Institiúid Teicneolaíochta Cheatharlach
INSTITUTE of TECHNOLOGY CARLOW
At the Heart of South Leinster

WINTRE: AN ADVERSARY EMULATION TOOL

Research Document

Student: Martin Earls / C00227207
Supervisor: Richard Butler

1 Table of CONTENTS

2 ABSTRACT

This document outlines the research required to develop an application for adversary emulation, where this tool could be used to run small, isolated security tests that are used by attackers in the real world. This tool will be developed using the MITRE ATT&CK Framework as a reference to help automate Purple Team activities. This is where an offensive and defensive security team simulate a cyber-attack on their network's endpoints. This project involves a C# GUI application that can execute tactics, techniques and procedures (TTPs) based on popular methods utilised by threat actors and advanced persistent threat groups. This app would be used in Purple Team engagements to launch TTPs and produce detailed logs to help an organisation test their detection analytics. This helps to generate indicators of compromise whilst providing documentation for an organisation to help gain visibility over what techniques they're able to detect in their environment and ones they're not able to.

Normally, a security analyst would have to research each TTP, manually code or script the tests. The analyst would then have to manually test each TTP to ensure they were executing as intended with the appropriate functionality. They would then work with the system administrators and or Security Operations Centre during execution to determine what techniques are not getting detected in their environment as well as documenting it (e.g. documenting credential theft, discovery techniques not being detected).

WINTRE aims to solve many of the challenges in creating such a tool, providing a convenient platform for adversary emulation, suitable for large and small enterprise environments to help protect Windows endpoints. WINTRE will also incorporate modern techniques, many of which have been openly researched but not incorporated into existing applications.

3 INTRODUCTION

Adversary emulation is becoming an increasingly popular method of testing and validating high-cost security controls within an organisation. Performing this manual testing is extremely time consuming and requires strong technical and theoretical knowledge in security analysis.

There is also a lack of tools available to perform this testing in an automated fashion, as well as many similar tools lacking features, being closed source or having disadvantages when testing Windows endpoints. The main disadvantages are when it comes to the lack of digital signatures blocking the execution of agents or techniques as well as some tools making it difficult to test individual techniques with the minimum functionality required, including subsets of techniques. With WINTRE, the tester does not require the technical knowledge or skill set of a security analyst as thoroughly tested TTPs are included with the application. Select TTPs should be easy to find and execute. TTPs should also allow customization from the GUI. By making the test allowing the documenting of findings to take precedence.

WINTRE gives purple teams and system administrators the ability to simulate an attacker's behaviour on their network, during engagements, providing an increasingly novel and effective way to increase an organisation's security posture. Advanced users such as penetration testers, security analysts or researchers could also make use of WINTRE by adding their own custom tests. All built-in tests will focus on several categories to facilitate assumed breach scenarios.

Below are the chosen tactics to be implemented for WINTRE. These tactics are highly likely to be adopted and used by hackers in a cyber-attack. Testing for them can help enable the cyber kill chain, allowing an organisation to detect and respond to an attack in an earlier stage, as a result of using adversary emulation to fine-tune detection mechanisms and security controls.

3.1 CORE TACTICS

- **Code Execution:** An adversary performing arbitrary code execution, code execution can take place in various forms, attackers can use scripting languages or programming languages like C, C++ or C# to create executables to achieve this. Techniques could involve built-in Windows, live-off-the-land binaries and utilising built-in Windows applications to execute arbitrary code.

- **Persistence:** Once a system has been compromised, an attacker will ideally seek to establish persistence. This tactic allows the attacker continuous, or repeatable access to the comprised device. Techniques could involve scheduled tasks, backdoor services, rootkits, malicious registry entries and other types of malware.

- **Privilege Escalation:** A somewhat more difficult tactic to emulate, as privilege escalation typically involves in-depth inspection of a system to determine attack vectors. Despite this, many of the actual techniques being used to achieve privilege escalation, are still feasible for automation and will certainly help improve detection analytics in this regard (Spotless 2020).

- **Defence Evasion:** Naturally, an attacker will attempt to obfuscate themselves to prevent being detected. Evasion usually comes down to evading anti-virus and EDR. Anti-virus is no longer suitable on its own, as attackers can easily bypass signature-based detection in executables. This is also true for command-line scripting, where PowerShell and Batch obfuscation are important techniques to test, along with non-command-line based techniques such as in-memory process manipulation.

- **Credential Theft:** An adversary's primary goal rather than system compromise, is to acquire credentials that can be further leveraged on the victim's network, enabling them greater access to other user's personally identifiable information, or a company's trade secrets and intellectual property. Microsoft has introduced many security controls built into modern Windows systems to help prevent and detect this tactic, such as Credential Guard and Local Security Authority Protection. One should assume a motivated attacker will still retrieve credentials, and thus the individual techniques involved should be tested regularly.

- **Discovery:** Once an adversary gains access to a system or target network, they will attempt to move laterally. Before moving laterally, an attacker must fingerprint the environment and determine what other PCs are running or perform local enumeration on the machine they're on, discovery can be noisy but also often goes undetected as the techniques themselves are seen in legitimate software and the Windows OS itself. Knowing that these events are still being logged is still highly important and relevant for these testing purposes.

- **Lateral Movement:** Preventing lateral movement and network pivoting is essential to implementing the cyber kill chain and establishing some level of resilience. There are many ways to move laterally between Windows systems, typically involving SMB for Windows endpoints. SMB lateral movement techniques will be the primary focus of implementation in this tactic.

3.2 Discretionary Tactics

- **Collection:** The gathering of data, typically for further analysis to determine vulnerabilities or retrieve sensitive information. Traditionally just gathering files, attackers could also use malware to gather network traffic, audio, video and keystrokes. One of the primary techniques of this category will be a keylogger to record keystrokes.

- **Data Exfiltration:** Once an attacker has acquired sensitive data, they need to exfiltrate it. These tests will involve the manipulation, archiving and processing of files for external transfer. While there's no server component to WINTRE, these techniques, will require a simple listener on an external system to validate. Netcat will suffice for the receiving end of the data transmission. Techniques here could involve steganography and encoding/encryption to obfuscate files being exfiltrated.

- **Command and Control:** C2 components are often seen with Advanced Persistent Threat (APT) group campaigns, involving an agent and server communicating. WINTRE is not intended to be a fully functioning C2 suite, however many techniques such as downloaders and various reverse shells with differing levels of obfuscation could be implemented as techniques.

- **Impact:** This tactic can involve denial of service, destruction of data or denying access (ransomware) to resources. The primary technique to be implemented here would be a controlled, ransomware test designed to encrypt a user's documents. Other techniques could involve resource-hogging or attempting to corrupt system files as used by attackers to destroy data on production servers. Premiums for cyber insurance related to ransomware are also on the rise given how common this attack is becoming increasing the relevance of testing this tactic (CISA 2020).

4 Topics Researched

4.1 Adversary Emulation Tool Overview

"WINTRE", a portmanteau of "Windows" and "MITRE" will be installed on a Windows endpoint. The user will then select and execute techniques to generate traffic that can be used to create detection analytics. The techniques themselves will be of a minimum functionality, i.e. they will be tested as separately as possible excluding other techniques. This is an important step in validating security controls, so an organisation can establish a baseline in order to increase their security posture. When testing a reverse shell for example, it is not providing value to the organisation by first testing a HTTPS compatible, beaconing, encrypted reverse shell. Techniques should begin simple and then develop into more complex sub techniques as previous ones are sufficiently detected. A much more valuable test in this case, to start with, would be a basic, non-staged TCP reverse shell which could gradually increase in complexity with subsequent tests.

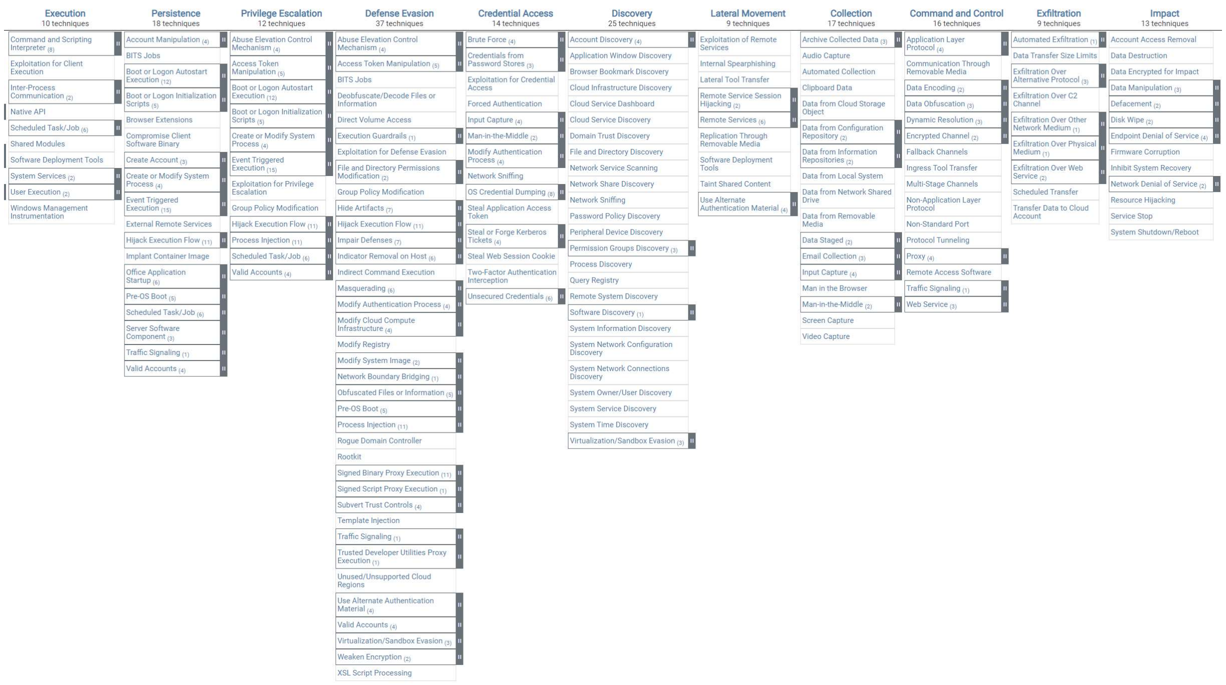
Once a user has ran their selected test, logs for each test will have accumulated on the OS and network. WINTRE will record the standard output, error and any notable test results to be reviewed on disk in log files as well as in an output view on the application.

WINTRE will then generate an automatic report using Microsoft.Office.Interop.Word namespace (Deenathayalan, M. 2019) which can be used to generate a Microsoft Word document and fill the core information of the report in, allowing the tester to tick which tests are being detected as they run. From here, system administrators or security analysts can use the test results to determine weaknesses in their detection pipeline. These tests will provide a low cost, repeatable and efficient way of testing an organisation's Security Information and Event Management (SIEM) controls.

4.2 MITRE ATT&CK

The MITRE corporation was founded as a private, non-profit research company for the US government in 1958. Their focus of research, evolving from traditional engineering and innovating technology expanded into computing, software development, communications, program management, and systems engineering. Working closely with the US government throughout its existence, MITRE has a long history of developing important technologies and contributing research projects.

MITRE ATT&CK (Adversarial Tactics, Techniques, and Common Knowledge) began as a result of a MITRE research project dubbed FMX, in 2013 to provide detailed documentation for TTPs that APT groups had used successfully in their attacks on companies (MITRE 2018). This allowed Windows (and later Linux) endpoint telemetry data and analytics to improve the chances an organisation has of detecting adversaries during the post-exploitation phase of an attack on an enterprise network.



Execution 10 techniques	Persistence 18 techniques	Privilege Escalation 12 techniques	Defense Evasion 37 techniques	Credential Access 14 techniques	Discovery 25 techniques
Command and Scripting Interpreter (8)	Account Manipulation (4)	Abuse Elevation Control Mechanism (4)	Abuse Elevation Control Mechanism (4)	Brute Force (4)	Account Discovery (4)
Exploitation for Client Execution	BITS Jobs	Access Token Manipulation (5)	Access Token Manipulation (5)	Credentials from Password Stores (3)	Application Window Discovery
Inter-Process Communication (2)	Boot or Logon Autostart Execution (12)	Boot or Logon Autostart Execution (12)	BITS Jobs	Exploitation for Credential Access	Browser Bookmark Discovery
Native API	Boot or Logon Initialization Scripts (5)	Boot or Logon Initialization Scripts (5)	Deobfuscate/Decode Files or Information	Forced Authentication	Cloud Infrastructure Discovery
Scheduled Task/Job (6)	Browser Extensions	Create or Modify System Process (4)	Direct Volume Access	Input Capture (4)	Cloud Service Dashboard
Shared Modules	Compromise Client Software Binary	Event Triggered Execution (15)	Execution Guardrails (1)	Man-in-the-Middle (2)	Cloud Service Discovery
Software Deployment Tools	Create Account (3)	Exploitation for Privilege Escalation	Exploitation for Defense Evasion	Modify Authentication Process (4)	Domain Trust Discovery
System Services (2)	Create or Modify System Process (4)	Group Policy Modification	File and Directory Permissions Modification (2)	Network Sniffing	File and Directory Discovery
User Execution (2)	Event Triggered Execution (15)	Hijack Execution Flow (11)	Group Policy Modification	OS Credential Dumping (8)	Network Service Scanning
Windows Management Instrumentation	External Remote Services	Process Injection (11)	Hide Artifacts (7)	Steal Application Access Token	Network Share Discovery
	Hijack Execution Flow (11)	Scheduled Task/Job (6)	Hijack Execution Flow (11)	Steal or Forge Kerberos Tickets (4)	Network Sniffing
	Implant Container Image	Valid Accounts (4)	Impair Defenses (7)	Steal Web Session Cookie	Password Policy Discovery
	Office Application Startup (6)		Indicator Removal on Host (6)	Two-Factor Authentication Interception	Peripheral Device Discovery
	Pre-OS Boot (5)		Indirect Command Execution	Unsecured Credentials (6)	Permission Groups Discovery (3)
	Scheduled Task/Job (6)		Masquerading (6)		Process Discovery
	Server Software Component (3)		Modify Authentication Process (4)		Query Registry
	Traffic Signaling (1)		Modify Cloud Compute Infrastructure (4)		Remote System Discovery
	Valid Accounts (4)		Modify Registry		Software Discovery (1)
			Modify System Image (2)		System Information Discovery
			Network Boundary Bridging (1)		System Network Configuration Discovery
			Obfuscated Files or Information (5)		System Network Connections Discovery
			Pre-OS Boot (5)		System Owner/User Discovery
			Process Injection (11)		System Service Discovery
			Rogue Domain Controller		System Time Discovery
			Rootkit		Virtualization/Sandbox Evasion (3)
			Signed Binary Proxy Execution (11)		
			Signed Script Proxy Execution (1)		
			Subvert Trust Controls (4)		
			Template Injection		
			Traffic Signaling (1)		
			Trusted Developer Utilities Proxy Execution (1)		
			Unused/Unsupported Cloud Regions		
			Use Alternate Authentication Material (4)		
			Valid Accounts (4)		
			Virtualization/Sandbox Evasion (3)		
			Weaken Encryption (2)		
			XSL Script Processing		

Lateral Movement 9 techniques	Collection 17 techniques	Command and Control 16 techniques	Exfiltration 9 techniques	Impact 13 techniques
Exploitation of Remote Services	Archive Collected Data (3)	Application Layer Protocol (4)	Automated Exfiltration (1)	Account Access Removal
Internal Spearphishing	Audio Capture	Communication Through Removable Media	Data Transfer Size Limits	Data Destruction
Lateral Tool Transfer	Automated Collection	Data Encoding (2)	Exfiltration Over Alternative Protocol (3)	Data Encrypted for Impact
Remote Service Session Hijacking (2)	Clipboard Data	Data Obfuscation (3)	Exfiltration Over C2 Channel	Data Manipulation (3)
Remote Services (6)	Data from Cloud Storage Object	Dynamic Resolution (3)	Exfiltration Over Other Network Medium (1)	Defacement (2)
Replication Through Removable Media	Data from Configuration Repository (2)	Encrypted Channel (2)	Exfiltration Over Physical Medium (1)	Disk Wipe (2)
Software Deployment Tools	Data from Information Repositories (2)	Fallback Channels	Exfiltration Over Web Service (2)	Endpoint Denial of Service (4)
Taint Shared Content	Data from Local System	Ingress Tool Transfer	Scheduled Transfer	Firmware Corruption
Use Alternate Authentication Material (4)	Data from Network Shared Drive	Multi-Stage Channels	Transfer Data to Cloud Account	Inhibit System Recovery
	Data from Removable Media	Non-Application Layer Protocol		Network Denial of Service (2)
	Data Staged (2)	Non-Standard Port		Resource Hijacking
	Email Collection (3)	Protocol Tunneling		Service Stop
	Input Capture (4)	Proxy (4)		System Shutdown/Reboot
	Man in the Browser	Remote Access Software		
	Man-in-the-Middle (2)	Traffic Signaling (1)		
	Screen Capture	Web Service (3)		
	Video Capture			

Figure 2. MITRE ATT&CK Matrix

The MITRE ATT&CK framework is an open-source repository of information relating to adversary tactics, techniques and procedures. The framework is split among Enterprise, Mobile and Industrial Control System TTPs, we will be focussing on Enterprise TTPs for the purpose of this research document. Each technique is mapped to "tactic" or category, such as dumping NTLM hashes, a technique which can be done many different ways, used as a Credential Theft tactic. All the techniques contained in the repository are a result of researchers analysing real-world threat actors and their corresponding malware campaigns, related to APT groups. The goal of ATT&CK is to ease the development of threat models and enable organisations to become more effective in defending their critical assets, by simulating attacks using these techniques.

There are 14 categories or tactics in the framework, 11 of these focus on the post exploitation phase of a cyber-attack, where an attacker has already gained some level of access to an organisation. These 11 categories will be the focus of implementing techniques into WINTRE, as it will be used to test TTPs in an assumed breach scenario. An important focus of WINTRE will also be to incorporate many new or recent procedures, i.e. in-the-wild techniques that have been observed and sub techniques.

4.3 APTs

MITRE, while collecting and organising documented techniques, has attributed them to individual APT groups, due to the complex nature of their attack vectors, resources and range of victims.

Profiling APT groups allows business to simulate certain groups that may target their industry. Some groups target a wide array of industries, and others target more specific ones, such as the recent ransomware campaigns targeting the US Healthcare industry with malware such as TrickBot, Ryuk and Conti (CISA 2020). Each malware was designed to harvest credentials, exfiltrate emails with sensitive or insider data, crypto-mining functionality as well as holding systems hostage via ransomware.

This has led to emulation tools having APT "profiles", i.e. a group of techniques attributed to a certain APT that an organisation can run to simulate the group breaching their network.

Whilst many of the techniques to be developed are mapped to various APTs in ATT&CK, I have chosen to try to implement techniques relating to a specific group, namely APT1. APT1 has been attributed to Unit 61398 of the People's Liberation Army, under China's state-sponsored cyber espionage program. China's Defence Ministry has however denied all accusations in relation to many cyber espionage campaigns (Mandiant 2014).

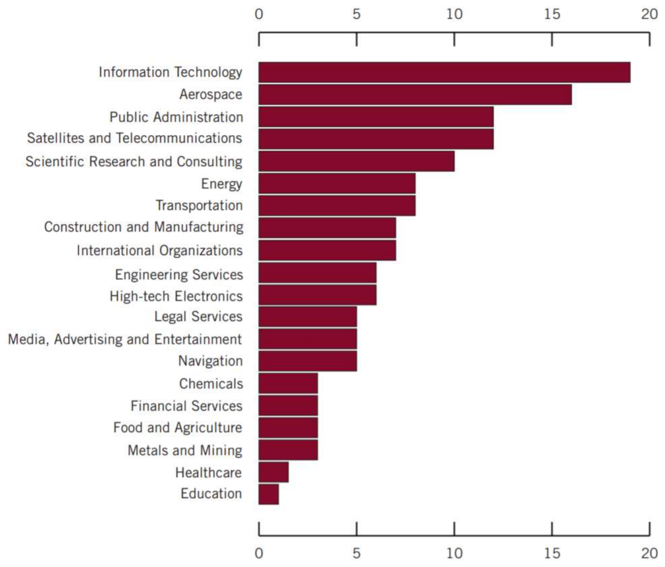
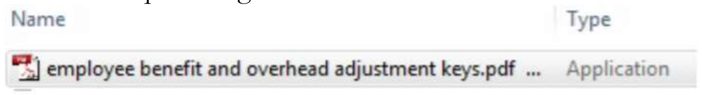


Figure 2. Industries Compromised by APT1 (Mandiant APT1 Report)

APT1's campaigns are ongoing and have targeted various industries from as early as 2006. There are many techniques APT1 has used to achieve compromise and extract sensitive user data on 141 occasions. On average the group would maintain access to a victim's network for close to 1 year, with the longest recorded maintained access at four years and ten months. This highlights more than ever the many advantages to adversary emulation and what automating these group's activities could help prevent, as many of the same techniques are commonly used by other APT groups.

The following techniques were chosen from Mandiant's report on the group to be implemented as standalone techniques that can also be selected as an APT "profile" or grouping or techniques.

1. Masquerading


Name Type

employee benefit and overhead adjustment keys.pdf ... Application

Figure 3.

The file shown in this image (Mandiant report) is not actually a PDF, but an executable with whitespace used to hide the file extension. The file's icon is also changed to match this.

2. Internal reconnaissance

Once an attacker has a foothold in the organisation, their goal is to firstly enumerate the local environment to increase the attack surface and thus increasing their chances of success by checking:

- Current network configuration
- Services running
- Processes running
- Local user accounts
- 3rd party software that may be common across endpoints
- If the machine is domain joined, gather basic AD information to explore later further
- Network shares
- Other systems on the network
- Current network connections

3. SMB lateral movement (PsExec)

Once an attacker has retrieved some form of credentials, e.g. local administrator NTLM hashes, if this local administrator account is standard across the domain it can then be used to move between systems using PsExec functionality, a publicly available tool from Microsoft's Sysinternals suite.

4. Archiving files for data collection

Archiving files for later data exfiltration is a key component of APT1's modus operandi, to steal and collect as much sensitive information as possible. APT1's largest known data theft from a single organisation was 6.5 terabytes over ten months.

4.4 PURPLE TEAMING

As threat actors and their campaigns continue to increase in complexity, in response to security controls improving (e.g. EDR over standard anti-virus), companies must also do the same. IT Security programs have significantly evolved over the years, rather than just having standard penetration tests, companies can now opt for secure application development, red team engagements and now purple teaming. Purple Teaming, in particular, is highly relevant to how WINTRE can be applied as part of a security program.

In a traditional red or blue team, the focus is typically one sided with specific goals in mind (performing or responding to a red team). Purple teaming is a set of activities that aims to improve the capabilities of both the red and blue teams. Both teams work closely together, typically with the red team simulating an attack, and the blue team validating that they're able to see and respond to it in real-time. It can be hard to come up with an accurate list of procedures to test, or even to get results if the engagement starts externally. However, with the ATT&CK framework, selecting a list of techniques to try is trivialised and even further so with adversary simulation.

Adversary simulation, in an assumed breach purple team engagement is an excellent way to improve both teams. The red team will likely have great initial success in executing techniques or reaching their goals depending on the maturity of the blue team. As the blue team then adapts and fine-tunes their detection pipeline, the red team must also adapt to avoid detection further, increasing the complexity of their payloads gradually (Redscan 2018).

Performing adversary simulation in an assumed breach setting, assumes the likely eventuality that your organisation will be compromised. This methodology of testing is advantageous, as it is likely at some point in future that a single missing patch, misconfiguration or 0-Day exploit will lead to some level of compromise in your organisation (Netsurion 2020). Assuming this, SOCs will have more chances and experience in dealing with incident response. This is in line with adversary simulation in which most tactics are tested from a standard endpoint that is presumed to have been compromised.

Therefore, WINTRE, which will have many techniques that can be launched with ease from the GUI, will make purple team engagements far easier to conduct. Since the tests won't rely on manual research, an experienced red teamer or penetration tester is also not required to initiate the tests. This serves as a much more meaningful way to increase the maturity of an organisation's network security program as these tests and their associated campaigns (set of tests to run for the engagement) can be relaunched and ideally configured with ease. The alternative to this, is hoping that when an actual security incident has occurred, or during a penetration test, that the SOC are ready and will be able to adequately respond.

This is flawed for the simple reason that regular, concentrated assumed breach engagements will provide far more experience to any SOC and give them the ability to create more alerts as well as fine-tune existing ones through their SIEM pipeline.

4.5 ANTI-VIRUS EVASION AND EDR

Anti-Virus, as a common security measure taken to prevent and detect malware is seen in most organisations and home computers nowadays. The most common OS for home desktops and endpoints in enterprises is Windows (cite this), which comes with Windows Defender at no extra cost. In the realm of security testing, a user or company needs to know that their anti-virus solution is protecting them adequately. Security analysts and researchers alike often write malware, that is synonymously used in penetration testing and security audits to bypass or evade anti-virus solutions.

This is a key component to WINTRE, as one of the ATT&CK categories is Defence Evasion, it is not sufficient enough for anti-virus products to simply detect a malicious program that has been obfuscated or employs some form of evasion poorly. It is also relevant to test individual evasion techniques, with non-malicious payloads.

Example of a non-malicious payload:
Below is a simple assembler program to launch calculator in Windows 10. Naturally, Windows Defender does not detect anything malicious from this executable.

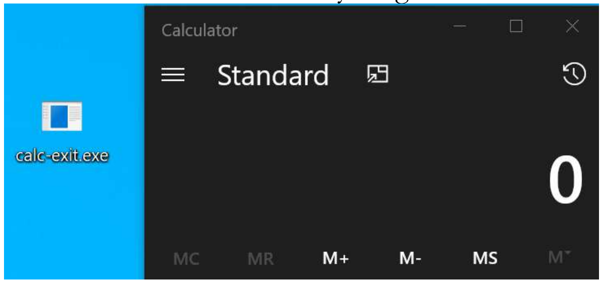
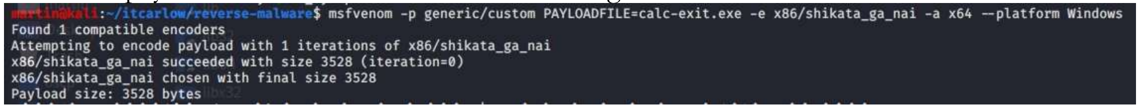


Figure 4. Non-malicious payload that simply launches calculator

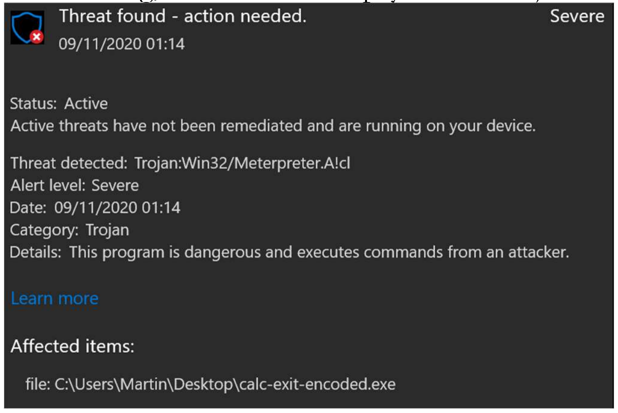
The same payload with x86 Shikata Ga Nai encoding via MSFVenom.



```
martin@kali:~/itcarlow/reverse-malware$ msfvenom -p generic/custom PAYLOADFILE=calc-exit.exe -e x86/shikata_ga_nai -a x64 --platform Windows
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 3528 (iteration=0)
x86/shikata_ga_nai chosen with final size 3528
Payload size: 3528 bytes
```

Figure 5. Basic PE Encoding on Kali Linux

Now when we try to execute the file, Defender is still capable of detecting the encoding despite the fact there is no associated malicious payload (as there normally would be with PE Backdooring, where malicious payloads are injected to another PE).



Threat found - action needed. Severe
09/11/2020 01:14

Status: Active
Active threats have not been remediated and are running on your device.

Threat detected: Trojan:Win32/Meterpreter.A!cl
Alert level: Severe
Date: 09/11/2020 01:14
Category: Trojan
Details: This program is dangerous and executes commands from an attacker.

Learn more

Affected items:

file: C:\Users\Martin\Desktop\calc-exit-encoded.exe

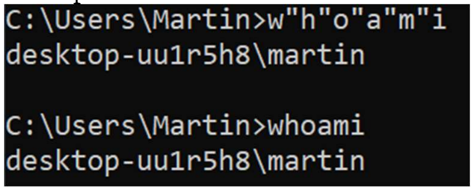
Figure 6. Defender detecting MSFVenom encoded PE

Among many techniques to test for defence evasion, one could be accomplished by implementing a crypter functionality. A crypter may store a malicious payload on disk, using a strong encryption algorithm. At runtime, this payload is decrypted and executed in memory, meaning it never touches disk. This is done using a stub method, which provides decryption functionality. The encryption key would typically be stored in bytes and could be obfuscated inside the code, or itself be hidden behind a complex algorithm. Typical encryption algorithms that could be used to achieve this include AES 128-bit and XOR encryption. Complex malware may utilise multiple layers of encryption, and stored the decryption keys on their Command and Control servers, to be retrieved at runtime.

The payloads themselves could be pre-made for test scenarios or could dynamically encrypt a PE file. C# is capable of making crypters, but given the complexity of their implementation it is likely the technique would be developed in C or C++ as there are many low-level aspects to be considered when dealing with this type of malware, such as process hollowing, virtual memory allocation and process handling at runtime.

Basic command-line obfuscation for Batch and PowerShell commands could also be implemented. There are many simple ways to obfuscate scripting commands, such as string concatenation that may evade getting detected when an organisation is filtering for keywords such as the command "whoami". Attacks may use this to determine what privileges they have for the account they have compromised. If the attacker obfuscated this slightly, and a SOC was only filtering their logs for string literals, it would not generate a notable alert. A user would be able to enter strings, or use template script commands which would then be obfuscated and executed.

Example:



```
C:\Users\Martin>w"h"o"a"m"i
desktop-uu1r5h8\martin

C:\Users\Martin>whoami
desktop-uu1r5h8\martin
```

Figure 7. Basic string manipulation to avoid being caught by filters looking for string literals

In addition to anti-virus, Endpoint Detection and Response is a term used to describe security suites, typically designed for large enterprises that are composed of several tools and utilities all usually wrapped into one package to provide protection for an organisation. EDR is typically more advanced and will be able to determine malicious API calls as opposed to anti-virus. This allows much greater potential in detecting malicious behaviour, rather than signature-based detection.

An EDR suite such as Symantec Endpoint Protection may provide utilities such as Download Insight (to block untrusted applications), Network Intrusion Prevention to prevent an endpoint from being compromised via man-in-the-middle attacks and advanced machine learning anti-virus (Symantec 2018). These advanced utilities are typically installed on each endpoint in an organisation, with log correlation to provide maximum protection. Symantec also provides services where their own SOC analysts will help analyse and monitor threats on a client's network using the EDR licences they have purchased.

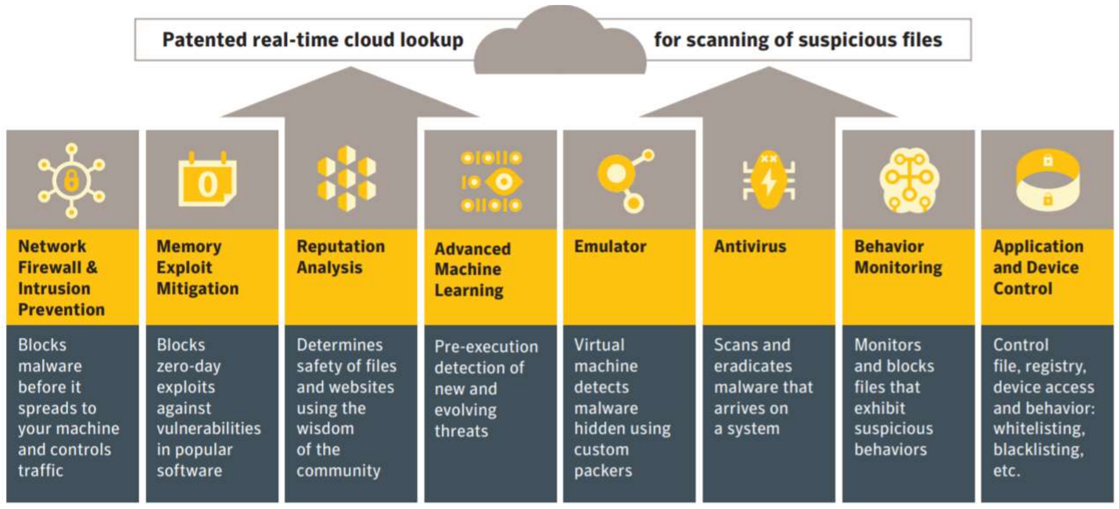

Patented real-time cloud lookup
for scanning of suspicious files
Network Firewall & Intrusion Prevention
Blocks malware before it spreads to your machine and controls traffic
Memory Exploit Mitigation
Blocks zero-day exploits against vulnerabilities in popular software
Reputation Analysis
Determines safety of files and websites using the wisdom of the community
Advanced Machine Learning
Pre-execution detection of new and evolving threats
Emulator
Virtual machine detects malware hidden using custom packers
Antivirus
Scans and eradicates malware that arrives on a system
Behavior Monitoring
Monitors and blocks files that exhibit suspicious behaviors
Application and Device Control
Control file, registry, device access and behavior: whitelisting, blacklisting, etc.

Figure 8. SEP's multi-layered prevention approach (Symantec Endpoint Protection 14 Data Sheet)

EDR has become a standard in how organisations defend themselves. These solutions are also costly, where it is estimated that the adoption of integrated cloud and on-premise solutions for EDR will be valued at $7.23 billion by 2026 (Statistics Marketing Research Consulting 2020).

This makes it highly important to test and ensure EDR configurations are functioning as expected, using adversary emulation tools.

4.6 Living off the Land Binaries

Red teamers and APT groups are heavily utilising "live-off-the-land" techniques. This means using built-in OS functionality to accomplish their goals, i.e. in the case of Windows using binaries that are native to the OS but also have a potentially malicious, unintended use case.

Advantages of using a signed binary are numerous, EDR and anti-virus products may apply implicit trust to these applications given that they are developed and (usually) signed by Microsoft.

Figure 9. LOLBAS Logo

The LOLBAS project is an open-source collaborative effort of security researchers to find and document LOLBINs or living-off-the-land binaries and scripts (Oddvar, M. 2020). Given that these binaries can be executed from a command line, they make excellent candidates for adversary emulation as the only development cost would be functions to execute the known binary.

MpCmdRun.exe for example, is used to manage settings in Windows Defender, however it could also be used to download malicious payloads. Whilst Defender will still scan the file to determine if it's malicious, however if a piece of malware has no known signatures it makes it much easier to download additional payloads unless SOC teams create and monitor for alerts relating to this binary.

```
MpCmdRun.exe -DownloadFile -url https://attacker.server/beacon.exe -path c:\\temp\\beacon.exe
```

Many techniques can be implemented using LOLBINs, including the top 10 threats to the education industry (Red Canary 2020) that could be implemented in WINTRE.

1. Scheduled Task - used to establish persistence, harvest data.
2. Windows Admin Shares - used for lateral movement, remote code execution.
3. Process Injection - used for defence evasion, e.g. TrickBot running via svchost.exe.
4. Scripting - many different attack vectors, applicable to all tactics.
5. Disabling Security Tools - used for stealth but also execution, can add exclusions to AV.
6. Service Execution - can be used for persistence, privilege escalation and as spyware.
7. Windows Management Instrumentation - used for reconnaissance and execution.
8. PowerShell - many different attack vectors, directly interface with .NET CLR.
9. Masquerading - used for phishing or obfuscating malware that is installed.
10. DLL Search Order Hijacking - primarily used for privilege escalation, to then establish persistence.

5 TECHNOLOGY STACK

5.1 C#

C# will be used as the main programming language as C# executables utilise the .NET framework, which is installed by default on all Windows 10 machines as of 2019 (Microsoft 2020). C# as a high-level language offers simplified ways of interacting with WinAPIs through the .NET framework that would've traditionally had to be accessed using a lower-level language such as C or Assembly. C# is object-oriented and type-safe, applications are highly scalable and easy to update and C# is packed with a rich plethora of libraries to enhance its interoperability. While not as fast as C or lower-level languages, compilation and execution speed are still sufficient for modern Windows applications. C# binaries do not require any 3rd party libraries to execute on modern Windows endpoints.

These reasons, while positive are also some of the main reasons threat actors have switched to using C#, to purposefully exploit its compatibility with modern Windows systems in order to develop malware. This matches our use case for the techniques needed to perform adversary simulation, many are essentially malware that will be running in a controlled and specific context. C# also reduces unnecessary, extra components that would be needed in a tool like this. Many similar applications require Linux based servers and have a much larger setup cost to an organisation. While a more complex infrastructure can benefit certain categories such as Command and Control for C2 testing, C# allows us to locally compile each individual test as its own executable. In more complex frameworks, this is not possible and the tester is forced to remove and update an agent each time they want to run a series of tests. Furthermore, tests that require server-side interaction can still be carried out easily using FOSS such as Netcat, as a listener to catch reverse shells which requires no configuration.

Ethical hackers and penetration testers have also primarily switched to a "Bring Your Own Land" philosophy, referencing the more traditional "Living off the Land" techniques, i.e. using standard Windows binaries and functionality to bypass security controls. Whilst very effective, some of these LOL techniques are now detected by anti-virus, EDR solutions or heavily monitored, such as CertUtil.exe. CertUtil.exe is a built-in Windows executable which can be used to download malicious payloads to a compromised system. An organisation should test to ensure they have visibility over this technique being used in their environment. Thus, C# has become a natural solution for many of the reasons outlined above. More offensive-tool developers have switched to C#, allowing them to create highly portable tools that can be compiled and ran easily on Windows.

5.2 XAML

The Extensible Application Markup Language simplifies the process creating a UI compatible with .NET Core applications. XAML allows each UI component to be declared and separated from run-time logic. XAML syntax is self-explanatory and easy to modify, even after many UI elements are joined together. XAML UI design is directly integrated with Visual Studio along with C# and Windows Presentation Forms. UI can be transferred with ease. This is useful, if in the future an application was moving from a desktop to a web or mobile implementation, there are great cost savings for the developers both in time and resources needed to migrate the UI to a new platform (Microsoft 2019).

XAML code is typically short and also supports vector as well as bitmap images; developers can also utilise its extensibility to create custom controls and elements.

5.3 PowerShell

PowerShell scripting can be utilised for practically every MITRE tactic or category of technique. PowerShell's intended usage is for automation and configuration management (Microsoft 2020). It provides a command-line shell to the end user from which they enter commands. This functionality is built upon the .NET Common Language Runtime. PowerShell is also extensible, allowing users to define cmdlets, this can be done directly using compiled code or scripts. PowerShell also has providers that provide direct access to important data stores. These data stores include the registry and file system, 2 very important and often targeted areas on a Windows computer. While intended for system administration, attackers may add malicious registry key entries or steal credentials from the registry.

PowerShell also has extensive console output display handling, allowing users to easily view console output in a readable format, preventing the need for formatting methods for every cmdlet. PowerShell also has a built-in Windows Defender module, this may allow some level of integration with Windows Defender as it will allow WINTRE to know which test binaries may have been detected by issuing cmdlets.

Example cmdlets (*Defender Module | Microsoft Docs*):

- Get-MpComputerStatus - Gets the status of antimalware software on the computer.
- Get-MpThreat - Gets the history of threats detected on the computer.
- Update-MpSignature - Updates the antimalware definitions on a computer.

C# binaries can also directly issue PowerShell cmdlets which will make testing offensive PowerShell more efficient. Frameworks such as PowerSploit written in PowerShell provide many modules for credential theft, process manipulation, discovery and data exfiltration. Red Canary's Threat Detection Report for 2020 shows that PowerShell is still a relevant technique for security testing that is being exploited despite many recent security controls developed for Windows by Microsoft.

PowerShell allows attackers to run compiled code in memory, execute many different types of downloaders, interact with .NET and WinAPIs as well as remote code execution capabilities (Harmj0y 2018).
For the purpose of adversary emulation, the following controls would have to considered when implementing tests (Metcalf, S. 2016):

- Script Block Logging - Log de-obfuscated PowerShell to the event log.
- Constrained Language Mode - Limits PowerShell functionality and cmdlet access.
- Antimalware Scan Interface - Request a dynamic analysis of executable content from different engines to determine if malware is present.

5.4 Windows Command Line Scripting

Similar to PowerShell scripting, all supported versions of Windows rely on built-in Win32 console commands to execute instructions from a command line. One of the first things a threat actor may attempt when gaining access to a Windows system is to determine their integrity level and local privileges. The ability to easily retrieve file system and OS information is appealing to attackers, as this will likely aid in further compromise of the local or neighbouring systems.

5.5 C

C would be used in a limited capacity for evaluating a handful of techniques, where coding them in C may be more beneficial than C# as C has greater functionality to interact with the Windows

OS at a lower-level. Payloads would be compiled using the Microsoft (R) C/C++ Optimizing Compiler.

5.6 C++

Similarly, to the C usage in this project, C++ would be used for evaluating select techniques where the C# equivalent may not be feasible or the C/C++ version may utilise different APIs more efficiently and therefore would be considered a sub technique and require separate detection analytics. Payloads would be compiled using the Microsoft (R) C/C++ Optimizing Compiler.

5.7 X86 ASSEMBLY

Assembly language will be used sparingly, similarly to the C and C++ techniques to be developed. The primary usage of this assembly code would be to use WinAPIs to write shellcode in order to be cached and executed in a memory buffer for techniques such as process injection. The shellcode could simply launch calculator or a message box. This would be sufficient for a technique in which process injection is demonstrated. The shellcode will be included with the software, when testing, virtual memory space would be allocated and then the buffer would be copied to that memory space and executed.

6 ALGORITHMS

The following algorithms are the main testing components of WINTRE based on draft techniques from various categories, most requiring interaction with Windows APIs to achieve successful implementation.

6.1 CREATE A MEMORY DUMP OF LSASS.EXE ON DISK

LSASS process memory contains a variety of user credentials after a user logs in, a memory dump of LSASS can be further analysed post-exfiltration to extract NTLM hashes that can in turn be used in pass-the-hash attacks or cracked for simply logging in as that user. Dozens of APT groups are known to make use of this technique due to how reliably Windows credentials can be recovered when preventative measures, and detection mechanisms are not in place.

Abstract implementation steps:

1. Determine target process ID, process name.
2. Invoke the MiniDumpWriteDump WinAPI to create a process dump (Nagel, G 2014).
3. Save this process dump on disk using a file stream.

6.2 SERVER EXECUTABLE BACKDOOR

Backdoors can come in many different formats, one way of accomplishing this is by replacing an existing service with a custom backdoor one. This typically requires administrator privileges, or service binaries with misconfigurations. This technique would also allow a user to achieve persistence.

Abstract implementation steps:

1. Ask user for service/service binary to replace.
2. Store the original name.
3. Try rename the service binary so when the computer is rebooted the service no longer points to it, if successful continue, otherwise access is likely denied.
4. Compile a service-executable with a payload to be ran as SYSTEM.
5. Paste the malicious svc-exe in place of the original. Restart the service.

6.3 DECRYPT VAULT CREDENTIALS

Useful credentials can also be extracted from the Windows Password Vault. Windows may store
Abstract implementation steps:

1. Using the Windows.Security.Credentials namespace, create an object to access the Password Vault.
2. Use built-in functionality to retrieve credentials from the vault, the credentials should be decrypted in the context of the user accessing them, as their user credentials will act as the master key.

6.4 WIRELESS AP CREDENTIALS

Retrieving AP credentials may help determine a pattern in user passwords or provide attackers with an additional way of accessing a network's resources.

Abstract implementation steps:

1. Using Window's built-in Network Shell utility netsh, to retrieve profile information of wireless access points.
2. Since a user will be logged in at this stage, they can run netsh and specify key=clear as an argument to retrieve their access point passwords.

7 Discretionary Algorithms

7.1 Crypter

A crypter may cryptographically alter an executable file to evade anti-virus. In our case, we will generate various different shellcodes using encryption such as AES or XOR. These will be stored encrypted on disk and decrypted at runtime using crypter stub functions (DTM 2016).

Abstract implementation steps:

1. Generate malicious payloads (MSFVenom or Assembly).
2. Encrypt the payloads.
3. Store payloads in PE in an encrypted format.
4. Allocate memory for payloads.
5. Decrypt the payloads at runtime and execute the buffer containing the payloads.

7.2 Process Injection

The purpose of process injection is to inject our payload into a remote process and have it executed from there. Process injection is a highly favourable technique as attackers may target whitelisted applications to run their payloads.

Abstract implementation steps:

1. Open a handle to the remote process.
2. Allocate memory in that process.
3. Write our payload to the allocated memory segment.
4. Create a remote threat and execute our payload.

7.3 PsExec SMB Lateral Movement

PsExec is a Sysinternals utility that acts as a telnet replacement to allow an administrator to execute processes on other systems. As useful as this tool was for administrators, attackers may also use it for lateral movement. Assuming that some credentials have been compromised, e.g. a local administrator's NTLM hash. Even without knowing the plaintext value of this hash, attackers can use the hash to create a reverse shell by manipulating services remotely via the SMB protocol. An attacker can create a service that will run commands, to download and execute arbitrary payloads. The Impacket toolkit has a Python version of this tool, allowing users to gain remote sessions from Linux distributions on Windows systems. This implementation would focus on moving from one Windows endpoint to another.

Abstract implementation steps (Borean, J. 2018):

1. Create an SMB connection to the victim's machine.
2. Copy a PE to be executed to the ADMIN$ share of the victim's machine (using plaintext passwords or NTLM hashes).
3. Bind the Windows Service Manager using RPC to the IPC$ share.
4. Create and configure a Windows Service pointing to the uploaded PE.
5. Connect to the named pipe attributed to the uploaded PE created by the service.
6. Start the process via the service, manage the standard output, standard error and standard input pipe.
7. Continue to read standard output/standard error pipe until execution has finished.

8 Similar Applications

- WINTRE - the proposed product solution to be developed.
- Scythe - Paid tool, designed with large enterprises in mind. Techniques are compiled into a separate agent for each testing session, communicating via a client-server model. This can be cumbersome and takes more time to test compared to local compilation. Scythe also encourages the user to run multiple techniques at once, making it harder to gather detection analytics to determine which technique was actually detected given that security control alerts can often be ambiguous.
- CALDERA - Utilises a client server model, using 3rd party open-source agents to run its techniques.
- PurpleSharp - A straightforward CLI program to test various MITRE techniques mainly based off of Windows APIs. No functionality for generating a report but implements a range lateral movement techniques.
- RTA - Red Team Automation provides a library of Python scripts to execute TTPs based on the MITRE ATT&CK framework. RTA requires a python installation and only supports command line tests. Focusing only on command-line tests limits testing capability.
- APT Simulator - Focuses on the running of scripts to perform simulation, similar to RTA but utilises batch scripts. Does not contain any reporting functionality and attempts to run all tests in one go.

Despite how useful many of these solutions are, most lack local compilation and or API based testing; this is less efficient for forensics and adversary simulation in general. The primary concern is that when the tests are not compiled locally, anti-virus or EDR present on the system may block execution outright to any of the tests. This is something I encountered during a Purple Team engagement on my internship when dealing with manual tests. This resulted in having to manually patch each binary on my workstation, which was ultimately very time-consuming. The alternative is to compile them locally so that there is an implicit trust by the security controls, to not block execution outright. Many of these solutions have also not been updated with the latest LOLBINs.

Application	OS	FOSS	Separate Binaries	Compiled Locally	No. of Techniques	MITRE Categories	Detailed Logging	Reports Generation	Detection Docs	Note Taking	API Based	Script-Only	Infrastructure	Campaigns
WINTRE	W	Yes	Yes	Yes	TBD	11	Yes	Yes	Yes	Yes	Yes	No	Client	Yes
Scythe	W/L	No	Yes	No	~50	14	Yes	Yes	Yes	Yes	?	No	Client-Server	Yes
CALDERA	W/L	Yes	Yes	No	50+	12	No	Yes	No	No	Yes	No	Client-Server	Yes
PurpleSharp	W	Yes	No	No	38	6	Yes	No	No	No	Yes	No	Client	No
RTA	W	Yes	No	No	50	6	No	No	No	No	No	Yes	Client	No
APT Simulator	W	Yes	No	No	26	6	No	No	No	No	No	Yes	Client	No

9 IMPLEMENTATION AND TESTING

The implementation of this tool, focuses on integrating it into a realistic environment where it can actually be tested and used to develop a report with meaningful findings. It is necessary to simulate a production environment, i.e. with active anti-virus and security controls in place. Below are three different test scenarios, two would be tested in a home lab with evaluation licenses for Windows as well as Windows Defender and BLUESPAWN EDR as the primary security controls. Since many small to medium-sized organisations may not have EDR, it is still relevant to separate these into 2 test scenarios.

All machines should be AD joined to a Windows Server domain controller to fully simulate a corporate environment. The second endpoint will be primarily utilised to demonstrate lateral movement via credential theft.

The third test scenario focuses on running selected tests on an isolated VM inside IT Carlow, with the collaboration of the Computing Services department.

9.1 TEST SCENARIO 1

The first test scenario would feature an Active Directory joined Windows 10 VM with Windows Defender fully enabled and default Firewall security settings enabled. The VM and virus definitions for Windows Defender would be fully updated to simulate a production environment. A report, using the analytics and findings from testing with WINTRE would be developed describing the impact of the TTPs that were not detected.

- Endpoint summary: 2x Windows 10 Enterprise
- Existing controls: Windows Defender

Tests will be running on the endpoint in this scenario. The Windows Server is a necessary component in simulating an AD joined network.

9.2 TEST SCENARIO 2

Similarly, to option 1, with the additional of open-source EDR. Would be more likely to alert on many TTPs due to behavioural detection and also more comparable with a corporate/enterprise network. BLUESPAWN is described as an "Active Defense and EDR" program which is specifically developed with the MITRE ATT&CK framework in mind. BLUESPAWN could be leveraged to further test TTPs to simulate an actual enterprise environment, where the BLUESPAWN agent will attempt to actively respond and mitigate certain techniques implemented in WINTRE.

Depending on whether or not lateral movement techniques could be implemented in the development process, additional endpoints could be added to test this.

- Endpoint summary: 2x Windows 10 Enterprise
- Existing controls: Windows Defender *and* BLUESPAWN EDR

BLUESPAWN will run on the Windows Server, with agents on both endpoints in addition to Windows Defender. BLUESPAWN will be able to correlate threat analytics to the Domain Controller, primarily for additional detection analytics.

9.3 Test Scenario 3

Dependent on permission and availability, this scenario involves testing on an isolated Windows 10 VM inside IT Carlow. This would involve a single machine matching the most common endpoint specification, it would also be AD joined so that Computing Services could correlate logs or detections. A list of techniques would be agreed upon and ran on the VM, these techniques could be matched to the 2020 Red Canary Threat Detection Report, which it summarises the most common techniques attackers are using against different industries, in this case Education. Using WINTRE, a report would be developed, documenting which techniques were not detected similarly to scenarios 1 and 2.

- Endpoint summary: 2x Windows 10 Education
- Existing controls: Windows Defender *and* ?

9.4 Test System Hardware Requirements

- OS: Microsoft Windows 10.0.18362 or later
- HDD: Application will likely be less than 50MB in size, any standard hard drive.
- RAM: 4GB
- CPU: Dual-core, x64 architecture

10 CONCLUSION

In conclusion, adversary emulation is becoming an increasingly necessary component of any organisation's security approach. This tool could also allow smaller organisations to benefit greatly by reducing the need to purchase a read team engagement or penetration test, which, are still incredibly valuable but quite expensive and vary in quality for smaller organisations who cannot afford exceptional red teamers on a regular basis.

Larger organisations also benefit from a tool like this, as there is no infrastructure requirement other than a Windows operating system which has the .NET framework installed to perform standard client-side tests. The .NET framework comes pre-installed nowadays avoiding a lengthy installation process. Larger enterprises are also the most likely to maintain a SIEM pipeline. This is essential to provide a SOC visibility over network endpoints during investigations in the case of a security incident.

There are numerous techniques that have been documented by researchers that could be included in an emulation framework such as WINTRE. Due to the choice in technology stack, the product would be easily expandable with any custom functionality at a later date while also allowing users to add their own simple tests that utilise the Windows command line (potential for new LOLBINs be added post-development).

A tool like this not only speeds up what is usually a tediously long development process, to test each script but also allows the testers to focus more on a results-oriented approach by preparing reports and detailed logging information in advance.

Ideally, previous groups of tests could be saved as campaigns, so that after a SOC has fine-tuned their detection analytics for those specific techniques, they can be re-tested with ease. This also allows for regular testing, whilst validating the increasing cost of existing security controls (Moore, S. 2017), it could also help decrease the cost of cyber insurance once an organisation shows they have a mature, well and regularly implemented adversary emulation program.

11 References

Nagel, G. (2014). *Getting started with PInvoke: calling MiniDumpWriteDump to create minidumps* [online]
Available from: https://gregsplaceontheweb.wordpress.com/2014/03/07/getting-started-with-pinvoke-calling-minidumpwritedump-to-create-minidumps/
[Accessed 4/10/2020]

Borean, J. (2018). *pypsexec . PyPI Run commands on a remote Windows host using SMB/RPC* [online]
Available from: https://pypi.org/project/pypsexec/
[Accessed 13/10/2020]

Wright, J. (2013). *How Browsers Store Your Passwords (and Why You Shouldn't Let Them)* [online]
Available from: http://raidersec.blogspot.com/2013/06/how-browsers-store-your-passwords-and.html
[Accessed 12/10/2020]

Microsoft. (2019). *XAML Overview - WPF .NET | Microsoft Docs* [online]
Available from: https://docs.microsoft.com/en-us/dotnet/desktop/wpf/fundamentals/xaml
[Accessed 28/10/2020]

Microsoft. (2020). *What is PowerShell? - PowerShell | Microsoft Docs* [online]
Available from: https://docs.microsoft.com/en-us/PowerShell/scripting/overview
[Accessed 29/10/2020]

Symantec. (2018). *Symantec Endpoint Protection 14* [online]
Available from: https://docs.broadcom.com/doc/endpoint-protection-14-en
[Accessed 27/10/2020]

Harmj0y. (2018). *PowerSploit - A PowerShell Post-Exploitation Framework* [online]
Available from: https://github.com/PowerShellMafia/PowerSploit/
[Accessed 24/10/2020]

Metcalf, S. (2016). *PowerShell Security: PowerShell Attack Tools, Mitigation, & Detection* [online]
Available from: https://adsecurity.org/?p=2921
[Accessed 26/10/2020]

Metcalf, S. (2016). *PowerShell Version 5 Security Enhancements* [online]
Available from: https://adsecurity.org/?p=2277
[Accessed 27/10/2020]

Mandiant. (2014). *APT1 Exposing One of China's Cyber Espionage Units* [online]
Available from: https://www.fireeye.com/content/dam/fireeye-www/services/pdfs/mandiant-apt1-report.pdf
[Accessed 11/10/2020]

Oddvar, M. (2020). *Living Off The Land Binaries and Scripts (and now also Libraries)* [online]
Available from: https://github.com/LOLBAS-Project/LOLBAS/blob/master/README.md
[Accessed 12/10/2020]

Netsurion. (2020). *The Assume Breach Paradigm* [online]
Available from: https://www.netsurion.com/articles/the-assume-breach-paradigm
[Accessed 4/10/2020]

MITRE. (2020). *APT1* [online]
Available from: https://attack.mitre.org/groups/G0006/
[Accessed 14/10/2020]

Redscan. (2018). *What is purple teaming and how can it strengthen your cyber security?* [online]
Available from: https://www.redscan.com/news/purple-teaming-can-strengthen-cyber-security/
[Accessed 7/10/2020]

Moore, S. (2017). *What's the Cost of Endpoint Detection & Response?* [online]
Available from: https://redcanary.com/blog/whats-the-cost-of-endpoint-detection-response-edr/
[Accessed 25/10/2020]

Solarwinds. (2020). *7 Best Practices for C# Logging (With Examples)* [online]
Available from: https://www.papertrail.com/solution/tips/7-best-practices-for-c-logging-with-examples/
[Accessed 14/10/2020]

Microsoft. (2020). *Compile code programmatically by using C# compiler - C# | Microsoft Docs* [online]
Available from: https://docs.microsoft.com/en-us/troubleshoot/dotnet/csharp/compile-code-using-compiler
[Accessed 15/10/2020].

Kojecký, L. (2014). *Compiling C# Code at Runtime* [online]
Available from: https://www.codeproject.com/tips/715891/compiling-csharp-code-at-runtime
[Accessed 5/10/2020].

MITRE. (2020). *FAQ | MITRE ATT&CK* [online]
Available from: https://attack.mitre.org/resources/faq/
[Accessed 1/10/2020].

Hexacorn. (2019). Hexacorn | *Blog UAC Bypass* [online]
Available from: https://www.hexacorn.com/blog/category/uac-bypass/
[Accessed 13/10/2020].

Microsoft. (2020). *CodeDomProvider Class* [online]
Available from: https://docs.microsoft.com/en-us/dotnet/api/system.codedom.compiler.codedomprovider
[Accessed 21/10/2020].

Spotless. (2020). *Privilege Escalation - Red Teaming Experiments* [online]
Available from: https://www.ired.team/offensive-security/privilege-escalation
[Accessed 19/10/2020].

DTM. (2016). *Crypters - Instruments of the Underground* [online]
Available from: https://0x00sec.org/t/crypters-instruments-of-the-underground/386
[Accessed 20/10/2020].

Microsoft. (2016). *Credentials Processes in Windows Authentication.* [online]
Available from: https://docs.microsoft.com/en-us/windows-server/security/windows-authentication/credentials-processes-in-windows-authentication
[Accessed 15/10/2020].

Red Canary. (2020). *Top MITRE ATT&CK Techniques by Industry - Red Canary Report* [online]
Available from: https://redcanary.com/threat-detection-report/threat-detection-report-by-industry/
[Accessed 12/10/2020].

CISA. (2020). *Ransomware Activity Targeting the Healthcare and Public Health Sector* [online]
Available from: https://us-cert.cisa.gov/ncas/alerts/aa20-302a
[Accessed 3/10/2020].

Deenathayalan, M. (2019). *How to Create Word Document in C#* [online]
Available from: https://www.c-sharpcorner.com/UploadFile/muralidharan.d/how-to-create-word-document-using-C-Sharp/
[Accessed 13/10/2020].

MITRE. (2018). *Our History | The MITRE Corporation* [online]
Available from: https://www.mitre.org/about/our-history
[Accessed 1/10/2020].

Stratistics Market Research Consulting Pvt Ltd. (2020). *Endpoint Security - Global Market Outlook (2019-2027)* [online] Available from: https://www.globenewswire.com/news-release/2020/08/14/2078616/0/en/Global-Endpoint-Security-Market-Outlook-2019-to-2027-Featuring-Cisco-Microsoft-Malwarebytes-Among-Others.html
[Accessed 4/10/2020].