspace," Washington, DC, USA, 2011 [Online]. Available: https://csrc.nist.gov/CSRC/media/Projects/ISPAB/documents/DOD-Strategy-for-Operating-in-Cyberspace.pdf

[2] J. Garamone and L. Ferdinando, "DOD initiates process to elevate U.S. Cyber Command to unified combatant command," DOD News, Aug. 18, 2017 [Online]. Available: https://www.defense.gov/Explore/News/Article/Article/1283326/dod-initiates-process-to-elevate-us-cyber-command-to-unified-combatant-command/

[3] US Cyber Command, "Command vision for U.S. Cyber Command: Achieve and maintain cyberspace superiority," Washington, DC, USA, 2018 [Online]. Available: https://www.cybercom.mil/Portals/56/Documents/USCYBERCOM%20Vision%20April%202018.pdf?ver=2018-06-14-152556-010

[4] G. Huskaj, "The current state of research in offensive cyberspace operations," *Eur. Conf. Cyber Warf. Secur.*, pp. 660–667, 2019 [Online]. Available: https://www.diva-portal.org/smash/get/diva2:1337484/FULLTEXT01.pdf

[5] U.S. House. 116th Congress, 2nd Session. (2020, Mar. 4). *Statement of General Paul M. Nakasone Commander United States Cyberspace Command Before the House Committee on Armed Services Subcommittee on Intelligence and Emerging Threats and Capabilities* [Online]. Available: https://docs.house.gov/meetings/AS/AS26/20200304/110592/HHRG-116-AS26-Wstate-NakasoneP-20200304.pdf

[6] 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

[7] 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

[8] 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

[9] 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.

[10] 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

[11] M. Karjalainen and T. Kokkonen, "Comprehensive cyber arena; the next generation cyber range," 2020 *IEEE Eur. Symp. Secur. Priv. Workshop (EuroS&PW)*, pp. 11–16, Sep. 2020 [Online]. Available: https://doi.org/10.1109/EuroSPW51379.2020.00011

[12] E. Glaessgen and D. Stargel, "The digital twin paradigm for future NASA and U.S. Air Force vehicles," in *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, American Institute of Aeronautics and Astronautics, 2012 [Online]. Available: https://doi.org/10.2514/6.2012-1818

[13] 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

[14] E. Bugnion, J. Nieh, and D. Tsafrir, Hardware and software support for virtualization. San Rafael, CA: Morgan & Claypool Publishers, 2017 [Online]. Available: ProQuest

[15] J. Hammerstein and C. May, "The CERT approach to cybersecurity workforce development," Carnegie Mellon Univ Softw. Eng Inst, Pittsburg, PA, USA, Rep. CMU/SEI-2010-TR-045, 2010 [Online]. Available: https://doi.org/10.21236/ADA537055

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

[17] D. D. Updyke, G. B. Dobson, and T. G. Podnar, "GHOSTS in the machine: a framework for cyber-warfare exercise NPC simulation," Carnegie Mellon Univ Softw. Eng Inst, Pittsburg, PA, USA, Rep. CMU/SEI-2018-TR-005, 2018 [Online]. Available: https://resources.sei.cmu.edu/asset_files/TechnicalReport/2018_005_001_534326.pdf

[18] M. Baker, A. Galyardt, and D. Ross, "Federal Virtual Training Environment (FedVTE)," Carnegie Mellon Univ Softw. Eng Inst, Pittsburg, PA, USA, 2017 [Online]. Available: https://resources.sei.cmu.edu/asset_files/WhitePaper/2017_019_001_499755.pdf

[19] G. F. Belli, "Extensible simware architecture for flexible training scenarios," M.S. thesis, Dept. of Comp. Sci., NPS, Monterey, CA, USA, 2016.

[20] E. S. Lowney, “Network communications protocol for the Malicious Activity Simulation Tool (MAST),” M.S. thesis, Dept. of Comp. Sci., NPS, Monterey, CA, USA, 2015.

[21] S. Chapman, R. Smith, L. Maglaras, and H. Janicke, “Can a network attack be simulated in an emulated environment for network security training?,” J. Sens. Actuator Netw., vol. 6, no. 3, p. 16, Aug. 2017 [Online]. Available: https://doi.org/10.3390/jsan6030016

[22] 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

[23] 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%20-%202015%20-%20Experimentation%20on%20operational%20cyber%20security%20in%20CRATE.pdf

[24] J. Almroth and T. Gustafsson, “CRATE exercise control – A cyber defense exercise management and support tool,” in *2020 IEEE Eur. Symp. Secur. Priv. Workshop (EuroS&PW)*, pp. 37–45, Sep. 2020, [Online]. Available: https://doi.org/10.1109/EuroSPW51379.2020.00014

[25] F. J. R. Melon, T. Väisänen, and M. Pihelgas, “EVE and ADAM: situation awareness tools for NATO CCDCOE cyber exercises,” *Systems Concepts and Integration (SCI) Panel SCI-300 Specialists’ Meeting on Cyber Physical Security of Defense Systems*, 2018 [Online]. Available: https://www.sto.nato.int/publications/STO%20Meeting%20Proceedings/STO-MP-SCI-300/MP-SCI-300-10.pdf

[26] Tenable, *Solution Brief: ACAS, Powered by Tenable*, 2018 [Online]. Available: https://www.tenable.com/sites/drupal.dmz.tenablesecurity.com/files/uploads/documents/whitepapers/Solution-Brief-ACAS_Powered_by_Tenable.pdf

[27] Tenable Documentation, “Log correlation engines.” Accessed Jul. 21, 2021 [Online]. Available: https://docs.tenable.com/tenablesc/Content/LogCorrelationEngines.htm

[28] J. Galliani, “What is DISA’s Host Based Security System (HBSS)?” Segue Technologies, Jul. 29, 2015 [Online]. Available: https://www.seguetech.com/disas-host-based-security-system-hbss/

[29] Director, Operational Test and Evaluation, “FY 2020 Annual Report,” Washington, DC, USA, 2021 [Online]. Available: https://www.dote.osd.mil/Portals/97/pub/reports/FY2020/dod/2020jrss.pdf?ver=O20HPjGL8Bei8QsCfWNf_w%3d%3d

[30] Zeek, “The Zeek network security monitor.” Accessed Sep. 15, 2021 [Online]. Available: https://zeek.org/.

[31] Oracle, “What’s new in MySQL 8.0,” Oracle, Austin, TX, USA, 2018 [Online]. Available: https://www.mysql.com/why-mysql/white-papers/whats-new-mysql-8-0/

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

[33] Puppet, “Welcome to Bolt.” Accessed Jul. 18, 2021 [Online]. Available: https://puppet.com/docs/bolt/latest/bolt.html

[34] Node.js, “Introduction to Node.js.” Accessed Jul. 11, 2021 [Online]. Available: https://nodejs.dev/learn

[35] MDN Web Docs, “About JavaScript - JavaScript.” Accessed May 7, 2021 [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript

[36] Java, “How is JavaScript different from Java?” Accessed May 7, 2021 [Online]. Available: https://java.com/en/download/help/java_javascript.html

[37] PHP, “PHP: Hypertext Preprocessor.” Accessed Jul. 11, 2021 [Online]. Available: https://www.php.net/

[38] OASIS, “TOSCA Simple Profile in YAML Version 1.2” Accessed September 1, 2021 [Online]. Available: https://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.2/csd01/TOSCA-Simple-Profile-YAML-v1.2-csd01.html

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California