NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

HARDENING WINDOWS-BASED HONEYPOTS TO PROTECT COLLECTED DATA

by

Joseph T. Meier

June 2022

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

THIS PAGE INTENTIONALLY LEFT BLANK

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

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

4. TITLE AND SUBTITLE HARDENING WINDOWS-BASED HONEYPOTS TO PROTECT COLLECTED DATA	**5. FUNDING NUMBERS** RCPY0
6. AUTHOR(S) Joseph T. Meier	
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) DOE	**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)

Digital honeypots are computers commonly used to collect intelligence about new cyberattacks and malware behavior. To be successful, these decoys must be configured to allow attackers to probe a system without compromising data collection. Previous research at the Naval Postgraduate School developed an industrial control system (ICS) honeypot simulating a small electric-distribution system. This honeypot was attacked, and its log data was deleted. Our research analyzed the attacks and developed methods to harden the main weakness of the publicly accessible user interface. The hardened honeypot included more robust data collection and logging capabilities and was deployed in a commercial cloud environment. We observed significant scanning and new attacks, including the well-known BlueKeep exploit. Our results showed that the added security controls, monitoring, and logging were effective but imperfect in protecting the honeypot's data and event logs. This work can help improve the security of industrial control systems used in both the government and private sectors.

14. SUBJECT TERMS honeypot, industrial control systems, ICS, cybersecurity, cyberdeception	**15. NUMBER OF PAGES** 95
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

HARDENING WINDOWS-BASED HONEYPOTS TO PROTECT COLLECTED DATA

Joseph T. Meier
Captain, United States Marine Corps
BSCE, The Ohio State University, 2014

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
June 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

Digital honeypots are computers commonly used to collect intelligence about new cyberattacks and malware behavior. To be successful, these decoys must be configured to allow attackers to probe a system without compromising data collection. Previous research at the Naval Postgraduate School developed an industrial control system (ICS) honeypot simulating a small electric-distribution system. This honeypot was attacked, and its log data was deleted. Our research analyzed the attacks and developed methods to harden the main weakness of the publicly accessible user interface. The hardened honeypot included more robust data collection and logging capabilities and was deployed in a commercial cloud environment. We observed significant scanning and new attacks, including the well-known BlueKeep exploit. Our results showed that the added security controls, monitoring, and logging were effective but imperfect in protecting the honeypot's data and event logs. This work can help improve the security of industrial control systems used in both the government and private sectors.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION....1
A. RESEARCH PLAN....2
B. THESIS OUTLINE....2

II. BACKGROUND AND RELATED WORK....5
A. INDUSTRIAL CONTROL SYSTEM SECURITY....5
1. Hardening ICS....5
2. Current Threats....6
B. HONEYPOTS....6
C. EVASION OF LOGGING....7
D. ICS HONEYPOTS....8

III. METHODOLOGY....11
A. NETWORKING PROTOCOLS....11
1. HTTP....11
2. Syslog....11
3. IEC 60870-5-104....11
4. Remote Desktop Protocol....12
B. CONPOT AND GRIDPOT....14
C. HONEYPOT LOGGING....15
1. Sysmon....15
2. NXLog....16
3. PyRDP....16
4. Windows Event Logs....17
D. OTHER NPS HONEYPOTS....18

IV. DESIGN AND IMPLEMENTATION....21
A. OUR HONEYPOT DESIGN....21
1. Problems With the Dougherty Design....21
2. User-Interface Droplet....23
3. GridPot Droplet....25
4. Logging Droplet....28
B. EXPERIMENT IMPLEMENTATION....29
C. DATA ANALYSIS METHOD....32

V. RESULTS AND DISCUSSION....35
A. EXPERIMENT 1 RESULTS....35

1. External Scanning Observed and Login Attempts 35
2. Post-login Activities 39
3. Malicious Actions in the Windows 10 Virtual Machine 40
B. EXPERIMENT 2 RESULTS 41
1. External Scanning Observed 41
C. EXPERIMENT 3 RESULTS 42
1. External Scanning Observed and Login Attempts 42
2. 44
3. Post-login Activities 44
D. EXPERIMENT 4 RESULTS 45
1. External Scanning Observed and Login Attempts 45
2. Post-login Activities 47
3. Malicious Actions in the Windows 10 Virtual Machine 48
E. DISCUSSION 51

VI. CONCLUSION AND FUTURE WORK 55
A. SUMMARY OF FINDINGS 55
B. FUTURE WORK 56

APPENDIX A. WINDOWS AUDIT POLICY IMPLEMENTATION 57

APPENDIX B. NXLOG CONFIGURATION FILE 59

APPENDIX C. ANALYSIS TOOLS AND SCRIPTS 61

APPENDIX D. EXPERIMENT 1 INFECTION REPORT 63

APPENDIX E. BLUEKEEP EXPLOIT SCAN 71

LIST OF REFERENCES 73

INITIAL DISTRIBUTION LIST 77

LIST OF FIGURES

Figure 1. RDP Connection Sequence. Source: Microsoft (2022a).....13

Figure 2. Architecture of Dougherty's Phase 2 ICS Honeypot. Source: Dougherty (2020).....19

Figure 3. Interaction between the User Interface and the IEC104 Server.....21

Figure 4. Our Complete Architecture with Interactions.....23

Figure 5. Diagram of Our User-Interface Droplet.....24

Figure 6. Diagram of GridPot Droplet Used in Experiments 1–3 without an HTTP Server.....27

Figure 7. Diagram of GridPot Droplet Used in Experiment 4 with an HTTP Server.....28

Figure 8. Diagram of Logging Droplet.....29

Figure 9. NMAP Service Scan with PyRDP Running.....30

Figure 10. NMAP Service Scan without PyRDP Running.....30

Figure 11. GridPot Webpage Displayed to Attract ICS-specific Attacks.....32

Figure 12. Experiment 1 – Graph of TCP Scans on Port 3389 (RDP).....36

Figure 13. NTLM Brute-Force Attack in Network Traffic.....37

Figure 14. Experiment 2 – Graph of TCP Scans on Port 3389 (RDP).....42

Figure 15. Experiment 3 – Graph of TCP Scans on Port 3389 (RDP).....43

Figure 16. Experiment 4 – Graph of TCP Scans on Port 3389 (RDP).....46

Figure 17. Percentage of Scanning by Country for Each Experiment.....51

Figure 18. Event Log Entry for Successful Logon to the Windows Machine.....63

Figure 19. PCAP Data that Shows RDP CONNECTION.....64

Figure 20. DNS Request from Windows Machine for download.advanced_ip_scanner.com.....64

Figure 21. DNS Response for download.advanced_ip_scanner.com.....64

Figure 22. Registry Key for the Advanced IP Scanner Tool.....65
Figure 23. IP Range to Scan for Advanced IP Scanner.....65
Figure 24. Event Log Entries for Possible Log Manipulation or Evasion 67
Figure 25. BlueKeep Exploit Detected in the PyRDP Logs.....71
Figure 26. BlueKeep Exploit Detected in RDP Connection Sequence (PCAP). 72

LIST OF TABLES

Table 1. Experiment 1 – Top Ten Most-Guessed Usernames. 38

Table 2. Countries that Guessed Username “TINHOCTHUCHANH” 39

Table 3. Experiment 1 – Successful Logins to the Windows Interface. 40

Table 4. Experiment 3 – Top Ten Most-guessed Usernames 44

Table 5. Experiment 3 – Successful Logins to the Windows Interface 45

Table 6. Experiment 4 – Top Ten Most-Guessed Usernames 46

Table 7. Experiment 4 – Successful Logins to the Windows Interface. 48

Table 8. Summary of Scans by Advanced Port Scanner Tool. 49

Table 9. Local Times of the Alleged Countries That Started RDP Sessions. 52

Table 10. Dataset Size Comparison across All Experiments 53

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ASDU	Application Service Data Unit
CISA	Cybersecurity and Infrastructure Security Agency
FIPS	Federal Information Processing Standards
HMI	Human-Machine Interface
HTTP	Hypertext Transfer Protocol
ICS	Industrial Control System
IEC	International Electrotechnical Commission
IEC101	IEC 60870-5-104
IEC104	IEC 60870-5-101
NIST	National Institute of Standards and Technology
NTLM	New Technology Local-Area Network Manager
RDP	Remote Desktop Protocol
SCADA	Supervisory Control and Data Acquisition
SSH	Secure Shell Protocol
SYSLOG	System Logging Protocol
TTP	Tactics, techniques, and procedures
VM	Virtual Machine

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisors, Professor Nguyen and Professor Rowe. Your guidance throughout this journey has been invaluable and I can honestly say I am a better computer scientist because of your mentorship.

I would also like to thank all my friends and family for their encouragement during the past several years. I could not have done this without your support.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Industrial control systems (ICSs) can monitor and control many critical processes in societal infrastructure. For instance, ICSs are used in water-treatment plants, electrical grids, and mass transportation. A population relies on these systems every day to provide clean water and energy. In the past, ICSs were mainly run on isolated (air-gapped) networks. Today they operate in large distributed networks connected to the Internet. The increasing connectivity of these systems combined with their criticality has made them increasingly attractive targets to malicious actors. Attacks on ICSs have become common on critical infrastructure in the United States. In 2021 two attacks highlighted vulnerabilities in ICSs with national security implications. One attack tried to poison the water supply at a water treatment facility in Florida (Cybersecurity and Infrastructure Security Agency, 2021), and another attack used ransomware and shut down critical oil and gas pipelines along the US East Coast (Parfomak & Jaikaran, 2021).

Cyberattacks have become more common in larger military operations. Recognizing the importance of the cyber domain in warfare, in 2018 the United States Cyber Command was elevated to an independent and unified combatant command (United States Cyber Command, 2022). Other countries around the world are also developing similar commands for this domain. During the recent conflict between Russia and Ukraine, cyberattacks have been used with major military operations. In April 2022, the Russian hacker group Sandworm launched cyberattacks called Industroyer2 and CaddyWiper against an ICS network that controls electrical substations in Ukraine (ESET Research, 2022). Russian cyber actors also used Industroyer in 2015 and 2016 against electric grids in Ukraine which caused widespread blackouts (Cherepanov, 2017). Industroyer could exploit several protocols commonly used in ICSs to control electrical devices including IEC 60870-5-101 (IEC101), IEC 60870-5-104 (IEC104), and IEC 61850. Industroyer2, the updated malware Russia used in its most recent cyberattack, only targets devices communicating over the IEC104 protocol.

A recent advisory from the Cybersecurity and Infrastructure Security Agency and Federal Bureau of Investigation warned US energy organizations of Russian advanced

persistent threats (CISA, 2022). The advisory details historical tactics, techniques, and procedures used by Russia to target critical infrastructure, and recommends measures to address threats. ICSs are high-value targets for attackers and would likely be targeted during a major conflict. Even outside of conflicts, ICSs are probed for weaknesses that could be exploited in the future.

Honeypots (decoy systems) are one tool to collect data on attacks and malware. As attackers access the honeypot and move within it, logs provide defenders with attack patterns, entry points, and malware types. In previous research at NPS, a honeypot simulated the ICS for an electric grid (Dougherty, 2020). The honeypot included a Windows-based human-machine interface (HMI) that monitored and controlled the simulated grid. The user interface hosted a Supervisory Control and Data Acquisition (SCADA) program that communicated with the electrical grid over the IEC104 protocol. Problems with monitoring and collecting data from the honeypot were met when the honeypot was deployed. The honeypot was deployed twice; both times the Windows machine was attacked, and the tools monitoring the honeypot were disabled. Incomplete logs made recreating the attacks impossible. The challenge this research faced was to make the logging of the honeypot more robust so attackers could not evade it.

A. RESEARCH PLAN

We wanted to improve the honeypot used in previous research at NPS to better protect the collected data for offline analysis, or "harden" it. We developed a framework to make more secure the software configuration of the publicly accessible ICSs. Our approach uses a robust logging mechanism that records security-relevant events in detail and stores them on a separate site. We deployed the hardened honeypot in a cloud environment and ran four experiments. The data collected was analyzed for the tactics that attackers used to enable fine-tuning the honeypot's monitoring and logging.

B. THESIS OUTLINE

Chapter II gives an overview of ICS security, honeypot types, and work related to our research. Chapter III details protocols used in ICSs, the remote-desktop protocol, and other ICS honeypot research relevant to our project. Chapter IV discusses the experimental

plan and implementation of our honeypot with attention given to its logging methods. Chapter V shows our analysis of the collected data, and Chapter VI provides our conclusions and ideas for future work.

THIS PAGE INTENTIONALLY LEFT BLANK

II. BACKGROUND AND RELATED WORK

A. INDUSTRIAL CONTROL SYSTEM SECURITY

1. Hardening ICS

The rise in attacks on critical infrastructures that rely on industrial control systems (ICSs) in the US has encouraged strategies to protect them. The Cybersecurity and Infrastructure Security Agency has published a strategy for securing ICSs (2020). Their report describes four pillars to securing ICSs, and pillar 3 states, "Build 'deep data' capabilities to analyze and deliver information that the ICS community can use to disrupt the Cyber Kill Chain." The data collected by honeypots designed to imitate components of ICS systems can provide such deep data. It will show the attack patterns and the tactics, techniques, and procedures (TTPs) that attackers use against ICSs and will help develop defenses and harden systems that host ICS software.

The Instrumentation, Systems, and Automation Society described three steps an attacker would take to compromise an ICS: access its network, discover the industrial processes and how they are operated, and then control the process (Instrumentation, Systems and Automation Society, 2006). In the past, physically separating systems was relied on to prevent unauthorized access. It is more common now for ICSs to connect to the Internet. Remote-work mandates due to COVID-19 have caused even more ICS resources to be remotely accessible.

The National Institute of Standards and Technology (NIST) details strategies for securing ICSs (NIST, 2015). Along with basic controls for securing a network such as authentication, applying the principle of least privilege, and firewalls, this publication discusses network segmentation and segregation of ICS. Companies that operate ICSs have corporate networks, used by employees for administrative tasks like email and file transfers, and networks for ICS devices. Administrative tasks require connecting the corporate network to the Internet, but ICS networks should not. NIST Special Publication 800-82 recommends separating these networks as much as possible to prevent vulnerabilities in the corporate network from affecting the ICS network. Boundaries

between the two networks should include demilitarized zones and firewalls that restrict the flow of traffic between them.

2. Current Threats

As industrial control systems become more automated, the threat from cyberattacks is growing. In February 2021, a water treatment plant was attacked, and the amount of lye added to the water was increased (Cybersecurity and Infrastructure Security Agency, 2021). It is believed that the attackers exploited a vulnerability in a remote-desktop service to access its SCADA system. Obfuscation techniques were used by the attackers to hide malware and details of access to the system. The CISA report recommended strong passwords in remote-desktop sharing, two-factor authentication, and auditing the remote-desktop logs.

Attacks against ICS systems can involve ransomware, which encrypts data and makes it useless until the operator pays a ransom. Colonial Pipeline was the victim of a ransomware attack in April 2021 (Parfomak & Jaikaran, 2021). It is believed that initial access was gained using stolen credentials for the virtual private network. Attackers infected the computers with ransomware that demanded money in exchange for the encryption key, causing the oil pipeline operated by Colonial to shut down for six days.

B. HONEYPOTS

Honeypots are computer systems intended to deceive attackers that they are legitimate systems (Provos, 2004). Honeypots can be for research or production (Manzanares, 2017). Research honeypots collect attack patterns and behaviors to learn about existing or new cyber threats. Research honeypots can be more complex than production honeypots because good ones must provide attackers with services with which they can interact. Production honeypots can detect intruders in a network. They are decoys that will not provide genuine services and are less likely to produce false positives when probed or compromised. For example, a fake server could be deployed on a company network, but no systems on that network would be configured to connect to the fake server. Any intrusion that accessed the network and probed the server would indicate that the network had been compromised.

Honeypots can also be classified as providing a high, medium, or low level of interaction with users (Franco et al. 2021). Low-interaction honeypots only allow access to services that do not require interactive responses such as DNS or HTTP. These services need not be fully implemented and may only respond to certain messages. Low-interaction honeypots are easy to implement but are of limited effectiveness. High-interaction honeypots simulate more services, and higher-fidelity information can be gathered on the attackers' behaviors and tactics. Medium-interaction honeypots offer a degree of interactivity between low-interaction and high-interaction honeypots. Privilege escalation, data exfiltration, and persistence mechanisms are more likely observed in high-interaction honeypots (MITRE ATT&CK, 2021).

C. EVASION OF LOGGING

Attackers can try to conceal their activity with many evasion techniques (MITRE ATT&CK, 2021). A honeypot is useless if its monitoring can be subverted by an attacker. Network traffic monitors like Wireshark are popular for gathering data in a honeypot since they can reveal IP addresses and protocols that connect to the honeypot, ports used to access services, and data coming into or going out of the honeypot. Tools used for monitoring attackers should be carefully protected to avoid data loss. Operating systems log security events such as user logons and logoffs, remote connections, registry modification, and processes created in different logs. Attackers can try to disable event logging in general or for certain suspicious actions they take. MITRE offers several protections against event log disabling (MITRE ATT&CK, 2021).

The security audit policy on Windows systems defines which events should be logged (Microsoft, 2021a), so system administrators should periodically review it to check that changes have not been made. Another improvement is restricting permissions for modifying registry keys or files that affect logging. An attacker may erase the event logs to conceal their activity; this technique is easy to detect since there will be a large gap in the data, but attackers can conceal their behaviors. Mitigations for this technique include encrypting log files, logging remotely, and restricting access to files that contain event logs.

D. ICS HONEYPOTS

ICS honeypots must simulate services running on an industrial process. They can be deployed in operational ICS networks to detect unauthorized access (Industrial Control Systems Cyber Emergency Response Team, 2016). They can also be standalone systems that collect attack data. Because they imitate a system that controls a physical process, ICS honeypots are typically more complex than honeypots that only imitate non-physical systems like a Web server. Low-interaction ICS honeypots may simulate a service for communicating with a PLC or responding to queries of static information (Franco et al., 2021). High-interaction ICS honeypots may have sophisticated simulations of physical devices found in ICS, such as a water pump or electric switch, and may have human-machine interfaces to control them and show real-time status updates.

An early honeypot for ICS systems was the SCADA HoneyNet Project (Pothamsetty & Franz, 2004). It simulated industrial processes like SCADA and PLCs and collected data from attacks on them. In 2015, an implementation of the SCADA HoneyNet Project was deployed in locations around the world using Amazon's cloud environment as a platform (Serbanescu et al., 2015). This system simulated both Modbus and IEC104 protocols. To reduce the chance of losing logging information, a separate database node collected data from all deployments of their honeypot. The modularity of their honeypot let them to deploy their honeypot on a global scale with centralized management and data collection. They used a structured query language database to store their collected data on a central node. Since our honeypot is a standalone system, we stored our data in a traditional file system to simplify the implementation.

Low-interaction honeypots simply expose ports of well-known services and monitor their connections. More convincing honeypots should behave more like an ICS system (Navarro et al., 2018) and should simulate the infrastructure (water treatment, electric grid, etc.). Attackers should interact with the simulation and see data that appears to come from a physical machine. Another characteristic of a realistic ICS honeypot is concealment of monitoring, hiding honeypot activities like copying log data to not raise suspicion. Navarro et al. (2018) built a high-interaction SCADA honeypot that simulated a water treatment plant, a user interface, and a monitoring system. They collected two years

of data and found that most attacks targeted the software of the SCADA system rather than control of the industrial processes.

Another project deployed a realistic honeypot that simulated an entire company complete with a website and fake employee information (Hilt et al., 2020). It simulated a small industrial prototyping company and included several types of programmable logic controllers, an interface, a firewall, and a file server. Virtual networking computing (VNC) tools were publicly exposed on a workstation in their honeypot. Two ransomware attacks on the workstation were observed using a playback tool that showed the desktop as the attackers were inside the system. Both attacks installed malicious files and generated a message that said the files were encrypted and demanded payment in cryptocurrency to a wallet address. One attack installed cryptocurrency mining software on the machine.

Another group deployed honeypots in different regions of the world and compared their data to determine common exploits (Kelly et al. 2021). Honeypots were deployed on the Google Cloud Platform, Microsoft Azure, and Amazon Web Services. On each platform, multiple honeypots were deployed to implement industrial protocols. The honeypots used were VNClowpot (Magisterquis, 2019), a low-interaction honeypot that imitates a VNC service, and RDPY (Citronneur, 2020), a Python implementation of the RDP protocol. Their results showed that the most common attacks on cloud-service providers were against desktop sharing services. CVE-2001-0540 (MITRE, 2002) a vulnerability that causes denial-of-service against RDP on Windows NT and Windows 2000, was most often used against honeypots in Google Cloud Platform (Kelly et al. 2021).

THIS PAGE INTENTIONALLY LEFT BLANK

III. METHODOLOGY

This chapter discusses networking protocols relevant to our research and methods for logging activity in a honeypot. We also describe previous implementations of ICS honeypots at the Naval Postgraduate School that form the basis of our research.

A. NETWORKING PROTOCOLS

A variety of protocols can connect to ICS systems and their management systems. Our work studied some of the most important.

1. HTTP

The Hypertext Transfer Protocol (HTTP) is an application-level protocol communicating between Web servers and clients (Fielding et al., 1999). Port 80 is the best-known port it uses. Requests go from a client to a server using a "method", and the two most common are GET and POST. GET asks the server for a specified resource, and POST sends input to the server.

2. Syslog

Syslog is a protocol frequently used for log management. Syslog collects logs from "generator" processes and sends them to a central log server or "collector" (Gerhards, 2009). With log forwarding enabled on a "generator" system, if an attacker erases the logs on the originating computer, the logs will have already been backed up. Discrepancies between the logs on the syslog collector and the originating computer indicate compromise (Dahlberg & Pulls, 2016).

3. IEC 60870-5-104

IEC 60870-5-104, typically shortened to IEC104, is a protocol for electrical-power substation control and supervision (Clarke & Reynders 2004). It is an application-level protocol that communicates over TCP/IP. Data is transmitted in an Application Service Data Unit (ASDU) that can contain monitoring or control information. The three frame formats for IEC104 are I-format, S-format, and U-format. I-format sends information like

control messages that manipulate power levels, set points, and other parameters at the power substation. S-format reports supervisory information from the programmable logic controllers (PLCs). U-format activates and deactivates connections between a controlled station and the controlling station.

4. Remote Desktop Protocol

The Remote Desktop Protocol (RDP) is a desktop-sharing protocol (Microsoft, 2021c) common on Microsoft Windows computers. RDP enables network administrators to remotely control computers on a network, but it can also allow users to access unique resources available at a workstation. The increase in employees working from home due to COVID-19 has encouraged such use. We chose it to study because it is popular with attackers.

Figure 1 shows the connection sequence between a client and an RDP server (Microsoft, 2022a). Establishing a connection has ten main steps: connection initiation, basic settings exchange, channel connection, RDP-security startup, secure settings exchange, optional connect-time auto-detection, licensing, optional multi-transport bootstrapping, capabilities exchange, and connection completion. Once the last step occurs, data can be transferred between server and client. Typically, the server sends data representing its desktop screen to the client machine. The client displays the screen image and sends keystrokes and cursor movements to the server for processing.

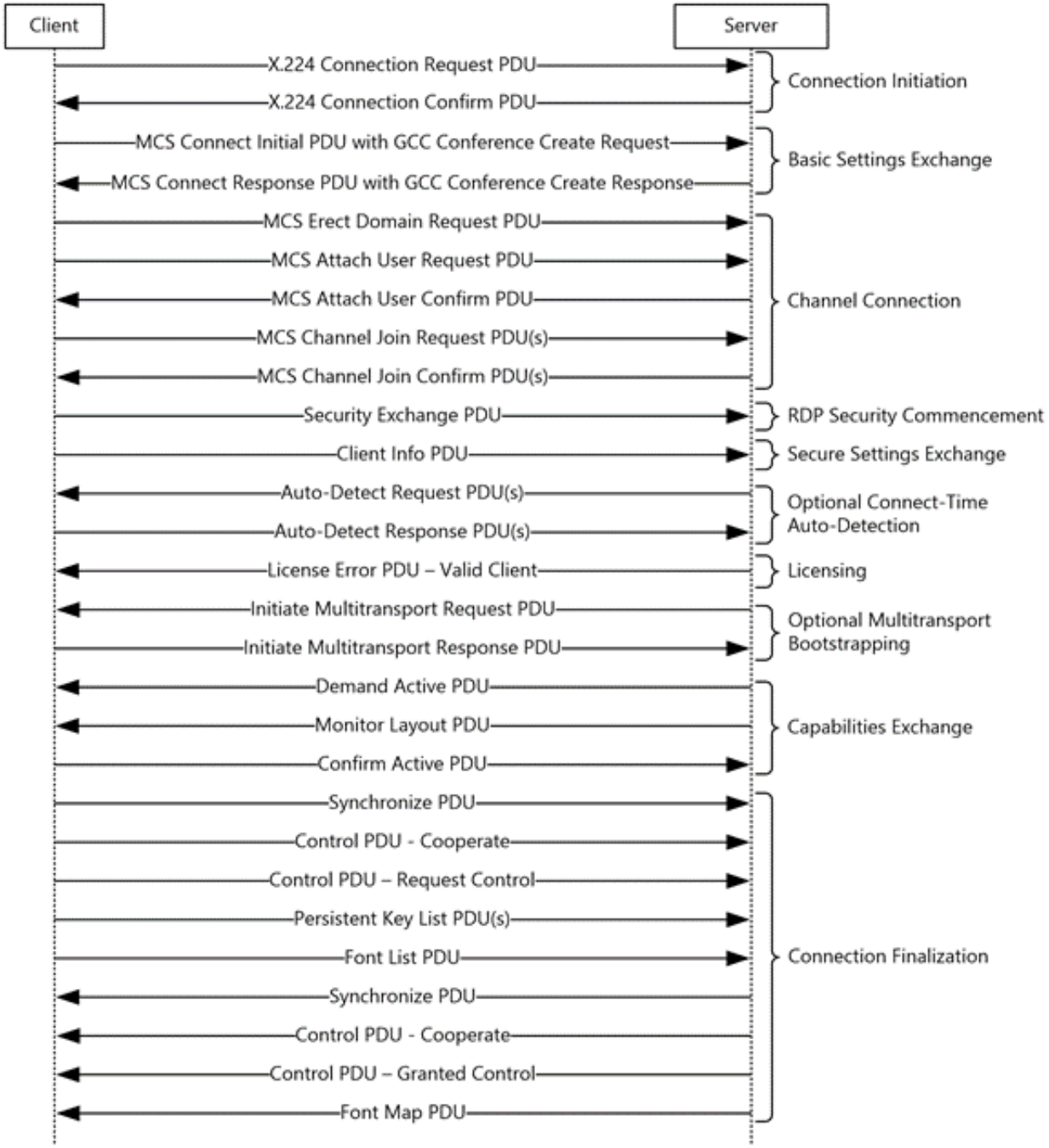


Figure 1. RDP Connection Sequence. Source: Microsoft (2022a)

During an active RDP connection, virtual channels are established between client and server (Microsoft, 2022a). There are two types of virtual channels, static and dynamic. Static channels are established during the basic settings exchange of the RDP connection sequence and persist through the RDP session. They communicate between components of

the server and the client. Common static channels are for sound output, clipboards, and filesystem access. Unlike static virtual channels, dynamic virtual channels can be connected and disconnected at any time during the RDP session. Their endpoints are applications which use the channel to exchange messages. Both the server and client must have channel managers to initialize and maintain them during the RDP session.

RDP offers many features that can be exploited by malware. BlueKeep and DejaBlue are well-known exploits against RDP (MITRE, 2019a) which target the virtual channels. BlueKeep starts with the client requesting a virtual channel MS_T120, a channel mainly used for internal communication. Remote connections that request its creation are not legitimate and such attempts are commonly attributed to BlueKeep. DejaBlue exploits a vulnerability in decompression of data sent over the DRDYNVC channel. Sending a carefully crafted string over this channel overflows the heap and allows the attacker to control the system. Since DRDYNVC is commonly used in legitimate RDP connections, its use is an insufficient indicator that DejaBlue has been attempted.

B. CONPOT AND GRIDPOT

Conpot is a low-interaction ICS honeypot that simulates common industrial processes and protocols (Conpot.org, n.d.). It can emulate several services common in ICSs including IEC104, HTTP, Modbus, and S7Comm. The default template for Conpot simulates an electric-grid with Siemens SIMATIC S7-200 programmable logic controllers and an HTTP server. Conpot can be extended by connecting it to real hardware like switches or pumps, or to an electric grid simulation like GridLab-D. It can be further extended by interfacing with an IEC104 server which can pass messages to an electric-grid simulation, and it can receive data back from the simulation to be displayed. Conpot also comes with an open-source Python implementation of an HTTP server.

GridPot is a medium-interaction honeypot that includes a high-fidelity simulation of electrical components, GridLab-D (Sk4ld, 2015). It uses a modified Conpot that can interface with GridLab-D. Changes were made to GridPot in a previous NPS thesis to translate messages sent to its IEC104 server into variables that are understood by the GridLab-D simulation (Dougherty, 2020). GridPot's modified Conpot HTTP server

displays information about the electrical components modeled by GridLab-D. Our GridPot implementation activates an HTTP server that displays data from the simulated electric grid.

GridLab-D simulates power distribution and was developed by the U.S. Department of Energy. It can simulate switches, transformers, regulators, and other typical parts of an electric grid. It supports several circuit models including the IEEE 13 Node Test Feeder model. We chose to use this model to compare our results with the results of previous NPS honeypot projects. Since our research focused on how attackers interact with a Windows-based interface that controls a power distribution system, we only needed to simulate a small electric grid that looked realistic and responded to control commands.

C. HONEYPOT LOGGING

1. Sysmon

Sysmon, short for system monitor, is part of the suite of Sysinternals tools (Russinovich & Garnier, 2022). Sysmon rules can filter out benign events and highlight suspected malicious events. Running the Sysmon tool under Windows creates a new event log that supplements the standard events recorded by the Windows event logging. Sysmon logs can identify anomalous behaviors and threats. Sysmon events that are relevant to this research include the following.

- Event ID 1, Process Creation: This records processes that are created. Information in a log record includes time, file path, user that created it, process ID, parent process ID, command-line arguments, and hashes of the executable file. Hashes can be submitted to known-malware databases to detect malicious files even if they were later deleted.

- Event ID 3, Network Connection: This records source and destination IP addresses and port numbers of TCP and UDP network connections. Much of this data can be ignored since it reports mostly routine activity.

- Event ID 4, Sysmon Service State Changed: This reports the start, stop, and logging policy changes of the Sysmon service. With no reason to stop

Sysmon on a honeypot, this event could indicate compromise and should be investigated.

- Event ID 8, Create Remote Thread: This often indicates that an attacker attempted injection, a common technique for attackers to hide their malware in legitimate processes.
- Event ID 11, File Create: This reports files created or overwritten. This can catch malware downloads that are subsequently deleted.
- Event ID 12/13/14, Registry Modification: These report changes to the Windows registry such as creating, deleting, or setting values of registry keys. The registry is often modified by malware to achieve persistence.
- Event ID 17/18, Pipe Created or Connected: These report instances of pipes being created or connected. Attackers often use pipes for inter-process communication. However, pipes are also used by legitimate processes.

2. NXLog

Attackers may erase logs to make post-exploitation analysis harder (MITRE ATT&CK, 2021). NXLog is an open-source log forwarding and collection tool that can forward Windows event logs to a centralized server (NXLog, 2022). Forwarding of event log records can happen quickly so even if an attacker immediately erases the event logs on a machine, most log records would have already been sent to the centralized log server. Storing logs of an attack on a secure remote machine increases the probability that they will be complete and unmodified.

3. PyRDP

PyRDP is an open-source tool that can be a “man-in-the-middle” (intermediate node) to intercept and decrypt RDP traffic (GoSecure, 2021). PyRDP records RDP sessions and replays them in either “headless” mode, showing only keyboard and mouse input, or in graphical mode, showing the desktop as if it had been recorded. PyRDP logs attempted

connections and recognizes some common RDP exploits like BlueKeep using known signatures, e.g., requests to set up MS_T120 channel.

Replaying RDP sessions can reveal attacker actions like opening a Web browser to download a file, running a SCADA program on an ICS system, or opening PowerShell to run a script. Analyzing suspicious events using PyRDP can be easier than searching large log files to find indicators of compromise. PyRDP does not record background processes that might be launched by an attacker, so it only supplements other monitoring mechanisms.

4. Windows Event Logs

Microsoft Windows' main logging mechanism is the Windows Event Log, whose data can be examined in the Event Viewer utility (Microsoft, 2021b). Five categories of events can occur:

1. Error: A significant problem such as data loss or loss of functionality.
2. Warning: A possible future problem such as low disk space.
3. Information: A successful operation of an application, driver, or service.
4. Success Audit: A successful security access event such as a login.
5. Failure Audit: A failed security access attempt, as when a user tries to access a file for which they lack permissions.

Every event has a unique Event ID that indicates what kind of event was logged. Event IDs associated with RDP are:

- 4624 – Successful logon attempt
- 4625 – Failed logon attempt
- 4778 – RDP session connected
- 4779 – RDP session disconnected
- 131 – RDP connection attempts

- 98 – RDP successful connections

Event IDs provide a quick way to track attempts to access the computer through RDP (Poling, 2018).

D. OTHER NPS HONEYPOTS

The first instance of GridPot at NPS (Kendrick & Rucker, 2019) ran HTTP, Modbus, and S7Comm servers, and collected data for 19 days. Most network traffic was HTTP, which responded with some fabricated data of the simulated electric grid. Of the 9,641 HTTP requests that were observed from the network traffic, 64% were POST requests that indicated attackers trying to manipulate the grid. Even though the Shodan Honeyscore tool characterized their system as "highly likely" to be a honeypot, attackers still tried to interact with it.

A subsequent project improved this design to create a high-interaction honeypot with an interface to a simulated electric grid (Dougherty, 2020). In Phase 2, a separate workstation running on a Windows 8 operating system showed a graphical user interface implemented with IndigoSCADA (Enscada, 2021). The interface allowed remote users to monitor and control the simulated electric grid by sending IEC104 messages to the GridPot server. RDP was enabled on the Windows 8 machine, so attackers could connect remotely and manipulate the machine once they logged in. Figure 2 shows the architecture of the improved honeypot.

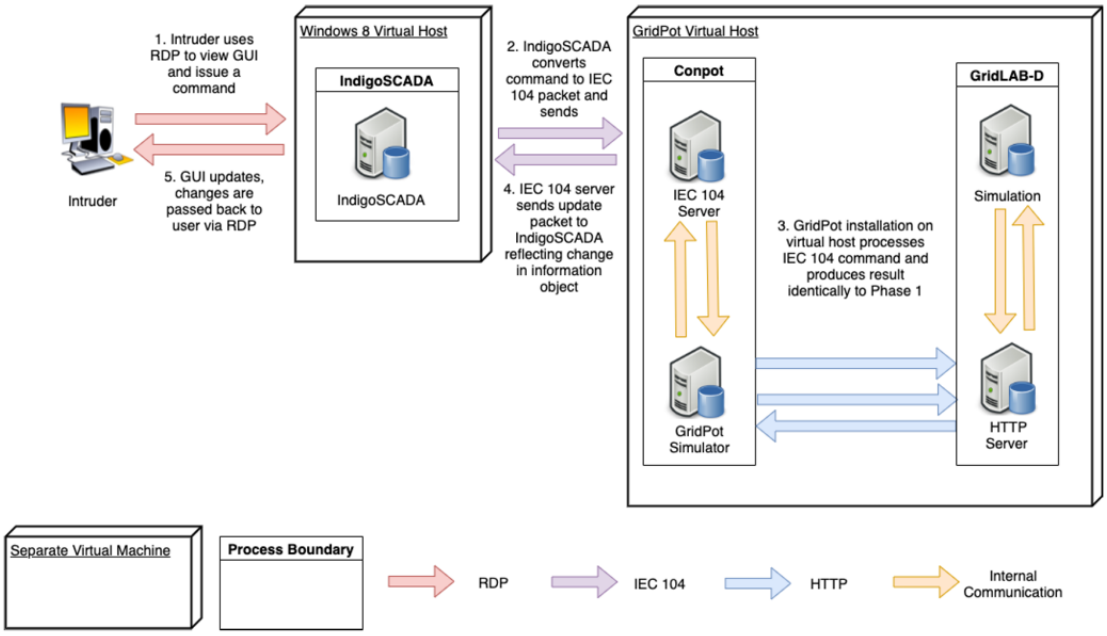


Figure 2. Architecture of Dougherty's Phase 2 ICS Honeypot. Source: Dougherty (2020).

The honeypot was deployed twice in Phase 2, and both times the logs were compromised. Wireshark on the Windows machine was turned off by the attackers, which made reconstructing the attacks difficult. Event logs on the Windows machine and what was left of the network traffic from Wireshark allowed some events of the attack to be pieced together. During the first attack, a cryptocurrency-mining tool was installed and run on the guest account. A desktop locking tool was also installed, and configured to run whenever the guest account logged in. The second attack disabled Windows Defender and installed a program suspected to be malware. The attacker also enabled the built-in Administrator account and created a new account in the Administrators group. Both attacks showed more interest in compromising the Windows operating system rather than in using the ICS functions.

Further work expanded on this by deploying the Phase 1 version of the honeypot, without the interactive user interface, in a cloud environment (Bieker & Pilkington, 2020). This version was limited to IEC104 and HTTP network traffic. They deployed two instances of GridPot on DigitalOcean "droplets" (virtual machines) in the United States

and one instance on a virtual machine in Asia. All three honeypots collected data for 18 days during the same period. Bieker and Pilkington found little difference in the traffic to GridPot in a cloud environment, except that some attacks may have been more sophisticated due to the additional obfuscation they added to make the honeypot look more realistic.

Another project used T-Pot as a honeypot deployment method (Washofsky, 2021). T-Pot is an integrated honeypot environment that supports several kinds of honeypots. Honeypots used in previous research and the T-Pot containerized versions of them were deployed over the same period and the data collected was compared. Results from this research showed little difference in the types and amount of traffic between the non-containerized honeypots and the T-Pot versions. This showed that T-Pot is a workable deployment method, and the analysis tools that come packaged with T-Pot enable a more robust analysis of the collected data.

IV. DESIGN AND IMPLEMENTATION

A. OUR HONEYPOT DESIGN

To lure attackers, our honeypot needed to provide a publicly accessible site that exposed an interface that appeared to monitor and control an electric grid. The computer that hosts the interface must be highly monitored, but the monitoring must also be stealthy to not raise suspicion about this system being a honeypot. Furthermore, the data collected must be protected from modification and deletion.

1. Problems With the Dougherty Design

Previous research at NPS mentioned in section III.D used a Windows-based system to implement a user interface for an ICS (Dougherty, 2020). Figure 3 shows the interaction between it and an IEC104 server that controls a simulated electric grid. Users remotely accessed the Windows workstation using the RDP protocol, and the user interface on the workstation communicated with the IEC104 server over port 2404. When the IEC104 server received a request for information from the electric grid, it queried the relevant electrical component in the simulated grid for the requested information, and the interface displayed the reply.

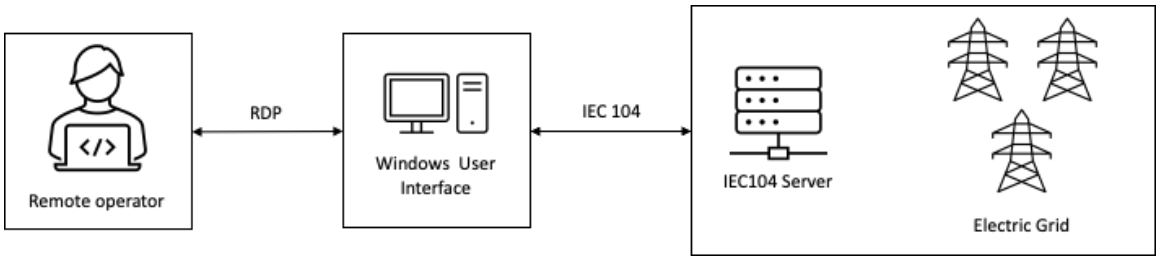


Figure 3. Interaction between the User Interface and the IEC104 Server.

Dougherty's Phase 2 honeypot provided the basis for our research. However, its logging system was easily disabled by attackers as described earlier. Goals of our research were to protect the data already collected and preventing attackers from stopping or subverting logging.

The Windows machine was the most vulnerable part of our honeypot since attackers could easily access it by the RDP protocol. Although the only account accessible by RDP had minimal privileges, the attacker could exploit unpatched vulnerabilities or misconfiguration to gain administrative privileges to stop processes, delete logs, or change logs. In Dougherty's honeypot, Wireshark was running in the administrative account, but the attackers could still stop it. Our solution was to run Windows in a virtual machine and to run Wireshark on a Linux host machine outside the Windows environment. Wireshark and logging could not be stopped by malware on the Windows system unless the attacker exploited a vulnerability in the hypervisor software to escape the virtual machine. The improved architecture of our honeypot and the interactions among its components are in Figure 4.

Our honeypot was implemented on three DigitalOcean virtual machines. DigitalOcean refers to their virtual machines as "droplets" (DigitalOcean, 2020). Hence, we named our three virtual machines the User-Interface Droplet, GridPot Droplet, and Logging Droplet. The User-Interface Droplet hosted a Windows 10 virtual machine running SCADA interface software and the Logging Droplet hosted the logging server. The User-Interface and Logging Droplets used Ubuntu Linux 20.04 LTS, the latest stable Ubuntu release. The GridPot Droplet was the same as that used in Bieker and Pilkington's research, which used Ubuntu Linux 18.04 LTS. Each DigitalOcean virtual machine has a public network interface that can be accessed over the Internet. Also, DigitalOcean allows users to create what they call "virtual private clouds" (VPC), a private network interface (DigitalOcean, 2020). This interface is not publicly accessible, but allows communications between virtual machines in the same VPC. We configured each machine with a private IP address to let them communicate over the VPC.

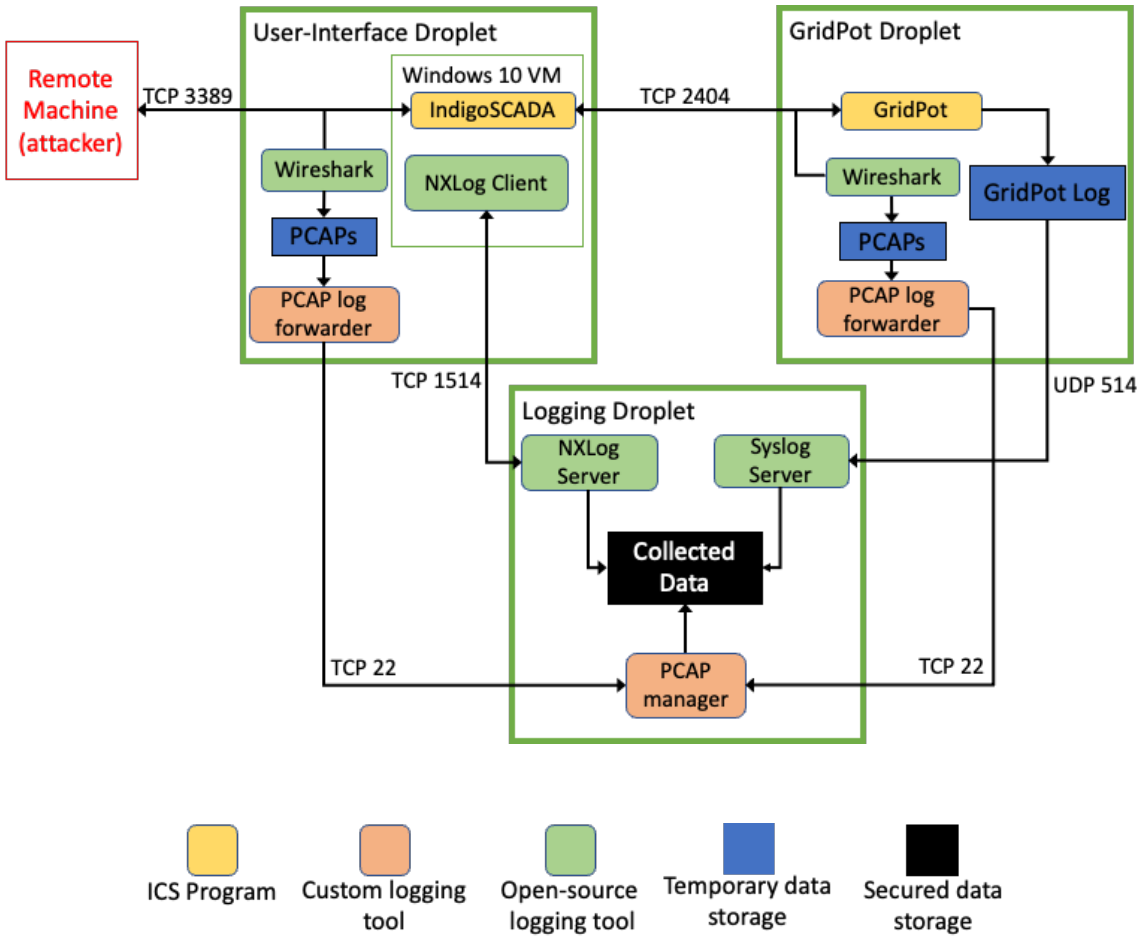


Figure 4. Our Complete Architecture with Interactions.

2. User-Interface Droplet

The User-Interface Droplet was a general-purpose machine with 16GB of memory and 100GB of solid-state disk (SSD) storage. It had one public IP address that could be accessed from the Internet and a private IP address for communicating with the other two machines. The User-Interface Droplet provided a SCADA interface running on a Windows 10 machine that attackers can use to manipulate the electric grid simulated by GridPot. Since DigitalOcean lacks pre-built Windows images, we could not run Windows 10 directly. Instead we ran a Windows 10 Education version 20H2 virtual machine using VirtualBox in the User-Interface Droplet as depicted in Figure 5.

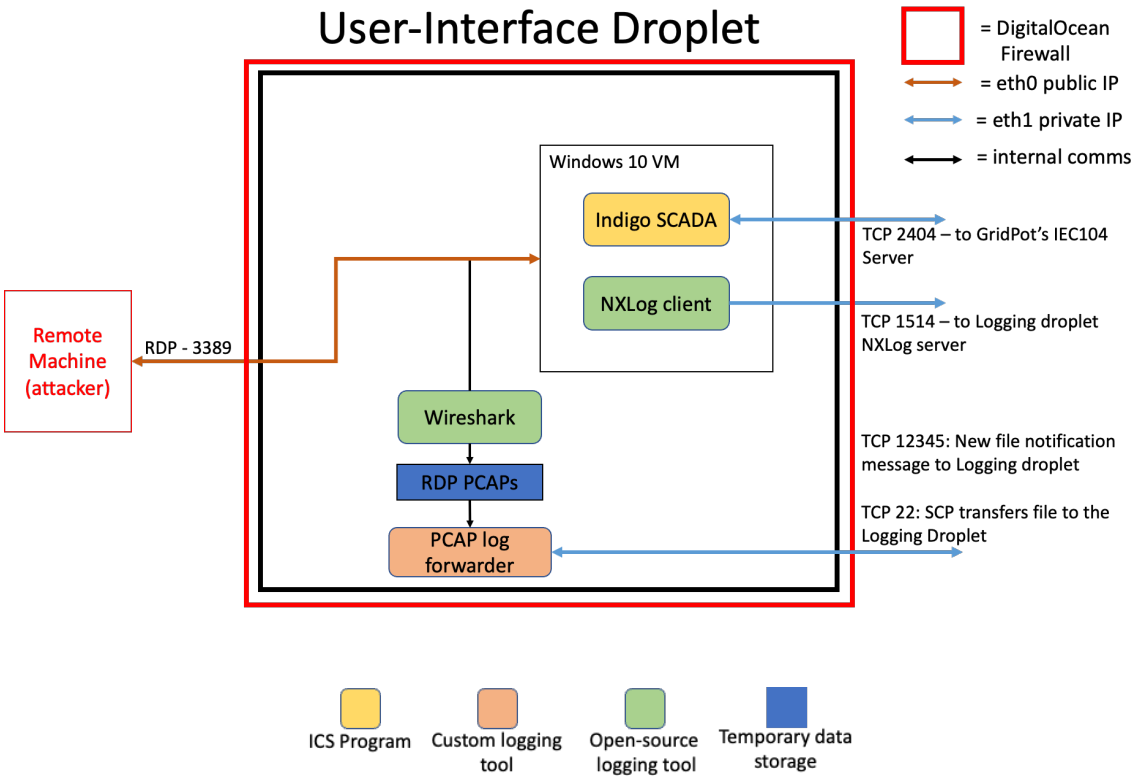


Figure 5. Diagram of Our User-Interface Droplet

We set the DigitalOcean firewall to only allow network traffic to port 3389 on the public interface and ports 22, 2404, 1514, and 12345 on the private interface. The Windows virtual machine ran with network-address translation so traffic to port 3389 on the external interface of the User-Interface Droplet was forwarded to port 3389 of the virtual machine. Wireshark ran outside the virtual machine and collected traffic on the public interface of the User-Interface Droplet. Wireshark was configured to save PCAP files every 20MB to a directory; files in the directory were automatically forwarded to the Logging Droplet. When PCAP storage reached the 20MB size limit, the forwarder notified the Logging Droplet over the private interface on TCP port 12345. Then the Logging Droplet started a secure-shell (SSH) session over TCP port 22 on the private interface of the User-Interface Droplet to copy the PCAP file to a directory on the Logging Droplet.

The Windows 10 virtual machine was configured similarly to the Windows 8 workstation used by Dougherty (2020). Two accounts were created on the Windows

machine: one with administrative privileges for configuring the workstation and one to use the SCADA interface. Only the latter would be accessible by the RDP protocol. We disabled remote logon to the administrator account to prevent attackers from tampering with the event logs. RDP is commonly accessed over port 3389, so attackers could scan for it to discover hosts with RDP publicly accessible.

The sole entry point on Dougherty's honeypot was the Guest account that was accessible by RDP. Remote logons to the Guest account were removed between Windows 8 and Windows 10 for security reasons. To make our honeypot as vulnerable as Dougherty's, we created a user account with a commonly used account name and a blank password. We also disabled network-level authentication so attackers would see the Windows logon screen without first entering the correct credentials, which encouraged attackers to access the user account. We were more interested in observing the actions attackers took once they accessed the user interface than in how they accessed it, hence we used an insecure RDP configuration.

To protect the Windows event logs, we used NXLog to forward the event logs from the Windows virtual machine to the Logging Droplet. Before the event logs were forwarded to the NXLog server on the Logging Droplet, they were converted to JSON format. On the Logging Droplet, NXLog listened on TCP port 1514 for data from the Windows virtual machine. We also configured NXLog to send log files every 100 MB. Once a log file reached the 100 MB limit it would close, and a new log file would be opened. The old log file had an integer appended to the end of the filename and was incremented by one for every new file.

3. GridPot Droplet

Our GridPot Droplet was the same system that Bieker and Pilkington deployed (Bieker & Pilkington, 2020; Dougherty, 2020). It ran the GridPot software with GridLab-D simulating the IEEE 13-node model with houses. An HTTP server in GridPot provided a user interface for the honeypot in which data from the electric grid was displayed but could not be controlled. Experiments 1-3 blocked traffic to the HTTP server listening on port 80 because we wanted RDP to be the only access to our honeypot. In Experiment 4

we allowed traffic to the HTTP server to advertise our honeypot as an ICS. Only limited information about the electric grid simulated by GridPot was displayed by the HTTP server. It also revealed the public IP address of the User-Interface Droplet, and directed users to access the Windows machine using RDP.

Figures 6 and 7 show the GridPot Droplet used in our experiments. GridPot's IEC104 server listened on TCP port 2404. This was the only way for the user interface to communicate with the GridPot Droplet. The SSH secure-shell protocol was also enabled, but it would only accept connections on the public interface from authorized users for administration. Wireshark monitored the private interface and was configured to rotate files every 20MB. When PCAP storage reached its size limit, it would be forwarded to the Logging Droplet. For further backup, we also modified the configuration file of GridPot to forward its logs to the Logging Droplet using the syslog protocol over UDP port 514 on its private network interface. Collecting packets with both Wireshark and the GridPot provided redundancy, so if one was lost or corrupted, the other could fill in the gaps.

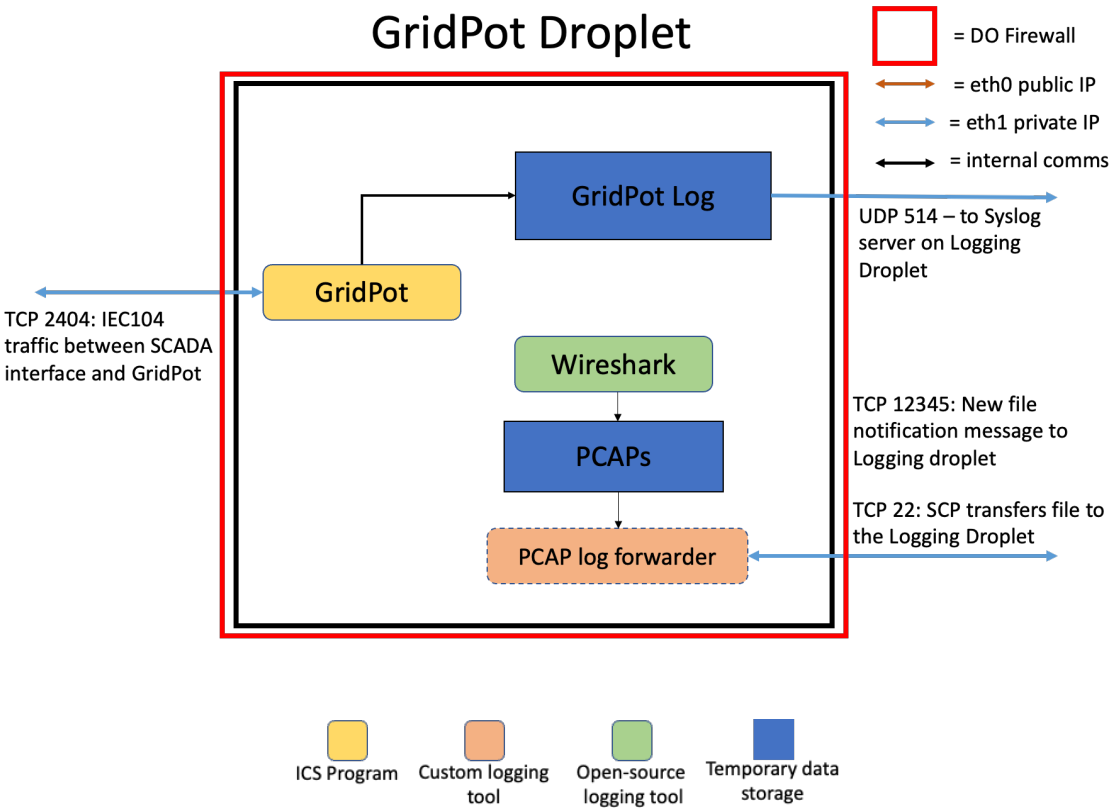


Figure 6. Diagram of GridPot Droplet Used in Experiments 1–3 without an HTTP Server

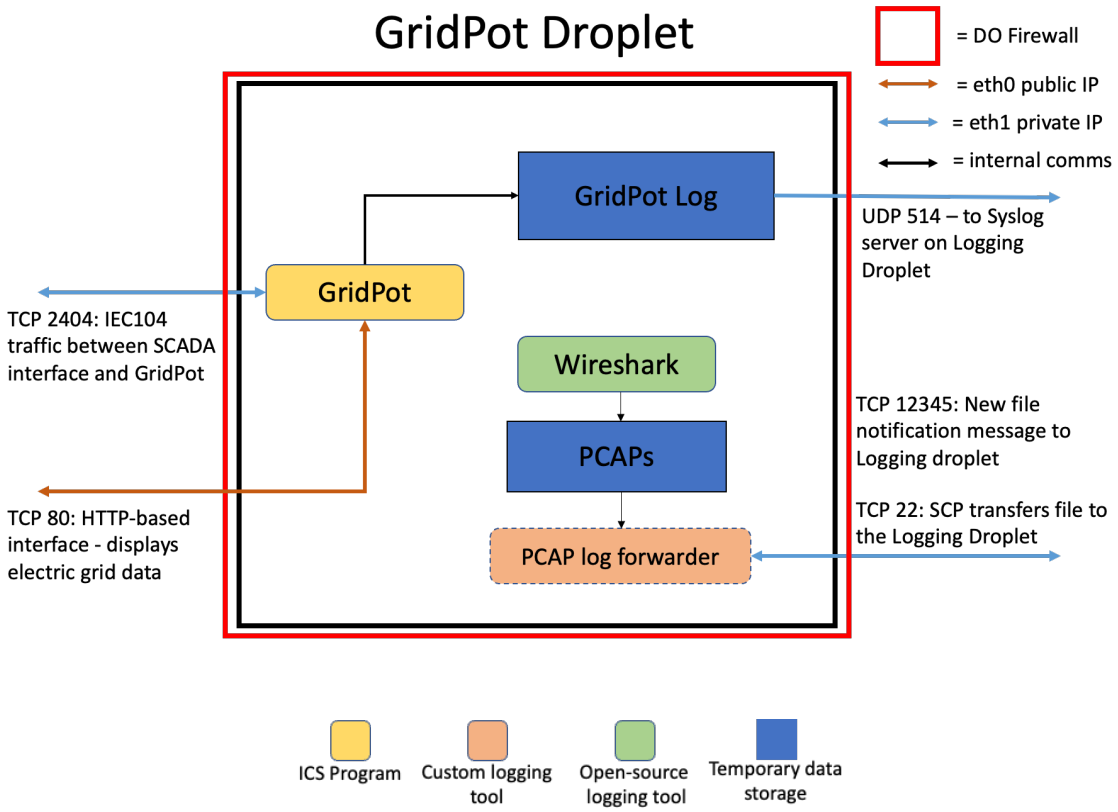


Figure 7. Diagram of GridPot Droplet Used in Experiment 4 with an HTTP Server.

4. Logging Droplet

The Logging Droplet was a general-purpose machine with 8GB of memory and 125GB of secondary storage. Each experiment generated considerable data that needed to be moved to external storage before the next experiment could begin. We used a cloud-based storage service to back up all data after it was collected on the Logging Droplet, to prevent the Droplet's disk from running out of space. This machine collected packets and logs from all data sources in our honeypot (Figure 8). On the private interface, a syslog server listened on port 514 for incoming logs from GridPot, and an NXLog server listened on port 1514 for event logs from the Windows machine. The PCAP manager listened on port 12345 of the private interface for notifications that a new PCAP file was ready to be transferred. When a notification was received, it started an SSH session to copy the PCAP file from the machine that sent the notification.

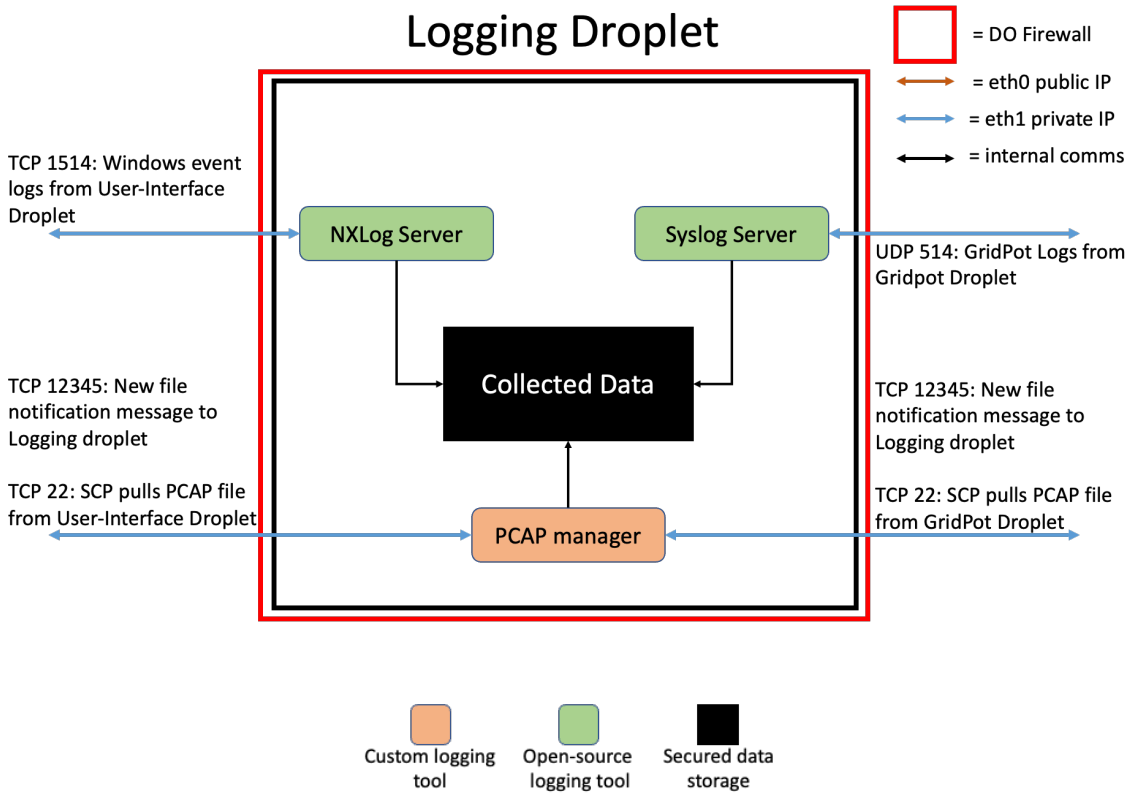


Figure 8. Diagram of Logging Droplet

B. EXPERIMENT IMPLEMENTATION

We ran four experiments with our honeypot deployed in DigitalOcean's cloud environment. The results of each experiment were analyzed to determine the tactics the attackers used. We then adjusted the logging to capture the data to recreate the attacks.

Experiment 1 started November 24, 2021, and collected data for 39 days until January 2, 2022. We expected cyberattacks would be more likely then since many employees, including network-security personnel, take time off during the holidays. During Experiment 1, the RDP certificate on our Windows machine expired. After that, all RDP sessions captured in the PCAP data could not be decrypted, so we attempted a different method for replaying RDP sessions in Experiment 2.

Before Experiment 2, we modified the user interface to include the RDP monitoring tool PyRDP, which could capture and replay RDP sessions. Experiment 2 ran from January

25, 2022, until February 18, 2022. It was stopped when we discovered that with PyRDP running in this "man-in-the-middle" mode, our user interface no longer looked like a Windows 10 machine. Figure 9 shows the view of the NMAP scanner with PyRDP running. The scan should identify the service listening on port 3389, but could not. We concluded that PyRDP handles network traffic in a way that prevents NMAP from correctly identifying its traffic as RDP. Figure 10 shows what the same scan looked like without PyRDP running, in which it correctly identified the service as Microsoft Terminal Services, another name for RDP.

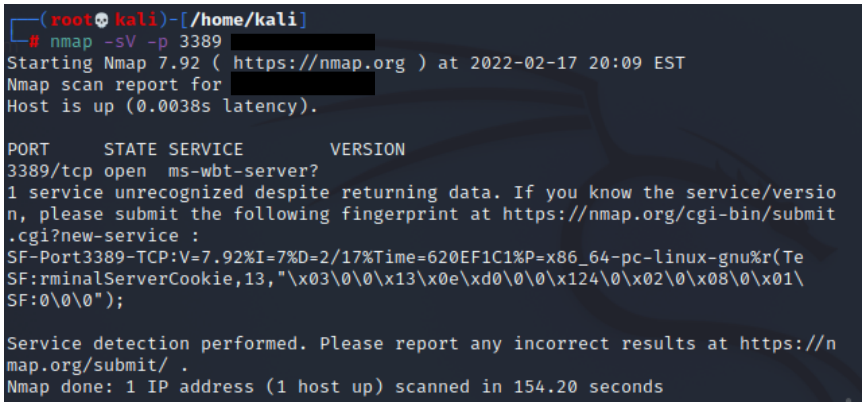


Figure 9. NMAP Service Scan with PyRDP Running.

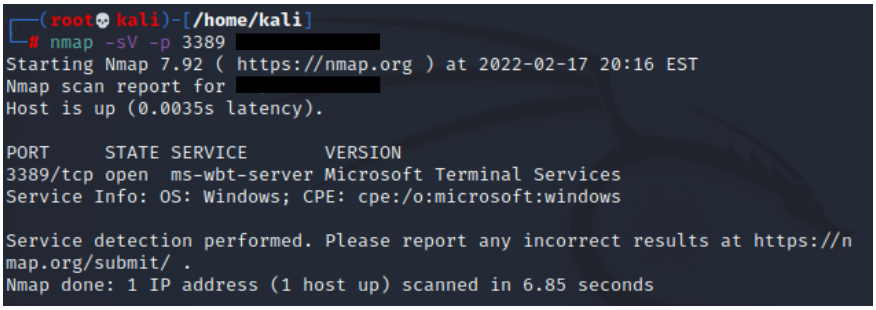


Figure 10. NMAP Service Scan without PyRDP Running.

Experiment 3 ran from February 19, 2022, to March 14, 2022. We reverted to the design used in Experiment 1. This time we ensured the RDP certificate would not expire in the middle of the experiment by renewing the RDP certificate and confirming that it would not expire during the experiment.

To generate interest in the honeypot, we changed the name of the Windows workstation to “HMI-OPERATOR.” When the system is scanned, the RDP certificate will show the computer’s name which will give a clue that it is part of an ICS system. The IndigoSCADA program was also set to automatically run when a user logs in, to make it look more like an ICS system.

Experiment 4 ran from March 15, 2022, to April 12, 2022. We changed the logging mechanisms significantly and gave the user interface a new look. We also moved the user-interface machine to a different IP address so it would appear to be a new system.

We modified the Windows machine’s audit policy for process creation to record the command line for any process created. This would show any options, flags, or arguments passed to a process. We also enabled Script Block Logging for PowerShell to record any script run in a PowerShell process. We also used the Windows logging service Sysmon (Russinovich, 2022). With its default configuration file, Sysmon produced many log entries that were irrelevant to Experiment 4. Hence, we changed the Sysmon configuration file to filter out uninteresting events, but also not create monitoring gaps that an attacker could hide in if they studied the configuration file. We used a configuration file created by SwiftOnSecurity (2021) because it was well-documented and captured events to help us analyze attacks.

We also enabled the GridPot’s HTTP server. External connections to the GridPot Droplet over port 80 displayed a Web page that refreshed every two seconds with updated readings from the electric-grid simulation, GridLab-D. We made this page advertise that it was for an electric-distribution system. The page displayed a fake company name and a message that directed employees to log on to the Windows user interface with RDP at a specific IP address. Figure 11 shows the webpage.

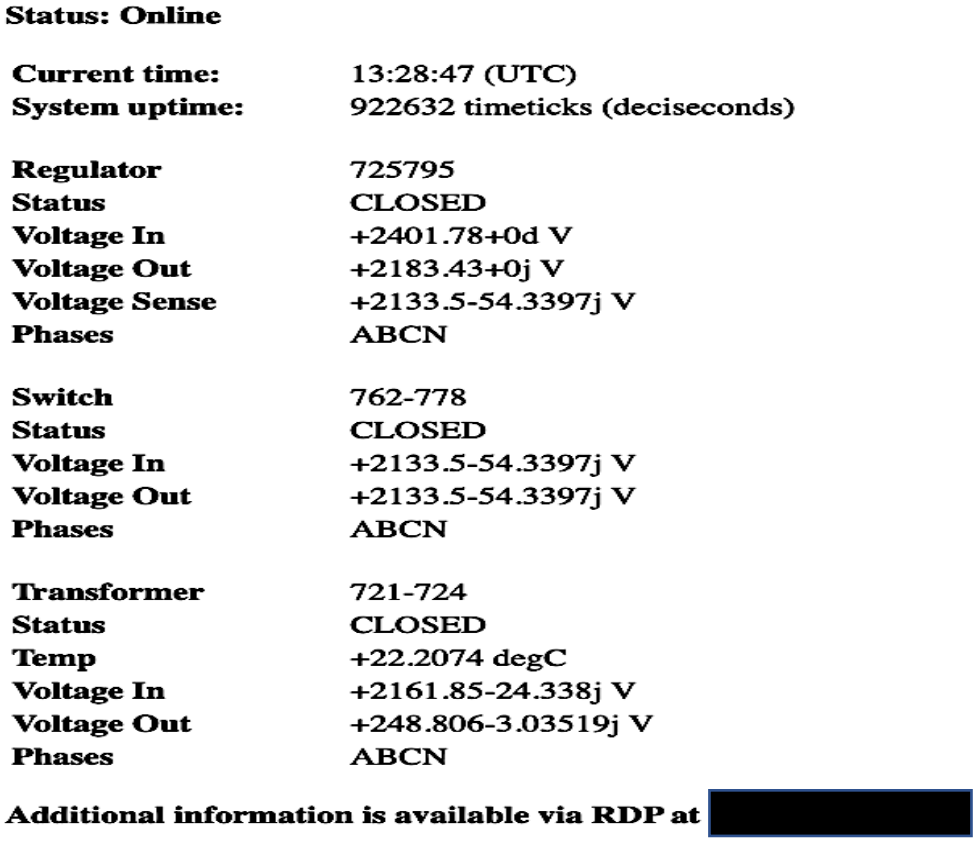


Figure 11. GridPot Webpage Displayed to Attract ICS-specific Attacks.

C. DATA ANALYSIS METHOD

Since honeypots are not production systems, all packets sent to them can be analyzed for interesting data. Windows event logs are not as rich a source of data because, even without user interaction, many routine events are logged. Therefore, we needed a strategy to filter the logs our honeypot produced.

Scanning events were filtered first. RDP connections and logon attempts indicated when attackers and legitimate scanning services were probing the honeypot. Event ID 140 from the Microsoft-Windows-RemoteDesktopServices-RdpCoreTS log records when an RDP logon fails. Its data includes the time of the event, source IP address, and port number. Information from event ID 4625 in the Microsoft-Windows-Security-Auditing log complements event ID 140 by recording the username of the failed logon attempt. Scanning could also be identified in the network traffic captured by Wireshark. Any traffic to the

RDP listening port 3389 that did not result in an RDP connection could be considered scanning.

To find evidence of successful logins to the Windows machine, we first scanned the event logs for event ID 4624 which indicates a user login using RDP. The corresponding logoff event is 4634. We could correlate logon/logoff events with the traffic captured in the PCAP files. Once an RDP session was identified, we could use the RDP server key to decrypt the traffic and convert it to an MP4 video file using the PyRDP tool.

Other events that we identified in the logs were process creation, registry modification, Windows Defender alerts, changes to logging, and network activity. We could use event logs to recreate actions which might be unobservable in the RDP video playback alone. For example, installation and running of a malicious program that installed a backdoor on the system would not be shown on the computer's display. We must use the Windows event logs to find such activity.

After each experiment, the collected data were analyzed for attack patterns and techniques. Data captured from Wireshark on the public interface of the User-Interface Droplet contained information about how attackers accessed the Windows user interface over RDP. With the decrypted RDP packet captures, the PyRDP tool can replay the RDP sessions. Watching the replay can give insights into what actions the attacker took, since attackers can run scripts whose functions may not be visible in the RDP session. Data from the event logs will show scripts they executed, the processes they started, and the connections they made. Finally, the GridPot logs can be analyzed to look for IEC104 traffic between the Windows user interface and GridPot. When someone logs into the IndigoSCADA program running on the Windows workstation, GridPot will start sending IEC104 messages that update the values of the electrical components simulated by GridLab-D. If we see any IEC104 traffic, we know that an attacker has logged into the Windows workstation and started the IndigoSCADA program. We can use this data to harden the host machine against attacks.

THIS PAGE INTENTIONALLY LEFT BLANK

V. RESULTS AND DISCUSSION

A. EXPERIMENT 1 RESULTS

1. External Scanning Observed and Login Attempts

Experiment 1 ran from November 24, 2021, to January 2, 2022. We considered scanning to be any network traffic to the RDP listening port on our User-Interface Droplet. Port scanning of our honeypot started only ten minutes after the DigitalOcean firewall allowed network traffic on port 3389. Our IP address was not advertised anywhere, so the scanning suggested bots that search the Internet looking for publicly-accessible services were used. 1,046,198 TCP connections were made to port 3389 of the User-Interface Droplet. Traffic from 54 countries was observed; Figure 12 shows the top 20 countries that made TCP connections to the User-Interface Droplet. Russian IP addresses made the most connections, with the United States and Netherlands second and third. Once attackers identified an RDP service was running on our User-Interface Droplet, they appeared to use a New Technology LAN Manager (NTLM) brute-force attack to guess the usernames and passwords of accounts on the Windows 10 virtual machine. NTLM is a protocol used in Windows software to authenticate users (Microsoft, 2022a).

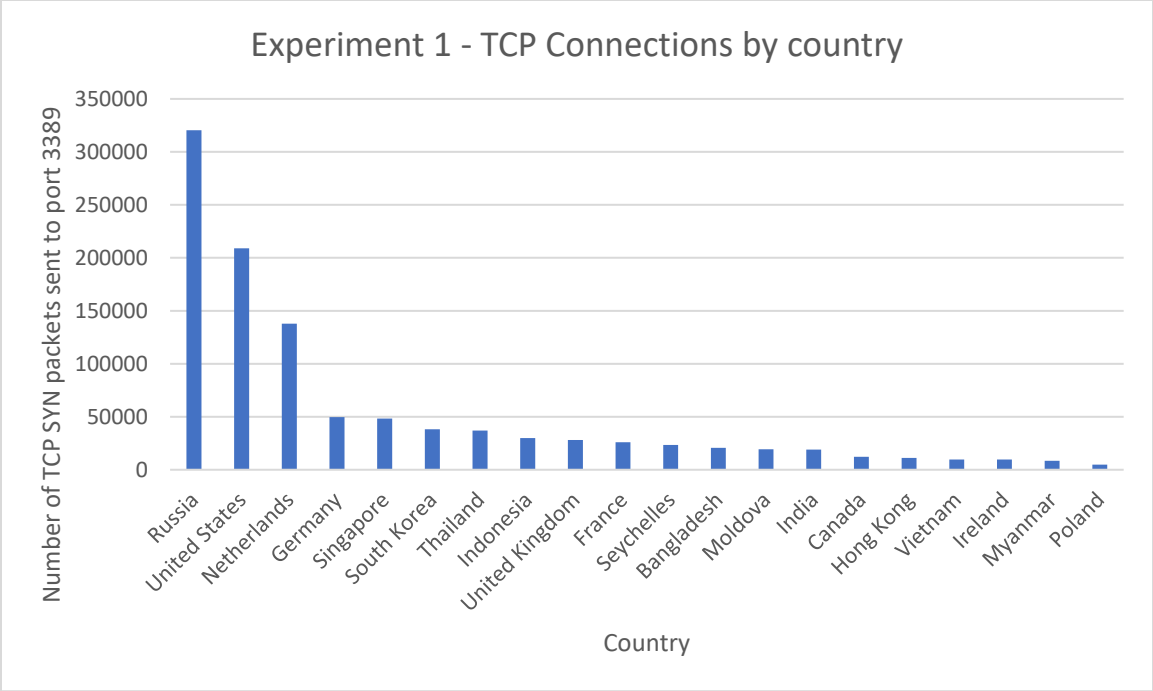
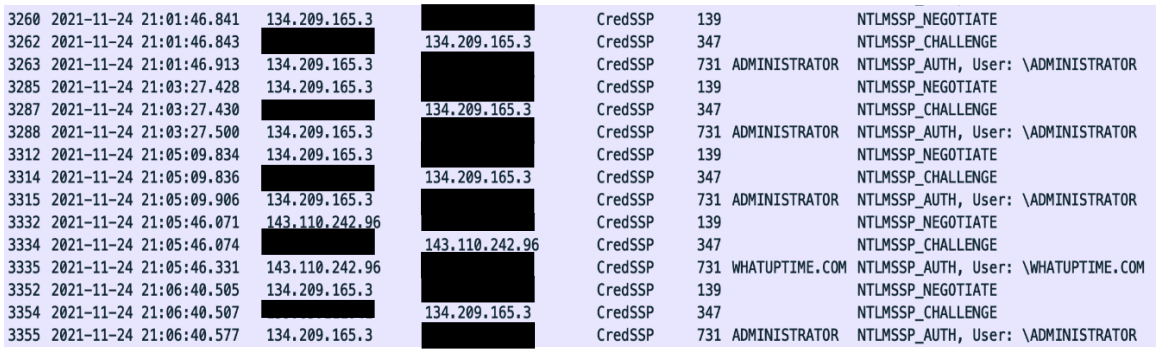


Figure 12. Experiment 1 – Graph of TCP Scans on Port 3389 (RDP).

Attackers began by guessing account names on the Windows machine. This behavior was logged in both the packets sent to the User-Interface Droplet and the Windows event logs. Every time the attacker guessed an account name, an event was logged on the Windows machine. Event ID 4625 ("Account failed to logon") logged the username the attacker unsuccessfully guessed and the authentication method, which was NTLM here. In the packets this behavior could be found in the NTLMSSP messages.

Figure 13 shows sample log data for this attack. The attacker starts the connection with a NTLMSSP_NEGOTIATE message, and the server responds by sending a randomly generated value in the NTLMSSP_CHALLENGE message. The attacker then encrypts the randomly generated value with a guessed password and sends the encrypted output back to the Windows machine with a username, most commonly "Administrator," in a NTLMSSP_AUTH message. The server looks up the password for the client-supplied username and performs the same encryption on the randomly generated value. If the encrypted message that the attacker sent was what the server generated, then the attacker

is authenticated. If either the password is incorrect or the account does not exist, the server does not respond.



3260	2021-11-24 21:01:46.841	134.209.165.3	[redacted]	CredSSP	139		NTLMSSP_NEGOTIATE
3262	2021-11-24 21:01:46.843	[redacted]	134.209.165.3	CredSSP	347		NTLMSSP_CHALLENGE
3263	2021-11-24 21:01:46.913	134.209.165.3	[redacted]	CredSSP	731	ADMINISTRATOR	NTLMSSP_AUTH, User: \ADMINISTRATOR
3285	2021-11-24 21:03:27.428	134.209.165.3	[redacted]	CredSSP	139		NTLMSSP_NEGOTIATE
3287	2021-11-24 21:03:27.430	[redacted]	134.209.165.3	CredSSP	347		NTLMSSP_CHALLENGE
3288	2021-11-24 21:03:27.500	134.209.165.3	[redacted]	CredSSP	731	ADMINISTRATOR	NTLMSSP_AUTH, User: \ADMINISTRATOR
3312	2021-11-24 21:05:09.834	134.209.165.3	[redacted]	CredSSP	139		NTLMSSP_NEGOTIATE
3314	2021-11-24 21:05:09.836	[redacted]	134.209.165.3	CredSSP	347		NTLMSSP_CHALLENGE
3315	2021-11-24 21:05:09.906	134.209.165.3	[redacted]	CredSSP	731	ADMINISTRATOR	NTLMSSP_AUTH, User: \ADMINISTRATOR
3332	2021-11-24 21:05:46.071	143.110.242.96	[redacted]	CredSSP	139		NTLMSSP_NEGOTIATE
3334	2021-11-24 21:05:46.074	[redacted]	143.110.242.96	CredSSP	347		NTLMSSP_CHALLENGE
3335	2021-11-24 21:05:46.331	143.110.242.96	[redacted]	CredSSP	731	WHATUPTIME.COM	NTLMSSP_AUTH, User: \WHATUPTIME.COM
3352	2021-11-24 21:06:40.505	134.209.165.3	[redacted]	CredSSP	139		NTLMSSP_NEGOTIATE
3354	2021-11-24 21:06:40.507	[redacted]	134.209.165.3	CredSSP	347		NTLMSSP_CHALLENGE
3355	2021-11-24 21:06:40.577	134.209.165.3	[redacted]	CredSSP	731	ADMINISTRATOR	NTLMSSP_AUTH, User: \ADMINISTRATOR

Figure 13. NTLM Brute-Force Attack in Network Traffic.

Most account names guessed were for the administrator accounts. Table 1 shows the top ten usernames guessed in brute-force attacks. The username "WHATUPTIME.COM" was one of the top five most-guessed usernames for all experiments. This is the default username for a version of Windows created specifically to run on DigitalOcean's platform. This indicates that attackers guessed our honeypot was a Windows machine and recognized the IP address as owned by DigitalOcean.

Table 1. Experiment 1 – Top Ten Most-Guessed Usernames.

Experiment 1	
Count observed	**Username**
396882	ADMINISTRATOR
32306	administrator
27803	ADMIN
11247	USER
8957	WHATUPTIME.COM
2280	Administrator
2123	TINHOCTHUCHANH
1707	MINER
1677	CHIA
1529	TEST

The username "TINHOCTHUCHANH" was guessed 2123 times. It is a Vietnamese phrase that, according to Google Translate, means "practical informatics." Since many Vietnamese IP addresses scanned our honeypot, we wanted to find out if that username was correlated with scanning from those addresses. We filtered our data to find the countries associated with the IP addresses that guessed that username. Table 2 shows the results of this search. It turned out that no Vietnam-based scanning used that username.

Table 2. Countries that Guessed Username “TINHOCTHUCHANH”

Count Observed	Country
741	Singapore
585	India
480	Bangladesh
151	United States
82	United Arab Emirates
25	United Kingdom
24	Hungary
12	Brazil
10	Germany
8	Hong Kong
5	Canada

2. Post-login Activities

At least 13 logins to the remotely accessible account on the Windows machine occurred. User activity was also observed in the event logs without a corresponding login event, so either some logins were not logged or deleted by the attacker. Table 3 shows the times of the recorded logins and durations of the RDP sessions. Noticeably, all logins occurred between the holidays of Christmas and New Year’s Day, a period when many employees take leave from their jobs. Most were started from an IP address in the United States. The remainder had significantly shorter RDP sessions lasting two minutes or less. Only one instance of interaction with the SCADA program occurred. In this experiment, the SCADA program did not automatically start when a user logged in, but an icon on the desktop needed to be clicked to open it. This was marked in the Windows event logs with Event ID 4688 indicating that the IndigoSCADA process was created. The attacker did not interact with the program beyond launching it and appeared to log out shortly after the program was launched. We characterized the attackers’ actions for each session in the “Interaction” column in Table 3. Recon refers to actions taken by the attacker to gather information about their target like active services, IP addresses, or finding other machines in the same network as the target. If any interaction with the SCADA software was

observed, we also made a note in the table. Interactions marked with “Firefox” related to an attack involving the Firefox Web browser.

Table 3. Experiment 1 – Successful Logins to the Windows Interface.

Time (PST)	Length of RDP Session (minutes)	Source IP Address	Source Country	Interaction if any	Source Local Time
12/27/21 9:15	<1	87.251.64.137	Russia	Recon	19:15
12/27/21 10:28	2	146.0.40.37	Germany	Firefox	19:28
12/28/21 7:19	34	45.130.83.24	United States	Firefox	07:19
12/28/21 9:42	17	45.130.83.150	United States	Firefox	09:42
12/28/21 14:07	43	45.130.83.24	United States	Firefox	14:07
12/28/21 16:52	<1	45.227.254.118	Belize		17:52
12/29/21 2:56	1	77.83.36.32	Ukraine	SCADA	12:56
12/29/21 16:52	8	154.6.16.155	United States		16:52
12/29/21 17:12	21	154.6.16.155	United States	Firefox	17:12
12/29/21 17:43	31	198.255.5.170	United States		17:43
12/30/21 11:06	3	45.130.83.150	United States	Firefox	11:06
12/30/21 13:41	1	185.191.32.160	Russia		01:41
12/31/21 12:51	46	45.130.83.145	United States	Firefox	12:51

3. Malicious Actions in the Windows 10 Virtual Machine

Before the first login noted in Table 3, user activity occurred on December 26 between 3:53AM and 3:56AM PST. A program *Advanced_IP_Scanner.exe* was executed, but no other events indicated what it did. The program was executed from the C:\Users\<username>\AppData folder. No events indicated how the program got on the machine. There also was no RDP traffic recorded during this time which might mean an

attacker logged on at an even earlier time and dropped a program or script to execute the Advanced IP Scanner tool. This attacker likely evaded our logging system by either being careful not to trigger certain events or erasing logs that they knew would be recorded.

The next attack happened the following day, December 27, at 10:28AM PST. We recreated the attack using data collected from the Windows event logs and PCAPs (Appendix D). An RDP session began, and the attacker dropped a program *c.exe* on the desktop of the Windows machine which started a chain of events. The program made an HTTP GET request to codeproject.com to download a Firefox installer program. It then made two near-simultaneous connections to websites, icanhazip.com and ipinfo.io. Both websites can track machines already infected. After *c.exe* received responses from both websites, the Firefox installer program downloaded the Firefox Web browser, and the attacker logged off the machine.

Over the next four days, several similar RDP sessions were started from an IP address in the United States. They all used the Firefox browser to access travel websites including mytrip.com, kayak.com, and destinationholidays.com. These sites may have been hosting adware to which the version of Firefox was vulnerable. After Experiment 1 ended, we took a snapshot of the Windows machine for analysis and opened the Firefox browser while running Process Explorer, a Sysinternals tool, to see if it spawned other processes, but we did not observe anything suspicious. We also submitted the SHA256 hash to VirusTotal.com and it identified it as a legitimate Firefox browser. Further analysis is needed to determine the role of the Firefox browser in this attack.

B. EXPERIMENT 2 RESULTS

1. External Scanning Observed

Experiment 2 ran from January 25, 2022, to February 18, 2022. While it collected data for 24 days, it only saw 67,930 TCP connections to port 3389. For comparison, Experiment 3 ran for the same number of days but saw 16 times more connections to the RDP port. Scanning was significantly reduced in Experiment 2. We attribute this to PyRDP changing the way our honeypot responded to RDP traffic to make it less clear that it was a Windows 10 machine with RDP enabled. Figure 14 shows the number of TCP connections

to the RDP listening port on the User-Interface Droplet. PyRDP may have given clues that the machine was a honeypot. Nothing tried to log in to the Windows machine.

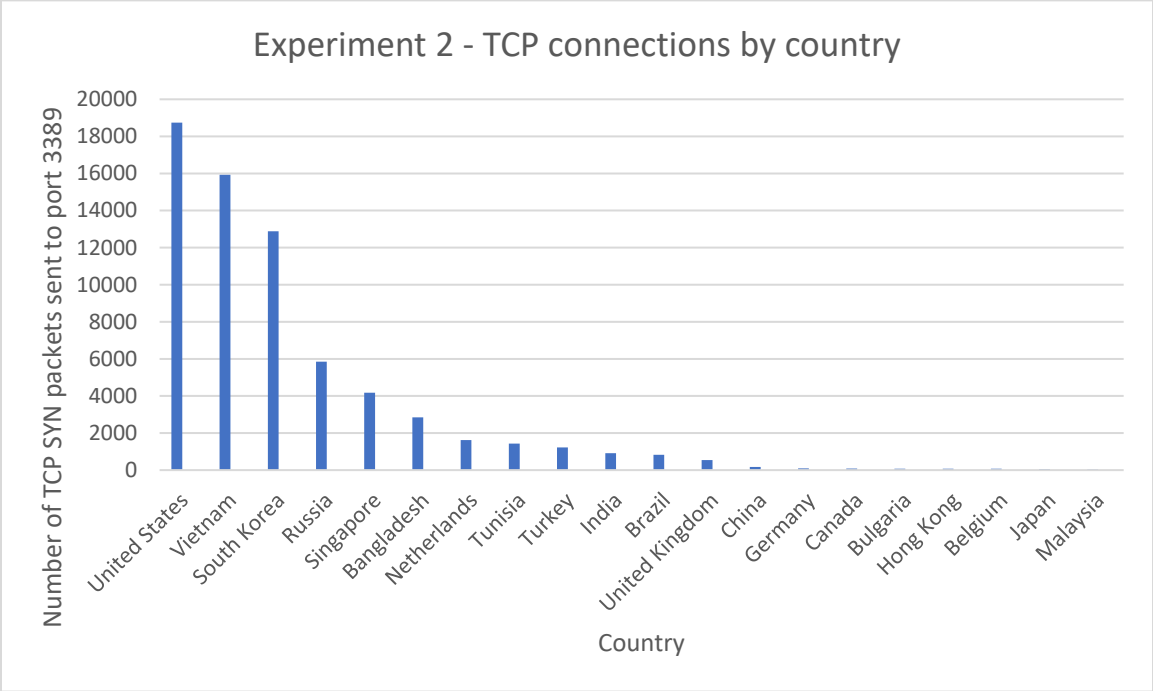


Figure 14. Experiment 2 – Graph of TCP Scans on Port 3389 (RDP).

We still observed scanning on port 3389 in the PyRDP logs and attempts to connect to the MS_T120 channel during the channel-connection phase of the RDP sequence. PyRDP logged these events as possible BlueKeep exploits since the MS_T120 channel relates to the exploit (Yen, 2019). Since Windows 10 is not vulnerable to the BlueKeep exploit, the attempts were likely automated and only directed at our honeypot because port 3389 was exposed. Log entries from PyRDP showing detections of the BlueKeep exploit are in Appendix E.

C. EXPERIMENT 3 RESULTS

1. External Scanning Observed and Login Attempts

Experiment 3 ran from February 19, 2022, to March 14, 2022. Before running Experiment 3 we reverted our honeypot design to the Experiment 1 configuration without

PyRDP. We saw scanning return to the same levels observed in Experiment 1, which indicated that PyRDP alone made our system less attractive to intruders. A total of 1,056,909 were made to port 3389 of the User-Interface Droplet. Figure 15 shows the breakdown of countries observed. Most traffic came from China, which was not even in the top 20 during Experiment 1.

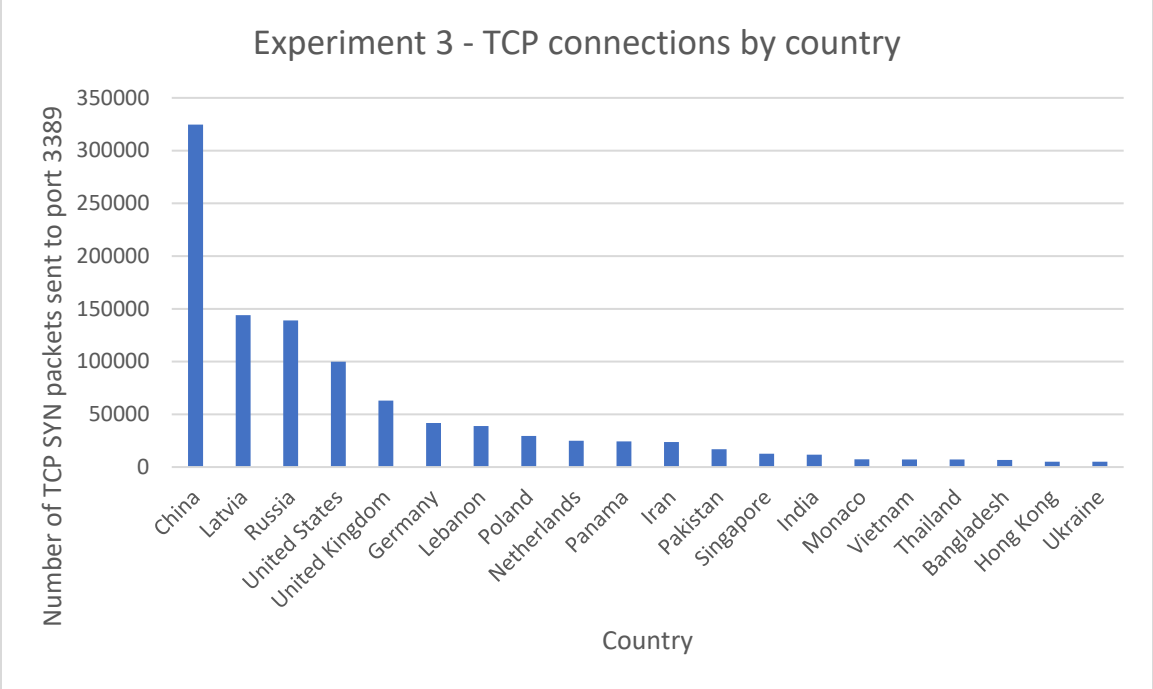


Figure 15. Experiment 3 – Graph of TCP Scans on Port 3389 (RDP).

We also observed more NTLM brute-force guessing of account usernames on the Windows machine. Table 4 shows the ten most-guessed usernames. The username "WHATUPTIME.COM" saw the second most guesses with about a 4-fold relative increase compared to "ADMINISTRATOR" than that in Experiment 1. This suggests that more attackers knew this machine was a Windows virtual machine running in DigitalOcean. Also, since we changed the name of the Windows machine to "HMI-OPERATOR" before Experiment 3, we saw "HMI-OPERATOR", "HMI", and "OPERATOR" among the top ten most-guessed usernames, which could mean attackers used a more intelligent method for selecting usernames than in Experiment 1.

Table 4. Experiment 3 – Top Ten Most-guessed Usernames

Experiment 3	
Count observed	**Username**
511486	ADMINISTRATOR
156302	WHATUPTIME.COM
59417	Administrator
44569	ADMIN
9339	HMI-OPERATOR
7064	Administrador
6461	OPERATOR
6209	HMI
6184	Administrateur
4858	USER

2.

3. Post-login Activities

Despite much scanning, attackers interacted very little with the Windows machine. As shown in Table 5, only eight logins succeeded over RDP and all but one was less than a minute long. The long session lasted for 19 minutes, but the event logs did not show the attacker starting the SCADA interface or any other process. The RDP sessions were likely automated to determine if it was possible to log in. We concluded that attackers lacked interest in our honeypot; running PyRDP in Experiment 2 might have suggested it was a honeypot.

Table 5. Experiment 3 – Successful Logins to the Windows Interface

Time	Length of RDP Session (minutes)	IP Address	Country	Interaction if any	Source Local Time
2/20/22 3:48	<1	82.102.22.17	Norway		12:48
2/20/22 5:19	19	82.102.22.17	Norway		14:19
2/21/22 0:19	<1	185.220.101.62	Germany		09:19
2/23/22 0:38	<1	87.251.64.20	Russia		10:38
2/27/22 0:01	<1	87.251.64.20	Russia		10:01
3/7/22 15:39	<1	91.213.50.223	Russia		01:39
3/7/22 17:44	<1	185.176.222.106	Latvia		03:44
3/8/22 3:55	<1	37.120.152.196	Bulgaria		13:55

D. EXPERIMENT 4 RESULTS

Before we ran Experiment 4, we moved our User-Interface Droplet to a different IP address, and changed the name of the Windows machine and user accounts to make it appear to be a new system. Experiment 4 ran for 28 days from March 15, 2022, to April 12, 2022.

1. External Scanning Observed and Login Attempts

The amount of scanning on the RDP listening port was consistent with the experiments without PyRDP. Of the 986,001 TCP connections made to port 3389 of the User-Interface Droplet, 67% came from alleged Russian IP addresses. Their level of scanning dwarfed that of all other countries and could relate to the ongoing conflict in Ukraine. Figure 16 shows the scanning activity for Experiment 4.

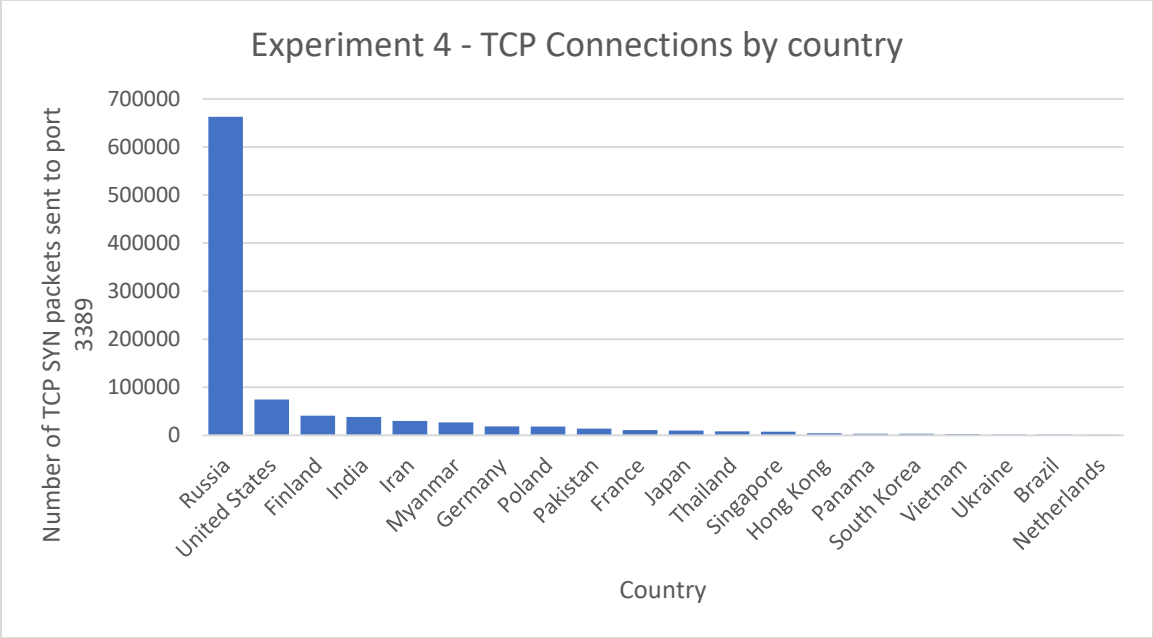


Figure 16. Experiment 4 – Graph of TCP Scans on Port 3389 (RDP).

The "ADMINISTRATOR" account was still the most guessed username and "WHATUPTIME.COM" was in the top five again, showing that some attackers were aware we had a virtual machine running on a DigitalOcean server (Table 6).

Table 6. Experiment 4 – Top Ten Most-Guessed Usernames

Experiment 4	
Count observed	**Username**
193243	ADMINISTRATOR
44622	administrator
33276	Administrator
22049	WHATUPTIME.COM
14642	ADMIN
9151	USER
5586	Administrador
5188	Administrateur
1534	null
1401	WINADMIN

2. Post-login Activities

Table 7 shows all successful logins during Experiment 4. More logins occurred during Experiment 4 than in the previous experiments, although most of them were short (1 minute or less). Only one RDP session was longer than two minutes, that on March 22, 2022, at 10:20AM PST. Unique to this experiment was the high number of logins from IP addresses that appeared to originate from Ukraine. Also, all RDP sessions from alleged Ukrainian IP addresses began after 6:00PM local time. These logins may have related to the conflict between Ukraine and Russia and the cyberwarfare then happening. Two factors could have caused greater activity on our honeypot. First, the webpage from GridPot's HTTP server revealed the IP address of our SCADA interface, and that it could be accessed using RDP. Second, we used "User" as the username for the Windows machine, a top-ten most-guessed usernames in previous experiments.

Table 7. Experiment 4 – Successful Logins to the Windows Interface.

Time (PST)	Duration of RDP session (minutes)	Source IP Address	Source Country	Interaction if any	Source Local Time
3/18/22 10:02	<1	31.43.185.9	Ukraine		20:02
3/18/22 22:38	<1	185.56.150.158	Germany		07:38
3/18/22 22:38	2	5.114.247.195	Iran		10:08
3/18/22 22:41	<1	185.56.150.158	Germany		07:41
3/19/22 10:21	<1	31.43.185.9	Ukraine		20:21
3/19/22 10:52	<1	31.43.185.9	Ukraine		20:52
3/20/22 8:45	<1	94.232.42.186	Russia		18:45
3/21/22 8:53	<1	31.43.185.9	Ukraine		18:53
3/21/22 9:33	<1	51.195.189.7	France		18:33
3/21/22 9:36	1	5.120.227.231	Iran	SCADA	21:06
3/22/22 9:29	<1	87.251.64.26	Russia		19:29
3/22/22 10:20	6	159.242.234.121	Germany	Recon	18:20
3/22/22 11:07	<1	31.43.185.9	Ukraine		21:07
3/23/22 8:52	1	31.43.185.9	Ukraine		18:52
3/24/22 9:01	<1	31.43.185.9	Ukraine		19:01
3/27/22 5:25	<1	43.245.46.142	Myanmar		18:55
3/27/22 7:42	2	5.119.130.106	Iran		19:42
3/29/22 8:09	1	31.43.185.9	Ukraine		18:09
4/1/22 11:52	<1	58.186.205.49	Vietnam		01:52
4/1/22 14:54	1	5.190.75.174	Iran		02:24

3. Malicious Actions in the Windows 10 Virtual Machine

Although more logins occurred during this experiment, we did not observe as much interaction with the Windows virtual machine as in Experiment 1. In one RDP session, on March 21, 2022, at 9:36AM PST, the attacker opened the SCADA program but did not interact with it. An attacker apparently did reconnaissance on March 22 at 10:20AM PST. They downloaded the Advanced Port Scanner tool using Microsoft Edge and used it for a limited scan of the local network on which the Windows virtual machine resided. Table 8 shows the addresses and ports that the attacker saw. They first scanned the target host to

find other running services not remotely accessible. Then they sequentially scanned the subnet starting at the private address 10.0.2.2, stopping at 10.0.2.4. With NAT networking mode in VirtualBox, the gateway router defaults to 10.0.2.2 and the virtual machine is assigned 10.0.2.15. The IP addresses that the attacker scanned indicate that they may have known this was a virtual machine running in VirtualBox. Some ports scanned were for common services, like SSH (port 22), SNMP (port 161), and IPP (631). Other scanning looked for other machines that were running RDP (port 3389) or VNC (port 5901). Finally, they scanned port 80 of IP address 188.40.30.100, an IP address in Germany that hosts the advanced-port-scanner.com domain. This behavior indicates that the attacker was trying to discover targets to move laterally in the network.

Table 8. Summary of Scans by Advanced Port Scanner Tool.

IP Address	**Port**
10.0.2.15 (Host)	135, 139, 445, 3389, 5040, 5100, 6102, 6103, 6104, 6105
10.0.2.2, 10.0.2.3, 10.0.2.4	161, 22, 631, 3389, 5901
188.40.30.100	80

Processor use of the User-Interface Droplet suddenly increased in the middle of Experiment 4. Typically, during the previous experiments when the honeypot was operational, it was around 30%, but on April 1 it jumped to about 53% and remained at that level until April 11. The sudden increase coincided with the last login to the Windows virtual machine. We did not see any user activity in the event logs that would explain such a large jump in processor use. On April 11 the Windows virtual machine crashed and reverted to the snapshot taken just before beginning Experiment 4. We have been unable to confirm the cause of the crash, but we have several hypotheses.

One hypothesis is that an attacker logged in to the Windows virtual machine on April 1, installed malware for processor-intensive tasks such as cryptocurrency mining, and disabled remote access to everyone but themselves. They also would have had to tamper with the logging policies to prevent their actions from being logged, i.e., absent records of this activity in the event logs. The cryptocurrency mining software could explain the observed large spike in processor use on April 1. Further, it appears they also disabled remote access to the Windows virtual machine so no one would interrupt their activities. This would explain why logins suddenly stopped.

Another hypothesis is that the crash related to a well-publicized attack called Sandworm by a Russian hacker group (ESET Research, 2022). The attack targeted Windows machines in an ICS network owned by a Ukrainian electric-grid operator. Ukraine claimed to have thwarted the attack, but the campaign could have spread to ICS networks outside of Ukraine including our honeypot. The timelines for the Russian attacks match what we saw on our honeypot. Once the Russian attack on Ukraine's power grid was thwarted, the attackers that accessed our honeypot may have reverted our Windows virtual machine back to an earlier snapshot before they attacked it to conceal their activity. The scan from the Advanced Port Scanner tool suggests that the attacker determined our Windows machine was running in VirtualBox, since the IP addresses they scanned were default settings for using network-address translation (NAT) in VirtualBox. An advanced attacker could have used this information to exploit the VirtualBox hypervisor to revert the virtual machine to an earlier snapshot. Further analysis of our data is needed to find evidence for this theory.

Once our Windows virtual machine reverted to an earlier snapshot, our honeypot was no longer remotely accessible. The snapshot was created before the user account that could use RDP was created. Nothing in the event logs indicated that the user account was deleted, just that a system reboot had occurred. We also noticed that when the virtual machine rebooted, the first several event logs that were sent to the Logging Droplet had a timestamp of March 15. That was the date of the last snapshot we took of the virtual machine before it was deployed. After Windows booted up, it corrected its time, and subsequent events were logged with the correct date of April 11.

E. DISCUSSION

Figure 17 shows the first three experiments had somewhat random country distributions of scans, with the most-common country around a third of the total scans observed. Experiment 4 was different in that 67% of scanning originated from IP addresses that appear to be from Russia in support of their invasion of Ukraine then. It also may have been preparation for the attack on Ukraine's electric grids that targeted Windows machines in ICS systems (ESET Research, 2022). Our honeypot emulated the same kind of systems that were targeted by that cyberattack.

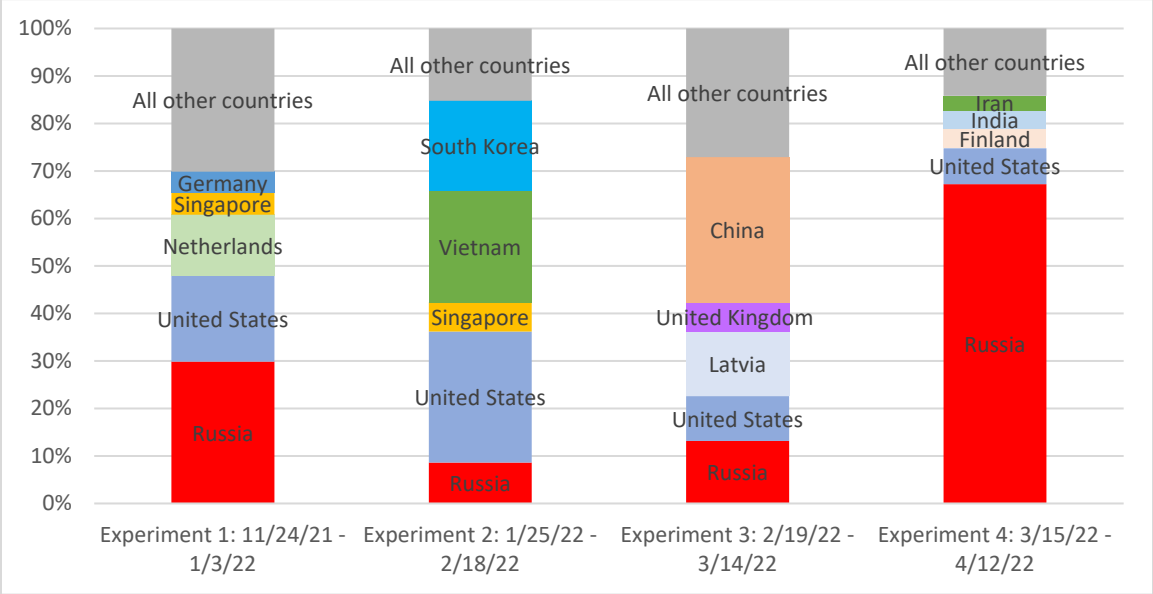


Figure 17. Percentage of Scanning by Country for Each Experiment.

Another feature of our data we analyzed was the local time when they started an RDP session with our honeypot since that let us conclude something about the type of attacker. Logins during typical working hours would more likely come from a professional organization such as a state-sponsored cyber group. All local times of the RDP sessions are in Table 9. Experiment 2 data was excluded from the Table 9 because no logins were observed. We could not discern any timing patterns that would indicate professional activity after analyzing the data, as the local times of the RDP sessions were evenly spread throughout the day. Furthermore, we could not be certain that the attackers did not use

software or other techniques to make their IP address appear to be in country other than their true location. In fact, it is likely that an attacker would use a tool to change their IP address to conceal their identity.

Table 9. Local Times of the Alleged Countries That Started RDP Sessions.

Experiment 1			Experiment 3			Experiment 4		
Time (PST)	**Alleged Source Country**	**Source Local Time**	**Time (PST)**	**Alleged Source Country**	**Source Local Time**	**Time (PST)**	**Alleged Source Country**	**Source Local Time**
12/27/21 9:15	Russia	19:15	2/20/22 3:48	Norway	12:48	3/18/22 10:02	Ukraine	20:02
12/27/21 10:28	Germany	19:28	2/20/22 5:19	Norway	14:19	3/18/22 22:38	Iran	10:08
12/28/21 7:19	United States	7:19	2/21/22 0:19	Germany	9:19	3/18/22 22:38	Germany	7:38
12/28/21 9:42	United States	9:42	2/23/22 0:38	Russia	10:38	3/18/22 22:41	Germany	7:41
12/28/21 14:07	United States	14:07	2/27/22 0:01	Russia	10:01	3/19/22 10:21	Ukraine	20:21
12/28/21 16:52	Belize	17:52	3/7/22 15:39	Russia	1:39	3/19/22 10:52	Ukraine	20:52
12/29/21 2:56	Ukraine	12:56	3/7/22 17:44	Latvia	3:44	3/20/22 8:45	Russia	18:45
12/29/21 16:52	United States	16:52	3/8/22 3:55	Bulgaria	13:55	3/21/22 8:53	Ukraine	18:53
12/29/21 17:12	United States	17:12				3/21/22 9:33	France	18:33
12/29/21 17:43	United States	17:43				3/21/22 9:36	Iran	21:06
12/30/21 11:06	United States	11:06				3/22/22 9:29	Russia	19:29
12/30/21 13:41	Russia	1:41				3/22/22 10:20	Germany	18:20
12/31/21 12:51	United States	12:51				3/22/22 11:07	Ukraine	21:07
						3/23/22 8:52	Ukraine	18:52
						3/24/22 9:01	Ukraine	19:01
						3/27/22 5:25	Myanmar	18:55
						3/27/22 7:42	Iran	19:42
						3/29/22 8:09	Ukraine	18:09
						4/1/22 11:52	Vietnam	1:52
						4/1/22 14:54	Iran	2:24

The logging system we implemented on our honeypot preserved data that would have been lost during Experiment 4. Had NXLog not been backing up the event logs from the Windows machine to our logging server, we would have lost all event-log data when the virtual machine reverted to a snapshot taken before Experiment 4 began. We can also confidently say that no data was lost from the PCAP dataset; Wireshark was not interrupted during any of the four experiments and all PCAP files were forwarded to the logging server. Table 10 shows a breakdown of the total size of both the Event Log and PCAP datasets for each experiment. Experiment 2 produced the least data since it lacked logins to the Windows machine. Experiment 4's increase in data was because with Sysmon running on the Windows machine, more events were being logged, as well due to, allegedly, increased Russian cyber activities related to Ukraine.

Table 10. Dataset Size Comparison across All Experiments

	File Size (GB)		
Experiment	**Event Log**	**PCAP**	**Experiment Duration (Days)**
1	26.08	11.52	39
2	3.85	6.34	24
3	19.52	5.69	24
4	27.86	12.21	28

In Experiment 4, we enabled GridPot's HTTP server. The server displayed a webpage that contained the IP address of our User-Interface Droplet and instructed users to access it using RDP. We wanted to determine if the information displayed on the webpage caused more traffic to the User-Interface Droplet. We extracted all IP addresses that sent an HTTP GET request to retrieve the GridPot interface webpage and compared them against IP addresses that started a TCP connection to the RDP listening port of the User-Interface Droplet. Of the approximately 1700 unique IP addresses that established connections to either of the two machines, only 21 IP addresses visited both GridPot's HTTP server and the User-Interface Droplet. Further analysis showed that 15 of the 21 IP

addresses visited the HTTP server before visiting the User-Interface Droplet. Time between visits ranged from 5 hours to 15 days. Also, no IP addresses tried to make an RDP connection with the Windows machine. They only started a TCP connection to the RDP port which indicates that they were only scanning the machine to find an open port. Evidence was insufficient to conclude that enabling the HTTP server caused an increase of traffic to the User-Interface Droplet. However, it is possible that once an attacker gathered information from the HTTP server, they used a machine with a different IP address to connect to the User-Interface Droplet. In that case, we could not correlate those two connections.

VI. CONCLUSION AND FUTURE WORK

A. SUMMARY OF FINDINGS

Our honeypot was sufficiently realistic to entice professional attackers to interact with it. Each experiment we ran improved the honeypot's logging. Logging was hardened to withstand attacks and still provide enough data to determine attack patterns and behaviors. Despite our efforts to draw attention to the ICS functions of our honeypot, intruders were more interested in attacking the Windows machine than the ICS functions.

In Experiment 1, an attacker logged into the Windows machine and installed an IP-scanning and port-scanning tool to scan the network for potential targets to move laterally. Another attacker dropped an executable on the desktop of the Windows machine that downloaded the Firefox Web browser. Subsequent logins to the Windows machine launched the Firefox browser and visited travel-planning websites. We suspect those websites contained adware controlled by the attacker.

Although we did not capture many interesting attacks during Experiment 2, we discovered that PyRDP, while useful for analyzing RDP sessions, allows attackers to identify honeypots. We also noticed attempts to scan our Windows machine for the BlueKeep vulnerability. Without PyRDP running in Experiment 3, we observed a similar amount of scanning as in Experiment 1, but attackers did not interact with the Windows machine even when they logged in. The IP address of our system might have been flagged either as uninteresting because of its protocols or as a honeypot.

During Experiment 4 we observed an attack that may have related to a Russian attack on Ukraine's electric grid. On the same day and at nearly the same time of the Russian attack, our Windows virtual machine crashed and reverted to a snapshot that was taken before Experiment 4 began; data on the Windows machine was erased. We could not determine the cause of the crash, but the timing and rarity of such an event under other conditions was suspicious.

B. FUTURE WORK

More analysis must be done on our data because finding relevant data in Windows event logs can be a challenge with the many events that get logged. A more focused subset of events should be analyzed to make recreating attacks easier.

We recommend making an ICS-honeypot environment more restrictive so attackers are more likely to interact with the ICS functions of the honeypot rather than the Windows machine itself. Windows has a "kiosk mode" that only allows one application to be run when logged in. Running the Windows machine in this mode could increase the likelihood of an attacker interacting with the SCADA interface.

We were unsuccessful in running RDP replay tools; they could let us see the desktop and mouse movements of an RDP session. Some RDP sessions that we tried to replay showed the mouse movements but on a black screen. More research could determine why the screens did not record mouse movements and whether that was a deliberate tactic used by the attacker to avoid monitoring.

APPENDIX A. WINDOWS AUDIT POLICY IMPLEMENTATION

This appendix contains instructions for setting the audit policy of the honeypot's Windows machine (Microsoft, 2021a). This configuration is the audit policy implemented in Experiment 4 after applying lessons learned from all previous experiments.

On the Windows virtual machine:

1) Open gpedit.msc and navigate to Computer Configuration → Windows Settings → Security Settings → Local Policies →Audit Policy
2) Set the policies to the following:
    i) Audit account logon events: Success, Failure
    ii) Audit account management: Success, Failure
    iii) Audit directory service access: Success, Failure
    iv) Audit logon events: Success, Failure
    v) Audit object access: Success, Failure
    vi) Audit policy change: Success, Failure
    vii) Audit privilege use: Success, Failure
    viii) Audit process tracking: Success, Failure
    ix) Audit system events: Success, Failure
3) Go back to Security Settings → Advanced Audit Policy Configuration → System Audit Policies – Local Group Policy Object → Detailed Tracking
4) Set policies to the following:
    i) Audit Process Creation: Success, Failure
    ii) Audit Process Termination: Success, Failure
5) Go back to Computer Configuration → Administrative Templates → Windows Components → Windows PowerShell
6) Enable the following settings:
    i) Turn on PowerShell Script Block Logging
    ii) Turn on Module Logging

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. NXLOG CONFIGURATION FILE

This appendix contains the configuration files for NXLog. We created two configuration files: one for the Windows machine and the other for our logging server. The Log Generator Configuration is the file used on the Windows machine that generates event logs. It converts the event logs into JSON format and sends them to a remote server on port 1514. The Log Collector Configuration was the configuration file we used on our log server. It received the logs on port 1514 and saved them to our log directory. It also rotated the files every 100 megabytes.

Log Generator Configuration:

```
<Extension _json>
      Module      xm_json
</Extension _json>
<Input in>
      Module      im_msvistalog
      Exec        to_json();
</Input>
<Output out>
      Module      om_tcp
      Host        ***IP Address of remote server***
      Port        1514
</Output>
<Route 1>
      Path        in => out
</Route>
```

Log Collector Configuration:

```
<Input in>
      Module      im_tcp
      Host        ***IP Address of localhost***
      Port        1514
</Input>
<Output fileout>
      Module      om_file
      File        ***path to file where logs will be stored***
      <Schedule>
            When  @hourly
            <Exec>
                  if file_size(file_name()) >= 100M
                  {
                        file_cycle(file_name());
                        reopen();
                  }
            </Exec>
      </Schedule>
</Output>
<Route 1>
      Path  in => fileout
</Route>
```

APPENDIX C. ANALYSIS TOOLS AND SCRIPTS

This appendix contains command-line tools we used to capture and analyze our data. Several examples are given as a starting point, but they are flexible and can be modified to the specific needs of the user.

jq: Command line JSON processor

```
cat <event log file name> | jq 'select(.<json key>= = <value>)'
```

Example for returning all logon events (EventID 4624):

```
cat <name of json event log file> | jq 'select(.EventID == 4624)'
```

Example for returning all events from Sysmon log:

```
cat <name of json event log file> | jq 'select(.SourceName == "Microsoft-Windows-Sysmon/Operational")'
```

Tshark commands used:

Use the following lines to start Tshark on both the User-Interface Droplet and the GridPot Droplet. These will rotate files every 20 megabytes.

```
nohup tshark -i eth0 -w <path_to_filename> -b filesize:20000 > /dev/null &
nohup tshark -i eth1 -w <path_to_filename> -b filesize:20000 > /dev/null &
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D. EXPERIMENT 1 INFECTION REPORT

On 26 December 2021 at 3:53AM PST a successful logon occurred on the Windows machine. The event log entry (Figure 18) shows that this RDP session was reconnected which indicates that this client had logged on previously. Figure 19 shows the corresponding network connection to port 3389 of our Windows machine. Although we could not decrypt RDP traffic then because the certificate had renewed with a different key, we could still identify the source IP address of the connection as 151.80.148.159. According to whatismyipaddress.com, this IP address was a Tor exit node in Florida.


```
{
 "EventTime": "2021-12-26 03:53:59",
 "Hostname": "DESKTOP-1OFNB15",
 "Keywords": 11529215046068470000,
 "EventType": "INFO",
 "SeverityValue": 2,
 "Severity": "INFO",
 "EventID": 25,
 "SourceName": "Microsoft-Windows-TerminalServices-LocalSessionManager",
 "ProviderGuid": "{5D896912-022D-40AA-A3A8-4FA5515C76D7}",
 "Version": 0,
 "Task": 0,
 "OpcodeValue": 0,
 "RecordNumber": 1186,
 "ActivityID": "{F4202128-FC34-4DC2-9B04-7A26E0B90000}",
 "ProcessID": 1004,
 "ThreadID": 94204,
 "Channel":     "Microsoft-Windows-TerminalServices-LocalSessionManager/
Operational",
 "Domain": "NT AUTHORITY",
 "AccountName": "SYSTEM",
 "UserID": "S-1-5-18",
 "AccountType": "User",
 "Message": "Remote Desktop Services: Session reconnection succeeded:\r\
n\r\nUser: DESKTOP-1OFNB15\\Visitor\r\nSession ID: 10\r\nSource Network
Address: 10.0.2.2",
 "Opcode": "Info",
 "EventReceivedTime": "2021-12-27 16:12:13",
 "SourceModuleName": "in",
 "SourceModuleType": "im_msvistalog"
}
```


Figure 18. Event Log Entry for Successful Logon to the Windows Machine.

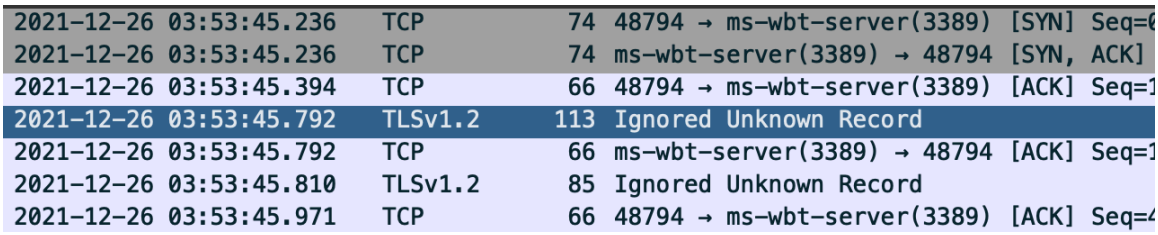


Figure 19. PCAP Data that Shows RDP CONNECTION.

A DNS request (Figure 20) was sent from the Windows machine for download.advanced_ip_scanner.com. The DNS response (Figure 21) returned two IP addresses. We searched the PCAPs for both IP addresses and found an HTTPS session with 104.26.1.126 that downloaded a file. This file was suspected to be the Advanced IP Scanner tool that we observed during analysis of the event logs. Checking the registry keys for evidence of the tool revealed a key entry for Computer\ HKEY_USERS\<SID>\SOFTWARE\famatech\advanced_ip_scanner.

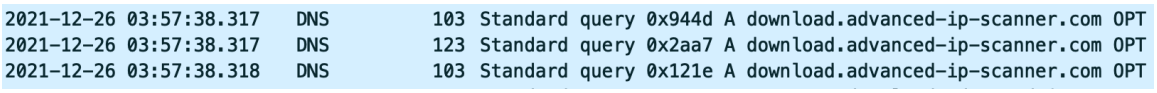


Figure 20. DNS Request from Windows Machine for download.advanced_ip_scanner.com

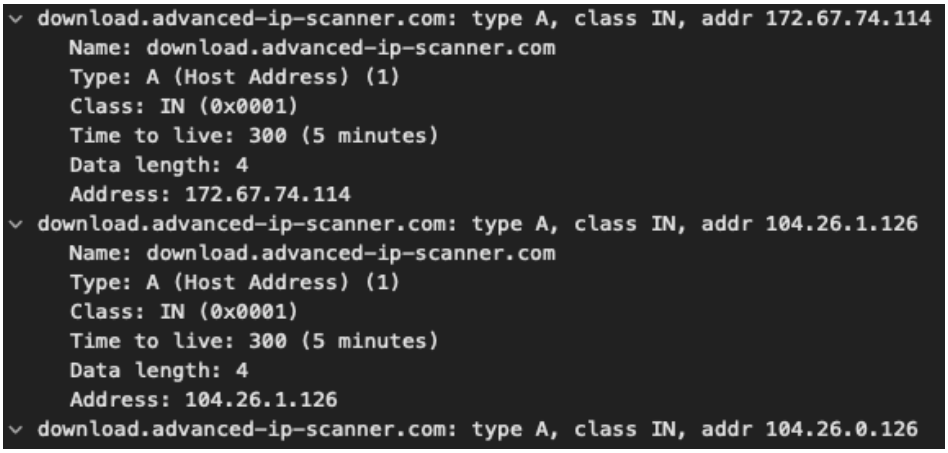


Figure 21. DNS Response for download.advanced_ip_scanner.com

Figures 22 and 23 show the state of the registry after the Advanced IP Scanner tool was installed on the Windows machine. One registry key for the tool contains the range of IP addresses on which to perform the scan. In this instance, it scanned the local network of the target machine, 10.0.2.0/24. The attacker may have had knowledge that this was a virtual machine running in VirtualBox because that is the default address for VirtualBox virtual machines using NAT.

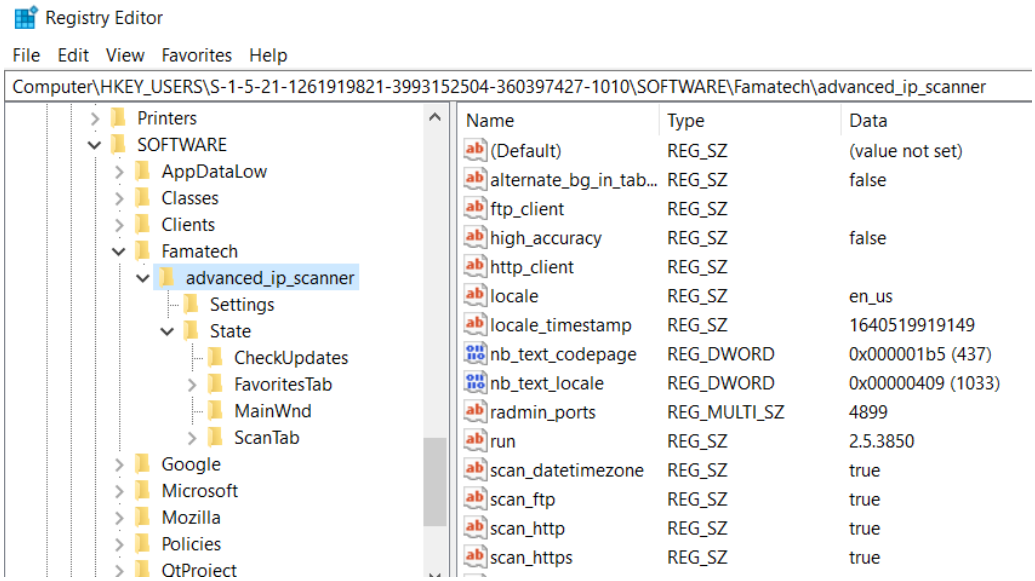


Figure 22. Registry Key for the Advanced IP Scanner Tool

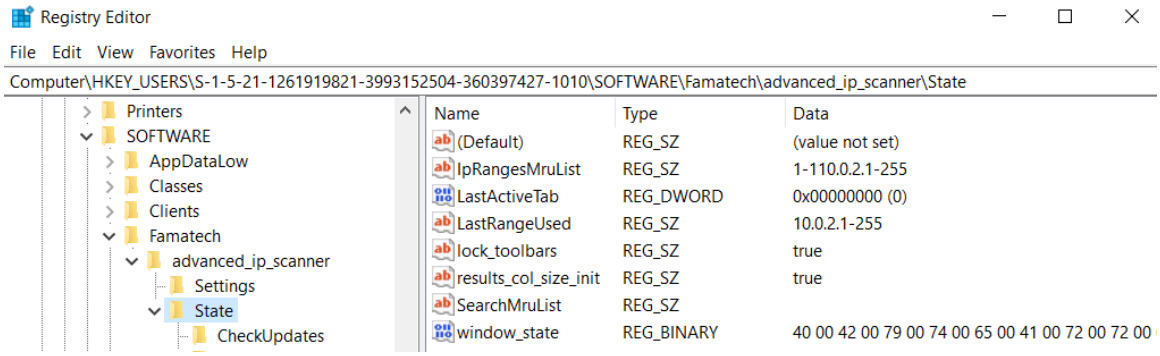


Figure 23. IP Range to Scan for Advanced IP Scanner

Just before the download of the tool, PowerShell was launched with two suspicious entries (Figure 24) that suggest manipulated registry event logging. The first event shows that RegProv was started which “enables management applications to retrieve and modify data in the system registry, and receive notifications when changes occur” (Microsoft, 2021b). The second event shows that MS_NT_EVENTLOG_PROVIDER was started in the Windows Management Instrumentation command line interface. The event logs provided insufficient evidence to say exactly what they did, but both events happened just before Advanced IP Scanner was downloaded; evidence of changes to the registry was lacking in the event logs, so we surmise the attacker ran a script that disabled logging for a short period.


```
{
 "EventTime": "2021-12-26 03:55:39",
 "Hostname": "DESKTOP-1OFNB15",
 "Keywords": 4611686018427388000,
 "EventType": "INFO",
 "SeverityValue": 2,
 "Severity": "INFO",
 "EventID": 5857,
 "SourceName": "Microsoft-Windows-WMI-Activity",
 "ProviderGuid": "{1418EF04-B0B4-4623-BF7E-D74AB47BBDAA}",
 "Version": 0,
 "Task": 0,
 "OpcodeValue": 0,
 "RecordNumber": 7331,
 "ProcessID": 338404,
 "ThreadID": 299924,
 "Channel": "Microsoft-Windows-WMI-Activity/Operational",
 "Domain": "NT AUTHORITY",
 "AccountName": "LOCAL SERVICE",
 "UserID": "S-1-5-19",
 "AccountType": "Well Known Group",
 "Message": "RegProv provider started with result code 0x0. HostProcess
=     wmiprvse.exe;     ProcessID     =     338404;     ProviderPath     =
%systemroot%\\system32\\wbem\\stdprov.dll",
 "Opcode": "Info",
 "EventReceivedTime": "2021-12-27 16:12:13",
 "SourceModuleName": "in",
 "SourceModuleType": "im_msvistalog"
}
{
 "EventTime": "2021-12-26 03:55:39",
 "Hostname": "DESKTOP-1OFNB15",
 "Keywords": 4611686018427388000,
 "EventType": "INFO",
 "SeverityValue": 2,
 "Severity": "INFO",
 "EventID": 5857,
```

```
 "SourceName": "Microsoft-Windows-WMI-Activity",
 "ProviderGuid": "{1418EF04-B0B4-4623-BF7E-D74AB47BBDAA}",
 "Version": 0,
 "Task": 0,
 "OpcodeValue": 0,
 "RecordNumber": 7332,
 "ProcessID": 268712,
 "ThreadID": 367560,
 "Channel": "Microsoft-Windows-WMI-Activity/Operational",
 "Domain": "NT AUTHORITY",
 "AccountName": "NETWORK SERVICE",
 "UserID": "S-1-5-20",
 "AccountType": "Well Known Group",
 "Message": "MS_NT_EVENTLOG_PROVIDER provider started with result code
0x0. HostProcess = wmiprvse.exe; ProcessID = 268712; ProviderPath =
%systemroot%\\system32\\wbem\\ntevt.dll",
 "Opcode": "Info",
 "EventReceivedTime": "2021-12-27 16:12:13",
 "SourceModuleName": "in",
 "SourceModuleType": "im_msvistalog"
}
```


Figure 24. Event Log Entries for Possible Log Manipulation or Evasion

An event logged in the DistributedCOM log provided a clue about the Advanced IP Scanner tool. The attacker tried to run the tool on the NAT network that VirtualBox creates by default. The scan would have provided no results since no other machines existed in that network. The message of the event log shows a path for advanced_ip_scanner.exe, but when that file location was inspected on the post-run virtual machine, it was no longer there. The only evidence that the tool was ever on the machine was the registry key that was left behind.


```
{
 "EventTime": "2021-12-26 04:01:39",
 "Hostname": "DESKTOP-1OFNB15",
 "Keywords": -9187343239835812000,
 "EventType": "ERROR",
 "SeverityValue": 4,
 "Severity": "ERROR",
 "EventID": 10028,
 "SourceName": "Microsoft-Windows-DistributedCOM",
 "ProviderGuid": "{1B562E86-B7AA-4131-BADC-B6F3A001407E}",
 "Version": 0,
 "Task": 0,
 "OpcodeValue": 0,
 "RecordNumber": 46559,
 "ActivityID": "{DEE92A88-BF6C-4921-BF84-3FA058008A42}",
 "ProcessID": 944,
```

```
 "ThreadID": 339664,
 "Channel": "System",
 "Domain": "DESKTOP-1OFNB15",
 "AccountName": "Visitor",
 "UserID": "S-1-5-21-1261919821-3993152504-360397427-1010",
 "AccountType": "User",
 "Message": "DCOM was unable to communicate with the computer 10.0.2.255
using any of the configured protocols; requested by PID 58ce8
(C:\\Users\\Visitor\\AppData\\Local\\Temp\\Advanced IP Scanner
2\\advanced_ip_scanner.exe), while activating CLSID {8BC3F05E-D86B-11D0-
A075-00C04FB68820}.",
 "Opcode": "Info",
 "param1": "10.0.2.255",
 "param2": "  58ce8",
 "param3":    "C:\\Users\\Visitor\\AppData\\Local\\Temp\\Advanced IP
Scanner 2\\advanced_ip_scanner.exe",
 "param4": "{8BC3F05E-D86B-11D0-A075-00C04FB68820}",
 "EventReceivedTime": "2021-12-27 16:12:13",
 "SourceModuleName": "in",
 "SourceModuleType": "im_msvistalog"
}
```


The following day, 27 December 2021, another remote login to the Windows machine began a series of suspicious events. After investigating the processes that were run during this remote session, we pieced together the following timeline:

1. 12-27-21 08:57:36 First TCP connection from IP 146.0.40.37 - Started TCP connection to port 3389 and immediately closed connection.

2. 12-27-21 10:28:13 Event ID: 4625 Failed login attempt to the Visitor account from IP 146.0.40.37.

3. 12-27-21 10:28:16 Client with IP 146.0.40.37 logs on to “Visitor” account through RDP.

4. 12-27-21 10:28:27 Event ID: 4624 – An account was successfully logged on (Visitor).

5. 12-27-21 10:28:37 A program name “c.exe” was created in and stored on the Desktop of the Visitor account.

6. 12-27-21 10:28:47 Event ID:4688 - “c.exe” process was started.

7. 12-27-21 10:28:52 “c.exe” connects to the following IP addresses:
   76.74.234.210 (http://codeproject.com).
   Sends HTTP Get request over port 80.
   Server responds with 302 Not found and redirects to https://codeproject.com.
   Downloads a file (suspected to be “Firefox Installer.exe”) over port 443.

8. 12-27-21 10:28:53 “c.exe” connects to the following IP addresses:
   104.18.115.97 (icanhazip.com) over port 80.
   Server responds with external IP address of the honeypot
   34.117.59.81 (ipinfo.io/json) over port 80.
   Server responds with external IP address of the honeypot.

9. 12-27-21 10:29:43 Event ID: 4689 - “c.exe” process exits.

10. 12-27-21 10:29:48 EventID: 4659 - “c.exe” gets deleted.

11. 12-27-21 10:29:58 Event ID: 4688 - “Firefox Installer.exe” process starts.

12. 12-27-21 10:29:59 Event ID:4688 - “Firefox Installer.exe” starts “setup-stub.exe” process to run.

13. 12-27-21 10:30:05 Event ID: 4625 - Failed local logon to the “Remote Admin” account.

14. 12-27-21 10:30:10 “setup-stub.exe” connects to IP 13.35.102.60 over port 443 to download a large file (suspected to be the actual Firefox browser).

15. 12-27-21 10:30:11 Event ID: 4634 - Visitor account logs off.

16. 12-27-2110:30:16“setup-stub.exe”calls
   “C:\\Users\\Visitor\\AppData\\Local\\Temp\\nshE484.tmp\\download.exe”.

17. 12-27-21 10:30:27 Event ID:4688 -
   “C:\\Users\\Visitor\\AppData\\Local\\Temp\\nshE484.tmp\\download.exe” calls
   “C:\\Users\\Visitor\\AppData\\Local\\Temp\\7zSC1B9C9D8\\setup.exe”.

18. 12-27-21 10:30:40 Event ID:4688 -
   “C:\\Users\\Visitor\\AppData\\Local\\Temp\\7zSC1B9C9D8\\setup.exe” calls
   "C:\\Windows\\System32\\regsvr32.exe".
   Attempts to launch with privileges, but is denied and exits.

19. 12-27-21 10:30:46 Event ID:4688 -
   “C:\\Users\\Visitor\\AppData\\Local\\Temp\\7zSC1B9C9D8\\setup.exe” calls
   “default-browser-agent.exe”.
   Scheduled task is registered that runs “default-browser-agent.exe” when Firefox is launched.

20. 12-27-21 10:30:49 Event ID:4688 -
   "C:\\Users\\Visitor\\AppData\\Local\\Temp\\7zS8B37139D\\setup-stub.exe" calls
   “C:\\Users\\Visitor\\AppData\\Local\\Mozilla Firefox\\firefox.exe”.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX E. BLUEKEEP EXPLOIT SCAN

This appendix shows two examples of BlueKeep scans that we observed in our honeypot data. Both detect a client's attempt to connect to the MS_T120 channel during the RDP connection sequence (Microsoft, 2022).

The first example (Figure 25) is from the PyRDP tool we used in Experiment 2; it explicitly states the BlueKeep scan or exploit was attempted. The boldface characters are the hexadecimal representation of "MS_T120".


```
{
 "sensor": "PyRDP",
 "message": "Exploit detected: %(exploitInfo)s. %(parserInfo)s",
 "loggerName": "pyrdp.mitm.connections.tcp",
 "timestamp": "2022-01-28T05:03:32.281024",
 "level": "INFO",
 "sessionID": "Richard841280",
 "clientIp": "74.82.47.5",
 "exploitInfo": "BlueKeep scan or exploit attempted",
 "parserInfo": "ClientConnectionParser =
01c0d800040008002003580201ca03aa09040000280a0000720064007000730063006
1006e000000000000000000000000000000000000000400000000000000000c00000000
000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000001ca01000000
0000ff000700010000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000
000000000001000000004c00c000900000000000000002c00c00000000000000000003c
01400010000004d535f543132300080800000"
}
```


Figure 25. BlueKeep Exploit Detected in the PyRDP Logs.

The second example (Figure 26) is from a network packet in the PCAP data. During that RDP connection, the client requested to establish the MS_T120 channel.

```
"rdp.client.networkData": {
       "rdp.header.type": "0x0000c003",
       "rdp.header.length": "20",
       "rdp.channelCount": "1",
       "rdp.channelDefArray": {
        "rdp.channelDef": {
         "rdp.name": "MS_T120",
         "rdp.options": "0x00008080",
         "rdp.options_tree": {
          "rdp.options.initialized": "0x00000000",
          "rdp.options.encrypt.rdp": "0x00000000",
          "rdp.options.encrypt.sc": "0x00000000",
          "rdp.options.encrypt.cs": "0x00000000",
          "rdp.options.priority.high": "0x00000000",
```

Figure 26. BlueKeep Exploit Detected in RDP Connection Sequence (PCAP).

LIST OF REFERENCES

Bieker, M., 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. http://hdl.handle.net/10945/66586

Cherepanov, A. (2017). *Win32/Industroyer: A new threat for industrial control systems.* WeLiveSecurity. https://www.welivesecurity.com/wp-content/uploads/2017/06/Win32_Industroyer.pdf

Citronneur (2020). RDPY. [Computer software]. Github. https://github.com/citronneur/rdpy

Clarke, G., & Reynders, D. (2004). *Practical modern SCADA protocols: Dnp3, 60870. 5 and related systems*. Elsevier Science & Technology.

Conpot.org (n.d.). *Conpot ICS/SCADA honeypot*. http://conpot.org

Cybersecurity and Infrastructure Security Agency [CISA] (2020). *Securing industrial control systems: A unified initiative.* https://www.cisa.gov/sites/default/files/publications/Securing_Industrial_Control_Systems_S508C.pdf

Cybersecurity and Infrastructure Security Agency [CISA] (2021). *Compromise of U.S. water treatment facility.* https://www.cisa.gov/uscert/sites/default/files/publications/AA21-042A_Joint_Cybersecurity_Advisory_Compromise_of_U.S._Drinking_Treatment_Facility.pdf

Dahlberg, R & Pulls, T. (2016). *Standardized Syslog Processing.* http://kau.diva-portal.org/smash/get/diva2:954004/FULLTEXT02.pdf

DigitalOcean (2020). *VPC Quickstart.* https://docs.digitalocean.com/products/networking/vpc/quickstart/

Dougherty, J. (2020). *Evasion of honeypot detection mechanisms through improved interactivity of ICS-based systems* [Master's thesis, Naval Postgraduate School]. NPS Archive: Calhoun. http://hdl.handle.net/10945/66065

Enscada (2021). IndigoSCADA [Computer software]. Github. https://github.com/enscada/IndigoSCADA

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

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., & Berners-Lee, T. (1999). *Hypertext transfer protocol-HTTP/1.1.*[Memorandum] Internet Engineering Task Force. https://www.w3.org/Protocols/rfc2616/rfc2616.html

Franco, J., Aris A., Canberk, Berk. & Uluagac, A. (2021). *A survey of honeypots and honeynets for Internet of Things, Industrial Internet of Things, and cyber-physical systems*. https://arxiv.org/pdf/2108.02287.pdf

Gerhards, R. (2009). *The Syslog Protocol, RFC 5424, DOI 10.17487/RFC5424.* https://www.rfc-editor.org/info/rfc5424.

GoSecure (2021). PyRDP [Computer software]. Github. https://github.com/GoSecure/pyrdp

Hilt, S., Maggi, F., Perine, C., Remorin, L., Rösler, M, & Vosseler, R. (2020). *Caught in the Act: Running a realistic factory honeypot to capture real threats.* Trend Micro Research. https://documents.trendmicro.com/assets/white_papers/wp-caught-in-the-act-running-a-realistic-factory-honeypot-to-capture-real-threats.pdf

Industrial Control Systems Cyber Emergency Response Team (2016). *Recommended practice: Improving industrial control system cybersecurity with defense-in-depth strategies.* https://www.cisa.gov/uscert/sites/default/files/recommended_practices/NCCIC_ICS-CERT_Defense_in_Depth_2016_S508C.pdf

Instrumentation, Systems and Automation Society. (2006). *Mitigations for security vulnerabilities found in control system networks.* https://www.cisa.gov/uscert/sites/default/files/recommended_practices/MitigationsForVulnerabilitiesCSNetsISA_S508C.pdf

Kelly, C., Pitropakis, N., Mylonas, A., McKeown, S., & Buchanan, W. J. (2021). *A comparative analysis of honeypots on different cloud platforms. Sensors, 21(7), 2433.* http://dx.doi.org/10.3390/s21072433

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

Magisterquis (2019). VNCLowPot [Computer software]. Github. https://github.com/magisterquis/vnclowpot

Manzanares, A. (2017). *HoneyIo4: The construction of a virtual, low- interaction IoT Honeypot.* https://upcommons.upc.edu/bitstream/handle/2117/108166/Alejandro_Guerra_Manzanares.pdf?sequence=1&isAllowed=y

Microsoft (2021a). *Audit policy*. https://docs.microsoft.com/en-us/windows/security/threat-protection/security-policy-settings/audit-policy

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

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

Microsoft (2022a). *Microsoft NTLM*. https://docs.microsoft.com/en-us/windows/win32/secauthn/microsoft-ntlm

Microsoft (2022b) *Remote desktop protocol: basic connectivity and graphics remoting.* https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-rdpbcgr/023f1e69-cfe8-4ee6-9ee0-7e759fb4e4ee

MITRE (2002). *CVE -CVE-2001-0540.* https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2001-0540

MITRE (2019a). *CVE -CVE-2019-0708.* https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-0708

MITRE (2019b). *CVE -CVE-2019-1182.* https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-1182

MITRE ATT&CK (2021). Defense Evasion. *ATT&CK Matrix for Enterprise*. https://attack.mitre.org/tactics/TA0005/

National Institute of Standards and Technology [NIST] (2015) *Guide to industrial control systems (ICS) security.* Federal Information Processing Standards Publications (FIPS PUBS) 800-82, Revision 2 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf

National Institute of Standards and Technology [NIST] (2018). *CVE-2001-0540 Detail*. https://nvd.nist.gov/vuln/detail/CVE-2001-0540#vulnCurrentDescriptionTitle

Navarro, Ó., Balbastre, S. & Beyer, S. (2019). Gathering intelligence through realistic industrial control system honeypots. critical information infrastructures security. *CRITIS 2018*. https://doi-org.libproxy.nps.edu/10.1007/978-3-030-05849-4_11

NXLog (2022). NXLog [Computer Software]. https://nxlog.co/documentation/nxlog-user-guide/about-nxlog.html

Parfomak, P. & Jaikaran, C. (2021) *Colonial pipeline: The DarkSide strikes*. Congressional Research Service. https://crsreports.congress.gov/product/pdf/IN/IN11667

Poling, J. (2018). *Windows RDP-related event logs: Identification, tracking, and investigation*. https://ponderthebits.com

Pothamsetty, V. & Franz, M. (2004). *SCADA HoneyNet project: Building honeypots for industrial networks.* http://scadahoneynet.sourceforge.net

Provos, Ni. (2004). *A virtual honeypot framework.* https://www.usenix.org/legacy/event/sec04/tech/full_papers/provos/provos_html/

Russinovich, M. & Garnier, T. (2022). *Sysmon v13.33.* https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon

Serbanescu, A., Obermeier, S., & Yu, D. (2015). A flexible architecture for industrial control system honeypots. *Proceedings of the 12th International Conference on Security and Cryptography*. https://www.scitepress.org/papers/2015/55225/55225.pdf

Sk4ld (2015). GridPot (Version 2) [Computer software]. Github. https://github.com/sk4ld/GridPot

SwiftOnSecurity (2021). Sysmon-config [Computer software]. Github. https://github.com/SwiftOnSecurity/sysmon-config

United States Cyber Command (n.d.) *Our history.* https://www.cybercom.mil/About/History/

Washofsky, A. (2021). *Deploying and analyzing containerized honeypots in the cloud with T-Pot* [Master's thesis, Naval Postgraduate School]. NPS Archive: Calhoun. http://hdl.handle.net/10945/68394

Yen, K. (2019). *BlueKeep: Detecting and remediating a critical and wormable remote code execution vulnerability.* https://www.opswat.com/blog/bluekeep-detecting-and-remediating-a-critical-and-wormable-remote-code-execution-vulnerability

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California