A Hitchhackers Guide to DACL-Based Detections (Part 1)

1 Introduction

If you were to collectively ask any Windows penetration tester or “red teamer” to recount their most common “attack paths,” there is no doubt that many, if not all of them, will include Active Directory (AD) based attacks. It’s easy to understand both why AD has been commonly dubbed the “attacker’s playground” and why a defender could become overwhelmed by the vast AD attack surface.

The goal of this post is to provide the “blue team” with a greater level of understanding on how these attacks “may” operate, but also help identify where an adversary may be hiding. As such, this post will strive to collectively identify those AD attributes that an attacker or adversary may modify within a target environment to lead into further access.

It is important to note that this blog is assuming that the adversary already has a foothold within the domain and has acquired the appropriate access they need to make modifications to the objects we will discuss. This post also does not examine any post exploitation (i.e., forged Kerberos tickets, etc.). We are only addressing the modifications given that the primary purpose of this exercise is to build detections to identify when changes are made. Furthermore, a level of “intelligence” (i.e., providing an attribution of attack to adversary) has not been incorporated. While “attribution matters,” for time purposes, intelligence has not been mapped to each attack.

Lastly, this post will, in a series of three (3) parts, provide classic Splunk SPL queries for detecting the attacks outlined, using only Windows Event IDs as described. Furthermore, this blog post only examines a subset of the Windows Event logging data source, and not all possible telemetry within this data set have been analyzed.

2 Using a Visual Roadmap - Object/Attribute Overview

The following chart, from The Hacker Recipes, provides a visual roadmap and serves as a basis to the AD Objects and Attributes that we will be working with throughout this three (3) part series. We will step through this roadmap in order to try to provide as much detection coverage as possible.

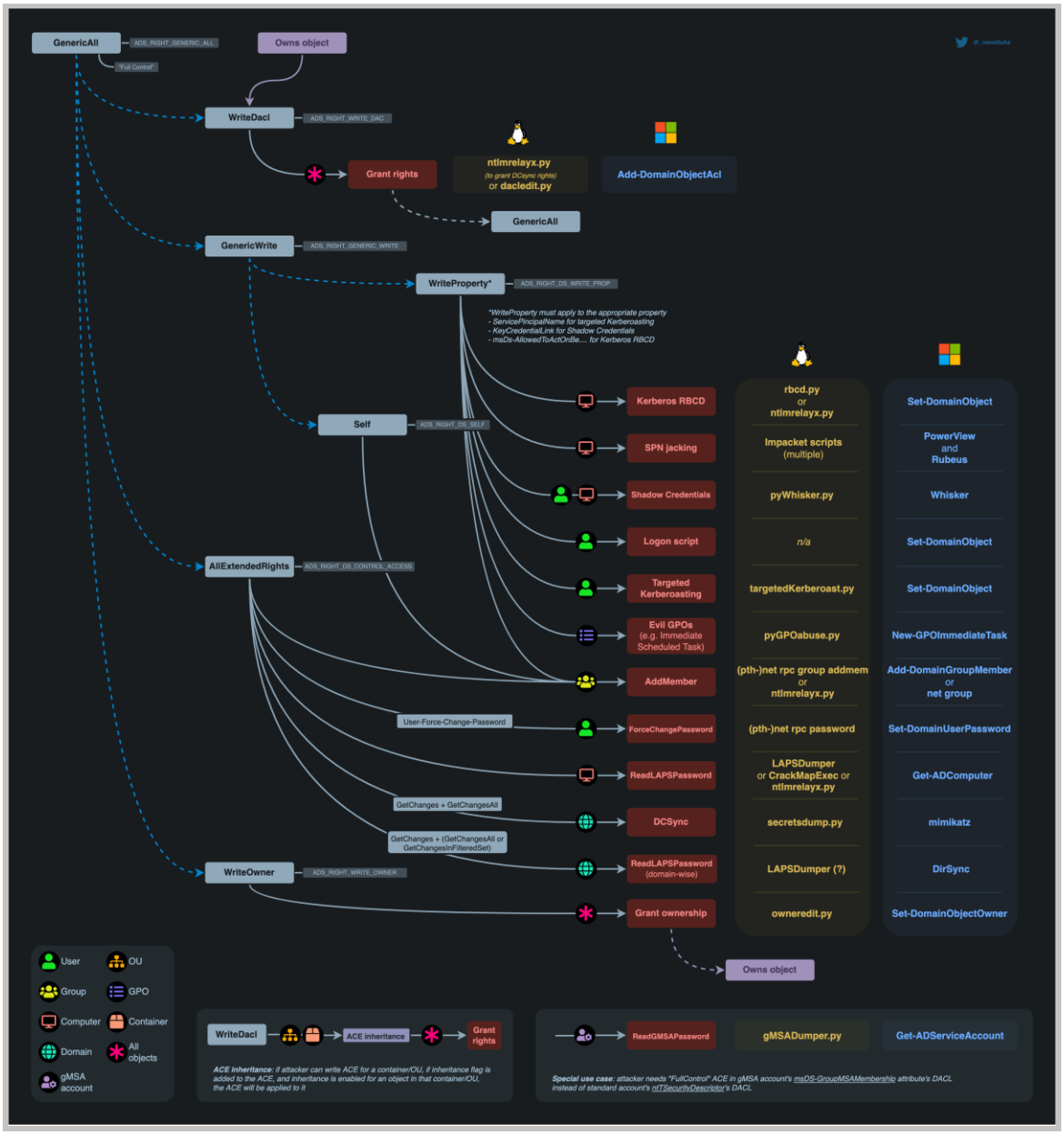


Figure 1 - Object Review Flow Chart From the Hacker Recipes

3 Logging Setup

3.1 Windows Events

It should be noted that Event ID 5136 is not enabled by default and can be configured by enabling:
Advanced Audit Policy Configuration > Audit Polices > DS Access > Audit Directory Service Changes.

However, there are some limitations with Event ID 5136, namely that it does not provide much contextual data for us to quickly identify what we would need to respond to a potential attack.

Enter correlation...and Windows Event IDs 4662 and 4624. Both events are part of the Advanced Auditing policies and may not be enabled by default. If we combine the data from all three (3) Event IDs, we can essentially build a template to build detections for the various modifications/changes to provide greater contextual representation.

Event 4662 is configured via by enabling:
Advanced Audit Policy Configuration > Audit Polices > DS Access > Audit Directory Service Access.

Event 4624 is frequently enabled by default but can be configured by enabling:
Advanced Audit Policy Configuration > Audit Polices > Logon/Logoff > Audit Logon

If we combine the data from all three (3) Event IDs, we can essentially build a query that provides a greater contextual representation of the attack.

In addition, for some detections, we may use other Events such as:

Event 5145, which can be configured by enabling:
Advanced Audit Policy Configuration > Audit Polices > Detailed Tracking > Audit Detailed File Share

Event 4742, which can be configured by enabling:
Advanced Audit Policy Configuration > Audit Polices > Audit Computer Account Management

Event 4738, which can be configured by enabling:
Advanced Audit Policy Configuration > Audit Polices > Audit User Account Management

3.2 SACL

Configuring a SACL is an **additional step** that must be taken even if the above listed Windows Events are currently being ingested.

For the purpose of this blog post, we have created a SACL entry on the root of our Domain to audit all objects; however, this can be done more granularly if logging volume is a concern.

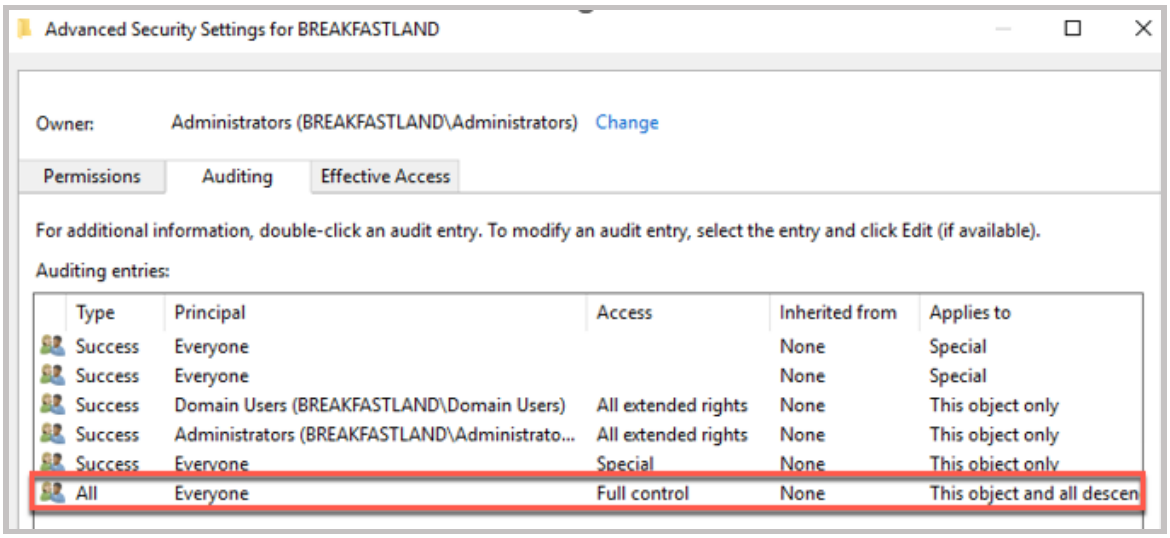


Figure 2 - SACL Configuration for BREAKFASTLAND.LOCAL

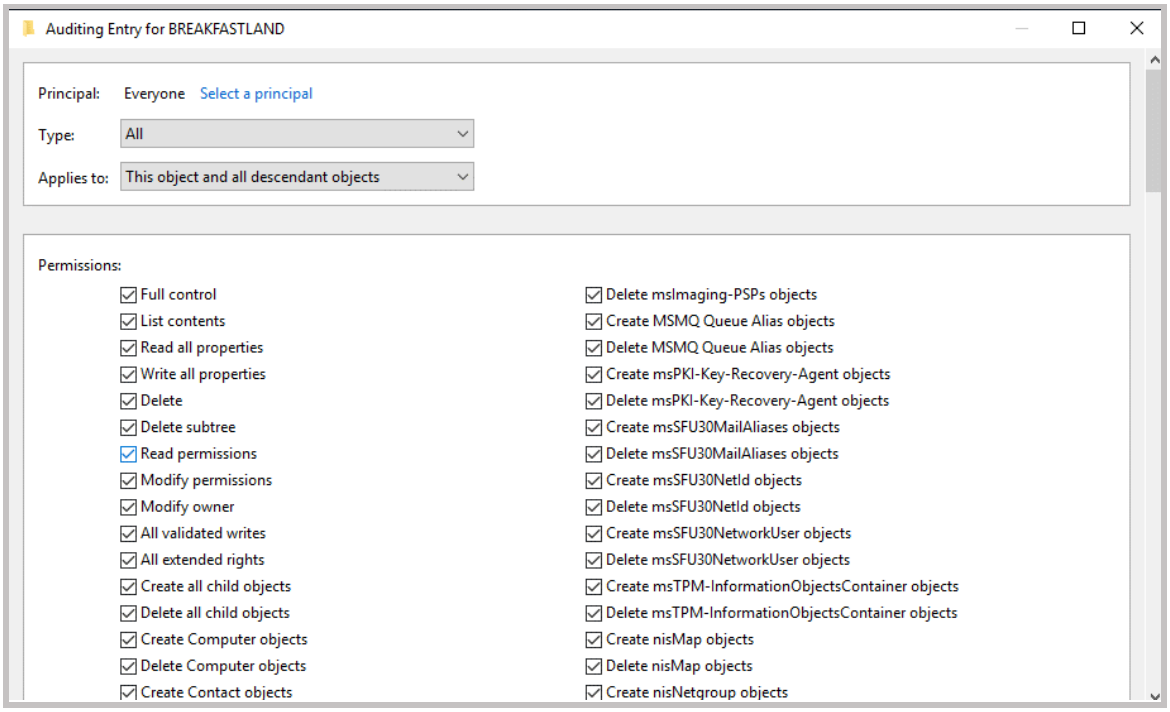


Figure 3 - SACL Configuration

In addition, you may need to enable auditing for specific User or Computer objects. We will attempt to call these items out specifically as we run through each detection; however, if you find you are not receiving the logging for the object that is being modified, be sure to check your SACL for the object as that is likely to be the issue.

4 Blog Format

Due to the length of this blog series and the number of attributes covered, it is important to do a quick overview of the format and what to expect.

Each section will contain the following headings:

- Name of the Attribute (CN of the attribute)
- Background
    - Will cover a brief overview of what the attribute (***LDAP-Display-Name***) is and the relevant links to Microsoft documentation
- Modifying the Attribute (Attack)
    - Will cover how the "attack" was performed, including relevant setup for modifying the attribute in question, screenshots/commands, and tools used
    - If additional auditing was enabled for building the detection, it will also likely be covered here—or, if additional setup was more complex, will be broken out into a preceding or subsequent heading
- Building the Detections
    - Will cover a variety of detections that will include a range of complexity
    - As was stated in the introduction, not all the possible telemetry data points within this data set have been analyzed. However, we have tried our best to cover the Event IDs that are most accessible and prominent for building out detections
    - Where necessary, we will provide a flow of logic for detections that involve more complexity or additional information to interpret what is being shown. However, most detections will follow a similar format and will not be explained in further detail

5 Object Modifications & Detections

5.1 Writing to msDS-Allowed-to-Act-On-Behalf-Of-Other-Identity

Beginning at the top of the Hacker Recipes flow chart, the first attribute modification on our list is regarding Resource Based Constrained Delegation (RBCD), whereby the attack may be writing to the attribute msDS-AllowedtoActOnBehalfOfOtherIdentity. This attribute was previously examined by Andrew, Jonathan Johnson, and Charlie Clark in this post.

As this has already been covered in detail, we will not be addressing this object within this post.

5.2 Writing to Service-Principal-Name (SPN)

5.2.1 Background

The Service Principal Name (SPN) of an object is a unique identifier that can be used by Kerberos to associate a "service instance" with an authentication attempt. SPNs are frequently abused by attackers using Impacket Modules such as GetUsersSPN.py or other hacker toolsets that exist to exploit existing SPNs or to create new ones that can be leveraged to bypass other authentication mechanisms.

5.2.2 Creating a Machine Account Using PowerMad

Before we can modify our SPN attribute, we are going to create a new machine account to use as our "victim" computer. This "victim" computer account will be used for many of the attribute modifications we will make with PowerMad and other tools moving forward through this blog series.

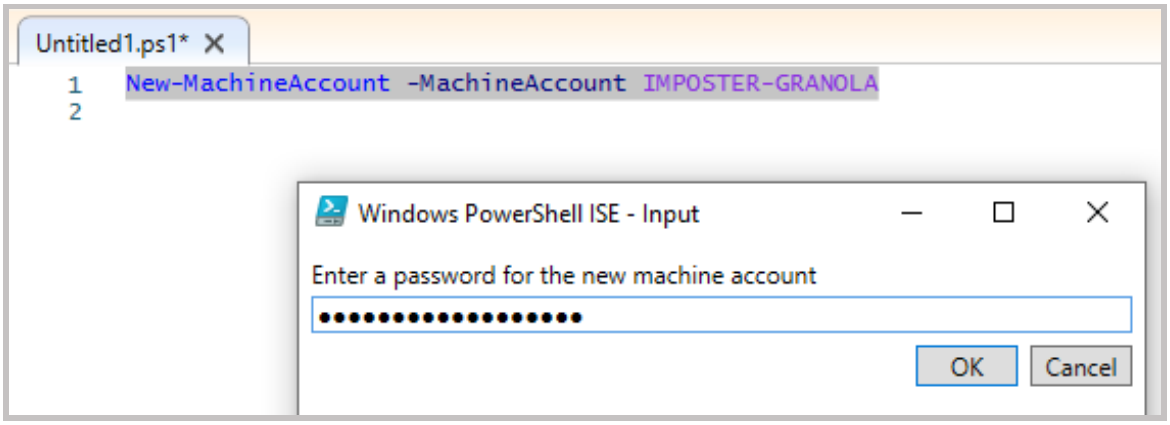


Figure 4 - Creating a New Computer Account

```
PS C:\Powermad-master> New-MachineAccount -MachineAccount IMPOSTER-GRANOLA
[+] Machine account IMPOSTER-GRANOLA added
```

Figure 5 - Computer Account Creation with PowerMad

Before changing any attributes, this is what the **IMPOSTER-GRANOLA$** machine account we created looks like as a freshly created object.

```
PS C:\Users\head.chef> Get-ADComputer -Identity "IMPOSTER-GRANOLA" -
Properties *


AccountExpirationDate              :
accountExpires                          : 9223372036854775807
AccountLockoutTime                 :
AccountNotDelegated                : False
AllowReversiblePasswordEncryption  : False
```

```
AuthenticationPolicy                : {}
AuthenticationPolicySilo            : {}
BadLogonCount                       : 0
badPasswordTime                     : 0
badPwdCount                         : 0
CannotChangePassword                : False
CanonicalName                       :
BREAKFASTLAND.LOCAL/Computers/IMPOSTER-GRANOLA
Certificates                        : {}
CN                                  : IMPOSTER-GRANOLA
codePage                            : 0
CompoundIdentitySupported           : {}
countryCode                         : 0
Created                             : 5/30/2023 11:59:24 AM
createTimeStamp                     : 5/30/2023 11:59:24 AM
Deleted                             :
Description                         :
DisplayName                         :
DistinguishedName                   : CN=IMPOSTER-
GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL
DNSHostName                         : IMPOSTER-
GRANOLA.breakfastland.local
DoesNotRequirePreAuth               : False
dSCorePropagationData               : {12/31/1600 4:00:00 PM}
Enabled                             : True
HomedirRequired                     : False
HomePage                            :
instanceType                        : 4
IPv4Address                         :
IPv6Address                         :
isCriticalSystemObject              : False
isDeleted                           :
KerberosEncryptionType              : {}
LastBadPasswordAttempt              :
LastKnownParent                     :
lastLogoff                          : 0
lastLogon                           : 0
LastLogonDate                       :
localPolicyFlags                    : 0
Location                            :
LockedOut                           : False
logonCount                          : 0
ManagedBy                           :
MemberOf                            : {}
MNSLogonAccount                     : False
Modified                            : 5/30/2023 11:59:24 AM
modifyTimeStamp                     : 5/30/2023 11:59:24 AM
msDS-User-Account-Control-Computed  : 0
Name                                : IMPOSTER-GRANOLA
nTSecurityDescriptor                :
System.DirectoryServices.ActiveDirectorySecurity
ObjectCategory                      :
CN=Computer,CN=Schema,CN=Configuration,DC=BREAKFASTLAND,DC=LOCAL
ObjectClass                         : computer
ObjectGUID                          : 863169ce-25a7-468d-a147-
3e193587df4f
objectSid                           : S-1-5-21-1865600711-3446354287-
```

```
3882071624-1113
OperatingSystem                     :
OperatingSystemHotfix               :
OperatingSystemServicePack          :
OperatingSystemVersion              :
PasswordExpired                     : False
PasswordLastSet                     : 5/30/2023 11:59:24 AM
PasswordNeverExpires                : False
PasswordNotRequired                 : False
PrimaryGroup                        : CN=Domain
Computers,CN=Users,DC=BREAKFASTLAND,DC=LOCAL
primaryGroupID                      : 515
PrincipalsAllowedToDelegateToAccount : {}
ProtectedFromAccidentalDeletion     : False
pwdLastSet                          : 133299467648286422
SamAccountName                      : IMPOSTER-GRANOLA$
sAMAccountType                      : 805306369
sDRightsEffective                   : 15
ServiceAccount                      : {}
servicePrincipalName                : {RestrictedKrbHost/IMPOSTER-
GRANOLA, HOST/IMPOSTER-GRANOLA,
                                      RestrictedKrbHost/IMPOSTER-
GRANOLA.breakfastland.local,
                                      HOST/IMPOSTER-
GRANOLA.breakfastland.local}
ServicePrincipalNames               : {RestrictedKrbHost/IMPOSTER-
GRANOLA, HOST/IMPOSTER-GRANOLA,
                                      RestrictedKrbHost/IMPOSTER-
GRANOLA.breakfastland.local,
                                      HOST/IMPOSTER-
GRANOLA.breakfastland.local}
SID                                 : S-1-5-21-1865600711-3446354287-
3882071624-1113
SIDHistory                          : {}
TrustedForDelegation                : False
TrustedToAuthForDelegation          : False
UseDESKeyOnly                       : False
userAccountControl                  : 4096
userCertificate                     : {}
UserPrincipalName                   :
uSNChanged                          : 376938
uSNCreated                          : 376936
whenChanged                         : 5/30/2023 11:59:24 AM
whenCreated                         : 5/30/2023 11:59:24 AM
```

We will also need to build a SACL for the **IMPOSTER-GRANOLA$** computer object in order to receive the appropriate logging within our SIEM. In this case I have enabled full auditing for this object.

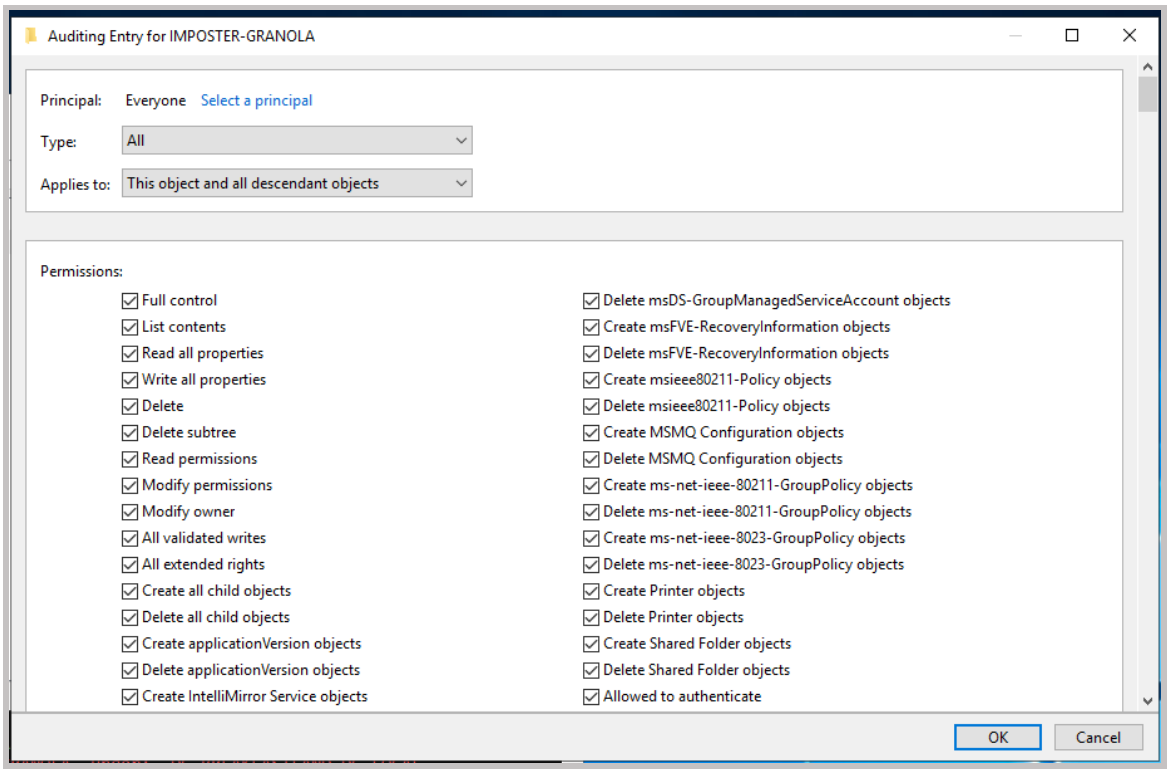


Figure 6 - Adding Auditing for IMPOSTER-GRANOLA

5.2.3 Modifying the Attribute (Attack)

To modify the SPN attribute directly, we will use the PowerMad toolset, leveraging the `Set-MachineAccountAttribute` cmdlet:

```
Set-MachineAccountAttribute -Attribute ServicePrincipalName -Value
'HOST/IMPOSTER-DEHYDRATOR.BREAKFASTLAND.LOCAL'
```

```
PS C:\Powermad-master> Set-MachineAccountAttribute -Attribute ServicePrincipalName -Value HOST/IMPOSTER-
cmdlet Set-MachineAccountAttribute at command pipeline position 1
Supply values for the following parameters:
MachineAccount: IMPOSTER-GRANOLA
[+] Machine account IMPOSTER-GRANOLA attribute ServicePrincipalName updated
```

Figure 7 - Modifying SPN Attribute

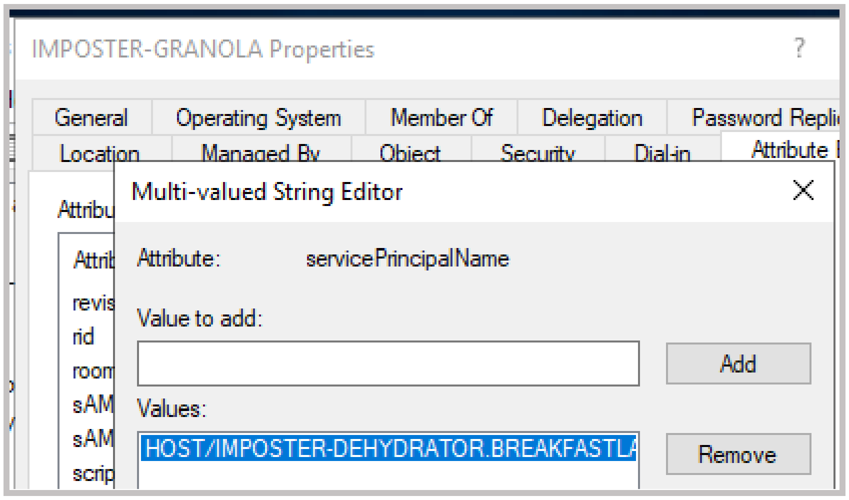
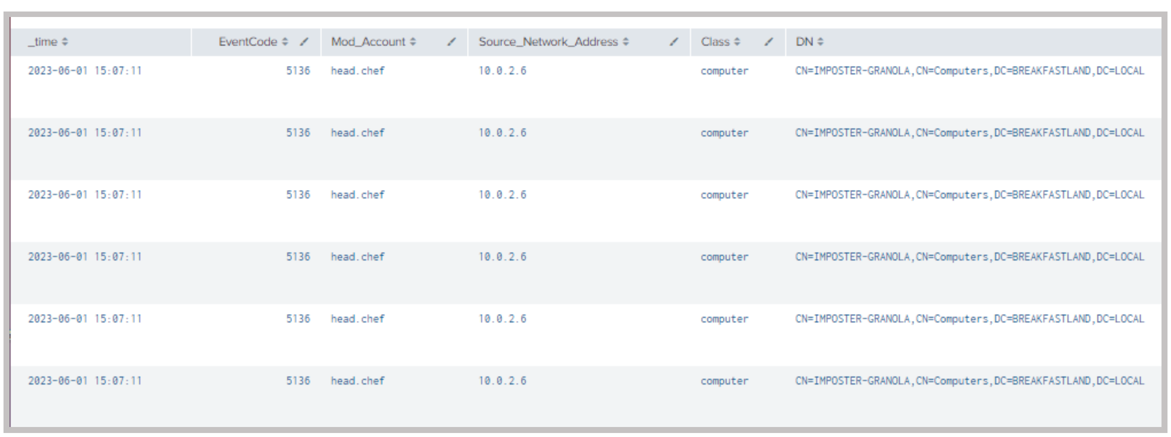


Figure 8 - ServicePrincipalName Attribute Post Modification

5.2.4 Building the Detections

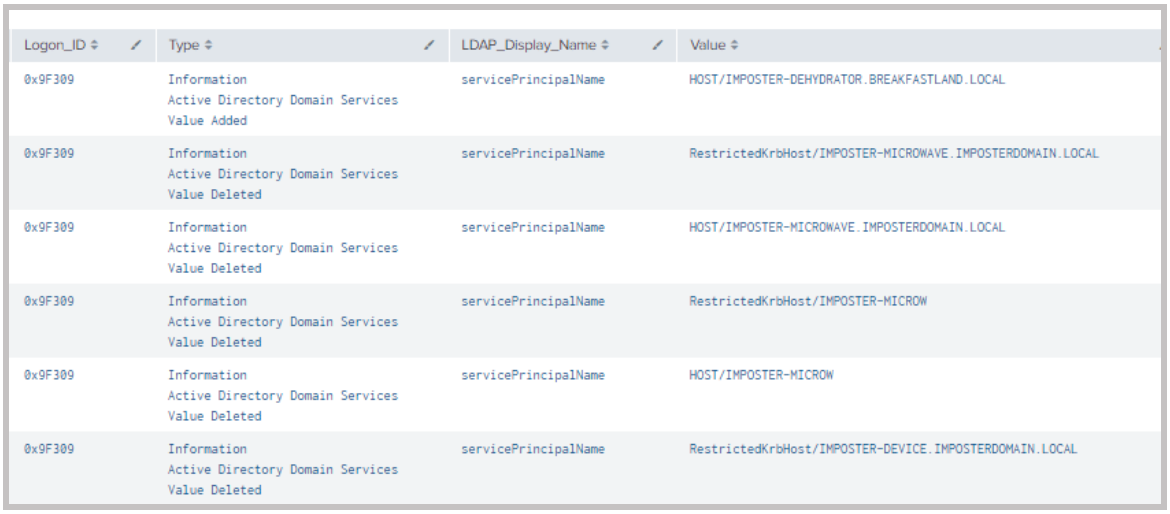
5.2.4.1 Detection With Event IDs 5136 and 4662

```
index=main ((EventCode=5136 AND LDAP_Display_Name=servicePrincipalName)
OR (EventCode=4624 AND Account_Name!="*$" AND Account_Name!="ANONYMOUS
LOGON" AND Account_Name!="SYSTEM"))
| eval Logon_ID=if(EventCode==4624,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval Mod_Account=if(EventCode==4624,mvindex(Account_Name,-1),
mvindex(Account_Name,-1))
| join type=outer Logon_ID
        [ search (EventCode=5136) OR (EventCode=4624)
        | stats count by Logon_ID, Account_Name, Source_Network_Address
        | table Account_Name,Logon_ID, Source_Network_Address]
| table _time, EventCode, Mod_Account, Source_Network_Address, Class, DN,
Logon_ID, Type, LDAP_Display_Name, Value
| where  len(Class)>0
```



_time	EventCode	Mod_Account	Source_Network_Address	Class	DN
2023-06-01 15:07:11	5136	head.chef	10.0.2.6	computer	CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL
2023-06-01 15:07:11	5136	head.chef	10.0.2.6	computer	CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL
2023-06-01 15:07:11	5136	head.chef	10.0.2.6	computer	CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL
2023-06-01 15:07:11	5136	head.chef	10.0.2.6	computer	CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL
2023-06-01 15:07:11	5136	head.chef	10.0.2.6	computer	CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL
2023-06-01 15:07:11	5136	head.chef	10.0.2.6	computer	CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL

Figure 9 - Detection with Event IDs 5136 and 4624 (1)

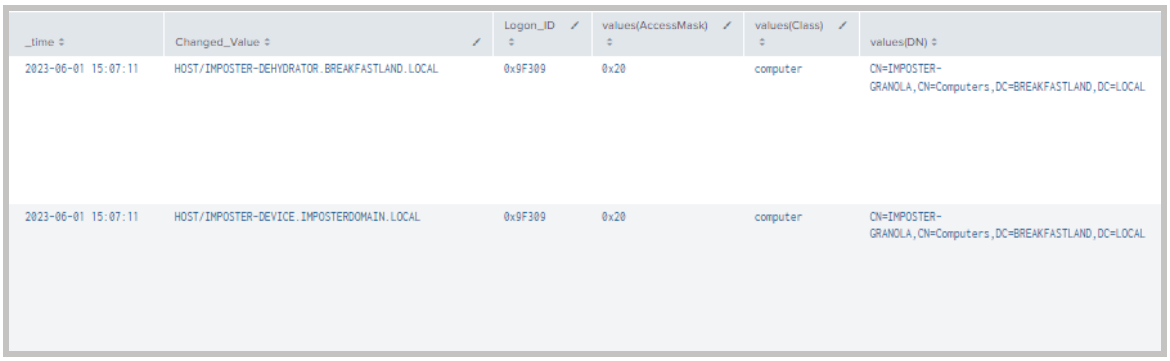


Logon_ID	Type	LDAP_Display_Name	Value
0x9F309	Information Active Directory Domain Services Value Added	servicePrincipalName	HOST/IMPOSTER-DEHYDRATOR.BREAKFASTLAND.LOCAL
0x9F309	Information Active Directory Domain Services Value Deleted	servicePrincipalName	RestrictedKrbHost/IMPOSTER-MICROWAVE.IMPOSTERDOMAIN.LOCAL
0x9F309	Information Active Directory Domain Services Value Deleted	servicePrincipalName	HOST/IMPOSTER-MICROWAVE.IMPOSTERDOMAIN.LOCAL
0x9F309	Information Active Directory Domain Services Value Deleted	servicePrincipalName	RestrictedKrbHost/IMPOSTER-MICROW
0x9F309	Information Active Directory Domain Services Value Deleted	servicePrincipalName	HOST/IMPOSTER-MICROW
0x9F309	Information Active Directory Domain Services Value Deleted	servicePrincipalName	RestrictedKrbHost/IMPOSTER-DEVICE.IMPOSTERDOMAIN.LOCAL

Figure 10 - Detection with Event IDs 5136 and 4624 (2)

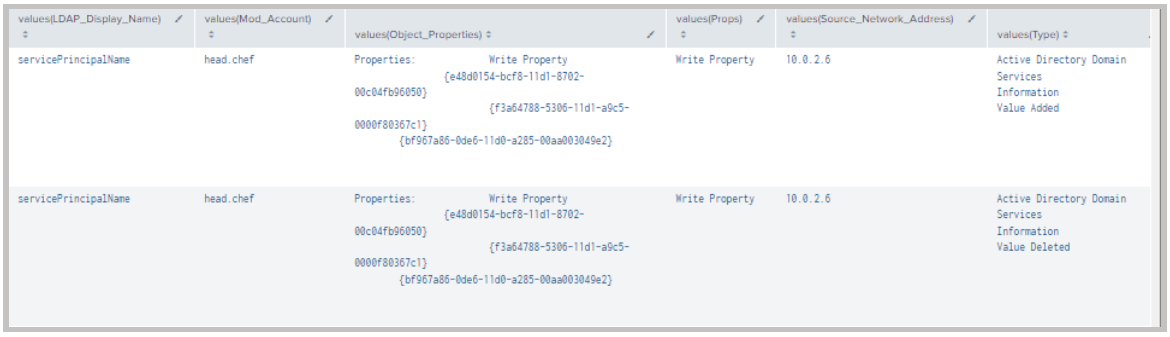
5.2.4.2Detection With Event IDs 5136, 4624, and 4662

```
index=main ((EventCode=5136 AND LDAP_Display_Name=servicePrincipalName)
OR (EventCode=4624 AND Account_Name!="*$" AND Account_Name!="ANONYMOUS
LOGON" AND Account_Name!="SYSTEM") OR (EventCode=4662 AND
Access_Mask=0x20))
| eval Logon_ID=if(EventCode==4624,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval Mod_Account=if(EventCode==4624,mvindex(Account_Name,-1),
mvindex(Account_Name,-1))
| eval Changed_Value=if(EventCode==5136,mvindex(Value,-1),
mvindex(Value,-1)) | join type=outer Logon_ID
        [ search (EventCode=5136) OR (EventCode=4624)
        | stats count by Logon_ID, Account_Name, Source_Network_Address
        | table Account_Name,Logon_ID, Source_Network_Address ]
| join type=outer Logon_ID
    [ search index=main Account_Name!=*$  EventCode=4662 Access_Mask =
0x20
    | eval Props=Properties
    | eval AccessMask=Access_Mask
    | eval ObjectType=Object_Type
    | eval ObjectName=Object_Name
    | rex field=Message
"(?<Object_Properties>(?ms)(?<=)Properties:(.*?)(?=Additional\s+))"
    |table Account_Name,Logon_ID,Props,AccessMask,ObjectType, ObjectName,
Object_Properties]
| table _time, Mod_Account, Source_Network_Address , Class, DN, Logon_ID,
Type, LDAP_Display_Name, Changed_Value, AccessMask, Props,
Object_Properties
| where  len(Class)>0
| stats values by _time, Changed_Value, Logon_ID
```



_time	Changed_Value	Logon_ID	values(AccessMask)	values(Class)	values(DN)
2023-06-01 15:07:11	HOST/IMPOSTER-DEHYDRATOR.BREAKFASTLAND.LOCAL	0x9F309	0x20	computer	CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL
2023-06-01 15:07:11	HOST/IMPOSTER-DEVICE.IMPOSTERDOMAIN.LOCAL	0x9F309	0x20	computer	CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL

Figure 11 – Detection with Event IDs 5136, 4662, 4624 (1)

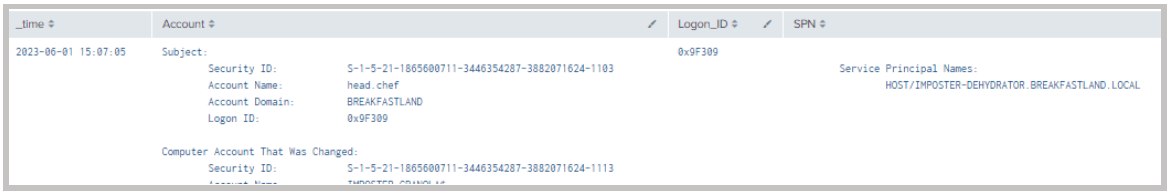


values(LDAP_Display_Name)	values(Mod_Account)	values(Object_Properties)	values(Props)	values(Source_Network_Address)	values(Type)
servicePrincipalName	head.chef	Properties: Write Property {e48d0154-bcf8-11d1-8702-00c04fb96050} {f3a64788-5306-11d1-a9c5-0000f80367c1} {bf967a86-0de6-11d0-a285-00aa003049e2}	Write Property	10.0.2.6	Active Directory Domain Services Information Value Added
servicePrincipalName	head.chef	Properties: Write Property {e48d0154-bcf8-11d1-8702-00c04fb96050} {f3a64788-5306-11d1-a9c5-0000f80367c1} {bf967a86-0de6-11d0-a285-00aa003049e2}	Write Property	10.0.2.6	Active Directory Domain Services Information Value Deleted

Figure 12 - Detection with Event IDs 5136, 4662, 4624 (2)

5.2.4.3Detection With Event ID 4742

```
index=main EventCode=4742   | rex field=Message
"(?<Account>(?ms).....................................................
.................Account\s+Name.*?(Account\s+Name:\s+)(\w+.........))"
| rex field=Message
"(?<SPN>(?ms)\s+Service\s+Principal\s+Name(.*).+?(?=Additional\s+))"    |
search SPN!="*Service Principal Names: -*" | table  _time, Account,
 Logon_ID, SPN
```



_time	Account	Logon_ID	SPN
2023-06-01 15:07:05	Subject: Security ID: S-1-5-21-1865600711-3446354287-3882071624-1103 Account Name: head.chef Account Domain: BREAKFASTLAND Logon ID: 0x9F309 Computer Account That Was Changed: Security ID: S-1-5-21-1865600711-3446354287-3882071624-1113	0x9F309	Service Principal Names: HOST/IMPOSTER-DEHYDRATOR.BREAKFASTLAND.LOCAL

Figure 13 - Detection With Event ID 4742

5.3 Writing to ms-DS-Allowed-to-Delegate-To

5.3.1 Background

The msDS-AllowedToDelegateTo attribute contains a list of Service Principal Names that are used to configure services so they can obtain Kerberos Tickets used for “Constrained Delegation” for the targeted account.

5.3.2 Modifying the Attribute (Attack)

For this particular attack/attribute modification, we will first create a second new machine account with PowerMad.

```
PS C:\Powermad-master> New-MachineAccount -MachineAccount COFFE
[+] Machine account COFFEPOT-PC added
```

Figure 14 - New Machine Account Creation

One thing to note with this attribute is that it cannot be modified unless the user making the change has the ***SeEnableDelegationPrivilege***. This article discusses the requirements in more detail and is an excellent read.

Because we are running these commands with a Domain Administrator account, I was able to modify the attribute.

```
PS C:\Powermad-master> Set-MachineAccountAttribute -Attribute msDS-AllowedToDelegateTo -Value COFFEPOT-PC$
cmdlet Set-MachineAccountAttribute at command pipeline position 1
Supply values for the following parameters:
MachineAccount: IMPOSTER-GRANOLA
[+] Machine account IMPOSTER-GRANOLA attribute msDS-AllowedToDelegateTo updated
```

Figure 15 - Modifying msDS-AllowedToDelegateTo Attribute

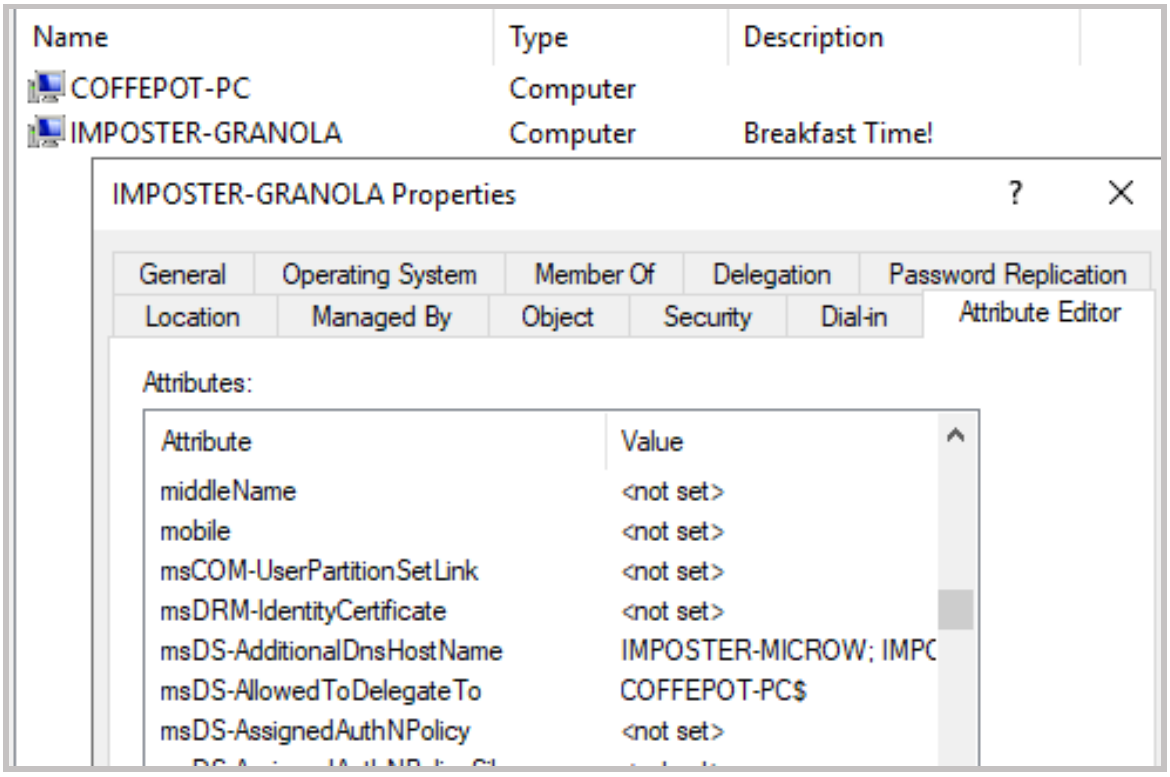


Figure 16 - Attribute Post Modification

5.3.3 Building the Detections

5.3.3.1 Detection With Event IDs 5136 and 4624

```
index=main ((EventCode=5136 AND LDAP_Display_Name=msDS-
AllowedToDelegateTo)  OR (EventCode=4624 AND Account_Name!="*$" AND
Account_Name!="ANONYMOUS LOGON" AND Account_Name!="SYSTEM"))
| eval Logon_ID=if(EventCode==4624,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval Mod_Account=if(EventCode==4624,mvindex(Account_Name,-1),
mvindex(Account_Name,-1))
| join type=outer Logon_ID
        [ search (EventCode=5136) OR (EventCode=4624)
        | stats count by Logon_ID, Account_Name, Source_Network_Address
        | table Account_Name,Logon_ID, Source_Network_Address]
| table _time, EventCode, Mod_Account, Source_Network_Address, Class, DN,
Logon_ID, Type, LDAP_Display_Name, Value
| where  len(Class)>0
```

_time	EventCode	Mod_Account	Source_Network_Address	Class
2023-06-01 12:50:31	5136	head.chef	10.0.2.6	computer

Figure 17 - Detection With Event IDs 5136 and 4624 (1)

DN	Logon_ID	Type	LDAP_Display_Name	Value
CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL	0x14411C	Information Active Directory Domain Services Value Added	msDS-AllowedToDelegateTo	COFFEPOT-PC$

Figure 18 - Detection With Event IDs 5136 and 4624 (2)

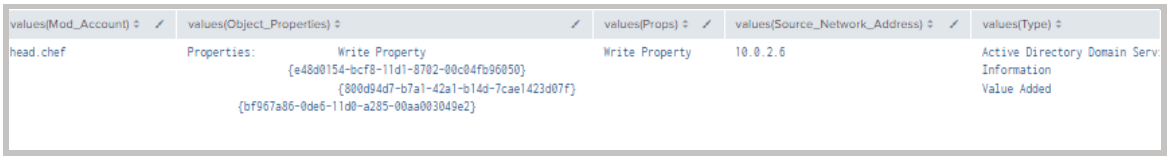
5.3.3.2 Detection With Event IDs 5136, 4624, and 4662

```
index=main ((EventCode=5136 AND LDAP_Display_Name=msDS-
AllowedToDelegateTo)  OR (EventCode=4624 AND Account_Name!="*$" AND
Account_Name!="ANONYMOUS LOGON" AND Account_Name!="SYSTEM") OR
(EventCode=4662 AND Access_Mask=0x20))
| eval Logon_ID=if(EventCode==4624,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval Mod_Account=if(EventCode==4624,mvindex(Account_Name,-1),
mvindex(Account_Name,-1))
| eval Changed_Value=if(EventCode==5136,mvindex(Value,-1),
mvindex(Value,-1))
| join type=outer Logon_ID
        [ search (EventCode=5136) OR (EventCode=4624)
        | stats count by Logon_ID, Account_Name, Source_Network_Address
        | table Account_Name,Logon_ID, Source_Network_Address ]
| join type=outer Logon_ID
    [ search index=main Account_Name!=*$  EventCode=4662 Access_Mask =
0x20
    | eval Props=Properties
    | eval AccessMask=Access_Mask
    | eval ObjectType=Object_Type
    | eval ObjectName=Object_Name
    | rex field=Message
```

```
"(?<Object_Properties>(?ms)(?<=)Properties:(.*?)(?=Additional\s+))"
    |table Account_Name,Logon_ID,Props,AccessMask,ObjectType, ObjectName,
Object_Properties]
| table _time, Mod_Account, Source_Network_Address , Class, DN, Logon_ID,
Type, LDAP_Display_Name, Changed_Value, AccessMask, Props,
Object_Properties
| where  len(Class)>0
| stats values by _time, Changed_Value
```

_time	Changed_Value	values(AccessMask)	values(Class)	values(DN)	values(LDAP_Display_Name)	values(Logon_ID)
2023-06-01 12:50:31	COFFEPOT-PC$	0x20	computer	CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL	msDS-AllowedToDelegateTo	0x14411C

Figure 19 - Detection With Event IDs 5136, 4662, 4624 (1)

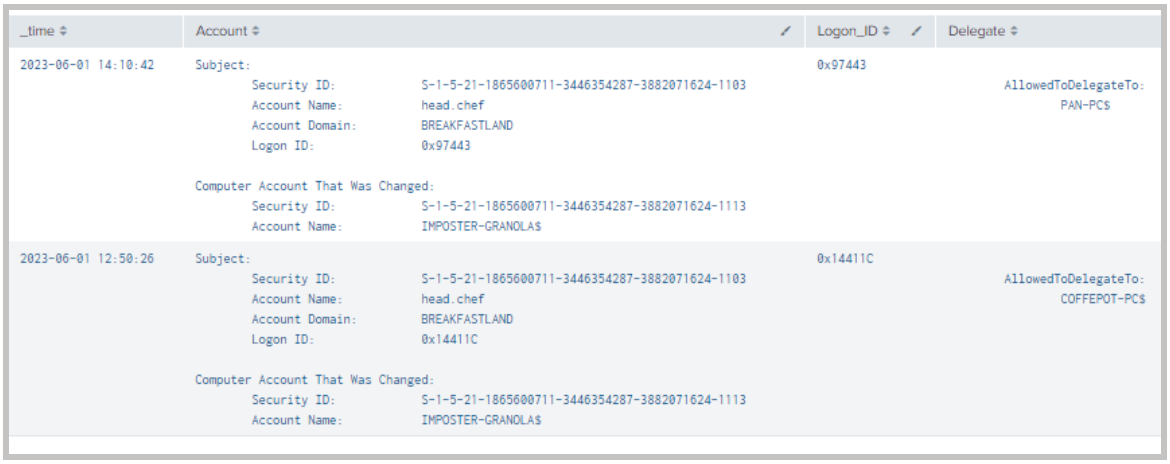


values(Mod_Account)	values(Object_Properties)	values(Props)	values(Source_Network_Address)	values(Type)
head.chef	Properties: Write Property {e48d0154-bcf8-11d1-8702-00c04fb96050} {800d94d7-b7a1-42a1-b14d-7cae1423d07f} {bf967a86-0de6-11d0-a285-00aa003049e2}	Write Property	10.0.2.6	Active Directory Domain Serv Information Value Added

Figure 20 - Detection With Event IDs 5136, 4662, 4624 (2)

5.3.3.3Detection With Event ID 4742

```
index=main EventCode=4742
| rex field=Message
"(?<Account>(?ms).....................................................
.................Account\s+Name.*?(Account\s+Name:\s+)(\w+.........))"
| rex field=Message
"(?<Delegate>(?ms)\s+AllowedToDelegateTo(.*).+?(?=Old\s+))"
| search Delegate!="*AllowedToDelegateTo: -*"
| table  _time, Account, Logon_ID, Delegate
```



_time	Account	Logon_ID	Delegate
2023-06-01 14:10:42	Subject: Security ID: S-1-5-21-1865600711-3446354287-3882071624-1103 Account Name: head.chef Account Domain: BREAKFASTLAND Logon ID: 0x97443 Computer Account That Was Changed: Security ID: S-1-5-21-1865600711-3446354287-3882071624-1113 Account Name: IMPOSTER-GRANOLA$	0x97443	AllowedToDelegateTo: PAN-PC$
2023-06-01 12:50:26	Subject: Security ID: S-1-5-21-1865600711-3446354287-3882071624-1103 Account Name: head.chef Account Domain: BREAKFASTLAND Logon ID: 0x14411C Computer Account That Was Changed: Security ID: S-1-5-21-1865600711-3446354287-3882071624-1113 Account Name: IMPOSTER-GRANOLA$	0x14411C	AllowedToDelegateTo: COFFEPOT-PC$

Figure 21 - Detection With Event ID 4742

5.4 Shadow Credentials - Writing to ms-DS-Key-Credential-Link

5.4.1 Background

The msDS-KeyCredentialLink attribute can be used to store a key-based alternate set of credentials for a given user object—in this case, our victim account **dacled.egg**.

5.4.2 Modifying the Attribute (Attack)

To modify the ***msDS-KeyCredentialLink*** attribute, we will be primarily following the attack walkthrough here.

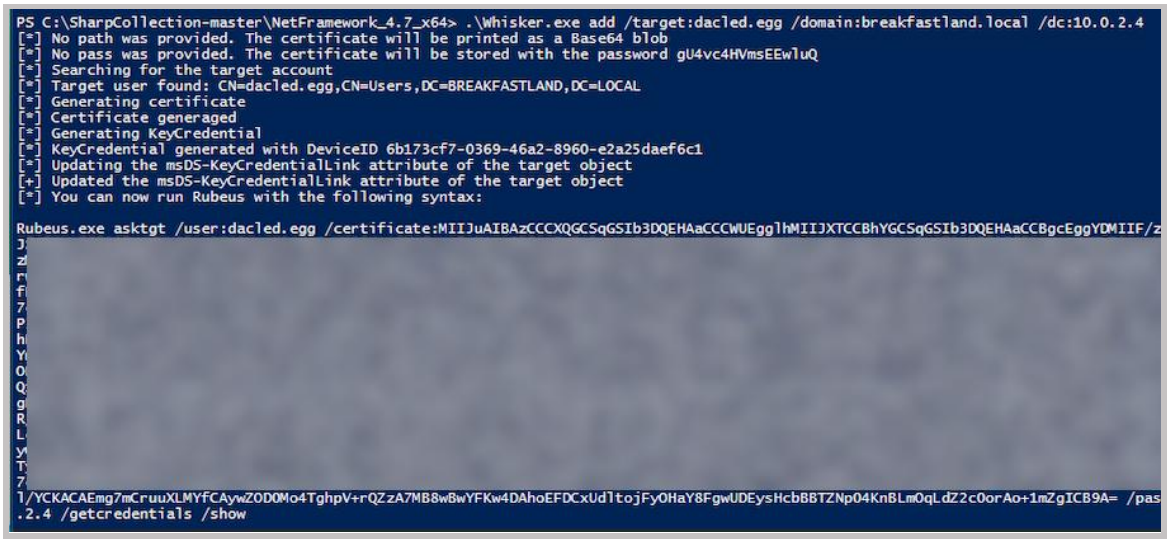


Figure 22 - Executing Shadow Credentials Attack With Whisker

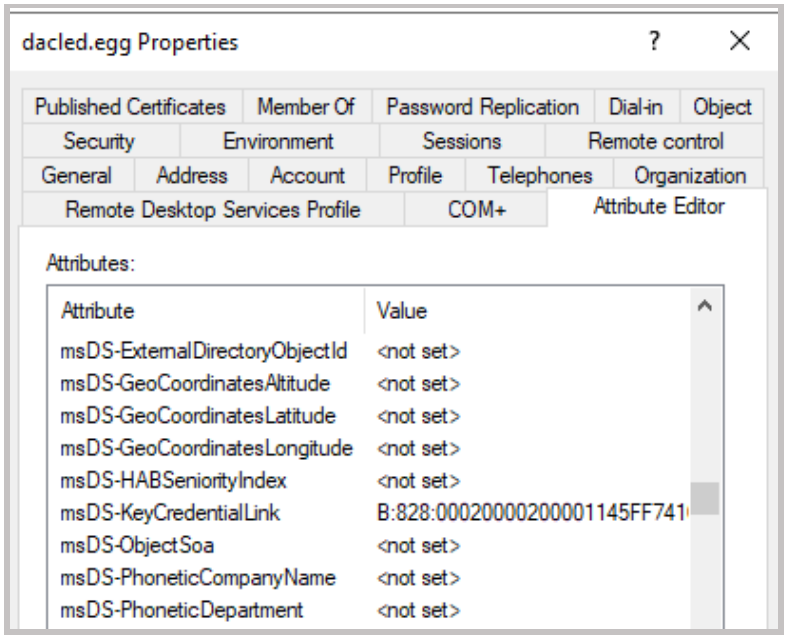


Figure 23 - Change in Object Post Attack

5.4.3 Building the Detections

5.4.3.1 Detection With Event IDs 5136 and 4624

```
index=main ((EventCode=5136 AND LDAP_Display_Name=msDS-
KeyCredentialLink)  OR (EventCode=4624 AND Account_Name!="*$" AND
Account_Name!="ANONYMOUS LOGON" AND Account_Name!="SYSTEM"))
| eval Logon_ID=if(EventCode==4624,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval Mod_Account=if(EventCode==4624,mvindex(Account_Name,-1),
mvindex(Account_Name,-1))
| join type=outer Logon_ID
        [ search (EventCode=5136) OR (EventCode=4624)
        | stats count by Logon_ID, Account_Name,
Source_Network_Address
        | table Account_Name,Logon_ID, Source_Network_Address]
| table _time, EventCode, Mod_Account, Source_Network_Address, Class, DN,
Logon_ID, Type, LDAP_Display_Name, Value
| where  len(Class)>0
```

_time	EventCode	Mod_Account	Source_Network_Address	Class	DN
2023-06-01 17:09:45	5136	head.chef	10.0.2.6	user	CN=dacled.egg,CN=Users,DC=BREAKFASTLAND,DC=LOCAL

Figure 24 - Detection of Modification for msKeyCredentialLink (1)

Logon_ID	Type	LDAP_Display_Name	Value
0x1EBB7A	Information Active Directory Domain Services Value Added	msDS-KeyCredentialLink	B:828:<Binary>:CN=dacled.egg,CN=Users,DC=BREAKFASTLAND,DC=LOCAL

Figure 25 - Detection of Modification for msKeyCredentialLink (2)

As a quick note, because we have not specified a class for this query—and for other queries—you do not need to write a separate query to pick up modifications to computer objects, as they will be picked up automatically.

_time	EventCode	Mod_Account	Source_Netw	Class	DN	Logon_ID	LDAP_Display_Name
2023-06-01 17:18:17	5136	head.chef	10.0.2.6	computer	CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL	0x2336A6	msDS-KeyCredentialLink
2023-06-01 17:09:45	5136	head.chef	10.0.2.6	user	CN=dacled.egg,CN=Users,DC=BREAKFASTLAND,DC=LOCAL	0x1EBB7A	msDS-KeyCredentialLink

Figure 26 - msKeyCredentialLink Modification Showing Changes to User and Computer Objects

5.5 Logon Script (Script-Path)

5.5.1 Background

The scriptPath attribute specifies the path designated for a user or computer object's logon script.

5.5.2 Modifying the Attribute (Attack)

As previously, we will modify the ***scriptPath*** attribute with the following PowerMad Command:

```
Set-MachineAccountAttribute -Attribute scriptPath -Value
'C:\TheFridge\Food.exe'
```

```
PS C:\Powermad-master> Set-MachineAccountAttribute -Attribute scriptPath -Value C:\TheFridge\Food.exe
cmdlet Set-MachineAccountAttribute at command pipeline position 1
Supply values for the following parameters:
MachineAccount: IMPOSTER-GRANOLA
[+] Machine account IMPOSTER-GRANOLA attribute scriptPath updated
```

Figure 27 - Modifying scriptPath Attribute

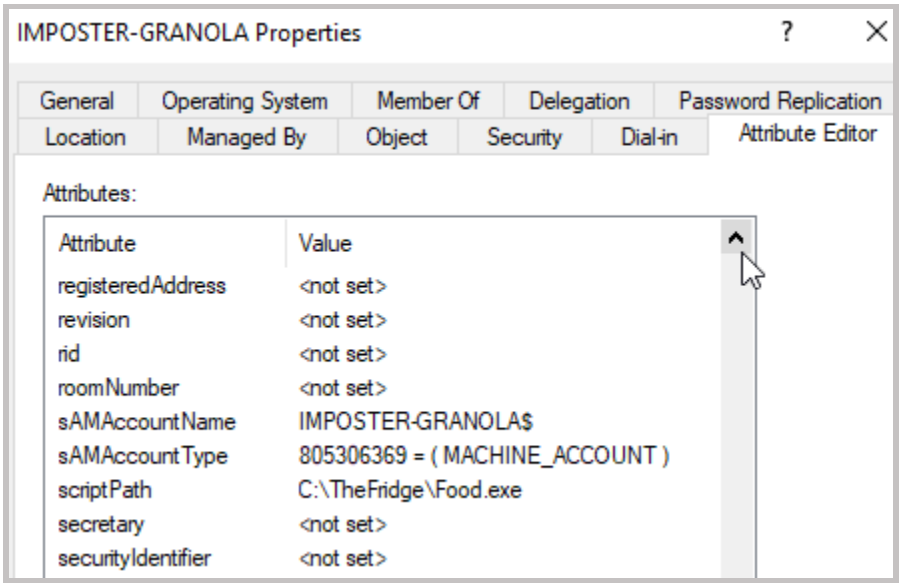


Figure 28 - scriptPath Attribute Post Modification

5.5.3 Building the Detections

5.5.3.1 Detection with Event IDs 5136 and 4624

```
index=main ((EventCode=5136 AND LDAP_Display_Name=scriptPath)  OR
(EventCode=4624 AND Account_Name!="*$" AND Account_Name!="ANONYMOUS
LOGON" AND Account_Name!="SYSTEM"))
| eval Logon_ID=if(EventCode==4624,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval Mod_Account=if(EventCode==4624,mvindex(Account_Name,-1),
mvindex(Account_Name,-1))
```

```
| join type=outer Logon_ID
        [ search (EventCode=5136) OR (EventCode=4624)
        | stats count by Logon_ID, Account_Name, Source_Network_Address
        | table Account_Name,Logon_ID, Source_Network_Address]
| table _time, EventCode, Mod_Account, Source_Network_Address, Class, DN,
Logon_ID, Type, LDAP_Display_Name, Value
| where  len(Class)>0
```

_time	EventCode	Mod_Account	Source_Network_Address	Class	DN
2023-06-01 17:31:36	5136	head.chef	10.0.2.6	computer	CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL

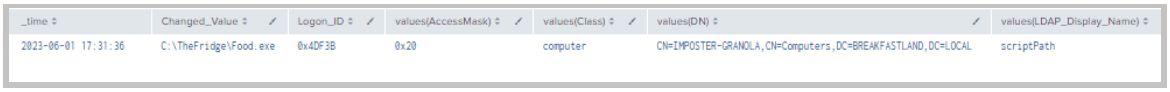
Figure 29 - Detection With Event IDs 5136 and 4624 (1)

Logon_ID	Type	LDAP_Display_Name	Value
0x4DF3B	Information Active Directory Domain Services Value Added	scriptPath	C:\TheFridge\Food.exe

Figure 30 - Detection With Event IDs 5136 and 4624 (2)

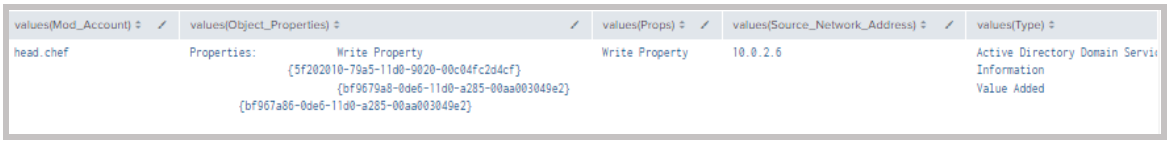
5.5.3.2Detection With Event IDs 5136, 4624, and 4662

```
index=main ((EventCode=5136 AND LDAP_Display_Name=scriptPath)  OR
(EventCode=4624 AND Account_Name!="*$" AND Account_Name!="ANONYMOUS
LOGON" AND Account_Name!="SYSTEM") OR (EventCode=4662 AND
Access_Mask=0x20))
| eval Logon_ID=if(EventCode==4624,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval Mod_Account=if(EventCode==4624,mvindex(Account_Name,-1),
mvindex(Account_Name,-1))
| eval Changed_Value=if(EventCode==5136,mvindex(Value,-1),
mvindex(Value,-1))
| join type=outer Logon_ID
        [ search (EventCode=5136) OR (EventCode=4624)
        | stats count by Logon_ID, Account_Name, Source_Network_Address
        | table Account_Name,Logon_ID, Source_Network_Address ]
| join type=outer Logon_ID
    [ search index=main Account_Name!=*$  EventCode=4662 Access_Mask =
0x20
    | eval Props=Properties
    | eval AccessMask=Access_Mask
    | eval ObjectType=Object_Type
    | eval ObjectName=Object_Name
    | rex field=Message
"(?<Object_Properties>(?ms)(?<=)Properties:(.*?)(?=Additional\s+))"
    |table Account_Name,Logon_ID,Props,AccessMask,ObjectType, ObjectName,
Object_Properties]
| table _time, Mod_Account, Source_Network_Address , Class, DN, Logon_ID,
Type, LDAP_Display_Name, Changed_Value, AccessMask, Props,
Object_Properties
| where  len(Class)>0
| stats values by _time, Changed_Value, Logon_ID
```



_time	Changed_Value	Logon_ID	values(AccessMask)	values(Class)	values(DN)	values(LDAP_Display_Name)
2023-06-01 17:31:36	C:\TheFridge\Food.exe	0x4DF3B	0x20	computer	CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL	scriptPath

Figure 31 - Detection With Event IDs 5136, 4662, 4624 (1)

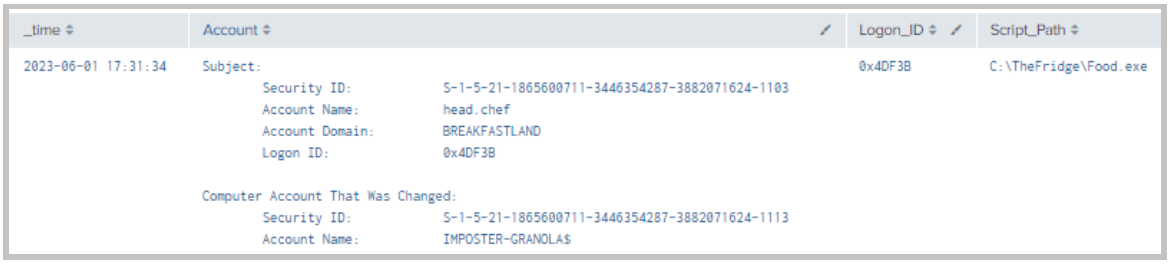


values(Mod_Account)	values(Object_Properties)	values(Props)	values(Source_Network_Address)	values(Type)
head.chef	Properties: Write Property {5f202010-79a5-11d0-9020-00c04fc2d4cf} {bf9679a8-0de6-11d0-a285-00aa003049e2} {bf967a86-0de6-11d0-a285-00aa003049e2}	Write Property	10.0.2.6	Active Directory Domain Servi Information Value Added

Figure 32 - Detection With Event IDs 5136, 4662, 4624 (2)

5.5.3.3Detection with Event ID 4742

```
index=main EventCode=4742 Script_Path!="*-*"
| rex field=Message
"(?<Account>(?ms).....................................................
.................Account\s+Name.*?(Account\s+Name:\s+)(\w+.........))"
| table  _time, Account, Logon_ID, Script_Path
```



_time	Account	Logon_ID	Script_Path
2023-06-01 17:31:34	Subject: Security ID: S-1-5-21-1865600711-3446354287-3882071624-1103 Account Name: head.chef Account Domain: BREAKFASTLAND Logon ID: 0x4DF3B Computer Account That Was Changed: Security ID: S-1-5-21-1865600711-3446354287-3882071624-1113 Account Name: IMPOSTER-GRANOLA$	0x4DF3B	C:\TheFridge\Food.exe

Figure 33 - Detection With Event ID 4742

5.6 ms-TS-Initial-Program

5.6.1 Background

The msTSInitialProgram attribute stores data for applications that should be started upon initial logon. This information will include the path and file name of the application(s).

5.6.2 Modifying the Attribute (Attack)

As previously, we will modify the ***msTSInitialProgram*** attribute with the following PowerMad Command:

```
Set-MachineAccountAttribute -Attribute msTSInitialProgram -Value
'C:\TheFridge\More_Food.exe'
```

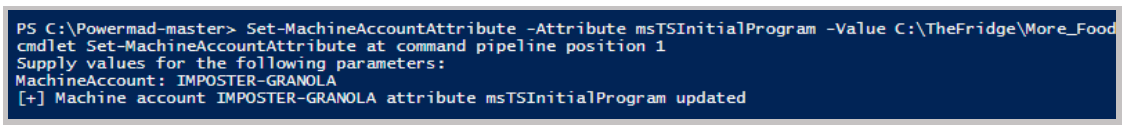


Figure 34 - Modifying msTSInitialProgram

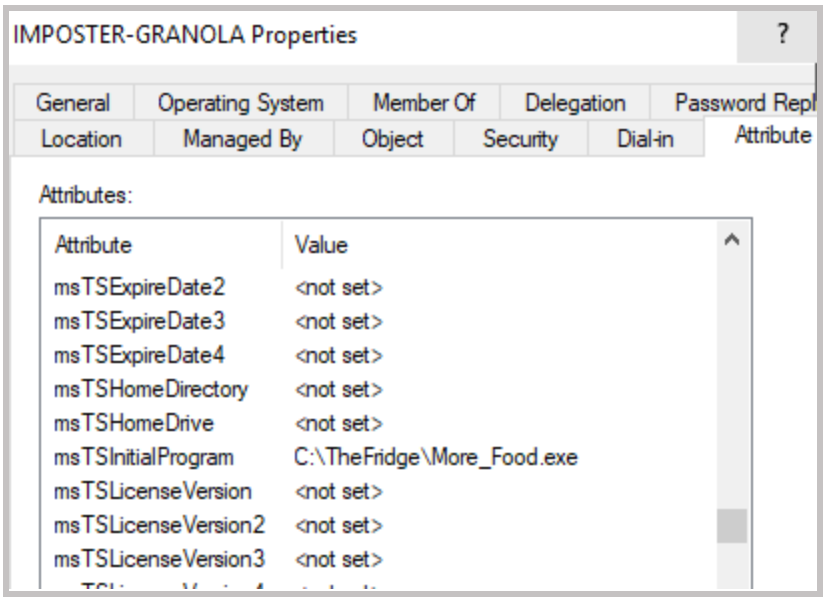


Figure 35 - msTSInitialProgram Post Modification

5.6.3 Building the Detections

5.6.3.1Detection With Event IDs 5136 and 4624

```
index=main ((EventCode=5136 AND LDAP_Display_Name=msTSInitialProgram)  OR
(EventCode=4624 AND Account_Name!="*$" AND Account_Name!="ANONYMOUS
LOGON" AND Account_Name!="SYSTEM"))
| eval Logon_ID=if(EventCode==4624,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval Mod_Account=if(EventCode==4624,mvindex(Account_Name,-1),
mvindex(Account_Name,-1))
| join type=outer Logon_ID
        [ search (EventCode=5136) OR (EventCode=4624)
        | stats count by Logon_ID, Account_Name, Source_Network_Address
        | table Account_Name,Logon_ID, Source_Network_Address]
| table _time, EventCode, Mod_Account, Source_Network_Address, Class, DN,
Logon_ID, Type, LDAP_Display_Name, Value
| where  len(Class)>0
```

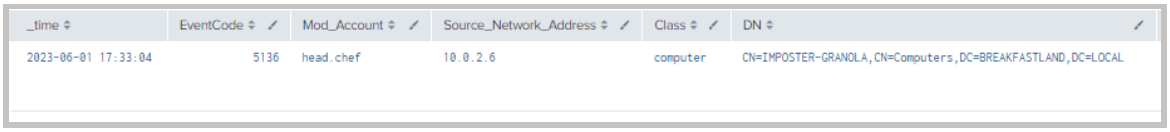


Figure 36 - Detection With Event IDs 5136 and 4624 (1)

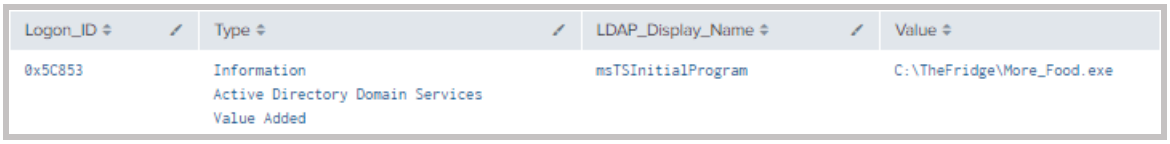
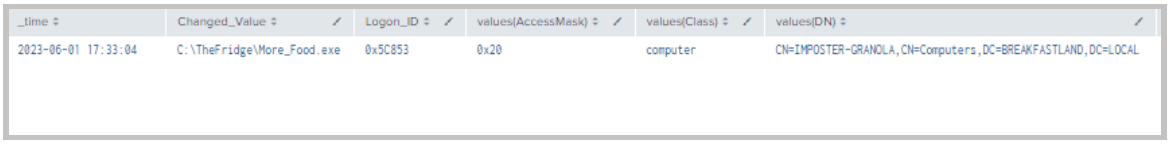


Figure 37 - Detection With Event IDs 5136 and 4624 (2)

5.6.3.2 Detection With Event IDs 5136, 4624, and 4662

```
index=main ((EventCode=5136 AND LDAP_Display_Name=msTSInitialProgram)  OR
(EventCode=4624 AND Account_Name!="*$" AND Account_Name!="ANONYMOUS
LOGON" AND Account_Name!="SYSTEM") OR (EventCode=4662 AND
Access_Mask=0x20))
| eval Logon_ID=if(EventCode==4624,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval Mod_Account=if(EventCode==4624,mvindex(Account_Name,-1),
mvindex(Account_Name,-1))
| eval Changed_Value=if(EventCode==5136,mvindex(Value,-1),
mvindex(Value,-1))
| join type=outer Logon_ID
        [ search (EventCode=5136) OR (EventCode=4624)
        | stats count by Logon_ID, Account_Name, Source_Network_Address
        | table Account_Name,Logon_ID, Source_Network_Address ]
| join type=outer Logon_ID
    [ search index=main Account_Name!=*$  EventCode=4662 Access_Mask =
0x20
    | eval Props=Properties
    | eval AccessMask=Access_Mask
    | eval ObjectType=Object_Type
    | eval ObjectName=Object_Name
    | rex field=Message
"(?<Object_Properties>(?ms)(?<=)Properties:(.*?)(?=Additional\s+))"
    |table Account_Name,Logon_ID,Props,AccessMask,ObjectType, ObjectName,
Object_Properties]
| table _time, Mod_Account, Source_Network_Address , Class, DN, Logon_ID,
Type, LDAP_Display_Name, Changed_Value, AccessMask, Props,
Object_Properties
| where  len(Class)>0
| stats values by _time, Changed_Value, Logon_ID
```



_time	Changed_Value	Logon_ID	values(AccessMask)	values(Class)	values(DN)
2023-06-01 17:33:04	C:\TheFridge\More_Food.exe	0x5C853	0x20	computer	CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL

Figure 38 - Detection With Event IDs 5136, 4662, 4624 (1)

values(LDAP_Display_Name)	values(Mod_Account)	values(Object_Properties)	values(Props)	values(Source_Network_Address)	values(Type)
msTSInitialProgram	head.chef	Properties: Write Property {771727b1-31b8-4cdf-ae62-4fe39fadf89e} {9201ac6f-1d69-4dfb-802e-d95510109599} {bf967a86-0de6-11d0-a285-00aa003049e2}	Write Property	10.0.2.6	Active Directory Domain Serv Information Value Added

Figure 39 - Detection With Event IDs 5136, 4662, 4624 (2)

5.7 GPO Abuse - Group-Policy-Container Class

5.7.1 Background

groupPolicyContainer is an AD Schema class that is modified when a GPO is updated through the Group Policy Editor. While modifying GPOs is a normal administrative task, it can also be abused by attackers who may use scheduled tasks or other GPO features to establish persistence or move laterally through the network.

5.7.2 Modifying the Attribute (Attack)

While GPO can be modified through the GUI, we are going to leverage the tools mentioned in the Hacker Recipes to remotely modify a GPO from a machine connected to the network.

Using GPOwned.py, we first need to procure the "unique ID/Name" of the GPO we are going to be attacking. The syntax is simple:

```
python3 GPOwned.py -u cheese.omlette -p <password> -d breakfastland.local
-dc-ip 10.0.2.4 -gpcmachine -listhpo
```

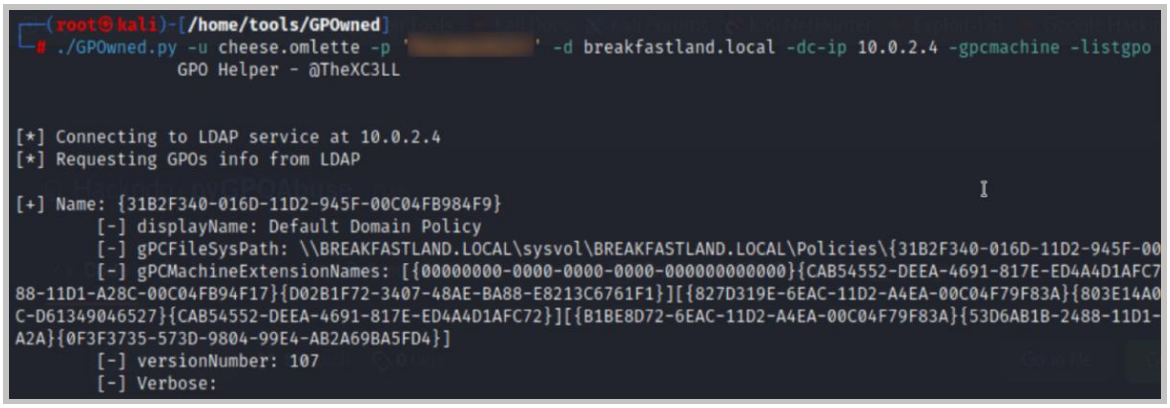


Figure 40 - GPOwned Output

Once you've enumerated a list of GPOs on the domain, you can identify the "Name" —in our case the Default Domain Policy GPO— of the GPO you wish to modify, and then you can run the following command:

```
python3 GPOwned.py -u head.chef -p <password> -d breakfastland.local -dc-
ip 10.0.2.4 -gpcmachine -gpoimmtask -name '{31B2F340-016D-11D2-945F-
00C04FB98}' -author 'BREAKFASTLAND\Domain Admins' -taskname
'ImaGPOAttack' -taskdescription 'For the blogs!' -dstpath
'c:\windows\system32\notepad.exe'
```

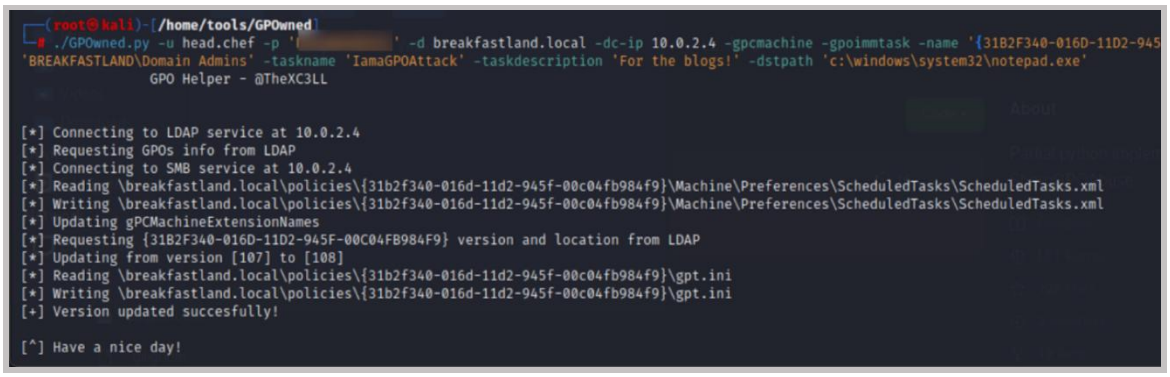


Figure 41 - GPOwned GPO Modification

Note: This attack was run originally as a non-privileged user and was not successful. It was successful as a privileged user. Second, the author of this script notes that it can be a bit buggy and should be used with precaution in production environments. Specifically, the bug we encountered was that the script would not pick up or drop GPOs that were new or deleted. Thus, keep in mind that there may be removed GPOs that are listed but no longer exist.

5.7.3 Building the Detections

At a very basic level, we can detect changes to the ***groupPolicyContainer*** "class" using Event ID 5136, as we have done with the majority of our previously built detections.

```
index=main EventCode=5136 Class=groupPolicyContainer
|table _time, EventCode, GUID, Class, Value, Type, DN, Correlation_ID
```

_time	EventCode	GUID	Class	Value	Type	DN	Correlation_ID
2023-09-01 16:27:39	5136	{6fd1273f-b7ef-43d4-9a20-f28a92ec69cc}	groupPolicyContainer	143	Information Active Directory Domain Services Value Added	CN={31B2F340-016D-11D2-945F-00C04FB984F9},CN=Policies,CN=System,DC=BREAKFASTLAND,DC=LOCAL	{fceb8d21-6b8f-4b3b-9c46-2c6a172c0b2c}
2023-09-01 16:27:39	5136	{6fd1273f-b7ef-43d4-9a20-f28a92ec69cc}	groupPolicyContainer	142	Information Active Directory Domain Services Value	CN={31B2F340-016D-11D2-945F-00C04FB984F9},CN=Policies,CN=System,DC=BREAKFASTLAND,DC=LOCAL	{fceb8d21-6b8f-4b3b-9c46-2c6a172c0b2c}

Figure 42 - Basic Query to Detect Changes to groupPolicyContainer

However, here we begin to run into some challenges with the limitations of Event ID 5136—namely, that while we can see evidence that the Group Policy GUID targeted in our attack was changed, we cannot see what exactly was changed or where it was changed from.

Given the breadth of function within GPO, this is critical to know in order to facilitate timely incident response and to assist analysts by adequately communicating information needs to AD/GPO Administrators.

To get the information we need, such a source IP address of the attacking host, as well as the change that was actually made to the GPO, we need to leverage Event ID 5145 and Event ID 4662.

Note: See the "Windows Events" section under "Logging Setup" within this blog for specifics on how to enable this logging.

In this case, the two (2) added events provide us with the following telemetry:

Event ID 5145:

- The Relative Target Name contained within this Event ID provides us with additional specifics on the actual network share/file/object accessed within our targeted GPO.
- Provides a source IP address

Event ID 4662:

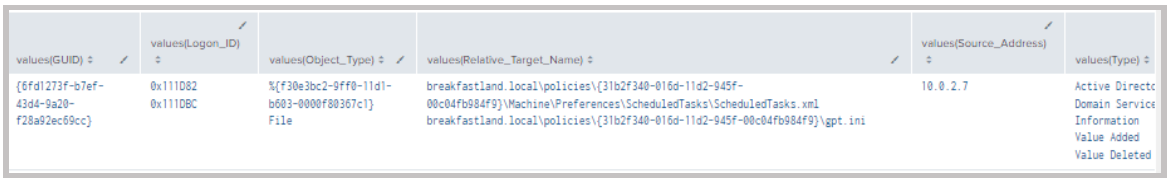
- Gives additional contextual data about the object/class/attributes involved
- Can be omitted if telemetry from Event IDs 5136 and 5145 is sufficient for organizational needs

5.7.3.1Detection With Event IDs 5136, 5145, and 4662

```
index=main ((EventCode=5136 AND  Class=groupPolicyContainer AND
(Type="Value Added" OR Type="Value Deleted"))  OR (EventCode=5145 AND
Accesses="WriteData (or AddFile)") OR (EventCode=4662 AND
Access_Mask=0x20 AND Object_Type="%{f30e3bc2-9ff0-11d1-b603-
0000f80367c1}"))
| eval new_time =strftime(_time, "%b %d, %Y %I:%M %p")
| table  new_time, Source_Address, Logon_ID, Account_Name, EventCode,
GUID, DN, Correlation_ID, Type, Relative_Target_Name, Access_Mask,
Object_Type, Class
| stats values by new_time
|sort by new_time
```

new_time	values(Access_Mask)	values(Account_Name)	values(Class)	values(Correlation_ID)	values(DN)	values(EventCode)
Sep 01, 2023 04:27 PM	0x2 0x20	head.chef	groupPolicyContainer	{5b7fa806-35bc-470b-8c40-930c7ac997cd} {fceb8d21-6b8f-4b3b-9c46-2c6a172c0b2c}	CN={31B2F340-016D-11D2-945F-00C04FB984F9},CN=Policies,CN=System,DC=BREAKFASTLAND,DC=LOCAL	4662 5136 5145

Figure 43 - Detecting Modifications to groupPolicyContainer Object Complex (1)



values(GUID)	values(Logon_ID)	values(Object_Type)	values(Relative_Target_Name)	values(Source_Address)	values(Type)
{6fd1273f-b7ef-43d4-9a20-f28a92ec69cc}	0x111D82 0x111DBC	%{f30e3bc2-9ff0-11d1-b603-0000f80367c1} File	breakfastland.local\policies\{31b2f340-016d-11d2-945f-00c04fb984f9}\Machine\Preferences\ScheduledTasks\ScheduledTasks.xml breakfastland.local\policies\{31b2f340-016d-11d2-945f-00c04fb984f9}\gpt.ini	10.0.2.7	Active Directc Domain Service Information Value Added Value Deleted

Figure 44 - Detecting Modifications to groupPolicyContainer Object Complex (2)

5.8 AddMember

5.8.1 Background

In this case, AddMember refers to the PowerView cmdlet or Linux RPC commands that can be used to perform the attack, and not necessarily the attribute/object misconfiguration that we are actually leveraging.

Mores specifically, ***AddMember*** is referring to the collective permissions granted to a security principal based on the security descriptor definition language (SDDL)/ACE defined for the group object based on the following AD permissions:

- GenericAll
- GenericWrite
- Self
- AllExtendedRights—*Note: When we experimented with modifying group membership by allowing for AllExtendedRights without adding additional permissions, we were unable to modify the group membership.*

OR

- Self-Membership

5.8.2 Modifying the Objects (Attack) - Generic Write

Once again, this blog post assumes that an attacker has discovered domain objects that are already misconfigured to allow for exploitation. However, in this case, we will need to perform the misconfiguration to our Domain Admins group object discretionary access control list (DACL) prior to running our "attack."

As done previously, we first created a new standard user account to easily track changes within our SIEM.

```
PS C:\PowerSploit-master\Recon> net user imposter.oatmeal [redacted] /domain /add
The request will be processed at a domain controller for domain BREAKFASTLAND.LOCAL.

The command completed successfully.
```

Figure 45 - New Imposter Account

Then, using our Domain Admin account **head.chef**, we modified the Domain Admins group directly through Active Directory Users and Computers (ADUC), giving the **imposter.oatmeal** account the ability to abuse the misconfigured DACL.

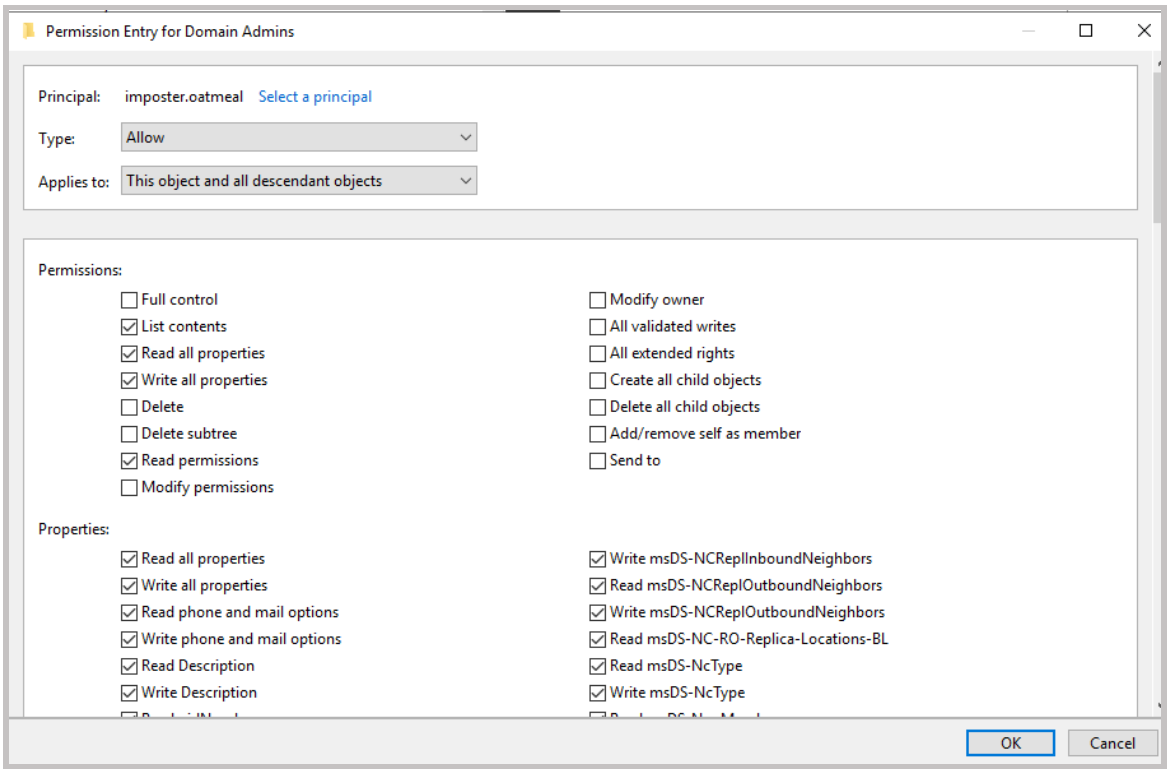


Figure 46 - Modifying User to Have Generic Write

And finally, to validate that we were able to abuse the misconfiguration to the Domain Admins DACL, we ran the attack as specified in the [Hacker Recipes](#) for ***AddMember***. As you can see, the **imposter.oatmeal** was able to successfully add itself to the Domain Admins group.

```
PS C:\PowerSploit-master\Recon> Add-DomainGroupMember -Identity 'Domain Admins' -Members 'imposter.oatmeal'
PS C:\PowerSploit-master\Recon> whoami
breakfastland\imposter.oatmeal
```

Figure 47 - Adding imposter.oatmeal to Domain Admins

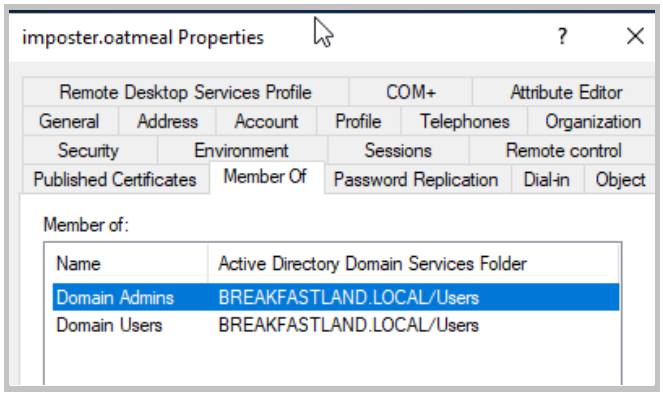


Figure 48 - Confirmation of Group Add

5.8.3 Building the Detections

5.8.3.1 Detection With Event IDs 5136 and 4662

Please note that for the following detection, you will need to adjust the following fields to account for the GUIDS/normalized names of the Groups you are looking to monitor:

- GUID
- Object_Name

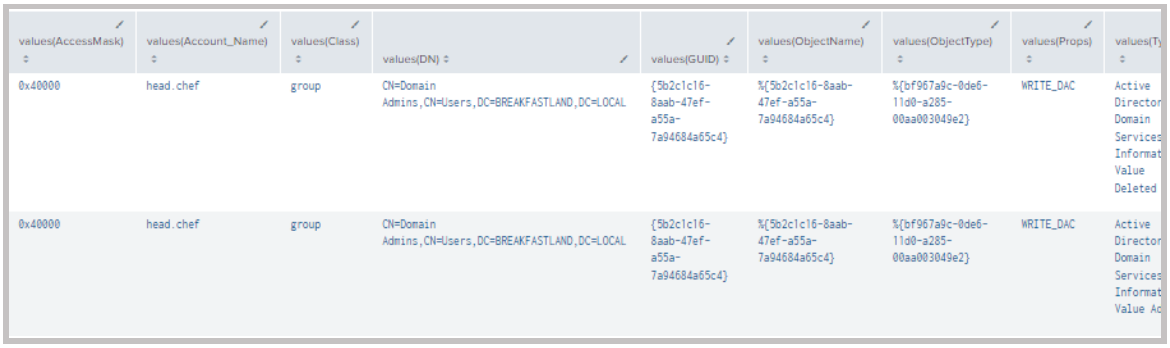
As an additional note, while the attack above only demonstrates adding “Generic Write” for our user object, this detection will also catch changes made to the other attributes listed above.

Finally, this detection is specifically identifying when changes to the Domain Admins group DACL are made, and not when the account is being added to the Domain Admins group.

```
((index=main
EventCode=5136   Class=group  LDAP_Display_Name=nTSecurityDescriptor
GUID="{5b2c1c16-8aab-47ef-a55a-7a94684a65c4}") OR (index=main
Account_Name!=*$ Object_Type="%{bf967a9c-0de6-11d0-a285-00aa003049e2}"
Object_Name="%{5b2c1c16-8aab-47ef-a55a-7a94684a65c4}"  EventCode=4662
Access_Mask = 0x40000))
| eval Logon_ID=if(EventCode==4662,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval user=if(EventCode==4662,mvindex( Account_Name,-1), mvindex(
Account_Name,-1))
| eval DACL=if(EventCode==5136,mvindex( Value,-1), mvindex( Value,-1))
| join type=outer Logon_ID
     [ search index=main Account_Name!=*$ Object_Type="%{bf967a9c-0de6-
11d0-a285-00aa003049e2}" Object_Name="%{5b2c1c16-8aab-47ef-a55a-
7a94684a65c4}"  EventCode=4662 Access_Mask = 0x40000
     | eval Props=Properties
     | eval AccessMask=Access_Mask
     | eval ObjectType=Object_Type
     | eval ObjectName=Object_Name
     |table Account_Name,Logon_ID,Props,AccessMask,ObjectType,
ObjectName]
| table  _time, Logon_ID, Account_Name, Props, AccessMask, ObjectType,
ObjectName,  DN, GUID, DACL, Class, Type
|stats values by _time, Logon_ID, DACL
```

_time	Logon_ID	DACL
2023-09-01 18:48:48	0x3C80C	O:DAG:DAD:AI(OA;;RP;46a9b11d-60ae-405a-b7e8-ff8a58d456d2;;S-1-5-32-560)(OA;;CR;ab721a55-1e2f-11d0-9819-00aa0040529b;;AU)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;DA)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;AO)(A;;LCRPLORC;;;PS)(A;;LCRPLORC;;;AU)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;SY)(OA;CIID;CR;89e95b76-444d-4c62-991a-0facbeda640c;;S-1-5-21-1865600711-3446354287-3882071624-1111)(OA;CIID;CR;1131f6aa-9c07-11d1-f79f-00c04fc2dcd2;;S-1-5-21-1865600711-3446354287-3882071624-1111)(OA;CIID;CR;1131f6ad-9c07-11d1-f79f-00c04fc2dcd2;;S-1-5-21-1865600711-3446354287-3882071624-1111)(OA;CIIOID;SW;9b026da6-0d3c-465c-8bee-5199d7165cba;bf967a86-0de6-11d0-a285-00aa003049e2;CO)(OA;CIIOID;SW;9b026da6-0d3c-465c-8bee-5199d7165cba;bf967a86-0de6-11d0-a285-00aa003049e2;PS)(OA;CIIOID;RP;b7c69e6d-2cc7-11d2-854e-00a0c983f608;bf967a9c-0de6-11d0-a285-00aa003049e2;ED)(OA;CIIOID;RP;b7c69e6d-2cc7-11d2-854e-00a0c983f608;bf967aba-0de6-11d0-a285-00aa003049e2;ED)(OA;CIIOID;WP;ea1b7b93-5e48-46d5-bc6c-4df4fda78a35;bf967a86-0de6-11d0-a285-00aa003049e2;PS)(OA;CIIOID;LCRPLORC;;4828cc14-1437-45bc-9b07-ad6f015e5f28;RU)(OA;CIID;LCRPLORC;;bf967a9c-0de6-11d0-a285-00aa003049e2;RU)(OA;CIIOID;LCRPLORC;;bf967aba-0de6-11d0-a285-00aa003049e2;RU)(OA;CIID;RPWP;3f78c3e5-f79a-46bd-a0b8-9d18116ddc79;;PS)(OA;CIID;RPWPCR;91e647de-d96f-4b70-9557-d63ff4f3ccd8;;PS)(A;CIID;LCRPWPRC;;;S-1-5-21-1865600711-3446354287-3882071624-1111)(A;CIID;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;S-1-5-21-1865600711-3446354287-3882071624-519)(A;CIID;LC;;;RU)(A;CIID;CCLCSWRPWPLOCRSDRCWDWO;;;BA)S:AI(AU;SA;WPWDWO;;;WD)(AU;CIIDSAFA;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;WD)(OU;CIIOIDSA;WP;f30e3bbe-9ff0-11d1-b603-0000f80367c1;bf967aa5-0de6-11d0-a285-00aa003049e2;WD)
2023-09-01 18:48:48	0x3C80C	O:DAG:DAD:AI(OA;;RP;46a9b11d-60ae-405a-b7e8-ff8a58d456d2;;S-1-5-32-560)(OA;;CR;ab721a55-1e2f-11d0-9819-00aa0040529b;;AU)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;DA)(A;CI;LCRPWPRC;;;S-1-5-21-1865600711-3446354287-3882071624-1112)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;AO)(A;;LCRPLORC;;;PS)(A;;LCRPLORC;;;AU)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;SY)(OA;CIID;CR;89e95b76-444d-4c62-991a-0facbeda640c;;S-1-5-21-1865600711-3446354287-3882071624-1111)(OA;CIID;CR;1131f6aa-9c07-11d1-f79f-00c04fc2dcd2;;S-1-5-21-1865600711-3446354287-3882071624-1111)(OA;CIID;CR;1131f6ad-9c07-11d1-f79f-00c04fc2dcd2;;S-1-5-21-1865600711-3446354287-3882071624-1111)(OA;CIIOID;SW;9b026da6-0d3c-465c-8bee-5199d7165cba;bf967a86-0de6-11d0-a285-00aa003049e2;CO)(OA;CIIOID;SW;9b026da6-0d3c-465c-8bee-5199d7165cba;bf967a86-0de6-11d0-a285-00aa003049e2;PS)(OA;CIIOID;RP;b7c69e6d-2cc7-11d2-854e-00a0c983f608;bf967a9c-0de6-11d0-a285-00aa003049e2;ED)(OA;CIIOID;RP;b7c69e6d-2cc7-11d2-854e-00a0c983f608;bf967aba-0de6-11d0-a285-00aa003049e2;ED)(OA;CIIOID;WP;ea1b7b93-5e48-46d5-bc6c-4df4fda78a35;bf967a86-0de6-11d0-a285-00aa003049e2;PS)(OA;CIIOID;LCRPLORC;;4828cc14-1437-45bc-9b07-ad6f015e5f28;RU)(OA;CIID;LCRPLORC;;bf967a9c-0de6-11d0-a285-00aa003049e2;RU)(OA;CIIOID;LCRPLORC;;bf967aba-0de6-11d0-a285-00aa003049e2;RU)(OA;CIID;RPWP;3f78c3e5-f79a-46bd-a0b8-9d18116ddc79;;PS)(OA;CIID;RPWPCR;91e647de-d96f-4b70-9557-d63ff4f3ccd8;;PS)(A;CIID;LCRPWPRC;;;S-1-5-21-1865600711-3446354287-3882071624-1111)(A;CIID;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;S-1-5-21-1865600711-3446354287-3882071624-519)(A;CIID;LC;;;RU)(A;CIID;CCLCSWRPWPLOCRSDRCWDWO;;;BA)S:AI(AU;SA;WPWDWO;;;WD)(AU;CIIDSAFA;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;WD)(OU;CIIOIDSA;WP;f30e3bbf-9ff0-11d1-b603-0000f80367c1;bf967aa5-0de6-11d0-a285-00aa003049e2;WD)

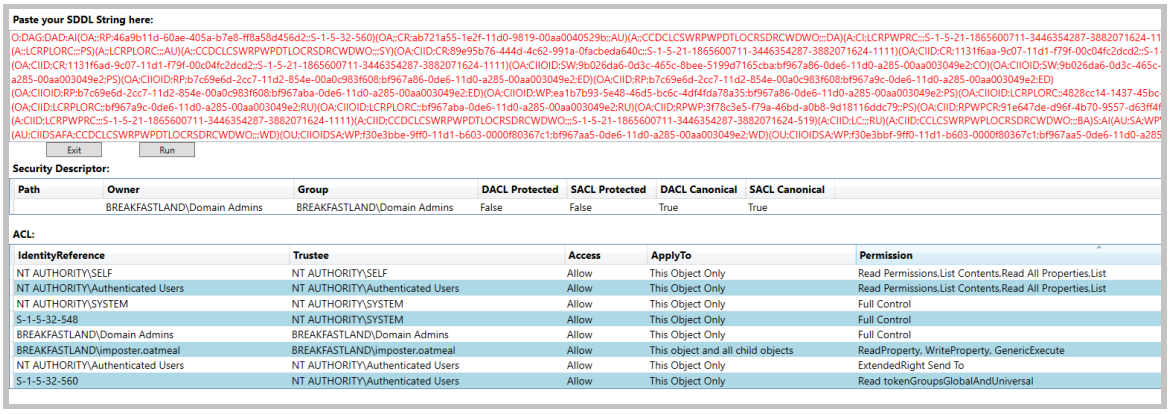
Figure 49 - AddMember Final Query (1)



values(AccessMask)	values(Account_Name)	values(Class)	values(DN)	values(GUID)	values(ObjectName)	values(ObjectType)	values(Props)	values(T
0x40000	head.chef	group	CN=Domain Admins,CN=Users,DC=BREAKFASTLAND,DC=LOCAL	{5b2c1c16-8aab-47ef-a55a-7a94684a65c4}	%{5b2c1c16-8aab-47ef-a55a-7a94684a65c4}	%{bf967a9c-0de6-11d0-a285-00aa003049e2}	WRITE_DAC	Active Director Domain Services Informat Value Deleted
0x40000	head.chef	group	CN=Domain Admins,CN=Users,DC=BREAKFASTLAND,DC=LOCAL	{5b2c1c16-8aab-47ef-a55a-7a94684a65c4}	%{5b2c1c16-8aab-47ef-a55a-7a94684a65c4}	%{bf967a9c-0de6-11d0-a285-00aa003049e2}	WRITE_DAC	Active Director Domain Services Informat Value Ac

Figure 50 - AddMember Final Query (2)

If we review the SDDL using SDDL-Converter, we can see the rights added for **imposter.oatmeal** to the Domain Admins DACL.



Paste your SDDL String here:

O:DAG:DAD:AI(OA;;RP;46a9b11d-60ae-405a-b7e8-ff8a58d456d2;;S-1-5-32-560)(OA;;CR;ab721a55-1e2f-11d0-9819-00aa0040529b;;AU)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;DA)(A;CI;LCRPWPRC;;;S-1-5-21-1865600711-3446354287-3882071624-11
(A;;LCRPLORC;;;PS)(A;;LCRPLORC;;;AU)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;SY)(OA;CIID;CR;89e95b76-444d-4c62-991a-0facbeda640c;;S-1-5-21-1865600711-3446354287-3882071624-1111)(OA;CIID;CR;1131f6aa-9c07-11d1-f79f-00c04fc2dcd2;;S-1-
(OA;CIID;CR;1131f6ad-9c07-11d1-f79f-00c04fc2dcd2;;S-1-5-21-1865600711-3446354287-3882071624-1111)(OA;CIIOID;SW;9b026da6-0d3c-465c-8bee-5199d7165cba;bf967a86-0de6-11d0-a285-00aa003049e2;CO)(OA;CIIOID;SW;9b026da6-0d3c-465c-
a285-00aa003049e2;PS)(OA;CIIOID;RP;b7c69e6d-2cc7-11d2-854e-00a0c983f608;bf967a86-0de6-11d0-a285-00aa003049e2;ED)(OA;CIID;RP;b7c69e6d-2cc7-11d2-854e-00a0c983f608;bf967a9c-0de6-11d0-a285-00aa003049e2;ED)
(OA;CIIOID;RP;b7c69e6d-2cc7-11d2-854e-00a0c983f608;bf967aba-0de6-11d0-a285-00aa003049e2;ED)(OA;CIIOID;WP;ea1b7b93-5e48-46d5-bc6c-4df4fda78a35;bf967a86-0de6-11d0-a285-00aa003049e2;PS)(OA;CIIOID;LCRPLORC;;4828cc14-1437-45bc-
(OA;CIID;LCRPLORC;;bf967a9c-0de6-11d0-a285-00aa003049e2;RU)(OA;CIIOID;LCRPLORC;;bf967aba-0de6-11d0-a285-00aa003049e2;RU)(OA;CIID;RPWP;3f78c3e5-f79a-46bd-a0b8-9d18116ddc79;;PS)(OA;CIID;RPWPCR;91e647de-d96f-4b70-9557-d63ff4f
(A;CIID;LCRPWPRC;;;S-1-5-21-1865600711-3446354287-3882071624-1111)(A;CIID;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;S-1-5-21-1865600711-3446354287-3882071624-519)(A;CIID;LC;;;RU)(A;CIID;CCLCSWRPWPLOCRSDRCWDWO;;;BA)S:AI(AU;SA;WP
(AU;CIIDSAFA;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;WD)(OU;CIIOIDSA;WP;f30e3bbe-9ff0-11d1-b603-0000f80367c1;bf967aa5-0de6-11d0-a285-00aa003049e2;WD)(OU;CIIOIDSA;WP;f30e3bbf-9ff0-11d1-b603-0000f80367c1;bf967aa5-0de6-11d0-a285

Exit Run

Security Descriptor:

Path	Owner	Group	DACL Protected	SACL Protected	DACL Canonical	SACL Canonical
	BREAKFASTLAND\Domain Admins	BREAKFASTLAND\Domain Admins	False	False	True	True

ACL:

IdentityReference	Trustee	Access	ApplyTo	Permission
NT AUTHORITY\SELF	NT AUTHORITY\SELF	Allow	This Object Only	Read Permissions,List Contents,Read All Properties,List
NT AUTHORITY\Authenticated Users	NT AUTHORITY\Authenticated Users	Allow	This Object Only	Read Permissions,List Contents,Read All Properties,List
NT AUTHORITY\SYSTEM	NT AUTHORITY\SYSTEM	Allow	This Object Only	Full Control
S-1-5-32-548	NT AUTHORITY\SYSTEM	Allow	This Object Only	Full Control
BREAKFASTLAND\Domain Admins	BREAKFASTLAND\Domain Admins	Allow	This Object Only	Full Control
BREAKFASTLAND\imposter.oatmeal	BREAKFASTLAND\imposter.oatmeal	Allow	This object and all child objects	ReadProperty, WriteProperty, GenericExecute
NT AUTHORITY\Authenticated Users	NT AUTHORITY\Authenticated Users	Allow	This Object Only	ExtendedRight Send To
S-1-5-32-560	NT AUTHORITY\Authenticated Users	Allow	This Object Only	Read tokenGroupsGlobalAndUniversal

Figure 51 - SDDL Review

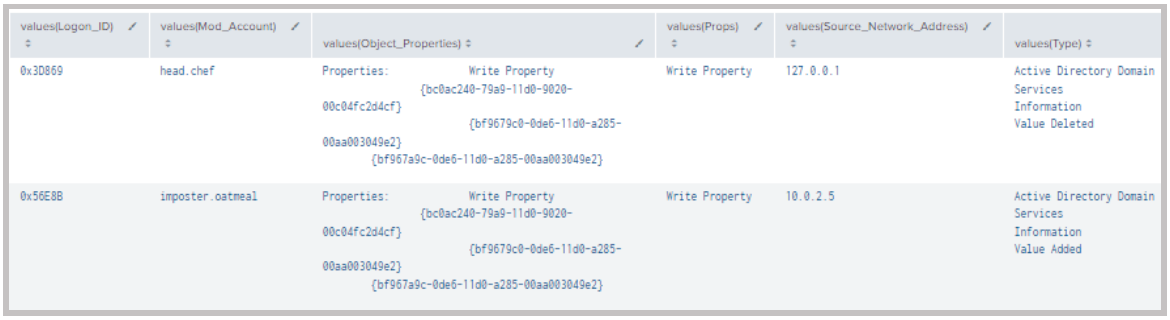
To detect the **imposter.oatmeal** account being added to the Domain Admins group, we will use the Member attribute as the LDAP_Display_Name.

5.8.3.2 Detection With Event IDs 5136, 4662, and 4624

```
index=main ((EventCode=5136 AND LDAP_Display_Name=member)  OR
(EventCode=4624 AND Account_Name!="*$" AND Account_Name!="ANONYMOUS
LOGON" AND Account_Name!="SYSTEM") OR (EventCode=4662 AND
Access_Mask=0x20))
| eval Logon_ID=if(EventCode==4624,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval Mod_Account=if(EventCode==4624,mvindex(Account_Name,-1),
mvindex(Account_Name,-1))
| eval Changed_Value=if(EventCode==5136,mvindex(Value,-1),
mvindex(Value,-1))
| join type=outer Logon_ID
        [ search (EventCode=5136) OR (EventCode=4624)
        | stats count by Logon_ID, Account_Name, Source_Network_Address
        | table Account_Name,Logon_ID, Source_Network_Address ]
| join type=outer Logon_ID
    [ search index=main Account_Name!=*$  EventCode=4662 Access_Mask =
0x20
    | eval Props=Properties
    | eval AccessMask=Access_Mask
    | eval ObjectType=Object_Type
    | eval ObjectName=Object_Name
    | rex field=Message
"(?<Object_Properties>(?ms)(?<=)Properties:(.*?)(?=Additional\s+))"
    |table Account_Name,Logon_ID,Props,AccessMask,ObjectType, ObjectName,
Object_Properties]
| table _time, Mod_Account, Source_Network_Address , Class, DN, Logon_ID,
Type, LDAP_Display_Name, Changed_Value, AccessMask, Props,
Object_Properties
| where  len(Class)>0
| stats values by _time, Changed_Value
```

_time	Changed_Value	values(AccessMask)	values(Class)	values(DN)	values(LDAP_Display_Name)
2023-09-01 19:26:01	CN=imposter.oatmeal,CN=Users,DC=BREAKFASTLAND,DC=LOCAL	0x20	group	CN=Domain Admins,CN=Users,DC=BREAKFASTLAND,DC=LOCAL	member
2023-09-01 19:26:11	CN=imposter.oatmeal,CN=Users,DC=BREAKFASTLAND,DC=LOCAL	0x20	group	CN=Domain Admins,CN=Users,DC=BREAKFASTLAND,DC=LOCAL	member

Figure 52 - Detecting Addition of Account to Domain Admins Group (1)



values(Logon_ID)	values(Mod_Account)	values(Object_Properties)	values(Props)	values(Source_Network_Address)	values(Type)
0x3D869	head.chef	Properties: Write Property {bc0ac240-79a9-11d0-9020-00c04fc2d4cf} {bf9679c0-0de6-11d0-a285-00aa003049e2} {bf967a9c-0de6-11d0-a285-00aa003049e2}	Write Property	127.0.0.1	Active Directory Domain Services Information Value Deleted
0x56E8B	imposter.oatmeal	Properties: Write Property {bc0ac240-79a9-11d0-9020-00c04fc2d4cf} {bf9679c0-0de6-11d0-a285-00aa003049e2} {bf967a9c-0de6-11d0-a285-00aa003049e2}	Write Property	10.0.2.5	Active Directory Domain Services Information Value Added

Figure 53 - Detecting Addition of Account to Domain Admins Group (2)

5.9 ForceChangePassword

5.9.1 Background

ForceChangePassword is the section referenced within the Hacker Recipes that defines a collection of permissions that allow for control over changing the password of another user/object. More specifically, the permissions referenced are:

- GenericAll
- AllExtendedRights
- User-Force-Change-Password

For this particular section, we will focus on the "User-Force-Change-Password" right. However, the detections built should detect the use of all three (3) rights.

5.9.2 Modifying the Objects

Note: For this section, you will need to enable auditing on the specific user/computer object you are looking to monitor.

Below, you can see that in addition to the default privileges that are generated when you create a regular user account in AD, I have added **imposter.oatmeal** as a grantee on an ACE, checked "Reset Password," and then applied those changes to the user.

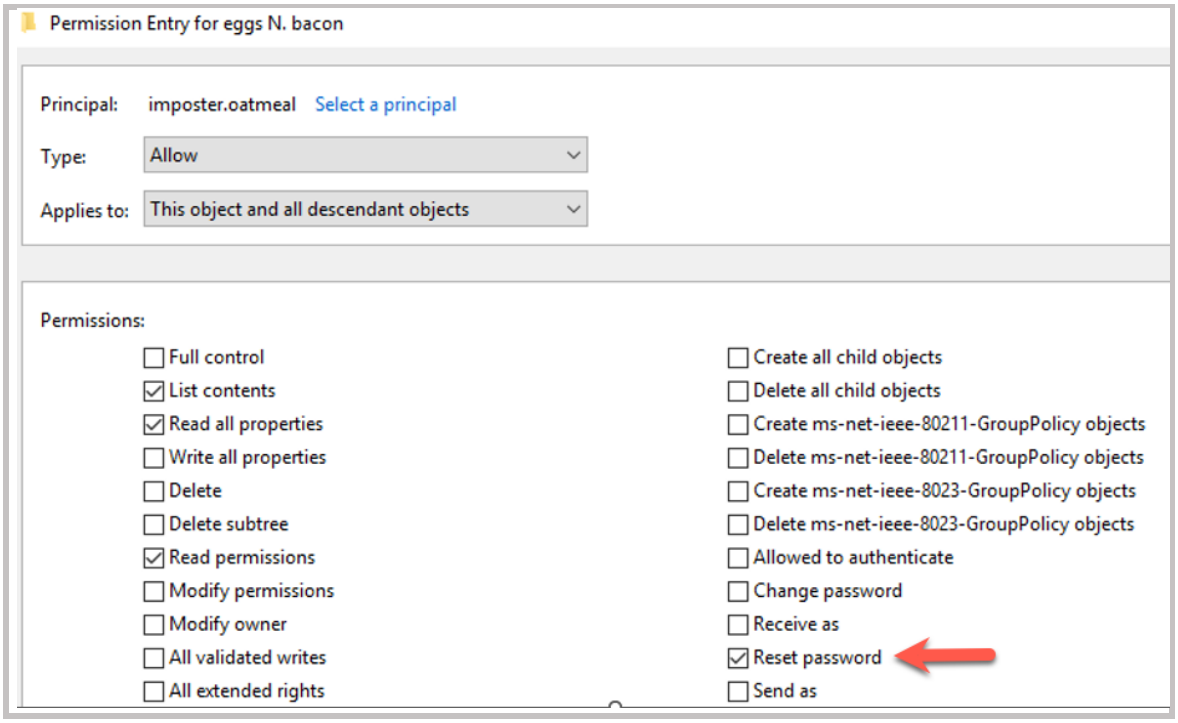


Figure 54 - Adding Privileges to Account

Now we can utilize ***PowerSploit*** to change the password of our victim account.

```
PS C:\PowerSploit-master\Recon>
PS C:\PowerSploit-master\Recon> $NewPassword = ConvertTo-SecureString 'IamSo3ecure!' -AsPlainText -Force
PS C:\PowerSploit-master\Recon> Set-DomainUserPassword -Identity 'eggs.bacon' -AccountPassword $NewPassword
PS C:\PowerSploit-master\Recon> whoami
breakfastland\imposter.oatmeal
PS C:\PowerSploit-master\Recon>
```

Figure 55 - Changing Account Password with PowerSploit

```
┌──(root㉿kali)-[/home/tools/GPOwned]
└─# crackmapexec smb 10.0.2.4 -u eggs.bacon -p 'IamSo3ecure!'
SMB         10.0.2.4        445    BREAKFAST-DC-01  [*] Windows 10.0 Build 17763 x64 (name:BREAKFAST-DC-01) (domain:BREAKFASTLAND.LOCAL)
SMB         10.0.2.4        445    BREAKFAST-DC-01  [+] BREAKFASTLAND.LOCAL\eggs.bacon:IamSo3ecure!
```

Figure 56 - Validating Password Change With CrackMapExec

5.9.3 Building the Detections

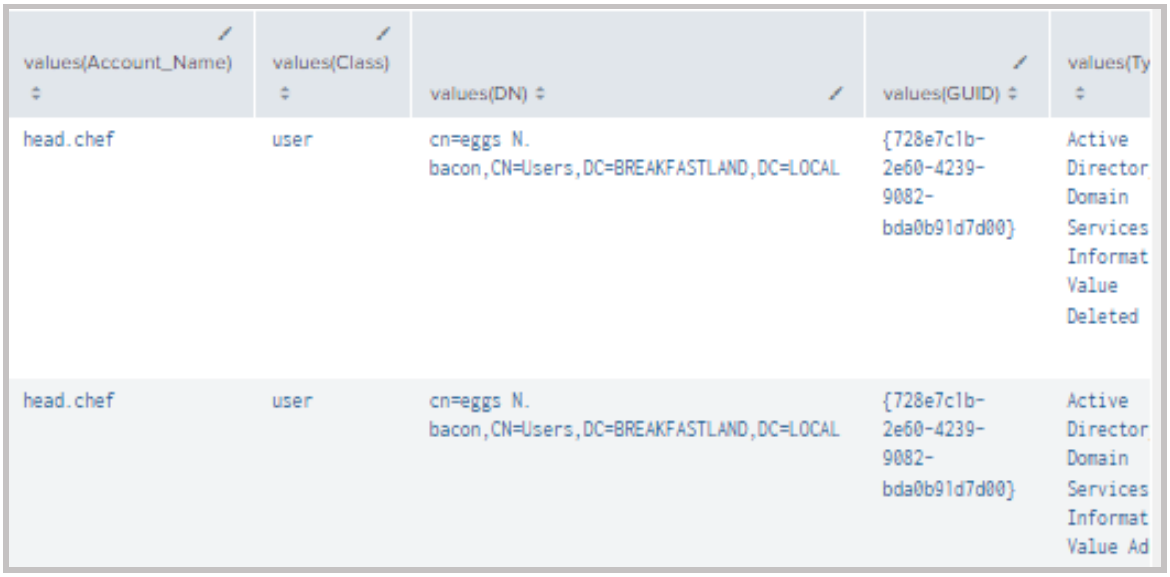
5.9.3.1 Detection With Event IDs 5136 and 4662

This detection is nearly identical to the one that was built for the ***AddMember*** section; the only modifications you will need to make are to ***Object_Type***, which reflects the "User" object within your AD domain. Remember that this is a lab environment and not an organizational production environment; you may wish to use the ***Object_Name*** filter to either exclude or to build a list of users to monitor if logging volume is high for this particular detection.

As was done with the ***AddMember*** section previously, the following detection focuses on adding the “ForceChangePassword” right to the user object’s **egg.bacon’s** DACL.

```
 ((index=main
EventCode=5136   Class=user  LDAP_Display_Name=nTSecurityDescriptor) OR
(index=main Account_Name!=*$ Object_Type="%{bf967aba-0de6-11d0-a285-
00aa003049e2}" EventCode=4662 Access_Mask = 0x40000))
| eval Logon_ID=if(EventCode==4662,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval user=if(EventCode==4662,mvindex( Account_Name,-1), mvindex(
Account_Name,-1))
| eval DACL=if(EventCode==5136,mvindex( Value,-1), mvindex( Value,-1))
| join type=outer Logon_ID
    [ search index=main Account_Name!=*$ Object_Type="%bf967aba-0de6-
11d0-a285-00aa003049e2}}" EventCode=4662 Access_Mask = 0x40000
    | eval Props=Properties
    | eval AccessMask=Access_Mask
    | eval ObjectType=Object_Type
    | eval ObjectName=Object_Name
    |table Account_Name,Logon_ID,Props,AccessMask,ObjectType,
ObjectName]
| table  _time, Logon_ID, Account_Name, Props, AccessMask, ObjectType,
ObjectName,  DN, GUID, DACL, Class, Type
|stats values by _time, Logon_ID,DACL
```

Figure 57 - Final Query Detecting ForceChangePassword Modification (1)



values(Account_Name)	values(Class)	values(DN)	values(GUID)	values(Ty
head.chef	user	cn=eggs N. bacon,CN=Users,DC=BREAKFASTLAND,DC=LOCAL	{728e7c1b-2e60-4239-9082-bda0b91d7d00}	Active Director Domain Services Informat Value Deleted
head.chef	user	cn=eggs N. bacon,CN=Users,DC=BREAKFASTLAND,DC=LOCAL	{728e7c1b-2e60-4239-9082-bda0b91d7d00}	Active Director Domain Services Informat Value Ad

Figure 58 - Final Query Detecting ForceChangePassword Modification (2)

5.10 GrantOwnerShip

5.10.1 Background

GrantOwnerShip (or Generic Write on User) is the section referenced within the Hacker Recipes that defines a collection of permissions that allows for more general control of another user/object. More specifically, the permissions referenced are:

- GenericAll
- WriteOwner

For this section, we will be focusing on the WriteOwner permission. However, the detections built should detect the use of both rights.

5.10.2 Modifying the Objects (Attack)

First, we begin by "misconfiguring" our **dacled.egg** object to give **imposter.oatmeal**, our attacker account, WriteOwner permissions.

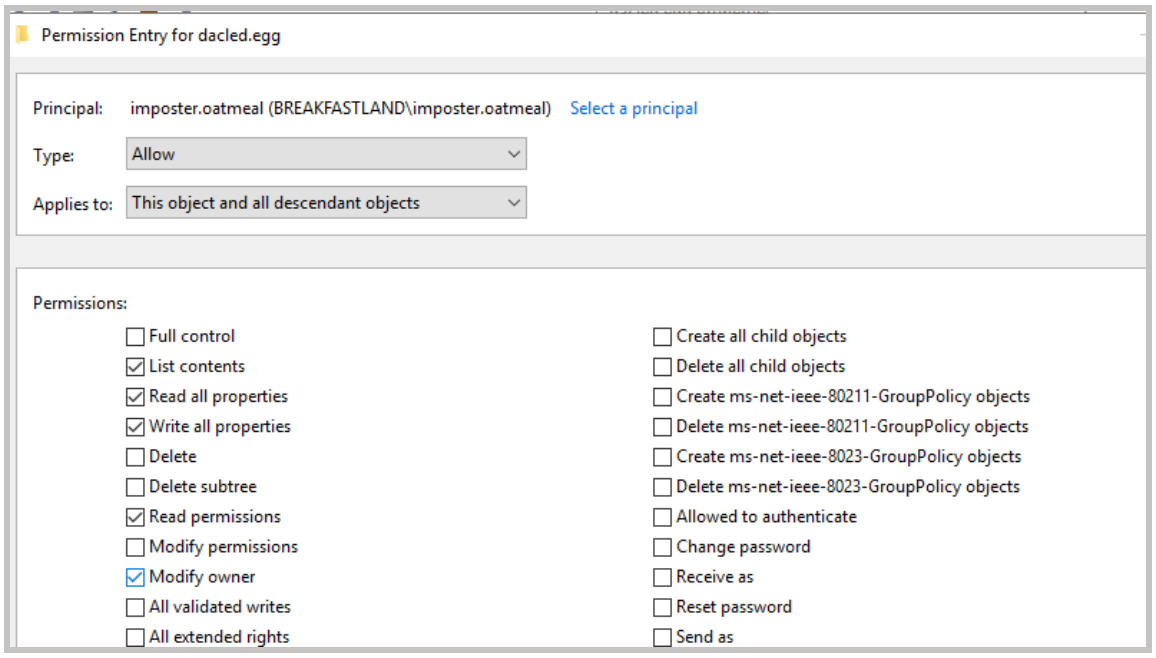


Figure 59 - Modifying User Account for WriteOwner

Then we use ***PowerView*** to set ownership.

```
PS C:\PowerSploit-master\Recon> Set-DomainObjectOwner -Identity 'dacled.egg' -OwnerIdentity 'imposter.oatmeal'
PS C:\PowerSploit-master\Recon>
```

Figure 60 - Setting Ownership with PowerView

5.10.3 Building the Detections

The same detection we built to detect the modifications to user objects for ***ForceChangePassword*** will also pick up changes made to the user object for ***WriteOwner***.

Figure 61 - Detecting Misconfiguration of User Object (1)

values(Account_Name)	values(Class)	values(DN)	values(GUID)	values(Ty
head.chef	user	CN=dacled.egg,CN=Users,DC=BREAKFASTLAND,DC=LOCAL	{d7f6993b-3688-46dc-b64c-c309bce0ada9}	Active Director Domain Services Informat Value Ad Value Deleted
head.chef	user	CN=dacled.egg,CN=Users,DC=BREAKFASTLAND,DC=LOCAL	{d7f6993b-3688-46dc-b64c-c309bce0ada9}	Active Director Domain Services Informat Value Ad Value Deleted

Figure 62 - Detecting Misconfiguration of User Object (2)

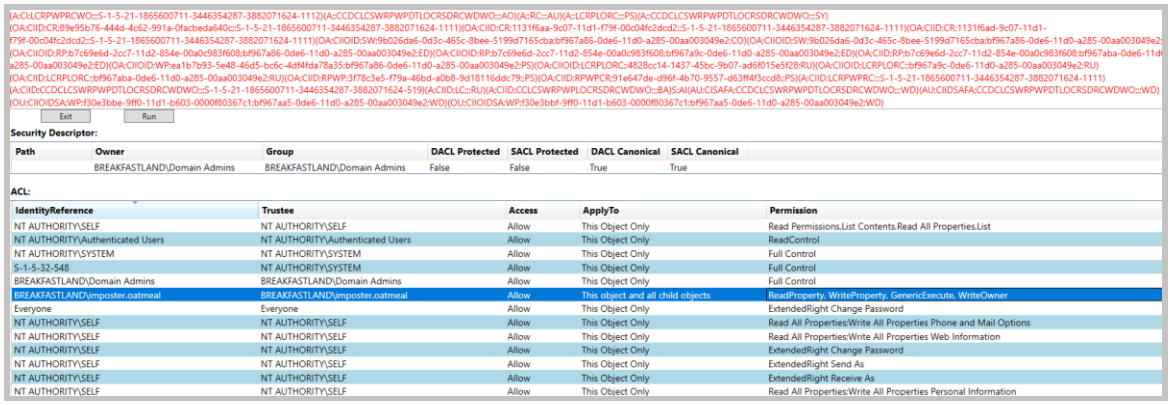


Figure 63 - SDDL Detecting WriteOwner Privileges on dacled.egg

It will also identify when the owner of the object has been changed with the PowerShell cmdlet.

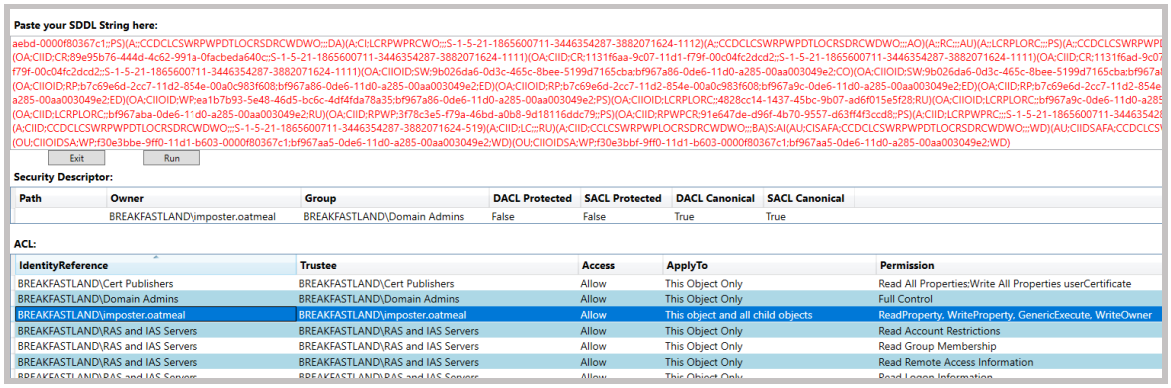


Figure 64 -SDDL Post Ownership Change

5.11 LAPS and gMSA

The following sections will involve two (2) attributes specific to both gMSA and LAPS. For background context and detections, please refer to Andrew's post on gMSA and Megan's post on Local Administrator Password Solution (LAPS).

5.11.1 ms-Mcs-AdmPwdExpirationTime (LAPS):

5.11.2 Background

The ms-Mcs-AdmPwdExpirationTime attribute determines when a LAPS password is due to expire and when it will, by proxy, refresh the local administrative password for the machine.

5.11.3 Modifying the Attribute

Before we can modify the attribute, we must first convert the time of the new password expiration that's stored inside the given AD attribute in Windows File time format.

The easiest way to do this is to simply use the PowerShell AD module to first query the computer and object whose expiration time you're trying to identify, and then use PowerShell to convert the time stamp to human-readable format.

```
Get-ADComputer PAN-PC -Properties ms-Mcs-AdmPwdExpirationTime
```

```
[datetime]::FromFileTimeUTC(133308023294324030)
```

```
PS C:\Powermad-master> Get-ADComputer PAN-PC -Properties ms-MCS-AdmPwdExpirationTime


DistinguishedName           : CN=PAN-PC,OU=KitchenDevices,DC=BREAKFASTLAND,DC=LOCAL
DNSHostName                 : PAN-PC.BREAKFASTLAND.LOCAL
Enabled                     : True
ms-MCS-AdmPwdExpirationTime : 133308023294324030
Name                        : PAN-PC
ObjectClass                 : computer
ObjectGUID                  : d7dc8b34-e8c8-4302-906a-40d3046cecee
SamAccountName              : PAN-PC$
SID                         : S-1-5-21-1865600711-3446354287-3882071624-1105
UserPrincipalName           :

PS C:\Powermad-master> [datetime]::FromFileTimeUTC(133308023294324030)

Friday, June 9, 2023 4:38:49 PM
```

Figure 65 - Reading ms-MCS-AdmPwdExpirationTime Attribute

When changing the expiration time using PowerMad, your specified value will need to be submitted in the Windows File time format.

```
PS C:\Powermad-master>  Set-MachineAccountAttribute -Attribute ms-Mcs-AdmPwdExpirationTime -Value '133308023586380806'
cmdlet Set-MachineAccountAttribute at command pipeline position 1
Supply values for the following parameters:
MachineAccount: IMPOSTER-GRANOLA
[+] Machine account IMPOSTER-GRANOLA attribute ms-Mcs-AdmPwdExpirationTime updated
```

Figure 66 - Modifying Attribute with PowerMad

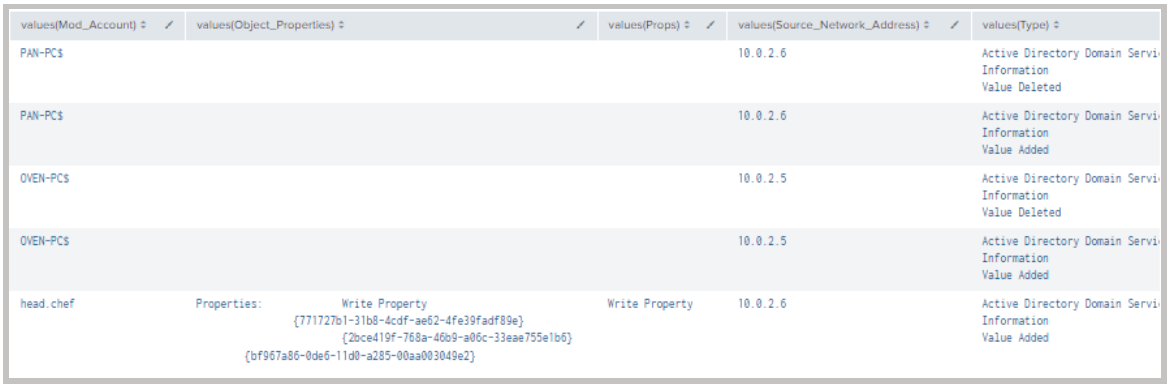
5.11.4 Building the Detections

5.11.4.1 Detection With Event IDs 5136, 4662, and 4624

```
index=main ((EventCode=5136 AND LDAP_Display_Name=ms-Mcs-
AdmPwdExpirationTime)  OR (EventCode=4624 AND Account_Name!="*$" AND
Account_Name!="ANONYMOUS LOGON" AND Account_Name!="SYSTEM") OR
(EventCode=4662 AND Access_Mask=0x20))
| eval Logon_ID=if(EventCode==4624,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval Mod_Account=if(EventCode==4624,mvindex(Account_Name,-1),
mvindex(Account_Name,-1))
| join type=outer Logon_ID
        [ search (EventCode=5136) OR (EventCode=4624)
        | stats count by Logon_ID, Account_Name, Source_Network_Address
        | table Account_Name,Logon_ID, Source_Network_Address ]
| join type=outer Logon_ID
    [ search index=main Account_Name!=*$  EventCode=4662 Access_Mask =
0x20
    | eval Props=Properties
    | eval AccessMask=Access_Mask
    | eval ObjectType=Object_Type
    | eval ObjectName=Object_Name
    | rex field=Message
"(?<Object_Properties>(?ms)(?<=)Properties:(.*?)(?=Additional\s+))"
    |table Account_Name,Logon_ID,Props,AccessMask,ObjectType, ObjectName,
Object_Properties]
| table _time, Mod_Account, Source_Network_Address , Class, DN, Logon_ID,
Type, LDAP_Display_Name, Value, AccessMask, Props, Object_Properties
| where  len(Class)>0
| stats values by _time, Value, Logon_ID
```

_time	Value	Logon_ID	values(AccessMask)	values(Class)	values(DN)	values(LDAP_Display_Name)
2023-05-30 09:38:56	133272013066754459	0x37D55		computer	CN=PAN-PC,OU=KitchenDevices,DC=BREAKFASTLAND,DC=LOCAL	ms-Mcs-AdmPwdExpirationTime
2023-05-30 09:38:56	133308023294324030	0x37D55		computer	CN=PAN-PC,OU=KitchenDevices,DC=BREAKFASTLAND,DC=LOCAL	ms-Mcs-AdmPwdExpirationTime
2023-05-30 09:39:13	133277112539474394	0x3C432		computer	CN=OVEN-PC,OU=KitchenDevices,DC=BREAKFASTLAND,DC=LOCAL	ms-Mcs-AdmPwdExpirationTime
2023-05-30 09:39:13	133308023586380806	0x3C432		computer	CN=OVEN-PC,OU=KitchenDevices,DC=BREAKFASTLAND,DC=LOCAL	ms-Mcs-AdmPwdExpirationTime
2023-06-26 15:17:01	133308023586380806	0x203869	0x20	computer	CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL	ms-Mcs-AdmPwdExpirationTime

Figure 67 - Detection With Event IDs 5136, 4624, 4662 (1)



values(Mod_Account)	values(Object_Properties)	values(Props)	values(Source_Network_Address)	values(Type)
PAN-PC$			10.0.2.6	Active Directory Domain Servi Information Value Deleted
PAN-PC$			10.0.2.6	Active Directory Domain Servi Information Value Added
OVEN-PC$			10.0.2.5	Active Directory Domain Servi Information Value Deleted
OVEN-PC$			10.0.2.5	Active Directory Domain Servi Information Value Added
head.chef	Properties: Write Property {771727b1-31b8-4cdf-ae62-4fe39fadf89e} {2bce419f-768a-46b9-a06c-33eae755e1b6} {bf967a86-0de6-11d0-a285-00aa003049e2}	Write Property	10.0.2.6	Active Directory Domain Servi Information Value Added

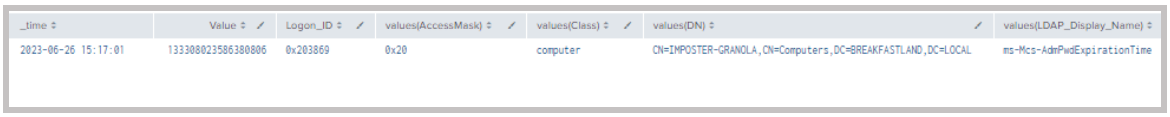
Figure 68 - Detection With Event IDs 5136, 4624, 4662 (2)

5.11.4.2 Detection With Event IDs 5136, 4662, and 4624, Excluding Machine Accounts

It is important to note that the previous detection will also catch legitimate changes to LAPS passwords. To filter out the legitimate events, we can remove the machine accounts.

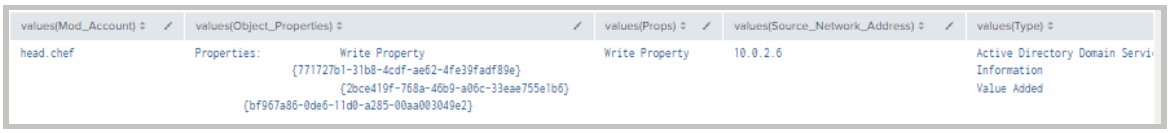
That said, it is advisable to keep both queries to ensure that detection is not evaded if an attacker creates or compromises a machine account.

```
index=main ((EventCode=5136 AND LDAP_Display_Name=ms-Mcs-
AdmPwdExpirationTime)  OR (EventCode=4624 AND Account_Name!="*$" AND
Account_Name!="ANONYMOUS LOGON" AND Account_Name!="SYSTEM") OR
(EventCode=4662 AND Access_Mask=0x20))
| eval Logon_ID=if(EventCode==4624,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval Mod_Account=if(EventCode==4624,mvindex(Account_Name,-1),
mvindex(Account_Name,-1))
| join type=outer Logon_ID
        [ search (EventCode=5136) OR (EventCode=4624)
        | stats count by Logon_ID, Account_Name, Source_Network_Address
        | table Account_Name,Logon_ID, Source_Network_Address ]
| join type=outer Logon_ID
    [ search index=main Account_Name!=*$  EventCode=4662 Access_Mask =
0x20
    | eval Props=Properties
    | eval AccessMask=Access_Mask
    | eval ObjectType=Object_Type
    | eval ObjectName=Object_Name
    | rex field=Message
"(?<Object_Properties>(?ms)(?<=)Properties:(.*?)(?=Additional\s+))"
    |table Account_Name,Logon_ID,Props,AccessMask,ObjectType, ObjectName,
Object_Properties]
| search Mod_Account!=*$
| table _time, Mod_Account, Source_Network_Address , Class, DN, Logon_ID,
Type, LDAP_Display_Name, Value, AccessMask, Props, Object_Properties
| where  len(Class)>0
| stats values by _time, Value, Logon_ID
```



_time	Value	Logon_ID	values(AccessMask)	values(Class)	values(DN)	values(LDAP_Display_Name)
2023-06-26 15:17:01	133308023586380806	0x203869	0x20	computer	CN=IMPOSTER-GRANOLA,CN=Computers,DC=BREAKFASTLAND,DC=LOCAL	ms-Mcs-AdmPwdExpirationTime

Figure 69 - Detection Excluding Machine Accounts (1)



values(Mod_Account)	values(Object_Properties)	values(Props)	values(Source_Network_Address)	values(Type)
head.chef	Properties: Write Property {771727b1-31b8-4cdf-ae62-4fe39fadf89e} {2bce419f-768a-46b9-a06c-33eae755e1b6} {bf967a86-0de6-11d0-a285-00aa003049e2}	Write Property	10.0.2.6	Active Directory Domain Servi Information Value Added

Figure 70 - Detection Excluding Machine Accounts (2)

5.12 PrincipalsAllowedToRetrieveManagedPassword/ms-DS-GroupMSAMembership (gMSA)

5.12.1 Background

The ***PrincipalsAllowedToRetrieveManagedPassword*** AD attribute governs who can read the gMSA Password.

However, it is important to outline that the ***"PrincipalsAllowedToRetrieveManagedPassword"*** value isn't the attribute that we should be auditing, since this value is only used when making changes within PowerShell.

Instead, the change made with the ***"PrincipalsAllowedToRetrieveManagedPassword"*** flag stores the value assigned in the PowerShell cmdlet to the ***ms-DS-GroupMSAMembership*** attribute within AD. This interpretation is also backed up by [this] article.

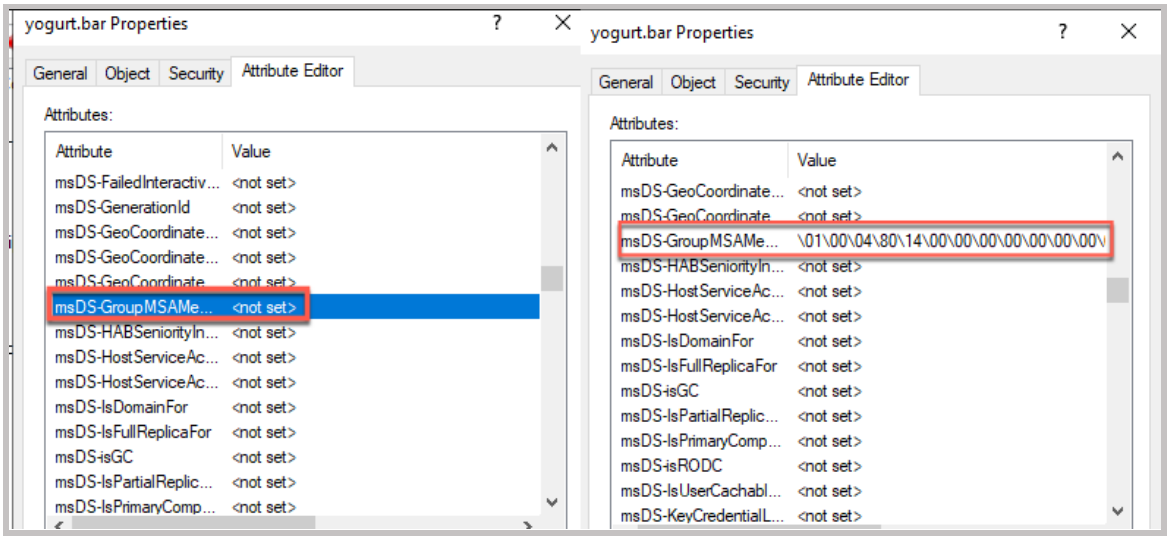


Figure 71 - Pre- and Post-Change Values of msDS-GroupMSAMEmbership attribute

An attacker who can insert themselves into this attribute then has the ability to read the gMSA account password itself.

5.12.2 Modifying the Objects (Attack)

Using the Active Directory PowerShell module, we first will use the *"PrincipalsAllowedToRetrieveManagedPassword"* flag to give the created gMSA account **granola.bar** permissions over the Domain Admins group.

```
PS C:\Powermad-master> Set-ADServiceAccount -Identity granola.bar -PrincipalsAllowedToRetrieveManagedPassword "Domain Admins"
```

Figure 72 - Modifying the gMSA Account

```
Get-ADServiceAccount -Identity granola.bar -Properties *
```

```
Name                                  : granola.bar
nTSecurityDescriptor                  : System.DirectoryServices.ActiveDirectorySecurity
ObjectCategory                        : CN=ms-DS-Group-Managed-Service-Account,CN=Schema,CN=Configuration,DC=BREAKFASTLAND,DC=LOCAL
ObjectClass                           : msDS-GroupManagedServiceAccount
ObjectGUID                            : 4489d276-5210-4540-b5a5-39ac29f63ace
objectSid                             : S-1-5-21-1865600711-3446354287-3882071624-1116
PasswordExpired                       : False
PasswordLastSet                       : 6/2/2023 3:58:05 PM
PasswordNeverExpires                  : False
PasswordNotRequired                   : False
PrimaryGroup                          : CN=Domain Computers,CN=Users,DC=BREAKFASTLAND,DC=LOCAL
primaryGroupID                        : 515
PrincipalsAllowedToDelegateToAccount  : {}
PrincipalsAllowedToRetrieveManagedPassword : {CN=Domain Admins,CN=Users,DC=BREAKFASTLAND,DC=LOCAL}
ProtectedFromAccidentalDeletion       : False
pwdLastSet                            : 133302202853750997
SamAccountName                        : granola.bar$
sAMAccountType                        : 805306369
sDRightsEffective                     : 15
ServicePrincipalNames                 :
SID                                   : S-1-5-21-1865600711-3446354287-3882071624-1116
SIDHistory                            : {}
```

Figure 73 - Validating Changes to gMSA Account

5.12.3 Building the Detections

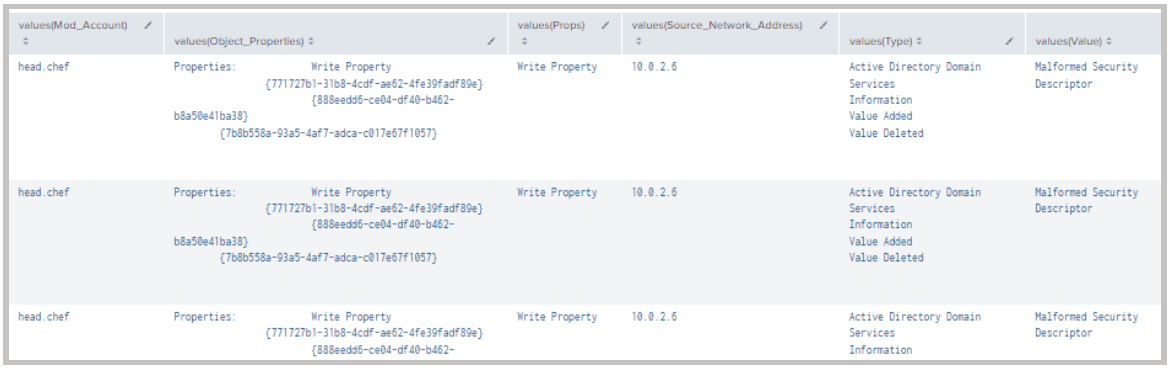
5.12.3.1 Detection with Event IDs 5136, 4662, and 4624

```
index=main ((EventCode=5136 AND LDAP_Display_Name=msDS-
GroupMSAMembership)  OR (EventCode=4624 AND Account_Name!="*$" AND
Account_Name!="ANONYMOUS LOGON" AND Account_Name!="SYSTEM" AND
Source_Network_Address!="-") OR (EventCode=4662 AND Access_Mask=0x20))
| eval Mod_Account=if(EventCode==4624,mvindex(Account_Name,-1),
mvindex(Account_Name,-1))
| eval LogonID=mvindex(Logon_ID,0)
| eval LogonID_2=mvindex(Logon_ID,1)
| join type=outer Logon_ID
  [ search (EventCode=5136 AND LDAP_Display_Name=msDS-GroupMSAMembership)
        | table Logon_ID, ]
| join type=outer LogonID_2
  [ search (EventCode=4624 AND Account_Name!="*$" AND
Account_Name!="ANONYMOUS LOGON" AND Account_Name!="SYSTEM" AND
Source_Network_Address!="-")
  | table Account_Name,LogonID_2, Source_Network_Address ]
| join type=outer Logon_ID
    [ search index=main Account_Name!=*$  EventCode=4662 Access_Mask =
0x20
    | eval Props=Properties
    | eval AccessMask=Access_Mask
```

```
    | eval ObjectType=Object_Type
    | eval ObjectName=Object_Name
      | rex field=Message
"(?<Object_Properties>(?ms)(?<=)Properties:(.*?)(?=Additional\s+))"
    |table Account_Name,Logon_ID,Props,AccessMask,ObjectType, ObjectName,
Object_Properties]
| search Mod_Account!=*$
| table _time, Logon_ID, Mod_Account, Source_Network_Address , Class, DN,
Type, LDAP_Display_Name, Value, AccessMask, Props, Object_Properties
| where len(Class)>0
| stats values by _time, Logon_ID, DN
```

_time	Logon_ID	DN	values(AccessMask)	values(Class)	values(LDAP_Display_Name)
2023-07-14 15:06:46	0x76E8F	CN=granola.bar,CN=Managed Service Accounts,DC=BREAKFASTLAND,DC=LOCAL	0x20	msDS-GroupManagedServiceAccount	msDS-GroupMSAMembership
2023-07-14 15:12:36	0x76E8F	CN=granola.bar,CN=Managed Service Accounts,DC=BREAKFASTLAND,DC=LOCAL	0x20	msDS-GroupManagedServiceAccount	msDS-GroupMSAMembership
2023-07-14 15:17:11	0x76E8F	CN=testGMSA,CN=Managed Service Accounts,DC=BREAKFASTLAND,DC=LOCAL	0x20	msDS-GroupManagedServiceAccount	msDS-GroupMSAMembership

Figure 74 - Detection With Event IDs 4662, 5136 and 4624 (1)



values(Mod_Account)	values(Object_Properties)	values(Props)	values(Source_Network_Address)	values(Type)	values(Value)
head.chef	Properties: Write Property {771727b1-31b8-4cdf-ae62-4fe39fadf89e} {888eedd6-ce04-df40-b462-b8a50e41ba38} {7b8b558a-93a5-4af7-adca-c017e67f1057}	Write Property	10.0.2.6	Active Directory Domain Services Information Value Added Value Deleted	Malformed Security Descriptor
head.chef	Properties: Write Property {771727b1-31b8-4cdf-ae62-4fe39fadf89e} {888eedd6-ce04-df40-b462-b8a50e41ba38} {7b8b558a-93a5-4af7-adca-c017e67f1057}	Write Property	10.0.2.6	Active Directory Domain Services Information Value Added Value Deleted	Malformed Security Descriptor
head.chef	Properties: Write Property {771727b1-31b8-4cdf-ae62-4fe39fadf89e} {888eedd6-ce04-df40-b462-	Write Property	10.0.2.6	Active Directory Domain Services Information	Malformed Security Descriptor

Figure 75 - Detection With Event IDs 4662, 5136 and 4624 (2)

5.13 DCSync Rights

5.13.1 Background

The ability to perform a DCSync is prized by attackers everywhere—unsurprising, considering that having the ability to extract a domain ***ntds.dit*** file is, in many ways, how you can obtain the proverbial keys to the AD kingdom.

However, while there are many detections written to detect a DCSync happening, there are far fewer detections designed to identify when the objects that make up the privileges to do a DCSync are modified to grant an unapproved user access to perform a DCSync. The tool that I leveraged to add DCSync rights added the three (3) following rights:

- DS-Replication-Get-Changes
- DS-Replication-Get-Changes-All
- DS-Replication-Get-Changes-In-Filtered-Set

However, it is important to note that DS-Replication-Get-Changes-In-Filtered-Set is not specifically needed for an account to perform a DCSync. Only the first two (2) bullets are needed.

An important note shared with us from Jim Sykora (@jimsycurity) is that AllExtendedRights include all DCSync rights. Additionally, GenericAll includes AllExtendedRights. So, if you are looking for just the three (3) DCSync rights listed above, you may miss the broader permission sets.

5.13.2 Creating a New Account & Modifying the Objects

To start, we create a new user called **imposter.egg** that will be the account we will use to modify DACL rights to perform the DCSync.

```
net user imposter.egg /domain
```

```
PS C:\PowerSploit-master\Recon> net user imposter.egg /domain
The request will be processed at a domain controller for domain BREAKFASTLAND.LOCAL.

User name                    imposter.egg
Full Name
Comment
User's comment
Country/region code          000 (System Default)
Account active               Yes
Account expires              Never

Password last set            4/7/2023 11:04:32 AM
Password expires             5/19/2023 11:04:32 AM
Password changeable          4/8/2023 11:04:32 AM
Password required            Yes
User may change password     Yes

Workstations allowed         All
Logon script
User profile
Home directory
Last logon                   Never

Logon hours allowed          All

Local Group Memberships
Global Group memberships     *Domain Users
The command completed successfully.
```

Figure 76 - Creating an "Imposter" Account

We can use ***PowerView*** to add an ACE to domain root to grant DCSync permissions to the attacker user account. A change is made on the domainDNS root DACL via an ACE that corresponds to the user's SID.

```
Add-ObjectACL -PrincipalIdentity imposter.egg -Rights DCSync
```

```
PS C:\PowerSploit-master\Recon>   Add-ObjectACL -PrincipalIdentity imposter.egg -Rights DCSync
```

Figure 77 - Adding DCSync Rights

If we look at our privileges now, we can see the new rights reflected:

```
======================== ============================================
breakfastland\imposter.egg S-1-5-21-1865600711-3446354287-3882071624-1607
```

Figure 78 - Account SID

```
AceQualifier           : AccessAllowed
ObjectDN               : DC=BREAKFASTLAND,DC=LOCAL
ActiveDirectoryRights  : ExtendedRight
ObjectAceType          : DS-Replication-Get-Changes-In-Filtered-Set
ObjectSID              : S-1-5-21-1865600711-3446354287-3882071624
InheritanceFlags       : None
BinaryLength           : 56
AceType                : AccessAllowedObject
ObjectAceFlags         : ObjectAceTypePresent
IsCallback             : False
PropagationFlags       : None
SecurityIdentifier     : S-1-5-21-1865600711-3446354287-3882071624-1607
AccessMask             : 256
AuditFlags             : None
IsInherited            : False
AceFlags               : None
InheritedObjectAceType : All
OpaqueLength           : 0

AceQualifier           : AccessAllowed
ObjectDN               : DC=BREAKFASTLAND,DC=LOCAL
ActiveDirectoryRights  : ExtendedRight
ObjectAceType          : DS-Replication-Get-Changes
ObjectSID              : S-1-5-21-1865600711-3446354287-3882071624
InheritanceFlags       : None
BinaryLength           : 56
AceType                : AccessAllowedObject
ObjectAceFlags         : ObjectAceTypePresent
IsCallback             : False
PropagationFlags       : None
SecurityIdentifier     : S-1-5-21-1865600711-3446354287-3882071624-1607
AccessMask             : 256
AuditFlags             : None
IsInherited            : False
AceFlags               : None
InheritedObjectAceType : All
OpaqueLength           : 0

AceQualifier           : AccessAllowed
ObjectDN               : DC=BREAKFASTLAND,DC=LOCAL
ActiveDirectoryRights  : ExtendedRight
ObjectAceType          : DS-Replication-Get-Changes-All
ObjectSID              : S-1-5-21-1865600711-3446354287-3882071624
InheritanceFlags       : None
BinaryLength           : 56
AceType                : AccessAllowedObject
ObjectAceFlags         : ObjectAceTypePresent
IsCallback             : False
PropagationFlags       : None
SecurityIdentifier     : S-1-5-21-1865600711-3446354287-3882071624-1607
AccessMask             : 256
```

Figure 79 - Newly Added ACEs Added to Domain Root DACL

Now that our new user has been given the right access, we can perform DCSync using Impacket's secretsdump.py.

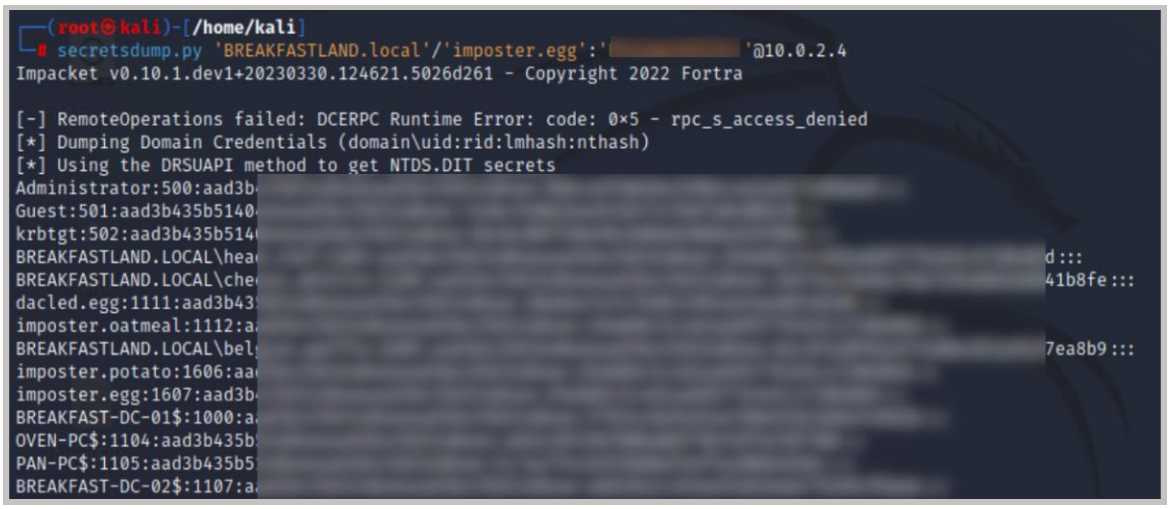


Figure 80 - DCSync

5.13.3 Building the Detections

A couple of quick notes here—the **imposter.egg** security principal is the grantee of the ACE added to the domain root DACL. When building a detection to identify DCSync rights added to a user or computer account, we will not be building a detection that searches for ***DS-Replication-Get-Changes*** or ***DS-Replication-Get-Changes-All***. Instead, we will be building detections that utilize the ***domainDNS*** class, as changes for DCSync rights are made to the domain root, and not the attributes that we are adding.

Note: If you modify the DCSync objects directly, you will generate an Event ID 5136 event with the objects of the same name. However, when I attempted this (adding an **imposter.egg** *account to each object), I was unable to perform a successful DCSync. Jim helped clarify that the reason behind this is because for DCSync (and several other rights), it only matters if the Security Principal is granted/delegated those rights at the domain root object.*

Instead, this time, we will be hunting for an Event 5136, with ***domainDNS*** as the class and the actual domain as the "DN."

5.13.3.1 Detection with Event ID 5136

```
index=main EventCode=5136 Class=domainDNS DN="DC=BREAKFASTLAND,DC=LOCAL"
| table  _time, Logon_ID, DN, GUID, Value, Class, Type
```

_time	Logon_ID	DN	GUID	Class	Type
2023-04-21 13:45:37	0x100427	DC=BREAKFASTLAND,DC=LOCAL	{754fb287-55d2-4d68-b7fc-0332e1746740}	domainDNS	Information Active Directory Domain Services Value Added
2023-04-21 13:45:37	0x100427	DC=BREAKFASTLAND,DC=LOCAL	{754fb287-55d2-4d68-b7fc-0332e1746740}	domainDNS	Information Active Directory Domain Services Value Deleted
2023-04-21 13:45:37	0x100427	DC=BREAKFASTLAND,DC=LOCAL	{754fb287-55d2-4d68-b7fc-0332e1746740}	domainDNS	Information Active Directory Domain Services Value Added

Figure 81 - Basic DCSync Query (1)

Figure 82 - Basic DCSync Query (2)

This is a decent query for detecting adding privileges for the broader domain and works both with privilege modifications through ***Powerview*** and when privileges are modified manually in ADUC. However, in our opinion, this alone lacks a lot of telemetry. Adding a correlation with Event ID 4662 for additional context is a good idea.

The “object type” in this case is the GUID of the ***domainDNS*** object, and “object_name” is the GUID of our domain (**BREAKFASTLAND.LOCAL**), which was used in our detection based on 5136 and can be filtered to look for “WRITE_DAC” access via an access mask of 0x4000.

5.13.3.2 Detection with Event ID 4662

```
index=main Account_Name!=*$ Object_Type="%{19195a5b-6da0-11d0-afd3-
00c04fd930c9}" Object_Name="%{754fb287-55d2-4d68-b7fc-
0332e1746740}"  EventCode=4662
Access_Mask = 0x40000 | table time, Logon_ID, Account_Name, Access_Mask,
Object_Type, Object_Name, Properties
```

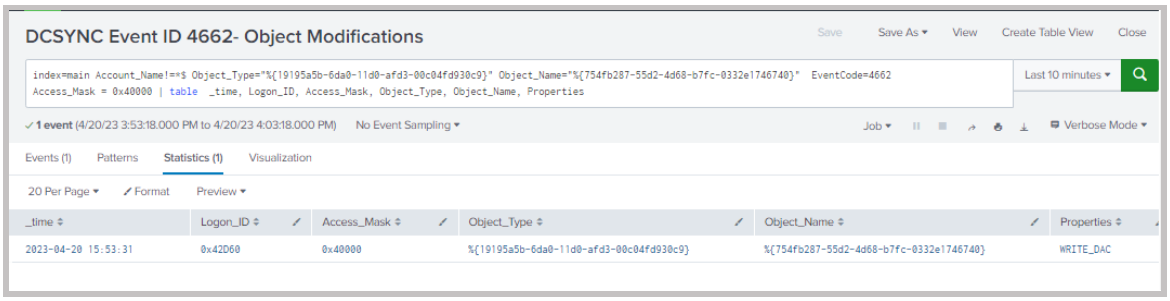


Figure 83 - DCSync With Event ID 4662

Then, if you are using a SIEM with the ability to utilize a complex query language, you can combine the two (2) queries to put all the telemetry in one (1) view. As a quick note, this does place all the entries in the “Value” field in one (1) column, but it condenses the table view in Splunk to one (1) line, making it easy to quickly look at and reference the events in a separate tab if more detail is needed.

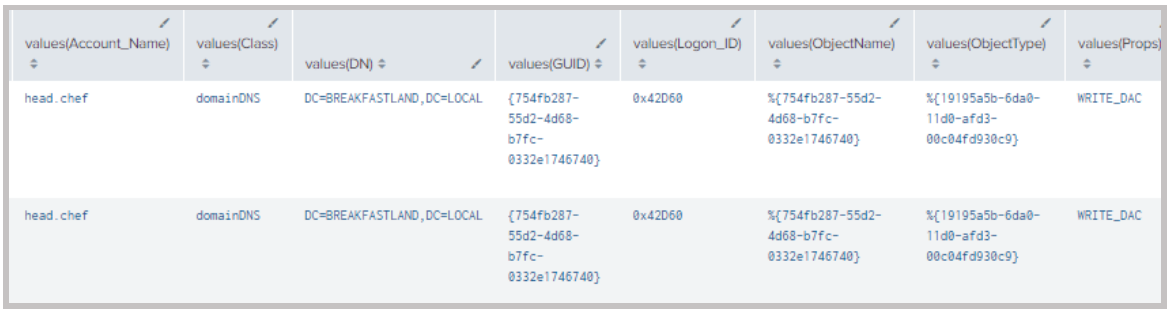
5.13.3.3 Detection with Event ID 5136 and 4662

```
index=main ((EventCode=5136 Class=domainDNS
DN="DC=BREAKFASTLAND,DC=LOCAL") OR (index=main Account_Name!=*$
Object_Type="%{19195a5b-6da0-11d0-afd3-00c04fd930c9}"
Object_Name="%{754fb287-55d2-4d68-b7fc-0332e1746740}"  EventCode=4662
Access_Mask = 0x40000))
| eval Logon_ID=if(EventCode==4662,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval user=if(EventCode==4662,mvindex( Account_Name,-1), mvindex(
Account_Name,-1))
| join type=outer Logon_ID
    [ search index=main Account_Name!=*$ Object_Type="%{19195a5b-6da0-
11d0-afd3-00c04fd930c9}" Object_Name="%{754fb287-55d2-4d68-b7fc-
0332e1746740}"  EventCode=4662 Access_Mask = 0x40000
    | eval Props=Properties
    | eval AccessMask=Access_Mask
    | eval ObjectType=Object_Type
    | eval ObjectName=Object_Name
```

```
    |table Account_Name,Logon_ID,Props,AccessMask,ObjectType, ObjectName]
| table time, Logon_ID, Account_Name, Props, AccessMask, ObjectType,
ObjectName,  DN, GUID, Value, Class
|stats values by Value
```

Value	values(AccessMask)
O:BAG:BAD:AI(OA;;CR;1131f6aa-9c07-11d1-f79f-00c04fc2dcd2;;S-1-5-21-1865600711-3446354287-3882071624-498)(OA;;CR;1131f6ad-9c07-11d1-f79f-00c04fc2dcd2;;DD)(OA;CIIO;SW;9b026da6-0d3c-465c-8bee-5199d7165cba;bf967a86-0de6-11d0-a285-00aa003049e2;CO)(OA;CIIO;SW;9b026da6-0d3c-465c-8bee-5199d7165cba;bf967a86-0de6-11d0-a285-00aa003049e2;PS)(OA;CIIO;RP;b7c69e6d-2cc7-11d2-854e-00a0c983f608;bf967a86-0de6-11d0-a285-00aa003049e2;ED)(OA;CIIO;RP;b7c69e6d-2cc7-11d2-854e-00a0c983f608;bf967a9c-0de6-11d0-a285-00aa003049e2;ED)(OA;CIIO;RP;b7c69e6d-2cc7-11d2-854e-00a0c983f608;bf967aba-0de6-11d0-a285-00aa003049e2;ED)(OA;CIIO;WP;ea1b7b93-5e48-46d5-bc6c-4df4fda78a35;bf967a86-0de6-11d0-a285-00aa003049e2;PS)(OA;;CR;89e95b76-444d-4c62-991a-0facbeda640c;;BA)(OA;;CR;1131f6aa-9c07-11d1-f79f-00c04fc2dcd2;;BA)(OA;;CR;1131f6ab-9c07-11d1-f79f-00c04fc2dcd2;;BA)(OA;;CR;1131f6ac-9c07-11d1-f79f-00c04fc2dcd2;;BA)(OA;;CR;1131f6ad-9c07-11d1-f79f-00c04fc2dcd2;;BA)(OA;;CR;1131f6ae-9c07-11d1-f79f-00c04fc2dcd2;;BA)(OA;;CR;e2a36dc9-ae17-47c3-b58b-be34c55ba633;;S-1-5-32-557)(OA;CIIO;LCRPLORC;;4828cc14-1437-45bc-9b07-ad6f015e5f28;RU)(OA;CIIO;LCRPLORC;;bf967a9c-0de6-11d0-a285-00aa003049e2;RU)(OA;CIIO;LCRPLORC;;bf967aba-0de6-11d0-a285-00aa003049e2;RU)(OA;;CR;05c74c5e-4deb-43b4-bd9f-86664c2a7fd5;;AU)(OA;;CR;89e95b76-444d-4c62-991a-0facbeda640c;;ED)(OA;;CR;ccc2dc7d-a6ad-4a7a-8846-c04e3cc53501;;AU)(OA;;CR;280f369c-67c7-438e-ae98-1d46f3c6f541;;AU)(OA;;CR;1131f6aa-9c07-11d1-f79f-00c04fc2dcd2;;ED)(OA;;CR;1131f6ab-9c07-11d1-f79f-00c04fc2dcd2;;ED)(OA;;CR;1131f6ac-9c07-11d1-f79f-00c04fc2dcd2;;ED)(OA;;CR;1131f6ae-9c07-11d1-f79f-00c04fc2dcd2;;ED)(OA;CI;RPWP;3f78c3e5-f79a-46bd-a0b8-9d18116ddc79;;PS)(OA;CIIO;RPWPCR;91e647de-d96f-4b70-9557-d63ff4f3ccd8;;PS)(A;;CCLCSWRPWPLOCRRCWDWO;;;DA)(A;CI;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;S-1-5-21-1865600711-3446354287-3882071624-519)(A;;RPRC;;;RU)(A;CI;LC;;;RU)(A;CI;CCLCSWRPWPLOCRSDRCWDWO;;;BA)(A;;RP;;;WD)(A;;LCRPLORC;;;ED)(A;;LCRPLORC;;;AU)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;SY)S:AI(OU;CISA;WP;f30e3bbe-9ff0-11d1-b603-0000f80367c1;bf967aa5-0de6-11d0-a285-00aa003049e2;WD)(OU;CISA;WP;f30e3bbf-9ff0-11d1-b603-0000f80367c1;bf967aa5-0de6-11d0-a285-00aa003049e2;WD)(OU;CISAFA;CR;89e95b76-444d-4c62-991a-0facbeda640c;;WD)(OU;CISAFA;CR;1131f6aa-9c07-11d1-f79f-00c04fc2dcd2;;WD)(OU;CISAFA;CR;1131f6ad-9c07-11d1-f79f-00c04fc2dcd2;;WD)(AU;SA;CR;;;DU)(AU;SA;CR;;;BA)(AU;SA;WPWDWO;;;WD)(AU;CISAFA;LCRPWPRC;;;WD)	0x40000
O:BAG:BAD:AI(OA;;CR;1131f6aa-9c07-11d1-f79f-00c04fc2dcd2;;S-1-5-21-1865600711-3446354287-3882071624-498)(OA;;CR;1131f6ad-9c07-11d1-f79f-00c04fc2dcd2;;DD)(OA;CIIO;SW;9b026da6-0d3c-465c-8bee-5199d7165cba;bf967a86-0de6-11d0-a285-00aa003049e2;CO)(OA;CIIO;SW;9b026da6-0d3c-465c-8bee-5199d7165cba;bf967a86-0de6-11d0-a285-00aa003049e2;PS)(OA;CIIO;RP;b7c69e6d-2cc7-11d2-854e-00a0c983f608;bf967a86-0de6-11d0-a285-00aa003049e2;ED)(OA;CIIO;RP;b7c69e6d-2cc7-11d2-854e-00a0c983f608;bf967a9c-0de6-11d0-a285-00aa003049e2;ED)(OA;CIIO;RP;b7c69e6d-2cc7-11d2-854e-00a0c983f608;bf967aba-0de6-11d0-a285-00aa003049e2;ED)(OA;CIIO;WP;ea1b7b93-5e48-46d5-bc6c-4df4fda78a35;bf967a86-0de6-11d0-a285-00aa003049e2;PS)(OA;;CR;89e95b76-444d-4c62-991a-0facbeda640c;;BA)(OA;;CR;1131f6aa-9c07-11d1-f79f-00c04fc2dcd2;;BA)(OA;;CR;1131f6ab-	0x40000

Figure 84 - Detecting Changes to the Domain (1)



values(Account_Name)	values(Class)	values(DN)	values(GUID)	values(Logon_ID)	values(ObjectName)	values(ObjectType)	values(Props)
head.chef	domainDNS	DC=BREAKFASTLAND,DC=LOCAL	{754fb287-55d2-4d68-b7fc-0332e1746740}	0x42D60	%{754fb287-55d2-4d68-b7fc-0332e1746740}	%{19195a5b-6da0-11d0-afd3-00c04fd930c9}	WRITE_DAC
head.chef	domainDNS	DC=BREAKFASTLAND,DC=LOCAL	{754fb287-55d2-4d68-b7fc-0332e1746740}	0x42D60	%{754fb287-55d2-4d68-b7fc-0332e1746740}	%{19195a5b-6da0-11d0-afd3-00c04fd930c9}	WRITE_DAC

Figure 85 - Detecting Changes to the Domain (2)

This is a decent query; however, it still has some issues—namely, that the detection itself does not *only* detect changes made to objects for our DCSync rights, but also other changes made to the domain. For example, when we modified the **dacled.egg** user to also have user create rights, as well as the DCSync rights on the domain, the query was triggered. The top portion shows the DCSync rights changes, and the bottom portion shows the change that was triggered when we added the user create rights.

Figure 86 - Additional Events

To build a higher fidelity detection, we need to address the 'Value' field and attempt to understand the hieroglyphic text that are SDDLs.

In order to filter down the detection output to include changes that only identify the addition/removal of DCSync rights added to the domain for a user/computer, we added the following SDDL prefix strings to each of the DCSync rights GUIDs. This effectively captured the addition of the DCSync rights but excluded changes made to other attributes, such as "User Creation" rights added to the domain root.

DS-Replication-Get-Changes-In-Filtered-Set	DS-Replication-Get-Changes	DS-Replication-Get-Changes-All	RightsGUID for User objects
OA;CI;CR;89e95b76-444d-4c62-991a-0facbeda640c;;	OA;CI;CR;1131f6aa-9c07-11d1-f79f-00c04fc2dcd2;;)	OA;CI;CR;1131f6ad-9c07-11d1-f79f-00c04fc2dcd2;;	OA;CI;CC;bf967aba-0de6-11d0-a285-00aa003049e2;;

The SDDL string values to note here are:

- OA - Object Access Allowed

- CI - Container Inherit; DirectoryObjectControlAccess.
- CC - DirectoryObjectCreateChild
- CR - All Extended Rights

And if we translate the SDDL using our SDDL conversion tool, we can see that the permissions for the **dacled.egg** attribute were in fact made to the domain.

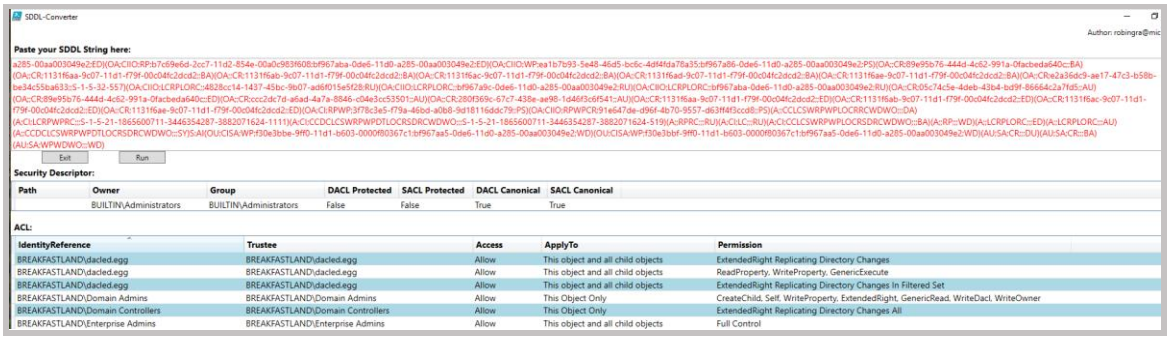


Figure 87 - SDDL Editor String for DCSync

With a slight modification, we were able build an additional query that did not pick up the "User Creation" right that added an additional search value looking for the string `OA;CI;CR;.`

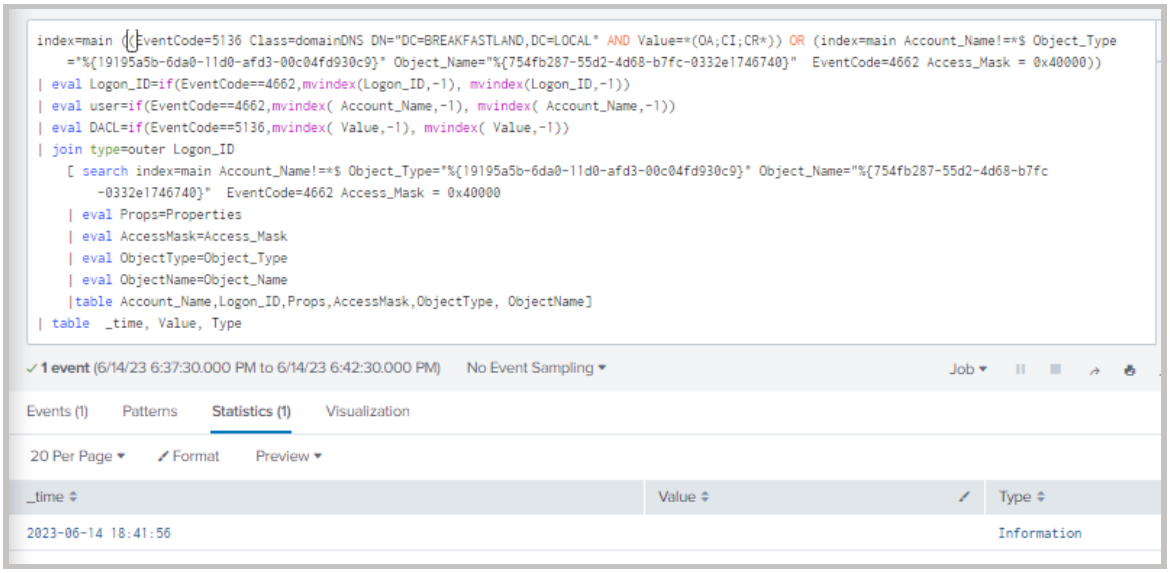


Figure 88 - Comparison Post Right User Creation Added

However, the query still identified when the DCSync rights were added.

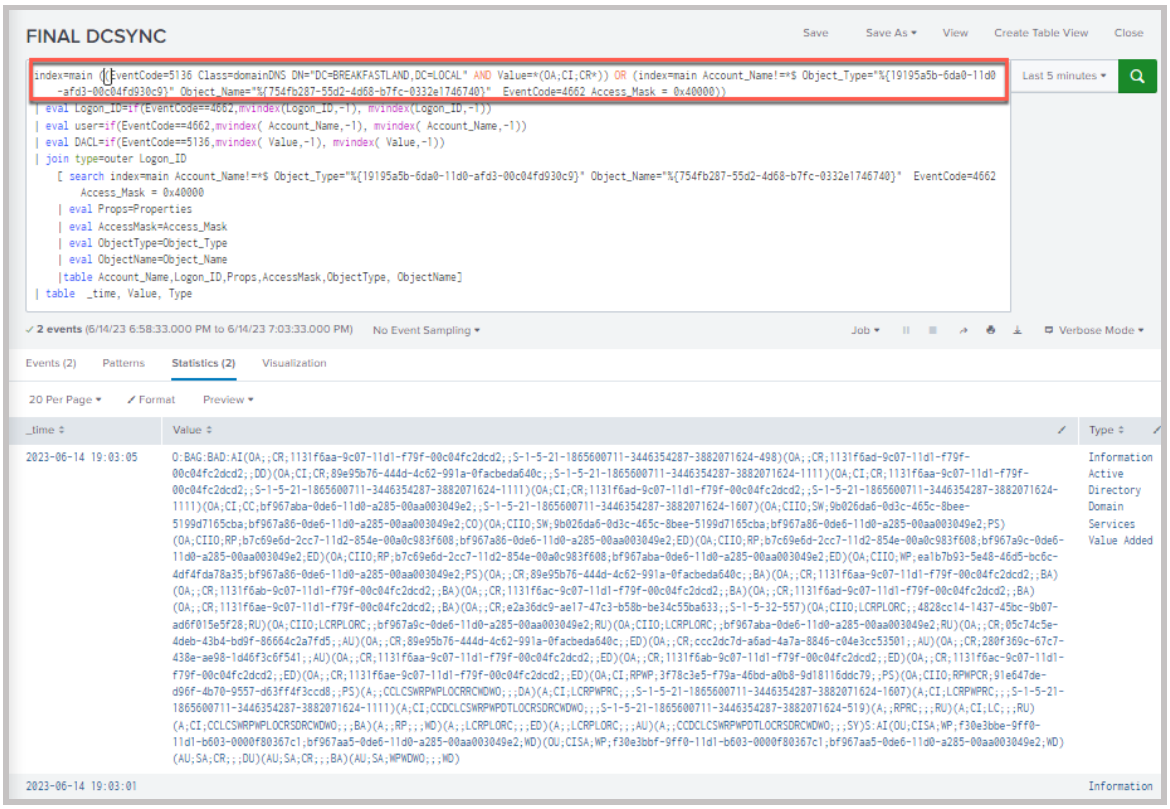


Figure 89 - Comparison Post DCSync Added

5.13.3.4 Detection with Event ID 5136 and 4662 (Improved Filtering)

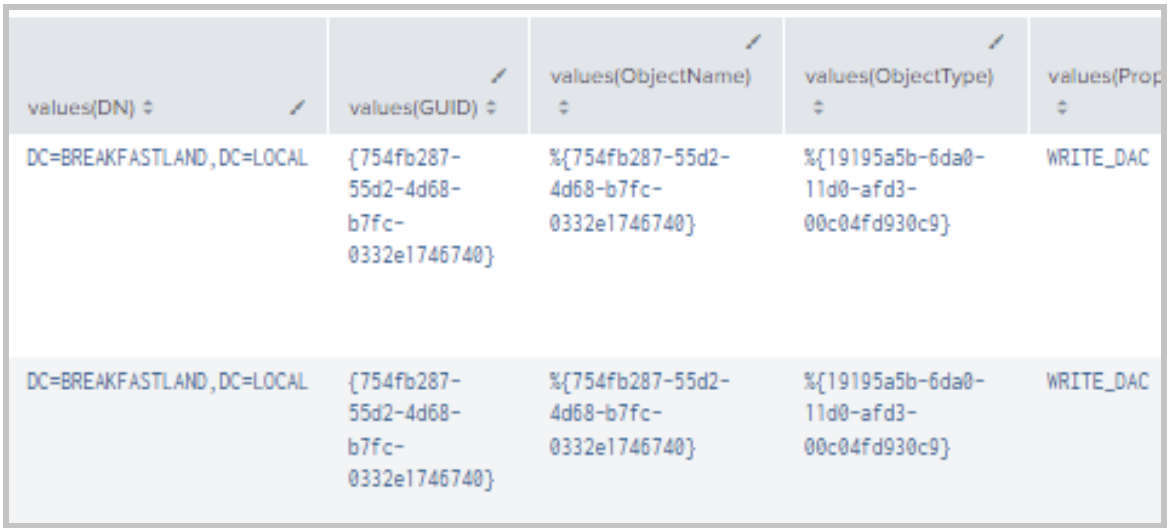
And then lastly, one (1) final change was made to the query—instead of using only the `OA;CI;CR;` as the filter parameter, it was expanded to include the DCSync rights objects.

```
index=main ((EventCode=5136 Class=domainDNS
DN="DC=BREAKFASTLAND,DC=LOCAL" AND Value=*(OA;CI;CR;89e95b76-444d-4c62-
991a-0facbeda640c)(OA;CI;CR;1131f6aa-9c07-11d1-f79f-
00c04fc2dcd2)(OA;CI;CR;1131f6ad-9c07-11d1-f79f-00c04fc2dcd2)*) OR
(index=main Account_Name!=*$ Object_Type="%{19195a5b-6da0-11d0-afd3-
00c04fd930c9}" Object_Name="%{754fb287-55d2-4d68-b7fc-0332e1746740}"
EventCode=4662 Access_Mask = 0x40000))
| eval Logon_ID=if(EventCode==4662,mvindex(Logon_ID,-1),
mvindex(Logon_ID,-1))
| eval user=if(EventCode==4662,mvindex( Account_Name,-1), mvindex(
Account_Name,-1))
| eval DACL=if(EventCode==5136,mvindex( Value,-1), mvindex( Value,-1))
| join type=outer Logon_ID
    [ search index=main Account_Name!=*$ Object_Type="%{19195a5b-6da0-
11d0-afd3-00c04fd930c9}" Object_Name="%{754fb287-55d2-4d68-b7fc-
0332e1746740}"  EventCode=4662 Access_Mask = 0x40000
    | eval Props=Properties
    | eval AccessMask=Access_Mask
    | eval ObjectType=Object_Type
```

```
    | eval ObjectName=Object_Name
    |table Account_Name,Logon_ID,Props,AccessMask,ObjectType, ObjectName]
| table  _time, Logon_ID, Account_Name, Props, AccessMask, ObjectType,
ObjectName,  DN, GUID, DACL, Class
|stats values by Logon_ID
```

Logon_ID	values(AccessMask)	values(Account_Name)	values(Class)	values(DACL)
0x149DA3	0x40000	head.chef	domainDNS	O:BAG:BAD:AI(OA;;CR;1131f6aa-9c07-11d1-f79f-00c04fc2dcd2;;S-1-5-21-1865600711-3446354287-3882071624-498)(OA;;CR;1131f6ad-9c07-11d1-f79f-00c04fc2dcd2;;DD)(OA;CI;CR;89e95b76-444...
0x387B2	0x40000	head.chef	domainDNS	O:BAG:BAD:AI(OA;;CR;1131f6aa-9c07-11d1-f79f-00c04fc2dcd2;;S-1-5-21-1865600711-3446354287-3882071624-498)(OA;;CR;1131f6ad-9c07-11d1-f79f-00c04fc2dcd2;;DD)(OA;CI;CR;89e95b76-444...

Figure 90 - Final DCSync Query (1)



values(DN)	values(GUID)	values(ObjectName)	values(ObjectType)	values(Prop
DC=BREAKFASTLAND,DC=LOCAL	{754fb287-55d2-4d68-b7fc-0332e1746740}	%{754fb287-55d2-4d68-b7fc-0332e1746740}	%{19195a5b-6da0-11d0-afd3-00c04fd930c9}	WRITE_DAC
DC=BREAKFASTLAND,DC=LOCAL	{754fb287-55d2-4d68-b7fc-0332e1746740}	%{754fb287-55d2-4d68-b7fc-0332e1746740}	%{19195a5b-6da0-11d0-afd3-00c04fd930c9}	WRITE_DAC

Figure 91 - Final DCSync Query (2)

6 Conclusion of Part 1

While this isn't the end of this series (stay tuned for parts 2 and 3), there are a few key takeaways from this post that are important to note. Most notably, while Event ID 5136 is the core event throughout all the detections that were built throughout Parts 1 through 3, there are accompanying events that are equally important to these detections. Event ID 4624 will give you a source IP, while Event ID 4662 can provide context to the detection.

It's also important to note that in most cases, multiple queries were built for each detection where possible. This is because it is critical to have multiple points of detection to fall back

on in case attackers manage to develop new attacks that may not trigger one (1) solitary detection. Supplementally, most if not all queries in this post should be tweaked to account for differing GUIDs and objects within your own domain, while some will remain the same, such as the GUIDs for the DCSync Rights objects. Other methods of detection can also be incorporated here, such as PowerShell event logging, and while we were unable to find these detections in LDAP logs, they could possibly exist in that telemetry as well.

And lastly, it is important to remember that for SIEMs that may not have the advanced query language that exists within Splunk, all these queries can be broken into smaller sections or multiple queries.

Lastly, this blog would not have been possible without help from the following people:

Charlie Bromberg (@shutdownrepo)
Jonathan Johnson (@jsecurity101)
Jim Sykora (@jimsycurity)
Kelsey Segure (@KelseySegrue)

7 References:

https://www.thehacker.recipes/ad/movement/dacl

Windows Events:
https://learn.microsoft.com/en-us/windows/security/threat-protection/auditing/event-4662
https://learn.microsoft.com/en-us/windows/security/threat-protection/auditing/event-4624
https://learn.microsoft.com/en-us/windows/security/threat-protection/auditing/event-5145
https://learn.microsoft.com/en-us/windows/security/threat-protection/auditing/event-4742
https://learn.microsoft.com/en-us/windows/security/threat-protection/auditing/event-4738

msDS-AllowedtoActOnBehalfOfOtherIdentity:
https://learn.microsoft.com/en-us/windows/win32/adschema/a-msds-allowedtoactonbehalfofotheridentity
https://jsecurity101.medium.com/defending-the-three-headed-relay-17e1d6b6a339

Service Principal Name (SPN):
https://learn.microsoft.com/en-us/windows/win32/ad/service-principal-names
https://www.semperis.com/blog/spn-jacking-an-edge-case-in-writespn-abuse/
https://blog.harmj0y.net/activedirectory/targeted-kerberoasting/
https://learn.microsoft.com/en-us/windows/win32/ad/mutual-authentication-using-kerberos
https://github.com/fortra/impacket
https://github.com/fortra/impacket/blob/master/examples/GetUserSPNs.py

https://github.com/Kevin-Robertson/Powermad

msDS-AllowedtoDelegateTo:
https://learn.microsoft.com/en-us/windows/win32/adschema/a-msds-allowedtodelegateto
https://skyblue.team/posts/delegate-krbtgt/
https://csandker.io/2020/02/10/KerberosDelegationAWrapUp.html

msDS-KeyCredentialLink:
https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-ada2/45916e5b-d66f-444e-b1e5-5b0666ed4d66
https://posts.specterops.io/shadow-credentials-abusing-key-trust-account-mapping-for-takeover-8ee1a53566ab
https://cyberstoph.org/posts/2022/03/detecting-shadow-credentials/

ScriptPath:
https://learn.microsoft.com/en-us/windows/win32/adschema/a-scriptpath

msTSInitalProgram:
https://learn.microsoft.com/en-us/windows/win32/adschema/a-mstsinitialprogram

GPO:
https://learn.microsoft.com/en-us/windows/win32/adschema/c-grouppolicycontainer
https://github.com/Hackndo/pyGPOAbuse
https://github.com/X-C3LL/GPOwned
https://www.thehacker.recipes/ad/movement/group-policies
https://learn.microsoft.com/en-us/windows/win32/adschema/c-grouppolicycontainer
https://wald0.com/?p=179
https://serverfault.com/questions/692772/group-managed-service-accounts-principalsallowedtoretrievemanagedpassword
https://serverfault.com/questions/692772/group-managed-service-accounts-principalsallowedtoretrievemanagedpassword
https://labs.withsecure.com/tools/sharpgpoabuse

AddMember:
https://www.thehacker.recipes/ad/movement/dacl/addmember
https://github.com/PowerShellMafia/PowerSploit
https://learn.microsoft.com/en-us/windows/win32/adschema/r-self-membership
https://learn.microsoft.com/en-us/windows/win32/adschema/a-member

ForceChangePassword:
https://www.thehacker.recipes/ad/movement/dacl/forcechangepassword
https://learn.microsoft.com/en-us/windows/win32/adschema/r-user-force-change-password

GrantOwnerShip:
https://www.thehacker.recipes/ad/movement/dacl/grant-ownership

LAPS/GMSA:
https://www.trustedsec.com/blog/splunk-spl-queries-for-detecting-gmsa-attacks/
https://www.trustedsec.com/blog/a-lapse-in-judgement/
https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-ada2/60acc5e9-e6dc-481f-a3ff-2cb763ab2d33
https://learn.microsoft.com/en-us/powerquery-m/datetime-fromfiletime
https://adsecurity.org/?p=4367
https://learn.microsoft.com/en-us/powershell/module/activedirectory/?view=windowsserver2022-ps

DCSync:
https://github.com/fortra/impacket
https://www.alteredsecurity.com/post/a-primer-on-dcsync-attack-and-detection
https://www.thehacker.recipes/ad/movement/credentials/dumping/dcsync
https://itconnect.uw.edu/tools-services-support/it-systems-infrastructure/msinf/other-help/understanding-sddl-syntax/

msDS-GroupManagedServiceAccount/msDS-ManagedServiceAccount References:
https://woshub.com/group-managed-service-accounts-in-windows-server-2012/
https://blog.netwrix.com/2022/10/13/group-managed-service-accounts-gmsa/

PowerMad/Set-MachineAcccountAttribute:
https://github.com/Kevin-Robertson/Powermad
https://stackoverflow.com/questions/39226518/filtering-only-second-account-name-in-windows-event-log-using-a-regex
https://learn.microsoft.com/en-us/troubleshoot/windows-server/identity/useraccountcontrol-manipulate-account-properties
https://skyblue.team/posts/delegate-krbtgt/

Other:
An ACE in the Hole Stealthy Host Persistence via Security Descriptors [Corrected Audio]
https://specterops.io/wp-content/uploads/sites/3/2022/06/an_ace_up_the_sleeve.pdf