Web CARTT: The Web-based Cyber Automated Red Team Tool

Joseph Berrios, Alan Shaffer, Gurminder Singh
Naval Postgraduate School, Monterey, California, USA
joseph.berrios@nps.edu
alan.shaffer@nps.edu
gsingh@nps.edu

Abstract: Red teaming is a well-established methodology for ensuring and augmenting cyber system security; however, the training, expertise, and knowledge of appropriate tools and techniques required to perform effective red teaming come with a significant cost in time and resources. To address these issues, we have previously developed a "red team in a box" (RTIB) capability, called CARTT (Cyber Automated Red Team Tool), to perform automated red team actions on the internal enterprise network without the need for its users to be experts in this field. This current research has extended CARTT by developing a client/server model system that allows operators to perform red team testing on target networks from a simple remote web interface. Using a command-and-control architecture, the extended CARTT provides the ability for cyber operators and network administrators to identify hosts on a target network, conduct vulnerability analysis on those hosts and the target network, attempt to exploit discovered vulnerabilities based on user selected options, and generate the results of these red teaming actions. Additionally, CARTT now provides a tiered role system, so that higher level "commander" users can direct and monitor the actions and results of subordinate "operator" users; as well, the system provides an "administrator" management role. By providing a simple user interface that automates interaction with the underlying tools, operators are able to utilize CARTT without extensive training or experience in red team operations. The ease of use and reliance on open source software greatly reduces the requirements for organizations to use this tool for red teaming their networks.

Keywords: Red team, defensive cyber operations, automated vulnerability analysis, web-based assessment.

1. Introduction

Red teaming has a proven history of ensuring cyber security when it is conducted and documented properly. However, the training, expertise, and knowledge of appropriate tools and techniques required to perform effective red teaming come with a significant cost in time and resources. As a result, the Office of the Secretary of Defense (OSD) has sponsored research into developing an automated red team tool that performs automated red teaming actions without need for its user to be an expert in this field. The goal of this research has been to design and implement a software tool that achieves this objective.

Red teaming requires an in depth understanding of specialized systems, tools, and techniques that allow one to fully assess a target system for vulnerabilities that may be exploitable by adversaries, as well as the target's ability to withstand and respond to those exploitations. Developing red team expertise through training, however, is expensive and time consuming, and many of the tools used for these purposes are expensive or lack intuitive interfaces for non-expert users. These constraints create high barriers for large organizations such as the Department of Defense (DoD) to fully train and maintain adequate red team experts to test their massive number of systems (Director, Operational Test and Evaluation, 2019; DoD Office of Inspector General, 2020).

This work extends previous research on the Cyber Automated Red Team Tool (CARTT) (Plot, 2019; Edwards, 2019), by developing a highly automated client/server system that enables non-expert users to perform network discovery, vulnerability analysis, and exploitation testing using open source software on a target network. Research was conducted on the techniques used to implement client/server architectures with strong authentication, as well as on toolkits and products, both commercial and open source, that could be employed within the system to support automated red teaming.

CARTT functionality and features were extended in order to transition it from a stand-alone application to a multi-user, multi-platform client/server model. Additional research was conducted on OpenVAS, PHP, Metasploit, Python, and mySQL in designing and implementing the new version of CARTT to ensure that critical features could be brought forward into the new system. Once the CARTT client/server model was complete, an assessment of its functionality and feature set was performed to determine the benefits and/or shortcomings of the new implementation.

Section 2 of this paper describes previously work related to this research. Section 3 describes the design and architecture of the CARTT system, and Section 4 details the test environment and functionality of CARTT. Section 5 concludes with a discussion of the results of this research and future work.

2. Background

The Department of Defense Information Network (DoDIN) is under constant barrage from malicious actors, as demonstrated by the fact that the Defense Information Systems Agency (DISA) blocked an average of 300 million malicious actions every day in 2019 (Johnson, 2019). By necessity, the DoD has refocused its efforts on improving the security of its networks and cyber systems. The DoDIN is a critical component of the U.S. military's ability to project power and conduct operations. It is massive in scope and globally dispersed, consisting of networks on ships, aircraft, satellites, cloud services, industrial control systems (ICS), tactical communication systems, mobile devices, and commercial and private infrastructure. There are approximately 15,000 networks inside the DoDIN that provide service for ~3 million users spread across 3,500 locations in 26 different countries (Craft, 2019; DISA, 2019).

While the DoD is making significant efforts to improve cybersecurity of the DoDIN, it cannot mitigate all of the existing vulnerabilities present in often decades-old fielded systems. The Government Accountability Office (GAO, 2018) found that the DoD more often required security in traditional IT systems than in weapon systems, not fully understanding the vulnerabilities associated with weapon systems being integrated into the DoDIN. As a result, the majority of currently fielded systems have little to no cybersecurity in place, increasing the risk to the entire DoDIN. Additionally, there are legacy systems and applications throughout the DoDIN that no longer receive vendor support or patching but are still allowed to remain in service due to the lack of a suitable replacement (GAO, 2018).

To address the vulnerabilities of aging DoDIN systems and emerging cyber threats, the DoD has turned to red teaming. A red team is comprised of highly educated and experienced cybersecurity professionals who attempt to penetrate and exploit computer networks using the tactics, techniques, and procedures (TTPs) of real-world cyber threats. Their goal is to discover vulnerabilities and gaps in the security posture, training, and system configurations of their own systems so that corrective or remediation actions can be taken to prevent a malicious actor from penetrating the network. Additionally, red team activities tend to increase the response and reaction proficiency of system defenders in detecting, diagnosing, and mitigating cyber-attacks.

The GAO (2018) found that almost every major acquisition program tested in the previous five-year period had mission-critical cyber vulnerabilities; the OSD Director, Operational Test and Evaluation (DOT&E, 2019) had similar results in their review of the DoDIN. Examples of common vulnerabilities included use of weak or default passwords to critical systems and accounts, and a lack of or improper application of encryption systems. These are rudimentary security measures for which the National Institute of Technology and Standards (NIST) provides guidance (NIST, 2017). These results highlight the importance of performing red team assessments to ensure that Program Offices are properly implementing cybersecurity controls in new systems integration.

The primary threats to the DoDIN are nation-state level actors, which are typically more advanced, harder to detect, and more persistent than hacktivists or script kiddies. In order to adequately defend the DoDIN, its network defenders must be highly trained to identify, respond to, and mitigate cyber-attacks. As such, the DOT&E and the GAO both rely heavily on DoD Red Team assessments to gain perspective on the status of the DoDIN's cybersecurity. Red teaming offers a unique opportunity for local defenders to "fight the network" as they get an opportunity to learn how their systems, firewalls, and anti-virus and intrusion prevention/detection systems respond to red team TTPs. These actions improve a defender's ability to recognize and respond to real-world events, and provide feedback about what mitigation efforts were most and least successful, further improving the response capability of local defenders.

Even from a non-technical perspective, red teams provide valuable insights into the security posture of the DoDIN. The DoD mandates that all personnel complete annual cyber-security training, which includes information on detecting and preventing insider threats and phishing attacks; this training can normally be completed in less than one hour (DISA, 2020). One of the DoD Red Teams, in a publicly available interview, stated that, "Most cyber-attacks are user driven… [they're] easiest to get at and yield the most reliable results.

We've never had a phishing campaign that has failed" (Piedfort, 2018). This statement, even if somewhat embellished, demonstrates a shortcoming in the mandated training.

The first indications of a compromise or data breach will likely be well after an adversary has conducted a cyber-attack. IBM assesses the average time in the private sector between a breach and its detection to be 279 days (IBM, 2020). For the DoD, where lives and national security are dependent on operational security and information superiority, that type of lag in detection can have disastrous consequences.

A client/server red teaming system allows the DoD to avoid vulnerabilities present in client-side systems. Using a web-based client removes the requirements for every command node to manage its own system software configuration and updates, and greatly reduces the cost of deploying the tool to thousands of sites at the enterprise level. Additionally, the only device that performs red teaming functions would be a centrally managed server, eliminating the requirement to add exceptions or elevated access control permissions to potentially thousands of devices or user accounts across the enterprise. This greatly reduces the risk of the tool being compromised by drastically reducing its attack surface.

This research was conducted using open source tools, thus eliminating licensing costs and offering improved transparency of the underlying systems, as compared to commercial products. Additionally, while the CARTT system is highly automated from a user perspective, the system owner still has the ability to create custom exploits and payloads that less technically skilled users would be able to leverage.

3. Design Methodology

CARTT allows local network administrators to conduct a red team audit of their network through a simple user interface (UI) that automates interactions with the Metasploit Framework (MSF), a tools suite for developing and running exploit shellcode against remote targets; OpenVAS, an open-source vulnerability scanning and analysis tool; and other specialized red teaming and pen-testing tools. This design removes the need for CARTT users to be experts in these tools, and alleviates the requirement to deploy these tools on every network.

The CARTT Client is designed to support three different user roles: Administrator, Commander, and Operator. The CARTT Administrator manages user accounts, including setting appropriate roles, and performing password resets and account unlock. The Administrator is also responsible for maintaining the commands table in the database. The CARTT Commander role primarily exists for users at headquarters who do not perform red team assessments themselves, but who direct the actions of CARTT Operators. CARTT provides Commanders with a message channel to send tasking and receive communication from CARTT Operators. For example, information on a target network and specific vulnerabilities of interest can be passed to Operators from Commanders.

The Client user logs in to CARTT via a client/browser interface and selects their role for the session. When logged in as an Operator, the user can proceed to configure and conduct a scan of a target network (such as their organization's local network). To do this the Operator enters some basic information, such as the IP range and a unique name for the scan, then directs CARTT to perform a vulnerability analysis on the IP range, returning the results via the CARTT Client.

Once the scan has completed, the Operator has additional options available, for example, with a few simple webforms and clicks, they can select a host, vulnerability, and payload to audit. The Client provides information on the selected vulnerability and exploit modules, allowing the Operator to choose the best options based on their operational objective. Once satisfied with the selections, the Operator can initiate the audit on the target host.

CARTT then launches the exploit with the appropriate options set by the Operator, then collects the results and returns them via the Client. The Operator may choose to run additional exploits, then review and save their results. To run additional exploits the Operator can select different options for host, vulnerability, and module, which enables them to perform multiple tests without having to navigate multiple menus and subsystems.

3.1 System Implementation

3.1.1 Architecture

CARTT is designed so that a single server runs all of the processes and applications required to perform a red team assessment; we will refer to this system as the "CARTT Server". As described earlier, the CARTT Client has three user roles: Operator, Commander, and Administrator. The CARTT Client is the web-based interface through which a user interacts with the functionality provided by the CARTT Server. The Target Network consists of any internet-connected devices on an IP range provided by the Operator. The CARTT Server, CARTT Client(s), and Target Network must be connected via the internet. A sample notional CARTT architecture, with each of the elements described above, is shown in Figure 1. Note that the user may access the CARTT Client from within the Target Network, but this is not necessary (as depicted in Figure 1).

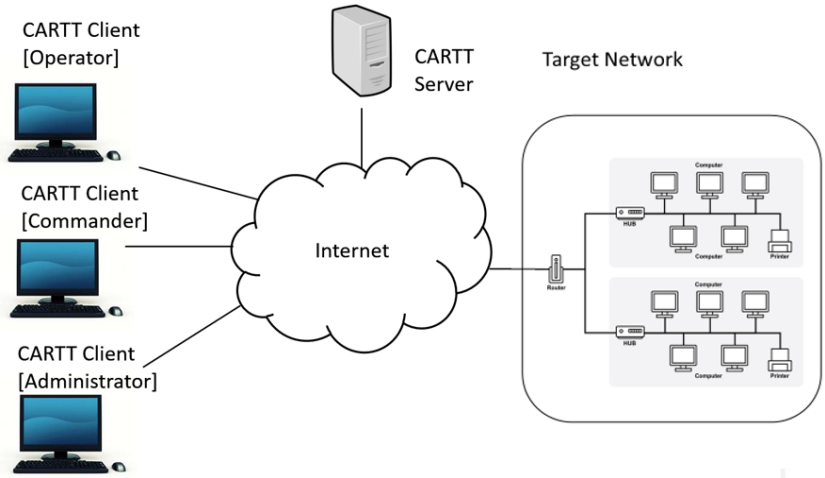

Figure 1: Sample CARTT Architecture

3.1.2 CARTT Server Design

The CARTT Server consists of a PHP server, a series of PHP scripts, the OpenVAS and MSF programs (each with related backend databases), a report folder, and a mySQL database. A diagram of the CARTT Server is shown in Figure 2, and each of the Server elements is described further in this section.

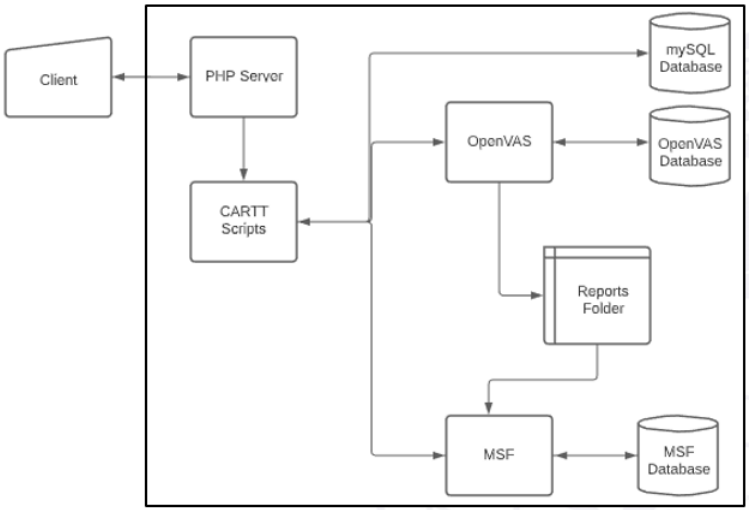

Figure 2: CARTT Server Diagram

The PHP server is the core of CARTT, which is defined by a series of PHP scripts; the scripts that direct user actions by accepting input and providing feedback comprise the CARTT Client. PHP was chosen as the scripting language and webserver because it is a free, server-side language that is platform-independent and well-documented online. PHP is an interpreted language, making it easy to implement, test, and revise in real-time. Additionally, it has built-in features to handle dynamic content, integrate with various databases, and perform web security functions. Running PHP on the server side is essential, since the other required programs reside are on the CARTT Server.

OpenVAS is an open source vulnerability scanning and analysis tool provided by Greenbone Networks (the tool was recently rebranded as Greenbone Vulnerability Management, or GVM). It was selected for CARTT primarily due to its ability to integrate with MSF and its free cost. OpenVAS has a robust set of customization options, allowing the user to tune a vulnerability scan to their specific needs. OpenVAS also supports a variety of report formats, allowing users to analyze scan results directly, or to channel them to some other analyzer. Additionally, Linux distributions such as Kali and Ubuntu maintain packages or come pre-installed with OpenVAS, making it easy to configure and run the program. OpenVAS uses a PostgreSQL or SQLite database to store user information, report data, and scan, task, and target configuration data. OpenVAS also uses a Redis database to store temporary data during scans.

CARTT uses MSF to parse the vulnerability report from OpenVAS, perform vulnerability exploitation, and interact with the OpenVAS scanner. MSF is well-documented online and widely used, which, along with the built-in integration with OpenVAS, were key factors in deciding to use it in CARTT. MSF itself relies on a PostgreSQL database to store imported scan data, such as hosts and vulnerabilities found in an OpenVAS scan. Finally, Rapid7 provides a free version of MSF, which keeps the project at zero cost.

Since CARTT is a role-based system, users must be allowed to send messages amongst themselves, which requires some type of dedicated messaging database. We chose mySQL for this because it is free, open source, well documented, works on numerous operating systems, and has robust PHP integration. There are also mySQL tools available that allow GUI-based management, simplifying the initial setup and configuration of the database. The mySQL database processes queries quickly and is designed to be scalable, both essential for wide deployment of CARTT.

In order to manage the flow of completed vulnerability scans from OpenVAS to MSF, CARTT instructs OpenVAS to save a local copy of the specified scan on the Client system. CARTT then commands MSF to import the scan from the saved local file. The scan is saved in a folder in the local directory, which allows CARTT to use relative paths to download and import the scan reports.

3.1.3 CARTT Client Design

Individual CARTT users may have access to multiple roles depending on their position or duties. Separating CARTT users into roles makes it easier to conceptualize the relationships that exist between users at various levels in the CARTT Client. Upon logging in, if the user possesses multiple role permissions, they are presented with a choice of roles and must select only one for a particular session. This role selection determines what functionality is available to the user. If the user possesses only one role in CARTT, they are automatically directed to the menu screen relevant to that role.

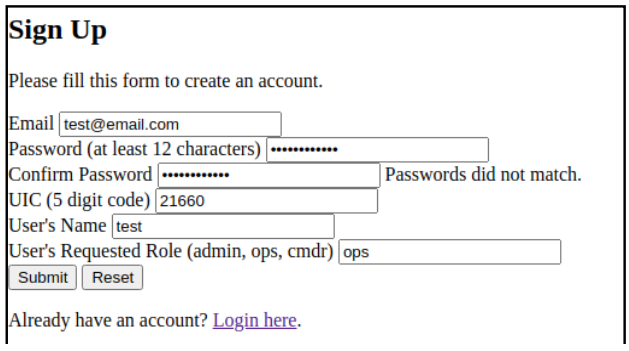

Figure 3: CARTT Registration page

In order to maintain simplicity, radio buttons are used whenever possible to allow the user to select desired actions. Text fields are used only when non-standard user input is needed, and such fields are labelled and programmed to provide feedback if the user input is invalid or improper. An example of the Registration "Sign Up" page is shown in Figure 3. Note that it shows feedback to the user for an invalid password entry.

4. Scenario and Funcationality

In order to test CARTT, we assumed a user in the Operator role and assigned a target network to assess. The assessment entailed conducting a vulnerability scan on the target network, importing the scan results, selecting a target host and vulnerability from the scan, and finally, selecting an exploit module to test against the target host/vulnerability. To facilitate the test an isolated target range was created consisting of 32 virtual machines running various operating systems, including Windows versions XP, 7, and 10. A CARTT Server was deployed on the target network and a separate Windows device was used as the CARTT Client.

4.1 Operator Process Flow

The process flow for an Operator logging into CARTT and attempting exploitation of a target host is shown in Figure 4. Every interaction with CARTT starts in the same way, with the user logging in to the CARTT Client via web-based interface. The CARTT Server validates the user's credentials and redirects the user to the correct menu page based on their session role.

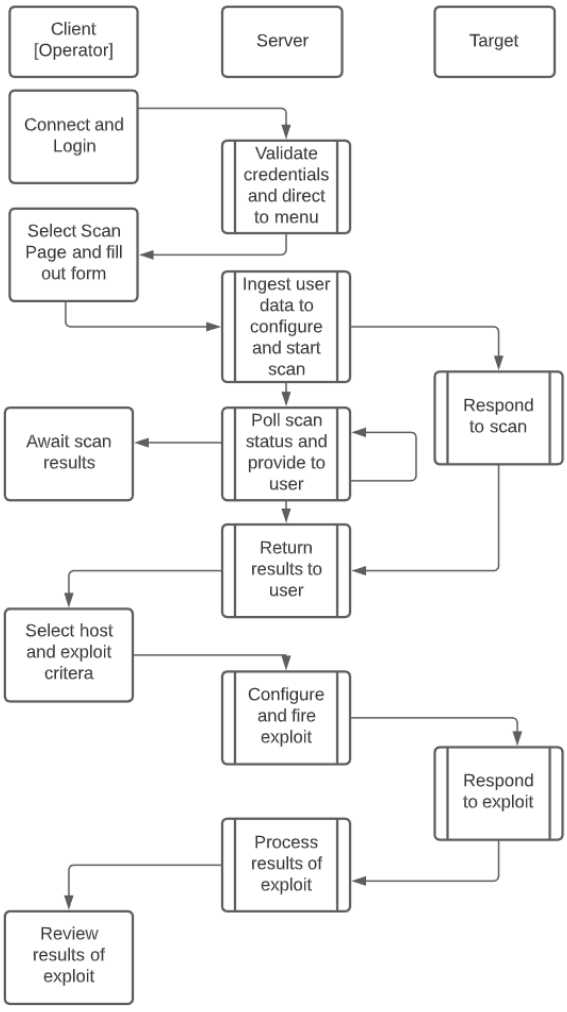

Figure 4: CARTT Operator Process Flow

To start an assessment the Operator fills in a webform to configure and conduct a scan. The data provided by the Operator is used by the CARTT Client to create the commands necessary for the CARTT Server to direct OpenVAS to conduct a vulnerability scan of the target network. As part of its scan, OpenVAS first conducts host discovery on the network. The Operator can close the browser once the scan has been initiated and return later with no loss of continuity. The Operator Menu also allows the user to enter the name of a scan, which the Client will use to query OpenVAS for scan status information. Once the scan is completed, the Operator will be directed to a Scan Import page.

On the Scan Import page, the Operator instructs CARTT to import a scan by name. The CARTT Server first downloads the scan as an Extensible Markup Language (XML) file from OpenVAS, then imports that file into MSF as a PostgreSQL workspace designated by the Operator. The Operator is then presented a list of hosts and vulnerabilities, based on the scan results. In order to improve search efficiency and page response times the hosts and vulnerabilities are also stored on the CARTT Server as text files. This allows the Client to display the host and vulnerability data without needing to reconnect to MSF.

After MSF imports the vulnerability scan, the Client presents the Operator with a page to select the host, vulnerability, and MSF exploit module from available choices. Unlike the host and vulnerability information, an MSF connection is required to search for the appropriate modules, which causes several seconds of delay between selecting a vulnerability and the Operator being presented with the list of matching modules. Once the Operator selects a module the Client packages the inputs into a resource (.rc) file for the CARTT Server to pass to MSF as an argument.

```
use exploit/windows/rdp/cve_2019_0708_bluekeep_rce
set RHOST 17.10.42.124
set ExitOnSession true
set color 'false'
spool user_data/results_qwert.txt
exploit -z
exit -y
```

Figure 5: Sample .rc File

An example .rc file is shown in Figure 5, with the selected exploit module in this case being BlueKeep Remote Code Execution. This .rc file directs MSF on which module to use, which remote host (RHOSTS) to target, to exit the session once connected, and to print the output to a text file that is labelled with the Operator's login (in this example, 'qwert' was the login ID for the Operator). Once MSF has been properly configured, the exploit is attempted with the '-z' option, which ensure that any connections to the target are closed upon completion of the exploit. Once the test is complete, the Operator gets feedback on the Client from the spooled output. An example of the output provided by the Client is shown in Figure 6; in this case the exploit module leveraged BlueKeep to perform a psexec on the target. As shown in Figure 6, an MSF Meterpreter Shell session is created between the CARTT Server and the target host.

Attempting to use exploit/windows/smb/ms17_010_psexec on host 17.10.42.239

Attempting Exploit.....

Results:

```
[*] Spooling to file user_data/results_qwert.txt...
resource (user_data/exploit_qwert.rc)> exploit -z
[*] Started reverse TCP handler on 17.10.42.120:4444
[*] 17.10.42.239:445 - Target OS: Windows 5.1
[*] 17.10.42.239:445 - Filling barrel with fish... done
[*] 17.10.42.239:445 - <---------- | Entering Danger Zone | ---------->
[*] 17.10.42.239:445 - [*] Preparing dynamite...
[*] 17.10.42.239:445 - [*] Trying stick 1 (x86)...Boom!
[*] 17.10.42.239:445 - [+] Successfully Leaked Transaction!
[*] 17.10.42.239:445 - [+] Successfully caught Fish-in-a-barrel
[*] 17.10.42.239:445 - <---------- | Leaving Danger Zone | ---------->
[*] 17.10.42.239:445 - Reading from CONNECTION struct at: 0x81210010
[*] 17.10.42.239:445 - Built a write-what-where primitive...
[+] 17.10.42.239:445 - Overwrite complete... SYSTEM session obtained!
[*] 17.10.42.239:445 - Selecting native target
[*] 17.10.42.239:445 - Uploading payload... FkcPltNg.exe
[*] 17.10.42.239:445 - Created \FkcPltNg.exe...
[+] 17.10.42.239:445 - Service started successfully...
[*] 17.10.42.239:445 - Deleting \FkcPltNg.exe...
[*] Sending stage (175174 bytes) to 17.10.42.239
[*] Meterpreter session 1 opened (17.10.42.120:4444 -> 17.10.42.239:1935) at 2020-10-27 14:03:32 -0700
[*] Session 1 created in the background.
resource (user_data/exploit_qwert.rc)> exit -y
```

Figure 6: Sample Target Exploitation

4.2 Operator Requirements

This prototype test demonstrated that CARTT significantly reduced the required level of expertise that an Operator would encounter if trying to use OpenVAS and Metasploit to achieve the same results. An Operator with little or no experience in red teaming is able to use CARTT to conduct a vulnerability scan of a target network, and to attempt exploitation of hosts on that network with no knowledge of how the underlying CARTT systems function. The only information the Operator needed to conduct the vulnerability scan was the IP range of the target network, which could be provided through the built-in CARTT Messaging interface.

For exploitation, CARTT provided the Operator with the necessary information by allowing them to choose from a list of cyber actions, based on the results of a vulnerability scan. In order to construct and test an exploit Operators must be familiar with the Common Vulnerabilities and Exposures (CVE) database, however, we expect that most system administrators should meet this requirement. Additionally, the CARTT Messaging interface can be used to provide the Operator with the specific CVE(s) that require testing.

5. Conclusions

The purpose of this research was to develop an automated client/server red teaming tool to enable non-expert users to perform red teaming processes. We concluded, based on our prototype system testing, that CARTT accomplishes this goal. The CARTT user requires only minimal information about the target network in order to conduct a full vulnerability scan, and subsequently test exploits against hosts on the target network. The information gained from CARTT can be used to identify and mitigate cyber security risks for the DoDIN, in support of, though not replacing, full red team analysis. While no automated red team tool can replace a fully formed cyber red team, it can allow local network administrators to conduct limited red team operations in

support of compliance, remediation, and training. This would allow the DoD Red Teams to focus more of their resources and expertise on emulating real-world adversary threats.

In the current prototype the CARTT Client has a basic user interface that presents the Operator with a minimum amount of target host information. An extension of the user interface and database, to include previous session results, would allow the CARTT Client to gather more information from target network analysis, and reduce the amount of time spent polling MSF for information. Additionally, CARTT would provide more value to the Operator if it was able to provide information about the exploit being tested, particularly offering mitigation measures.

Since MSF supports custom module development, the system owner for CARTT is capable of creating and adding new exploits into CARTT's instance of MSF that CARTT Operators would be able to access. This would allow CARTT to be used as a system to test the functionality or reliability of experimental exploits.

6. Acknowledgements

This material is based upon work supported by the Naval Research Program. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Naval Research Program.

References

Craft, C. (2019) "Fight the DoDIN", [online], DISA, https://www.disa.mil/-/media/Files/DISA/News/Events/Symposium-2019/1---COL-Craft_Fight-the-DODIN_approved-Final.ashx

Department of Defense Office of Inspector General. (2020) *Followup Audit on Corrective Actions Taken by DOD Components in Response to DOD Cyber Red Team-Identified Vulnerabilities and Additional Challenges Facing DOD Cyber Red Team Missions,* Department of Defense Office of Inspector General, Washington DC.

Director, Operational Test and Evaluation. (2019) *FY 2019 Annual Report* Director, Operational Test and Evaluation, Washington DC.

National Institute of Standards and Technology. (2017) *NIST Special Publication 800-63B Digital Identity Guidelines,* U.S. Department of Commerce, Washington, DC.

Defense Information Systems Agency. (2019) "DISA Fact Sheet", [online], DISA, https://www.disa.mil/-/media/Files/DISA/Fact-Sheets/DISA-Capabilities.ashx?la=en&hash=058FFF28911BFA504B699B2D3642AB82C292F482

Defense Information Systems Agency. (2020) "Cyber Awareness Challenge", [online], DISA, https://public.cyber.mil/training/cyber-awareness-challenge/

Edwards, P. (2019) "Cyber Automated Red Team Tool", [online], Naval Postgraduate School, Monterey, CA, https://calhoun.nps.edu/handle/10945/64145

IBM. (2019) "Cost of a Data Breach Report", [Online], IBM, https://www.ibm.com/security/data-breach

Johnson, M. (2019) "DISA seeks automated tool to identify potential cyber threats, conduct analysis", DISA, [online], https://disa.mil/NewsandEvents/2019/automated-tools-cyber-threats-analysis

Piedfort, S. (2018) "SSC Atlantic Red Team: The Good 'Bad Guys'", [Online], Navy Imformation Warfare Systems Command, https://www.navy.mil/submit/display.asp?story_id=104650

Plot, J. (2019) "Red Team in a Box(RTIB): Developing Automated Tool to Identify, Assess, and Expose Cybersecurity Vulnerabilities in Department of the Navy Systems", [online], Naval Postgraduate School, Monterey, CA, https://calhoun.nps.edu/handle/10945/62832

United States Government Accountability Office. (2018) *Weapon Systems Cybersecurity DOD Just Beginning to Grapple with Scale of Vulnerabilities,* United States Government Accountability Office, Washington DC.