MOBIA

MOBIA is a business technology integrator with over thirty years of experience and 500+ employees across Canada and USA. Our talented bench of technical engineers and trusted advisors deliver process improvements and business transformations within our core pillars of Cloud, Infrastructure, Software Development, Cybersecurity and Broadband & Wireless Services.

Our inside-out approach allows us to understand business challenges from deep within the organization, mapping the impact as it ripples outward. This insight enables us to create future-proof solutions that maximize results and repeatedly exceed our client's expectations.

Adversary Simulation Workshop

Lab Guide:
Chaining Use Case Development

Objective:

Run multiple techniques to simulate an attack. Additionally, the goal of this lab will be to simulate the following scenario:

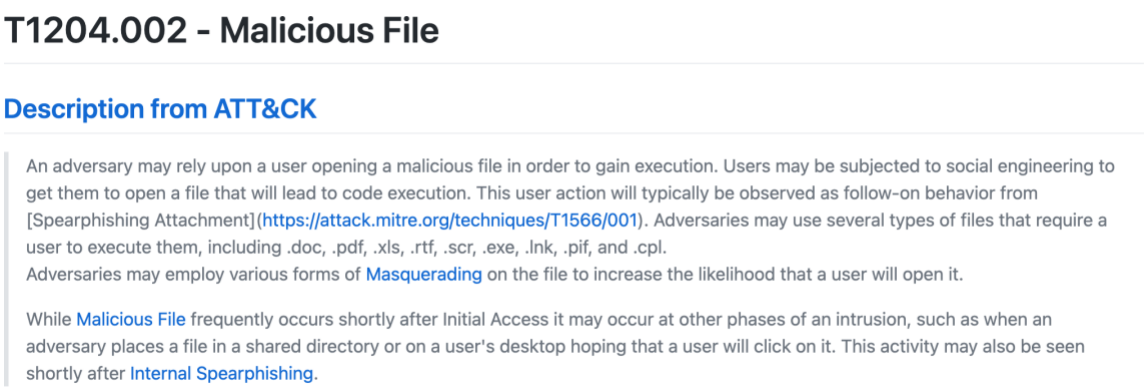
- STEP 1: Initial attack launched via Word Doc as unprivileged user
- STEP 2: Resultant process adds persistence via scheduled task
- STEP 3: Perform discovery (local service exploits)
- STEP 4: Use discovery to escalate privileges to SYSTEM
- STEP 5: Create new local account

Instructions:

The first step in this exercise will be to use to ATT&CK Navigator to identify the techniques for each steps. Once we've identified the steps we'll review the atomic test for that technique to see if it fits our need.

Initial Access

Step 1 aligns well with T1204 and T1566.001. Open the T1204.002 and T1566.001 descriptions review the tests available.



T1204.002 - Malicious File

Description from ATT&CK

> An adversary may rely upon a user opening a malicious file in order to gain execution. Users may be subjected to social engineering to get them to open a file that will lead to code execution. This user action will typically be observed as follow-on behavior from [Spearphishing Attachment](https://attack.mitre.org/techniques/T1566/001). Adversaries may use several types of files that require a user to execute them, including .doc, .pdf, .xls, .rtf, .scr, .exe, .lnk, .pif, and .cpl.
> Adversaries may employ various forms of Masquerading on the file to increase the likelihood that a user will open it.
>
> While Malicious File frequently occurs shortly after Initial Access it may occur at other phases of an intrusion, such as when an adversary places a file in a shared directory or on a user's desktop hoping that a user will click on it. This activity may also be seen shortly after Internal Spearphishing.

T1566.001 - Spearphishing Attachment

Description from ATT&CK

> Adversaries may send spearphishing emails with a malicious attachment in an attempt to gain access to victim systems. Spearphishing attachment is a specific variant of spearphishing. Spearphishing attachment is different from other forms of spearphishing in that it employs the use of malware attached to an email. All forms of spearphishing are electronically delivered social engineering targeted at a specific individual, company, or industry. In this scenario, adversaries attach a file to the spearphishing email and usually rely upon [User Execution](https://attack.mitre.org/techniques/T1204) to gain execution.
> There are many options for the attachment such as Microsoft Office documents, executables, PDFs, or archived files. Upon opening the attachment (and potentially clicking past protections), the adversary's payload exploits a vulnerability or directly executes on the user's system. The text of the spearphishing email usually tries to give a plausible reason why the file should be opened, and may explain how to bypass system protections in order to do so. The email may also contain instructions on how to decrypt an attachment, such as a zip file password, in order to evade email boundary defenses. Adversaries frequently manipulate file extensions and icons in order to make attached executables appear to be document files, or files exploiting one application appear to be a file for a different one.

There are a multitude of options for us to choose from here but we need something that will download our payload and execute. Under T1204 test 6 will download a payload from the internet and execute:

Atomic Test #6 - Excel 4 Macro

This module creates an Excel 4 Macro (XLM) enabled spreadsheet and executes it. The XLM will first write a "malicious" VBS file to %TEMP%, then execute this file. The VBS will download Process Explorer to the same directory (%TEMP%) and exec.

A note regarding this module. By default, this module will pull the current username from the system and places it into the macro. If you'd like to utilize the "=GET.WORKSPACE(26)" method, that many maldoc authors use, you will need to ensure that the User Name associated with Excel matches that of the local system. This username can be found under Files -> Options -> Username

Supported Platforms: Windows

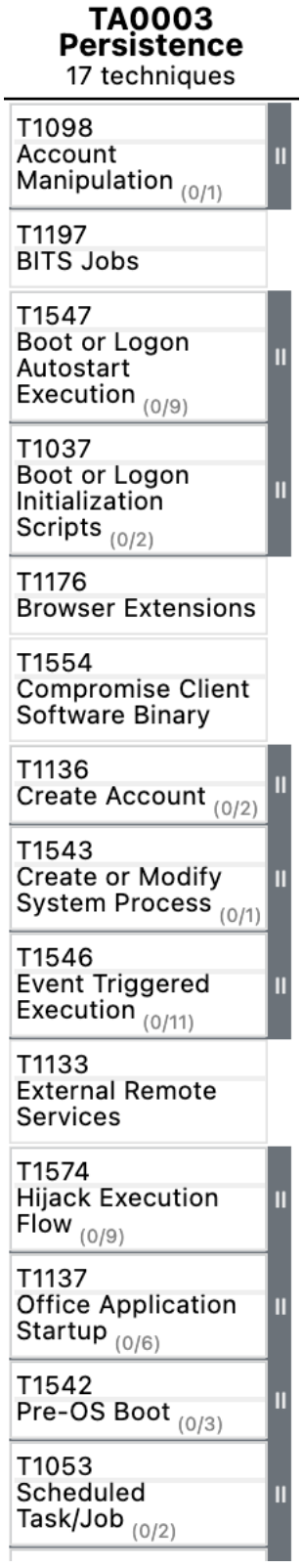
Inputs:

Name	Description	Type	Default Value
download_url	Download URL	String	https://live.sysinternals.com/procexp.exe
uname	Username for pathing	String	$env:Username

The input option download_url can be overridden with our payload so will be perfect for our test.

Persistence

Step 2 requires us to provide persistence for our attack. We're going to be a unprivileged user at this point so we'll need to ensure our persistence technique takes that into account. Let's look at the most common persistence techniques. Head back to the ATT&CK Navigator and find the Persistence tactic.


TA0003
Persistence
17 techniques
T1098
Account Manipulation (0/1)
T1197
BITS Jobs
T1547
Boot or Logon Autostart Execution (0/9)
T1037
Boot or Logon Initialization Scripts (0/2)
T1176
Browser Extensions
T1554
Compromise Client Software Binary
T1136
Create Account (0/2)
T1543
Create or Modify System Process (0/1)
T1546
Event Triggered Execution (0/11)
T1133
External Remote Services
T1574
Hijack Execution Flow (0/9)
T1137
Office Application Startup (0/6)
T1542
Pre-OS Boot (0/3)
T1053
Scheduled Task/Job (0/2)

The first option T1098 in the list aligns well to our Step 5 so let's keep that one in mind.

T1098 - Account Manipulation

Description from ATT&CK

> Adversaries may manipulate accounts to maintain access to victim systems. Account manipulation may consist of any action that preserves adversary access to a compromised account, such as modifying credentials or permission groups. These actions could also include account activity designed to subvert security policies, such as performing iterative password updates to bypass password duration policies and preserve the life of compromised credentials. In order to create or manipulate accounts, the adversary must already have sufficient permissions on systems or the domain.

We need something that gives us task scheduler execution. T1053.005 looks like it does exactly what we need.

T1053.005 - Scheduled Task

Description from ATT&CK

> Adversaries may abuse the Windows Task Scheduler to perform task scheduling for initial or recurring execution of malicious code. There are multiple ways to access the Task Scheduler in Windows. The `schtasks` can be run directly on the command line, or the Task Scheduler can be opened through the GUI within the Administrator Tools section of the Control Panel. In some cases, adversaries have used a .NET wrapper for the Windows Task Scheduler, and alternatively, adversaries have used the Windows netapi32 library to create a scheduled task.
> The deprecated at utility could also be abused by adversaries (ex: At (Windows)), though `at.exe` can not access tasks created with `schtasks` or the Control Panel.
>
> An adversary may use Windows Task Scheduler to execute programs at system startup or on a scheduled basis for persistence. The Windows Task Scheduler can also be abused to conduct remote Execution as part of Lateral Movement and or to run a process under the context of a specified account (such as SYSTEM).

We won't be able to option escalation at this point, that's going to come later. For now a scheduled task as our unprivileged will fit right in to our plan.

Test #2 includes an option to schedule a task and provide an executable to run at a scheduled time daily.

Atomic Test #2 - Scheduled task Local

Upon successful execution, cmd.exe will create a scheduled task to spawn cmd.exe at 20:10.

Supported Platforms: Windows

Inputs:

Name	Description	Type	Default Value
task_command	What you want to execute	String	C:\windows\system32\cmd.exe
time	What time 24 Hour	String	72600

Discovery

Let's move on to Step 3, this one is about discovering system details. These details we'll use to gain privileged access to the machine. Time again to head to ATT&CK Navigator and locate the Discovery tactic:

There are a lot of discovery options here as well but we want to find something that will help us specifically target weak service configuration as that's a good way to gain SYSTEM level access to a machine.

T1082 - System Information Discovery

Description from ATT&CK

> An adversary may attempt to get detailed information about the operating system and hardware, including version, patches, hotfixes, service packs, and architecture. Adversaries may use the information from [System Information Discovery](https://attack.mitre.org/techniques/T1082) during automated discovery to shape follow-on behaviors, including whether or not the adversary fully infects the target and/or attempts specific actions.
> Tools such as Systeminfo can be used to gather detailed system information. A breakdown of system data can also be gathered through the macOS `systemsetup` command, but it requires administrative privileges.
>
> Infrastructure as a Service (IaaS) cloud providers such as AWS, GCP, and Azure allow access to instance and virtual machine information via APIs. Successful authenticated API calls can return data such as the operating system platform and status of a particular instance or the model view of a virtual machine.(Citation: Amazon Describe Instance)(Citation: Google Instances Resource) (Citation: Microsoft Virutal Machine API)

Reviewing the tests for T1082 there's nothing that really dynamic enough for us to customize on our chain. Griffon (#9) might get us some details that we want, if we don't like that output we'll have to have some type of vbs file to execute to get what we want.

Atomic Test #9 - Griffon Recon

Griffon is a sophisticated tool believed to be in use by one of more "APT" groups. This atomic is for detecting, specifically, the reconnaissance part of the tool. This script used here was reduced by security researcher Kirk Sayre (github.com/kirk-sayre-work/1a9476e7708ed650508f9fb5adfbad9d), and it gives the exact same recon behavior as the original (minus the C2 interaction). For more information see also e.g. https://malpedia.caad.fkie.fraunhofer.de/details/js.griffon and https://attack.mitre.org/software/S0417/

Supported Platforms: Windows

Inputs:

Name	Description	Type	Default Value
vbscript	Path to sample script	String	PathToAtomicsFolder\T1595.002\src\griffon_recon.vbs

A promising technique is T1007. It does look like what we want to do but there's only a customized output.

T1007 - System Service Discovery

Description from ATT&CK

> Adversaries may try to get information about registered services. Commands that may obtain information about services using operating system utilities are "sc," "tasklist /svc" using [Tasklist](https://attack.mitre.org/software/S0057), and "net start" using [Net](https://attack.mitre.org/software/S0039), but adversaries may also use other tools as well. Adversaries may use the information from [System Service Discovery](https://attack.mitre.org/techniques/T1007) during automated discovery to shape follow-on behaviors, including whether or not the adversary fully infects the target and/or attempts specific actions.

Atomic Tests

- Atomic Test #1 - System Service Discovery
- Atomic Test #2 - System Service Discovery - net.exe

Atomic Test #2 - System Service Discovery - net.exe

Enumerates started system services using net.exe and writes them to a file. This technique has been used by multiple threat actors.

Upon successful execution, net.exe will run from cmd.exe that queries services. Expected output is to a txt file in c:\Windows\Temp\service-list.txt.s

Supported Platforms: Windows

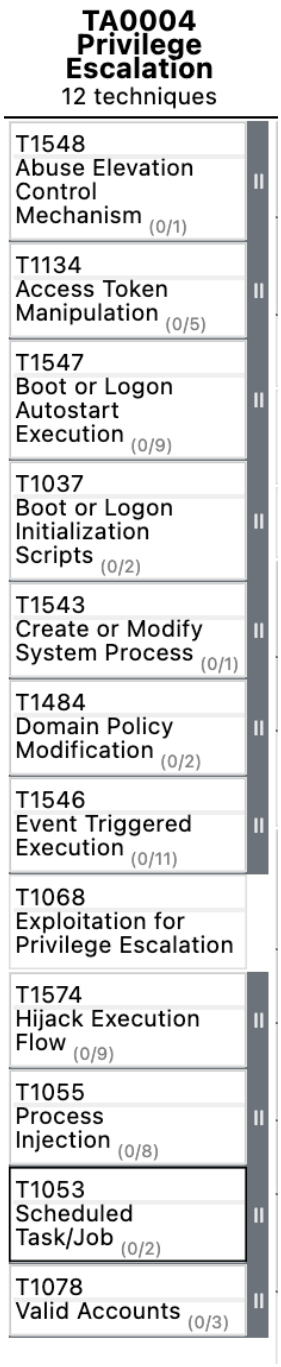
Inputs:

Name	Description	Type	Default Value
output_file	Path of file to hold net.exe output	Path	C:\Windows\Temp\service-list.txt

This one we'll have to lead until later. Let's move on to Step 4.

Privilege Escalation

Step 4 requires us to use the discovery we gathered from the system to privilege escalate to SYSTEM. Back to ATT&CK Navigator this time to the Privilege Escalation tactic:



Path exploit would work well for what we need. T1574.009 provides a method for us to test unquoted service path.

T1574.009 - Path Interception by Unquoted Path

Description from ATT&CK

> Adversaries may execute their own malicious payloads by hijacking vulnerable file path references. Adversaries can take advantage of paths that lack surrounding quotations by placing an executable in a higher level directory within the path, so that Windows will choose the adversary's executable to launch.
> Service paths (Citation: Microsoft CurrentControlSet Services) and shortcut paths may also be vulnerable to path interception if the path has one or more spaces and is not surrounded by quotation marks (e.g., `C:\unsafe path with space\program.exe` vs. `"C:\safe path with space\program.exe"`). (Citation: Help eliminate unquoted path) (stored in Windows Registry keys) An adversary can place an executable in a higher level directory of the path, and Windows will resolve that executable instead of the intended executable. For example, if the path in a shortcut is `C:\program files\myapp.exe`, an adversary may create a program at `C:\program.exe` that will be run instead of the intended program. (Citation: Windows Unquoted Services) (Citation: Windows Privilege Escalation Guide)
>
> This technique can be used for persistence if executables are called on a regular basis, as well as privilege escalation if intercepted executables are started by a higher privileged process.

Atomic Test #1 - Execution of program.exe as service with unquoted service path

When a service is created whose executable path contains spaces and isn't enclosed within quotes, leads to a vulnerability known as Unquoted Service Path which allows a user to gain SYSTEM privileges. In this case, if an executable program.exe in C:\ exists, C:\program.exe will be executed instead of test.exe in C:\Program Files\subfolder\test.exe.

Supported Platforms: Windows

Inputs:

Name	Description	Type	Default Value
service_executable	Path of the executable used for the service and as the hijacked program.exe	path	PathToAtomicsFolder\T1574.009\bin\WindowsServiceExample.exe

It's definitely an option but it doesn't out of the box have quite enough flexibility. Maybe we need another way to view the available tests to see where we might focus. We can use this Windows Index to search by test name.

A common service configuration issue with misconfigured permission for the service directory. Searching the tests for permissions we find a few matches:

1108 lines (1096 sloc) | 77.8 KB

- T1053.005 Scheduled Task
  - Atomic Test #1: Scheduled Task Startup Script [windows]
  - Atomic Test #2: Scheduled task Local [windows]
  - Atomic Test #3: Scheduled task Remote [windows]
  - Atomic Test #4: Powershell Cmdlet Scheduled Task [windows]
  - Atomic Test #5: Task Scheduler via VBA [windows]
- T1053 Scheduled Task/Job CONTRIBUTE A TEST
- T1546.002 Screensaver
  - Atomic Test #1: Set Arbitrary Binary as Screensaver [windows]
- T1547.005 Security Support Provider
  - Atomic Test #1: Modify SSP configuration in registry [windows]
- T1574.010 Services File Permissions Weakness CONTRIBUTE A TEST
- T1574.011 Services Registry Permissions Weakness
  - Atomic Test #1: Service Registry Permissions Weakness [windows]
  - Atomic Test #2: Service ImagePath Change with reg.exe [windows]
- T1547.009 Shortcut Modification
  - Atomic Test #1: Shortcut Modification [windows]
  - Atomic Test #2: Create shortcut to cmd in startup folders [windows]
- T1055.003 Thread Execution Hijacking CONTRIBUTE A TEST

T1574.010 looks like it would do exactly what we need…but wait no existing test is available! Looks like we either have to change our test or we have to write a test and contribute (more on that later).

Privilege Escalation – Part 2

Finally, Step5, creating a local account. T1078.003 exist and does do what we need…sort of.

T1078.003 - Local Accounts

Description from ATT&CK

> Adversaries may obtain and abuse credentials of a local account as a means of gaining Initial Access, Persistence, Privilege Escalation, or Defense Evasion. Local accounts are those configured by an organization for use by users, remote support, services, or for administration on a single system or service.
> Local Accounts may also be abused to elevate privileges and harvest credentials through OS Credential Dumping. Password reuse may allow the abuse of local accounts across a set of machines on a network for the purposes of Privilege Escalation and Lateral Movement.

Atomic Tests

- Atomic Test #1 - Create local account with admin priviliges

Atomic Test #1 - Create local account with admin priviliges

After execution the new account will be active and added to the Administrators group

Supported Platforms: Windows

Attack Commands: Run with `command_prompt` ! Elevation Required (e.g. root or admin)

```
net user art-test /add
net user art-test Password123!
net localgroup administrators art-test /add
```

For this one to work we'd need to be able to execute the test itself, there's not download, path or user account. But we have a plan and a little bit of work to do. Our test plan looks like this:

Execution Plan

- STEP 1: T1566.001 Initial attack launched via Word Doc as unprivileged user
- STEP 2: T1053.005 Resultant process adds persistence via scheduled task
- STEP 3: T1082 or T1007 Perform discovery (local service exploits)
- STEP 4: T1574.009 or T1574.010 Use discovery to escalate privileges to SYSTEM
- STEP 5: T1078.003 Create new local account

Our plan is going to require some customized test and possibly a little script writing. Especially if we want to emulate our original concept.