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:
Execute Atomic Tests

Objective:

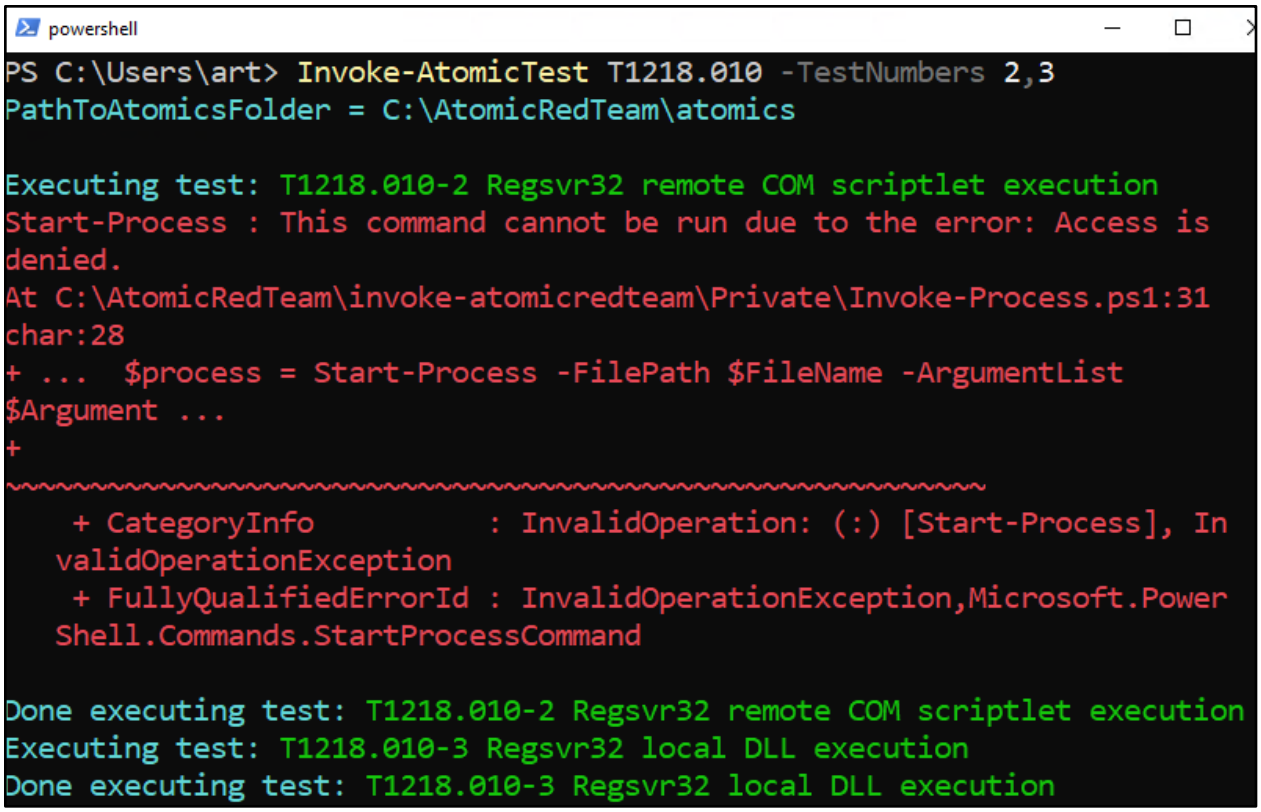
Use the execution framework to execute atomic tests.

Instructions:

There are two methods that can be used to execute atomic tests using the PowerShell execution framework, local and remote. In the “local” method, the framework executes the atomic tests on the same machine that the framework is installed on. Whereas with remote, the execution framework is installed on one machine and it is used to execute atomic tests on a remote machine through a PowerShell remoting session. You can read more about executing atomic tests remotely on the Wiki here.

We actually gave you a sneak peak at executing a test in the previous lab but we will show some additional examples here. We can execute multiple tests by name or by number using a comma separated list.

```
Invoke-AtomicTest T1218.010 -TestNumbers 2,3
```

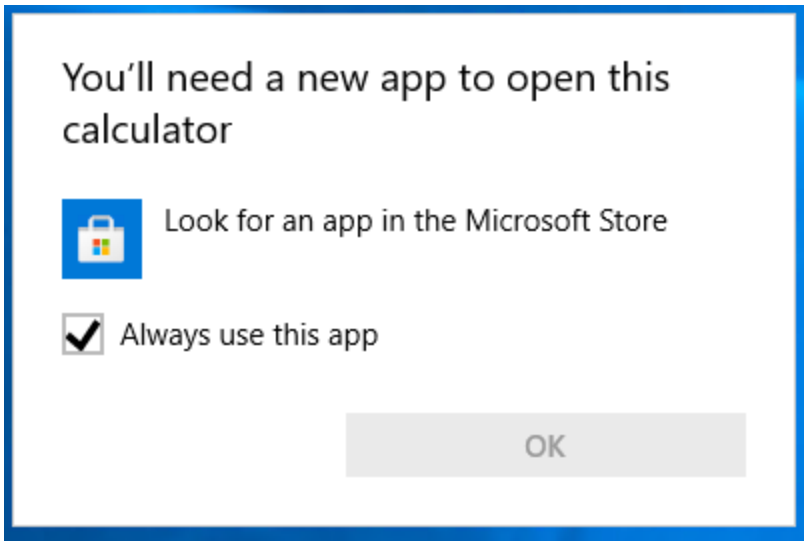


Here we ran [tests 2 and 3 of T1218.010]. We could have specified these two tests by name instead of number as shown below.

```
Invoke-AtomicTest T1218.010 -TestNames "Regsvr32 remote COM scriptlet execution", "Regsvr32 local DLL execution"
```

The output shows an error while running test #2 with an “Access is denied” message. This could happen if the test requires execution from an elevated prompt and we didn’t provide one. However, in this case this error is a result of Windows Defender blocking this attack technique. You might notice the popup from Windows Defender in the corner stating that it blocked something malicious.

Also, interesting note that these two tests attempt to launch the calculator app which isn’t installed on Windows 10 by default any more. So for successful test execution you’ll just see a popup like the one below instead of actually seeing the calculator launch.



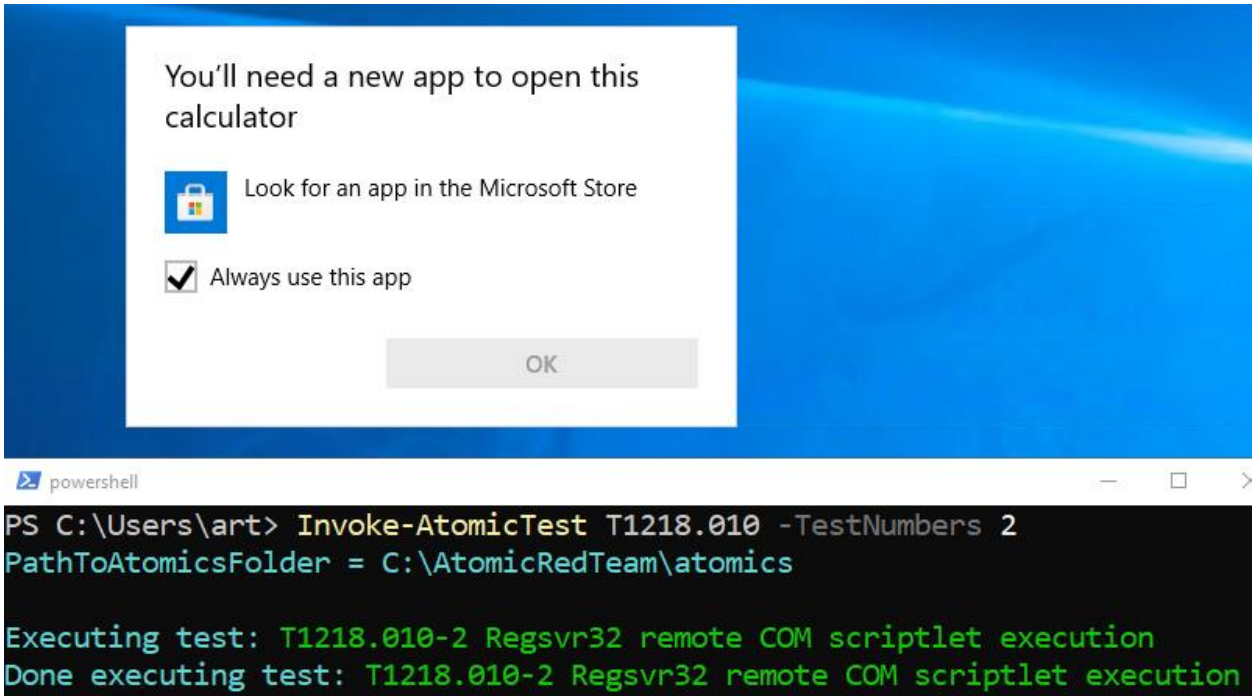
Just for fun, let's disable Windows Defender for a minute and confirm that it was blocking successful execution of test #2. We can do this by running the following command from an elevated PowerShell prompt (Run as administrator).

```
Set-MpPreference -DisableRealtimeMonitoring $true
```

Click here for additional details on disabling Defender if needed.

Now try executing test #2 which was previously blocked by Windows Defender and you will see that the test executes successfully and executes calc.exe.

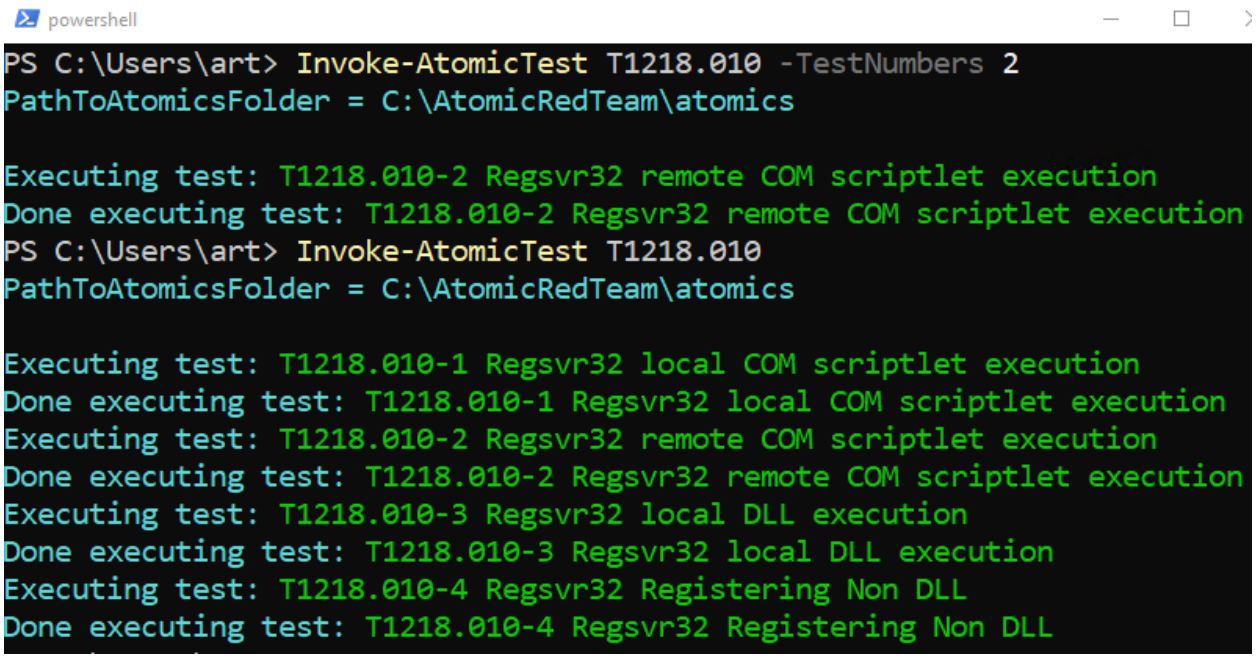
```
Invoke-AtomicTest T1218.010 -TestNumbers 2
```



You may choose to turn Windows Defender back on at this point depending on your testing preferences. However, I suggest running all atomics with AV disabled to ensure that if something does get past AV, it will still be detected. This is because there is always a way to bypass AV and you want to operate under the principle that "prevention is ideal, but detection is a must".

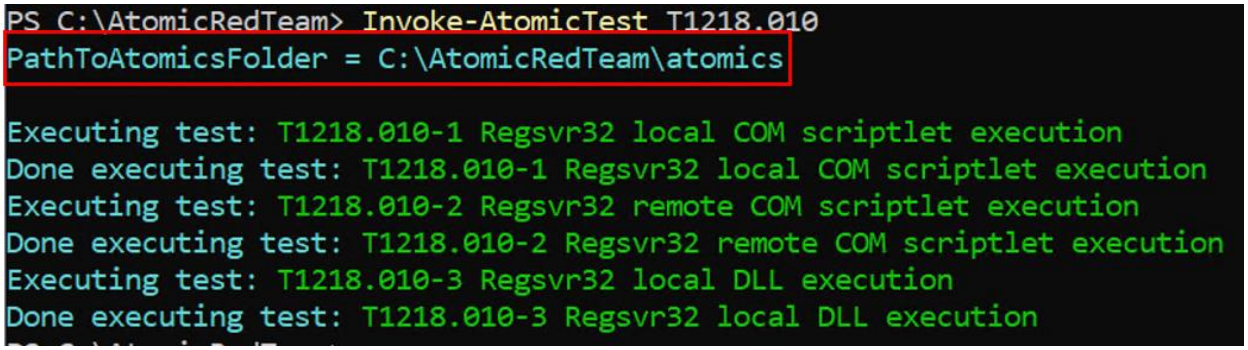
Executing each test individually might be time consuming. If you want to run all tests within a specific T#, you can just specify the T#.

```
Invoke-AtomicTest T1218.010
```



All four tests in the T1218.010 technique ran successfully with a single command.

Notice that when we run any test, the first line output to the screen is the “PathToAtomicsFolder”.



The default installation location for both the execution framework and the atomics folder is “C:\AtomicRedTeam” on Windows or “~/AtomicRedTeam” on Linux\macOS. When the execution framework runs, it assumes that the

atomic test definition YAML files are inside T# folders at the default location of “C:\AtomicRedTeam\atomics”.

If you want to run atomics from another location, you need to tell the execution framework where to find the atomic test definition YAML files by using the “PathToAtomicsFolder” flag.

```
Invoke-AtomicTest T1218.010 -PathToAtomicsFolder C:\my-private-atomics
```

This would allow you to run your own custom/private atomics from a different folder.

If you don’t want to have to specify your custom path to the atomics folder for every test execution, you can add the following to your PowerShell profile.

```
$PSDefaultParameterValues = @{"Invoke-AtomicTest:PathToAtomicsFolder"="C:\my-private-atomics"}
```

Each atomic test has a unique identifier called the GUID. You won’t see the GUID in the markdown file or with the “ShowDetailsBrief” flag but you will find it in the YAML files and in the details of each test.

```
PS C:\Users\art> Invoke-AtomicTest T1218.010 -TestNumbers 1 -ShowDetails
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

[********BEGIN TEST********]
Technique: Signed Binary Proxy Execution: Regsvr32 T1218.010
Atomic Test Name: Regsvr32 local COM scriptlet execution
Atomic Test Number: 1
Atomic Test GUID: 449aa403-6aba-47ce-8a37-247d21ef0306
Description: Regsvr32.exe is a command-line program used to register and
unregister OLE controls. Upon execution, calc.exe will be launched.
```

You can execute tests by specifying a GUID instead of a test name/number as follows.

```
Invoke-AtomicTest T1218.010 -TestGuids 449aa403-6aba-47ce-
8a37-247d21ef0306
```

```
PS C:\Users\art> Invoke-AtomicTest T1218.010 -TestGuids 449aa403-6aba-47c
e-8a37-247d21ef0306
PathToAtomicsFolder = C:\AtomicRedTeam\atomics

Executing test: T1218.010-1 Regsvr32 local COM scriptlet execution
Done executing test: T1218.010-1 Regsvr32 local COM scriptlet execution
```

You might be asking yourself “Why in the world would I do that?”. While the GUID is not an easy thing to type on the fly, it comes in handy when writing scripts to automate the execution of multiple tests to ensure that the script always runs the same test. The test number is simply an indication of the order the atomic test definition shows up in the YAML file and this can change as tests are added and removed. Also the test name can change as developers decide to make the name more descriptive or fix typo’s.

Now a word on test execution timeout. The execution framework starts a hidden command window to execute each test but redirects its output to the window you are executing from. If the execution takes longer than two minutes by default, the process and its children will be terminated. You can specify an alternate timeout period with the “TimeoutSeconds” flag.

```
Invoke-AtomicTest T1218.010 -TestNumbers 1 -TimeoutSeconds
15
```

Not all processes that are spawned during a test will be terminated after the timeout depending on how they are launched.

While not recommended under normal circumstances, you can also execute all atomic tests with the “Invoke-AtomicTest All” command.

By default, the details of which atomics were run and at what time, is logged to the temp directory in a file called “Invoke-AtomicTest-ExecutionLog.csv”. You can view this file with the following command.

```
cat $env:TEMP\Invoke-AtomicTest-ExecutionLog.csv
```

However, viewing the csv log file in Excel is easier to read as shown below. The Excel application is not included on the lab machine so you won’t be able to open this view from there.

Execution Time (UTC)	Execution Time (Local)	Technique	Test N	Test Name	Hostname	Username	GUID
2020-06-18T23:39:23Z	2020-06-18T23:39:23	T1016	5	List Open Egress Ports	art-vm0	art-vm0\art	4b467538-f102
2020-06-18T23:54:07Z	2020-06-18T23:54:07	T1016	5	List Open Egress Ports	art-vm0	art-vm0\art	4b467538-f102
2020-06-19T00:04:13Z	2020-06-19T00:04:13	T1485	1	Windows - Overwrite f	art-vm0	art-vm0\art	476419b5-aebf
2020-06-19T00:22:46Z	2020-06-19T00:22:46	T1485	1	Windows - Overwrite f	art-vm0	art-vm0\art	476419b5-aebf
2020-06-19T00:38:42Z	2020-06-19T00:38:42	T1218.001	1	Compiled HTML Help L	art-vm0	art-vm0\art	5cb87818-0d7
2020-06-19T00:39:03Z	2020-06-19T00:39:03	T1218.001	2	Compiled HTML Help F	art-vm0	art-vm0\art	0f8af516-9818
2020-06-19T00:48:00Z	2020-06-19T00:48:00	T1218.001	1	Compiled HTML Help L	art-vm0	art-vm0\art	5cb87818-0d7
2020-06-19T00:48:12Z	2020-06-19T00:48:12	T1218.001	2	Compiled HTML Help F	art-vm0	art-vm0\art	0f8af516-9818

If you would like to specify a different path for your log file, you can do that with the “ExecutionLogPath” flag.

```
Invoke-AtomicTest T1218.010 -ExecutionLogPath
'C:\Users\art\log.csv'
```

Or to set the execution log path permanently for all executions, you could set it permanently by adding the following line to your PowerShell profile.

```
$PSDefaultParameterValues = @{"Invoke-
AtomicTest:ExecutionLogPath"="C:\Users\art\log.csv"}
```

This covers all the basic options for executing an atomic test. In the follow on labs, we will learn how to specify custom input arguments and clean up after test execution.

End of Lab