Real-time detection scenarios in Active Directory environments

By Scarred Monk

Published: 2022-05-06 · Archived: 2026-04-05 17:17:51 UTC

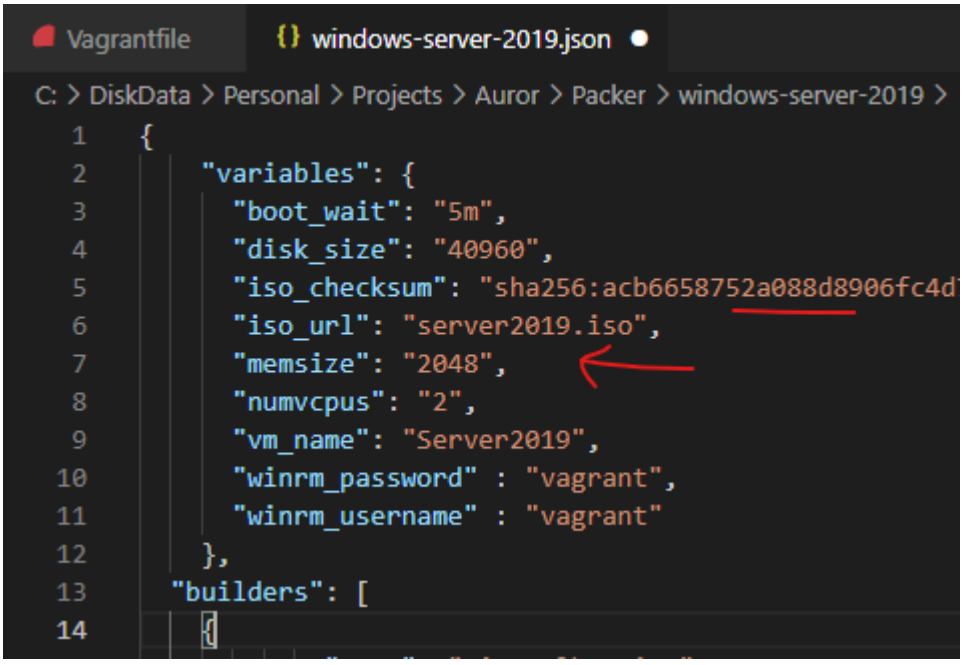
Introduction

Recently, I joined the 2nd session of the course “3 Machine Labs” under the Auror Project, an initiative led by Sudarshan Pisupati where we were doing deep-dive in Active Directory. I liked the concept that this course is focussed on challenge-based learning - where each session is followed by a challenge which will ultimately help in understanding the core concepts for the members. And best thing is that, this project is open for anyone to join. Infact, it was started with a goal to drive meaningful infosec career outcomes for its members by creating community-driven opportunities.

In the first session of “3 Machine Labs”, the challenge was to automate the 3 machine lab environment for testing different scenarios. To achieve this, we can use `Packer` and `Vagrant` and there are other ways to automate the whole process as well. Packer will help to create a golden image from a particular ISO file. Then, using the configuration scripts, we can use `Vagrant` to provision the virtual machine inside VirtualBox.

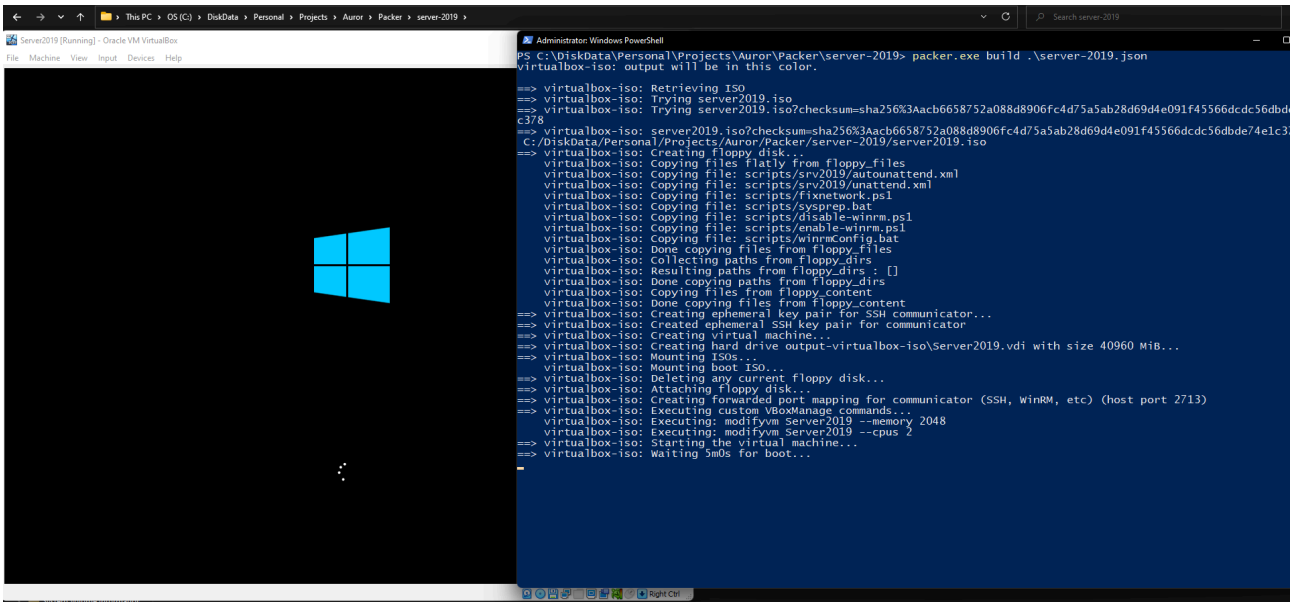
- Packer — https://www.packer.io/downloads
- Vagrant — https://www.vagrantup.com/downloads
- Virtualbox — https://www.virtualbox.org/wiki/Downloads

We will need to configure the packer configuration files with the details to automate the process, like entering the path to the Windows-10/server-2019 ISO image file, sizing requirements and so on.



Once the configuration is done, we'll run the packer to build the machines using below command by passing the json config for the machine.

```
packer.exe build .\your-server-2019-file.json
```



For more details on automation, there are great writeups by the fellow members of the Auror Project who have explained different ways in their writeups as to how we can automate the whole process. Below are the references:

- https://sbasu7241.medium.com/auror-project-challenge-1-automated-active-directory-lab-deployment-53e323445f4d
- https://medium.com/@deepakkeshav98/3-machine-labs-1-automating-lab-a9870fc42c54
- https://www.passthehacks.com/post/the-auror-project
- https://pswalia2u.medium.com/automate-active-directory-installation-packer-provisioning-vagrant-e5b059d8fda

Real-time detection scenarios 🕵️

In the 2nd session, we discussed a lot of important concepts related to Active Directory Users and Security Groups. Based on that, we were asked to go through few challenges such as below by programming or through scripts or creating Analytics Dashboard:

- Detect when a computer account is added to any of the created domain security groups
- Detect an attempt to spray passwords using user attributes
- Detect a change to the domain admins group membership and notify this activity
- Detect when a helpdesk administrator is also a server administrator
- Gather the count of administrators on the crown jewel machine and domain controller (including local accounts). Detect when this number changes

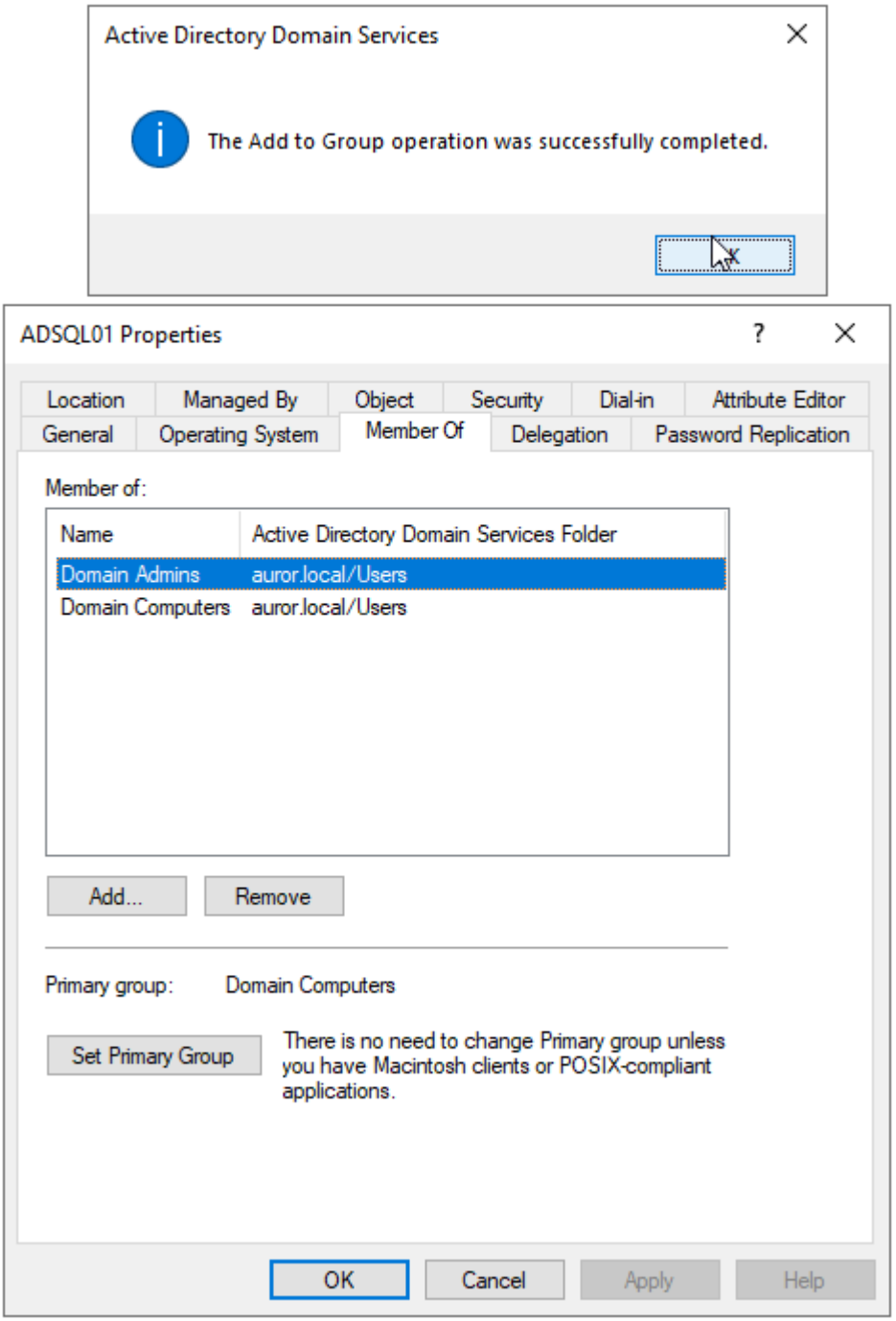
In this post, I will go through C# programs that I created to look for these near real-time detection scenarios of above activities in an Active Directory domain environment.

> **Scenario 1 - Real-time detection when a computer account is added to any domain security groups**

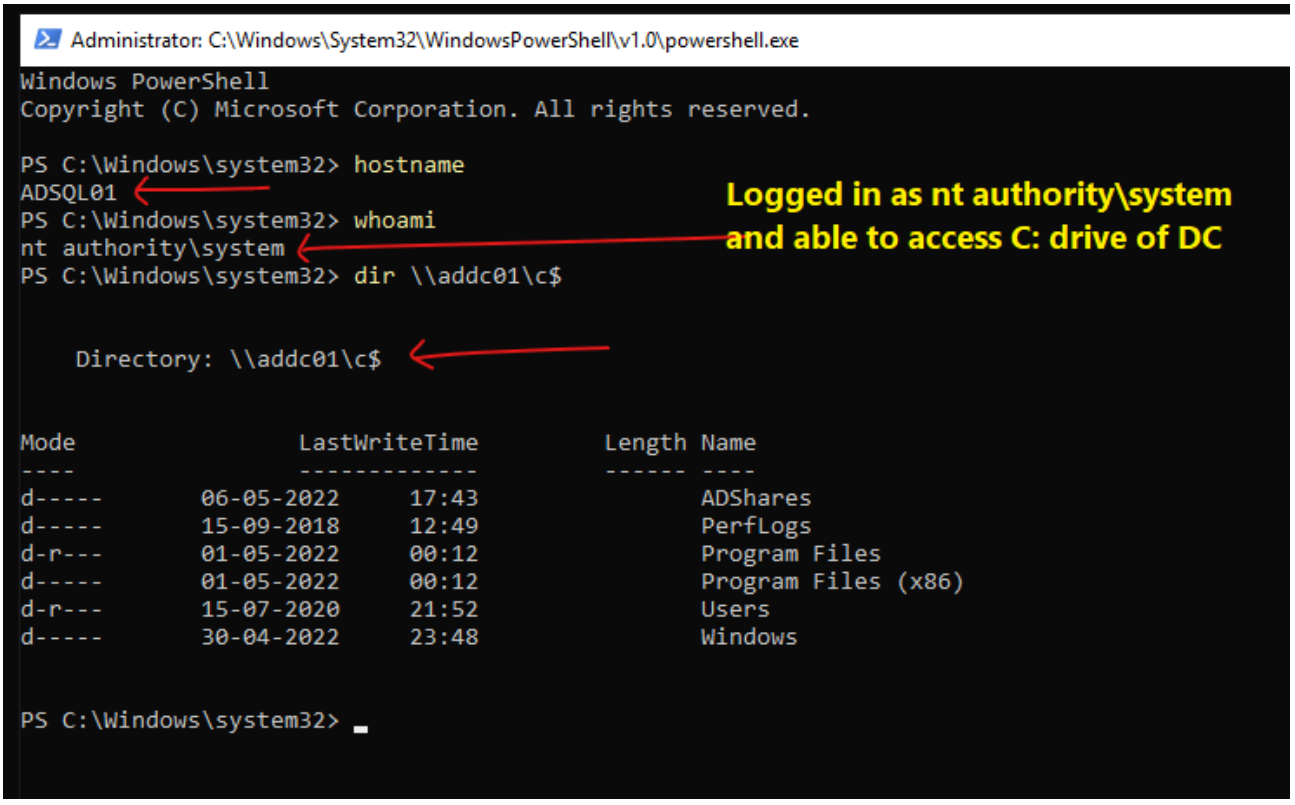
Detect Evil Machine 😈

I came across this concept of what could happen when we add a domain machine into a privileged group. Normally we add the user accounts into security groups which allow them to perform different tasks based on the permissions of security groups. Usually the machine accounts are not added into the security groups.

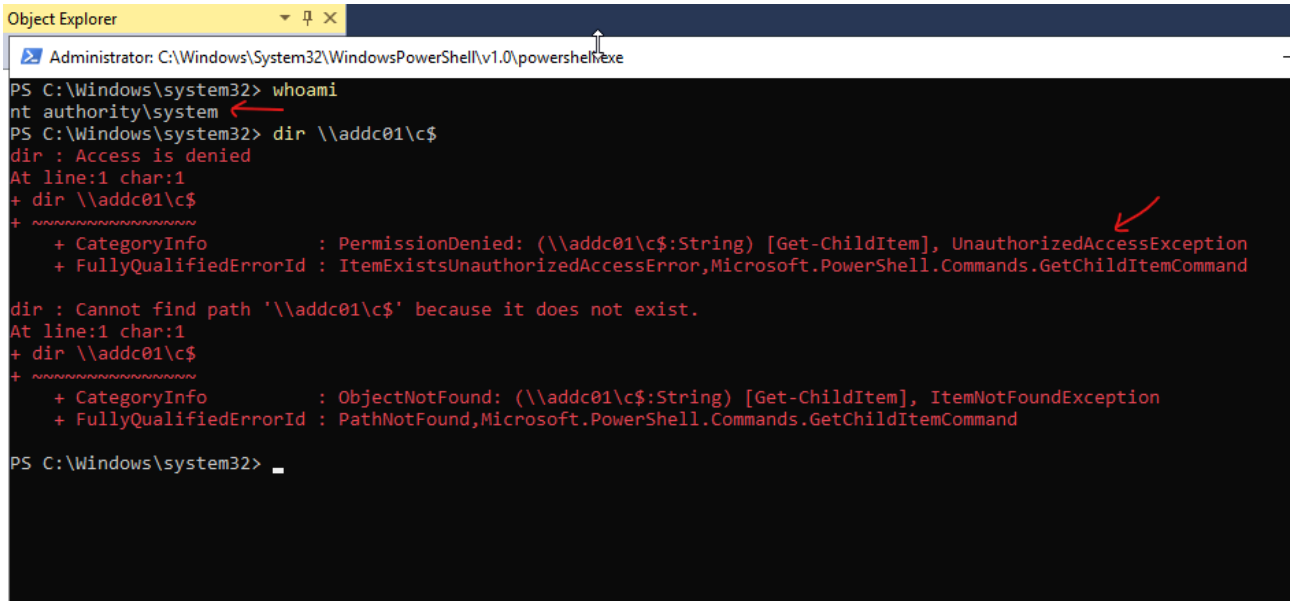
Let’s see what happens if we add a machine account into the security groups (for example- into Domain Admins)



When I added a machine account into the security group (Domain Admins), the local system account (NT AUTHORITY\SYSTEM) was able to access the domain resources. I'll try to access system drive of the domain controller, can perform lateral movement as well using the same account. Quite interesting!



Normally, a machine account is not added into domain admins or other domain security groups, so that's why when we run as local system account (NT AUTHORITY\SYSTEM) and we are not allowed to access the system drive of the domain controller, which is quite normal.

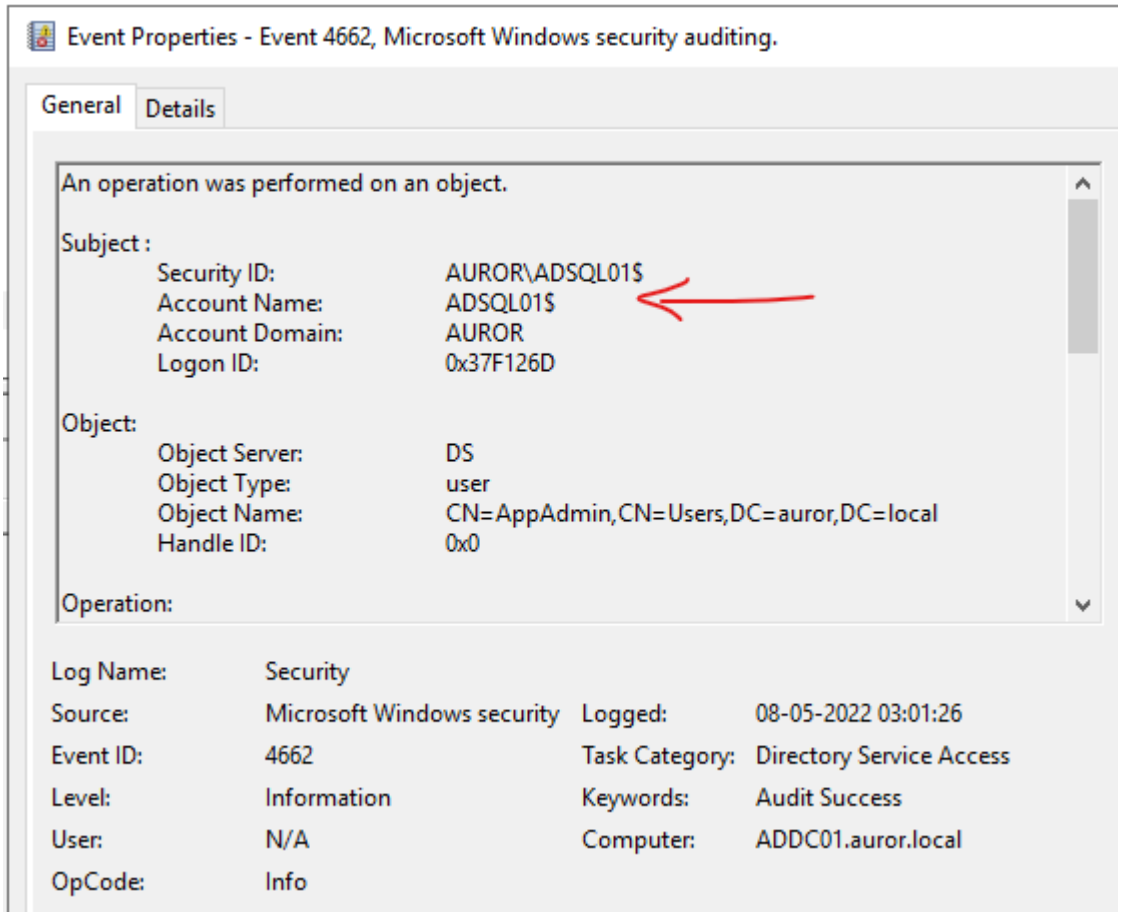


Local system account (NT AUTHORITY\SYSTEM) only has the highest privileges on the windows local machine (for example- it has full control to all files on an NTFS volume). It is used by the operating system and by the services that run by Windows in the background for different tasks by logging in internally using this account.

In normal scenarios, this account doesn't even show up in Local Users and Groups Management console, and cannot be added to any groups from console. Sometimes, when we get a remote shell as NT AUTHORITY\SYSTEM, we are able to perform a lot of activities on the local machine, but we cannot use that account to move laterally.

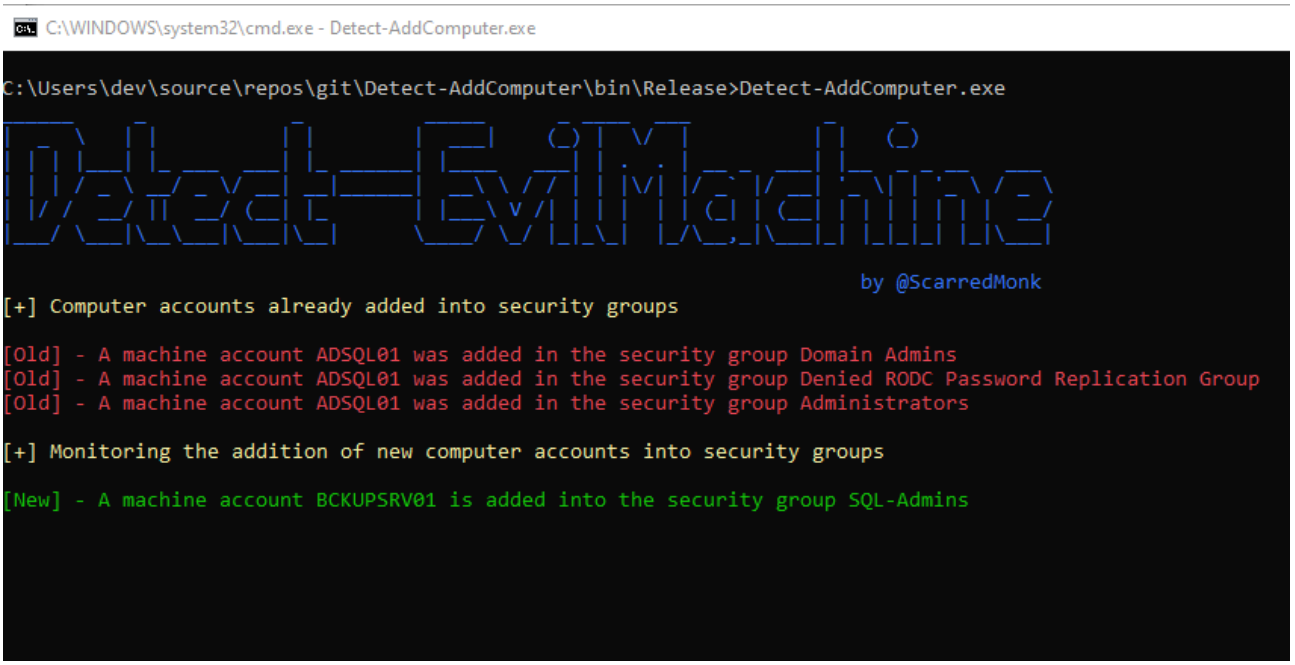
But if the machine (that we got access to), is added into a security group, the machine account gets the permissions for the relevant critical server or resources based on that security group. Sometimes, system administrators do this to run some domain level tool which is having permission issues or for troubleshooting such issues, and then forget to remove the machine accounts from the groups. While doing Red team engagements, offensive security team members can look for such machine accounts that have highest privileges. And the threat detection teams can check for any machine accounts present in any of the security groups, as a part of the attack surface management.

Think of scenarios like where adversaries have such type of access to a machine, then the source user in the logs would be the machine account ending with $ (such as ADSQL01$) instead of the user accounts. This can bypass many threat detection SIEM conditions in which the machine accounts (ending with $) are filtered to reduce noise.



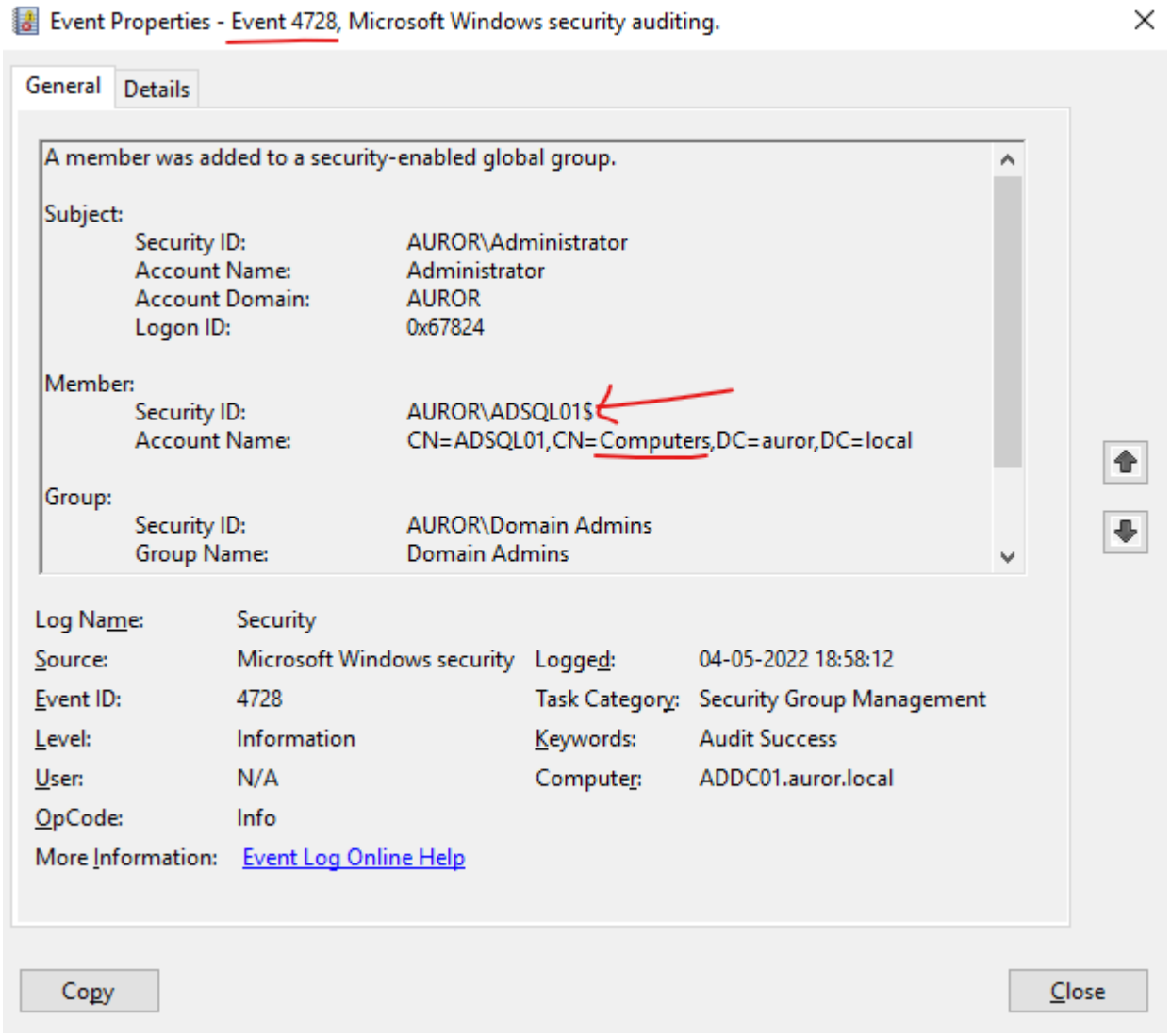
I have created a POC tool in C#, which performs below functions:

1. Displays already present machine accounts in any of the security groups in the Active Directory Domain
2. Monitors for any new machine accounts being added into any of the security groups
3. Notifies on the console about the new machine accounts being added into any of the security groups



As seen in the screenshot, the tool shows the machine accounts that are already added into the security groups. And it also looks for new additions in the real-time and notifies on the console.

Threat detection teams can also create a detection logic to detect such activity by looking into Windows Event Logs with event ID 4728 and check if a computer account is added into a security group.



Tool Link - https://github.com/ScarredMonk/Detect-Evil-Machine

---

> **Scenario 2 - Real-time detection of attempts to spray passwords using user attributes**

Detect Password Spraying 🥷

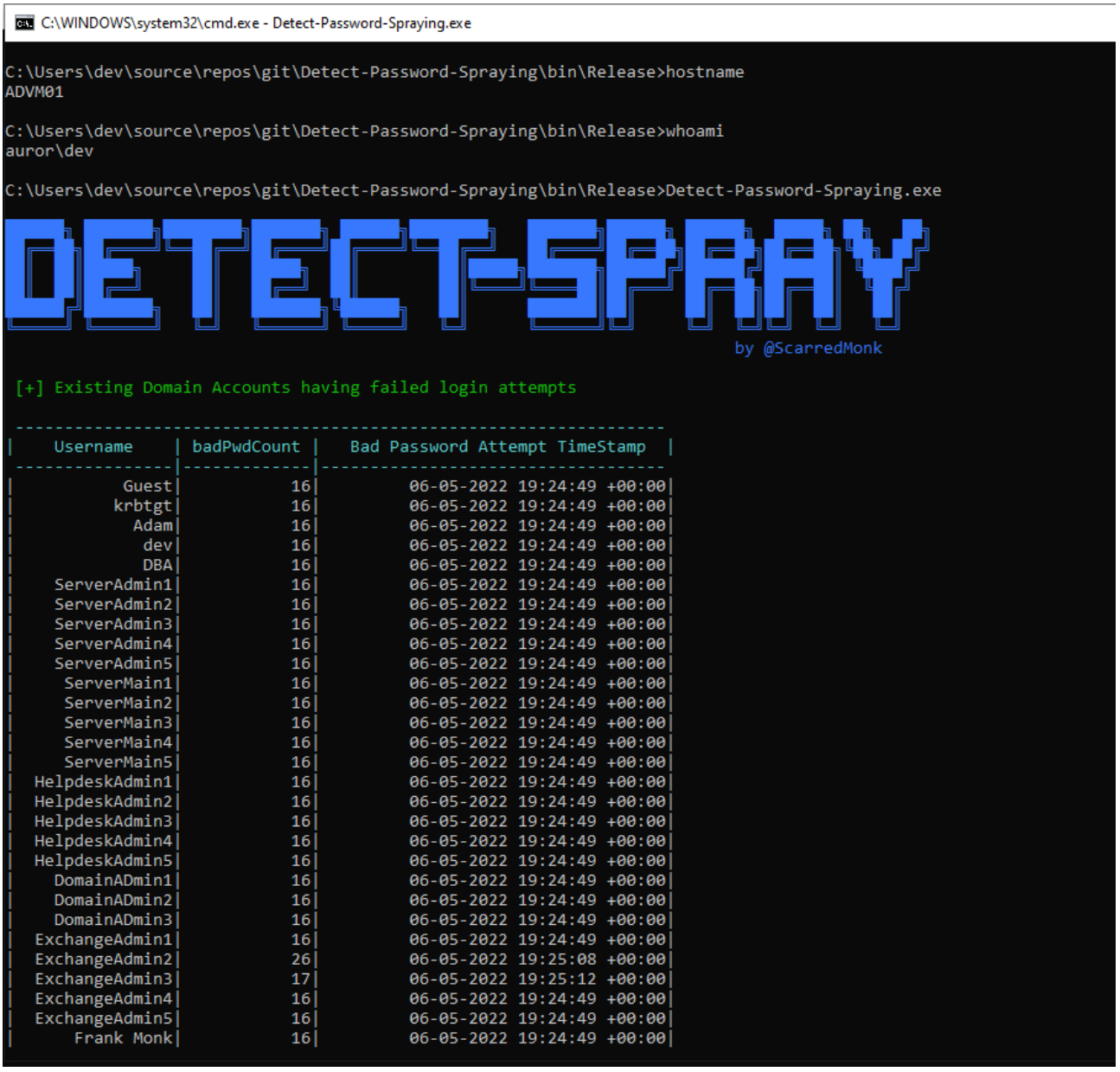
In this scenario, I'll perform a password spraying attack and try to detect it in real time. Password spraying attack is a type of brute force attack in which adversaries try to brute force logins based on a password (for example - 'pass@123'), or a small list of commonly used passwords are attempted against different accounts on a network to avoid account lockouts that would normally occur when brute forcing a single account with many passwords.

For this POC, I have written a tool, that check for multiple bad password attempts from the user attributes.

First of all, I am fetching all the accounts using LDAP, that have value of `badPwdCount` > 0.

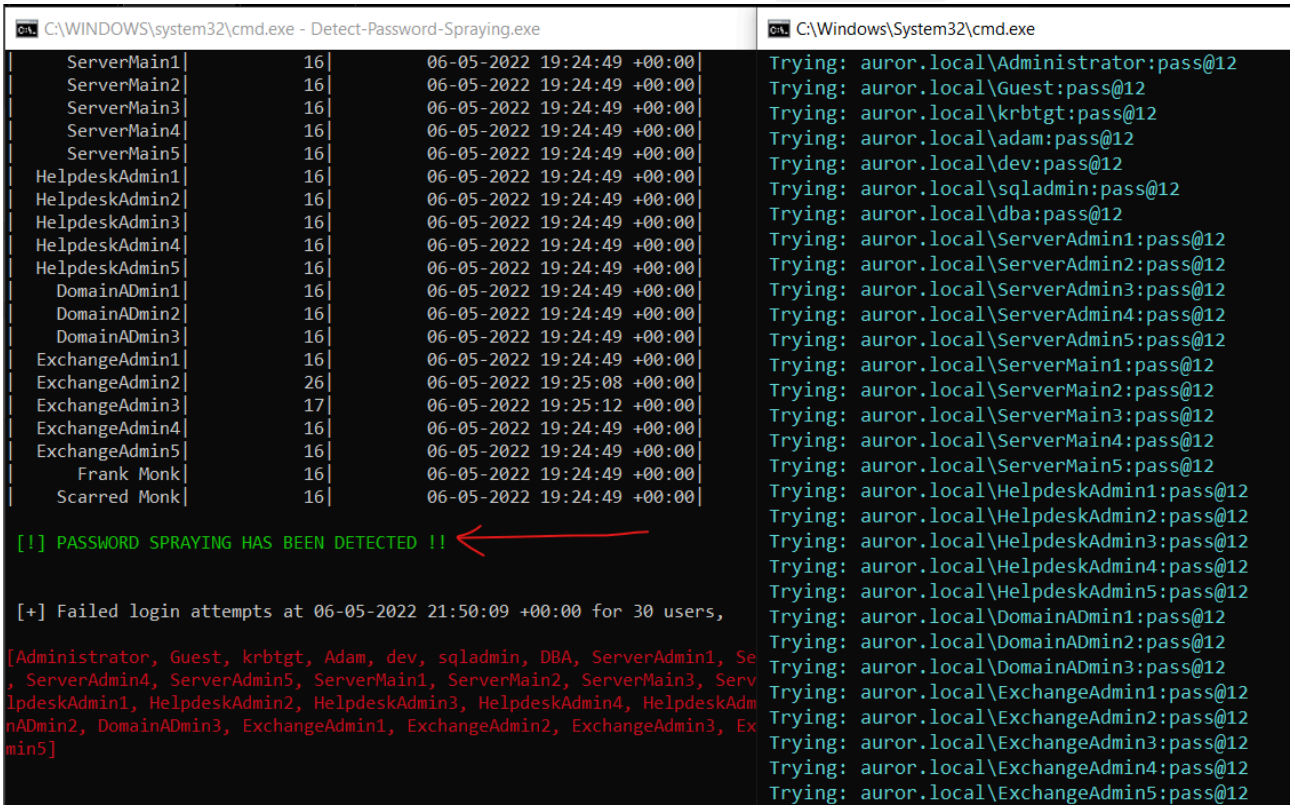
```
    DirectoryEntry adObject = new DirectoryEntry();
    DirectorySearcher searcher = new DirectorySearcher(adObject)
```

```
    {
        SearchScope = SearchScope.Subtree,
        Filter = "(&(objectclass=user)(!(objectclass=computer))(!(badPwdCount=0)))"
    };
    var queryattributes = searcher.FindAll();
    CheckPassSpray(true);
```



Then, I'm leveraging the Active Directory user attributes `badPasswordTime` and `badPwdCount`. I checked for the `badPwdCount` for multiple users and grouping them on basis of `badPasswordTime` attribute. When I'll try

password spraying for multiple users, it will get detected by the tool `Detect-spray` as shown in the screenshot:

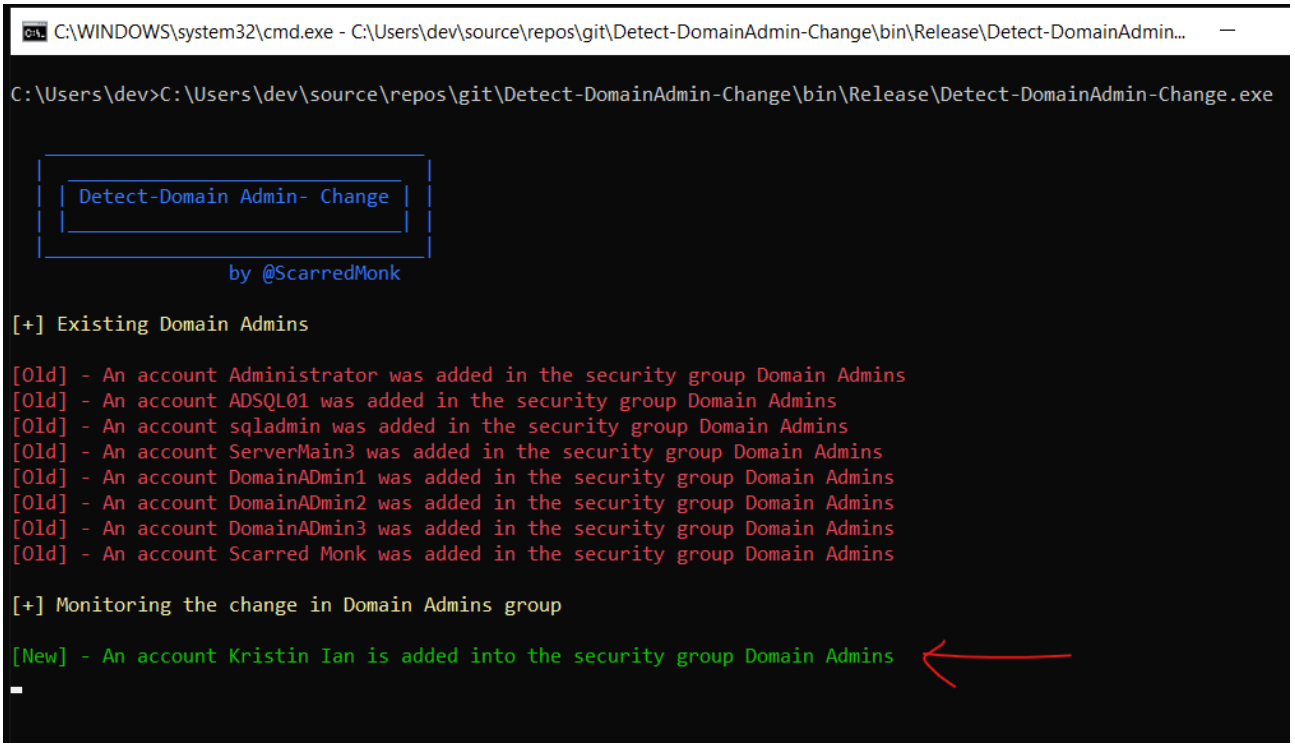


Tool Link - https://github.com/ScarredMonk/Detect-Spray

> **Scenario 3 - Real-time detection of change in Domain Admins group membership**

Detect Change in Domain Admins 👀

The attack surface is highly dependent on how many administrators are there in a particular domain. It should be limited and contiously monitored for the same reason. In this scenario, I am looking for any change in the Domain Admins group. This tool prints the existing members of domain administrators and notify on console if there is a new member added to the group.



Threat detection teams can also create a detection logic to detect such activity by looking into Windows Event Logs with event ID 4728 and check for any attempts of addding users into a privileged group.

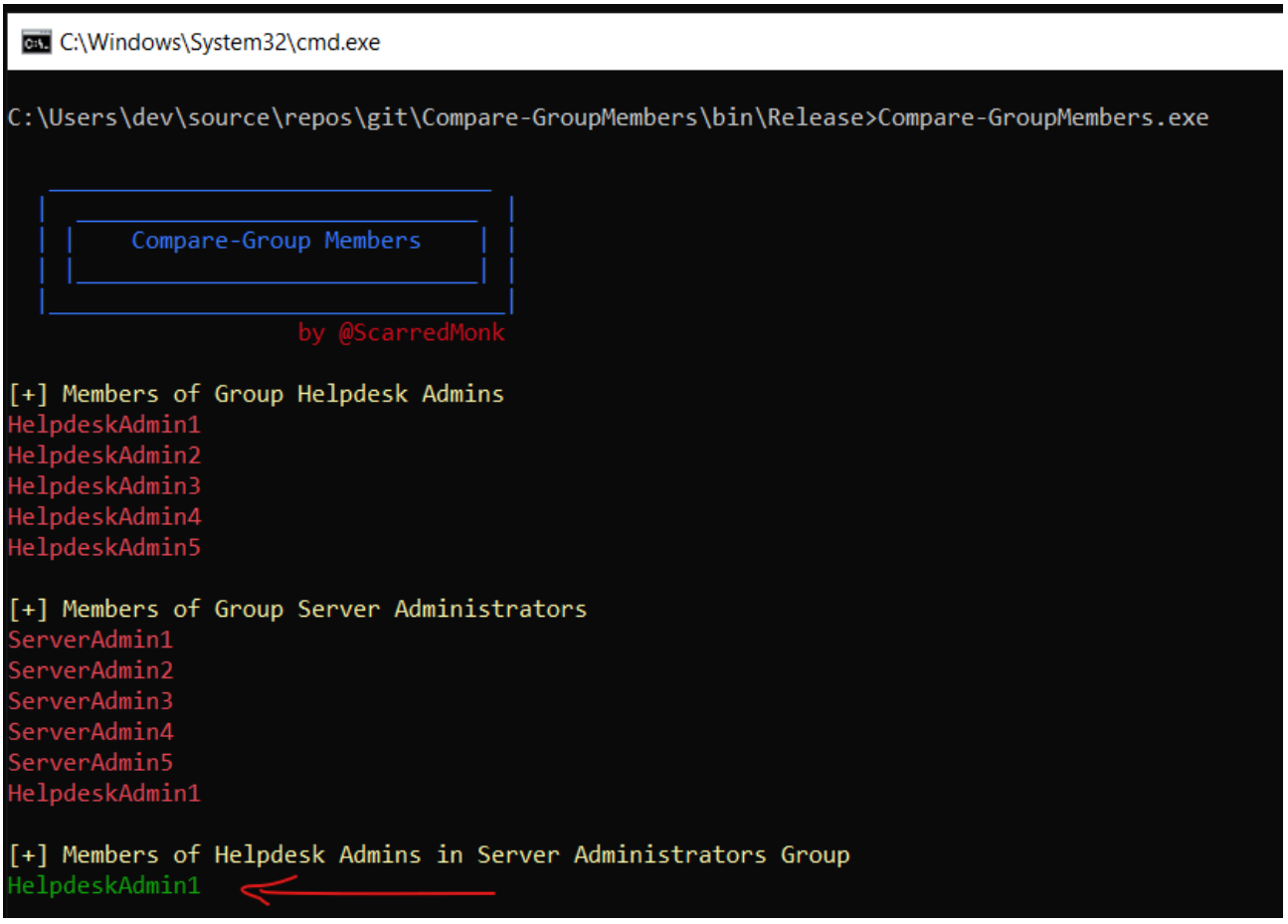
Tool Link - https://github.com/ScarredMonk/Detect-DomainAdmin-Change

---

> **Scenario 4 - Check if a member of Group 1 is present in Group 2 in Active Directory Domain**

Check member of one Group in other group 👥

Sometimes, it is possible that a low privilege user is a member of a high privileged group. In this challenge, I compared the members of one group in other group to see if it is present in other high privilege group. For the challenge, we had to check if any user of helpdesk admins is a part of Server Administrators group.



Tool Link - https://github.com/ScarredMonk/Compare-Group-Members

---

> **Scenario 5 - Gather the count of administrators on the crown jewel machine and domain controller (including local accounts). Detect when this number changes**

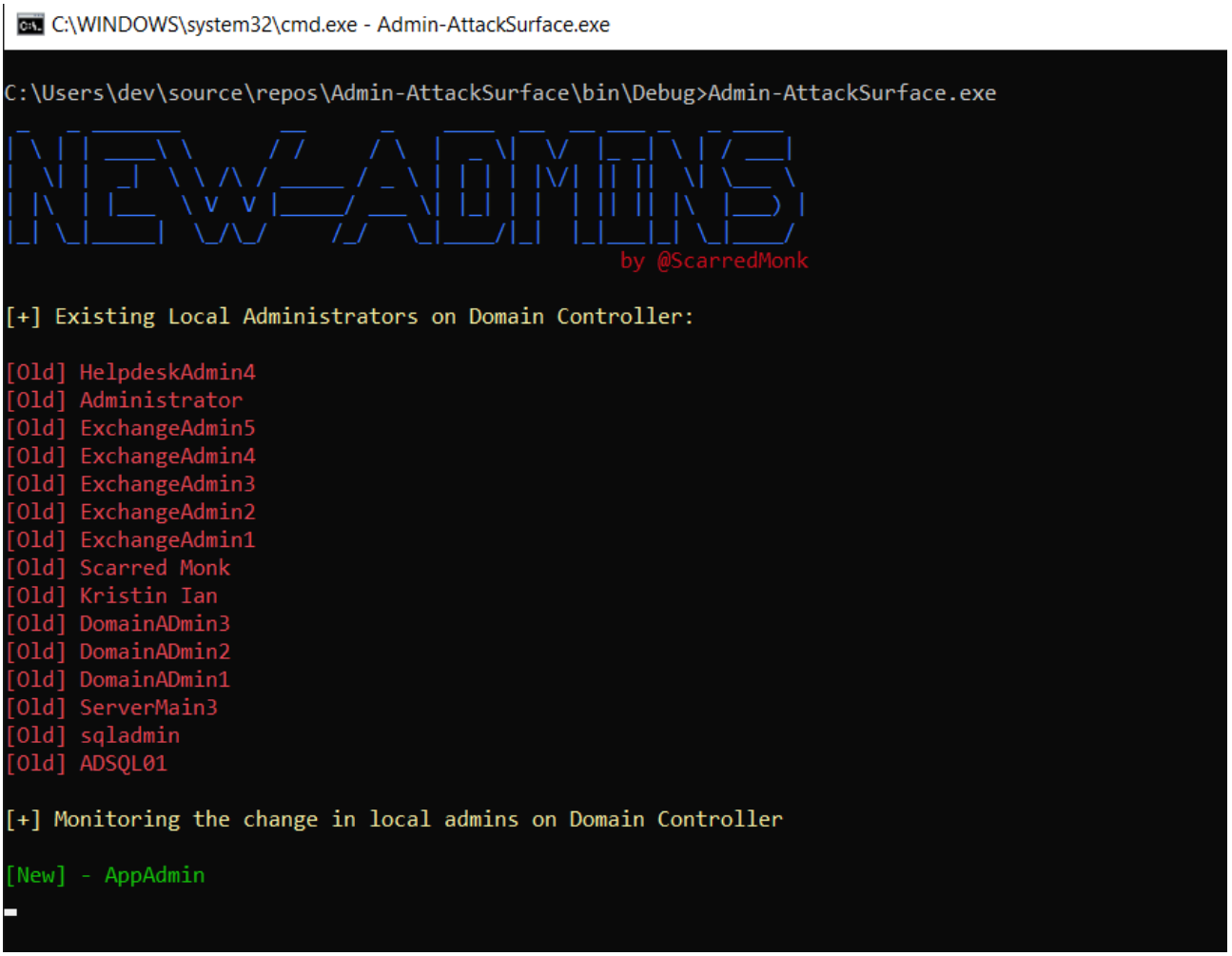
Local admin attack surface managememt 🕸

This is done for attack surface management. It is very important to keep a close eye on Privileged Group Membership. In this challenge, I detected the change in local administrators group of domain controller and crown jewel machine (critical machine). Below is a very good article by Microsoft on reducing the Active Directory Attack Surface:

https://docs.microsoft.com/en-us/windows-server/identity/ad-ds/plan/security-best-practices/reducing-the-active-directory-attack-surface

Local administrators group of domain controller

Whenever a new domain admin is added, it is notified on the console of this tool.



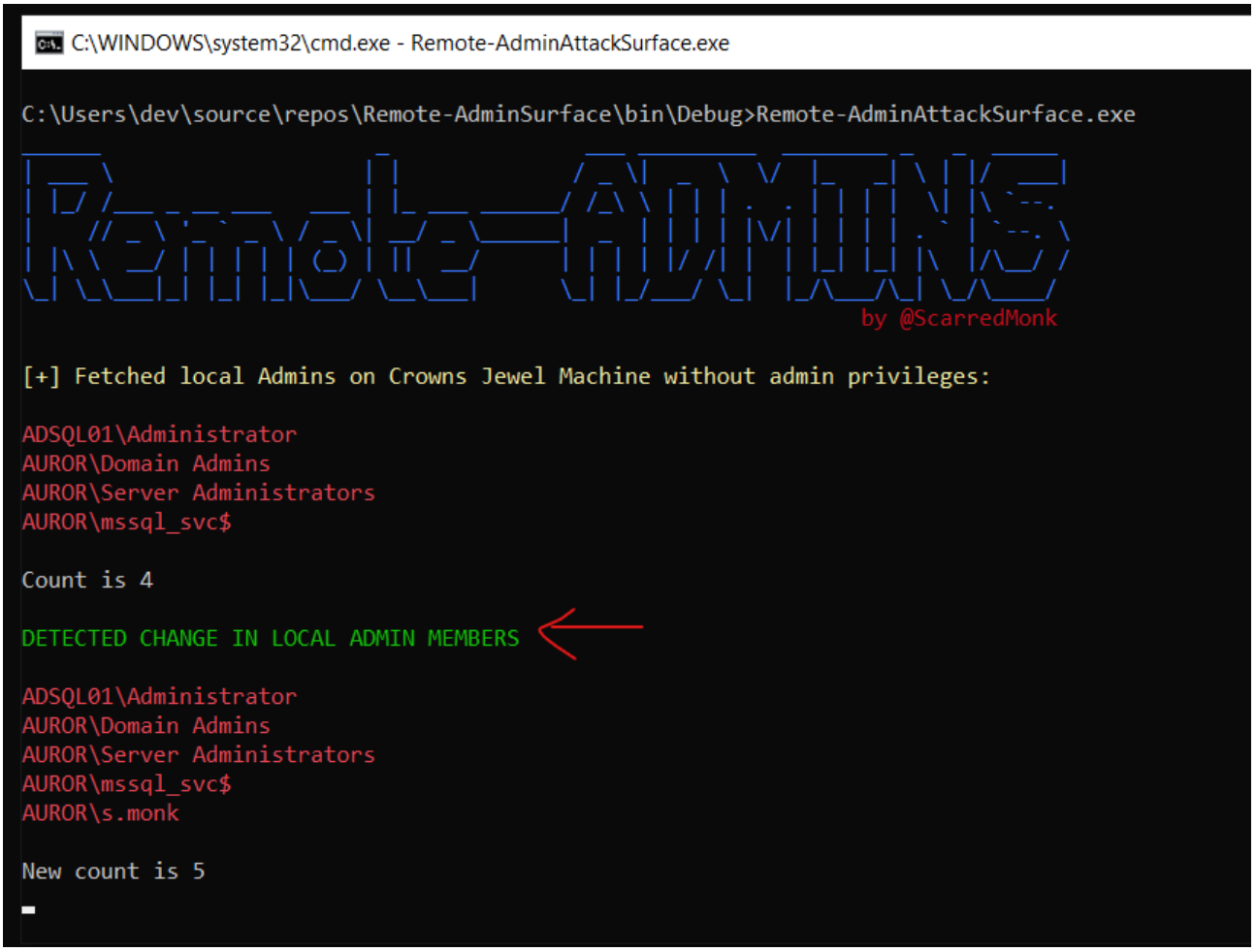
The same thing can be monitored from Windows Event Logs with event ID 4728 and check for any attempts of addding users into a Administrators groups of a domain controller.

Tool Link - https://github.com/ScarredMonk/ASM-NewDCAdmins

Local administrators group of Crown Jewel machine

The challenge required not to use domain admin / local admin privileges to retreive local admin group members on a domain machine, but a tweak on other permissions. So I tried different permissions and ways to achieve this without becoming a domain administrator or a local administrator on the remote machine. Finally, I was able to find a way to enumerate remotely after providing my scanning user permissions to enumerate through WMI on remote machine.

- Open WMImgmt.msc
- Go to the Properties of WMI Control
- Go to the Security Tab
- Select “Root” and open security
- Select “Remote Enable” permission for my monitoring user account



Tool Link - https://github.com/ScarredMonk/ASM-NewRemoteAdmins

The same thing can be monitored from Windows Event Logs with event ID 4728 and check for any attempts of addding users into a Administrators groups of a local machines.

It is very important to monitor other privileged groups as well, apart from Domain Admins such as : Administrators, Print Operators, DHCP Admins, Backup Operators, Account Operators, Cert Publishers, Network Configuration Operators, Group Policy Creator Owners, Domain Controllers, Enterprise Admins, Server Operators, RAS and IAS Servers, Schema Admins etc.

Summary

To quickly recap, let’s summarize the above scenarios and monitoring recommendations for the Active Directory environments.

- Monitor any changes to privileged groups in the whole environment in order to reduce the attack surface.
- Detect when computer account is added to any of the created domain security groups because of its consequences.
- Quickly detect an attempt to spray passwords by leveraging the user attributes, even when the attack bypasses the event logs.
- Get notified of the activities when domain admins group membership is modified.

- Check why a member of low privileged group suddenly became member of a high privileged group.
- Have handly dashboards showing changes in the count of administrators on the crown jewel machines/domain controllers (including local accounts).

For most of the organisations, the Active Directory environments are created initially and since then, there are lot of group additions and changes happened over a period of time. So there are many possibilities that administrators might have lost track of many of the security groups in the domain as to why are those groups there if they are not in use anymore. So this is very important that organisations do attack surface management from time to time. We cannot block these activities in the environments such as Active Directory enumeration, because it is continously happening for legitimate purposes by business tools and even by Windows applications. But what we can do is to monitor these areas in real-time and work on managing and reducing the attack surface in Active Directory.

This was just for one session that was related to users and groups. There are many interesting things in the Auror Project which not only just involves coding but also analytical thinking and will help in building problem solving mindset, so I would recommend the readers to join the project for something new.
