2022-09

BEHAVIORAL CHARACTERIZATION OF ATTACKS ON THE REMOTE DESKTOP PROTOCOL

Ramirez, Ryan

Monterey, CA; Naval Postgraduate School

http://hdl.handle.net/10945/71026

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

BEHAVIORAL CHARACTERIZATION OF ATTACKS ON THE REMOTE DESKTOP PROTOCOL

by

Ryan Ramirez

September 2022

Thesis Advisor:	Thuy D. Nguyen
Co-Advisor:	Neil C. Rowe

Approved for public release. Distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** September 2022	**3. REPORT TYPE AND DATES COVERED** Master’s thesis

4. TITLE AND SUBTITLE BEHAVIORAL CHARACTERIZATION OF ATTACKS ON THE REMOTE DESKTOP PROTOCOL	**5. FUNDING NUMBERS**
6. AUTHOR(S) Ryan Ramirez	
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 Remote Desktop Protocol (RDP) is popular for enabling remote access and administration of Windows systems; however, attackers can take advantage of RDP to cause harm to critical systems using it. Detection and classification of RDP attacks is a challenge because most RDP traffic is encrypted, and it is not always clear which connections to a system are malicious after manual decryption of RDP traffic. In this research, we used open-source tools to generate and analyze RDP attack data using a power-grid honeypot under our control. We developed methods for detecting and characterizing RDP attacks through malicious signatures, Windows event log entries, and network traffic metadata. Testing and evaluation of our characterization methods on actual attack data collected by four instances of our honeypot showed that we could effectively delineate benign and malicious RDP traffic and classify the severity of RDP attacks on unprotected or misconfigured Windows systems. The classification of attack patterns and severity levels can inform defenders of adversarial behavior in RDP attacks. Our results can also help protect national critical infrastructure, including Department of Defense systems.

14. SUBJECT TERMS remote desktop protocol, RDP, attack, characterization, industrial control systems, ICS, honeypot, deception	**15. NUMBER OF PAGES** 129
	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

THIS PAGE INTENTIONALLY LEFT BLANK

BEHAVIORAL CHARACTERIZATION OF ATTACKS ON THE REMOTE DESKTOP PROTOCOL

Ryan Ramirez
Civilian, CyberCorps: Scholarship for Service
BS, California State University, Monterey Bay, 2019

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 2022

Approved by: Thuy D. Nguyen
Advisor

Neil C. Rowe
Co-Advisor

Gurminder Singh
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The Remote Desktop Protocol (RDP) is popular for enabling remote access and administration of Windows systems; however, attackers can take advantage of RDP to cause harm to critical systems using it. Detection and classification of RDP attacks is a challenge because most RDP traffic is encrypted, and it is not always clear which connections to a system are malicious after manual decryption of RDP traffic. In this research, we used open-source tools to generate and analyze RDP attack data using a power-grid honeypot under our control. We developed methods for detecting and characterizing RDP attacks through malicious signatures, Windows event log entries, and network traffic metadata. Testing and evaluation of our characterization methods on actual attack data collected by four instances of our honeypot showed that we could effectively delineate benign and malicious RDP traffic and classify the severity of RDP attacks on unprotected or misconfigured Windows systems. The classification of attack patterns and severity levels can inform defenders of adversarial behavior in RDP attacks. Our results can also help protect national critical infrastructure, including Department of Defense systems.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVIATIONS

CPU	Central Processing Unit
DCS	Distributed Control System
DNS	Domain Name System
GCC	Generic Conference Control
GPU	Graphics Processing Unit
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
ICS	Industrial Control System
MSF	Metasploit Framework
MCS	Multipoint Communication System
PCAP	Packet Capture
PDU	Protocol Data Unit
PLC	Programmable Logic Controller
RDP	Remote Desktop Protocol
SCADA	Supervisory Control And Data Acquisition
SSL	Secure Socket Layer
SSH	Secure Shell
TPDU	Transport Protocol Data Unit
VM	Virtual Machine
VNC	Virtual Network Computing

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank Professor Nguyen and Professor Rowe for their guidance throughout the development of this thesis. I faced many new challenges that helped me to grow as a researcher and as a computer scientist.

I would like to thank my friends and family. Through the difficult times, late nights, and all-nighters of my educational journey, I could always think of them and know that I was supported.

Finally, I would like to thank former Scholarship for Service student and NPS graduate Casandra Martin. From her guidance, I learned of SFS and NPS, which ultimately put me on the path to earn a graduate-level degree.

This material is based upon activities supported by the National Science Foundation under Agreement No. 1565443. Any opinions, findings, and conclusions or recommendations expressed are those of the author and do not necessarily reflect the views of the National Science Foundation.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. MOTIVATION

Remote desktops are interactive graphical user interfaces of remote machines that can access and control remote systems from a local environment. Industrial control systems (ICSs), which control services for water, electricity, gas, transportation, and manufacturing, often use remote-desktop software to supervise and control industrial processes. Though it has been used for over two decades, remote-desktop software has become more popular since COVID-19 caused people to work from home. Remote-desktop software helps administer remote machines, but it creates additional system vulnerabilities which may be exploited for unauthorized access to systems. In February 2021, TeamViewer remote-desktop software was exploited to access a municipal water treatment plant in Florida and remotely sabotage water treatment by adding sodium hydroxide to the water (CISA, 2021).

Microsoft's Remote Desktop Protocol (RDP) is a popular choice for implementing remote access to cyber systems since it comes preinstalled on most versions of Windows. With its presence on millions of machines, increased use since COVID-19 lockdowns, and known vulnerabilities in the protocol and application software, many security exploits have taken advantage of systems that use RDP. Past exploits of RDP have allowed attackers to access confidential information, deny service, and execute code on vulnerable machines. When RDP is used for critical infrastructure, adversaries can exploit it in cyberattacks to gain control over critical systems and industrial processes.

The discovery and analysis of adversarial tactics in cyberattacks can help prevent future attacks. One way that data can be effectively collected for analysis is through honeypots, decoy systems that mimic the behavior of real systems. An attacker interacting with a honeypot will generate data for defenders to collect and analyze to gain intelligence on attack methods and techniques. To better understand the methods used for attacks on RDP and the remote hosts that use it, a honeypot can use RDP as its only access method. Also, it can simulate a more attractive target like an ICS.

B. RESEARCH PLAN

This research developed ways to find signatures of attacks on RDP and using it to exploit or harm a remote Windows host. We collected instrumented and actual attack data on an ICS honeypot at our school. We used the data to develop methods to classify attacks. We correlated Windows event logs and attack signatures to evaluate our methods for characterizing RDP activity.

C. THESIS OUTLINE

Chapter II provides background information on the key concepts of remote-desktop software, industrial control systems, and honeypots. Chapter III describes the RDP remote-desktop software that we studied in this work, the issues in RDP attack characterization, and our approach for classifying signatures of RDP attacks. Chapter IV describes our experiments, the methods used for data generation and analysis, and the attack-characterization methodology. Chapter V discusses the results of our experiments and the effectiveness of our methodology. Chapter VI gives our conclusions and recommendations for subsequent research.

II. BACKGROUND

This chapter provides background information on remote-desktop software, honeypots, and ICSs, the components of our research. We also describe previous honeypot experiments to which our research relates.

A. REMOTE DESKTOP ACCESS

1. Common Methods of Remote Desktop Access

Remote desktop access is connection to a remote machine that allows a user to interact with its desktop environment as though it were local. The degree of interaction offered to a user can range from a command-line interface to access of the machine's graphical desktop user interface. Use cases of remote desktop access include people accessing their home computers from remote locations, employees remotely logging into their office computers to work from home, and administrative staff remotely logging in to client machines to fix problems.

Common methods of remote desktop access are network protocols such as Secure Shell (SSH), Telnet, Microsoft Remote Desktop Protocol, and Virtual Network Computing (VNC). Third-party applications such as TeamViewer (TeamViewer, n.d.), RemotePC (IDrive, 2022), and Zoho Assist (Zoho, 2022) that use their own proprietary protocols can also support it. Establishing a remote desktop connection requires a special application running on the remote machine. The service listens for incoming connections from a user's client and connects the client and the remote desktop service (Microsoft, 2020b).

2. Microsoft Remote Desktop Protocol

This thesis focused on the Microsoft Remote Desktop Protocol (RDP) in a honeypot environment. RDP is a proprietary protocol developed by Microsoft for Remote Desktop Services (formerly called Terminal Services) and compatible remote-desktop clients (Microsoft, 2020a). Connections between remote-desktop client software (such as Microsoft Remote Desktop Connection) and a Remote Desktop Services host are enabled by RDP through TCP port 3389 (Microsoft, 2010). RDP can also be used by non-Windows systems; remote desktop clients exist for Mac and UNIX-based machines (Microsoft, 2022c) and the open-source Xrdp server can be deployed on Linux computers (NeutrinoLabs, 2022).

RDP uses the T.120 suite of multimedia conferencing protocols of the International Telecommunications Union (Microsoft, 2021d). It uses multiple virtual channels for data transmission of the server's display data to the client and the client's keyboard and mouse data to the server (Microsoft, 2020a). The RC4 stream cipher secures data in transit (Microsoft, 2020a). Other notable features of RDP are bandwidth reduction by caching and compression, which can improve performance by reducing the amount of data sent over a network, and clipboard mapping, which allows a user to copy and paste objects between applications on their local machine and on the remote machine (Microsoft, 2020a). Besides counting the number of packets sent by a client, determining the origin of remote-desktop traffic, and checking the duration of a remote-desktop session, RDP traffic analysis may include identifying the creation of specific virtual RDP channels and decrypting RDP traffic for video replay of remote sessions and payload analysis.

RDP is also popular among attackers since it opens several avenues of attack to the machines and networks that use it. Brute-force or credential-stuffing attacks can be effective against systems with weak RDP credentials (Cloudflare, 2022). Also, vulnerabilities in Microsoft Remote Desktop Services and the RDP protocol itself have recently enabled BlueKeep and DejaBlue remote-code execution exploits which affect several versions of Windows running Remote Desktop Services (Buchanan, 2019). Since these exploits are also wormable (i.e., they can spread without human aid), they are often associated with EternalBlue, a Windows remote code-execution exploit which attacks the SMBv1 protocol; it enabled the propagation of the WannaCry and NotPetya ransomware in 2017, causing billions of dollars' worth of damage worldwide (CISA, 2017). Even with Microsoft releasing patches for BlueKeep and DejaBlue, and with the National Security Agency issuing an advisory on BlueKeep exploitation, millions of unpatched machines likely remain vulnerable to these RDP-based attacks (CISA, 2019).

Following the discovery of the BlueKeep remote code-execution vulnerability and motivated by its potential to impact critical infrastructure and public services through ransomware, a 2019 study quantified the global threat to RDP (Boddy et al., 2019). By deploying ten geographically distributed RDP honeypots, researchers measured how quickly machines with Internet-facing RDP services could be discovered, the frequency of daily attacks, and the usernames commonly associated with login attempts (Boddy et al., 2019). Researchers discovered different levels of persistence in attacks on remote-desktop tools, with some attackers attempting to log in just a few times before giving up, and others spending days trying to figure out the machine's administrator credentials. While impacts to industrial control systems motivated the study, the honeypots used in the research only offered a login screen to attackers and did not try to emulate an industrial system or public service. In the honeypot deployed for our research, attackers can interact beyond a login screen.

A 2021 study compared RDP honeypots for capturing and analyzing attacker behaviors (Ahlman, 2021). The experiment deployed two honeypot designs. The first design, called the XRDP honeypot, used a Linux machine running Xrdp to offer RDP services, a machine for packet capture and traffic forwarding, and an analysis machine running RDP Replay to view captured RDP sessions (Ahlman, 2021). The second design, called the PyRDP honeypot, used a Windows Server to offer RDP services and the open-source PyRDP software on an Ubuntu machine, to capture and analyze RDP sessions (Ahlman, 2021). The research concluded that the XRDP honeypot was effective for the Linux platform as an alternative to Windows-based RDP honeypots.

B. CYBERDECEPTION

A major challenge for defenders is the advantage that attackers have in the preparation and timing of an attack (Climek et al. 2016). However, through cyberdeception, defenders can make attacking a cyber system harder by intentionally misleading attackers. For an attacker, deception can cause confusion, uncertainty, misallocation of resources, and disclosure of attack methods (Climek et al. 2016). Cyberdeception alters an adversary's perception of reality and influences their decision-making to help the defender (Hancock, n.d.). Many deceptions from conventional warfare translate to cyberspace. Defenders can camouflage or conceal their networks to avoid drawing attention, plant false information for an adversary to find, or deploy decoys on which an adversary can waste their resources (Hancock, n.d.). Also, by monitoring an attacker, a defender can identify the tactics that attackers use, which can help preparedness (Hancock, n.d.).

Cyber defenders can use several methods and technologies to implement deception in their networks. Honeytokens in the form of fabricated data (such as fake email addresses, fake database records, and executable files) can reveal goals of an attacker and their methods (Fortinet, 2022). Honeypots also enable defenders to learn more about attack methods (Rapid7, n.d.-c). Commercial products from cybersecurity companies, such as Deception Technology from Rapid7 (Rapid7, n.d.-b), ThreatDefend from Attivo Networks (Attivo Networks, 2022), and Shadowplex from Alcalvio (Alcalvio Technologies, n.d.) simplify the task of implementing deceptions for defenders by automating the deployment of decoys and the collection of attack data.

C. HONEYPOTS

Honeypots should have no real users other than administrators and except for scanning traffic, very little of the remaining traffic to and from them should be legitimate (Rowe, 2007). This means that traffic to a honeypot is rich in clues to attack methods (Rowe, 2007). Low-interaction honeypots emulate services with limited user interaction such as SSH, mail services, and file transfer services, and do not offer access to a real operating system (Franco et al., 2021). Low-interaction honeypots are easy to set up, and inexpensive to purchase and maintain, but their limited capabilities enable them to be more easily detected by attackers as honeypots, and the data they collect is limited (Franco et al., 2021). High-interaction honeypots use real or virtualized hardware to offer more services, and usually allow access to an operating system (Franco et al., 2021). High-interaction honeypots can collect more complete data on an attacker’s methods, but are harder to set up and maintain, while possibly aiding attacks against other machines (Franco et al., 2021).

Deception can be more elaborate when a user can interact with a high-interaction honeypot. For example, a honeypot with a graphical user interface can be configured to look like it is actively used by personalizing the desktop background, showing additional downloaded applications, or posting files with fabricated contents that look to be significant to the user or organization. Multilayered deception with fake user accounts, files, and activity may keep an attacker engaged longer, which could encourage their interaction and our analysis of attack methods (Wang et al., 2013).

D. INTRUSION-DETECTION SYSTEMS

Intrusion-detection systems (IDSs) help defend computer networks. An IDS monitors network traffic and notifies administrators of suspicious events and possible exploitation attempts (Palo Alto Networks, 2022). There are several types of IDSs. Network-based IDSs analyze network and application protocol activity and are often deployed near firewalls or routers in a network; wireless IDSs monitor the protocols used for wireless networking and are deployed in proximity to wireless networks; network-behavior analysis IDSs monitor for unusual traffic flows and are usually deployed between an organization's network and partner networks; and host-based IDSs are deployed on single hosts in a network to monitor system logs, processes, and application activity (Scarfone & Mell, 2007). We used a network intrusion-detection system in this research to detect signatures of attack on the Remote Desktop Protocol.

E. INDUSTRIAL CONTROL SYSTEMS

Industrial control systems integrate information-technology capabilities with physical systems to automate industrial processes (Stouffer et al., 2015). The architectures of ICSs vary; They can be supervisory control and data acquisition (SCADA) systems, distributed control systems (DCS), or programmable-logic controllers (PLC) (Stouffer et al., 2015). Because of their use in industries for electricity, gas, water, food production, manufacturing, transportation, and more, operational ICSs are essential to the critical infrastructure of the United States and other nations (Stouffer et al., 2015).

While distributed control systems and programmable-logic controllers control a local area of an industrial plant or a specific industrial process, their information is often collected and sent to a SCADA system to allow more centralized monitoring and control of subsystems (Stouffer et al., 2015). SCADA systems are typically used for the centralized management and distribution of resources, including water, electricity, gas, and public transportation (Stouffer et al., 2015). Human-machine interfaces on SCADA systems display collected information about industrial processes; operators can then view this aggregated data and ensure the stability of the system and the proper distribution of resources (Stouffer et al., 2015). In this thesis, SCADA software which emulates control over an electrical grid was used as part of a high-interaction ICS honeypot deployment which attackers can access using RDP.

Maintaining the security of ICSs is a challenge. Inherent insecurities with ICSs include a lack of system processing power to simultaneously run security software and properly execute industrial processes (typically with older ICSs), and the insecurity of communications between controllers (Mathezer, 2021). Also, new vectors of attack have been created when ICS functions are made accessible from the cloud (Mathezer, 2021).

ICSs are high-value targets for advanced persistent threats and nation-state adversaries (Mathezer, 2021). Successful attacks on ICSs can have severe effects. For example, the 2021 ransomware attack on Colonial Pipeline caused a five-day outage of fuel transportation across the east of the United States, with secondary effects of higher gas prices, gas shortages, and the declaration of a state of emergency by the United States government (OCESER, n.d.). Also, during the current invasion of Ukraine by Russia, Ukrainian ICSs have been targeted by Russian cyberattacks. The Industroyer malware, used in an attack on the Ukrainian power grid in 2016, was recently updated and redeployed against Ukraine to cause power disruptions (ESET Research, 2022). Though this attack is reported to have been thwarted by Ukrainian cyber defenders, it could have further supported the Russian invasion and likely cost lives.

F. DIGITALOCEAN CLOUD ENVIRONMENT

DigitalOcean is a cloud-service provider that offers infrastructure-as-a-service through virtualized computing resources (DigitalOcean, 2022). DigitalOcean calls their Linux-based virtual machine that runs on top of virtualized hardware a “droplet,” which we call a *DigitalOcean machine* (*machine* for short). Resources in processors, memory, secondary storage, and geographical region of the server can be specified when creating a machine. DigitalOcean machines in the same region are on the same Virtual Private Network, which lets them communicate using their private IP addresses. A Web interface administers the machines by which resource use can be monitored, firewall rules can be managed and changed, snapshots of the virtual machine can be created, and storage devices can be managed.

G. RELATED WORK

A 2019 thesis from the Naval Postgraduate School analyzed the threat to the energy-grid through the deployment of a GridPot honeypot and the monitoring of protocol-specific network traffic (Kendrick & Rucker, 2019). Over a nineteen-day period, this honeypot received network traffic from 67 different countries with 9,641 HTTP requests, 621 MODBUS connections, 606 MODBUS traffic instances, and 102 S7Comm connections (Kendrick & Rucker, 2019). Though this honeypot was fingerprinted as a honeypot by the Shodan network-monitoring tool, attackers were not deterred from interacting with it. Future work recommendations from this thesis suggested improvements to the interaction of the honeypot.

Using Kendrick and Rucker's honeypot as a start, a 2020 thesis from the Naval Postgraduate School sought to evade honeypot detection mechanisms through improved interaction, proposing and implementing two new designs for the honeypot (Dougherty, 2020). The first design, which let a remote user interact with GridPot through IEC 60870-5-104 ("IEC 104" for short), a protocol for supervision and control of electrical grids, received a score of 0.0 by Shodan's Honeyscore evaluation, indicating that Shodan did not consider the scanned host to be a honeypot. The second design used a graphical user interface that was accessible by remote desktop software, though the results for the experiment using the second design were incomplete because the honeypot was disabled twice by attackers (Dougherty, 2020).

Following Dougherty's honeypot research, another 2020 thesis performed a comparative analysis between physical and cloud deployments of honeypots (Bieker & Pilkington, 2020). Using DigitalOcean as a cloud platform for hosting honeypot deployments, three experiments were devised to explore the effects of a general cloud deployment of GridPot, a deployment of GridPot with improved deception, and a deployment of GridPot in Asia (Bieker & Pilkington, 2020). Measurements of similarity across the three experiments and Dougherty's phase 1 experiment revealed that deployments of GridPot locally (on premise) and on the cloud were not significantly different with respect to the total traffic and total IEC 104 traffic received, showing that cloud deployments of honeypots could be effective for studying attacker behavior.

Research on cloud-based GridPot continued with a 2021 thesis that analyzed containerized honeypots using T-Pot, a honeypot management platform (Washofsky, 2021). A containerized version of GridPot was created and integrated with T-Pot running on the cloud. Results from Washofsky's experiments were compared to Bieker and Pilkington's results, showing only slight increases in IEC 104 traffic received using GridPot with T-Pot.

III. METHODS

Of remote-access methods such as Secure Shell, Telnet, Virtual Network Computing, and remote-desktop applications like TeamViewer or RemotePC, Microsoft Remote Desktop Services (Microsoft, 2020a) and the Remote Desktop Protocol (RDP) were chosen for this study. RDP was also used for the Windows user interface of (Dougherty, 2020), which provided a start for our research into remote-desktop attack methods. This chapter discusses RDP and identifies issues in characterizing signatures of remote-desktop attacks. We also discuss our approach to tool selection, honeypot deployment, and analysis of collected data.

A. HONEYPOT DESIGN

This research builds upon a previous honeypot implementation done at our school (Meier, 2022). Meier's implementation of the GridPot honeypot in (Dougherty, 2020) hardened the honeypot's user interface and improved its logging mechanisms. Meier's honeypot used three DigitalOcean machines in the same geographic region for ICS simulation, user interaction with the ICS, and secure logging of data and used the design shown in Figure 1.

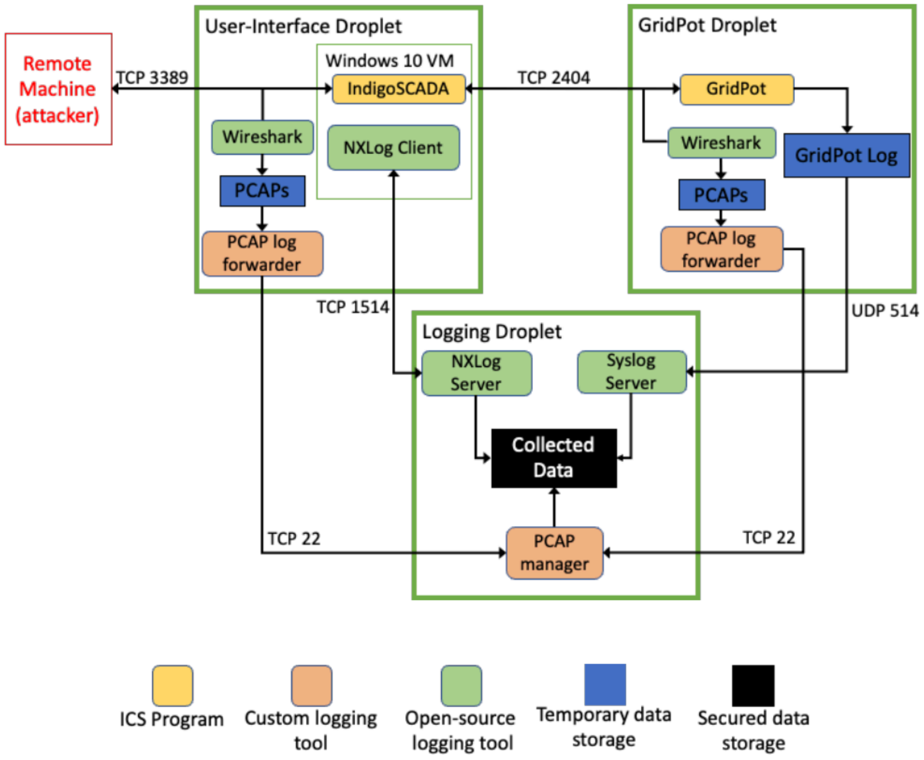


Figure 1. Meier GridPot Honeypot Architecture. Source: Meier (2022).

The user-interface machine runs a Windows 10 virtual machine that is remotely accessible by the Remote Desktop Protocol. Attackers can use the Windows virtual machine to interact with the power-grid simulation running on the GridPot machine. The GridPot machine provides a power-grid simulation and sends simulation data to the user-interface machine, where it appears on the Windows virtual machine's desktop. The display of the simulation data is managed by the IndigoSCADA application. The GridPot machine also runs a simple web server using the Hypertext Transfer Protocol (HTTP) to display limited simulation data and a message signaling the remote-desktop access method of the Windows virtual machine. The logging machine securely stores packet captures and event logs generated by the user-interface and GridPot machines using NXLog (NXLog, 2022), Syslog (Gerhards, 2009), and forwarding scripts. Secure Shell is enabled on each machine which allows administrative access to the honeypot.

Our honeypot used the same GridPot and logging machines that were used in (Meier, 2022), but we modified the user interface to use a different IP address and include additional software for capture and monitoring of remote-desktop network traffic.

Figure 2 shows the honeypot architecture used for Experiment 1 of our research. For this experiment, PyRDP was used with our honeypot to intercept and forward attacker connections to the Windows remote-desktop service.

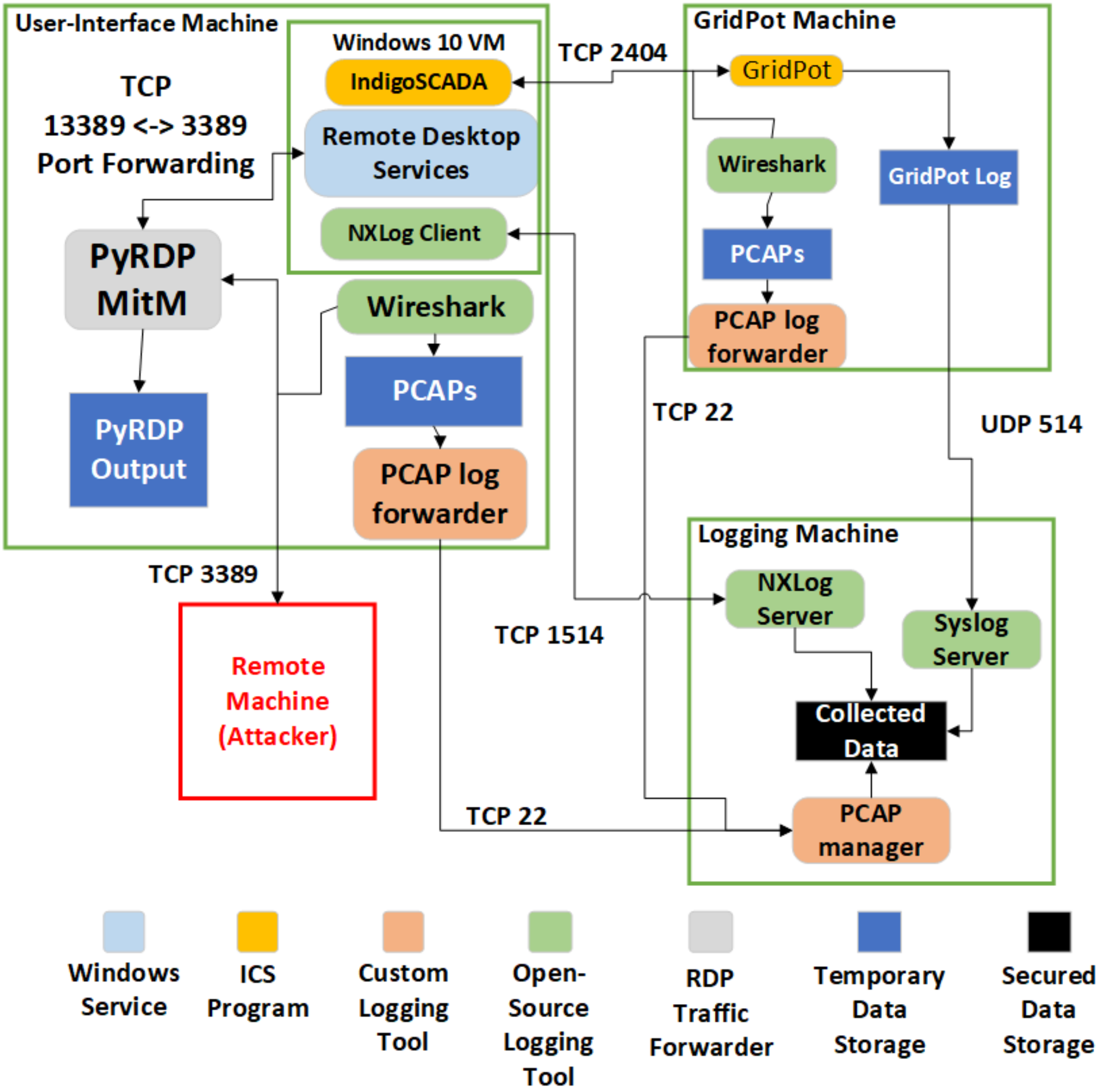


Figure 2. GridPot Honeypot Architecture of Experiment 1, which Used PyRDP. Adapted from Meier (2022).

Figure 3 shows the changes made for Experiments 2, 3, and 5 when PyRDP was removed and Snort was added to monitor traffic. The GridPot HTTP service was also enabled. The Kali Linux attack machine in Figure 3 architecture is shown in more detail in Figure 4.

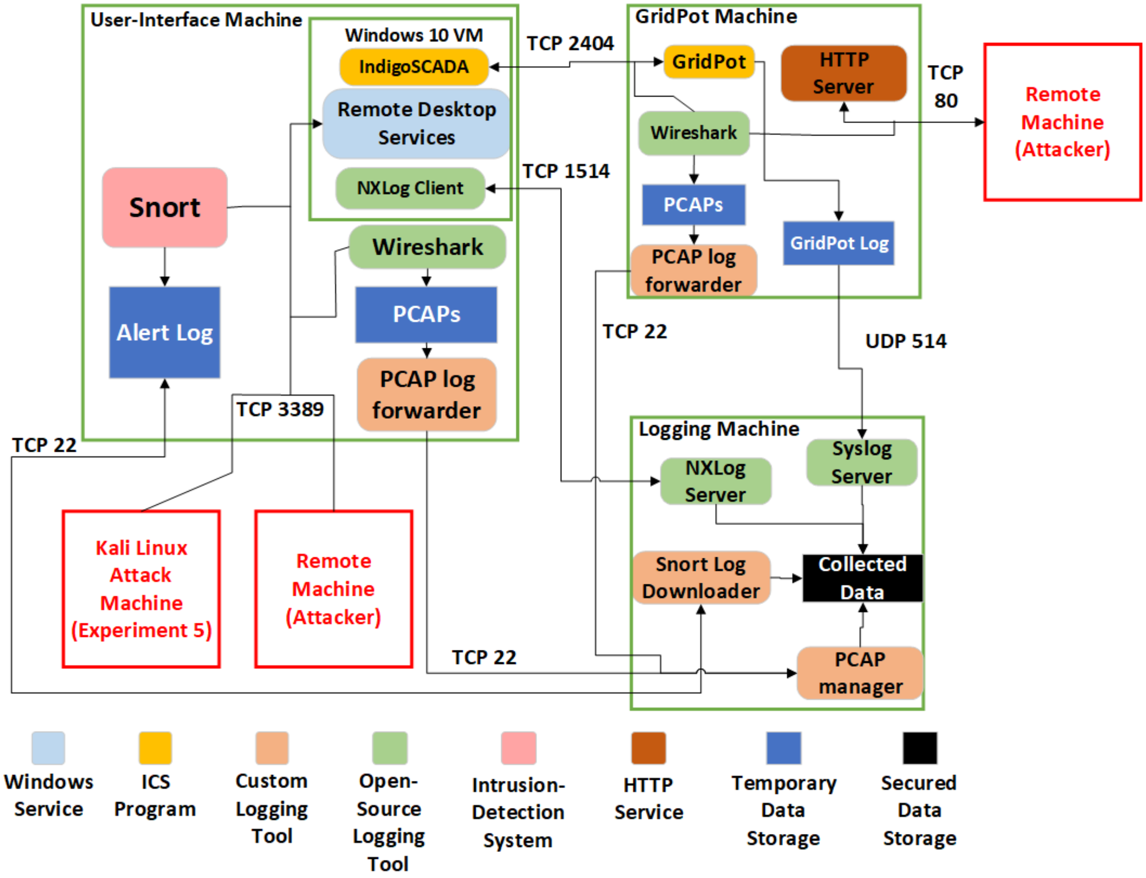


Figure 3. GridPot Honeypot Architecture of Experiments 2, 3, and 5 Using Snort and GridPot HTTP Server. Adapted from Meier (2022).

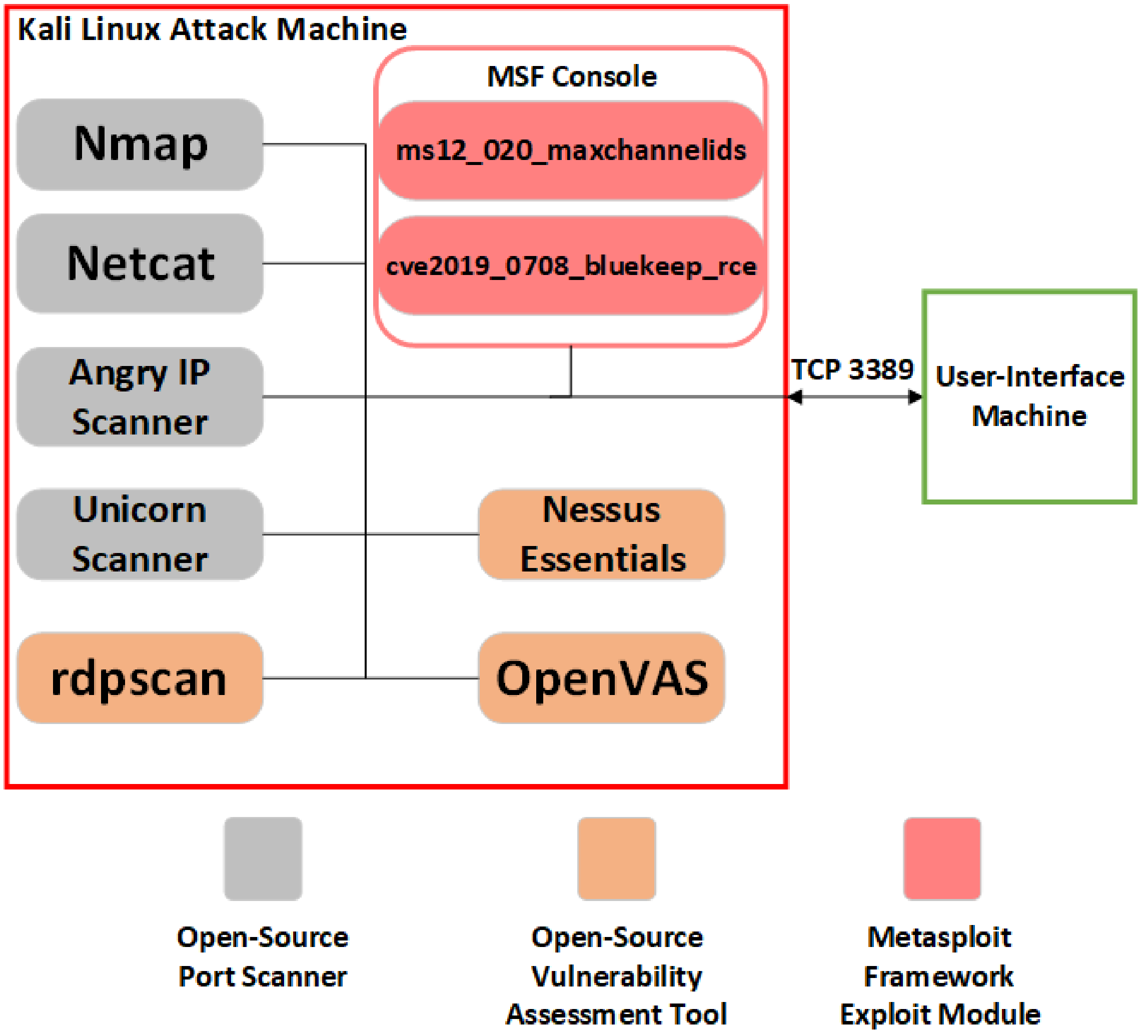


Figure 4. Kali Linux Attack Machine Used in Experiment 5

We created the Kali Linux attack machine by downloading additional attack tools to a standard Kali Linux virtual-machine appliance. The purpose of the Kali Linux attack machine was to send data to our honeypot for collection and analysis in Experiment 5.

Figure 5 shows the honeypot architecture of Experiment 4 where PyRDP and Snort intercepted and monitored traffic.

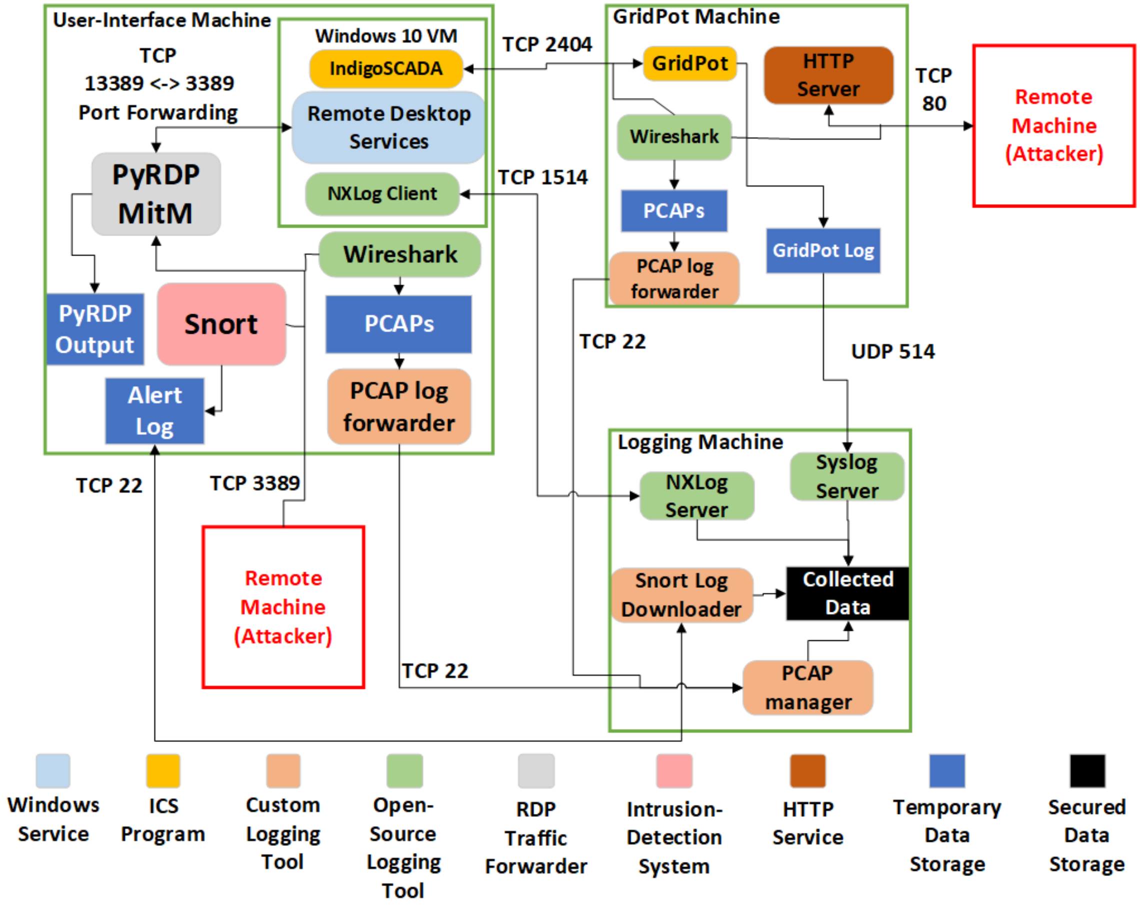


Figure 5. GridPot Honeypot Architecture of Experiment 4 Using PyRDP and Snort. Adapted from Meier (2022).

B. OVERVIEW OF THE REMOTE DESKTOP PROTOCOL (RDP)

Though useful for remote system administration, RDP gives attackers opportunities to disrupt the machines that use it. Attacks against a system can be executed over RDP, where the protocol provides legitimate access to a system before the attacker starts acting upon their goals. Also, flaws in the protocol itself can be exploited by an attacker for the unauthorized access of a system or denial of service.

1. Protocols and Standards

RDP follows a fixed connection sequence with several sub-protocols to control the flow of data between a remote desktop client and server while providing confidentiality and integrity to data in transit. The protocol stack is shown in Figure 6.

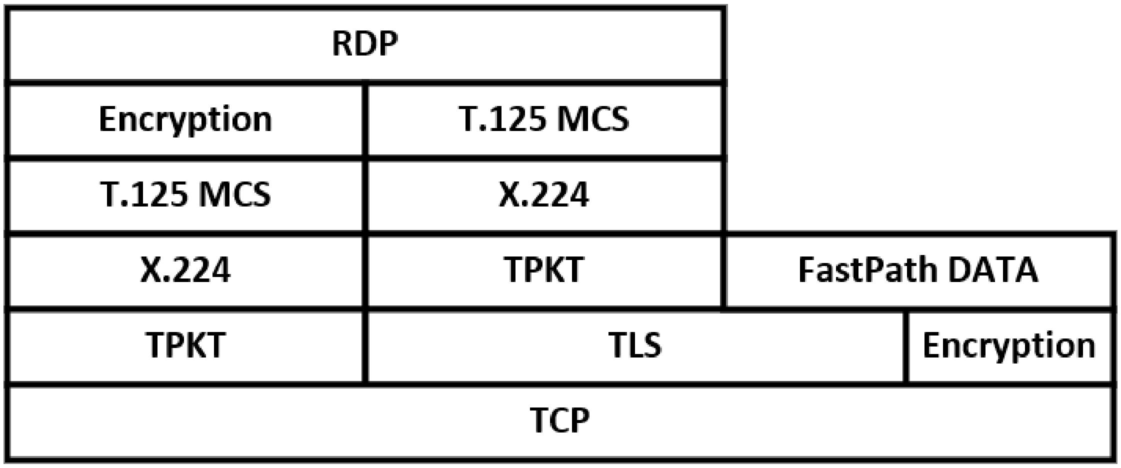


Figure 6. The RDP Protocol Stack. Adapted from Reiner (2020).

The sub-protocols that RDP uses are:

- TCP for basic networking (Microsoft, 2021d).
- The TPKT protocol to encapsulate networking data to be sent (Microsoft, 2021b). TPKT forms transport-protocol data units (TPDUs) containing networking data and embeds them as payloads inside TCP packets before sending them with TCP (Pouffary & Young, 1997).
- X.224, also called the Connection-Oriented Transport Protocol (COTP), for client connection requests (Microsoft, 2021b) and server responses.
- The Multipoint Communication Service (MCS) and Generic Conference Control (GCC) to create the channels for data such mouse movements, screen updates, and clipboards (Microsoft, 2021b).
- The RC4 stream cipher for secure communications. It can also use stronger security protocols like TLS, CredSSP, and RDSTLS for encryption, authentication, and integrity checks (Microsoft, 2021c).

2. Connection Sequence

Figure 7 shows the ten steps of the RDP connection sequence. Protocol Data Units (PDUs) are exchanged between the client and server in each step of the connection sequence; PDUs specify the settings used for processing data during the connection (Microsoft, 2022b).

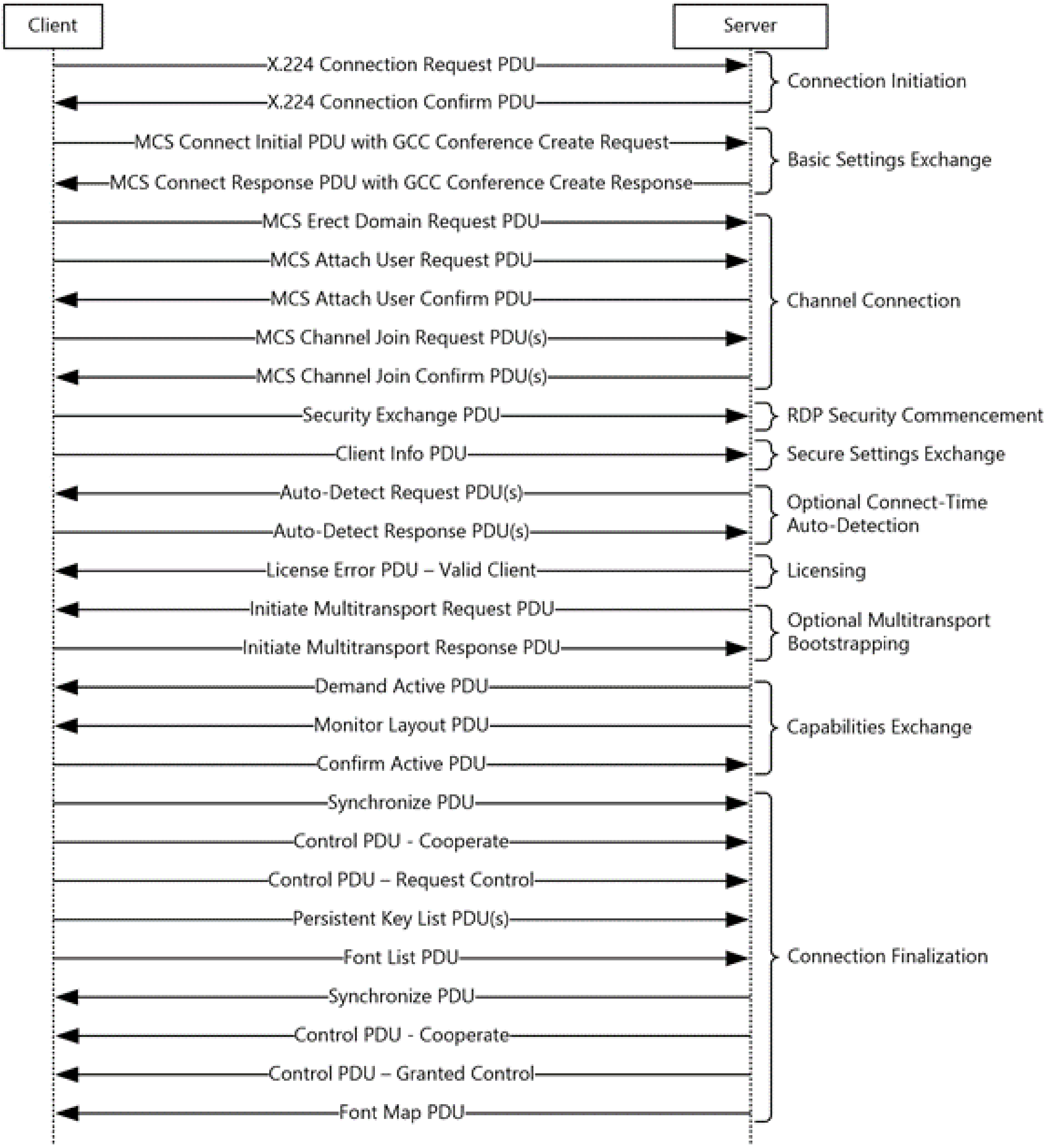

Client
Server
X.224 Connection Request PDU
X.224 Connection Confirm PDU
Connection Initiation
MCS Connect Initial PDU with GCC Conference Create Request
MCS Connect Response PDU with GCC Conference Create Response
Basic Settings Exchange
MCS Erect Domain Request PDU
MCS Attach User Request PDU
MCS Attach User Confirm PDU
MCS Channel Join Request PDU(s)
MCS Channel Join Confirm PDU(s)
Channel Connection
Security Exchange PDU
RDP Security Commencement
Client Info PDU
Secure Settings Exchange
Auto-Detect Request PDU(s)
Auto-Detect Response PDU(s)
Optional Connect-Time Auto-Detection
License Error PDU – Valid Client
Licensing
Initiate Multitransport Request PDU
Initiate Multitransport Response PDU
Optional Multitransport Bootstrapping
Demand Active PDU
Monitor Layout PDU
Confirm Active PDU
Capabilities Exchange
Synchronize PDU
Control PDU - Cooperate
Control PDU – Request Control
Persistent Key List PDU(s)
Font List PDU
Synchronize PDU
Control PDU - Cooperate
Control PDU – Granted Control
Font Map PDU
Connection Finalization

Figure 7. RDP Connection Sequence between a Client and Server. Source: Microsoft (2022b).

For detection of attacks on RDP itself, the most significant phases of the connection sequence are connection start, basic settings exchange, and security commencement. During the start phase, RDP client connection requests and server responses are transmitted in cleartext, so an attack on the protocol in this phase can be detected automatically by an intrusion-detection system if its signature is known. The basic settings exchange of the connection sequence is also important since it is where past exploits have taken advantage of RDP; signatures for known attacks, such as the BlueKeep exploit mentioned in Chapter II, may be found in a client's message to a server in this phase. Once RDP security commencement happens, all subsequent RDP network traffic will be encrypted (Microsoft, 2022b), so an intrusion-detection system cannot detect attacks on the protocol then.

C. METHODOLOGY FOR RDP DATA ANALYSIS

Network traffic of RDP can be decrypted using the server's private key (Duncan & Prakash, 2021), which permits analysis of data sent throughout its connection sequence. Decrypted traffic also permits reconstruction and replay of a remote-desktop session. Analysis of metadata from remote-desktop sessions can also reveal information about attacks on RDP and remote hosts.

Our goal was to develop methods to describe the features of attacks on RDP and the Windows system that used it. These features included recognizable attack patterns from the decrypted packet contents sent during the RDP connection sequence, characteristics of the remote-desktop network traffic, and the behavior of the attacks observable through replays of remote-desktop sessions or assembled from event logs. Our methods included ways to collect attack data, compare attack data to baseline data generated in a controlled environment, and analyze the attacks using RDP-analysis tools.

Our approach used two sets of data. One set contained RDP traffic collected by an Internet-facing ICS honeypot. The other contained benign and malicious RDP traffic created by us using port scanners, vulnerability-assessment tools, and the Metasploit Framework. By understanding the distinctive patterns of malicious and benign RDP network traffic from instrumented data, we could classify the actual RDP attacks received by a honeypot as malicious or benign, then determine the severity of the attacks on the honeypot. Table 1 shows the levels of severity that we used for characterizing RDP attacks from RDP connection data and the remote-desktop sessions that were established after the connection sequence of an RDP connection completed. For this work, severity is defined by the level of harm incurred by our Windows machine.

Table 1. Severity Levels for Attack Characterization

Severity Level	Description
Very High	An attacker has completed the RDP connection sequence and has spent more than five minutes connected to the machine. Windows event-log entries were associated with malicious actions.
High	An attacker has completed the RDP connection sequence and has spent more than five minutes connected to the machine but no Windows event-log entry was associated with malicious actions, or the attacker has spent between one and five minutes connected to the target machine and there were Windows event-log entries associated with malicious actions.
Moderate	An attacker has completed the RDP connection sequence and has spent between one and five minutes connected to the machine, and no Windows event-log entry was associated with malicious actions, or the attacker has spent less than a minute connected to the target machine with Windows event-log entries associated with malicious actions.
Low	An attacker has completed the RDP connection sequence, spent less than a minute connected to the target machine, and there was no Windows event-log entry associated with malicious actions, or the attacker has performed vulnerability scanning on the machine or tried to attack the machine with an ineffective exploit.
Very Low	The attacker has at most only partially completed the connection sequence and signatures of malicious activity have not been identified in or during an attacker's connection to the machine.

We used tools designed for RDP analysis to determine signatures of attack, where they occur in the RDP connection sequence, and the severity of the attacks. Methods for attack characterization compared actual data to instrumented data, correlating RDP server event logs to captured data, and detailed analysis of RDP session metadata.

D. DATA GENERATION TOOLS

For generation of benign-traffic data, we needed network traffic that is generally investigative without searching for vulnerabilities in the host machine's running services. Benign RDP network traffic does not harm the host by stealing information, changing host machine configurations, or denying service to or from the host. To create our instrumented datasets, we used open-source and free tools: port scanners, vulnerability scanners, and Metasploit Framework exploit modules. We used port-scanning tools to generate benign data that looked like routine scanning.

Vulnerability-assessment tools, which analyze hosts and their running services, allow attackers to determine the best ways to attack and infiltrate networks. We used these tools to generate malicious data, though they could also be used for legitimate security monitoring. The Metasploit Framework is a good source of exploits for common network protocols. We used modules from Metasploit to generate malicious data since attackers can use them to exploit systems.

1. Port-Scanning Tools Used

- Nmap is a popular open-source tool to discover hosts and services on a network (Nmap, n.d.). Beyond determining if a port on a host is open, Nmap can determine the version of the service running on that port and make guesses to the operating system that a host is running. Nmap can be used on the command line or through Zenmap, a graphical user interface. We used Nmap for its ease of use, its capability to generate data with varying types of scans, and its popularity as a networking tool.

- Netcat is a popular networking tool that can transmit and receive data over network connections, but it also supports port scanning (Giacobbi, 2006). We used Netcat for benign traffic generation due to its widespread use.

- Angry IP Scanner is an open-source tool using a multithreaded approach to port scanning (AngryIP, 2022). This tool first queries a host to see if it is up before searching for open ports. We used it due to its general port scanning capabilities. With over 29 million downloads, Angry IP Scanner is likely used by attackers for reconnaissance of networks.

- Unicornscan is a port-scanning tool that is part of Kali Linux (OffSec Services, 2022). This scanner provides similar capabilities to other port scanners, including OS and service version detection. Unicornscan can perform asynchronous and stateless scanning and has its own TCP/IP stack for improved speed. We used Unicornscan for its unique capabilities. Since Unicornscan also comes preinstalled on the popular Kali Linux, some attackers may favor it over other port-scanning tools.

2. Vulnerability-Assessment Tools Used

- Nessus is a commercial vulnerability-assessment product (Tenable, 2022b). A free version called Nessus Essentials has fewer features. Nessus uses plugins for detecting known vulnerabilities on hosts to generate vulnerability-assessment reports.

- Like Nessus, OpenVAS is a vulnerability-assessment tool that reports the vulnerabilities of hosts that it analyzes (Greenbone Networks, 2022).

- Rdpscan is a vulnerability-assessment tool dedicated to finding hosts vulnerable to BlueKeep exploitation (RobertDavidGraham, 2019). Hosts that are scanned by this tool are classified as vulnerable or safe against BlueKeep, though it is possible for some hosts to be classified as neither if they reply to the scan with unexpected behavior.

3. Metasploit Framework Modules Used

- The cve_2019_0708_bluekeep_rce Metasploit exploit module targets Windows 7 and Windows 2008 systems that are vulnerable to BlueKeep (Cook, 2019). It exploits a flaw in the basic settings-exchange phase of the RDP connection sequence for remote code execution on the target. Although our honeypot uses Windows 10 and is safe against BlueKeep exploitation, we used this module to collect data about other malicious RDP network activities.

- The ms12_020_maxchannelids exploit module in the Metasploit Framework exploits a flaw in the basic settings-exchange phase of the RDP connection sequence (Metasploit, 2022). This module uses a specially crafted packet to cause an invalid pointer to be used, resulting in a denial of service condition for the RDP service. This vulnerability was revealed in 2013 and our honeypot is safe against this exploit, but data about related malicious RDP network activity collected from its use can still be useful.

E. DATA ANALYSIS TOOLS

To analyze attacks on RDP and attacks using RDP against the Windows host of our honeypot, we used several tools to capture and replay RDP sessions. Some tools can also analyze activity logs generated by intrusion-detection software. Our selection criteria considered provenance (reputable origin), cost (open-source or free to use), maintainability (actively maintained or recently released), and whether the tool could process packets.

1. Wireshark

Wireshark is a widely known tool for capture and analysis of network traffic (Wireshark, n.d.). It is free to use and is actively maintained by developers worldwide. Wireshark can capture and save network traffic as packet captures (PCAPs). With the private key of an RDP server, Wireshark can decrypt RDP communication between a client and server and inspect for malicious content. This tool captured, decrypted, and analyzed RDP network traffic to our honeypot.

2. RDP Replay

RDP Replay was developed for analysis of RDP network traffic. Using it, RDP sessions can be replayed as they would be seen by an attacker or RDP client, showing screen updates, mouse movements, and key presses (CTXIS, 2016). It replays an RDP session from a packet capture file and a file containing the RDP server's local private SSL key for decryption of RDP network traffic.

Given this tool's ability to show RDP sessions from the point of view of an attacker, its ease of setup, and its capability for offline analysis, we considered using this tool with our honeypot. However, we discovered that it often crashed, stopped, or failed to generate replay files. Developer responses to community feedback on the RDP Replay GitHub page indicated that these issues were likely caused by the application's limited support for newer TLS security options, and the tool's last update was in June 2016. For these reasons, we stopped using RDP Replay.

3. PyRDP

PyRDP is a library of tools developed in 2018 that is actively maintained by GoSecure for capture and analysis of RDP network traffic (Beaulieu, 2020). PyRDP contains four applications. Its primary tool is the PyRDP "Monster-in-the-middle" (MitM) application which intercepts traffic between RDP clients and servers. If an RDP client connects to a PyRDP server instead of a target RDP server, PyRDP will respond to the client's connection request. Routing traffic through the PyRDP MitM application, PyRDP can log the credentials used to connect to the target server, the contents of the client's clipboard, transferred files and their contents, and shared drives and their contents that are mapped by the client for the RDP session. Captured information from each RDP connection is saved and can be replayed, complete with screen updates, mouse positions, and key presses. It can also run Windows PowerShell commands for each new connection.

Other tools in the PyRDP library are the PyRDP Player, the PyRDP Converter, and the PyRDP Certificate Cloner (Beaulieu, 2020). The PyRDP Player has several uses, including replaying RDP sessions saved by the PyRDP MitM application, viewing live RDP sessions that the PyRDP MitM application is currently capturing, hijacking RDP sessions that the PyRDP MitM is currently capturing, and interacting with client-mapped drives for active RDP connections. The PyRDP Converter can convert replays or packet captures to video or JSON files. The PyRDP Certificate Cloner is an auxiliary tool used automatically by the PyRDP MitM application to clone X.509 certificates of legitimate RDP servers to show and deceive newly connecting RDP clients.

4. Malcolm

Malcolm is a network-traffic analysis tool developed by Idaho National Laboratory (Idaho National Laboratory, n.d.). Malcolm includes third-party tools for intrusion detection, malware analysis, and data presentation to process packet captures and display metadata about them for analysis. Malcolm can analyze 47 protocols from packet captures and can create intrusion-detection logs through the network monitoring application Zeek (The Zeek Project, 2020).

Malcolm shows details through the Arkime (Arkime, n.d.) and OpenSearch Dashboards (OpenSearch, 2022). Arkime shows flow data about connections in the capture file including connection start and stop times, source and destination IP addresses and ports, the number of packets exchanged in a connection, and the number bytes exchanged in a connection. OpenSearch Dashboards shows graphs and plots of information such as the number of connections, percentage of network traffic by protocol, and times of high network activity.

We used Malcolm because it is actively maintained and can analyze metadata of RDP network traffic. It can also automatically generate Zeek intrusion-detection logs from RDP network traffic, though these logs only contain information from the data that is transmitted in cleartext in the first phase of the RDP connection sequence. Also, its graphs and plots provide capabilities beyond those in Wireshark.

5. Snort

Snort is an intrusion-detection and intrusion-prevention system that is actively maintained by Cisco (Cisco, 2022). Snort uses signature-based rules to detect malicious network traffic and create alerts which are stored in a log file. Different rulesets are available for community users, registered users, and subscribers. Snort is free to download, and users can register for free to access the registered-user ruleset of attack signatures.

We used Snort as an intrusion-detection system. The Snort ruleset that we used contained signatures of RDP attacks, but Snort cannot decrypt. Even though most RDP network traffic is encrypted after the first phase of the RDP connection sequence, we wanted to see if Snort could generate alerts for the traffic it processed.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. EXPERIMENT DESIGN AND IMPLEMENTATION

This chapter describes our honeypot experiments, our methods for generating benign and malicious RDP attack datasets, our analysis of generated attack data, and our RDP attack-characterization methodology. We also discuss methods for evaluating characterizations of actual attacks that occurred during our experiments.

A. DESIGN OF EXPERIMENTS

Five experiments were performed for data collection and analysis of RDP attacks on the GridPot honeypot. Experiment 1 used PyRDP MitM to generate video replays of attacks and characterized attacks using these replays with captured network traffic and event logs. Experiment 2 let attackers interact with the remote-desktop service of the honeypot directly while using Snort, network traffic analysis tools, and event logs to detect attempts to exploit our Windows machine. Experiment 3 was the same as Experiment 2 except that the public IP address of the honeypot was changed to renew attackers' interest in our honeypot. Experiment 4 was a second attempt at using PyRDP MitM to characterize remote-desktop attacks through video replays, event logs, and network traffic analysis. Experiment 5 established baseline datasets for benign and malicious RDP attacks.

During Experiments 1–4, our honeypot was publicly accessible to capture attacks from all sources. During Experiment 5, our honeypot was accessible from a private network interface to generate benign and malicious RDP attack data.

1. Experiment 1

This experiment ran from April 15th to May 8th. It used PyRDP MitM to intercept traffic between a remote-desktop client and the RDP server on our honeypot (Figure 2 in Section III.A). During this experiment, we successfully captured RDP network traffic, PyRDP event logs, and session replays.

On the user-interface machine of our honeypot, we created a Python virtual environment to install the required software packages for PyRDP to run. To avoid conflicts with two services binding to the same port to listen for connections, VirtualBox networking settings were adjusted to forward network traffic received on port 13389 (picked solely for VirtualBox port forwarding) of the Linux host to port 3389 (RDP) of the Windows guest virtual machine, which let both the PyRDP MitM application and Windows Remote Desktop Services use port 3389 simultaneously without conflicts. The PyRDP MitM application was then run on the Linux host, listening for an RDP server on port 13389 of the local machine. Figure 8 shows the port forwarding that occurs in this setup.

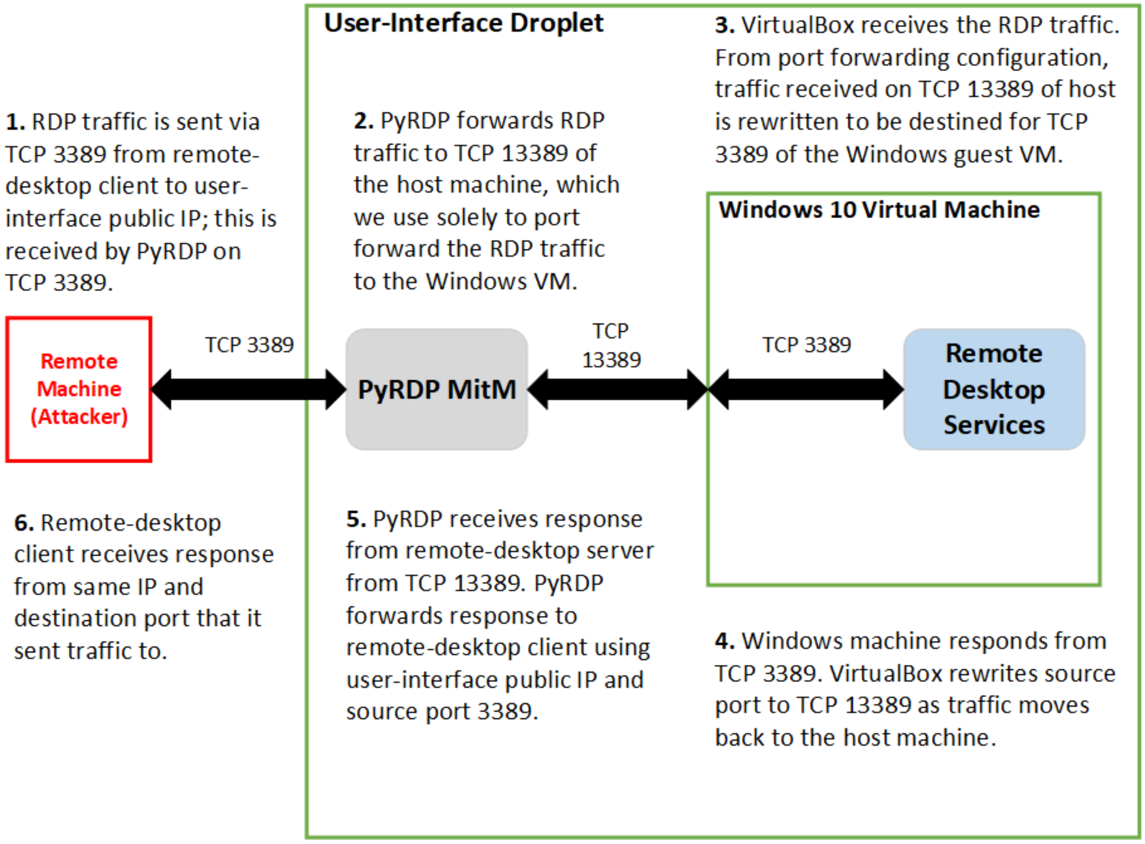

User-Interface Droplet
1. RDP traffic is sent via TCP 3389 from remote-desktop client to user-interface public IP; this is received by PyRDP on TCP 3389.
2. PyRDP forwards RDP traffic to TCP 13389 of the host machine, which we use solely to port forward the RDP traffic to the Windows VM.
3. VirtualBox receives the RDP traffic. From port forwarding configuration, traffic received on TCP 13389 of host is rewritten to be destined for TCP 3389 of the Windows guest VM.
Windows 10 Virtual Machine
Remote Machine (Attacker)
TCP 3389
PyRDP MitM
TCP 13389
TCP 3389
Remote Desktop Services
6. Remote-desktop client receives response from same IP and destination port that it sent traffic to.
5. PyRDP receives response from remote-desktop server from TCP 13389. PyRDP forwards response to remote-desktop client using user-interface public IP and source port 3389.
4. Windows machine responds from TCP 3389. VirtualBox rewrites source port to TCP 13389 as traffic moves back to the host machine.

Figure 8. Port Forwarding for the Honeypot Architecture of Experiment 1 and Experiment 4

The Windows virtual machine in this experiment followed the design of Meier's Experiment 3 (Meier, 2022). The Windows machine had two accounts: "Remote Admin" that was accessible only by honeypot administrators, and "HMI-Operator" which had no password and was accessible remotely by anyone. To indicate to attackers that the Windows machine connected to an ICS system, the IndigoSCADA application was set to launch whenever a user logged in, and the background of the desktop was changed to look like the machine belonged to an electrical company.

Using DigitalOcean's administrative interface, we created firewall rules to allow the user-interface machine to receive Internet Control Message Protocol (ICMP) network traffic from all sources, RDP network traffic on port 3389 from all sources, and SSH traffic on port 22 from administrative sources like the logging machine and the private network of the honeypot administrator. To capture network traffic to the user-interface machine of honeypot, we used Wireshark's command-line tool T-Shark on the Linux host of the user-interface machine. T-Shark was set to filter out administrative SSH network traffic and to limit PCAP file sizes to twenty-thousand bytes.

With the NXLog service running on the logging machine of our honeypot, Windows event logs from the Windows virtual machine were saved on the logging machine. The Syslog service on the logging machine logged events of the GridPot machine. Also, a custom script running on the user interface and GridPot machines told the logging machine of new PCAPs to be downloaded for safe storage. The HTTP server of the GridPot machine was not active during this experiment. Using the secure-copy command-line utility, output from the PyRDP MitM application was manually downloaded to the logging machine for storage on the logging server throughout the experiment, then once more at the experiment's end on May 8th.

2. Experiment 2

Our second experiment used standard Windows RDP without using PyRDP to intercept and forward traffic. This experiment used the same public IP address from Experiment 1 and ran from July 8th to July 15th. The architecture of the honeypot in this experiment is shown in Figure 3 in Section III.A.

The user-interface machine was modified to use a different Windows virtual machine that was created by Meier for his Experiment 4 (Meier, 2022). The new Windows machine had the name of a fake company, an SSL certificate with the name of the fake company, and new usernames for the administrative account and the publicly accessible account. Also, policies on the Windows guest machine for Sysmon and PowerShell were changed to show attacker interactions more clearly; this included filtering excess logging events and the logging of PowerShell commands. To allow RDP network traffic to reach the Windows machine, VirtualBox port forwarding on the user-interface machine was configured to forward traffic from port 3389 on the Linux host to port 3389 of the Windows guest.

The Snort intrusion-detection system was used in this experiment to test its ability to recognize RDP attacks. Snort 2.9 ran on the Linux host of the user-interface machine, and a free account was created on the Snort website to download the registered-user ruleset which contained attack signatures for the Remote Desktop Protocol. Though we were primarily interested in Snort's ability to detect attacks on RDP, we used the entire registered-user ruleset to detect signatures of attacks on other services if unauthorized traffic could bypass the DigitalOcean firewall to the user-interface machine. A script was run on the logging machine to retrieve the Snort alert log every two hours by an SSH secure copy.

The firewall for the user interface was configured the same as Experiment 1, allowing inbound traffic only for ICMP queries from all sources, RDP on port 3389 from all sources, and SSH from administrative sources. Capture of network traffic to the user-interface machine was done mostly the same as Experiment 1 except that they were saved in the PCAP format since Malcolm requires it.

The HTTP server of the GridPot machine was also enabled for this experiment. This server displayed a single webpage containing brief information about the power-grid simulation running on the GridPot machine and a message indicating the remote-desktop access method of the user-interface machine with its public IP address. The logging machine, besides logging Windows event logs and PCAPs from the user-interface and GridPot machines, also logged HTTP requests for this webpage.

3. Experiment 3

Our design for Experiment 3 was nearly identical to the design of Experiment 2 with the main difference being a new public IP address for our user-interface machine. Since the old public IP address had been publicly reachable by ICMP and RDP for 30 days between Experiment 1 and Experiment 2, attackers may have become familiar with it and uninterested in attacking it. Experiment 3 ran from July 15th to August 9th. During this experiment, our honeypot was scanned by Shodan twice and received over two-million RDP connections, which was about 15 times greater than the number received in Experiment 1, and over 160 times greater than what we observed in Experiment 2.

We modified the webpage for the GridPot machine's HTTP server to reflect the new public IP address of the user-interface machine. Logging for Windows event logs, PCAP forwarding, and Snort alerts were adjusted for the user-interface machine's new private IP address.

4. Experiment 4

With a new public IP address for the user-interface machine, we again wanted to see if we could use PyRDP to capture and replay attacker interactions with our honeypot. Experiment 4 used the same configurations as Experiment 3 for the user-interface firewall, capture of network traffic, and logging of events, PCAPs, and Snort alerts. The PyRDP MitM application was run with VirtualBox port forwarding configured as it was in Experiment 1. The diagram of the honeypot architecture of this experiment is shown in Figure 5 in Section III.A. Experiment 4 ran only for two days from August 9th to August 11th. We stopped it because the volume of traffic being received was much less than Experiment 3.

5. Experiment 5

Experiment 5 was run solely for collecting baseline Windows event-logs and instrumented data. This experiment used standard Windows RDP with the same configurations as Experiment 3, but DigitalOcean firewalls were modified to make the honeypot inaccessible from non-administrative sources. Using the port scanners, vulnerability assessment tools, and Metasploit modules mentioned in Chapter III, we attacked our honeypot to generate benign and malicious data and collect the Windows event logs from each attack. This experiment ran for less than a day and concluded when all tools discussed in Section III.D had been used to attack the honeypot.

B. RDP TRAFFIC CHARACTERIZATION

1. Scanner Testing

We used the port scanners, vulnerability-assessment tools, and Metasploit Framework (MSF) modules discussed Section III.D to attack our honeypot in Experiment 5. We also created our own malicious remote-desktop sessions to the honeypot to simulate successful logins by attackers. The attack data collected formed our instrumented datasets to develop methods for characterizing RDP attacks.

We used the Nmap command-line port scanner in two configurations to generate network traffic to our honeypot in Experiment 5. The first configuration used the -Pn flag to skip host discovery and the -p flag to scan only port 3389 of our honeypot's user interface. This scan took under seven seconds to complete and indicated that port 3389 was open. The second configuration of Nmap used the -sV flag to indicate a request for service versioning, the -O flag for OS detection, and the -p flag to scan only port 3389 of the user-interface machine of the honeypot. This scan took roughly twenty seconds and could identify the RDP software being used as Microsoft Terminal Services. Though the RDP service was running on the Windows virtual machine of the user interface, the scan's guess for the underlying operating system heavily favored Linux instead of Windows.

Netcat was used on the command-line in one configuration. Since Netcat is primarily for reading and writing data across connections, we used the -z flag to indicate that we wanted to use it for port scanning. The -v flag displayed output of the scan to the terminal, and the -n flag was used to skip Domain Name System (DNS) lookup. The public IP address of the user-interface machine used 3389 as the port to be scanned. This scan finished quickly and showed port 3389 was open.

The Angry IP Scanner uses a graphical interface for port scanning. To scan port 3389, we modified the preferences to scan "dead" hosts (i.e., hosts that do not reply to ICMP queries), increased the timeout for connecting to five seconds, and specified only port 3389 for the port range. This scan took about twenty-five seconds and indicated that port 3389 was open.

The Unicornscan port-scanning tool was used in two configurations. The first configuration used only the -v flag for verbose output to the terminal after completing the scan. The second configuration used the -v flag for verbose output and the -q flag with a value of 255 to perform the scan covertly. Both scans of the user-interface machine finished in about seven seconds and showed that port 3389 was open.

The Metasploit ms12_020_maxchannelids auxiliary exploit module was designed to cause a denial-of-service condition in the RDP service. This module ran on a Kali Linux "attack machine" with the RHOSTS option set to the public IP address of our user-interface machine. The RPORT option was configured for port 3389 by default. The remote-desktop service of the Windows machine was not vulnerable since it has been patched against the exploit.

To generate BlueKeep traffic, we used the Metasploit cve_2019_0708_bluekeep_rce exploit module in two configurations. This exploit module allows an attacker to customize client information that is reported to the RDP server upon connection, such as the RDP client's IP address, name, and domain, and the username that the client is trying to log in as. For the first configuration, default values of these options were used, i.e., the RDP client IP was 192.168.0.100, the client's name was "ethdev," there was no client domain name, and the username for connection was random. The RHOSTS option was also set to the public IP address of our user-interface machine. After it was started, the exploit was aborted since the vulnerability scan in it reported that the target was not vulnerable to the exploit. The second configuration of this exploit used the same settings, but with the ForceExploit flag set to proceed with the exploit despite a scanning report that states the target is not vulnerable. The exploit was aborted once again since our Windows machine was not vulnerable to BlueKeep exploitation.

Also related to BlueKeep, the Rdpscan tool generated RDP traffic associated with malicious scanning of the BlueKeep vulnerability. We ran it on the command line with our user interface's public IP address as its only argument. The report from this scan indicated that the user-interface machine was safe from BlueKeep attacks.

For malicious vulnerability scanning, we used two scanners, Greenbone OpenVAS and Nessus. In OpenVAS, we created a task to scan our honeypot's user interface on TCP port 3389. We used the "full and fast" scan configuration since it was the most comprehensive vulnerability scan with the Community Edition of OpenVAS. The value for minimum quality of detection was set to 75%. The vulnerability scan was then started, taking roughly seven minutes to finish, and reporting one medium-severity vulnerability related to the RDP server's support for older TLS versions.

We also used a free version of the Nessus scanner. We used the template for a basic network scan with the target set to the public IP address of the user-interface machine. We modified the template to a custom scan with the scan range set to port 3389 only. Nessus reported four medium-severity vulnerabilities from support of older versions of TLS, a self-signed SSL certificate being used, and one high-severity vulnerability related to the support of a weak SSL encryption method.

We established our own remote-desktop sessions to the honeypot to simulate attackers who successfully logged in to the Windows virtual machine. During these sessions, we investigated the file system, entered commands in PowerShell, accessed the web browser, and accessed the IndigoSCADA application. We collected the network traffic and event logs for these test sessions.

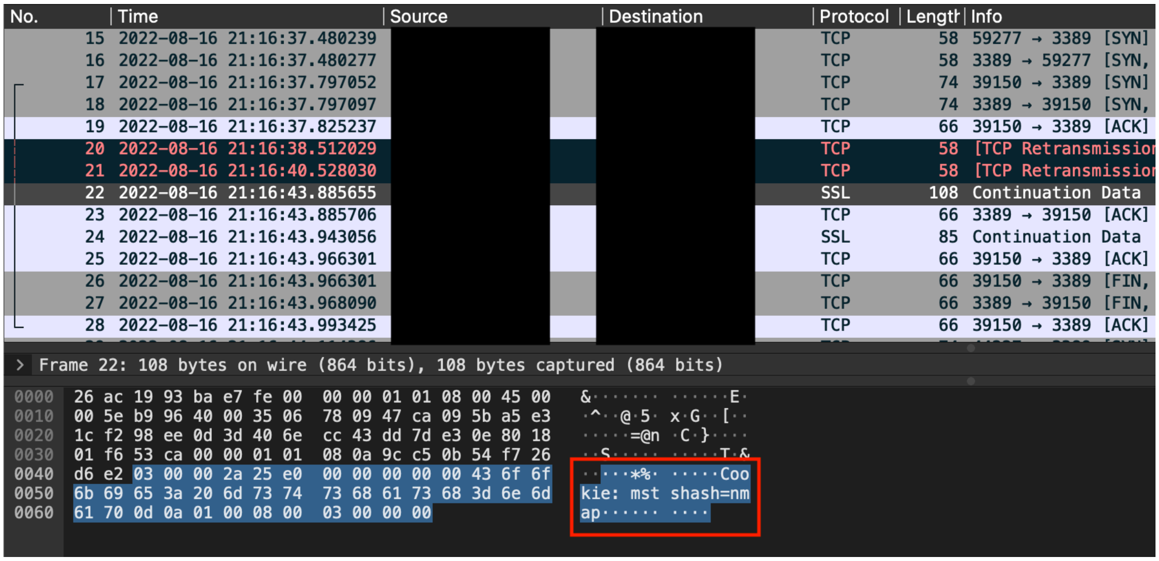
2. Data Analysis

To create methods for RDP traffic characterization, we analyzed the traffic of our generated datasets to determine the characteristics of benign and malicious RDP network traffic. Our analysis consisted of searching for signatures of port scanning, vulnerability scanning, exploitation in packets, and Windows event logs. We also investigated the network metadata for connections made to our remote-desktop server by each tool used for data generation.

a. Characteristics of Benign RDP Network Traffic

Since the tools we used for benign-data instrumentation were port scanners that checked if port 3389 was open, their RDP network traffic was relatively small. Nmap with service versioning and operating-system detection generated more network traffic for port 3389 than other port scanners. The exchange between Nmap and the user-interface's RDP service comprised 78 packets, with only the first phase of the RDP connection sequence being completed. Nmap in its non-versioning configuration and Angry IP Scanner generated the least amount of network traffic. The exchanges with the RDP server using both tools were four packets long and only completed the TCP three-way handshake before resetting the connection.

Nmap was also the only tool to use the RDP connection sequence when configured to detect the version of RDP service being used. Because of this, we could detect a signature for Nmap in the RDP cookie field of the Client X.224 Connection Request PDU that is sent to the RDP server in the first phase of the connection sequence (Microsoft, 2022a). Since the PDUs sent were unencrypted, the string "nmap" was logged when Nmap was trying to connect as a client to discover the version of RDP service being used. Figure 9 shows where this string was found from analysis in Wireshark.



No.	Time	Source	Destination	Protocol	Length	Info
15	2022-08-16 21:16:37.480239	[redacted]	[redacted]	TCP	58	59277 → 3389 [SYN]
16	2022-08-16 21:16:37.480277	[redacted]	[redacted]	TCP	58	3389 → 59277 [SYN,
17	2022-08-16 21:16:37.797052	[redacted]	[redacted]	TCP	74	39150 → 3389 [SYN]
18	2022-08-16 21:16:37.797097	[redacted]	[redacted]	TCP	74	3389 → 39150 [SYN,
19	2022-08-16 21:16:37.825237	[redacted]	[redacted]	TCP	66	39150 → 3389 [ACK]
20	2022-08-16 21:16:38.512029	[redacted]	[redacted]	TCP	58	[TCP Retransmission
21	2022-08-16 21:16:40.528030	[redacted]	[redacted]	TCP	58	[TCP Retransmission
22	2022-08-16 21:16:43.885655	[redacted]	[redacted]	SSL	108	Continuation Data
23	2022-08-16 21:16:43.885706	[redacted]	[redacted]	TCP	66	3389 → 39150 [ACK]
24	2022-08-16 21:16:43.943056	[redacted]	[redacted]	SSL	85	Continuation Data
25	2022-08-16 21:16:43.966301	[redacted]	[redacted]	TCP	66	39150 → 3389 [ACK]
26	2022-08-16 21:16:43.966301	[redacted]	[redacted]	TCP	66	39150 → 3389 [FIN,
27	2022-08-16 21:16:43.968090	[redacted]	[redacted]	TCP	66	3389 → 39150 [FIN,
28	2022-08-16 21:16:43.993425	[redacted]	[redacted]	TCP	66	39150 → 3389 [ACK]

```
> Frame 22: 108 bytes on wire (864 bits), 108 bytes captured (864 bits)

0000  26 ac 19 93 ba e7 fe 00  00 00 01 01 08 00 45 00   &·······  ······E·
0010  00 5e b9 96 40 00 35 06  78 09 47 ca 09 5b a5 e3   ·^··@·5·  x·G··[··
0020  1c f2 98 ee 0d 3d 40 6e  cc 43 dd 7d e3 0e 80 18   ·····=@n  ·C·}····
0030  01 f6 53 ca 00 00 01 01  08 0a 9c c5 0b 54 f7 26   ··S·····  ·····T·&
0040  d6 e2 03 00 00 2a 25 e0  00 00 00 00 00 43 6f 6f   ·····*%·  ·····Coo
0050  6b 69 65 3a 20 6d 73 74  73 68 61 73 68 3d 6e 6d   kie: mst  shash=nm
0060  61 70 0d 0a 01 00 08 00  03 00 00 00               ap······  ····
```

Figure 9. The String "nmap" in the Client X.224 Connection Request PDU of an Nmap Service Detection Scan

b. Characteristics of Malicious RDP Network Traffic

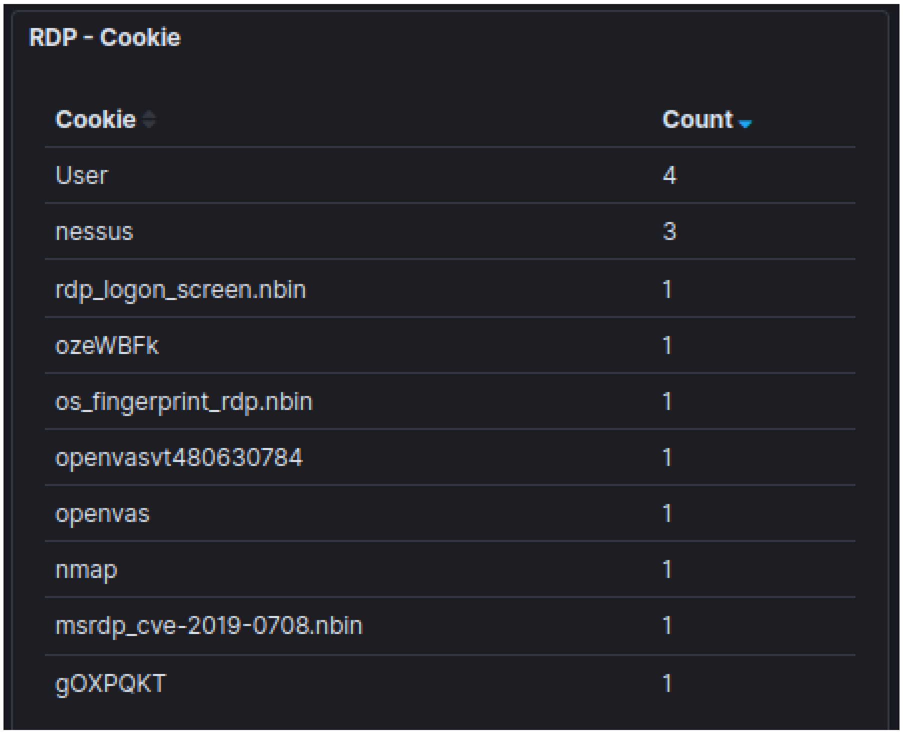
In analyzing network traffic created by our vulnerability scanners and Metasploit Framework modules, we saw more traffic compared to the port scanners used for benign-data testing. The tool for instrumenting malicious traffic that generated the fewest packets to port 3389 was the Metasploit ms12_020_maxchannelids exploit module. The exchange between this module and the RDP service of the user interface was 21 packets. Nessus generated the most RDP traffic, exchanging over 2000 packets when scanning port 3389 of the user-interface machine. OpenVAS exchanged 981 packets. The BlueKeep exploit module generated 205 and 272 packets for the two configurations used, and the Rdpscan vulnerability scanner generated 96 packets. Table 2 lists the packet counts for each tool.

Table 2. Experiment 5 – Number of Packets Generated by Malicious Tools

Packets Exchanged	Tool
2061	Nessus
981	OpenVAS
272	Metasploit BlueKeep
205	Metasploit BlueKeep (ForceExploit)
96	Rdpscan
21	Metasploit ms12_020_maxchannelids

Signatures for Nessus and OpenVAS were also found in the RDP cookie field of the Client X.224 Connection Request PDU that is sent in the first phase of RDP connection. Figure 10 shows counts of the RDP cookies used by our tools in Experiment 5, as reported by Malcolm. Different tools use different RDP cookie values when communicating with an RDP server. The “count” column indicates how many times a specific cookie value was used. For vulnerability scanning traffic generated by Nessus, the string “nessus” was used three times as a value for the RDP cookie. Other Nessus-related RDP cookies contained the string “<xxx>.nbin,” the filename of a plugin; “<xxx>“ identifies the particular plugin and “.nbin” indicates the binary code of a Nessus plugin (Tenable, 2022a). Figure 10 shows three RDP cookies with the substring “.nbin” for three different Nessus plugins used in vulnerability detection, each of which was seen once for RDP vulnerability assessment.

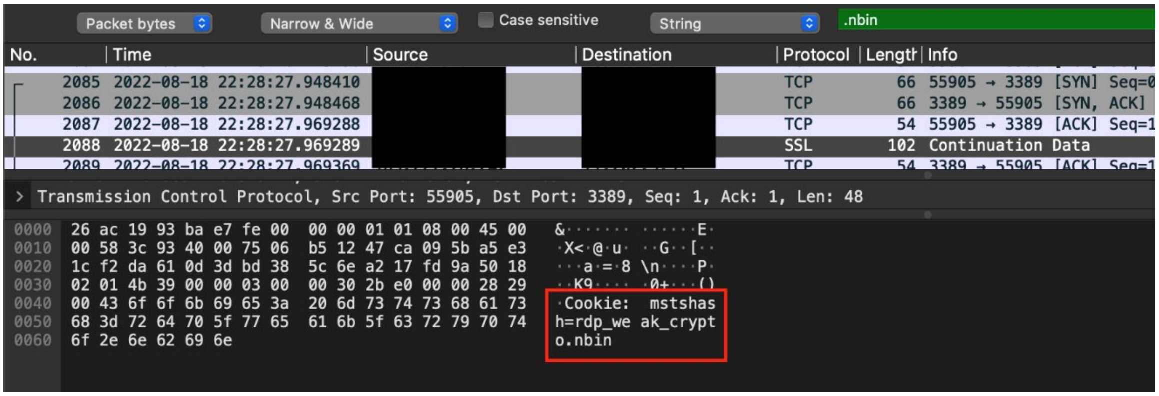
Two more Nessus plugins undetected by Malcolm were found in Wireshark by searching packets for the string ".nbin," shown in Figure 11 and Figure 12. For vulnerability scanning traffic generated by OpenVAS, "openvas" and "openvasvt<###>" were used as cookie values, where "openvasvt" indicates a network vulnerability test and "<###>" is the numerical identifier for that test (Greenbone Networks, 2022). These are also shown in Figure 10.



RDP - Cookie

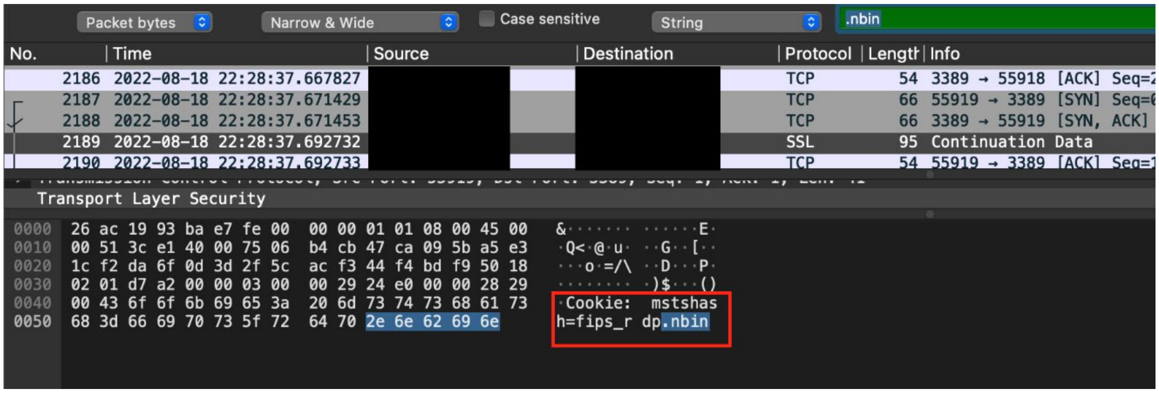
Cookie	Count
User	4
nessus	3
rdp_logon_screen.nbin	1
ozeWBFk	1
os_fingerprint_rdp.nbin	1
openvasvt480630784	1
openvas	1
nmap	1
msrdp_cve-2019-0708.nbin	1
gOXPQKT	1

Figure 10. Analysis of RDP Cookie values Using Malcolm from Data Generated in Experiment 5



```
No.        Time                         Source      Destination   Protocol  Length Info
2085  2022-08-18 22:28:27.948410   [redacted]  [redacted]    TCP        66  55905 → 3389 [SYN] Seq=0
2086  2022-08-18 22:28:27.948468                             TCP        66  3389 → 55905 [SYN, ACK]
2087  2022-08-18 22:28:27.969288                             TCP        54  55905 → 3389 [ACK] Seq=1
2088  2022-08-18 22:28:27.969289                             SSL       102  Continuation Data
2089  2022-08-18 22:28:27.969369                             TCP        54  3389 → 55905 [ACK] Seq=1

> Transmission Control Protocol, Src Port: 55905, Dst Port: 3389, Seq: 1, Ack: 1, Len: 48

0000  26 ac 19 93 ba e7 fe 00  00 00 01 01 08 00 45 00   &·······  ······E·
0010  00 58 3c 93 40 00 75 06  b5 12 47 ca 09 5b a5 e3   ·X<·@·u·  ··G··[··
0020  1c f2 da 61 0d 3d bd 38  5c 6e a2 17 fd 9a 50 18   ···a·=·8 \n····P·
0030  02 01 4b 39 00 00 03 00  00 30 2b e0 00 00 28 29   ··K9····  ·0+···()
0040  00 43 6f 6f 6b 69 65 3a  20 6d 73 74 73 68 61 73   ·Cookie:  mstshas
0050  68 3d 72 64 70 5f 77 65  61 6b 5f 63 72 79 70 74   h=rdp_we ak_crypt
0060  6f 2e 6e 62 69 6e                                  o.nbin
```

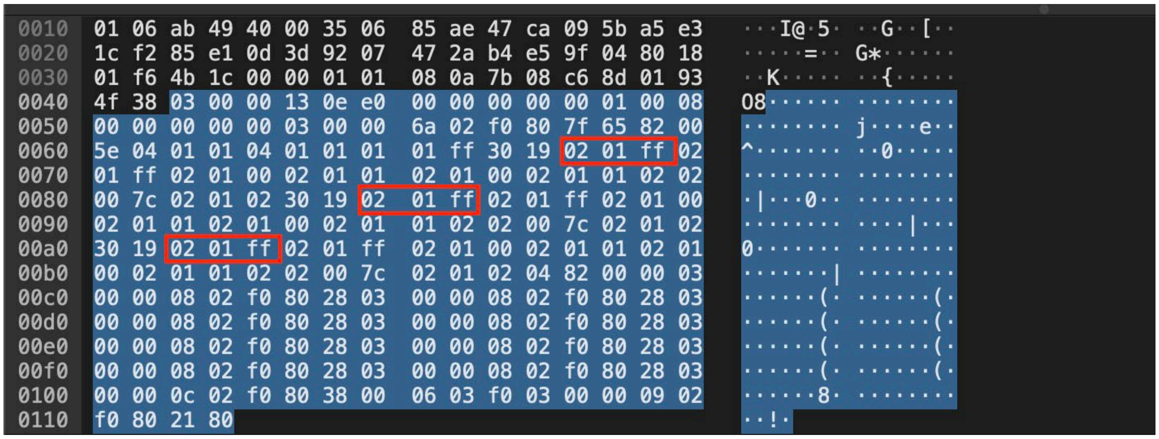
Figure 11. A Nessus Plugin for Detecting Cryptographic Vulnerabilities in RDP Identified by the Filename “rdp_weak_crypto.nbin” in Wireshark



```
No.        Time                         Source      Destination   Protocol  Length Info
2186  2022-08-18 22:28:37.667827   [redacted]  [redacted]    TCP        54  3389 → 55918 [ACK] Seq=2
2187  2022-08-18 22:28:37.671429                             TCP        66  55919 → 3389 [SYN] Seq=0
2188  2022-08-18 22:28:37.671453                             TCP        66  3389 → 55919 [SYN, ACK]
2189  2022-08-18 22:28:37.692732                             SSL        95  Continuation Data
2190  2022-08-18 22:28:37.692733                             TCP        54  55919 → 3389 [ACK] Seq=1

  Transport Layer Security

0000  26 ac 19 93 ba e7 fe 00  00 00 01 01 08 00 45 00   &·······  ······E·
0010  00 51 3c e1 40 00 75 06  b4 cb 47 ca 09 5b a5 e3   ·Q<·@·u·  ··G··[··
0020  1c f2 da 6f 0d 3d 2f 5c  ac f3 44 f4 bd f9 50 18   ···o·=/\  ··D···P·
0030  02 01 d7 a2 00 00 03 00  00 29 24 e0 00 00 28 29   ········  ·)$···()
0040  00 43 6f 6f 6b 69 65 3a  20 6d 73 74 73 68 61 73   ·Cookie:  mstshas
0050  68 3d 66 69 70 73 5f 72  64 70 2e 6e 62 69 6e      h=fips_r dp.nbin
```

Figure 12. A Nessus Plugin for Identifying RDP Vulnerabilities with Respect to Federal Information Processing Standards (FIPS) Compliance Identified by the Filename “fips_rdp.nbin” in Wireshark

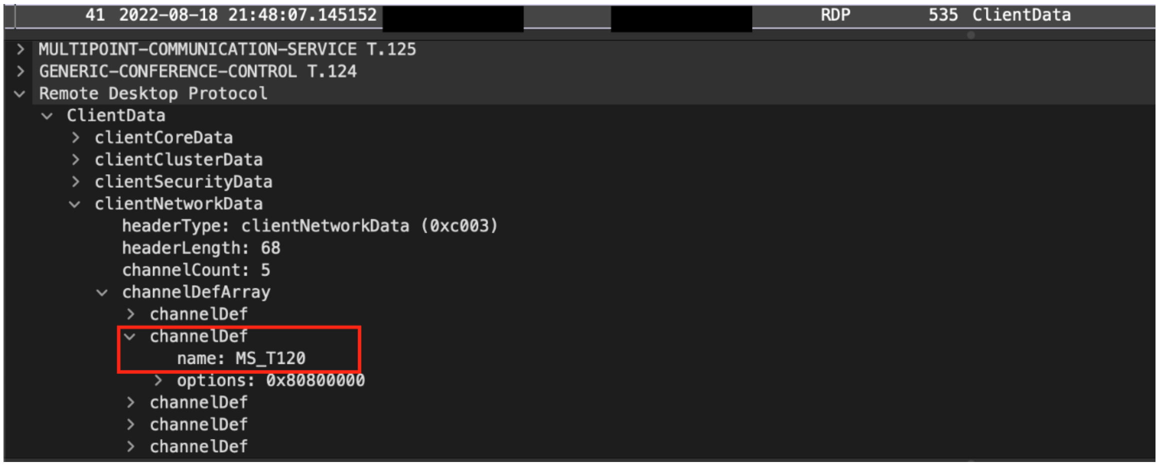
The payload signature for the Metasploit ms12_020_maxchannelids exploit was also found through packet analysis in Wireshark. The entire malicious payload used by this module can be observed in cleartext. From analysis of the source code of this module, one malicious packet is crafted with content for multiple steps of the RDP connection sequence, which is then sent to the RDP server without waiting for the server’s response. This means that the whole malicious packet can be clearly observed before encryption is started. Figure 13 shows the hex dump of a packet that was sent by the exploit module, in which the blue highlighted area matches the payload seen in its source code. The red highlighted areas of this image show the malicious value used for the maxChannelIDs field, which causes the denial-of-service condition to happen.



```
0010  01 06 ab 49 40 00 35 06  85 ae 47 ca 09 5b a5 e3   ···I@·5· ··G··[··
0020  1c f2 85 e1 0d 3d 92 07  47 2a b4 e5 9f 04 80 18   ·····=·· G*······
0030  01 f6 4b 1c 00 00 01 01  08 0a 7b 08 c6 8d 01 93   ··K····· ··{·····
0040  4f 38 03 00 00 13 0e e0  00 00 00 00 00 01 00 08   O8······ ········
0050  00 00 00 00 00 03 00 00  6a 02 f0 80 7f 65 82 00   ········ j····e··
0060  5e 04 01 01 04 01 01 01  01 ff 30 19 02 01 ff 02   ^······· ··0·····
0070  01 ff 02 01 00 02 01 01  02 01 00 02 01 01 02 02   ········ ········
0080  00 7c 02 01 02 30 19 02  01 ff 02 01 ff 02 01 00   ·|···0·· ········
0090  02 01 01 02 01 00 02 01  01 02 02 00 7c 02 01 02   ········ ····|···
00a0  30 19 02 01 ff 02 01 ff  02 01 00 02 01 01 02 01   0······· ········
00b0  00 02 01 01 02 02 00 7c  02 01 02 04 82 00 00 03   ·······| ········
00c0  00 00 08 02 f0 80 28 03  00 00 08 02 f0 80 28 03   ······(· ······(·
00d0  00 00 08 02 f0 80 28 03  00 00 08 02 f0 80 28 03   ······(· ······(·
00e0  00 00 08 02 f0 80 28 03  00 00 08 02 f0 80 28 03   ······(· ······(·
00f0  00 00 08 02 f0 80 28 03  00 00 08 02 f0 80 28 03   ······(· ······(·
0100  00 00 0c 02 f0 80 38 00  06 03 f0 03 00 00 09 02   ······8· ········
0110  f0 80 21 80                                        ··!·
```

Figure 13. Hex Payload Signature of Metasploit ms12_020_maxchannelids Exploit

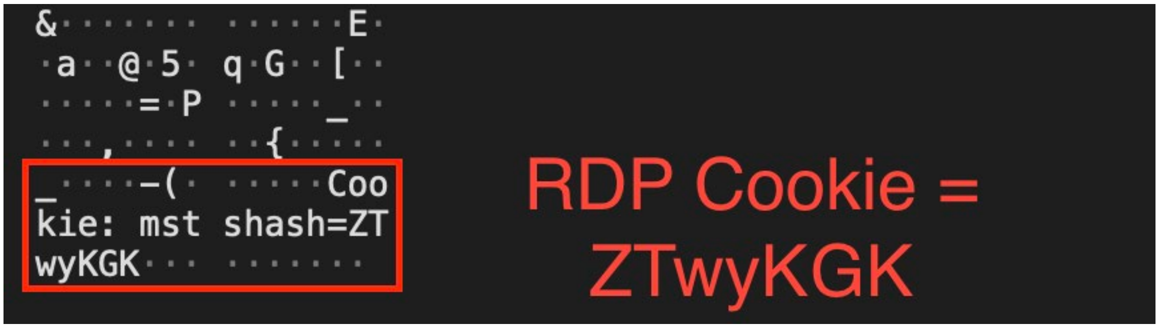
Signatures for attempted BlueKeep attacks or vulnerability scans could be observed in the basic settings-exchange phase of the RDP connection sequence. During this phase, a malicious client tried to create the MS_T120 virtual channel. The creation of this channel can be identified through decrypted RDP packets in Wireshark in Figure 14, though PyRDP MitM can automatically analyze network traffic for this signature.



```
41  2022-08-18 21:48:07.145152  [redacted]  [redacted]  RDP  535  ClientData
> MULTIPOINT-COMMUNICATION-SERVICE T.125
> GENERIC-CONFERENCE-CONTROL T.124
v Remote Desktop Protocol
  v ClientData
    > clientCoreData
    > clientClusterData
    > clientSecurityData
    v clientNetworkData
        headerType: clientNetworkData (0xc003)
        headerLength: 68
        channelCount: 5
      v channelDefArray
        > channelDef
        v channelDef
            name: MS_T120
          > options: 0x80800000
        > channelDef
        > channelDef
        > channelDef
```

Figure 14. Client Request for MS_T120 Virtual Channel During Basic Settings Exchange of RDP (Signature of BlueKeep)

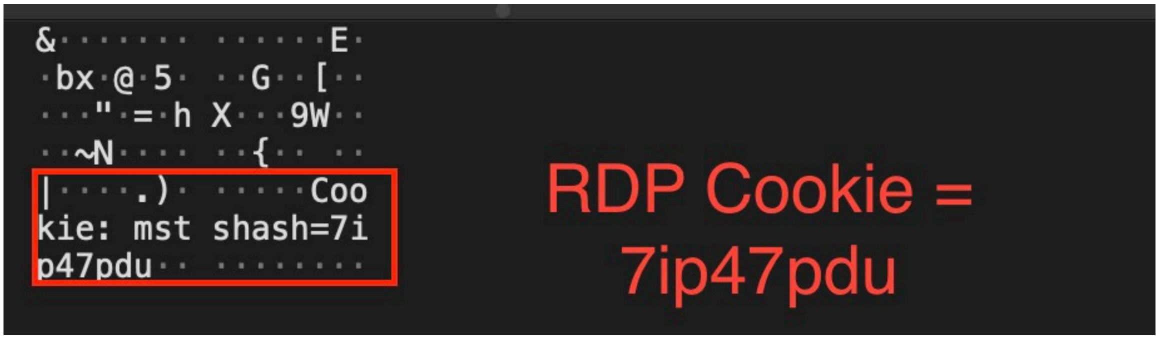
Other characteristics of BlueKeep exploitation, vulnerability scans using Metasploit modules, or the Rdpscan tool were their randomly generated RDP cookies. The description for the RDP_USER option of the Metasploit BlueKeep module (cve_2019_0708_bluekeep_rce), which is used as value for the RDP cookie, states “The username to report during connect, UNSET = random” (Rapid7, n.d.-a). From multiple tests using this module, we observed that the randomly generated value is a seven-character alphabetic string, two of which can be seen in Figure 10. Another example of a Metasploit BlueKeep cookie is shown in Figure 15. From analysis of the source code of Rdpscan, we found a function to generate an eight-character alphanumerical string for the RDP cookie’s value (RobertDavidGraham, 2019). One of these Rdpscan cookies is shown in Figure 16.



```
&·······  ······E·
·a··@·5· q·G··[··
·····=·P ·····_··
···,····  ··{·····
_····-(·  ·····Coo
kie: mst shash=ZT
wyKGK···  ·······
```

RDP Cookie = ZTwyKGK

Figure 15. Random Seven-Character Alphabetic RDP Cookie Generated by Metasploit Bluekeep Exploit



```
&·······  ······E·
·bx·@·5·  ··G··[··
···"·=·h X···9W··
··~N····  ··{··  ··
|····.)·  ·····Coo
kie: mst shash=7i
p47pdu··  ········
```

RDP Cookie = 7ip47pdu

Figure 16. Random Eight-Character Alphanumeric RDP Cookie Generated by Rdpscan BlueKeep Scanner

c. Windows Events Correlated with RDP Network Traffic

Windows event logs identify events on a Windows machine (Microsoft, 2021a). From our instrumented dataset, we observed these Windows Event IDs associated with RDP:

- 261 – TCP listener for RDP received a connection
- 1149 – User authentication succeeded for Remote Desktop Services
- 1158 – Remote Desktop Services accepted a connection
- 21 – Remote-desktop session logon succeeded
- 22 – Remote-desktop shell started (to display graphical data)
- 40 – Remote-desktop session disconnected

Windows Event ID 261 was the only event associated with RDP for benign traffic. It also correlated with Nmap when it was configured to detect a host's services and operating system. In this configuration Nmap started but did not complete the RDP connection sequence, so only this Event ID was logged.

The RDP network traffic of malicious tools triggered Event IDs 261, 1158, and 40 since they proceeded beyond the first phase of the RDP connection sequence. For Rdpscan and the BlueKeep Metasploit module, instances of Event IDs 261, 1158, and 40 were logged when they partially completed the RDP connection sequence beyond connection start. For the Nessus and OpenVAS vulnerability scanners, Event ID 261 was observed when traffic completed only the first phase of the connection sequence, but Event IDs 1158 and 40 were logged when their vulnerability tests created traffic that proceeded beyond the start. Only one instance of Event ID 261 was seen with the ms12_020_maxchannelids exploit module during connection start.

During Experiment 5 where we established our own remote-desktop session to our honeypot, we observed Event IDs 261, 1149, 21 and 22. Each event could be correlated to the times that we established a connection, successfully authenticated ourselves, and received the graphical output of the Windows desktop. We could also correlate Event ID 40 to when we disconnected the session by closing the RDP-client software.

3. RDP Attack-Characterization Methods

Using these observed characteristics of benign and malicious RDP network traffic, we created criteria to identify attacks on RDP. Our approach involves counting features of the attack data to first classify it as malicious or benign, then analyzing the content and events of the attack to classify its severity.

We began by identifying the RDP connections that are clearly malicious, those that contain cleartext signatures of known malicious attacks. These cleartext signatures include the Metasploit ms12_020_maxchannelids payload signature, and the RDP cookie signatures that we derived from malicious data instrumentation for Nessus, OpenVAS, Rdpscan, and Metasploit BlueKeep. Upon identifying a clearly malicious connection, all traffic from the source that created it is classified as clearly malicious. We then sequentially classify the remaining traffic as very suspicious, suspicious, or benign based on the size of the stream and the source from which it came. An RDP connection is classified as very suspicious if more than 500 packets are exchanged between the client and server, since this suggested that the RDP connection sequence has completed and a desktop session has been established. When a very suspicious RDP connection is identified, all network traffic from the source is also classified as very suspicious despite its size. Suspicious classifications occur if an exchange of 85 to 500 packets is observed during an RDP connection, as this indicates the partial completion of the RDP connection sequence after the connection start. All network traffic from the sources of suspicious connections is also classified as suspicious despite size. RDP connections that are less than 85 packets and from nonmalicious sources are classified as benign. The 85-packet cutoff point was the lower bound for the number of packets generated by a malicious tool, meaning a connection with less than 85 packets exchanged is likely a connection made by a port scanner. Figure 17 depicts the process for classifying benign and malicious RDP traffic.

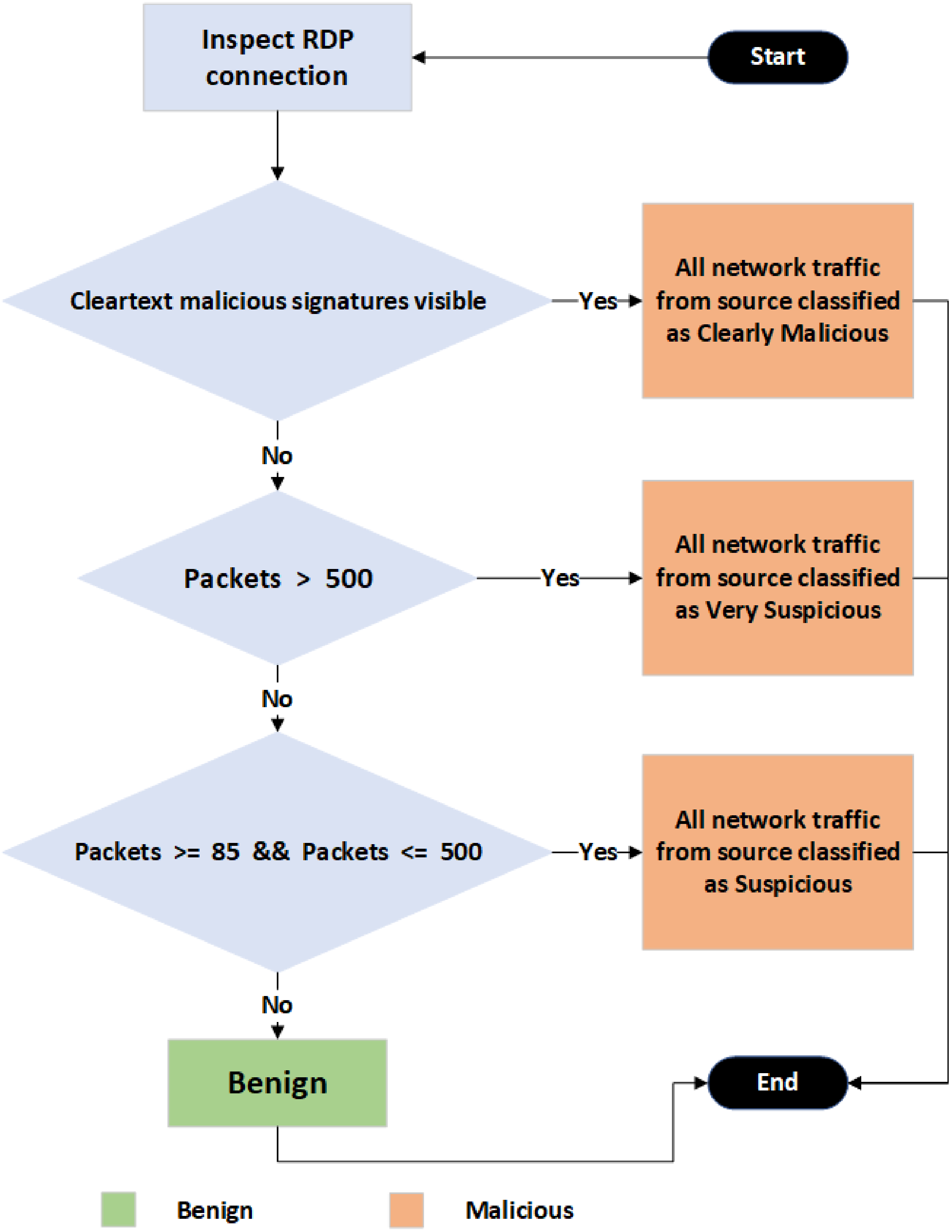

Start
Inspect RDP connection
Cleartext malicious signatures visible
Yes
All network traffic from source classified as Clearly Malicious
No
Packets > 500
Yes
All network traffic from source classified as Very Suspicious
No
Packets >= 85 && Packets <= 500
Yes
All network traffic from source classified as Suspicious
No
Benign
End
Benign
Malicious

Figure 17. Logic for Malicious or Benign Classification of RDP Traffic

We determine the severity of attacks after they have been classified as malicious or benign. Clearly malicious attacks in our experiments were classified as low severity since they are either vulnerability scans which do not harm our Windows machine or exploits that are ineffective against our Windows machine. The severity level of very suspicious traffic can be between low and very high based on the duration of the RDP connection and log records indicating malicious activity. Very suspicious attacks that last less than one minute are classified as low severity. Attacks that last between one and five minutes, but do not generate malicious event-log entries, are classified as moderate severity, whereas attacks that have generated malicious event-log entries over the same period are classified as high severity. Attacks that last longer than five minutes are classified as high severity but are elevated to very high severity if malicious event-log entries are observed. For suspicious traffic, we classify severity by decrypting RDP network traffic to search for signatures of BlueKeep. Suspicious attacks that display the BlueKeep MS_T120 signature are low severity since the exploit is ineffective against our version of Windows; attacks that do not display the BlueKeep MS_T120 signature are very low severity. All benign traffic is classified as very low severity as it is often scanning that does not proceed beyond RDP connection start. Figure 18 shows the process for classifying the severity of an RDP connection.

If an RDP connection that would normally not meet the criteria to be characterized as clearly malicious, very suspicious, or suspicious is classified as such by association with a malicious source, then it receives a severity classification equal to the highest severity classification of the malicious source. For example, if a benign port scan is characterized as very suspicious for being from the same source as a very suspicious high-severity connection, then it would also be classified as high severity.

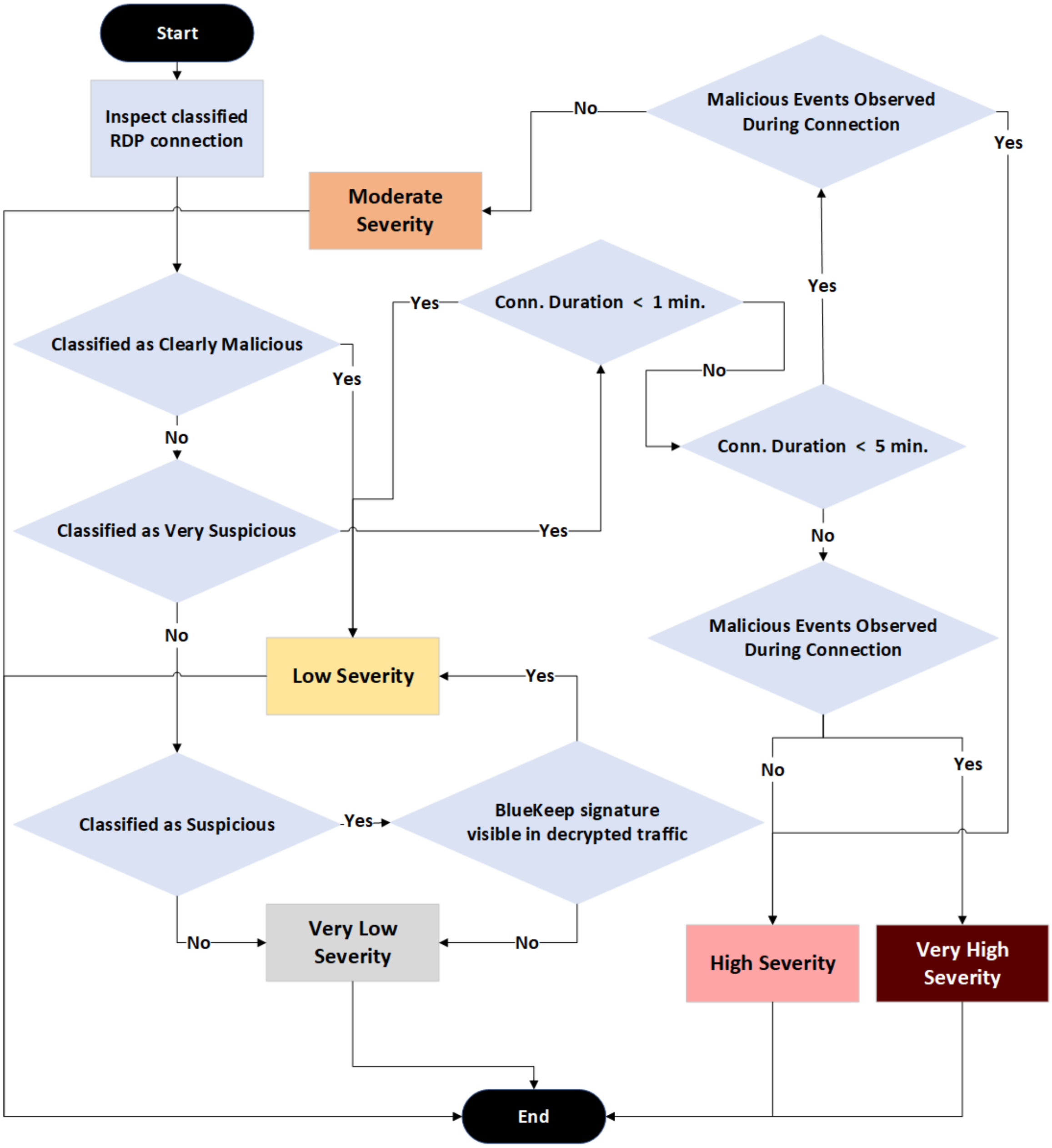

Start
Inspect classified RDP connection
Moderate Severity
Malicious Events Observed During Connection
No
Yes
Classified as Clearly Malicious
Yes
No
Conn. Duration < 1 min.
Yes
No
Conn. Duration < 5 min.
Yes
No
Classified as Very Suspicious
Yes
No
Low Severity
Malicious Events Observed During Connection
No
Yes
Classified as Suspicious
Yes
BlueKeep signature visible in decrypted traffic
Yes
No
No
Very Low Severity
High Severity
Very High Severity
End

Figure 18. Logic for Classifying the Severity of RDP Traffic

Classifying the severity of very suspicious traffic relies heavily on the observation of log data that we associate with malicious activity. These malicious events include changes to the Windows registry, running PowerShell commands, creation or deletion of files, DNS requests, and creation of user processes. Windows events associated with these actions are indicated by the following Event IDs:

- 12 – Registry object added or deleted
- 40961 – PowerShell console is starting up
- 800 – Pipeline execution details for command line
- 11 – File creation
- 23 – File deletion
- 22 – DNS query
- 1 – Process creation

4. Methods for Evaluation

Honeypots typically collect thousands of streams of network traffic, and analyzing them to determine if our characterization methods are correct would require much time. As such, our evaluation methods only sample part of the total network traffic received by our honeypot.

Traffic we classify as clearly malicious must include signatures of attacks that we characterize as malicious. While clearly malicious attacks can be confirmed through signatures in the form of RDP cookies like the ones use by tools like Nessus, OpenVAS, and Metasploit ms12_020_maxchannelids, possibly some RDP clients also used randomly generated cookies without attempting to scan or exploit BlueKeep through Rdpscan or Metasploit. Hence, evaluation of BlueKeep attack characterization through observation of randomly generated seven-to-eight-character RDP cookies must confirm whether an RDP client using a random cookie attempted to create the MS_T120 virtual channel.

Analyzing attacks classified as very suspicious where remote-desktop sessions were successfully established requires the most effort. To determine if our characterizations were correct, we must search the event logs to determine if any malicious events occurred, then compare the observed events to the actual changes that can be observed on the Windows machine and the patterns of collected traffic. For experiments using PyRDP MitM, we can view replays of sessions to see what the attacker did. We can also check PyRDP logs to see whether disk drives were attached to the session to transfer files, and if the clipboard transferred data during the session. For experiments without PyRDP, we can check for malicious events in the Windows event logs, then log in to the Windows system to find attack artifacts such as saved PowerShell transcripts, new files, and new registry keys. We can also observe outbound or inbound connections to the Windows machine during or after the session that might indicate an attacker communicating with a command-and-control server with malware that may have been installed.

5. Weaknesses of Our Attack-Characterization Methodology

Since our some of our RDP attack-characterization methods rely on the RDP cookie value in client connection requests, attackers could modify the value of the RDP cookie to fool us. For BlueKeep scripts, an attacker might set the value of the RDP cookie to something other than the randomly generated default value, which would characterize their RDP network traffic as benign instead of malicious if a tool like PyRDP is not used. Also, an attacker who could change the RDP cookie value used by vulnerability scanners like Nessus and OpenVAS might remove signatures that would indicate vulnerability scanning of the Windows machine.

Another weakness of this methodology is that much like a signature-based intrusion-detection system, it can only characterize attacks whose signatures and characteristics are currently known. This limits our methods to work only from the attack data that we generated from our choice of port scanners, vulnerability assessment tools, and Metasploit modules. A malicious attack that has not yet been analyzed is characterized as benign.

THIS PAGE INTENTIONALLY LEFT BLANK

V. RESULTS AND DISCUSSION

A. EXPERIMENT 1 RESULTS

1. Observed RDP Network Traffic

Our first experiment using PyRDP occurred from April 15, 2022 to May 8, 2022. The user-interface machine of our honeypot received 134804 RDP connections from 53 countries during the twenty-three days that it was made publicly accessible. The first external RDP connection occurred about eighteen minutes after the DigitalOcean firewall was adjusted to allow inbound traffic to port 3389. Most RDP connections were from the United States, Pakistan, and South Korea, respectively. Figure 19 shows the RDP connections to our user-interface machine by country.

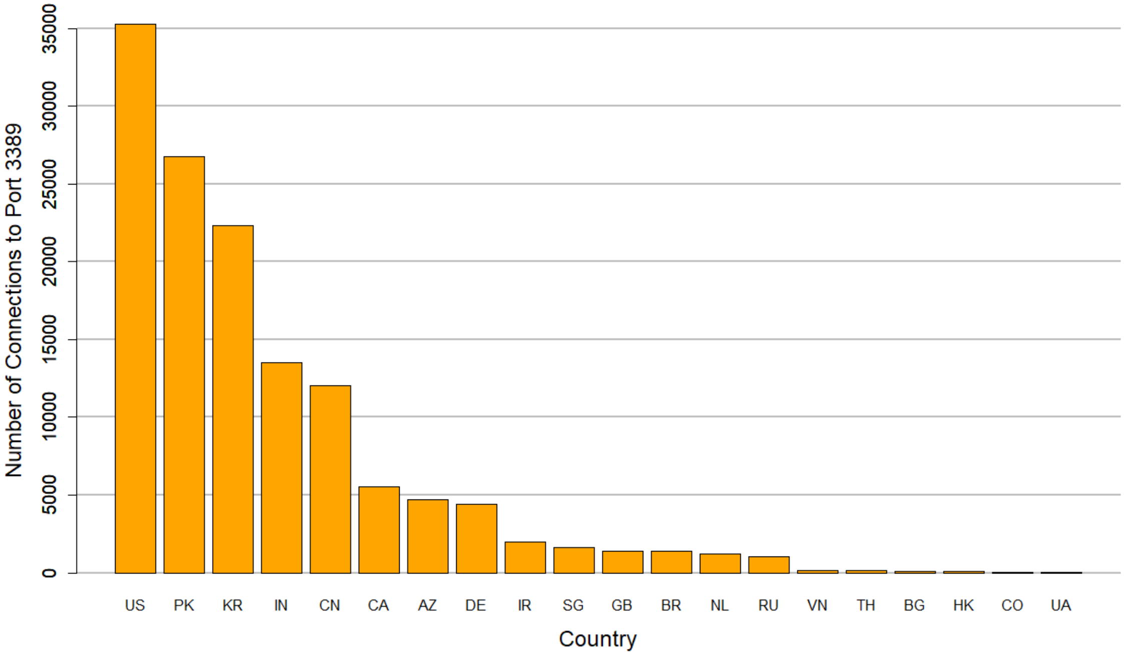
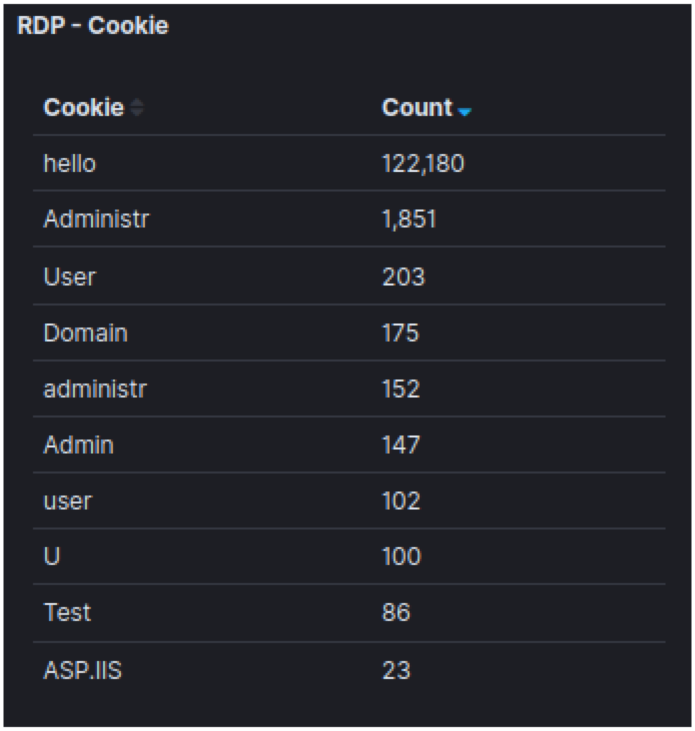


Figure 19. Experiment 1 – RDP Connections by Country

Analysis of the top ten RDP cookies showed us that about 90% of the RDP connections to our honeypot used the cookie “hello.” This cookie value is used by web crawlers searching for RDP services that are open to the Internet, and is also sent to ports other than 3389 to search for remote-desktop services running on non-standard RDP ports (Bruneau, 2018). Figure 20 shows the top ten RDP cookies observed during Experiment 1. Table 3 details the top ten sources who used “hello” as their RDP cookie when connecting to our honeypot. In this experiment, none of the top ten sources of “hello” created RDP connections that used a different cookie value.



RDP - Cookie

Cookie	Count
hello	122,180
Administr	1,851
User	203
Domain	175
administr	152
Admin	147
user	102
U	100
Test	86
ASP.IIS	23

Figure 20. Experiment 1 – Top Ten RDP Cookies Used in Client X.224 Connections

Table 3. Experiment 1 – Top Ten Sources of RDP Cookie “hello”

Count	Source IP Address	Source Country
22293	27.102.106.117	South Korea
21061	119.160.107.37	Pakistan
10992	122.185.97.98	India
7164	123.58.211.34	Hong Kong
6552	45.63.70.82	United States
6326	47.242.107.32	United States
5702	58.27.205.130	Pakistan
4669	31.171.72.162	Azerbaijan
4379	182.443.199.172	China
3949	173.10.190.253	United States

Also shown in Figure 20 are the RDP cookies “Administr” and “administr” which were frequent in connections. “Administr” was one of the top three cookies received in (Bruneau, 2021), and a user on GitHub issue page claimed that the “administr” value was used by Russian web crawlers (Tabdiukov, 2018). Table 4 shows the top five sources who used “Administr” or “administr” in their connections to our honeypot.

Table 4. Experiment 1 – Top Five Sources of “Administr” and Source of “administr” RDP Cookies

RDP Cookie	Count	Source IP Address	Source Country
Administr	415	194.28.112.140	Netherlands
Administr	200	20.124.213.72	United States
Administr	144	165.22.56.114	Singapore
Administr	100	20.25.52.248	United States
Administr	100	20.84.98.196	United States
administr	152	193.56.29.176	United Kingdom

2. Characterizing RDP Attacks

We classified 1052 RDP connections as clearly malicious with low severity; 21 RDP connections were observed with signatures of Rdpscan (Table 5), and their sources had created 1031 other RDP connections. We did not find any randomly generated cookies that indicated the use of the Metasploit BlueKeep module. No signatures of Nessus, OpenVAS, and the Metasploit ms12_020_maxchannelids module occurred, indicating that attackers were not using them to do a vulnerability scan or to attack our user-interface machine.

Table 5. Experiment 1 – Suspected BlueKeep Scanning and Exploitation

Time (PST)	RDP Cookie	Source IP Address	Source Country	Source Local Time	Source Connections to Port 3389
04/16/2022 09:47	pbgqnfb1	66.240.192.138	United States	09:47	4
04/17/2022 15:32	7rco04f4	193.118.53.194	Netherlands	00:32	3
04/17/2022 20:43	r4w3hjy4	165.227.62.247	United States	20:43	4
04/21/2022 06:10	n46ilmig	184.105.247.195	United States	06:10	3
04/24/2022 01:47	b2ffqykn	206.189.23.56	United Kingdom	09:47	1
04/24/2022 17:23	21bmwl60	93.174.95.106	Netherlands	02:23	4
04/25/2022 05:19	icv4jmhq	89.248.165.140	Netherlands	13:19	6
04/28/2022 02:05	wkfqlwrl	89.248.165.140	Netherlands	11:05	6
04/28/2022 14:57	zaskab0z	71.6.167.124	United States	14:57	4
04/28/2022 18:45	r56cr0f0	74.82.47.4	United States	18:45	6
04/28/2022 22:19	dgjs3zr0	192.53.122.89	United States	22:19	1
04/29/2022 15:54	umvhdym2	143.198.238.87	United States	15:54	8
04/29/2022 17:34	k6r2yks2	143.198.238.87	United States	17:34	8
05/03/2022 04:42	sluw57id	74.82.47.4	United States	04:42	6
05/05/2022 07:43	cg0hucvj	193.118.53.194	Netherlands	16:43	3
05/06/2022 02:05	wkfqlwrl	89.248.165.140	Netherlands	11:05	6

Time (PST)	**RDP Cookie**	**Source IP Address**	**Source Country**	**Source Local Time**	**Source Connections to Port 3389**
05/06/2022 04:30	zo7d4y0l	89.248.171.133	Netherlands	13:30	999
05/07/2022 02:38	AP6j6MwU	194.28.112.140	Netherlands	11:38	6
05/07/2022 02:41	0dgqbzbx	194.28.112.140	Netherlands	11:41	6
05/07/2022 02:47	6reyUYmB	194.28.112.140	Netherlands	11:41	6
05/07/2022 20:59	g5ml5paq	74.82.47.5	United States	20:59	3

Nine connections were classified as very suspicious with low severity; four RDP connections exchanged more than 500 packets with the user-interface machine, and five other RDP connections were created by their sources. We confirmed establishment of remote-desktop sessions through Windows event logs for the four connections with exchanges greater than 500 packets, though each session was disconnected almost immediately after it began. Table 6 lists the RDP connections that had established remote-desktop sessions in this experiment. No malicious Windows events were observed between connection and disconnection from the Windows machine, so the severity of all RDP connections created by the sources in Table 6 were classified as low.

Table 6. Experiment 1 – RDP Connections with Established Remote-Desktop Sessions with No Interaction

Time (PST)	**Duration of RDP Connection (minutes)**	**Source IP Address**	**Source Country**	**Source Local Time**	**Severity of Connection**
04/24/2022 01:38	< 1	66.228.47.31	United States	01:38	Low
04/28/2022 22:19	< 1	194.195.243.81	Germany	07:19	Low
04/28/2022 22:25	< 1	194.195.243.81	Germany	07:25	Low
05/02/2022 22:11	< 1	193.118.55.162	Netherlands	07:11	Low

Ten RDP connections were classified as suspicious; seven RDP connections were observed exchanging between 85 and 500 packets with our server (Table 7), and three more RDP connections originated from their sources. Analysis of each connection in Table 7 showed that none tried BlueKeep scanning or exploitation against our Windows machine. Due to this and the lack of malicious events in the logs, all ten RDP connections from these suspicious sources were classified as low severity.

Table 7. Experiment 1 – Suspicious RDP Connections with Exchanges of 85 to 500 Packets

Time (PST)	**Packets Exchanged**	**Source IP Address**	**Source Country**	**Source Local Time**	**Severity of Connection**
04/25/2022 15:50	211	183.136.225.9	China	06:50	Very Low
04/25/2022 18:39	182	183.136.225.9	China	09:39	Very Low
04/26/2022 00:11	484	128.14.136.78	United States	00:11	Very Low
04/26/2022 23:30	198	185.70.186.145	Netherlands	08:30	Very Low
04/27/2022 10:43	232	94.232.47.92	Netherlands	19:43	Very Low
05/06/2022 09:11	224	183.136.225.9	China	00:11	Very Low
05/06/2022 12:08	196	183.136.225.9	China	03:08	Very Low

After applying all methods of detecting malicious RDP network traffic through signature detection and RDP connection metadata, we characterized 1071 RDP connections as malicious. The remaining 133733 connections were characterized as benign with very low severity because none sent malicious RDP network traffic, and none came from sources that did. Table 8 shows the results of our characterization methods on RDP network traffic observed in this experiment.

Table 8. Experiment 1 – Results of RDP Attack-Characterization Methodology

Set of RDP Network Traffic	Traffic Classification	Severity	Count
Malicious	Clearly Malicious	Low	1052
Malicious	Very Suspicious	Very High	0
		High	0
		Moderate	0
		Low	9
Malicious	Suspicious	Low	0
		Very Low	10
Benign	Benign	Very Low	133734

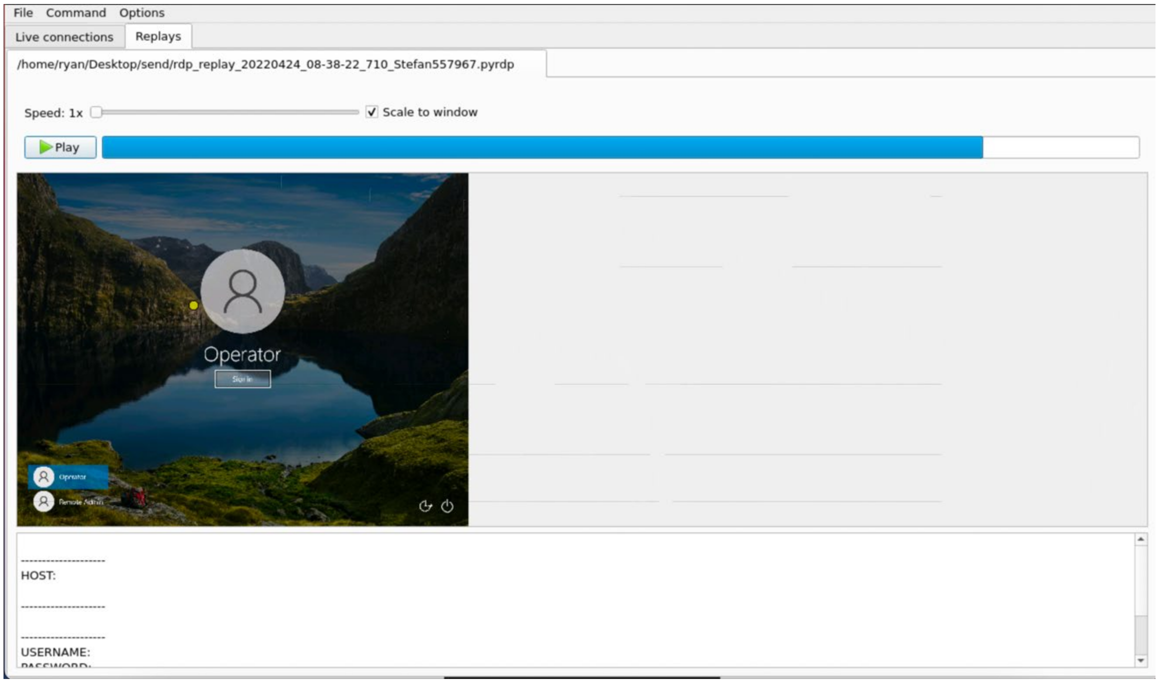
3. Evaluating Characterization Methods

In Experiment 1, PyRDP automatically logged BlueKeep events which we used to evaluate our method for detecting BlueKeep scanning and exploitation through observation of randomly generated RDP cookies. Of the 21 randomly generated eight-character alphanumeric RDP cookies that we observed, only 17 were related to BlueKeep scanning attempts; the other four RDP connections had used a randomly generated cookie but did not attempt BlueKeep scanning or exploitation. Table 9 shows the confusion matrix for BlueKeep characterization for all observed RDP connections in this experiment. The true positive rate of our BlueKeep characterization method was 89.4%, indicating that we could identify BlueKeep almost every time. The precision of the detection method was only 80%.

Table 9. Experiment 1 – Confusion Matrix for BlueKeep Detection Based on Randomly Generated RDP Cookies

n = 134753	BlueKeep	Not BlueKeep
Characterized as BlueKeep	17	4
Not Characterized as BlueKeep	2	134730

For evaluation of “very suspicious” characterizations, we used the video replays generated by PyRDP for each graphical remote-desktop session. Since each RDP connection was disconnected in under a minute, we saw that no attackers made it beyond the Windows login screen. From the PyRDP replays and lack of malicious event logs, we confirmed the severity of these attacks as low. Figure 21 shows an example of a short remote-desktop session being reviewed using the PyRDP Player tool.



/home/ryan/Desktop/send/rdp_replay_20220424_08-38-22_710_Stefan557967.pyrdp

```
--------------------
HOST:
--------------------
--------------------
USERNAME:
PASSWORD:
```

Figure 21. Experiment 1 - PyRDP Replay of an Established Remote-Desktop Session

We characterized all RDP connections in the suspicious category as very low severity because we did not observe any attempts to scan or exploit BlueKeep on our Windows machine. We evaluated these characterizations by comparing them against the BlueKeep logs generated by PyRDP. No RDP connections classified as suspicious were observed in PyRDP’s BlueKeep logs.

Evaluation of traffic that was characterized as “benign” also used PyRDP’s BlueKeep logs. Two connections responsible for BlueKeep events detected by PyRDP used the cookie “admin,” but each was only 18 packets long. Since the cookies were not randomly generated and the connection exchanged few packets with the Windows machine, these attacks fooled our methods for detection and characterization of BlueKeep attacks. Besides these two attacks, we did not find other evidence that would question the possible malicious intent of RDP connections that were initially characterized as benign.

B. EXPERIMENT 2 RESULTS

1. Observed RDP Network Traffic

Experiment 2 ran from July 7, 2022 to July 15, 2022 without PyRDP and used the same public IP address from Experiment 1. Due to an error when managing the storage of data on the logging machine, several packet captures were unintentionally deleted. As a result, network traffic collected was only comprehensive from July 12 to July 15. Analysis focused on this period.

From July 12 to July 15, our user-interface machine received 13676 RDP connections from 24 different countries (Figure 22). Most connections were from the United States, followed by Russia, then South Korea.

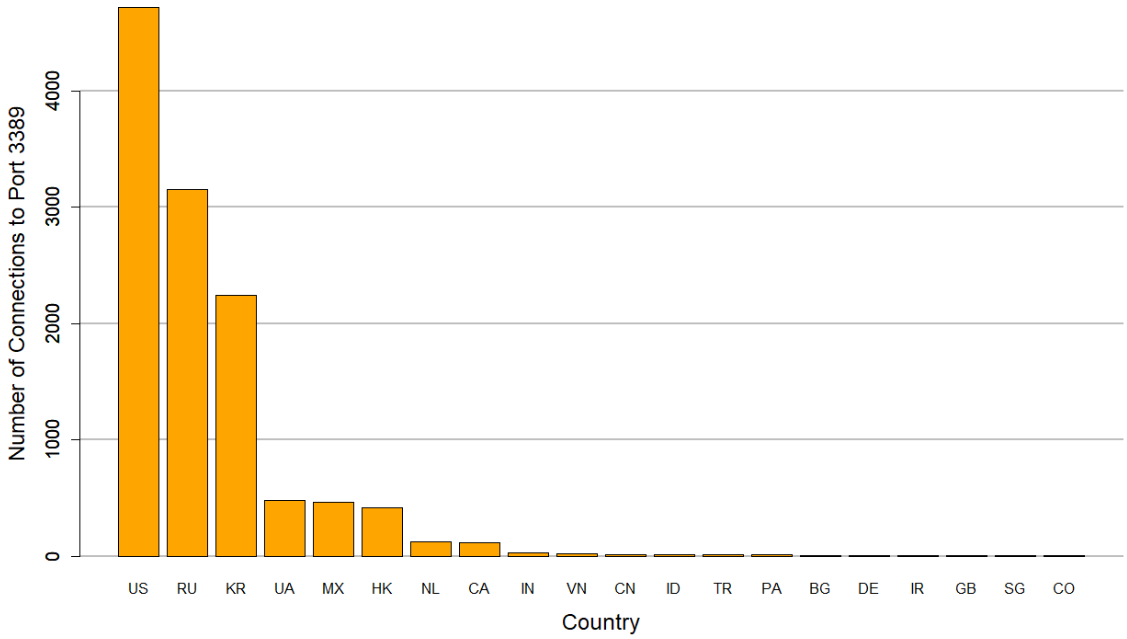
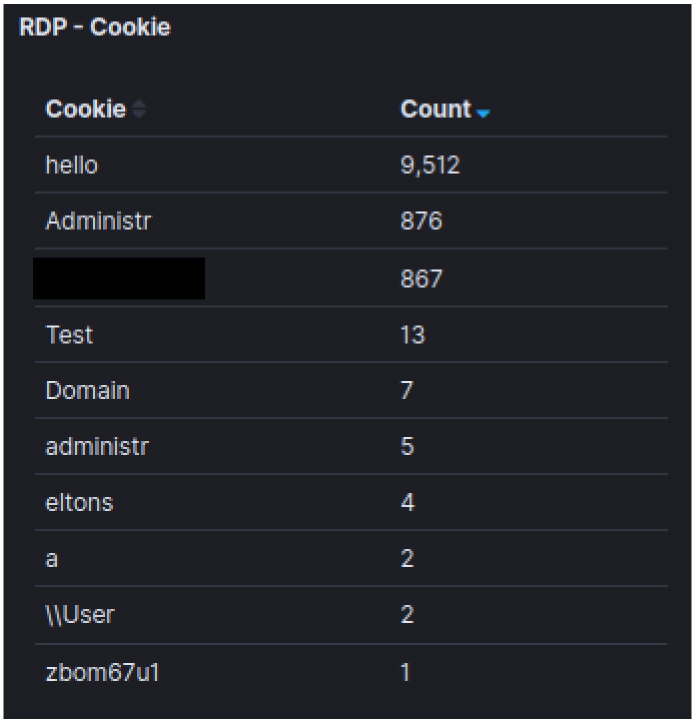


Figure 22. Experiment 2 – RDP Connections by Country

The top ten RDP cookies used during RDP connections are displayed in Figure 23. Like the results of Experiment 1, the most used RDP cookies were “hello” and “Administr,” accounting for about 75% of all cookies used by RDP connections. The top ten sources of “hello” are listed in Table 10. In Figure 23, the redacted RDP cookie value is the name of the Windows machine of our honeypot. Only two sources used the name of our Windows machine as an RDP cookie; details about these sources are in Table 11.



RDP - Cookie

Cookie	Count
hello	9,512
Administr	876
[redacted]	867
Test	13
Domain	7
administr	5
eltons	4
a	2
\\User	2
zbom67u1	1

Figure 23. Experiment 2 – Top Ten RDP Cookies Used in Client X.224 Connections

Table 10. Experiment 2 – Top Ten Sources of RDP Cookie “hello”

Count	Source IP Address	Source Country
2241	20.41.118.149	South Korea
2006	107.181.160.93	United States
1826	185.170.144.3	Netherlands
1341	87.251.75.145	Netherlands
834	20.205.154.29	Singapore
446	194.44.226.212	Ukraine
412	154.208.140.105	United States
95	167.114.178.136	Brazil
75	147.182.177.177	United States
64	45.142.182.110	Netherlands

From these top sources of “hello,” the source 185.170.144.3 was observed creating four other RDP connections using the cookie “Administr.” Also, the source 87.251.75.145 created six RDP connections using the cookie “Administr” and two connections using the cookie “Test.”

Table 11. Experiment 2 – Only Sources Using Our Windows Machine Name as an RDP Cookie

Count	Source IP Address	Source Country
853	87.251.67.65	Netherlands
14	45.143.201.62	Ukraine

The 87.251.67.65 source was responsible for nearly all RDP cookies that used the name of our Windows machine as a value. This same source was also responsible for 854 of the total 876 RDP connections that used the cookie “Administr.” The 45.143.201.62 source only created two other RDP connections, one with the cookie “Administr” and one with “Domain.”

2. Characterizing RDP Attacks

Thirteen RDP connections were classified as clearly malicious with low severity; three RDP connections were observed with signatures of Rdpscan (Table 12), and 10 other RDP connections came from their clearly malicious sources, causing them to receive the same classifications. No signatures indicated Nessus, OpenVAS, or the Metasploit ms12_020_maxchannelids modules targeted our Windows machine.

Table 12. Experiment 2 – Suspected BlueKeep Scanning and Exploitation

Time (PST)	RDP Cookie	Source IP Address	Source Country	Source Local Time	Source Connections to Port 3389
07/14/2022 02:16	g6k3iob0	216.218.206.68	United States	02:16	7
07/14/2022 04:26	ybsngyj0	159.89.150.106	United States	04:26	1
07/14/2022 17:03	zbom67u1	89.248.172.16	Netherlands	02:03	5

We classified eight RDP connections as very suspicious; seven as low severity and one as moderate severity. Three RDP connections exchanged more than 500 packets with our Windows RDP server (Table 13). From the Windows event logs, we confirmed that these connections had established remote-desktop sessions. The duration for two of these RDP connections was less than one minute, and no malicious events were observed during their remote-desktop sessions, so they were classified with low severity. The sources of these two low-severity RDP connections had created five other RDP connections which we also classified as very suspicious with low severity, bringing the total number of low-severity connections to seven. One RDP connection lasted for two minutes, though no malicious Windows events were logged while its remote-desktop session was active. This connection was classified with moderate severity. The source of this connection had not created other RDP connections, so no other RDP connections were given the same classification.

Table 13. Experiment 2 – RDP Connections with Established Remote-Desktop Sessions with No Interaction

Time (PST)	Duration of RDP Connection (minutes)	Source IP Address	Source Country	Source Local Time	Severity of Connection
07/14/2022 04:22	< 1	170.187.194.37	Canada	04:22	Low
07/14/2022 11:47	2	144.202.14.181	United States	11:47	Moderate
07/14/2022 14:20	< 1	5.120.91.52	Iran	01:50	Low

Two RDP connections were classified as suspicious with very low severity. One RDP connection from 45.132.226.221 was observed exchanging 476 packets with the RDP service of our honeypot. Decryption of its RDP packets showed that they did not attempt a BlueKeep scan or exploit against our Windows machine. We did not observe malicious behavior from this connection, so we characterized it as suspicious with very low severity. The same source also created one other RDP connection to the honeypot that appeared to be a port scan, so we also characterized it as suspicious with very low severity.

The remaining 13653 connections were characterized as benign with very low severity since they were not observed sending malicious network traffic and they were not associated with sources that did. Table 14 shows the results of the attack-characterization methodology in for this experiment.

Table 14. Experiment 2 – Results of RDP Attack-Characterization Methodology

Set of RDP Network Traffic	Traffic Classification	Severity	Count
Malicious	Clearly Malicious	Low	13
Malicious	Very Suspicious	Very High	0
		High	0
		Moderate	1
		Low	7
Malicious	Suspicious	Low	0
		Very Low	2
Benign	Benign	Very Low	13653

3. Evaluating Characterization Methods

The only connections in this experiment that were characterized as clearly malicious were suspected of attempting BlueKeep scans against our honeypot. To evaluate the method for characterizing BlueKeep scans and attacks without PyRDP logs, we decrypted the RDP application packets sent during these connections for signatures of BlueKeep. All connections that had used a randomly generated RDP cookie were observed trying to create the MS_T120 virtual channel, which indicated BlueKeep scanning or exploitation. This confirmed that random cookies are a good clue to BlueKeep.

For the two very suspicious RDP connections that were given low severity characterizations for lasting less than one minute, we looked for Windows events that would indicate malicious behavior such as DNS queries, file-system changes, registry changes, and PowerShell events, but did not find any of these.

Malicious behavior was not observed in the two RDP connections characterized as suspicious from 45.132.226.221. The RDP connection that was 476 packets long had not tried to exploit BlueKeep and had not completed the whole connection sequence. The other

RDP connection (from the same source) was 20 packets long and occurred one second before the 476-packet connection; it had only completed the first phase of the RDP connection sequence before ending the connection. This was likely a port scan to determine if the RDP service was running before performing a deeper probe. We surmise that a very low severity was likely the best characterization for these suspicious connections.

For evaluation of benign characterizations, we confirmed that all remaining connections were each under 30 packets long, which suggests port scanners and web crawlers that completed no more than the first phase of the RDP connection sequence.

C. EXPERIMENT 3 RESULTS

1. Observed RDP Network Traffic

Experiment 3 used a new public IP address and ran from July 15, 2022 to August 9, 2022. During this experiment, our honeypot received 2,059,623 RDP connections. The first connection was received ten minutes after the DigitalOcean firewall was adjusted to allow network traffic to port 3389 of the user-interface machine from external sources. RDP connections came from 73 countries, with Russia accounting for over half (Figure 24).

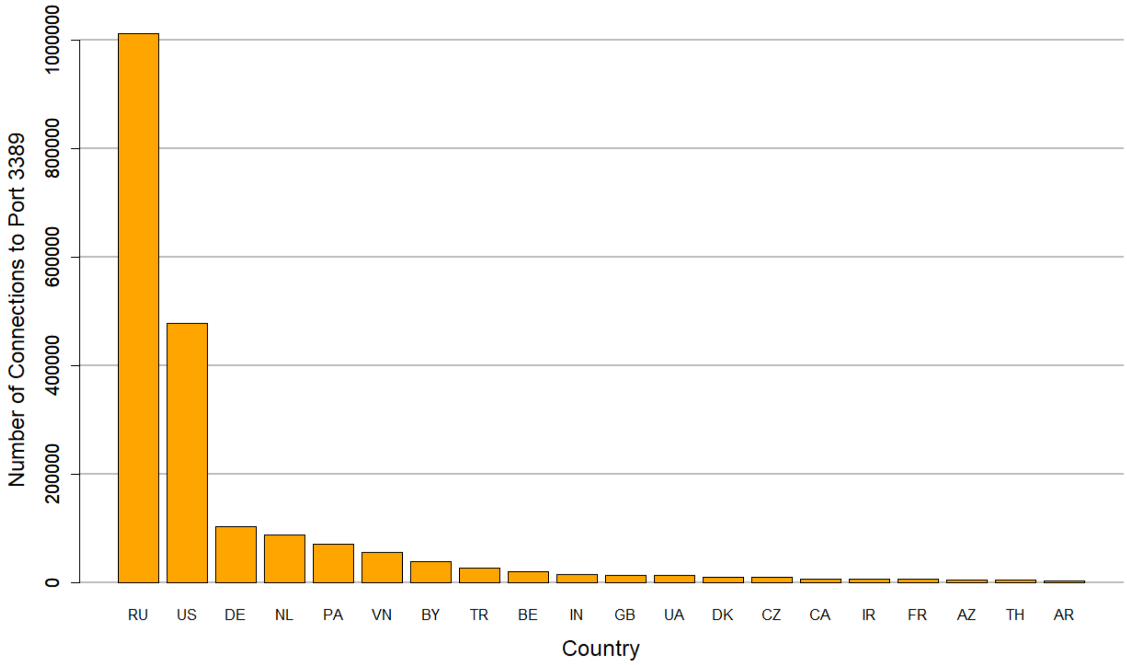
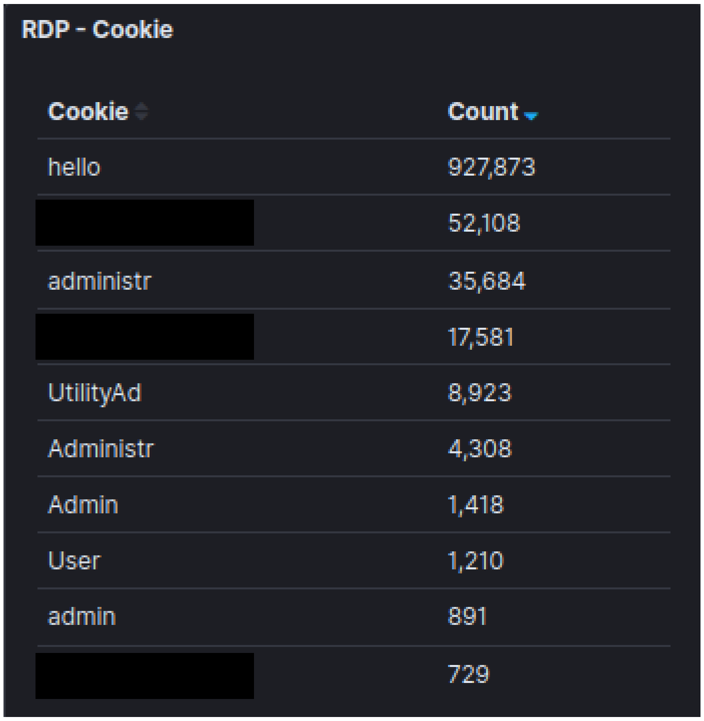


Figure 24. Experiment 3 – RDP Connections by Country

Analysis of the top RDP cookies yielded results like Experiments 1 and 2; many RDP connections to our Windows machine used the cookie “hello,” and the “administr” and “Administr” cookies were also often used to probe our remote-desktop service (Figure 25). The redacted cookies in Figure 25 are either the public IP address or name of our Windows machine. Table 15 shows the top ten sources of the cookie “hello.”



RDP - Cookie

Cookie	Count
hello	927,873
[redacted]	52,108
administr	35,684
[redacted]	17,581
UtilityAd	8,923
Administr	4,308
Admin	1,418
User	1,210
admin	891
[redacted]	729

Figure 25. Experiment 3 – Top Ten RDP Cookies Used in Client X.224 Connections

Table 15. Experiment 3 – Top Ten Sources of RDP Cookie “hello”

Count	Source IP Address	Source Country
232120	143.198.183.235	United States
71043	144.76.104.69	Germany
38027	185.99.135.135	Netherlands
35239	185.170.144.75	Netherlands
31626	137.184.95.78	United States
26943	47.242.107.32	United States
26306	27.72.180.139	Vietnam
22177	173.82.145.50	United States
21817	185.190.24.83	Aland Islands
21582	185.190.24.86	Aland Islands

Of the sources using “hello” as an RDP cookie, only two created other RDP connections that used different cookies. The 185.170.144.75 source created one other RDP connection that used the cookie “USER.” The 185.170.144.75 source created 52 connections that used the cookie “administr,” 52 connections that used the name of our

Windows machine in all capital letters as cookie values, and two connections which used the cookie “domain.”

In Figure 25, the redacted cookie with the count 52108 was the name of our Windows machine in all capital letters. The redacted cookie with a count of 17581 was the IP address of our user interface machine. The redacted cookie with the count 729 was the name of our Windows machine using capital and lowercase letters. The top-ten sources who used the capitalized name of our Windows machine are in Table 16. The top-ten sources who used the IP address are in Table 17.

Table 16. Experiment 3 – Top Ten Sources Using Our Capitalized Windows Machine Name as an RDP Cookie

Count	**Source IP Address**	**Source Country**
16556	94.232.44.88	Netherlands
8010	45.141.84.86	Russia
3203	91.240.242.8	Greece
3146	92.255.85.174	Russia
2390	179.60.149.111	Nicaragua
2250	91.240.242.5	Greece
2199	91.240.242.3	Greece
2098	92.255.85.168	Russia
1782	45.227.255.99	Panama
1760	80.66.76.145	Netherlands

Table 17. Experiment 3 – Top Ten Sources Using Our User-Interface Public IP Address as an RDP Cookie

Count	Source IP Address	Source Country
417	173.201.17.86	United States
393	162.244.33.99	United States
360	65.108.204.95	Finland
266	72.167.37.199	United States
252	139.99.135.225	Canada
248	12.246.210.62	United States
243	184.105.5.195	United States
237	216.206.190.102	United States
235	161.97.130.165	Germany
233	162.214.205.148	United States

2. Characterizing RDP Attacks

For clearly malicious RDP traffic, 71 RDP connections were classified as "clearly malicious" with low severity; we observed 11 RDP connections with signatures of Rdpscan (Table 18), and their sources created 60 other RDP connections. No signatures indicated malicious Nessus or OpenVAS scanning of our user-interface machine. We also did not see the payload signature of the ms12_020_maxchannelids Metasploit module.

Table 18. Experiment 3 – Suspected BlueKeep Scanning and Exploitation

Time (PST)	RDP Cookie	Source IP Address	Source Country	Source Local Time	Source Connections to Port 3389
07/19/2022 08:26	2chkcpjg	184.105.247.195	United States	08:26	3
07/20/2022 02:04	gc1liudi	138.197.198.158	United States	02:04	1
07/22/2022 04:04	vru7ygym	185.189.167.30	Romania	14:04	7
07/22/2022 11:40	avu6cxfo	71.6.167.124	United States	11:40	24
07/28/2022 14:48	m4avp7s4	5.188.86.98	Ireland	22:48	1
08/03/2022 17:23	iq54yhxl	74.82.47.3	United States	17:23	3
08/05/2022 05:34	rx0c2zyq	216.218.206.66	United States	05:34	3
08/05/2022 21:28	idhmabns	159.223.172.125	United States	21:28	1
08/06/2022 15:48	g6gbirkv	80.82.77.139	Netherlands	23:48	24
08/06/2022 20:27	i63o5c1v	184.105.139.69	United States	20:27	3
08/07/2022 07:26	6cdd755w	128.14.209.162	United States	07:26	1

We classified 63092 RDP connections as very suspicious; 102 RDP connections were observed exchanging more than 500 packets with the Windows RDP service, and their malicious sources created 62990 other RDP connections. The 102 RDP connections with more than 500 packets indicated attacks that established remote-desktop sessions. The first RDP connection to establish such a session occurred on July 17, though the duration of its RDP connection was less than 15 seconds. The first RDP connection longer than one minute occurred on July 20 at 7:31 AM PST. Due to the many attacks that exchanged more than 500 packets, we analyzed events only for the 28 RDP connections that were longer that one minute (Table 19). Since we did not analyze events for the 74 RDP connections less than one minute, we characterized them as very suspicious with low severity by default. The sources that created these 102 RDP connections created 62990 other RDP connections to our user-interface machine throughout the experiment, most of which appeared to be RDP service probes less than 30 packets long. These 62990 RDP connections were classified as very suspicious, and each was given the same severity classification as its source's highest severity RDP connection. Table 20 lists the severity classifications and counts for these RDP connections that had not exchanged more than 500 packets but were classified as very suspicious for coming from sources that did.

Table 19. Experiment 3 – RDP Connections with Established Remote-Desktop Sessions

Time (PST)	**Duration of RDP Conn. (minutes)**	**Source IP Address**	**Source Country**	**Interaction if any**	**Severity of Connection**
07/20/ 2022 07:31	7	5.121.216.47	Iran	Internet, PowerShell	Very High
07/21/ 2022 12:00	< 2	77.83.36.6	Ukraine	PowerShell	High
07/22/ 2022 05:16	9	146.0.40.37	Germany	Registry, File System	Very High
07/24/ 2022 05:41	4	193.201.9.156	United States	PowerShell	High
07/25/ 2022 01:48	< 2	45.227.255.59	Panama		Moderate
07/29/ 2022 06:42	< 2	77.83.36.32	Ukraine	PowerShell	High
07/30/ 2022 13:32	4	167.235.77.239	United States	PowerShell	High
08/01/ 2022 08:39	< 2	179.60.147.37	Russia		Moderate
08/02/ 2022 01:43	19	82.180.207.226	Denmark	PowerShell, SCADA	Very High

Time (PST)	Duration of RDP Conn. (minutes)	Source IP Address	Source Country	Interaction if any	Severity of Connection
08/02/ 2022 11:04	< 2	219.93.182.50	Malaysia		Moderate
08/02/ 2022 17:09	< 2	86.120.179.136	Romania		Moderate
08/02/ 2022 23:40	7	172.104.175.78	Singapore	PowerShell	Very High
08/03/ 2022 00:49	30	172.104.175.78	Singapore	SCADA	Very High
08/03/ 2022 03:43	26	159.223.80.155	United States	PowerShell, File System	Very High
08/03/ 2022 06:23	< 2	175.157.184.120	Sri Lanka		Moderate
08/03/ 2022 08:45	< 2	159.223.80.155	United States		Moderate
08/04/ 2022 03:39	6	175.157.184.120	Sri Lanka		High
08/04/ 2022 03:51	9	175.157.184.120	Sri Lanka	Internet	High
08/04/ 2022 07:47	< 2	175.157.184.120	Sri Lanka		Moderate
08/04/ 2022 09:37	3	159.223.80.155	United States		Moderate
08/05/ 2022 18:16	25	102.89.38.221	Nigeria	File System	Very High
08/07/ 2022 08:29	29	185.54.228.7	Australia	Internet, File System	Very High

Time (PST)	Duration of RDP Conn. (minutes)	Source IP Address	Source Country	Interaction if any	Severity of Connection
08/07/ 2022 14:19	< 2	185.170.144.75	Netherlands		Moderate
08/07/ 2022 14:21	6	185.99.135.134	Belarus		High
08/07/ 2022 14:32	5	5.180.209.127	Germany	File System	Very High
08/07/ 2022 14:37	22	185.229.59.109	United States	File System	Very High
08/07/ 2022 17:36	< 2	185.54.228.45	Australia		Moderate
08/08/ 2022 11:31	7	185.54.228.148	Australia	File System, Internet	Very High

Table 20. Experiment 3 – Severity Classifications for Very Suspicious RDP Connections with Exchanges Less Than 500 Packets

Severity Classification	Number of RDP Connections
Very High	100
High	17
Moderate	37872
Low	25001

Classifications for suspicious RDP traffic amounted to 4878 RDP connections; six RDP connections exchanged between 85 and 500 packets with our user-interface machine (Table 21), and their sources created 4872 other RDP connections. No signatures of BlueKeep occurred in the decrypted packets, so all RDP traffic from these sources was characterized as suspicious with very low severity.

Table 21. Experiment 3 – RDP Connections Classified as Suspicious

Time (PST)	**Source IP Address**	**Source Country**	**Source Local Time**	**Source Connections to port 3389**
07/16/2022 10:39	128.90.158.193	United States	10:39	16
07/19/2022 10:26	94.232.41.214	Netherlands	19:26	4848
07/21/2022 14:04	185.189.167.30	Russia	00:04	2
07/22/2022 21:46	47.90.214.198	United States	21:46	8
07/29/2022 11:09	193.37.69.211	Netherlands	20:09	4
07/29/2022 11:35	193.37.69.211	Netherlands	20:35	4

The remaining 1991582 connections were characterized as benign with very low severity for not sending malicious traffic and being unrelated to sources that did. This experiment is summarized in Table 22.

Table 22. Experiment 3 – Results of RDP Attack-Characterization Methodology

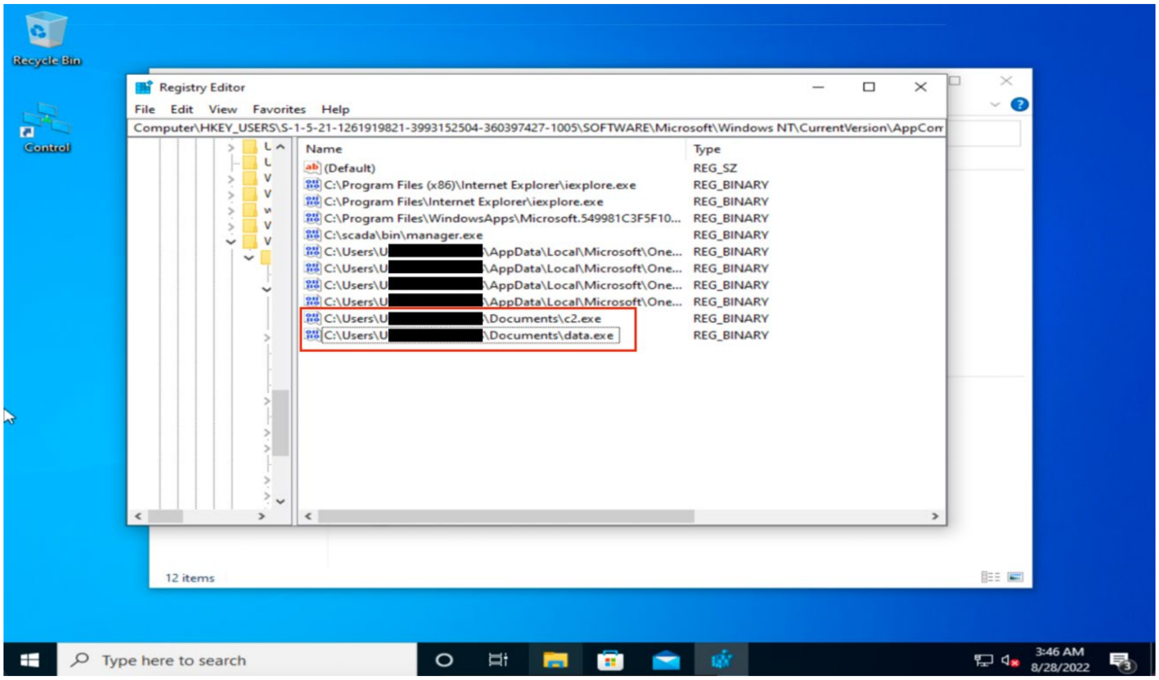
Set of RDP Network Traffic	Traffic Classification	Severity	Count
Malicious	Clearly Malicious	Low	71
Malicious	Very Suspicious	Very High	111
		High	24
		Moderate	37882
		Low	25075
Malicious	Suspicious	Low	0
		Very Low	4878
Benign	Benign	Very Low	1991582

3. Evaluating Characterizations Methods

To evaluate the RDP connections characterized as "clearly malicious" using randomly generated cookies, we decrypted their network traffic to search for the MS_T120 signature of BlueKeep. Each of the 11 connections using a randomly generated RDP cookie tried to scan or exploit BlueKeep.

For "very suspicious" network-traffic characterizations, we compared the events observed through Windows event logs against artifacts of attack left behind on our Windows virtual machine. This included checking PowerShell transcripts, new files in the file system, and new keys in the registry. Not every remote-desktop session left behind artifacts of attack, but we could confirm the severity of attacks for sessions that did. Because many remote-desktop sessions were established by attackers, we limited our evaluation to those which left behind interesting and visible artifacts of attack.

From the attack on July 22, 2022, from a German source, two files "data.exe" and "c2.exe" were created in the Documents folder of the publicly accessible user. Metadata for each file could be seen in the Sysmon event logs. For the "data.exe" file, the original file name was reported as "VBCECompiler," and for "c2.exe," the original filename was reported as “DedicStore.SystemInfoChecker.exe.” The MD5 hashes for each file were checked on VirusTotal (VirusTotal, n.d.), which confirmed their original names and reported that several security vendors had flagged the files as malicious. Artifacts for each malware file could be seen in the Windows registry. Figure 26 shows the changes that were made to the registry. The filename “c2.exe” resembled the filename “c.exe,” which was malware observed in Meier’s Experiment 1 (Meier, 2022). While the malware observed in Meier’s experiment had downloaded and used Firefox to visit travel websites suspected of hosting adware, “c2.exe” in our experiment had only attempted to access “www.codeproject.com,” which hosts forums and articles related to programming. We did not perform a deep investigation into the behavior of the malware, so further research into this malware is needed to confirm the behavior of these files.



Computer\HKEY_USERS\S-1-5-21-1261919821-3993152504-360397427-1005\SOFTWARE\Microsoft\Windows NT\CurrentVersion\AppCom

Name	Type
(Default)	REG_SZ
C:\Program Files (x86)\Internet Explorer\iexplore.exe	REG_BINARY
C:\Program Files\Internet Explorer\iexplore.exe	REG_BINARY
C:\Program Files\WindowsApps\Microsoft.549981C3F5F10...	REG_BINARY
C:\scada\bin\manager.exe	REG_BINARY
C:\Users\U[redacted]\AppData\Local\Microsoft\One...	REG_BINARY
C:\Users\U[redacted]\AppData\Local\Microsoft\One...	REG_BINARY
C:\Users\U[redacted]\AppData\Local\Microsoft\One...	REG_BINARY
C:\Users\U[redacted]\AppData\Local\Microsoft\One...	REG_BINARY
C:\Users\U[redacted]\Documents\c2.exe	REG_BINARY
C:\Users\U[redacted]\Documents\data.exe	REG_BINARY

Figure 26. Changes to the Windows Registry for the “data.exe” and “c2.exe” Files

The attack from a Nigerian source on August 5, 2022, changed the file system by adding the “laravel_scanner.exe,” “SenderSMS” files and directories, and several DLL files such as “_pytransform.dll,” “python27.dll” and “msvcm90.dll” which were likely needed for the scanner executable. From event logs, the process created from the "laravel_scanner.exe" image made many DNS requests to random web servers, including "tactyl-services.com," "cdlima.org.pe," "epu.edu.pe," "handle-wakemag.fr," and "chuckneedham.com." In investigating these domains, we could not connect to "tactyl-services.com," "cdlima.org.pe" appeared to belong to a Peruvian engineering college, "epu.edu.pe" redirected us to the University of San Martin de Porres in Peru, "handle-wakemag.fr" was the site for a French water sports magazine, and "chuckneedham.com" was a relatively simple WordPress blog. On the Windows machine, we could see that artifacts from this attack were visible on the desktop (Figure 27).

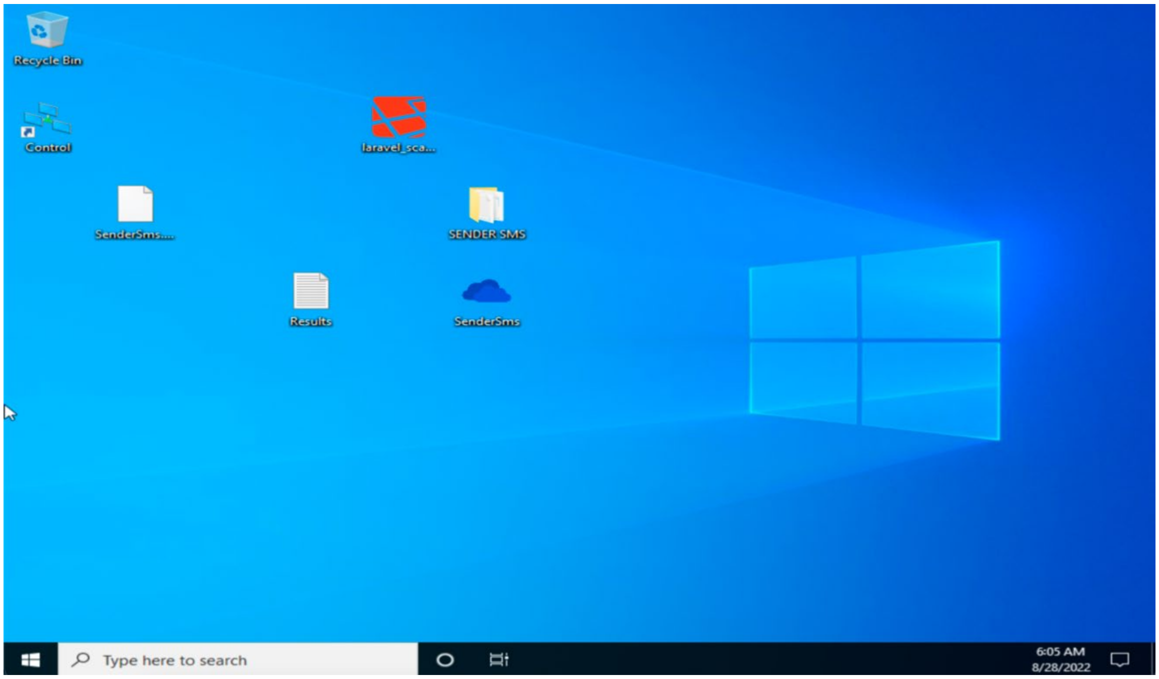
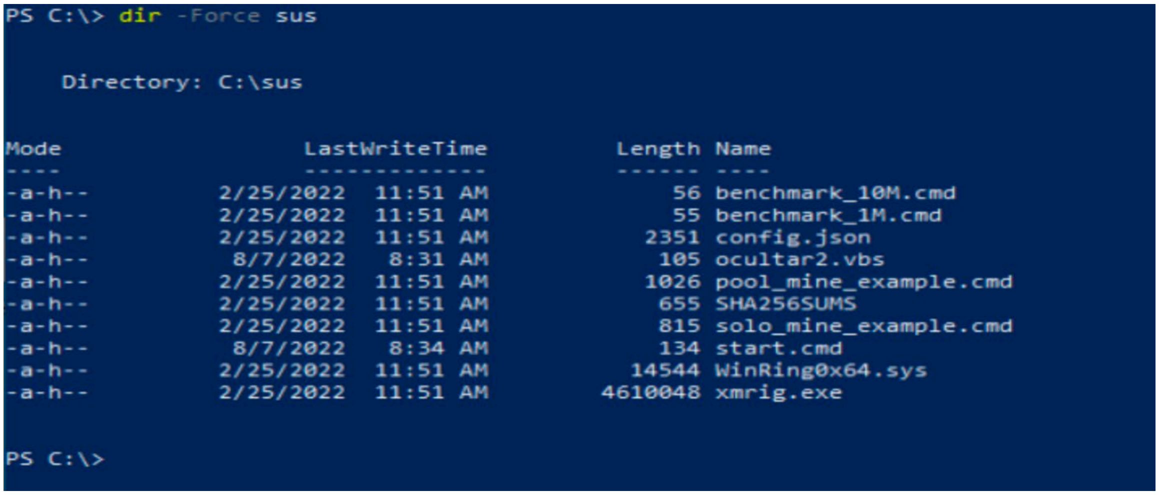


Figure 27. Artifacts of Attack from the Remote-Desktop Attack that Occurred on August 5, 2022

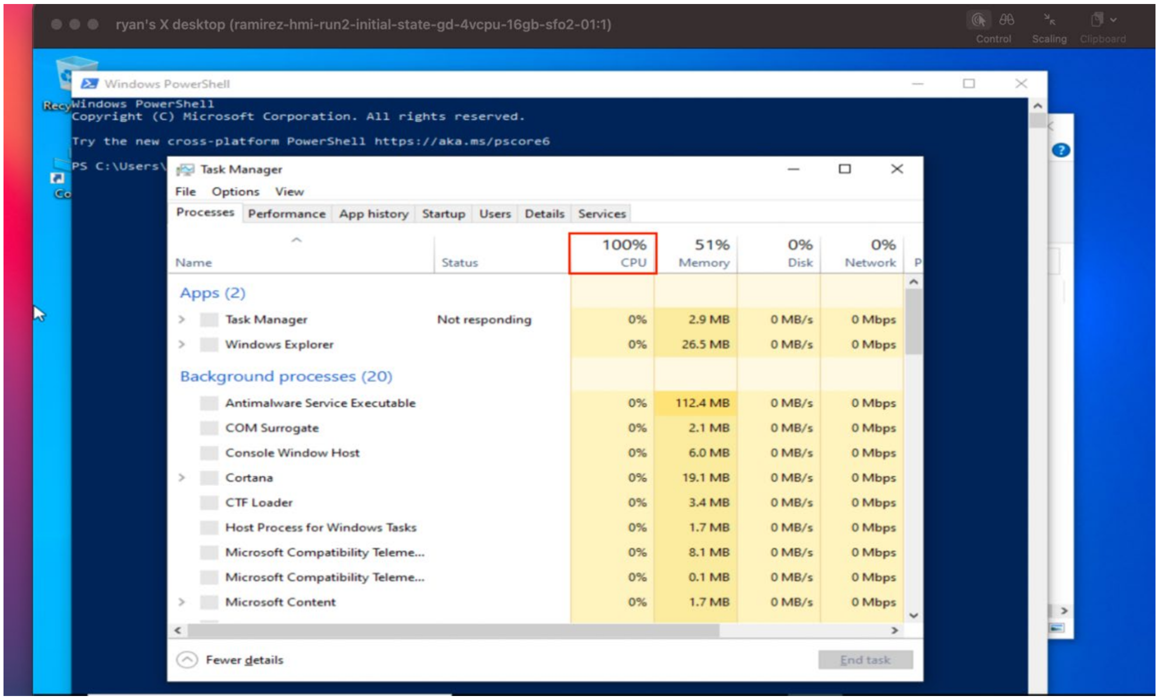
An analysis of a virtual machine snapshot taken three days after this attack revealed more information about the files downloaded by the attacker. The 254-kilobyte "Results.txt" file on the Desktop contained a list of HTTP Web servers. Inside of the "SenderSMS" directory, two files called "SHELL.txt" and "RDP.txt" contained what appeared to be login credentials and an IP address, respectively. Also, in "SenderSMS" was a folder which we had not observed in event logs called “CHASE 2022.zip,” which contained a folder called “SpoxV5” with PHP scripts and other Web related documents like “robots.txt,” and “visit_log.txt.”

On the morning of August 8, 2022, we noticed a spike in processor resource use from the administrative interface of DigitalOcean. This related to an attack from Australia which installed cryptocurrency mining files on our Windows machine. The cryptocurrency miner was “xmrig.exe” which, from an internet search, was revealed to be processor-mining software (XMRig, n.d.). This same cryptocurrency mining malware was downloaded as part of an attack observed by Dougherty in his Phase 2 GridPot honeypot experiment (Dougherty, 2020). For this specific attack, the malware was hidden in the directory “C:\sus\” (Figure 28). Task Manager also indicated that all the system’s processor resources were being used (Figure 29). High processor use was also observed by Dougherty for an attack that had installed XMrig on his honeypot.



```
PS C:\> dir -Force sus


    Directory: C:\sus


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a-h--         2/25/2022  11:51 AM             56 benchmark_10M.cmd
-a-h--         2/25/2022  11:51 AM             55 benchmark_1M.cmd
-a-h--         2/25/2022  11:51 AM           2351 config.json
-a-h--          8/7/2022   8:31 AM            105 ocultar2.vbs
-a-h--         2/25/2022  11:51 AM           1026 pool_mine_example.cmd
-a-h--         2/25/2022  11:51 AM            655 SHA256SUMS
-a-h--         2/25/2022  11:51 AM            815 solo_mine_example.cmd
-a-h--          8/7/2022   8:34 AM            134 start.cmd
-a-h--         2/25/2022  11:51 AM          14544 WinRing0x64.sys
-a-h--         2/25/2022  11:51 AM        4610048 xmrig.exe


PS C:\>
```

Figure 28. XMrig.exe and Related Malware in the “C:\sus” Directory


ryan's X desktop (ramirez-hmi-run2-initial-state-gd-4vcpu-16gb-sfo2-01:1)
Control
Scaling
Clipboard
Windows PowerShell
Task Manager
File Options View
Processes Performance App history Startup Users Details Services
Fewer details
End task

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\
```

Name	Status	100% CPU	51% Memory	0% Disk	0% Network
Apps (2)					
Task Manager	Not responding	0%	2.9 MB	0 MB/s	0 Mbps
Windows Explorer		0%	26.5 MB	0 MB/s	0 Mbps
Background processes (20)					
Antimalware Service Executable		0%	112.4 MB	0 MB/s	0 Mbps
COM Surrogate		0%	2.1 MB	0 MB/s	0 Mbps
Console Window Host		0%	6.0 MB	0 MB/s	0 Mbps
Cortana		0%	19.1 MB	0 MB/s	0 Mbps
CTF Loader		0%	3.4 MB	0 MB/s	0 Mbps
Host Process for Windows Tasks		0%	1.7 MB	0 MB/s	0 Mbps
Microsoft Compatibility Teleme...		0%	8.1 MB	0 MB/s	0 Mbps
Microsoft Compatibility Teleme...		0%	0.1 MB	0 MB/s	0 Mbps
Microsoft Content		0%	1.7 MB	0 MB/s	0 Mbps

Figure 29. High CPU Resource Usage of Our Windows Machine Likely Due to Cryptocurrency Mining Software Using All Available Resources

On August 8, 2022, we restored the Windows machine to its original clean state before it was attacked. Within ten minutes, another attack from Australia, likely from the same subnet as the first, installed the same cryptocurrency-mining software from the previous day's attack. This time, the attacker had placed the malware into the directory "C:\Qt\sus."

Our methods for characterizing the severity of established remote-desktop sessions through Windows event logs helped us find the artifacts left behind by attackers when they connected to our honeypot. Because of these artifacts, we classified these sessions as "very high severity" due to the changes made to the Windows machine.

To evaluate attacks classified as "suspicious," we decrypted their RDP packets in Wireshark to confirm that the attack had not attempted a BlueKeep scan or exploit by creating the MS_T120 virtual channel. We did not observe any BlueKeep-related signatures, and no other malicious behavior was found from event logs and analysis of the other RDP probes from these sources. Since only five connections were classified as suspicious, we assigned these a very low severity level.

Of the RDP connections that were characterized as benign, 99.1% exchanged less than 30 packets with the user-interface machine, and 99.9% exchanged less than 40 packets. The small numbers of packets exchanged during these RDP connections suggest web crawlers and port scanners checking whether our Windows host was alive and hosting an RDP service.

D. EXPERIMENT 4 RESULTS

1. Observed RDP Traffic

We restored our Windows machine to a clean state and ran Experiment 4 from August 9, 2022 to August 11, 2022. This experiment used PyRDP MitM in the same configuration as Experiment 1 with the same public IP address as Experiment 3. During Experiment 4, our honeypot received only 6809 RDP connections from 35 countries during the two days that it was publicly accessible. Most connections were from the United States, Canada, and Germany (Figure 30).

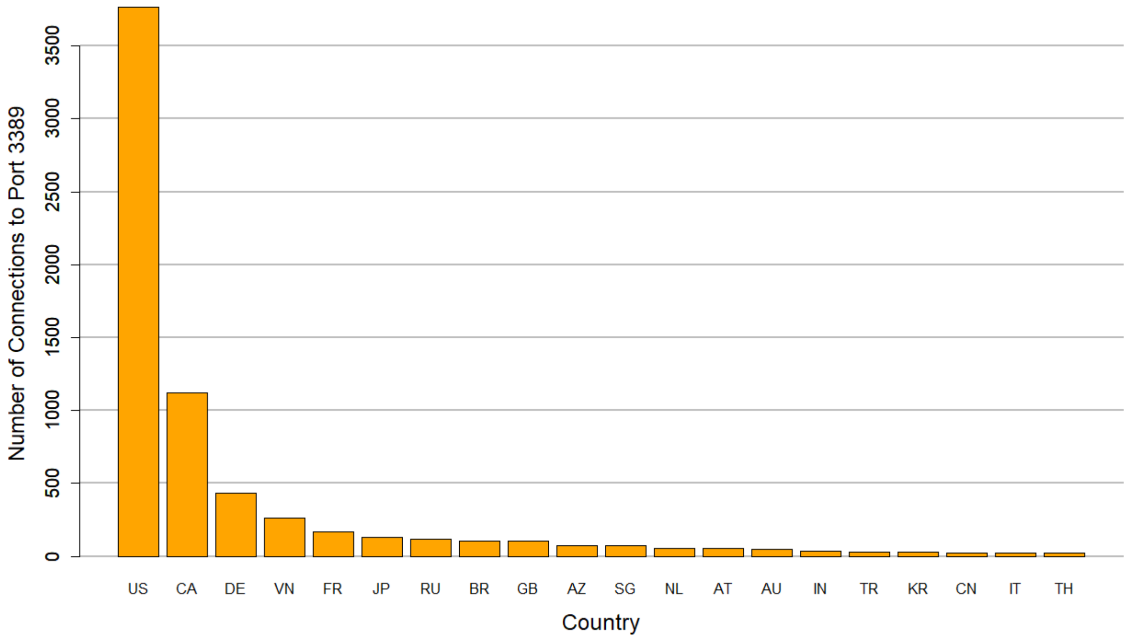
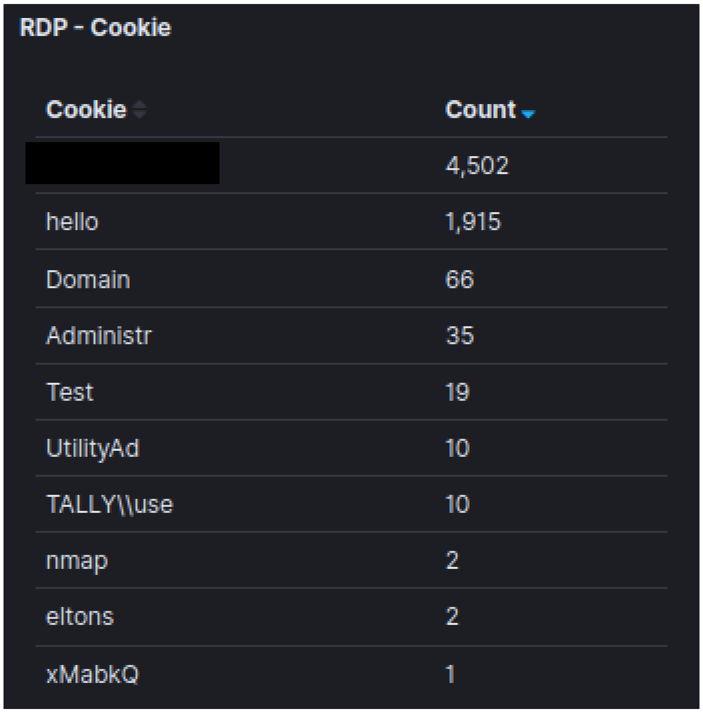


Figure 30. Experiment 4 – RDP Connections by Country

Unlike previous experiments, the most common RDP cookie was the public IP address of our user-interface machine, and "hello" was the second most common (Figure 31). The top ten sources who used an RDP cookie that matched our public IP address are listed in Table 23. The top ten sources who used the RDP cookie "hello" are in Table 24.



RDP - Cookie

Cookie	Count
[redacted]	4,502
hello	1,915
Domain	66
Administr	35
Test	19
UtilityAd	10
TALLY\\use	10
nmap	2
eltons	2
xMabkQ	1

Figure 31. Experiment 4 – Top Ten RDP Cookies Used in Client X.224 Connections

Table 23. Experiment 4 – Top Ten Sources Using Our User-Interface Public IP Address as an RDP Cookie

Count	Source IP Address	Source Country
141	162.244.33.99	United States
113	184.105.5.195	United States
109	174.136.15.208	United States
89	65.108.204.95	Finland
89	173.201.17.86	United States
84	208.109.214.246	Netherlands
81	13.90.243.247	United States
81	23.95.34.109	United States
61	162.214.205.148	United States
57	161.97.130.165	Germany

Table 24. Experiment 4 – Top Ten Sources of RDP Cookie "hello"

Count	Source IP Address	Source Country
850	158.69.59.110	Canada
354	54.226.141.243	United States
255	103.9.158.114	Vietnam
170	185.158.151.211	Turkey
73	81.21.86.170	Azerbaijan
25	186.208.139.104	Brazil
23	45.179.176.138	Brazil
19	40.115.49.158	Netherlands
16	195.133.235.226	Russia
14	91.240.118.113	Russia

None of the top sources who used our IP address as a cookie were observed creating RDP connections that used a different cookie value. Of the top sources who sent "hello" as a cookie, only the source 91.240.118.113 was observed creating an RDP connection that used a cookie besides it, a cookie "Domain."

2. Characterizing RDP Attacks

No indicators suggested that attackers had used Nessus, OpenVAS, or Metasploit ms12_020_maxchannelids to maliciously attack our honeypot. We also did not observe any randomly generated seven-to-eight-character RDP cookies that would indicate BlueKeep. No RDP connections were classified as "clearly malicious" in this experiment.

Five RDP connections were classified as very suspicious; two RDP connections were observed exchanging more than 500 packets with our RDP server (Table 25), and the three other RDP connections from their sources were classified as very suspicious by association. Of the two RDP connections that had exchanged more than 500 packets with the RDP server: one lasted two minutes and the other lasted roughly one minute. The two-minute connection was classified as high severity since Windows event logs indicated file-system changes and attempted Internet access during its remote-desktop session. No malicious events were observed for the one-minute RDP connection. The Seychelles source had previously created another RDP connection to scan the RDP service; we also classified this as "very suspicious" with low severity. The U.S. source had previously created two RDP connections to scan the RDP service; these were classified as "very suspicious" with high severity.

Table 25. Experiment 4 – RDP Connections with Established Remote-Desktop Sessions

Time (PST)	Duration of RDP Connection (minutes)	Source IP Address	Source Country	Interaction if any	Severity of Connection
08/10/2022 07:32	< 1	77.83.36.6	Seychelles		Low
08/11/2022 04:14	2	146.71.81.163	United States	File System, Internet	High

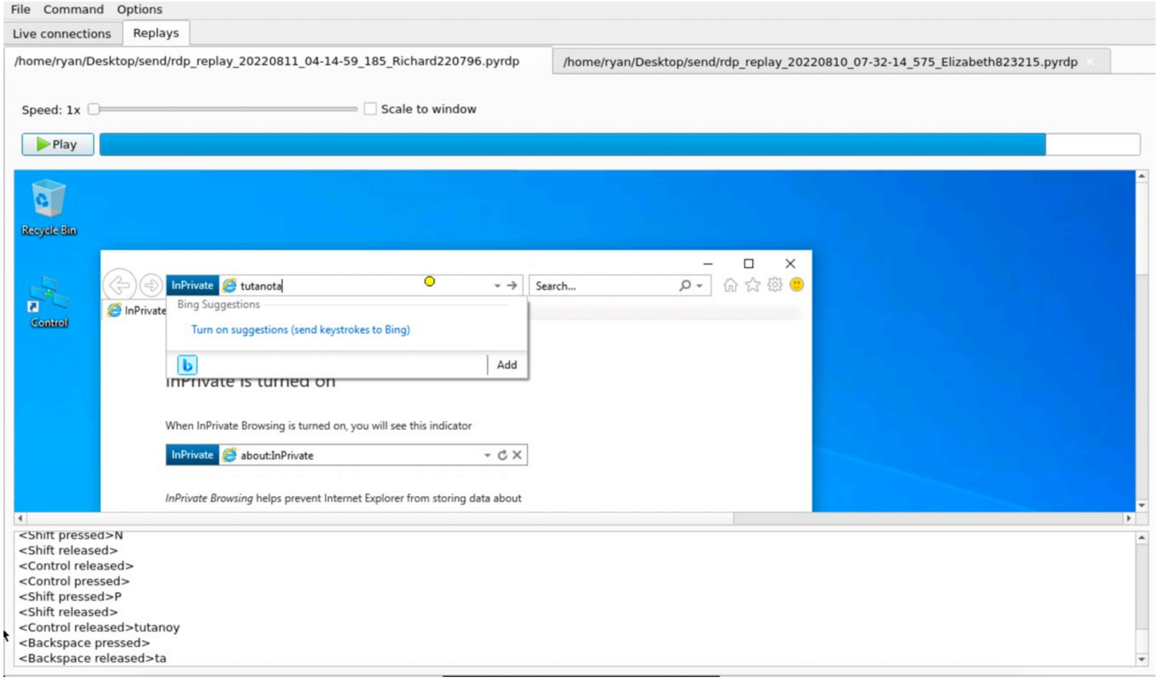
No connections had exchanged between 85 and 500 packets with the RDP service, so the remaining 6804 connections were characterized as benign. Table 26 summarizes this experiment.

Table 26. Experiment 4 – Results of Attack-Characterization Methodology

Set of RDP Network Traffic	Traffic Classification	Severity	Count
Malicious	Clearly Malicious	Low	0
Malicious	Very Suspicious	Very High	0
		High	3
		Moderate	0
		Low	2
Malicious	Suspicious	Low	0
		Very Low	0
Benign	Benign	Very Low	6804

3. Evaluating Characterization Methods

For our evaluation of very suspicious network traffic, we compared Windows event logs to the PyRDP replays of the two remote-desktop sessions observed. Because the duration of the first connection was less than a minute, we only observed the attacker interacting with the Windows login screen before disconnecting. This connection was classified as low severity. For the second connection that lasted roughly two minutes, we could see the attacker successfully log in and open the Microsoft Edge browser. This attacker tried to navigate to "www.tutanota.com," a website which advertises an encrypted email service, but was unsuccessful and disconnected shortly after. This attacker's interaction with the Windows machine is shown in Figure 32. We classified this attack as high severity.


File Command Options
Live connections
Replays
/home/ryan/Desktop/send/rdp_replay_20220811_04-14-59_185_Richard220796.pyrdp
/home/ryan/Desktop/send/rdp_replay_20220810_07-32-14_575_Elizabeth823215.pyrdp
Speed: 1x
Scale to window
Play
Recycle Bin
Control
InPrivate
tutanota
Search...
Bing Suggestions
Turn on suggestions (send keystrokes to Bing)
Add
InPrivate is turned on
When InPrivate Browsing is turned on, you will see this indicator
about:InPrivate
InPrivate Browsing helps prevent Internet Explorer from storing data about

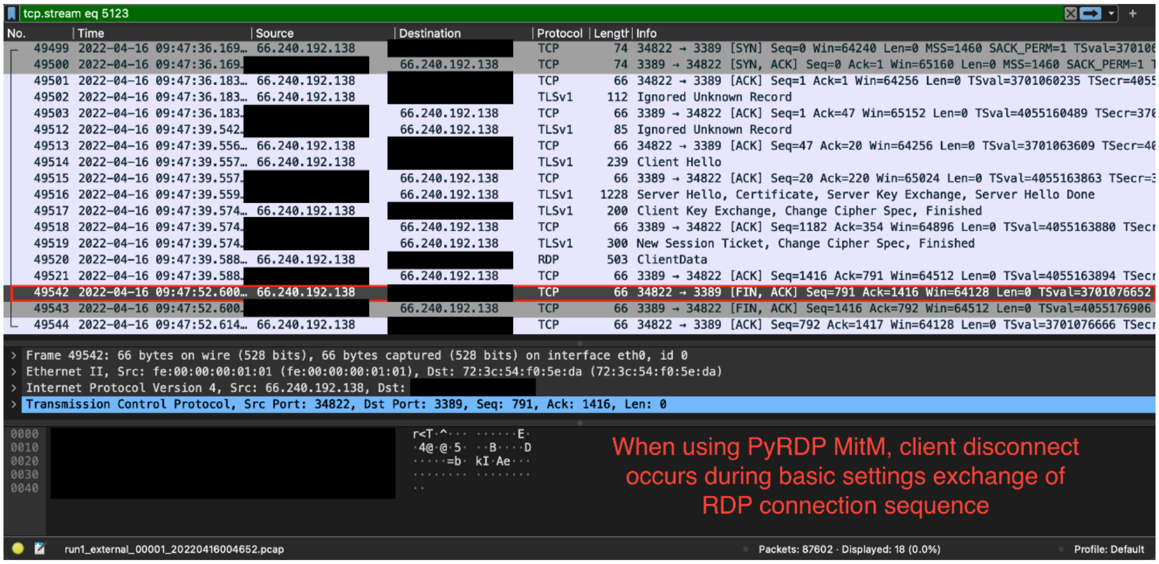
```
<Shift pressed>N
<Shift released>
<Control released>
<Control pressed>
<Shift pressed>P
<Shift released>
<Control released>tutanoy
<Backspace pressed>
<Backspace released>ta
```

Figure 32. Experiment 4 – PyRDP Analysis of an Established Remote-Desktop Session

For RDP traffic characterized as benign, we investigated further into connections that used the public IP address of the user-interface machine as their RDP cookie since those connections made up roughly 70% of the total RDP connections received. From the 100 such RDP connections that we sampled randomly, the connections did not display malicious behaviors, having completed only the first phase of the RDP connection sequence before disconnecting. We also investigated further into the eight randomly generated six-character alphabetic RDP cookies that were observed in this experiment since they did not match the pattern of Metasploit or Rdpscan generated cookies. They also only completed the first phase of the RDP connection sequence. Also, PyRDP did not generate any logs for BlueKeep-related events in this experiment. These analyses confirmed that these RDP connections are benign with very low severity.

E. DISCUSSION

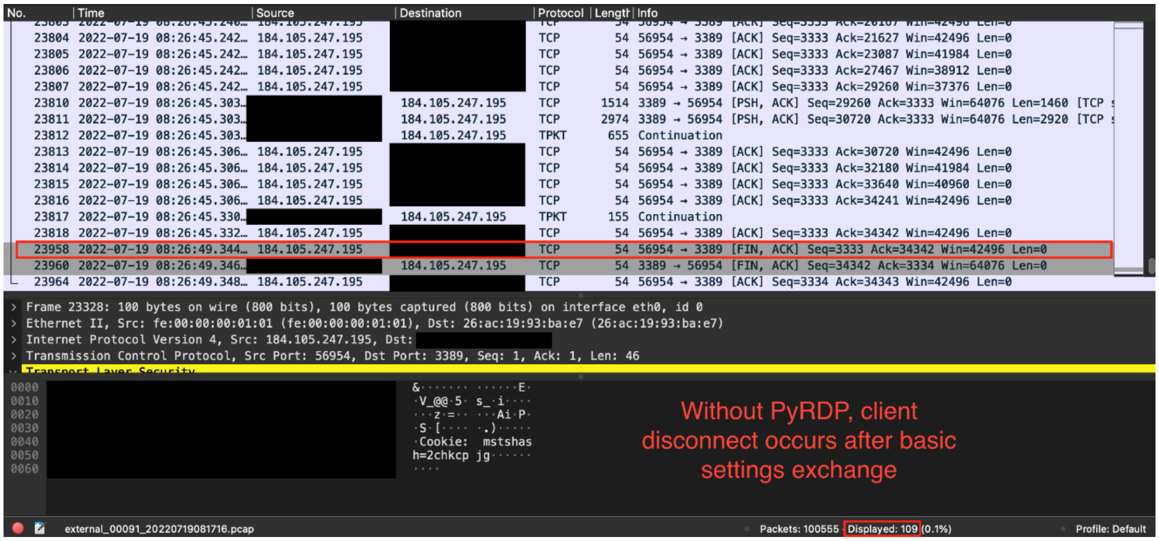
After using PyRDP MitM for interception and forwarding of attacker connections to our honeypot in Experiment 1, we saw some differences in network traffic compared to directly accessing the Windows remote-desktop service in subsequent experiments. The most noticeable difference was the number of packets exchanged for BlueKeep events. In Experiment 1, we noticed that each BlueKeep attack exchanged no more than 30 packets with our RDP service. Similar BlueKeep scanning and exploitation observed in Experiments 2 and 3 showed exchanges of roughly 100 packets. Figure 33 and Figure 34 show the first BlueKeep scans received by our honeypot in Experiment 1 and Experiment 3. From comparison of these images, we can see that client disconnects for BlueKeep scans occurred during basic settings exchange when using PyRDP, while client disconnects for the same type of scan occurred at connection finalization when not using PyRDP. This is further evidence of Meier's observations that PyRDP handles RDP network traffic differently than direct connections to the Windows RDP service (Meier, 2022).



```
tcp.stream eq 5123
No.    Time                      Source           Destination      Protocol Length Info
49499  2022-04-16 09:47:36.169…  66.240.192.138   [redacted]       TCP      74  34822 → 3389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=370106
49500  2022-04-16 09:47:36.169…  [redacted]       66.240.192.138   TCP      74  3389 → 34822 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM=1 T
49501  2022-04-16 09:47:36.183…  66.240.192.138   [redacted]       TCP      66  34822 → 3389 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3701060235 TSecr=4055
49502  2022-04-16 09:47:36.183…  66.240.192.138   [redacted]       TLSv1   112  Ignored Unknown Record
49503  2022-04-16 09:47:36.183…  [redacted]       66.240.192.138   TCP      66  3389 → 34822 [ACK] Seq=1 Ack=47 Win=65152 Len=0 TSval=4055160489 TSecr=370
49512  2022-04-16 09:47:39.542…  [redacted]       66.240.192.138   TLSv1    85  Ignored Unknown Record
49513  2022-04-16 09:47:39.556…  66.240.192.138   [redacted]       TCP      66  34822 → 3389 [ACK] Seq=47 Ack=20 Win=64256 Len=0 TSval=3701063609 TSecr=40
49514  2022-04-16 09:47:39.557…  66.240.192.138   [redacted]       TLSv1   239  Client Hello
49515  2022-04-16 09:47:39.557…  [redacted]       66.240.192.138   TCP      66  3389 → 34822 [ACK] Seq=20 Ack=220 Win=65024 Len=0 TSval=4055163863 TSecr=3
49516  2022-04-16 09:47:39.559…  [redacted]       66.240.192.138   TLSv1  1228  Server Hello, Certificate, Server Key Exchange, Server Hello Done
49517  2022-04-16 09:47:39.574…  66.240.192.138   [redacted]       TLSv1   200  Client Key Exchange, Change Cipher Spec, Finished
49518  2022-04-16 09:47:39.574…  [redacted]       66.240.192.138   TCP      66  3389 → 34822 [ACK] Seq=1182 Ack=354 Win=64896 Len=0 TSval=4055163880 TSecr
49519  2022-04-16 09:47:39.574…  [redacted]       66.240.192.138   TLSv1   300  New Session Ticket, Change Cipher Spec, Finished
49520  2022-04-16 09:47:39.588…  66.240.192.138   [redacted]       RDP     503  ClientData
49521  2022-04-16 09:47:39.588…  [redacted]       66.240.192.138   TCP      66  3389 → 34822 [ACK] Seq=1416 Ack=791 Win=64512 Len=0 TSval=4055163894 TSecr
49542  2022-04-16 09:47:52.600…  66.240.192.138   [redacted]       TCP      66  34822 → 3389 [FIN, ACK] Seq=791 Ack=1416 Win=64128 Len=0 TSval=3701076652
49543  2022-04-16 09:47:52.600…  [redacted]       66.240.192.138   TCP      66  3389 → 34822 [FIN, ACK] Seq=1416 Ack=792 Win=64512 Len=0 TSval=4055176906
49544  2022-04-16 09:47:52.614…  66.240.192.138   [redacted]       TCP      66  34822 → 3389 [ACK] Seq=792 Ack=1417 Win=64128 Len=0 TSval=3701076666 TSecr

> Frame 49542: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface eth0, id 0
> Ethernet II, Src: fe:00:00:00:01:01 (fe:00:00:00:01:01), Dst: 72:3c:54:f0:5e:da (72:3c:54:f0:5e:da)
> Internet Protocol Version 4, Src: 66.240.192.138, Dst: [redacted]
> Transmission Control Protocol, Src Port: 34822, Dst Port: 3389, Seq: 791, Ack: 1416, Len: 0

0000  [redacted]  r<T·^···  ······E·
0010              ·4@·@·5·  ··B····D
0020              ·····=b·  kI·Ae···
0030              ········  ········
0040              ··

run1_external_00001_20220416004652.pcap          Packets: 87602 · Displayed: 18 (0.0%)          Profile: Default
```

When using PyRDP MitM, client disconnect occurs during basic settings exchange of RDP connection sequence

Figure 33. Results of Rdpscan When PyRDP Was Used

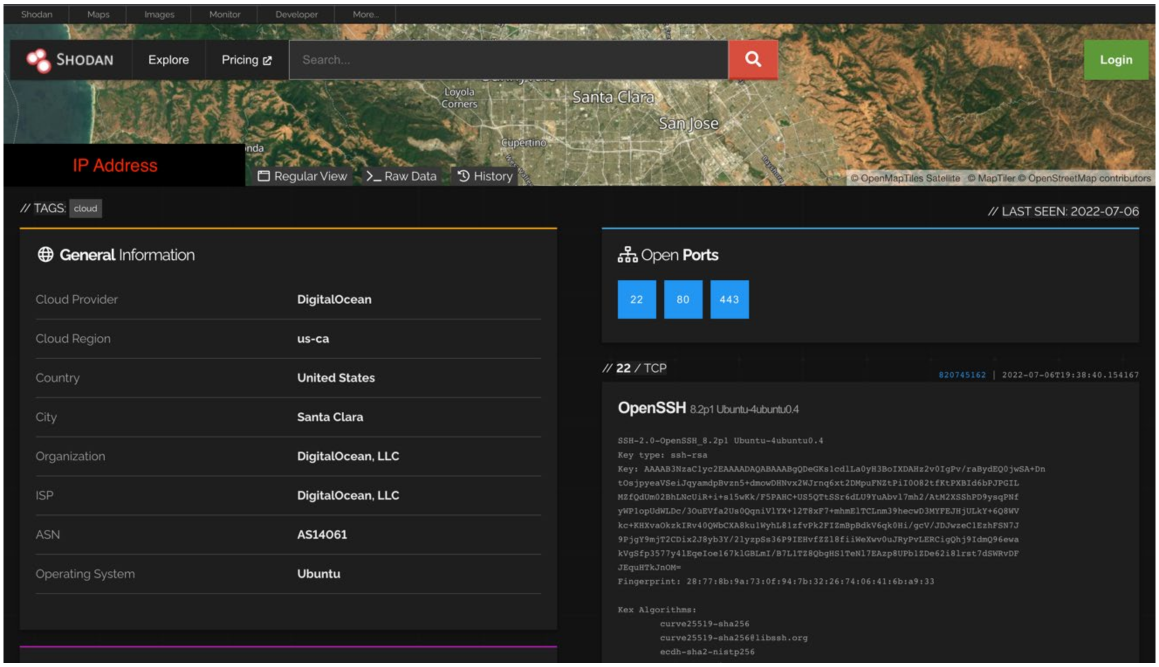


```
No.    Time                      Source           Destination      Protocol Length Info
23804  2022-07-19 08:26:45.242…  184.105.247.195  [redacted]       TCP      54  56954 → 3389 [ACK] Seq=3333 Ack=21627 Win=42496 Len=0
23805  2022-07-19 08:26:45.242…  184.105.247.195  [redacted]       TCP      54  56954 → 3389 [ACK] Seq=3333 Ack=23087 Win=41984 Len=0
23806  2022-07-19 08:26:45.242…  184.105.247.195  [redacted]       TCP      54  56954 → 3389 [ACK] Seq=3333 Ack=27467 Win=38912 Len=0
23807  2022-07-19 08:26:45.242…  184.105.247.195  [redacted]       TCP      54  56954 → 3389 [ACK] Seq=3333 Ack=29260 Win=37376 Len=0
23810  2022-07-19 08:26:45.303…  [redacted]       184.105.247.195  TCP    1514  3389 → 56954 [PSH, ACK] Seq=29260 Ack=3333 Win=64076 Len=1460 [TCP s
23811  2022-07-19 08:26:45.303…  [redacted]       184.105.247.195  TCP    2974  3389 → 56954 [PSH, ACK] Seq=30720 Ack=3333 Win=64076 Len=2920 [TCP s
23812  2022-07-19 08:26:45.303…  [redacted]       184.105.247.195  TPKT    655  Continuation
23813  2022-07-19 08:26:45.306…  184.105.247.195  [redacted]       TCP      54  56954 → 3389 [ACK] Seq=3333 Ack=30720 Win=42496 Len=0
23814  2022-07-19 08:26:45.306…  184.105.247.195  [redacted]       TCP      54  56954 → 3389 [ACK] Seq=3333 Ack=32180 Win=41984 Len=0
23815  2022-07-19 08:26:45.306…  184.105.247.195  [redacted]       TCP      54  56954 → 3389 [ACK] Seq=3333 Ack=33640 Win=40960 Len=0
23816  2022-07-19 08:26:45.306…  184.105.247.195  [redacted]       TCP      54  56954 → 3389 [ACK] Seq=3333 Ack=34241 Win=42496 Len=0
23817  2022-07-19 08:26:45.330…  [redacted]       184.105.247.195  TPKT    155  Continuation
23818  2022-07-19 08:26:45.332…  184.105.247.195  [redacted]       TCP      54  56954 → 3389 [ACK] Seq=3333 Ack=34342 Win=42496 Len=0
23958  2022-07-19 08:26:49.344…  184.105.247.195  [redacted]       TCP      54  56954 → 3389 [FIN, ACK] Seq=3333 Ack=34342 Win=42496 Len=0
23960  2022-07-19 08:26:49.346…  [redacted]       184.105.247.195  TCP      54  3389 → 56954 [FIN, ACK] Seq=34342 Ack=3334 Win=64076 Len=0
23964  2022-07-19 08:26:49.348…  184.105.247.195  [redacted]       TCP      54  56954 → 3389 [ACK] Seq=3334 Ack=34343 Win=42496 Len=0

> Frame 23328: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface eth0, id 0
> Ethernet II, Src: fe:00:00:00:01:01 (fe:00:00:00:01:01), Dst: 26:ac:19:93:ba:e7 (26:ac:19:93:ba:e7)
> Internet Protocol Version 4, Src: 184.105.247.195, Dst: [redacted]
> Transmission Control Protocol, Src Port: 56954, Dst Port: 3389, Seq: 1, Ack: 1, Len: 46
  Transport Layer Security

0000  [redacted]  &·······  ······E·
0010              ·V_@@·5·  s_·i····
0020              ···z·=··  ···Ai·P·
0030              ·S·[····  ·.)·····
0040              ·Cookie:  mstshas
0050              h=2chkcp  jg······
0060              ····

external_00091_20220719081716.pcap          Packets: 100555 · Displayed: 109 (0.1%)          Profile: Default
```

Without PyRDP, client disconnect occurs after basic settings exchange

Figure 34. Results of Rdpscan without Using PyRDP.

Another observation was a significant increase in network traffic received by our honeypot when we switched our public IP address at the beginning of Experiment 3. This could possibly be attributed to DigitalOcean reusing IP addresses of decommissioned machines. From what was observed on Shodan at the beginning of Experiment 3 (July 15, 2022), apparently our new public IP address had previously routed traffic to a web server. Figure 35 shows what had appeared on Shodan between July 6 and July 22 when searching for our new IP address of the user-interface machine. Shodan's report showed that as of July 6, 2022, our new IP address only had open ports for SSH, HTTP, and HTTPS.



// TAGS: cloud

// LAST SEEN: 2022-07-06

General Information

Cloud Provider	DigitalOcean
Cloud Region	us-ca
Country	United States
City	Santa Clara
Organization	DigitalOcean, LLC
ISP	DigitalOcean, LLC
ASN	AS14061
Operating System	Ubuntu

Open **Ports**

22 80 443

// **22** / TCP 820745162 | 2022-07-06T19:38:40.154167

OpenSSH 8.2p1 Ubuntu-4ubuntu0.4

```
SSH-2.0-OpenSSH_8.2p1 Ubuntu-4ubuntu0.4
Key type: ssh-rsa
Key: AAAAB3NzaC1yc2EAAAADAQABAAABgQDeGKs1cdlLa0yH3BoIXDAHz2v0IgPv/raBydEQ0jwSA+Dn
tOsjpyeaVSeiJqyamdpBvzn5+dmowDHNvx2WJrnq6xt2DMpuFNZtPiI0O82tfKtPXBId6bPJPGIL
MZfQdUm02BhLNcUiR+i+s15wKk/F5PAHC+US5QTtSSr6dLU9YuAbvl7mh2/AtM2XSShPD9ysqPNf
yWPlopUdWLDc/3OuEVfa2Us0QqniVlYX+12T8xF7+mhmElTCLnm39hecwD3MYFEJHjULkY+6Q8WV
kc+KHXvaOkzkIRv40QWbCXA8ku1WyhL81zfvPk2FIZmBpBdkV6qk0Hi/gcV/JDJwzeClEzhFSN7J
9PjgY9mjT2CDix2J8yb3Y/2lyzpSs36P9IEHvfZZl8fiiWeXwv0uJRyPvLERCigQhj9IdmQ96ewa
kVgSfp3577y4lEqeIoe167klGBLmI/B7L1TZ8QbgHS1TeNl7EAzp8UPblZDe62i8lrst7dSWRvDF
JEquHTkJnOM=
Fingerprint: 28:77:8b:9a:73:0f:94:7b:32:26:74:06:41:6b:a9:33

Kex Algorithms:
        curve25519-sha256
        curve25519-sha256@libssh.org
        ecdh-sha2-nistp256
```

Figure 35. Shodan Showing Open HTTP, HTTPS, and SSH Ports for the Previous Use of Our IP Address

When our IP address was rescanned by Shodan on July 22, Shodan's database was updated to include port 3389 in its list of open ports for our IP address; however the amount of traffic received by the honeypot steadily declined then (Figure 36). The Web and SSH ports did not disappear from Shodan's database until August 6 when our user-interface IP address was scanned again. Possibly our honeypot was attacked at a high rate at the beginning of Experiment 3 because attackers believed a Web server was running.

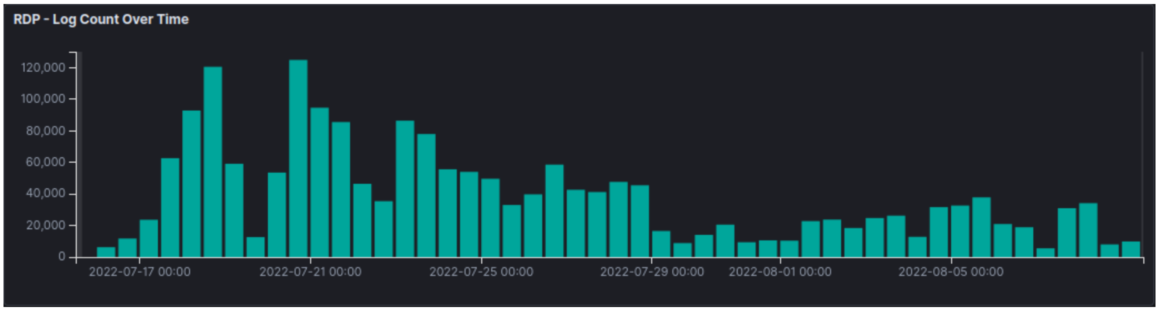


Figure 36. RDP Traffic Received During Experiment 3 between July 15 and August 9

Using Snort as an intrusion-detection system in our experiments produced no alerts for attempted RDP attacks since Snort cannot decrypt network traffic and most RDP connections are TLS-encrypted after the first phase of the connection sequence. From the Snort ruleset that we used, the BlueKeep detection rule was written as if it could see the creation of the MS_T120 virtual channel in cleartext.

We did not see any attacks use the Metasploit ms12_020_maxchannelids module to exploit our Windows machine. This is likely due to the age of the exploit and the number of Windows machines that are now patched against it. We observed many attempts to scan for and exploit BlueKeep, which is likely favored by attackers against newer versions of Windows because it can access machines instead of only causing denial of service.

While we had methods to characterize malicious vulnerability scanning using Nessus and OpenVAS, we did not observe any attackers use these tools to scan our honeypot. This could likely be from how long it took to perform a vulnerability scan using one of these tools; in our own testing, it took over 20 minutes to scan all common ports of our honeypot, and over 7 minutes to scan only port 3389. These tools might be used by persistent attackers when trying to figure out how to access a secured system. In our case, the attackers seemed to quickly realize that they could easily log into our Windows machine. Since our honeypot's login method was insecure by design, attackers had little reason to scan for vulnerabilities to access to our system.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. CONCLUSION AND FUTURE WORK

A. SUMMARY OF FINDINGS

Using our methods to generate and analyze RDP traffic, we could detect and characterize actual remote attacks on our honeypot. Our methods could distinguish RDP traffic that was “clearly malicious” through known cleartext signatures of attack, “very suspicious” for having established a remote-desktop session, “suspicious” for having exchanged more packets than a typical port scanner or web crawler, and “benign” for being unrelated to malicious sources and not having sent malicious data. We could also characterize the severity of malicious attacks through the decryption of RDP application data and the analysis of Windows event logs. In the experiments in which we used Snort as an intrusion-detection system, it was ineffective at detecting attacks on the Remote Desktop Protocol due to its inability to decrypt RDP network traffic. Overall, our methods for RDP attack characterization were effective for traffic received by the Windows machine of our GridPot honeypot, though they may not apply to other systems where an RDP private key cannot be extracted or a machine cannot be accessed to retrieve event logs.

During Experiment 1, we observed the most attempted BlueKeep scans and exploits against our Windows machine. All BlueKeep events were detected by PyRDP, and many of these we could detect and characterize by the features of randomly generated RDP cookies. We also saw some attackers establish remote-desktop sessions to our Windows machine; we could characterize the severity of these attacks as low because PyRDP-generated video replays of the sessions showed that no attackers made it past the Windows login screen.

From the small amount of network traffic that we could analyze from Experiment 2, we could detect and characterize several attacks as clearly malicious BlueKeep scanning due to the RDP cookies being used. Three malicious attacks that had established remote-desktop sessions to our Windows machine were characterized as low or moderate severity due to the duration of the remote-desktop sessions and the lack of malicious Windows event logs observed.

During Experiment 3, we saw over two-million RDP connections to our honeypot, the most of all experiments we ran. From this experiment, we further could evaluate the effectiveness of BlueKeep attack-characterization through analysis of randomly generated RDP cookies. We also observed many attacks that had established remote-desktop sessions; the severity of these attacks ranged from low to very high. Analysis of our Windows machine at the end of this experiment revealed that several of these high-severity attacks modified the system by infecting it with malware that we hypothesized to be extracting information about our system and network, scanning websites to collect information, and mining for cryptocurrency using our system's processing resources. This experiment also unintentionally introduced the potential for using recycled DigitalOcean IP addresses to attract the attention of attackers to a cloud-based honeypot.

From the small amount of data collected during Experiment 4, we did not observe any clearly malicious attacks, but we could use PyRDP for video replay of a "very suspicious" remote-desktop session that proceeded beyond the login screen. We saw an attacker trying to access a specific website using the Microsoft Edge Internet browser. By seeing attacker interactions with our honeypot, we could better confirm the severity of their attacks.

B. FUTURE WORK

We analyzed on the attacks that had established remote-desktop sessions longer than one minute to find artifacts of the attacks and confirm the severity of "very suspicious" attacks, but more static and dynamic analysis of the malware downloaded by attackers is needed to determine the exact behaviors of the executable files.

Our attack-characterization methods used well-known open-source tools. They can be refined and improved using newer tools as they are released from sources like GitHub and Metasploit. Another improvement is the creation of custom attack scripts that generate data for brute-force or dictionary attacks on an RDP server that is more secure than Windows machine used in this research. A machine-learning approach that uses the features of RDP connections can also be used to characterize the behaviors of attacks received by the Windows machine of the honeypot.

APPENDIX A. DATA COLLECTION AND ANALYSIS TOOLS

This appendix contains information to set up and run Malcolm, PyRDP, Snort, and T-Shark for capture and analysis of RDP data.

Malcolm Network Traffic Analysis Tool Suite:

The Malcolm Network Traffic Analyzer can be downloaded from GitHub at (CISAgov, 2022). A detailed installation guide can be found at (Malcolm, n.d.) which covers all steps needed to install it on a Linux Ubuntu 22.04 LTS machine. It is recommended for Malcolm to be installed on machines with at least 16 GB of memory, but more is better for analysis of extensive network traffic. It is important to note that Malcolm can only process network traffic in the pcap file format.

Start, stop, or clear data in Malcolm using the following commands:

```
./<malcolm root directory>/scripts/start
./<malcolm root directory>/scripts/stop
./<malcolm root directory>/scripts/wipe
```

Packet captures can be uploaded for automatic processing at:

```
https://localhost/upload
```

Processed network traffic can be viewed in Arkime to analyze information about each connection at:

```
https://localhost:443
```

OpenSearch Dasboards can visualize data of each connection at:

```
https://localhost:5601
```

Arkime can filter for specific connections using its search feature. The following is an example of the search filter that returns all information about connections destined for port 3389 that have exchanged more than 500 packets:

```
port.dst == 3389 && packets > 500
```

The following is an example of a search filter that returns all information about connections from a list of IP addresses that use the RDP cookie “hello”:

```
ip.src == [<addr 1>, <addr 2>] && zeek.rdp.cookie == hello
```

The “Hunt” feature of Arkime (described as PCAP grep) is useful for searching for strings in the packets exchanged during connections. Start by filtering for the connections that you want to search, then create a new Hunt job with the ASCII, hex, or regular expression that you are searching for. The following is an example of a regular expression used in a Hunt job to find signatures of Nessus plugins in connections:

```
mstshash=*.nbin
```

PyRDP:

PyRDP was downloaded from GitHub (GoSecure, 2022). Instructions for installation through the source-code was found in the “installing” section of its GitHub page. It is recommended to install PyRDP dependencies in a Python virtual environment to avoid issues with conflicting software packages on the host machine.

With the Python virtual environment activated, PyRDP MitM can be run with the following command:

```
pyrdp-mitm.py <rdp server address>:<rdp server port>
```

Launch the PyRDP Player’s graphical interface to view generated replays with the following command:

```
pyrdp-player.py
```

Snort:

Note: Snort version 2.9 was used for this work.

Download Snort 2.9 on a Linux machine using the following command:

```
sudo apt install snort
```

The latest ruleset for Snort 2.9 can be downloaded for registered users from (Cisco, 2022). Replace the default community ruleset with the registered-user ruleset with the following commands in the order shown:

```
tar -xvf <path to registered user ruleset>.tar.gz
sudo rm -r /etc/snort
sudo cp -r <path to registered user ruleset> /etc/snort
```

Snort can then be run in network intrusion-detection mode with the following command:

```
sudo snort -A full -i <network interface to listen on> -c
     > /etc/snort/etc/snort.conf
```

T-Shark:

To capture network traffic on the user-interface machine while filtering out administrative SSH traffic and rotating files every 20 MB, run:

```
tshark -i eth0 -f "not (<admin IP address> and tcp port 22)"
     > -w <output file> -b filesize:20000 &
```

To capture network on the user-interface machine, but using the pcap file format instead of pcapng, run:

```
tshark -i eth0 -f "not (<admin IP address> and tcp port 22)"
     > -w <output file> -F pcap -b filesize:20000 &
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. PCAPNG TO PCAP CONVERSION PYTHON SCRIPT

This appendix contains Python 3 code for converting files from the pcapng format (default T-Shark output format) to the pcap format to be compatible with Malcolm. This code automates the Editcap command-line utility that comes with Wireshark to process multiple pcapng files for conversion through the execution of a single command on a Linux machine. An input directory containing pcapng files is required as the first command-line argument for this script. The second command-line argument specifies the output directory for converted files. The naming of converted files follows the format “c_<pcapng file name>.pcap.” Converted files can be uploaded to Malcolm’s web interface without being rejected for being of an unsupported file format.

ConvertPCAPNGToPCAP.py

```
import os
import subprocess
import sys

CMD = “editcap -F pcap”

if len(sys.argv) != 3:
     print(“Must provide input and output directories”)
     sys.exit(1)

# Get user’s command-line input
input_dir = sys.argv [1]
output_dir = sys.argv [2]

# Check if input directory exists
if os.path.exists(input_dir)!= True:
     print(“Error: input directory does not exist”)
     sys.exit(1)

# Create output directory if it does not exist already
if os.path.exists(output_dir) != True:
     print(“INFO: creating output directory”)
     command = “ “.join([“mkdir,” output_dir])
     subprocess.call(command, shell=True)

# Convert each input file to pcap and place in output dir
pcapngs = os.listdir(input_dir)

for p in pcapngs:
     infile = input_dir + “/” + p
     outfile = output_dir + “/c_” + p
     command = “ “.join([CMD, infile, outfile])

     return_val = subprocess.call(command, shell=True)
     print(“return value: ,” return_val)
```

Example of use:

```
python3 ConvertPCAPNGtoPCAP.py <input dir> <output dir>
```

Assuming that the ConvertPCAPNGtoPCAP.py script is in the Desktop directory and we want to convert pcapng files from the input directory “captures” to the output directory “converted,” enter the following:

```
python3 ConvertToPCAP.py captures converted
```

If “file1.pcapng,” “file2.pcapng,” and “file3.pcapng” were in the “captures” directory, then their newly converted files would be “c_file1.pcap,” “c_file2.pcap,” and “c_file3.pcap” inside of the “converted” directory.

APPENDIX C. PYTHON SCRIPT FOR FORMATTING IP LISTS

This appendix contains Python 3 code to format lists of IP addresses as search terms for Malcolm’s Arkime interface. Arkime supports IP address lists as search terms for filtering of network traffic, which can be powerful to display or remove results from specific IP sources. This script takes as input a single file containing a list of IP addresses separated by newline characters; this is the same format that Arkime uses when exporting lists of unique IP addresses. The output of the script is an IP address list that is formatted as a search term that is ready to be used in Arkime, which can be pasted from the command-line to the Arkime search bar.

FormatIPfilter.py

```
import sys

# Check for command-line argument
if len(sys.argv) != 2:
     print(“Error: must provide input file”
     sys.exit(1)

# Open input file
ips = open(sys.argv [1], “r”)

# Create formatted IP list for Arkime search filtering
ipformat = “[“

for ip in ips:
     ipformat += ip [:-1]          # remove \n
     ipformat += “,”          # comma to separate next IP

ipformat = ipformat [:-1] + “]”    # remove final comma

# Print IP list search term to console
print(ipformat)
```

<u>Example of use:</u>

```
python3 FormatIPfilter.py <input file>
```

Assuming that the FormatIPfilter.py script is in the Desktop directory and we want to generate a formatted IP list from the file “unique.txt,” run:

```
python3 FormatIPfilter.py unique.txt
```

If the “unique.txt.” file had the following contents:

```
192.168.0.5
192.168.0.6
192.168.0.7
```

Then the output of the script would be:

```
[192.168.0.5,192.168.0.6,192.168.0.7]
```

To filter for network traffic related or not related to these IP addresses on Arkime, either of the following expressions could be used as search filters:

```
ip.src == [192.168.0.5,192.168.0.6,192.168.0.7]
ip.src != [192.168.0.5,192.168.0.6,192.168.0.7]
```

LIST OF REFERENCES

Ahlman, L. (2021). How sweet it is: a comparative analysis of remote desktop protocol honeypots. SANS Institute. https://sansorg.egnyte.com/dl/zPqkfv2ul1

AngryIP (2022). *IPScan* (Version 3) [Computer Software]. GitHub. https://github.com/angryip/ipscan

Arkime (n.d.). *Arkime—Full Packet Capture*. Retrieved August 8, 2022, from http://arkime.com

Attivo Networks (2022). *ThreatDefend® detection & response platform*. https://www.attivonetworks.com/product/threatdefend/

Beaulieu, A. (2020, October 20). *Announcing PyRDP 1.0*. GoSecure. https://www.gosecure.net/blog/2020/10/20/announcing-pyrdp-1-0/

Bieker, M. C., & Pilkington, D. (2020). Deploying an ICS honeypot in a cloud computing environment and comparatively analyzing results against physical network deployment [Master's thesis, Naval Postgraduate School]. NPS Archive: Calhoun. https://calhoun.nps.edu/handle/10945/66586

Boddy, M., Jones, B., & Stockley, M. (2019). *RDP exposed—the threat that's already at your door*. Sophos. https://www.sophos.com/en-us/medialibrary/PDFs/technical-papers/sophos-rdp-exposed-the-threats-thats-already-at-your-door-wp.pdf

Bruneau, G. (2018, December 16). *Random port scan for open RDP backdoor*. SANS. https://isc.sans.edu/diary/Random+Port+Scan+for+Open+RDP+Backdoor/24422

Bruneau, G. (2021, October 30). *Remote desktop protocol (RDP) discovery*. https://isc.sans.edu/diary/Remote+Desktop+Protocol+%28RDP%29+Discovery/27984

Buchanan, J. (2019, November 7). Securing RDP vulnerabilities: learnings from Bluekeep and DejaBlue. *Rapid7*. https://www.rapid7.com/blog/post/2019/11/07/the-anatomy-of-rdp-exploits-lessons-learned-from-bluekeep-and-dejablue/

Cisco (2022). Snort—network intrusion detection & prevention system. https://www.snort.org/

Climek, D., Macera, A., & Tirenin, W. (2016, March 8). *Cyber deception*. CSIAC. https://csiac.org/articles/cyber-deception/

Cloudflare (2022). *RDP security risks*. https://www.cloudflare.com/learning/access-management/rdp-security-risks/

Cook, B. (2019, September 6). Initial Metasploit Exploit Module for BlueKeep (CVE-2019-0708). *Rapid7*. https://www.rapid7.com/blog/post/2019/09/06/initial-metasploit-exploit-module-for-bluekeep-cve-2019-0708/

CISAgov (2022). *Malcolm* (Version 6) [Computer Software]. GitHub. https://github.com/cisagov/Malcolm

CTXIS (2016). *RDP-Replay* [Computer Software]. GitHub. https://github.com/ctxis/RDP-Replay

Cybersecurity and Infrastructure Security Agency [CISA] (2017, May 12). *Indicators associated with WannaCry ransomware*. https://www.cisa.gov/uscert/ncas/alerts/TA17-132A

Cybersecurity and Infrastructure Security Agency [CISA] (2019, June 4). *NSA releases advisory on BlueKeep vulnerability*. https://www.cisa.gov/uscert/ncas/current-activity/2019/06/04/NSA-Releases-Advisory-BlueKeep-Vulnerability

Cybersecurity and Infrastructure Security Agency [CISA] (2021, February 11). *Compromise of U.S. water treatment facility*. https://www.cisa.gov/uscert/ncas/alerts/aa21-042a

Dougherty, J. T. (2020). *Evasion of honeypot detection mechanisms through improved interactivity of ICS-based systems* [Master's thesis, Naval Postgraduate School]. NPS Archive: Calhoun. https://calhoun.nps.edu/handle/10945/66065

Duncan, B., & Prakash, V. (2021, April 1). *Wireshark tutorial: decrypting RDP traffic*. Palo Alto Networks. https://unit42.paloaltonetworks.com/wireshark-tutorial-decrypting-rdp-traffic/

ESET Research (2022, April 12). *Industroyer2: Industroyer reloaded*. WeLiveSecurity. https://www.welivesecurity.com/2022/04/12/industroyer2-industroyer-reloaded/

Fortinet (2022). *Honey tokens*. https://www.fortinet.com/resources/cyberglossary/honey-tokens

Franco, J., Aris, A., Canberk, B., & Uluagac, A. S. (2021). A survey of honeypots and honeynets for internet of things, industrial internet of things, and cyber-physical systems. ArXiv:2108.02287 [Cs]. http://arxiv.org/abs/2108.02287

Gerhards, R. (2009). *The Syslog protocol*. https://www.rfc-editor.org/info/rfc5424.

Giacobbi, G. (2006, November 1). *The GNU Netcat*. SourceForge. http://netcat.sourceforge.net/

GoSecure (2022). *PyRDP* (Version 1) [Computer Software]. GitHub. https://github.com/GoSecure/pyrdp

Greenbone Networks (2022). *OpenVAS - open vulnerability assessment scanner*. https://www.openvas.org/

Hancock, G. (n.d.). *Cyber deception: how to build a program*. Attivo Networks. Retrieved May 17, 2022, from https://www.attivonetworks.com/wp-content/uploads/sites/13/documentation/Attivo_Networks-Cyber_Deception_GH.pdf

Idaho National Laboratory. (n.d.). *Malcolm*. Idaho National Laboratory. Retrieved July 29, 2022, from https://inl.gov/ics-malcolm/

IDrive (2022). *RemotePC by IDrive*. https://www.remotepc.com//

Kendrick, M., & Rucker, Z. (2019). *Energy-grid threat analysis using honeypots* [Master's thesis, Naval Postgraduate School]. NPS Archive: Calhoun. https://calhoun.nps.edu/handle/10945/62843

Mathezer, S. (2021, May 14). *Introduction to ICS Security*. SANS. https://www.sans.org/blog/introduction-to-ics-security/

Malcolm (n.d.). *Documentation*. Retrieved August 28, 2022, from https://malcolm.fyi/documentation/#QuickStart

Meier, J. (2022). *Hardening Windows-based honeypots to protect collected data* [Master's Thesis, Naval Postgraduate School]. NPS Archive: Calhoun.

Metasploit (2022). *Ms12_020_maxchannelids* [Computer Software]. GitHub. https://github.com/rapid7/metasploit-framework/blob/60a045eaaa50718f7cde649d4687a0e90740198d/modules/auxiliary/dos/windows/rdp/ms12_020_maxchannelids.rb

Microsoft (2010). *How terminal services works: terminal services*. https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2003/cc755399(v=ws.10)

Microsoft (2020a). *Remote desktop protocol*. https://docs.microsoft.com/en-us/windows/win32/termserv/remote-desktop-protocol

Microsoft (2020b). *Remote desktop services (remote desktop services)*. https://docs.microsoft.com/en-us/windows/win32/termserv/terminal-services-portal

Microsoft (2021a). *About event logging*. https://docs.microsoft.com/en-us/windows/win32/eventlog/about-event-logging

Microsoft (2021b). *[MS-RDPBCGR]: client MCS connect initial PDU with GCC conference create request*. https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-rdpbcgr/db6713ee-1c0e-4064-a3b3-0fac30b4037b

Microsoft (2021c). *[MS-RDPBCGR]: enhanced RDP security*. https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-rdpbcgr/592a0337-dc91-4de3-a901-e1829665291d

Microsoft (2021d). *Understanding remote desktop protocol (RDP)*. https://docs.microsoft.com/en-us/troubleshoot/windows-server/remote/understanding-remote-desktop-protocol

Microsoft (2022a). *[MS-RDPBCGR]: client X.224 connection request PDU*. https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-rdpbcgr/18a27ef9-6f9a-4501-b000-94b1fe3c2c10

Microsoft (2022b). *[MS-RDPBCGR]: connection sequence*. https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-rdpbcgr/023f1e69-cfe8-4ee6-9ee0-7e759fb4e4ee

Microsoft (2022c). *Remote desktop clients*. https://docs.microsoft.com/en-us/windows-server/remote/remote-desktop-services/clients/remote-desktop-clients

NeutrinoLabs (2022). *Xrdp* [Computer Software]. GitHub. https://github.com/neutrinolabs/xrdp

Nmap (n.d.). *Nmap: The Network Mapper*. Retrieved July 29, 2022, from https://nmap.org/

NXLog (2022). *Documentation*. https://nxlog.co/documentation/nxloguser-guide/about-nxlog.html

Office of Cybersecurity, Energy Security, and Emergency Response [OCESER] (n.d.). *Colonial pipeline cyber incident*. Retrieved August 11, 2022, from https://www.energy.gov/ceser/colonial-pipeline-cyber-incident

OffSec Services (2022). *Unicornscan*. https://www.kali.org/tools/unicornscan/

OpenSearch (2022). *About OpenSearch*. https://opensearch.org/

Palo Alto Networks (2022). *What is an intrusion detection system?* https://www.paloaltonetworks.com/cyberpedia/what-is-an-intrusion-detection-system-ids

Rapid7 (n.d.-a). *CVE-2019-0708 BlueKeep RDP remote windows kernel use after free*. Retrieved August 22, 2022, from https://www.rapid7.com/db/modules/exploit/windows/rdp/cve_2019_0708_bluekeep_rce/

Rapid7 (n.d.-b). *Deception technology*. Retrieved August 11, 2022, from https://www.rapid7.com/solutions/deception-technology/

Rapid7 (n.d.-c). *Honeypots*. Rapid7. Retrieved August 11, 2022, from https://www.rapid7.com/fundamentals/honeypots/

Reiner, S. (2020, April 7). Explain like I'm 5: remote desktop protocol (RDP). *Cyberark*. https://www.cyberark.com/resources/threat-research-blog/explain-like-i-m-5-remote-desktop-protocol-rdp

RobertDavidGraham (2019). *Rdpscan* [Computer Software]. GitHub. https://github.com/robertdavidgraham/rdpscan

Scarfone, K. A., & Mell, P. M. (2007). *Guide to intrusion detection and prevention systems (IDPS)* (NIST SP 800-94; 0 ed., p. NIST SP 800-94). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-94

Stouffer, K., Pillitteri, V., Lightman, S., Abrams, M., & Hahn, A. (2015). *Guide to industrial control systems (ICS) security* (NIST SP 800-82r2; p. NIST SP 800-82r2). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-82r2

Tabdiukov (2018, November 11). *Mstshash=administr explained*. GitHub. https://github.com/olipo186/Git-Auto-Deploy/issues/221

TeamViewer (n.d.). *TeamViewer—the remote control and access solution for anyone*. Retrieved August 11, 2022, from https://www.teamviewer.com/en-us/products/teamviewer/

Tenable (2022a). *Unix content command line examples.* https://docs.tenable.com/nessus/compliancechecksreference/Content/UnixContentCommandLineExamples.htm

Tenable (2022b). *Nessus Essentials vulnerability scanner*. https://www.tenable.com/products/nessus/nessus-essentials

VirusTotal (n.d.). *VirusTotal—home*. https://www.virustotal.com/gui/home/upload

Wang, W. et al. (2013). Detecting targeted attacks by multilayer deception. *Journal of Cyber Security and Mobility*, *2*(2), 175–199. https://journals.riverpublishers.com/index.php/JCSANDM/article/view/6141

Washofsky, A. D. (2021). *Deploying and analyzing containerized honeypots in the cloud with T-pot* [Master's thesis, Naval Postgraduate School]. NPS Archive: Calhoun. https://calhoun.nps.edu/handle/10945/68394

Wireshark (n.d.). *Wireshark · Go Deep.* Retrieved August 08, 2022, from https://www.wireshark.org/

XMRig (n.d.). *XMRig*. https://xmrig.com/

The Zeek Project (2020). *The Zeek network security monitor*. https://zeek.org/

Zoho (2022). *Zoho Assist*. https://www.zoho.com/assist/index.html

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California