

Tech Community Live: Microsoft Security

Dec 03 2024, 07:00 AM - 11:30 AM (PST) 📍 Microsoft Tech Community

Find out more >

[Home](#) > [Security, Compliance, and Identity](#) > [Microsoft Defender for Endpoint Blog](#) > Hunting for reconnaissance activities using LDAP search filters

Back to Blog

< Newer Article

Older Article >

Hunting for reconnaissance activities using LDAP search filters

By  [Corina Feuerstein](#)

Published Aug 28 2019 05:00 AM

👁 64.2K Views



The Lightweight Directory Access Protocol ([LDAP](#)) protocol is heavily used by system services and apps for many important operations like querying for user groups and getting user information. It's a prime target for [Active Directory attacks](#), [Kerberoasting](#), and other reconnaissance steps after attackers have infiltrated a network.

Attackers are known to use LDAP to gather information about users, machines, and the domain structure. Attackers can then take over high-privileged accounts by finding the shortest path to sensitive assets. Spotting these reconnaissance activities, especially from patient zero machines, is critical in detecting and containing cyberattacks.

A new LDAP extension to Windows endpoints provides visibility into LDAP search queries. This instrumentation is captured by [Microsoft Defender ATP](#), allowing blue teams to hunt down suspicious queries and prevent attacks in their early stages. In this blog we'll demonstrate how you can use [advanced hunting](#) in Microsoft Defender ATP to investigate suspicious LDAP search queries.

Case study: Hunting down LDAP-based attacks

To demonstrate how the new LDAP instrumentation works, I set up a test machine and installed the popular red-team tool [BloodHound](#) and used [SharpHound](#) as data collector tool to gather and ingest domain data. SharpHound uses LDAP queries to collect domain information that can used later to perform attacks against the organization:

```
Initializing BloodHound at 9:04 AM on 7/17/2019
Resolved Collection Methods to Group, LocalAdmin, Session, Trusts, RDP, DCOM
Starting Enumeration for lmsdn.local
Status: 75 objects enumerated (+75 75/s --- Using 43 MB RAM )
Finished enumeration for lmsdn.local in 00:00:01.4400335
6 hosts failed ping. 0 hosts timedout.

Compressing data to .\20190717090402_BloodHound.zip.
You can upload this file directly to the UI.
Finished compressing files!
```

Figure 1. SharpHound is collecting domain objects from lmsdn.local domain

Microsoft Defender ATP captures the queries run by Sharphound, as well as the actual processes that were used. Using a simple advanced hunting query that performs the following steps, we can spot highly interesting reconnaissance methods:

Version history

Last update: Nov 14 2019 02:39 PM

Updated by: [Louie Mayor](#)

Labels

Advanced hunting 20

Share



1. Search for LDAP search filters events (ActionType = LdapSearch)
2. Parse the LDAP attributes and flatten them for quick filtering
3. Use a distinguished name to target your searches on designated domains
4. If needed, filter out prevalent queries to reduce noise or define specific filters
5. Investigate the machine and its processes used with suspicious queries

Here are some sample queries:

```
MiscEvents | where ActionType == "LdapSearch" and EventTime > ago(7h)
| project ComputerName, InitiatingProcessFileName, AdditionalFields
| extend ldap = parse_json(AdditionalFields)
| extend AttributeList = ldap.AttributeList
| extend ScopeOfSearch = ldap.ScopeOfSearch
| extend SearchFilter = ldap.SearchFilter
| extend DistinguishName = ldap.DistinguishedName
| where DistinguishName contains "[YourDistinguishedName]" and SearchFilter
| project-away AdditionalFields, ldap
```

```
MiscEvents | where ActionType == "LdapSearch" and EventTime > ago(7h)
| project ComputerName, InitiatingProcessFileName, AdditionalFields
| extend LDAP = parse_json(AdditionalFields)
| extend AttributeList = LDAP.AttributeList
| extend ScopeOfSearch = LDAP.ScopeOfSearch
| extend SearchFilter = LDAP.SearchFilter
| extend DistinguishName = LDAP.DistinguishedName
| where AttributeList has "admincount"
| where SearchFilter has "user" or SearchFilter has "computer" or SearchFilter has "ms-mcs-admpwdexpirationtime"
```

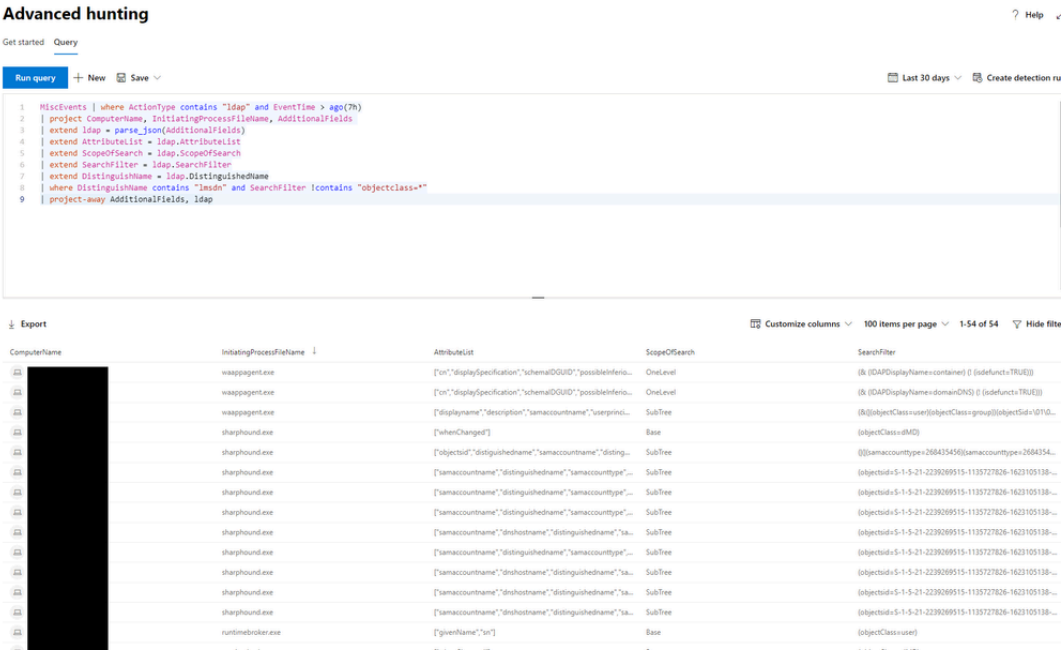


Figure 2. Advanced hunting showing example LDAP query results

One of the results that caught my attention is a generic LDAP query generated by *sharphound.exe* that aims to collect many different entities from the domain:

Process: sharphound.exe
DistinguishName: DC=lmsdn,DC=local
ScopeOfSearch: SubTree
AttributeList:
["objectsid","distiguishedname","samaccountname","distinguishedname","samaccounttype","member","cn","primarygroupid","dnshostname","ms-mcs-admpwdexpirationtime"]

```
(|
(|(samaccounttype=268435456)(samaccounttype=268435457)
(samaccounttype=536870912)(smaccounttype=536870913)
(primarygroupid=*))
(&(sAMAccountType=805306369)(!
(UserAccountControl:1.2.840.113556.1.4.803:=2)))
(objectclass=domain)
)
```

Breaking this search query into a visualized tree shows that this query gathers groups, enabled machines, users and domain objects:

- **OR**
 - **OR**
 - samaccounttype=268435456 (security groups)
 - samaccounttype=268435457 (non-security groups)
 - samaccounttype=536870912 (alias objects)
 - samaccounttype=536870913 (non-security alias objects)
 - primarygroupid=* (any object with a primary group id including users and machines)
 - **AND**
 - sAMAccountType=805306369 (machine objects)
 - **NOT**
 - UserAccountControl:1.2.840.113556.1.4.803:=2 (disabled objects)
 - objectclass=domain (domain object)

When looking at [SharpHound code](#), we can verify that the BuildLdapData method uses these filters and attributes to collect data from internal domains, and later uses this to build the BloodHound attack graph:

```
internal class LdapFilter
{

    internal static LdapData BuildLdapData(ResolvedCollectionMethod methods, bool excludeDc, string ldapFilter)
    {
        var filterparts = new List<string>();
        var props = new List<string> {"objectsid","distiguishedname"};
        if ((methods & ResolvedCollectionMethod.Group) != 0)
        {
            filterparts.Add("(|(samaccounttype=268435456)(samaccounttype=268435457)(samaccounttype=536870912)(samaccounttype=536870913)");
            props.AddRange(new[]
            {
                "samaccountname", "distinguishedname", "samaccounttype", "member", "cn", "primarygroupid", "dnshostname"
            });
        }

        if ((methods & ResolvedCollectionMethod.LocalAdmin) != 0 ||
            (methods & ResolvedCollectionMethod.Session) != 0 ||
            (methods & ResolvedCollectionMethod.LoggedOn) != 0 ||
            (methods & ResolvedCollectionMethod.RDP) != 0 ||
```

Figure 3. SharpHound code

As we can learn from the BloodHound example, when dealing with LDAP queries, search filters become an important need to specify, target and reduce the number of resulting domain entities. While BloodHound is just an example for such a case, there are many other tools out there that use the same method. Utilizing these new LDAP search filters events can help us gain better visibility into recon executions and detect suspicious attempts in no time.can help us gain better visibility into recon executions and detect suspicious attempts in no time!

Common suspicious LDAP search filters

The Microsoft Defender ATP Research Team has compiled a list of suspicious search filter queries found being used in the wild by commodity and recon tools. This list provides insights and highlights interesting LDAP query filters originating from fileless or file-based executions:

on our experience:

Q: Is this search filter generic (e.g., searching for all servers)? Did you spot wildcards?

A: In many cases we’ve observed, generic filters and wildcards are used to pull out entities from the domain. Usually, the filters were pointing to user information, machines, groups, SPNs, and domain objects.

Q: How often do you see this query? Did it try to run on many entities? Is it unique to the process or the user?

A: Anomalies can help you understand how common an activity is, and whether or not it deviated from its normal behavior. Uncommon queries originating from abnormal users, living-off-the-land binaries, injected processes, low-prevalent processes, or even known recon tools are areas that might be interesting to start investigations from.

Q: Did you encounter any interesting attributes (e.g., personal user data, machine info)?

A: Attributes can shed light on the intent and the type of data that is extracted. There is no real need to specify them, but in some cases, if appear, they can help understand what type of data was extracted.

Q: Is the scope of search is limited or multi-level (e.g., subtree vs. one-level)?

A: In many cases we’ve observed subtree search which intends to look at all child and based object which basically reduce the number of queries one would need to do.

Q: Did you find any additional artifacts for malicious activities?

A: While queries might look suspicious, it might not be enough to incriminate a malicious activity.

As true for many hunting cases, looking in additional activities could help conclude if this query was truly suspicious or not.


With these new LDAP search filter events, you can expand your threat hunting scenarios. [Advanced hunting](#) is a powerful capability in Microsoft Defender ATP that allows you to hunt for possible threats across your organization. If you are not yet reaping the benefits of Microsoft Defender ATP’s [industry-leading optics and detection capabilities](#), [sign up for free trial](#) today.

Building off of Microsoft Defender ATP’s threat hunting technology, we’re adding the ability to [hunt for threats across endpoints and email](#) through Microsoft Threat Protection. To learn more, [visit the Microsoft Threat Protection website](#).

Niv Sela, Corina Feuerstein
Microsoft Defender ATP Team

👍 6 Likes


4 Comments

- 

[Darren Mar-Elia](#) Copper Contributor

Aug 28 2019 05:01 PM

This is an interesting approach but I have to wonder about false positives in larger organizations. What are you seeing as to the signal-to-noise ratio of this type of monitoring in practice?

👍 0 Likes
- 

[canders3](#) Copper Contributor

Aug 29 2019 06:27 AM

Watching with anticipation for the next Sysmon update!

0 Likes

[AusSupport180](#) Brass Contributor

May 13 2022 07:35 PM

MiscEvents

Not identified?

0 Likes

[personabcd1234](#) Copper Contributor

May 19 2022 12:56 PM

@AusSupport180 Table naming has [changed](#), MiscEvents is now called "DeviceEvents"

0 Likes

You must be a registered user to add a comment. If you've already registered, sign in. Otherwise, register and sign in.

[Comment](#)

What's new

- Surface Pro 9
- Surface Laptop 5
- Surface Studio 2+
- Surface Laptop Go 2
- Surface Laptop Studio
- Surface Duo 2
- Microsoft 365
- Windows 11 apps

Business

- Microsoft Cloud
- Microsoft Security
- Dynamics 365
- Microsoft 365
- Microsoft Power Platform
- Microsoft Teams
- Microsoft Industry
- Small Business

Microsoft Store

- Account profile
- Download Center
- Microsoft Store support
- Returns
- Order tracking
- Virtual workshops and training
- Microsoft Store Promise
- Flexible Payments

Developer & IT

- Azure
- Developer Center
- Documentation
- Microsoft Learn
- Microsoft Tech Community
- Azure Marketplace
- AppSource
- Visual Studio

Education

- Microsoft in education
- Devices for education
- Microsoft Teams for Education
- Microsoft 365 Education
- Education consultation appointment
- Educator training and development
- Deals for students and parents
- Azure for students

Company

- Careers
- About Microsoft
- Company news
- Privacy at Microsoft
- Investors
- Diversity and inclusion
- Accessibility
- Sustainability