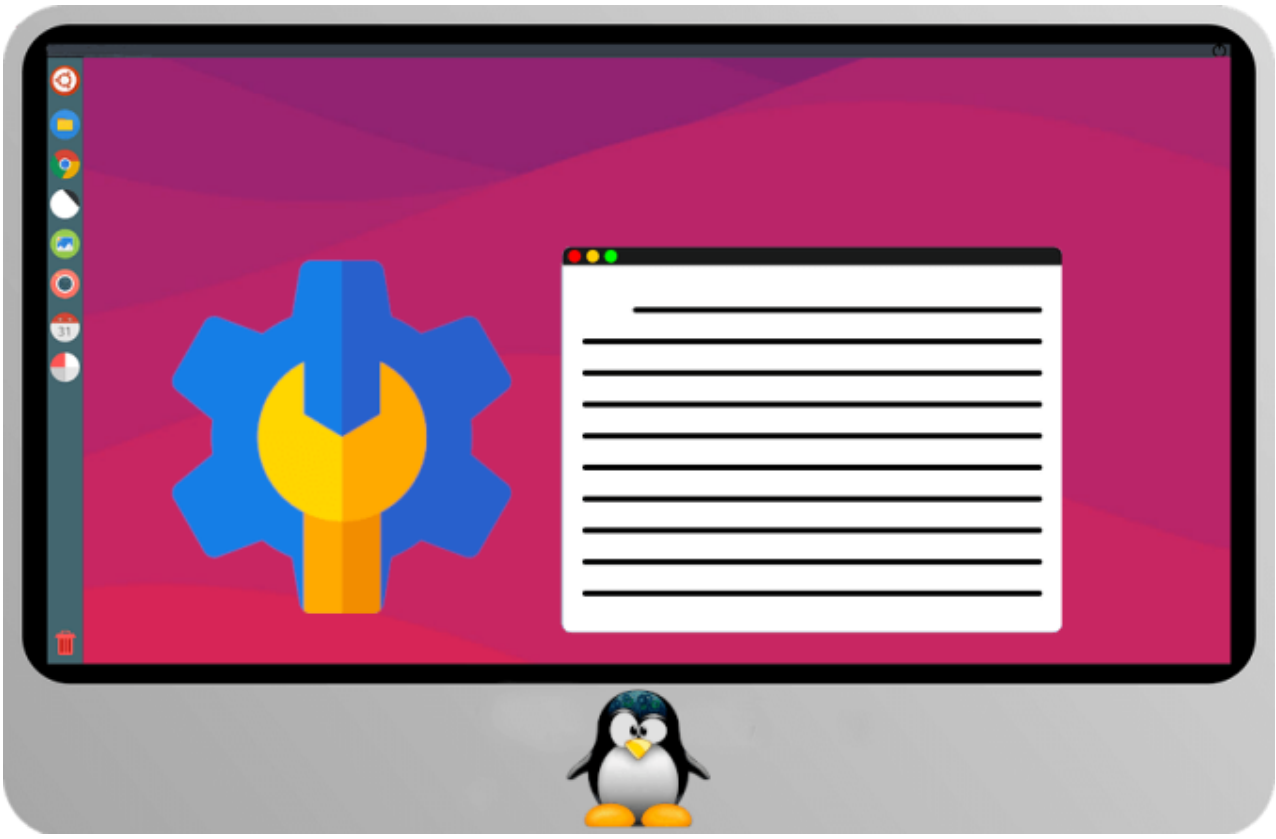
Managing SUDO from Active Directory

michaelwaterman.nl/2022/10/21/managing-sudo-from-active-directory

Michael Waterman October 21, 2022



Welcome to the last of a three part series about Ubuntu and Active Directory. In my previous posts I explained how you could, in just a few steps, join an Ubuntu machine to an Active Directory domain and manage it accordingly. This time I'm addressing centralized management of sudo users. Meaning who can execute commands as sudo on managed Linux desktops (in my case Ubuntu).

Lab Infrastructure

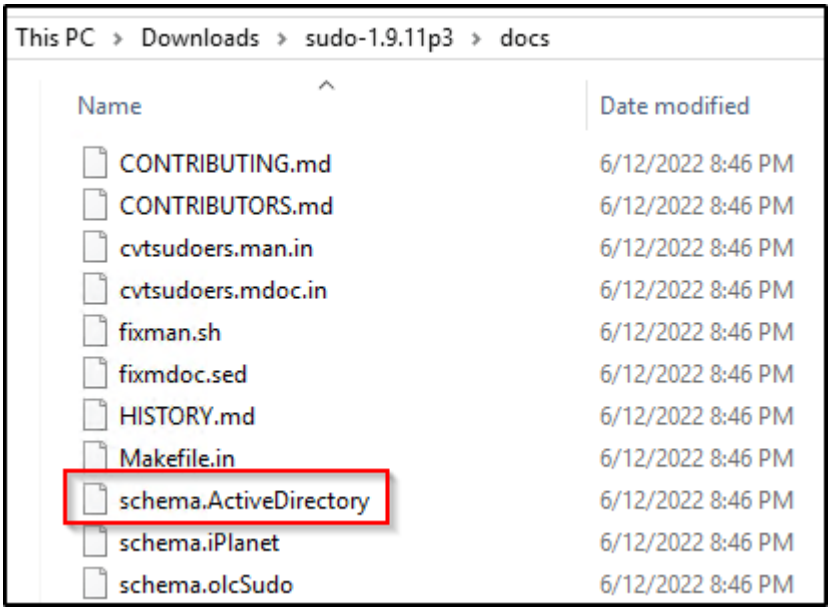
My setup is still the same, I've installed two Windows domain controllers in the domain "*water.lab*" and joined a Ubuntu 22.04 LTS to that domain. Users from that domain can happily login to the Ubuntu workstation using their domain credentials. In this post we'll have to deal with two users:

- Darth Vader – A soon to be root user managed from Active Directory.
- Superuser – A local user with locally configured sudo privileges.

Active Directory Schema Update and Configuration

Our goal will be to configure the domain and Ubuntu desktop in such a way that members of a domain security group will be able to use sudo once logged onto the Ubuntu desktop. The first task at hand is to make Active Directory capable for supporting centralized sudo

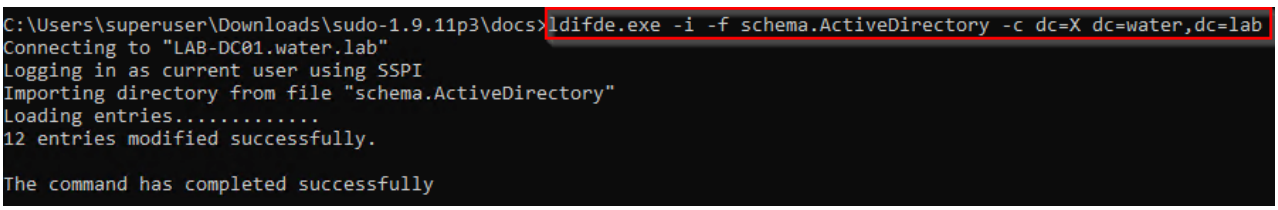
management. Out of the box AD can't be used for managing sudo users because the attributes aren't available in the AD schema. The cool part is that the good people that created and maintain sudo made it very easy for us to extend active directory with the appropriate attributes. First we need to download the latest stable sudo release from the official sudo website, Sudo In my case I've created this blog based on the 1.9.11p3 version, released on June 21, 2022. Once the package is extracted (7Zip Download (7-zip.org) is an excellent client on Windows for that) you end up with a folder that contains the file that we need. Browse to the doc folder.



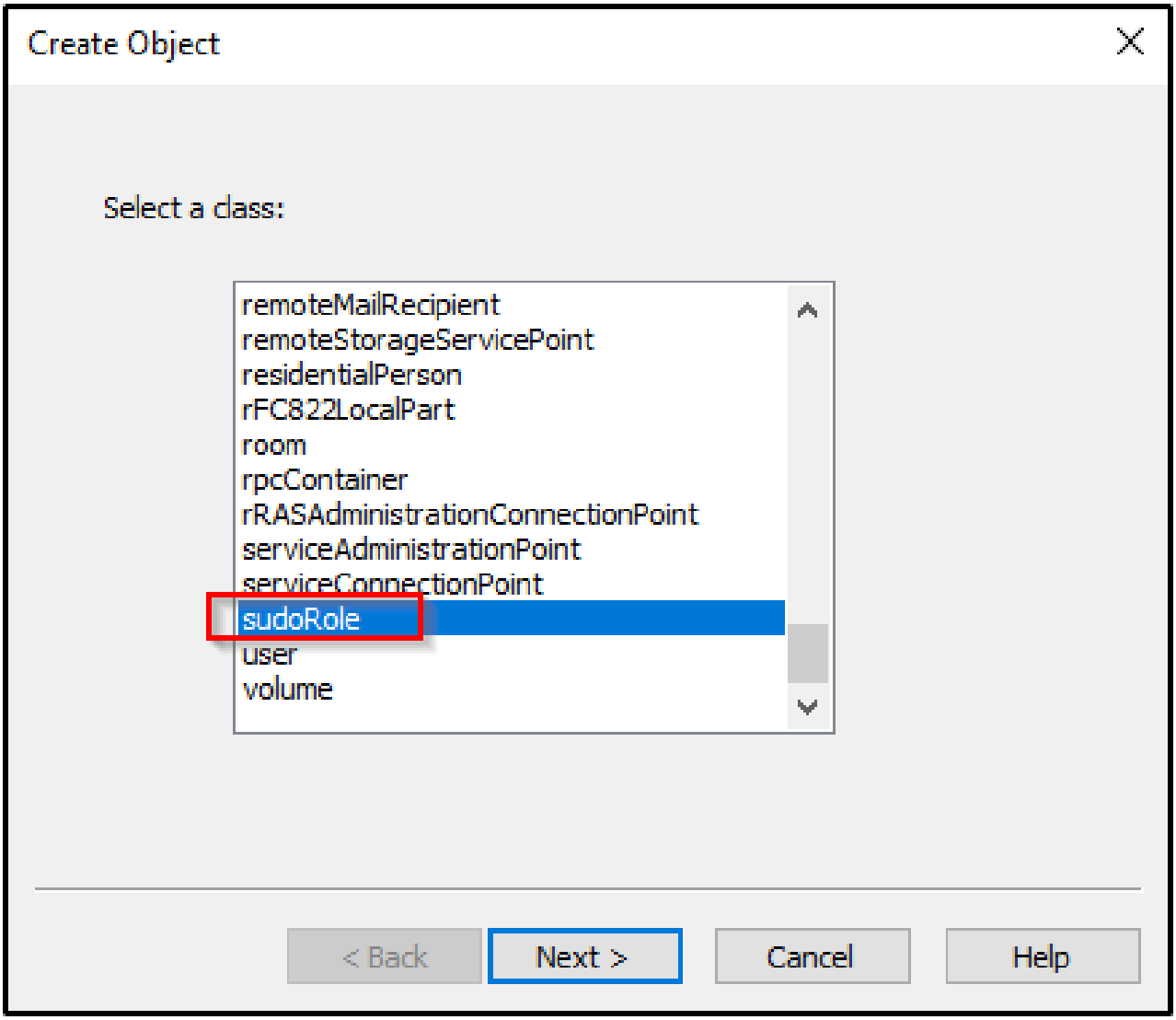
There should be a file named "*schema.ActiveDirectory*". This file is used to extent the Active Directory Schema to include the sudo attributes that we will be using. In your Active Directory environment, login to the domain controller that has the schema master role with an account that has the privileges to extent the schema. In my case I'll just use the all-powerful Administrator account. At a minimum copy the "schema.ActiveDirectory" file to that server and open up an elevated command prompt. In the command-prompt browse to the location where you stored the schema extension file. Execute the following "*ldifde.exe*" command, leave the command-line exactly as is, only replace the latter part of the line to reflect your domain structure

```
ldifde.exe -i -f schema.ActiveDirectory -c dc=X dc=water,dc=lab
```

BAT (Batchfile)



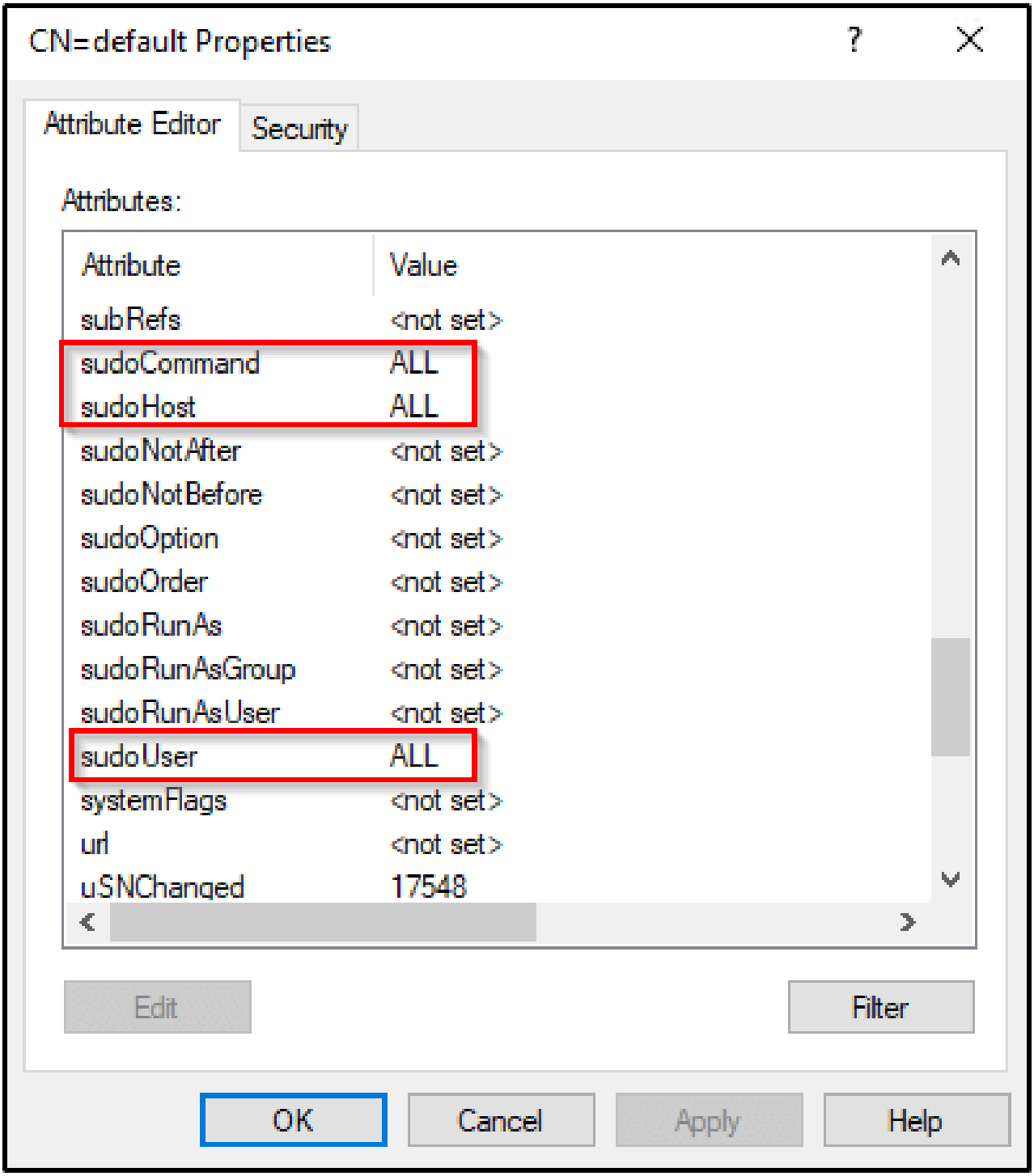
Next, open "*adsiedit.msc*" and connect to the "*Default naming context*". In the root create a new organization unit and give it a name "*sudoers*". This is the default location sudo will look for user defined rules in AD, don't worry, I'll show you later on how you can change that. What we need to do next is create a new object that will contain our attributes. On the organizational unit that you just created, right click and select "*new-object*". In the next window select "*sudoRule*".



If you don't see the object class immediately that probably means that the class hasn't been replicated to your DC yet, just give it some time. In the next screen use the name "*default*". In my experience it really doesn't matter what name you give the object. I tend to use a more descriptive name like, desktops or servers. That makes it a bit more clear on what purpose the object serves. At the last screen click "*Finish*" to create the "*sudoRole*" object. Now we need to edit the attributes of the default object we just created. This way we can configure the behavior of the sudo commands into what can (or cannot) be executed on the Linux host. Select properties on the "*cn=default,ou=sudoers*" object. Just for testing purposes, we're going to enable all sudo privileges on all machines. Edit these attributes:

- sudoCommand: ALL
- sudoHost: ALL

- sudoUser: ALL



Configuring the Ubuntu client

Ubuntu 22.0.4 and above have a enhanced way of applying management and configuring from the domain involving a technology alike Windows with Group policies. Actually the configuration is based on ADMX files. At the time of writing the technology isn't fully released yet. More on that in a later post. This Ubuntu release relies heavily on this new technology and lacks a critical package and configuration for applying sudoer from the domain using the procedure described here. Use the following steps to add the full sudoers support to the machine. Open a shell and type:

```
sudoaptinstalllibsss-sudo-y
```

Bash

Add the following to the "*/etc/nsswitch.conf*", just underneath the line: "*netgroup nis sss*":

```
sudoers:filessss
```

Bash

Add the following to the "*/etc/sssd/sssd.conf*", under the "*[sssd]*" section:

```
service=nss,pam,sudo
```

Bash

Restart the SSSD deamon.

```
sudosystemctlrestartssd
```

Bash

This should do it. Now to test this setup, run the following command, after entering your password you should be having the ability to execute as sudo!

```
sudo-l-Udarth
```

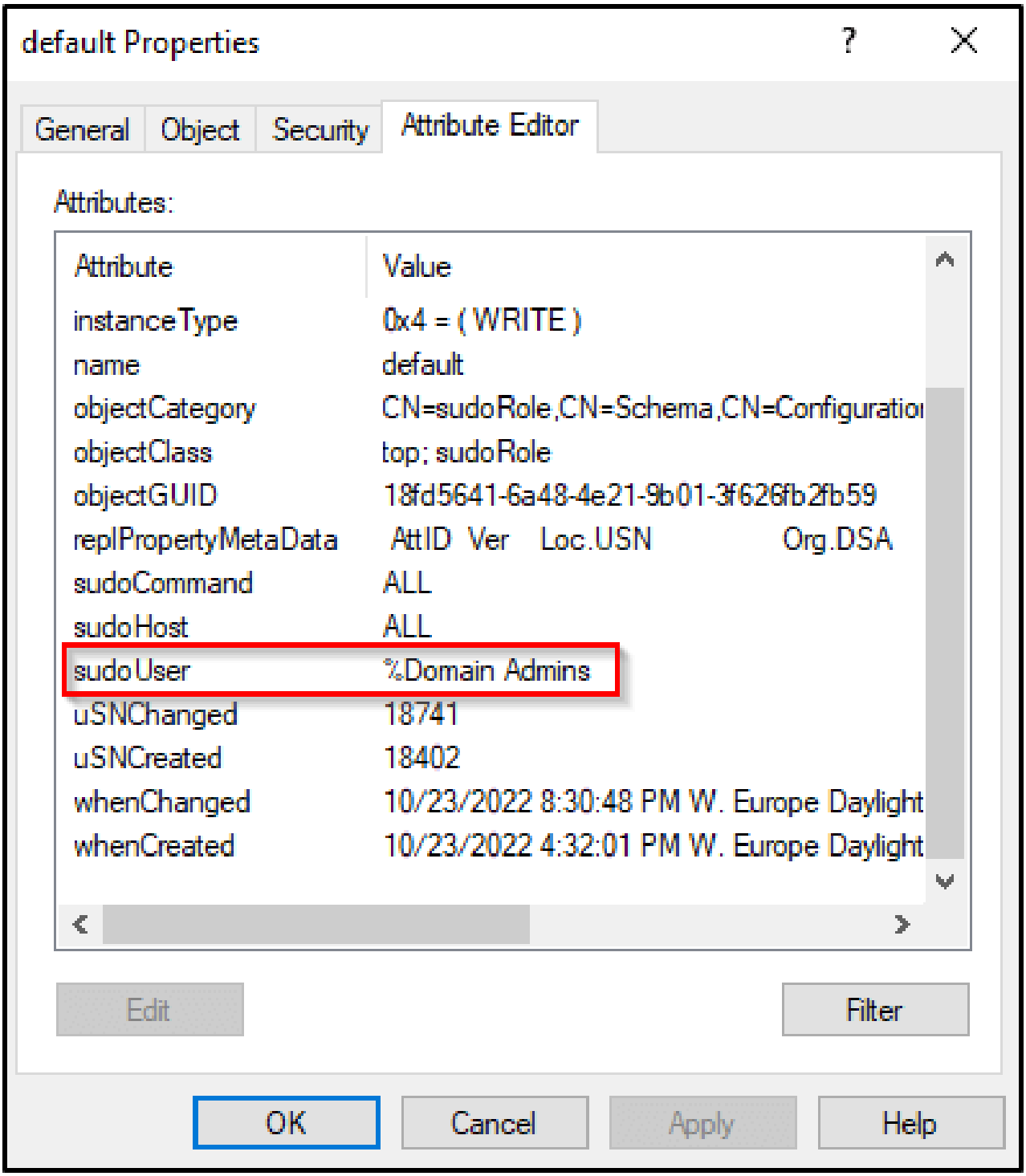
Bash

```
superuser@ubuntu-2204:~$ sudo -l -U darth
Matching Defaults entries for darth on ubuntu-2204:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin,
    use_pty

User darth may run the following commands on ubuntu-2204:
    (root) ALL
superuser@ubuntu-2204:~$
```

Domain Security Groups

In a managed infrastructure of some size it's highly unlikely that you will have many individual users that will be listed on the "sudoRole" object. Instead having a group listed there makes more sense. Instead of a user that can be set by just using the "*sAMAccountName*", a group will need to contain a % sign at the beginning. So suppose we want to copy the behavior of Windows and add the "Domain Admins" to every machine (Which is a terrible idea btw!) the sudoRole attribute would look like this:

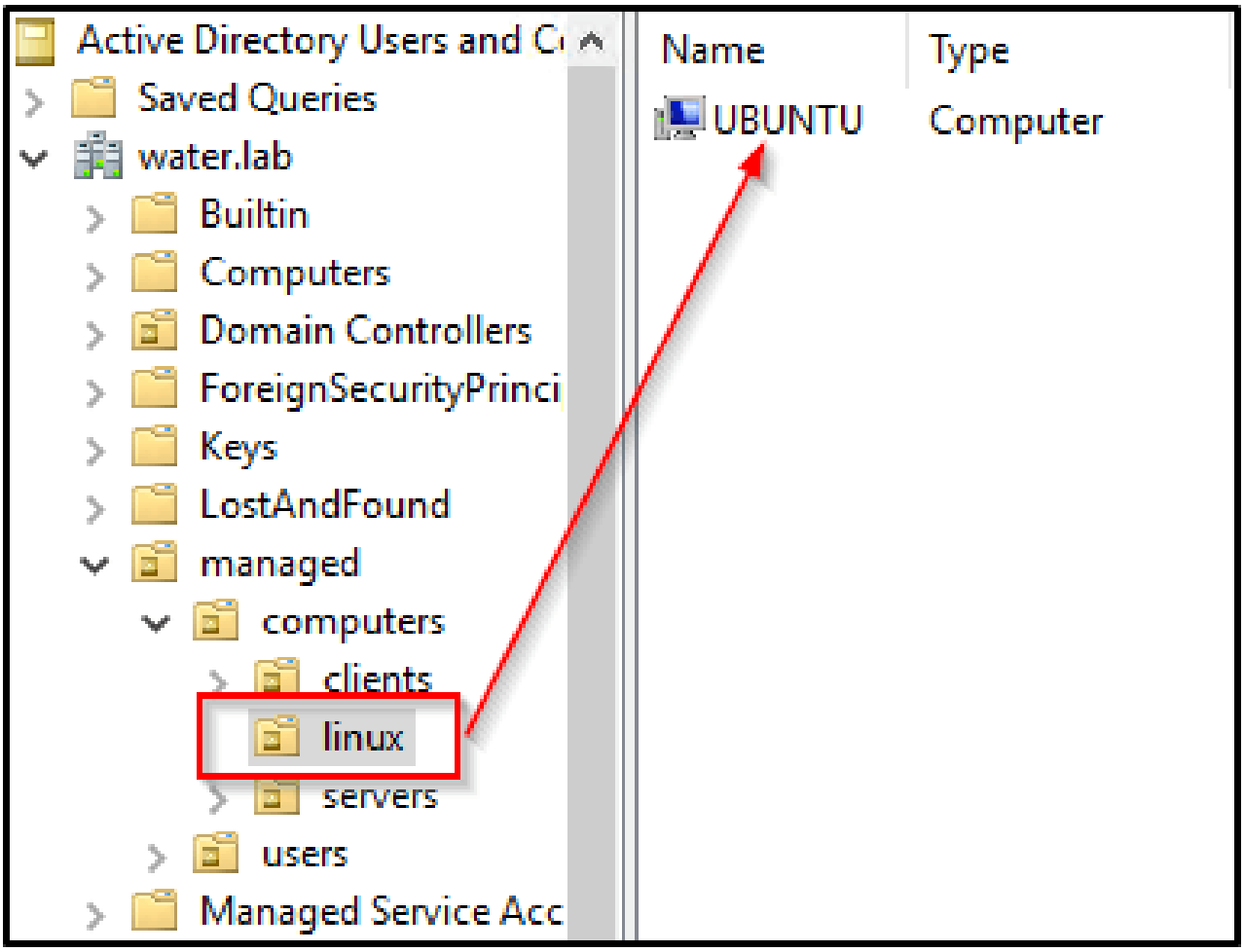


Note! Although you would expect there to be an escape character () after “%Domain” to cope with the space, this seems to be completely handled by the sssd service. Very cool if you ask me!

Note! As you can see there’s also a possibility to use wildcard characters for the attributes, (See sudoHost). There are more options available listed here: [Sudoers Manual | Sudo]

Organizational Units

By default sudo will look for an organizational unit with the name “*sudoers*” in the root of the directory. There is however a way to tell sudo to look elsewhere. For example if you have two OU’s assigned to different groups, you could use that setup to apply different sudo configurations. So let’s assume we use the organizational unit in our lab Active Directory.



The Ubuntu machine in our example belongs to that OU and we want to point it specifically to a new “*sudoRules*” object that resides in the same organizational unit. Actually it’s not hard to do. In the organizational structure I described, create a new “*sudoRole*” object. On the Ubuntu host, open the file “*etc/sssd/sssd.conf*” as sudo. Add the following line under the “*[domain/YourDomain]*“

```
ldap_sudo_search_base=ou=linux,ou=computers,ou=managed,dc=water,dc=lab
```

Bash
Reboot your machine and see the magic happen!

Troubleshooting Tips

Sometimes the caching mechanism from the sssd service is so persistent that the entries for sudo users remain, even after several reboot. One command in particular has helped me to clear the cache is:

```
sudosss_cache-E
```

Bash
If that doesn't work, stop the sssd service and delete the database files in "*/var/lib/sss/db**".

```
sudorm/var/lib/sss/db/*
```

Bash
Just remember that the cache receives a full update every six hours and an incremental one every 15 minutes. But that really doesn't help when the cache is broken. Those values can be altered by setting:

- ldap_sudo_full_refresh_interval=86400
- ldap_sudo_smart_refresh_interval=3600

I hope that these three blog posts on Ubuntu and Active Directory integration was useful to you, for me it was a fun ride figuring it out and gaining the knowledge! As always, you can leave comments, remarks or questions below or send me a direct message.

Reference

[How to integrate Ubuntu Desktop with Active Directory | Ubuntu]