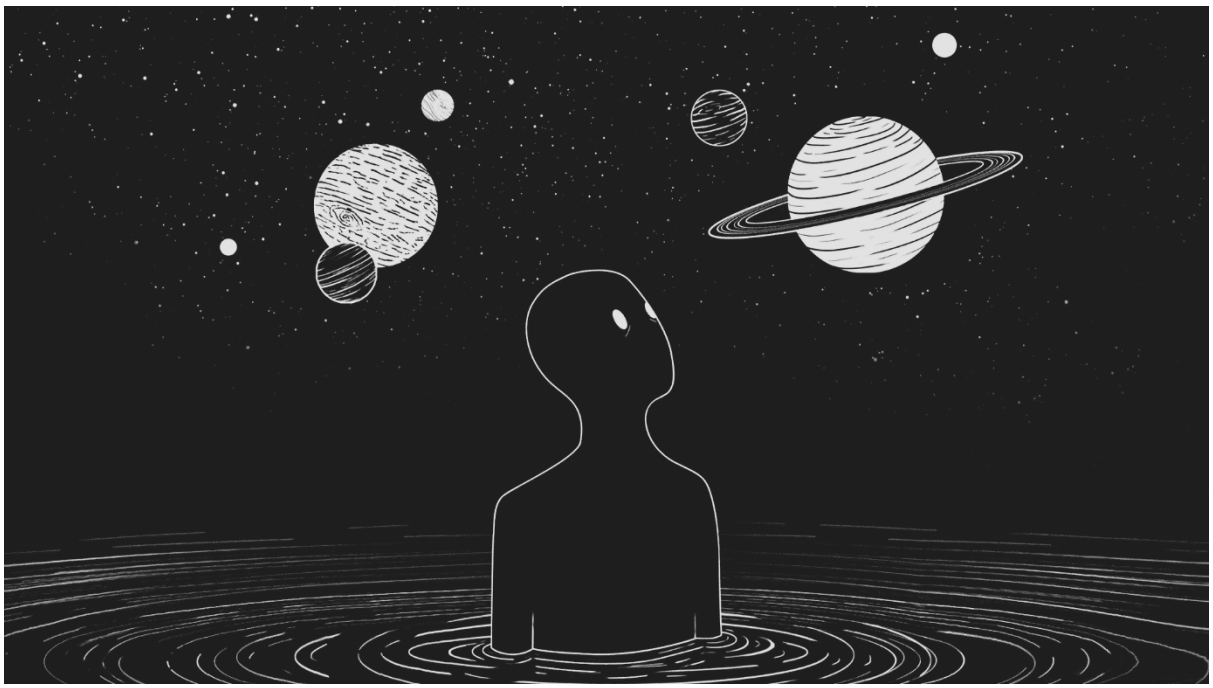
Change & Configuration Management



Appendix Installation and Configuration Documentation

Project Name: Operations Security & Project

Report Written by Victor Poh Hong Rong

Supervised by Mr Sim Xiangyuan

DSF-1802, Team 2

180272P

Team Members:

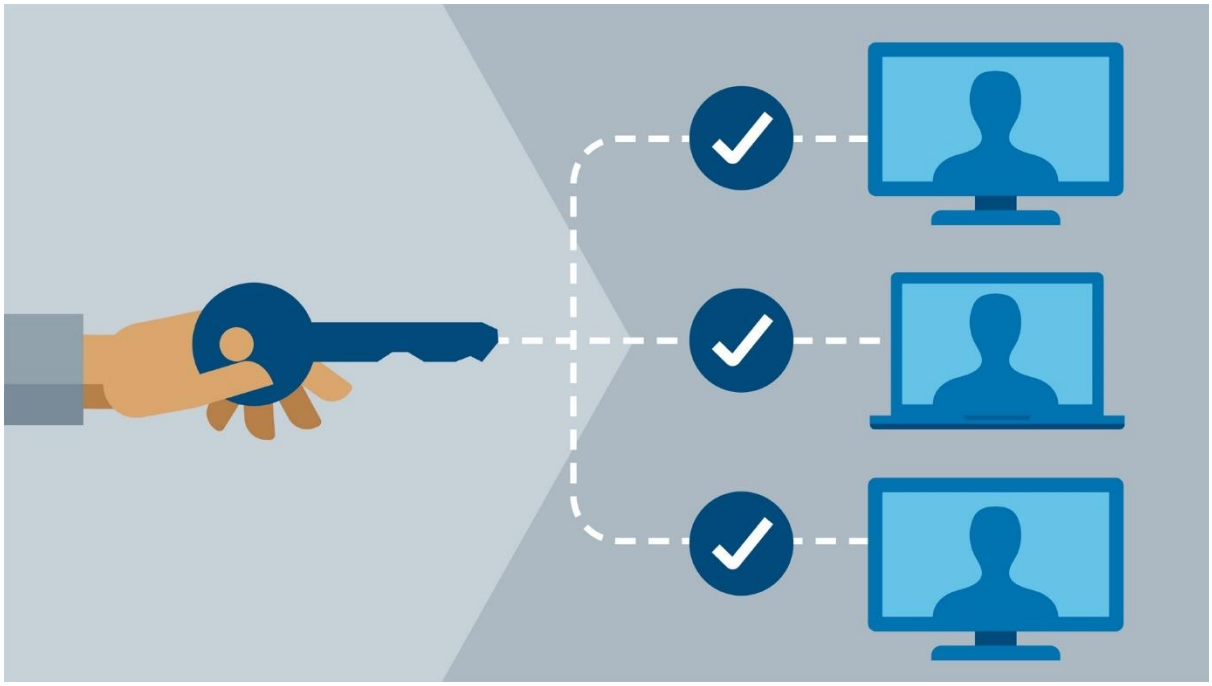
• Lye Jia Jun (Group Leader)	180245D	Continuous Monitoring
• Tan Song Ze Nigel	181391W	BCP & Backup
• Muhammad Noor Bin Saidee	185014T	Network Security
• Victor Poh Hong Rong	180272P	Change & Config Management

Reported last amended: 15/8/2020

Contents

Note:

Since my **main functionality** includes two separate (but independent) parts – the use of "Software Request Form" to install software into multiple servers, and the use of "Policy Request Form" to make changes to the policy in multiple servers, in my following documentation, I will only focus on demonstrating the installation and configuration documentation with the help of "Software Request Form" since both parts ("Software Request Form" and "Policy Request Form" functions very similarly – except that one is installing software into multiple servers while the other is making changes to policy in multiple servers)



I will be splitting my work into different portions throughout my documentation, categorized as:

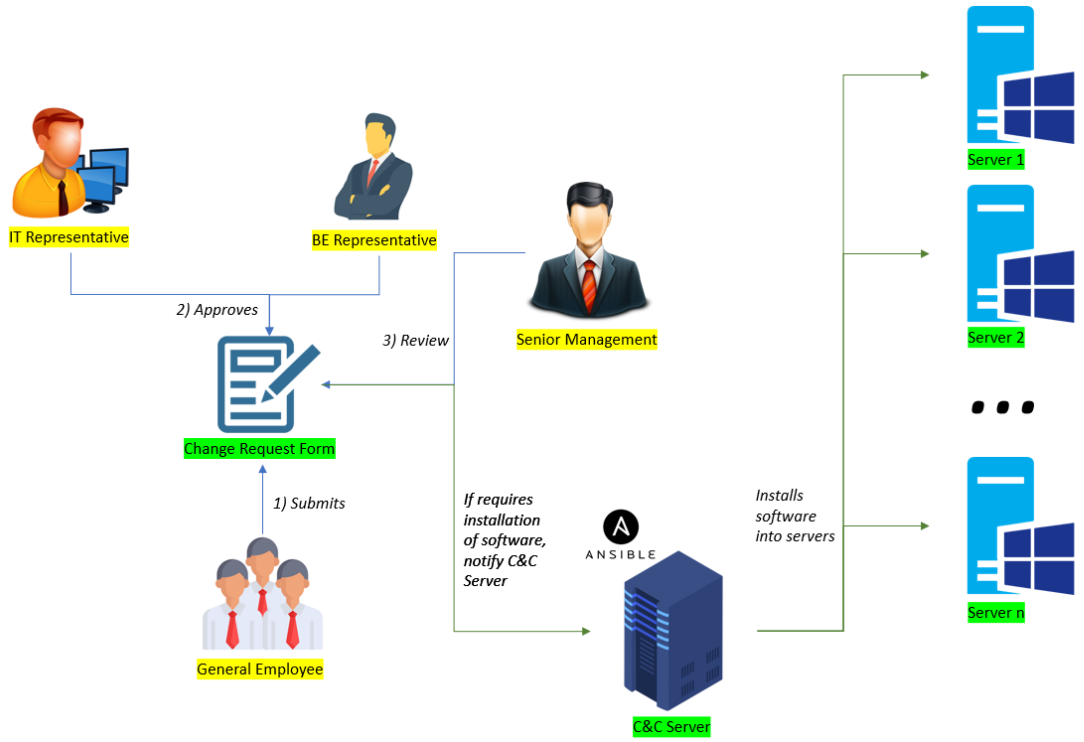
- Main functionality
- Supporting functionality

Main Functionality

Main functionality

- Getting C&C (Change & Configuration) Server with Ansible installed to work with self-coded Change Request Form, with the ultimate purpose of simultaneously configuring multiple servers
- Machines and applications involved:
  - Windows Server 2016 (Victim)
  - Ubuntu 18.04 Ansible Control Machine (Change & Configuration Server)
  - Change Request Form (Software Request Form)

This is how the final product will function:



1. General Employee requests software to be installed in target servers (Server 1, Server 2, …, Server n) using "Software Request Form" which is a category of "Change Request Form".
2. IT representative and BE (Business) representative of the company reviews the form and decide whether or not to approve such a request.
3. Upon approval by both sides, the form is made available for the Senior Management to do a final review and decide whether or not to approve, defer, or reject such a request.
4. Upon approval by senior management, the "Software Request Form" notifies the C&C Server of the names of the Software to be installed.
5. The C&C Server takes note of the names of the Software and installs them respectively in the target servers (Server 1, Server 2, …, Server n)

Main Functionality

Because the **main functionality** requires a lot of setups, I will break it into **two parts** for the Installation Documentation and Configuration Documentation. The two parts are:

a. Connecting Ansible machine to Windows machine
b. Connecting Change Request Form to Ansible machine

This is how the connection diagram works:

We will first look through the Installation Documentation and Configuration Documentation of **connecting Ansible machine to Windows machin**

Connecting Ansible machine to Windows machine

Installation Documentation

1. **Installing Ansible**

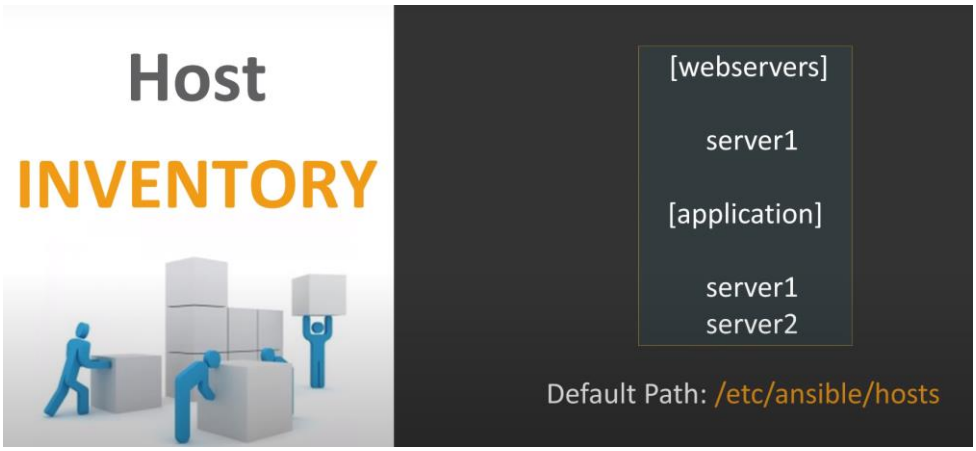
   To begin using Ansible as a means of managing our server infrastructure, we need to install the Ansible software on the C&C machine that will serve as the Ansible control node.

   For installing Ansible, we have to configure PPA on our machine. For this, we have to run the following line of code:

```
$ sudo apt-get update

$sudo apt-get install software-properties-common

$ sudo apt-add-repository ppa:ansible/ansible

$ sudo apt-get update

$ sudo apt-get install ansible
```

2. **Setting Up the Inventory File**

   The *inventory file* contains information about the hosts we'll manage with Ansible. We can include anywhere from one to several hundred servers in our inventory file, and hosts can be organized into groups and subgroups. The inventory file is also often used to set variables that will be valid only for specific hosts or groups, in order to be used within playbooks and templates. A sample syntax of the inventory file is as shown below:



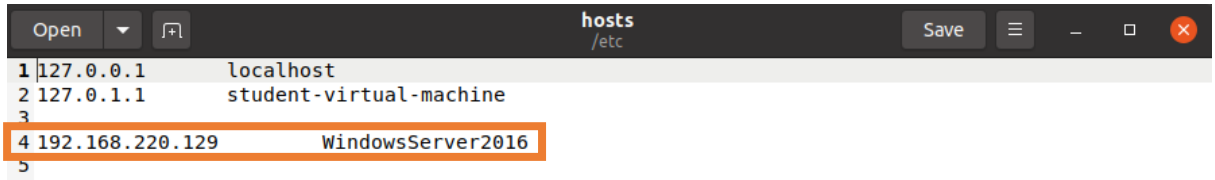
Source: https://www.youtube.com/watch?v=4nKW2eF-nIw&t=678s

To edit the contents of our default Ansible inventory, open the /etc/ansible/hosts file using the text editor of choice, on our Ansible Control Node:

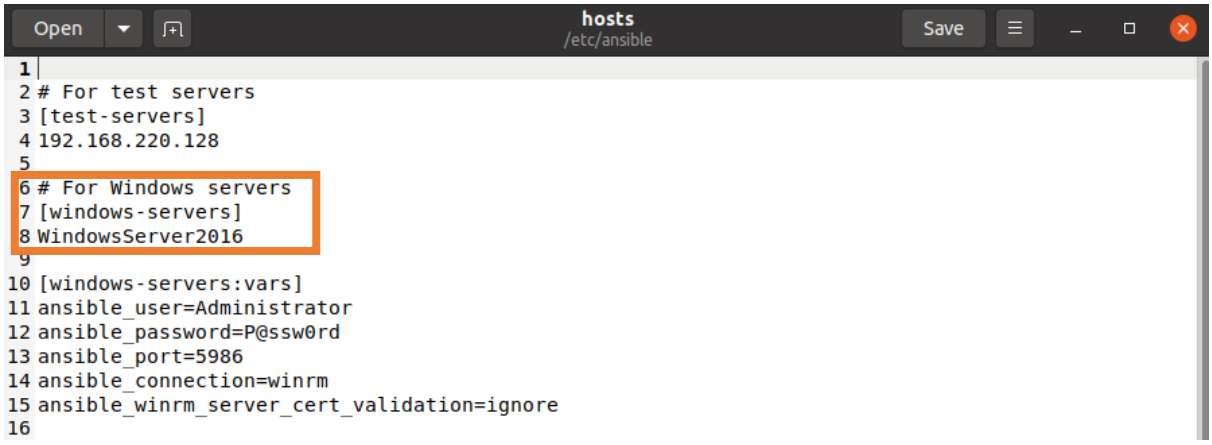
```
$ sudo gedit /etc/ansible/hosts &
```

Next, add the IP address or hostname of the machine(s) that we want to connect to and configure. For my project, I am using the hostname "WindowsServer2016" which I already did an IP-hostname mapping in my /etc/hosts file

/etc/hosts:



/etc/ansible/hosts



3. Testing Connection

After setting up the inventory file to include our servers, it's time to check if Ansible is able to connect to these servers and run commands via SSH.
However, because Windows doesn't support SSH by default, we need to use alternative way to connect Ansible with the Windows server(s).
Through online research, I found out that Ansible supports winrm protocol, which can be used to connect Ansible with the Windows server(s).

Testing Connection Step (1) – Installing winrm service into Windows server

The following steps on how to install winrm service into Windows server is solely based on this tutorial video, which can be found here
https://www.youtube.com/watch?v=SSaEFGlnqYY&t=188s
and installation of python pip and winrm can be found here
https://geekflare.com/connecting-windows-ansible-from-ubuntu/

On the Windows server, we need to upgrade PowerShell to PowerShell 3.0/4.0/5.1 and ASPNET version 4.5.2, to achieve this, we need to use the PowerShell script. The script can be found https://github.com/vipin-k/Ansible-Windows/blob/master/Upgrading%20PowerShell%20and%20.NET%20Framework.ps1

```
$url = https://raw.githubusercontent.com/jborean93/ansible-windows/master/scripts/Upgrade-Powershell.ps1

$file = "$env:temp\Upgrade-Powershell.ps1"

$username = "<username>"
$password = "<password>"

(New-Object -TypeName System.Net.WebClient).DownloadFile($url, $file)
Set-ExecutionPolicy -ExecutionPolicy Unrestricted -Force

#version can be 3.0, 4.0 or 5.1
&$file -Version 5.1 -Username $username -Password $password -Verbose
```

Next, we want to run a PowerShell script to enable the winrm service. You may obtain the script here
https://github.com/vipin-k/Ansible-Windows/blob/master/ConfigureRemotingForAnsible.ps1

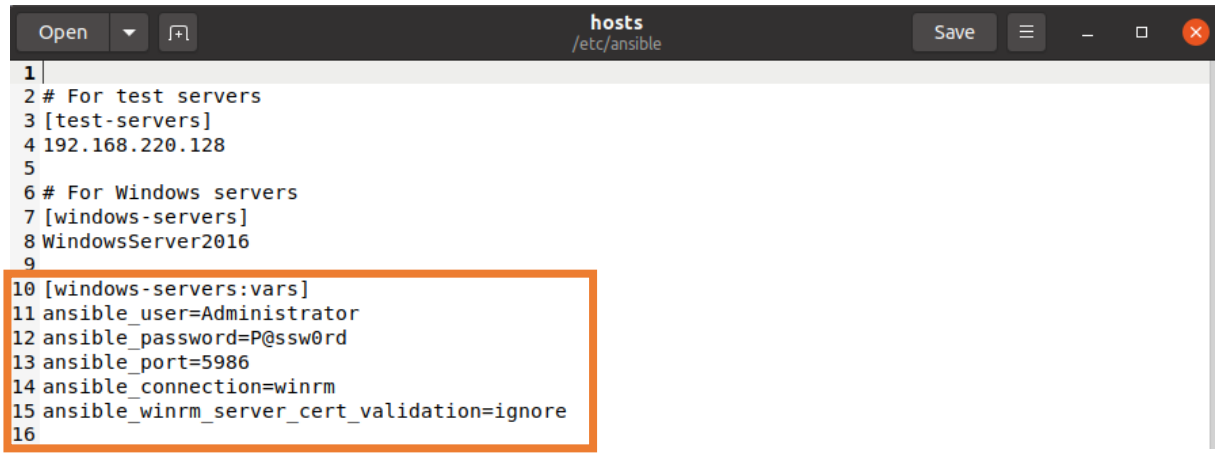
We have successfully installed winrm service into our Windows server!

Testing Connection Step (2) – Installing python-pip & python-winrm pacakages into Ansible Machine

Now, we are required to install the python-pip package into the Ansible machine because the Ansible machine uses the py-winrm package to communicate and manage the Windows machine.

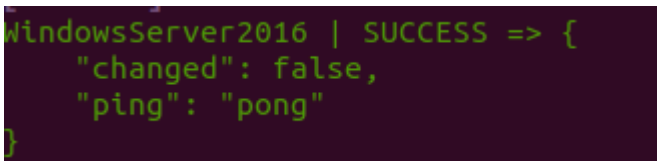
```
$ sudo apt-get update

$ sudo apt-get install gcc python-dev

$ sudo apt-get install python3-pip

$ sudo apt-get install python3-winrm
```

Next, we need to setup the parameters for Ansible to connect to the Windows server. In the /etc/ansible/hosts file, add the following parameters:



Finally, test the connection by running the following command

```
$ ansible windows-servers -m win_ping
```



We have successfully established connection between the Ansible control machine and Windows server using winrm service!

Conclusion

1. Installing Ansible
2. Setting up the Inventory File
3. Testing Connection
   - Installing winrm service into Windows server
   - Installing python-pip & python-winrm packages into Ansible machine

References

1. How to Install and Configure Ansible on Ubuntu 18.04
   https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-ansible-on-ubuntu-18-04#how-does-ansible-work

2. What Is Ansible? | Ansible Tutorial For Beginners | DevOps Tools | DevOps Training | Edureka
   https://www.youtube.com/watch?v=4nKW2eF-nIw&t=678s

3. How to Manage Windows Machine with Ansible
   https://www.youtube.com/watch?v=SSaEFGlnqYY&t=188s

4. How to Connect Ansible on Windows from Ubuntu?
   https://geekflare.com/connecting-windows-ansible-from-ubuntu/

5. Ansible Windows/Upgrading PowerShell and .NET Framework.ps1
   https://github.com/vipin-k/Ansible-Windows/blob/master/Upgrading%20PowerShell%20and%20.NET%20Framework.ps1

6. ConfigureRemotingForAnsible.ps1
   https://github.com/vipin-k/Ansible-Windows/blob/master/ConfigureRemotingForAnsible.ps1

Configuration Documentation

1. Obtain the Default Configurations of the Windows Server(s)

We can obtain the configurations (inventory) of the Windows Server(s) and store it into a file by running the following command on the Ansible machine

```
$ ansible windows-servers -m setup >>
/home/student/ansible/configurations/WindowsServers.txt
```

A sample portion of the output:

```
WindowsServer2016 | SUCCESS => {
    "ansible_facts": {
        "ansible_architecture": "64-bit",
        "ansible_bios_date": "07/28/2019",
        "ansible_bios_version": "6.00",
        "ansible_date_time": {
            "date": "2020-07-02",
            "day": "02",
            "epoch": "1593733981.01297",
            "hour": "23",
            "iso8601": "2020-07-03T06:53:01Z",
            "iso8601_basic": "20200702T235301002960",
            "iso8601_basic_short": "20200702T235301",
            "iso8601_micro": "2020-07-03T06:53:01.002960Z",
            "minute": "53",
            "month": "07",
            "second": "01",
            "time": "23:53:01",
            "tz": "Pacific Standard Time",
            "tz_offset": "-07:00",
            "weekday": "Thursday",
            "weekday_number": "4",
            "weeknumber": "26",
            "year": "2020"
        },
        "ansible_distribution": "Microsoft Windows Server 2016 Standard Evaluation",
        "ansible_distribution_major_version": "10",
        "ansible_distribution_version": "10.0.14393.0",
        "ansible_domain": "hq.ospjserver.org",
```

2. Automate this process of gathering configuration (inventory) information about the Windows server using cron

<u>Part 1 – Install and enable cron</u>

```
$ sudo apt-get update

$ sudo apt-get install cron
```

```
$ sudo systemctl enable cron
```

Part 2 – Setup automation script

In this case, we are saving the configuration information of all Windows Servers into the "WindowsServers.txt" file

```
#!/bin/bash
echo "Gathering configurations for all Windows Servers"
ansible windows-servers -m setup >> /home/student/ansible/configurations/WindowsServers.txt
```

Assign "execute" permission for this script

```
$ chmod +x /home/student/ansible/configurations/WindowsServers.txt
```

Part 3 – Schedule this script to run at a specific date/time with cron

Run the command to access the crontab

```
$ crontab -e
```

Tasks are scheduled in crontab in a structure like this:

minute	hour	day_of_month	month	day_of_week	command_to_run

The following instruction will run the command contained in the script "/home/student/ansible/configurations/gatherWindowsInformation"
$ ansible windows-servers -m setup >> /home/student/ansible/configurations/WindowsServers.txt
every Tuesday at 3:00PM

```
student@AnsibleControlMachine: ~/ansible/configurations
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
SHELL = /bin/bash
0 15 * * 2 /home/student/ansible/configurations/gatherWindowsInformation
```

Conclusion

1. Obtain the Default Configurations of the Windows Server(s)
2. Automate this process of gathering configuration information about the Windows server using cron
   - Install and enable cron
   - Setup automation script
   - Schedule this script to run at specific date/time using cron

References

1. How to Schedule Tasks on Linux: An Introduction to Crontab Files
   https://www.howtogeek.com/101288/how-to-schedule-tasks-on-linux-an-introduction-to-crontab-files/

2. How to use Cron to Automate Tasks on Ubuntu 18.04
   https://www.digitalocean.com/community/tutorials/how-to-use-cron-to-automate-tasks-ubuntu-1804#installing-cron

Now, we will look through the Installation Documentation and Configuration Documentation of **connecting self-coded Change Request Form with Ansible machine**.

Connecting Change Request Form with Ansible machine

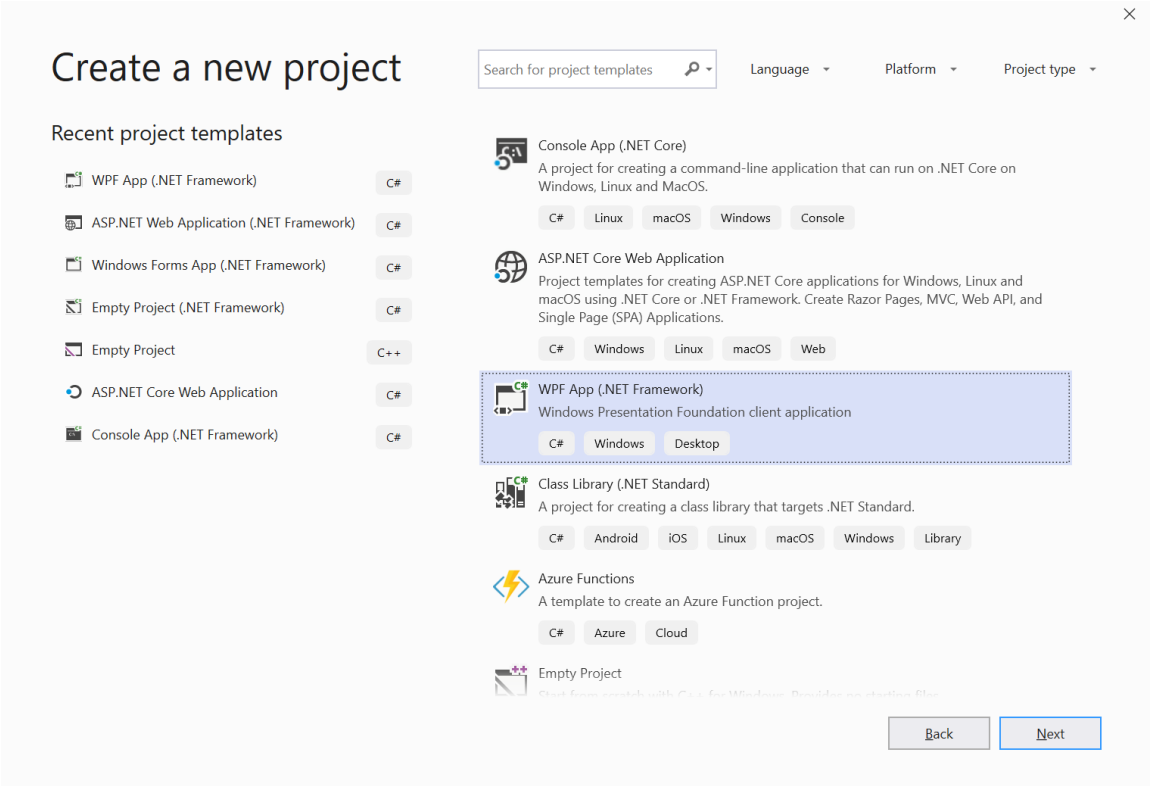
Installation & Configuration Documentation

1. **Create a new, empty project for the Change Request Form**

   In Microsoft Visual Studio 2019 -> Create a New Project -> WPF App (.Net Framework)

   WPF stands for Windows Presentation Foundation, which is a free and open-source graphical subsystem (similar to WinForms) originally developed by Microsoft for rendering user interfaces in Windows-based applications. WPF, previously known as "Avalon", was initially released as part of .NET Framework 3.0 in 2006.
   For more information, please visit:
   https://www.google.com/search?sxsrf=ALeKk03ICNsLIPJrmEFI7674olXUaw6-0Q%3A1594892750291&source=hp&ei=ziEQX9HKD9CR9QOTpKvgBQ&q=what+does+wpf+stand+for&oq=what+does+wpf+stand&gs_lcp=CgZwc3ktYWIQAxgAMgIIADIGCAAQFhAeMgYIABAWEB46BAgjECc6BAgAEEM6BQgAELEDOgcIABAUEIcCOggIABCxAxCDAToECAAQClClBliZHWDQImgAcAB4AIABdYgB1wuSAQQxNS40mAEAoAEBqgEHZ3dzLXdpeg&sclient=psy-ab

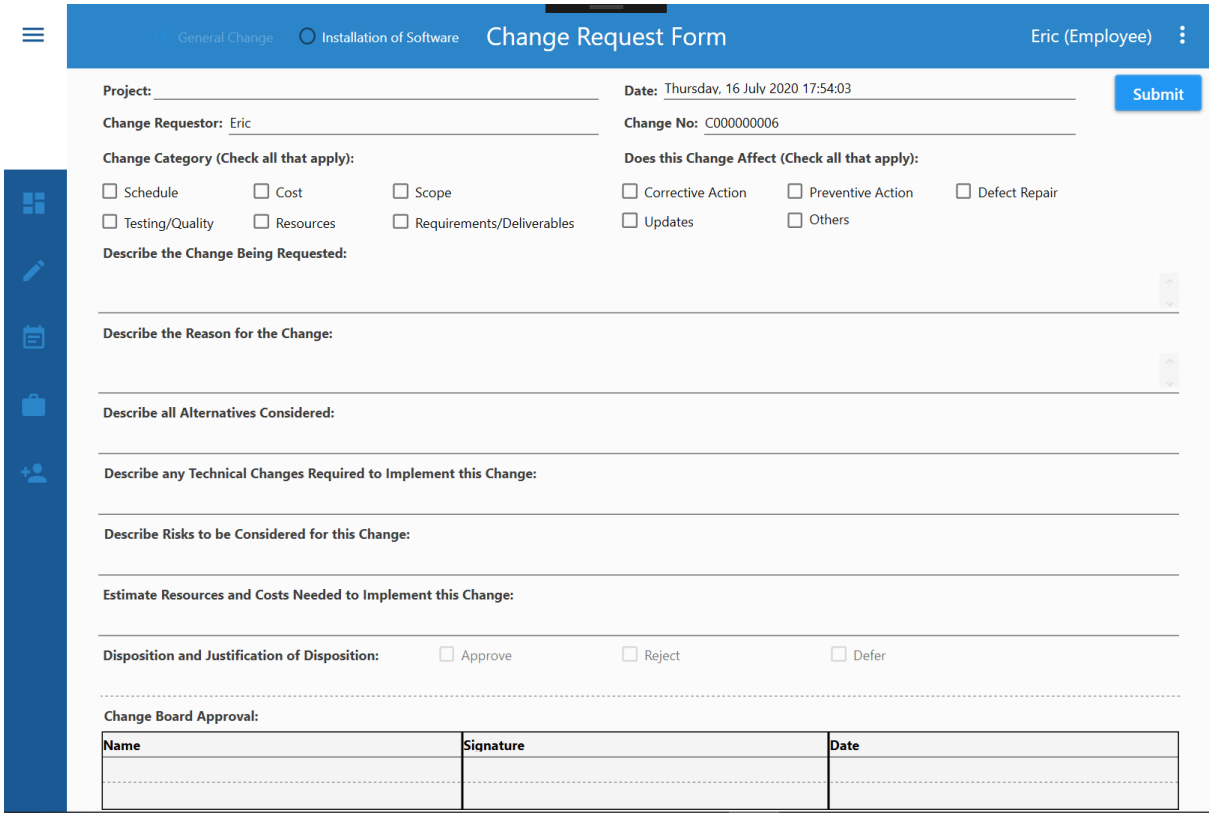
I am using “WPF App (.NET Framework)” C#-coded because it is newer and thereby more in tune with current standards and it’s more flexible. XAML makes it easy to create and edit the GUI (Graphical User Interface), and allows the work to be split between a designer (XAML) and a programmer (C#). In addition, WPF uses hardware acceleration for drawing the GUI, for better performance.
For more information, please visit: https://www.wpf-tutorial.com/about-wpf/wpf-vs-winforms/



2. Design the Change Request Form

Form (1) – General Change Request (Which can be any type of change that an employee wants to request for)

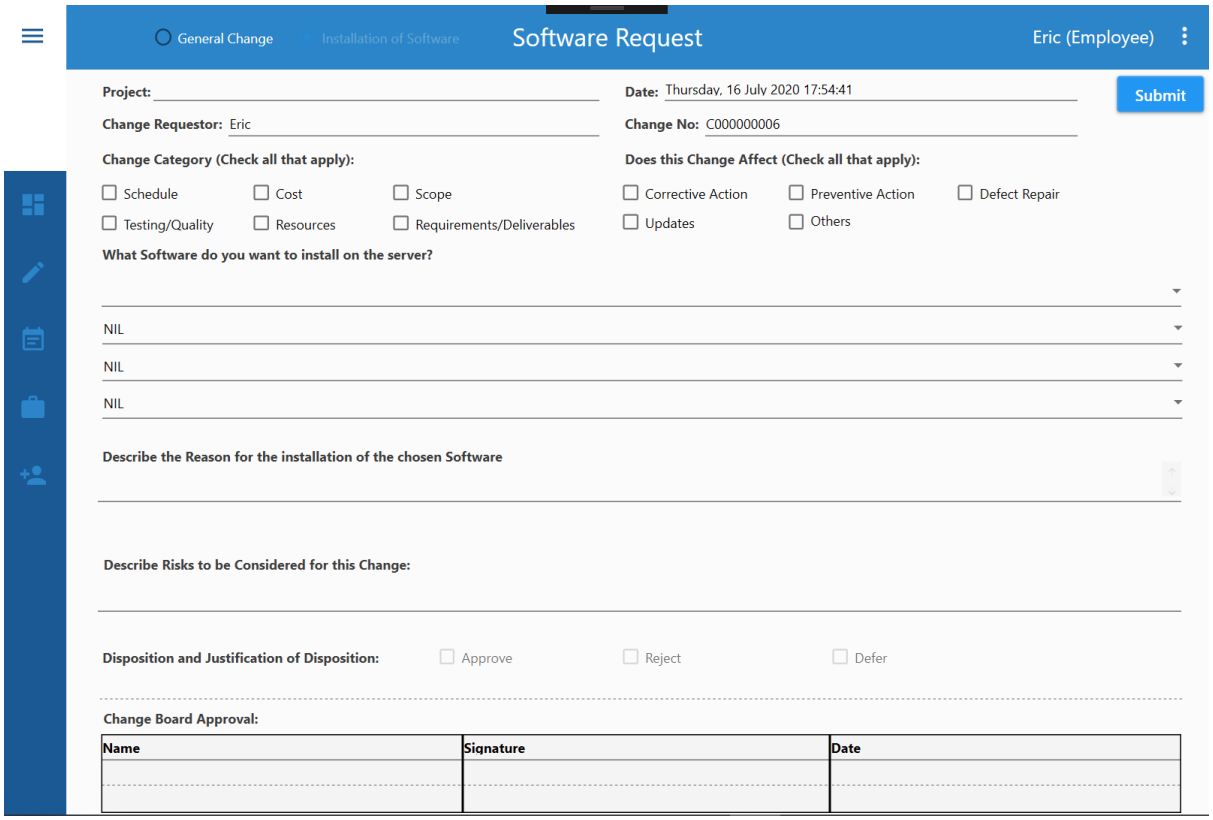
Consider risks, resources and cost, which are essential considerations, before requesting change



Form (2) – Software Request (Request for installation of software into servers)

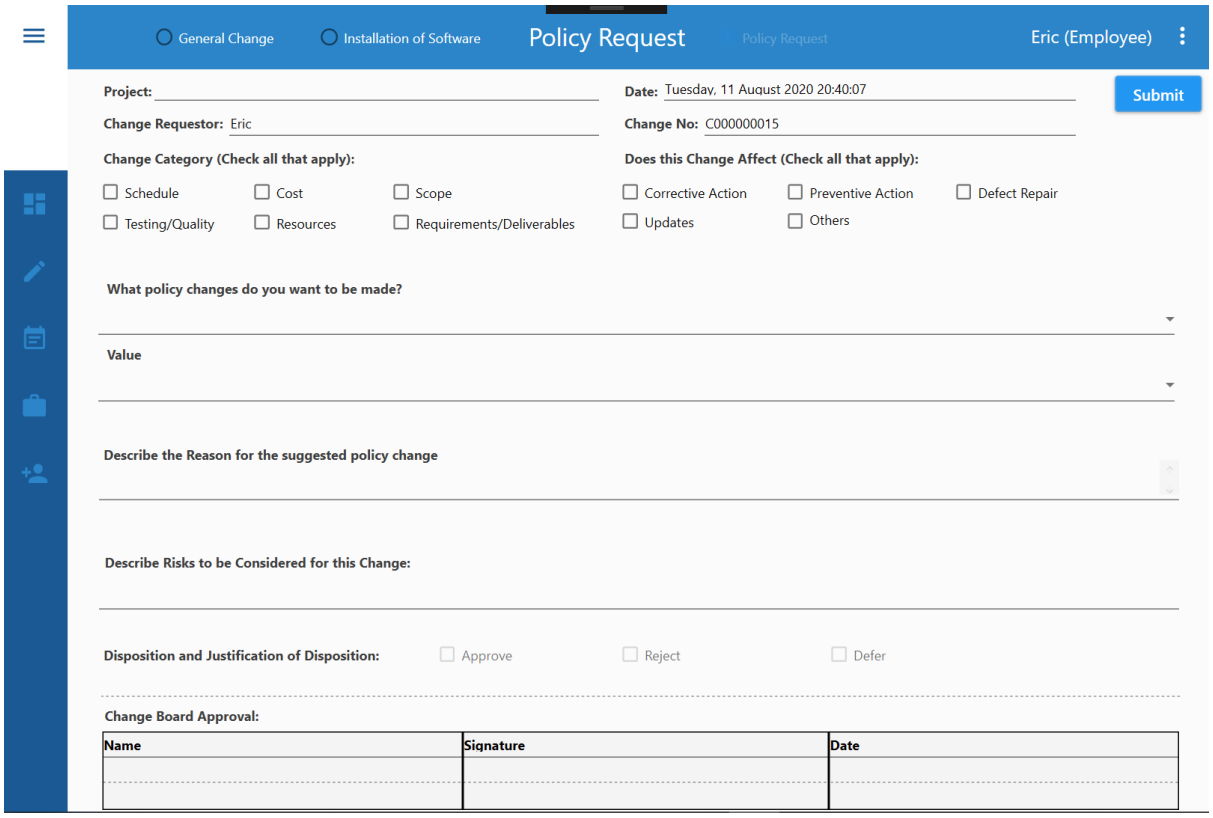
Can request for min. 1 – max. 4 software installation in a single form.

Risks also need to be considered before requesting the change



Form (3) – Policy Request (Request for policy changes to multiple servers)

Risks to be considered before requesting the change.



3. Setup database with necessary tables and columns

To use the change request form, users (username, password) would need to login first.

We would also need separation of duties for different users, we can do this by creating roles
(isAdmin, isIT, isBE, isSM, default: Employee) and then grouping the users together.

dbo.Users

Results | Messages

	Username	Password	isAdmin	isIT	isBE	isSM
1	Jackson	iXAUr1HnTlua4iaBv/s5hDCGqQWaK2YXnIppEOcuZj1lChhWDIK8...	0	0	1	0
2	John	wkX/mp9j7QmcpBqFWVWlkv8MR0oe4q2aG/a/sZIfgtQWAHmvAC...	1	0	0	0
3	Jack	4/E+VUAKZqLwOmF+LrkgHSJ9h1Iiu5U1BQiUxHiHiBnx8geQWBg...	0	1	0	0
4	Jason	0fzwZ3XDRUH/afZoGn4YYFO7QriKqSuLZjo/7jABuVyGHL+ia31b...	0	1	0	0
5	Jacob	YP0r4xwLQE2klpBwzGW5uf1BTWwo8/5gAz6WPn6iQmIv/tYAKUIl...	0	0	1	0
6	Ben	4GXMWU+C01zx/wksgJpLLdyKUD2YqLu4VdG09ZbbEEbbQpiW...	0	0	0	1
7	Benson	935LPboBTMf8j0ucDJXM/kZEKSWBPAzmAuiHDNyOy4M0odvgH...	0	0	0	1
8	Eric	03EGUni3vjTrX31qQI98iHath7rOByyq9ykwJ5QRsI5VtX1tZFtrkgvF...	0	0	0	0

Salts are used to safeguard passwords in storage (e.g. from dictionary/rainbow table attacks).

Historically a password was stored in plaintext on a system, but over time additional safeguards were developed to protect a user's password against being read from the system. A salt is one of those methods.

A new salt is randomly generated for each password.

dbo.Salt

Results | Messages

	Username	Salt
1	John	PE4szuy11aw=
2	Jack	Lz7PegjcOUQ=
3	Jackson	JiAzTZCFF6s=
4	Jason	4zPThGw8yoA=
5	Jacob	ZVqCovk8UvA=
6	Ben	1CAfD/sRReQ=
7	Benson	sRt0koEqjUo=
8	Eric	fXfauhpGBhk=

Remember how we designed our change request form for all three "General Change Request", "Software Request" and "Policy Request"? There are many fields to be filled in… To save these fields, we would need to allocate a database table for each of them.

Top fields:

- ChangeNo
- ChangeRequester
- Project
- Date

Checkbox fields:

- isSchedule
- isCost
- isScope
- isTesting
- isResources
- isRequirements
- isCorrectiveAction
- isPreventiveAction

- isUpdates
- isOthers

Reasons to support change fields:
- ChangeDescription
- ChangeReason
- Alternatives
- TechnicalChanges
- Risks
- ResourcesAndCosts

Acceptance/Rejection fields (for senior management):
- isApprove
- isReject
- isDefer
- Justification

Change Board information (for IT representative and BE representative) who reviewed the change fields:
- CBName1
- CBSignature1
- CBDate1
- CBName2
- CBSignature2
- CBDate2

Decision making field to check which type of change (General Change Request, Software Request, or Policy Request) field:
- isInstall
- isPolicy

Software installation request (only for Software Request change form) fields:
- Software1
- Software2
- Software3
- Software4

Policy changes request (only for Policy Request change form) fields:
- PolicyRequested
- PolicyValue

dbo.Forms

Results | Messages

	ChangeNo	ChangeRequester	Project	Date	isSchedule	isCost	isScope	isTesting	isResources
1	C000000001	Jack	OSPJ	Sunday, 10 May 2020 16:49:43	0	0	0	0	0
2	C000000002	Eric	OSPJ2	Thursday, 25 June 2020 20:08:22	1	1	1	1	1
3	C000000003	Eric	OSPJ3	Sunday, 28 June 2020 20:04:40	1	1	0	1	0
4	C000000004	Eric	OSPJ4	Sunday, 28 June 2020 20:17:28	1	0	0	0	0
5	C000000005	Eric	OSPJ	Friday, 3 July 2020 14:39:33	1	0	0	0	0

isRequirements	isCorrectiveAction	isPreventiveAction	isDefectRepair	isUpdates	isOthers	ChangeDescription	ChangeReason
1	1	0	0	0	0	CHANGE	REASON
1	1	1	1	1	1	Chg2	Reason2
0	1	1	1	0	0	Installation of Software	Reason2
0	0	0	0	1	0	Installation of Software	reason4
0	0	0	0	0	0	Installation of Software	R

Alternatives	TechnicalChanges	Risks	ResourcesAndCosts	isApprove	isReject	isDefer	CBName1	CBSignature1
ALT	TCHANGES	RISKS	RESCOSTS	1	0	0	Jackson	BE Representative
Alt2	Tech2	Risk2	Est2	0	1	0	Jackson	BE Representative
NIL	NIL	Risks	NIL	1	0	0	Jacob	BE Representative
NIL	NIL	risks4	NIL	1	0	0	Jackson	BE Representative
NIL	NIL	R	NIL	1	0	0	Jackson	BE Representative

CBDate1	CBName2	CBSignature2	CBDate2	Justification	isInstall	Software1
Thursday, 25 June 2020 18:55:24	Jack	IT Representative	Sunday, 10 May 2020 23:48:15	<empty>	0	NIL
Thursday, 25 June 2020 20:10:06	Jack	IT Representative	Thursday, 25 June 2020 20:10:46	<empty>	0	NIL
Sunday, 28 June 2020 20:05:28	Jason	IT Representative	Sunday, 28 June 2020 20:05:48	<empty>	1	Atom
Sunday, 28 June 2020 20:18:39	Jack	IT Representative	Sunday, 28 June 2020 20:18:16	<empty>	1	Vim
Friday, 3 July 2020 14:41:28	Jack	IT Representative	Friday, 3 July 2020 14:41:10	<empty>	1	PuTTY

Software2	Software3	Software4	PolicyRequested	PolicyValue	isPolicy
NIL	NIL	NIL	NIL	NIL	0
NIL	NIL	NIL	NIL	NIL	0
NIL	NIL	NIL	NIL	NIL	0
PuTTY	NIL	NIL	NIL	NIL	0
NIL	NIL	NIL	NIL	NIL	0

Connect the database with the C# application

Under C# application “App.config” add the following line of codes:

```
<connectionStrings>
   <add name="CMFormDB" connectionString="Data Source=LAPTOP-
         JU2SBIE0\SQLEXPRESS;Initial Catalog=CMForm;User
         ID=<username>;Password=<password>;Connect
         Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationIntent=R
         eadWrite;MultiSubnetFailover=False"/>
</connectionStrings>
```

Assign DB variable “connString” whenever db connections and operations are needed later on:

```
string connString = ConfigurationManager.ConnectionStrings["CMFormDB"].ToString();
```

And we are done setting up our database with the necessary tables and fields as well as establishing connection between the database and C# application!

4. Code the necessary functionalities for Change Request Form to work with Ansible machine

We would need the basics of DB operations such as inserting, updating of rows, etc. and the basics of C# coding to get the form to work, but this is not the purpose of this section. This section's purpose is to explain how to get the Change Request Form to work with Ansible machine. Therefore, I will not be elaborating the basics of C# coding and DB operations in this section, and I will skip to the portion of getting the Change Request Form to work with Ansible machine.

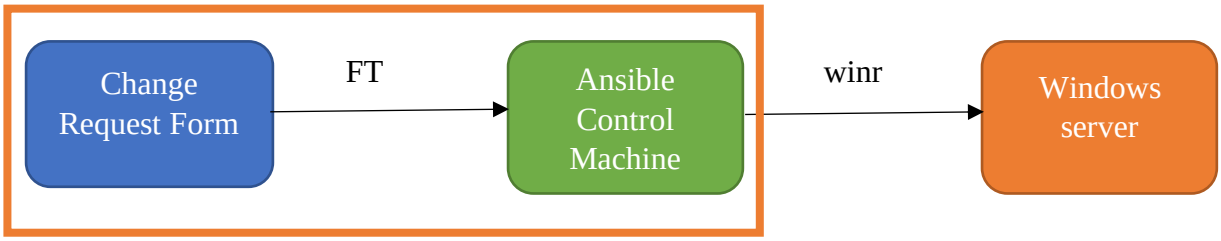
We want to code the Software Request form such that each time the Software Request form is approved by the Senior Management, it will trigger a sequence of events as follows:

1) Establish connection to the Ansible Control Machine (ACM)
2) Notify the ACM that there is/are new software going to be installed on the target server(s), and pass the software names to the ACM so that the ACM knows which software needs to be installed on the target server(s), and then install the software into the target server(s)

In our next steps, I will be explaining how to conduct the above process by configuring the Ansible machine to follow such a process, and by embedding the necessary codes into our Change Request Form.

1) Establish connection to the Ansible Control Machine (ACM)

Connection between the Change Request Form and Ansible machine can be made using FTP connection.



<u>Part 1 – Install and setup FTP server at the ACM</u>

Step 1 – Install and start up FTP server at the ACM

Run the following commands to install and start up FTP server.
$ sudo apt-get update

```
$ sudo apt-get install vsftpd

$ sudo systemctl start vsftpd

$ sudo systemctl enable vsftpd
```

Step 2 – Change Default FTP Directory (optional)

By default, the FTP server uses the /srv/ftp directory as the default directory. We can change this by creating a new directory and changing the FTP user home directory.

For my project, I am changing the default directory to /home/student/AnsiblePublish

```
$ sudo mkdir /home/student/AnsiblePublish

$ sudo usermod -d /home/student/AnsiblePublish ftp

$ sudo systemctl restart vsftpd
```

Step 3 – Allow uploading of files

Run the following commands and make the necessary changes to the vsftpd.conf file to allow authenticated users to upload files

```
$ sudo nano /etc/vsftpd.conf
```

Find the entry labelled write_enable=NO, and change the value to “YES”

```
# This directive enables listening on IPv6 sockets. By default, listening
# on the IPv6 "any" address (::) will accept connections from both IPv6
# and IPv4 clients. It is not necessary to listen on *both* IPv4 and IPv6
# sockets. If you want that (perhaps because you want to listen on specific
# addresses) then you must run two copies of vsftpd with two configuration
# files.
listen_ipv6=YES
#
# Allow anonymous FTP? (Disabled by default).
anonymous_enable=NO
#
# Uncomment this to allow local users to log in.
local_enable=YES
#
# Uncomment this to enable any form of FTP write command.
write_enable=YES
#
# Default umask for local users is 077. You may wish to change this to 022,
# if your users expect that (022 is used by most other ftpd's)
#local_umask=022
```

```
$ sudo systemctl restart vsftpd
```

We have completed installing and setting up our FTP server!

Part 2 – Add the necessary codes to the Software Request Form

Code Logic:

When senior management clicks on the "submit" button with "accepted" status,

Is the form a Software Request Form? If yes,

Obtain the 1st software to be installed and write it into a file contained in local system.

Obtain the 2nd software to be installed, if not NIL, write it into a file contained in local system

Obtain the 3rd software to be installed, if not NIL, write it into a file contained in local system

Obtain the 4th software to be installed, if not NIL, write it into a file contained in local system

Send the file containing the software to be installed into the Ansible Control Machine (ACM) using FTP connection

The following code snippet shows how to connect to the FTP server and upload file to the FTP server.

```
FtpWebRequest request = (FtpWebRequest)WebRequest.Create(server);
request.Method = WebRequestMethods.Ftp.UploadFile;
request.Credentials = new NetworkCredential(userName, password);

// Copy the contents of the file to the request stream.
byte[] fileContents;
using (StreamReader sourceStream = new StreamReader(fileName))
{
    fileContents = Encoding.UTF8.GetBytes(sourceStream.ReadToEnd());
}
request.ContentLength = fileContents.Length;
using (Stream requestStream = request.GetRequestStream())
{
    requestStream.Write(fileContents, 0, fileContents.Length);
}
```

We have finished setting up our Software Request Form to send the required software names to the ACM's FTP Server each time a Software Request Form is approved!

2) Notify the ACM that there is/are software to be installed on the target server(s), and pass the software names to the ACM so that the ACM knows which software needs to be installed on the target server(s), and then install the software into the target server(s)

   Using the tool "inotifywait", Ansible Control Machine (ACM) can be notified and trigger an event each time a particular file is modified, and in our scenario, this particular file is the file that will contain the required software to be installed onto the chosen server(s).

   Each time the Software Request Form is approved, the program will make use of the established FTP connection (shown in previous steps) to send the names (of the software to be installed) into that particular file contained in the Ansible Control Machine, and when this happens, the file is modified, and ultimately, immediately alerting/notifying the Ansible Control Machine to trigger an event (which is to install those software into the target server(s)).

Part 1 – Create an "Ansible Playbook Script" that will install software into target server(s)

Create an empty ansible playbook script (.yml extension)
$ mkdir /home/student/AnsiblePublish/win_package.yml

Playbooks are one of the core features of Ansible and tell Ansible what to execute. They are like a to-do-list for Ansible that contains a list of tasks. Playbooks contain the steps which the user wants to execute on a particular machine.

The following playbook installs the software firefox, curl, and winscp in the target server(s) (named windows-servers) using Windows Chocolatey package.

Part 2 – Install and enable inotifywait

At the ACM, install and enable "inotifywait" on a file.

"inotifywait" efficiently waits for changes to files using Linux's inotify(7) interface. It is suitable for waiting for changes to files from shell scripts. It can either exit once an event occurs, or continually execute and output events as they occur

Run the following commands to install inotifywait

```
$ sudo apt-get update

$ sudo apt-get install inotify-tools
```

Set up watches for the file (playbook script).

The following instruction will execute the command
$ ansible-playbook win_package.yml
each time the file "win_package.yml" is modified

The above command purpose is to trigger the Ansible Playbook Script ("win_package.yml") to install the software into the target server(s).

```
$ while [[ 1 ]]; do inotifywait -e modify win_package.yml; ansible-playbook
win_package.yml ; done
```

We have created an Ansible Playbook Script (APS) that installs software into target server(s), and setup "inotifywait" to monitor the APS such that when it is modified (i.e. A Software Request Form is approved and transfers the names (of software to be installed) to the APS), the APS is executed (to install the software into the target server(s)).

We have completed setting up connection from the Change Request Form to the Ansible Control Machine!

Conclusion

1. Creating and setting up Change Request Form as well as the associated database
2. Installing FTP Service, creating Ansible Playbook Script, and setting up inotifywait on the ACM
3. Making Software Request Form to connect to and upload contents into the ACM's FTP Server each time a form of such type is approved

References

1. How to install an FTP Server on Ubuntu with Vsftpd
   https://phoenixnap.com/kb/install-ftp-server-on-ubuntu-vsftpd

2. How to install inotify-tools in Ubuntu 18.04
   https://www.howtoinstall.me/ubuntu/18-04/inotify-tools/

3. Listen for changes and update output
   https://unix.stackexchange.com/questions/493879/is-there-a-way-to-get-ls-to-listen-for-changes-and-update-output-similar-to-tail

Previously, we have discussed the installation and configuration documentation for the **main functionality** for my project (which involves Ansible, Change Request Form and Target Web Server(s)).

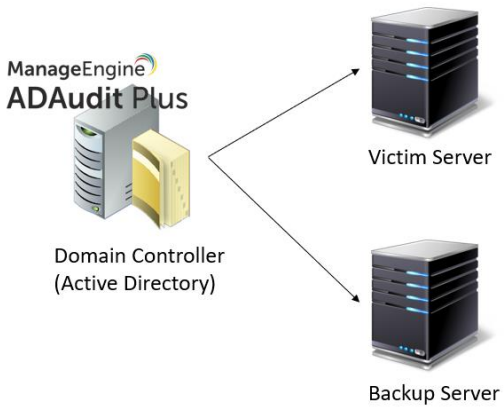
In the following sections, I will be discussing about the installation and configuration documentation for the **supporting functionality** which I have also done for my project to support the Change & Configuration Management Process.

Supporting Functionality

- Detecting changes (which can be any type of changes including file changes, configuration changes, etc.) made to protected systems in our organization.
- Machines and applications involved:
  - Windows Server 2016 (Active Directory Domain Controller)
  - AD Audit Plus (Software)
  - Windows Server 2016 (Victim)
  - Windows Server 2016 (Backup Server)

Note: All three of "Windows Server 2016" are different machines. They are not the same machines (i.e. one is the Domain Controller while the other are the victims).

The following diagram shows the relationship between the machines and applications involved:



Detecting changes with AD Audit Plus & Group Policy

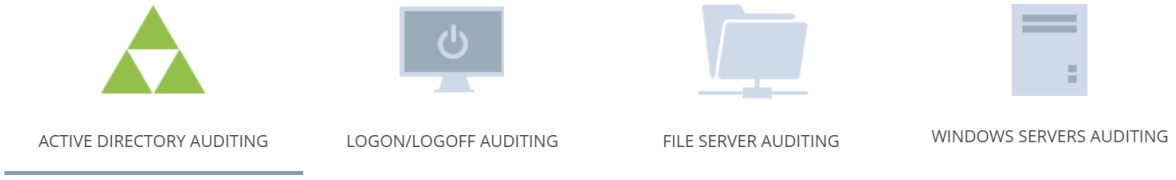
Installation Documentation

Installing AD Audit Plus

ADAudit Plus by ManageEngine is an on-premise auditing solution. Key features include tools that allow users to audit active directories, login and logoff records, file servers and Windows server data.

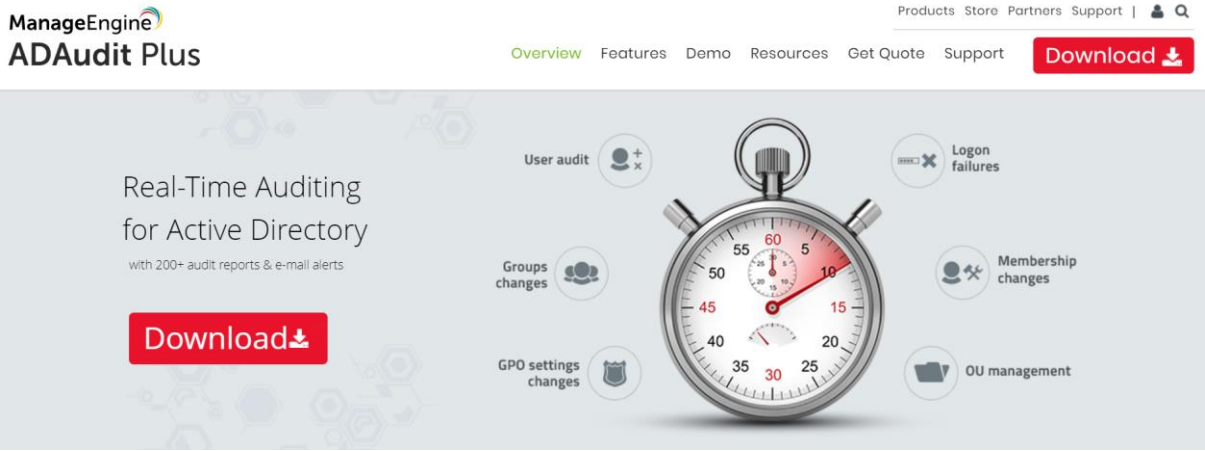
ADAudit Plus allows for the following auditing:

a. *Active Directory Auditing*
b. *Logon/Logoff Auditing*
c. *File Server Auditing*
d. *Windows Server Auditing*



ADAudit Plus allows **real-time Windows Active Directory auditing**.

To install AD Audit Plus, go to the website https://www.manageengine.com/products/active-directory-audit/ and click on "Download" located at the top-right hand of the screen.



For my project, I am installing AD Audit Plus into the Domain Controller (Active Directory).

Conclusion

1. Installing AD Audit Plus into the Domain Controller (Active Directory)

References

1. What is ManageEngine AD Audit Plus?
   https://www.softwareadvice.com/audit/adaudit-plus-profile/

2. ManageEngine AD Audit Plus Download
   https://www.manageengine.com/products/active-directory-audit/

Configuration Documentation

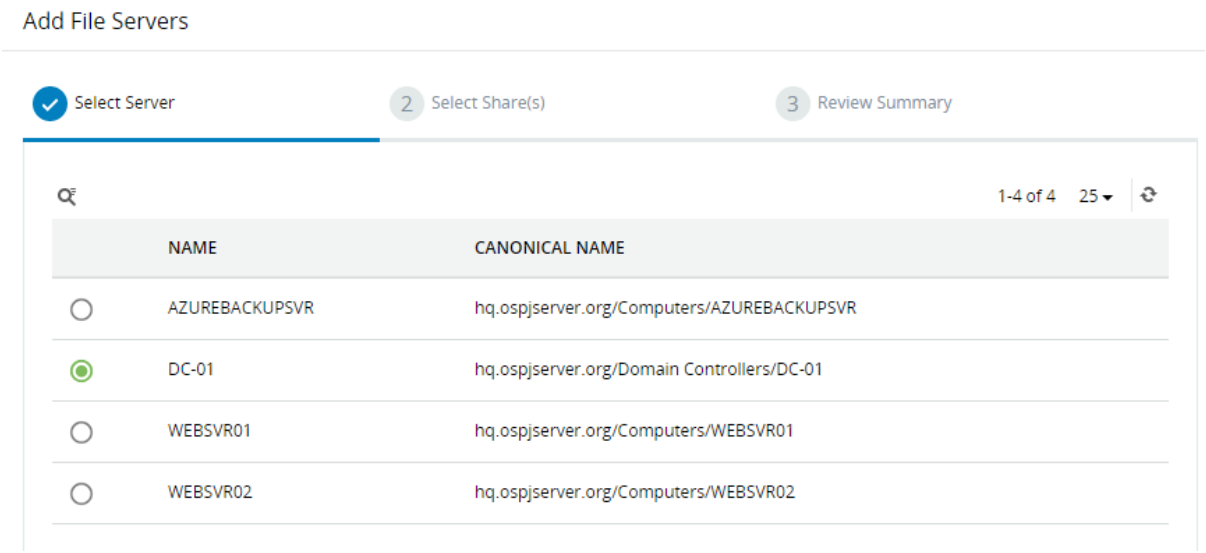
After installing AD Audit Plus in the Domain Controller (Active Directory), we want to configure AD Audit Plus to protect our systems (by detecting changes made to the protected systems).

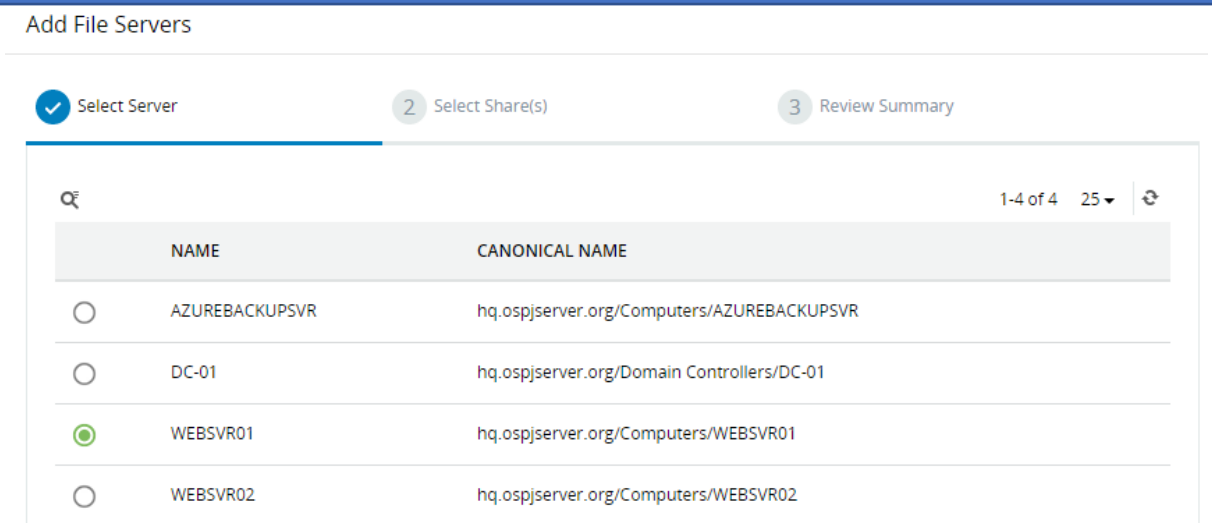
In the following scenarios, the machines aliases are as follows:

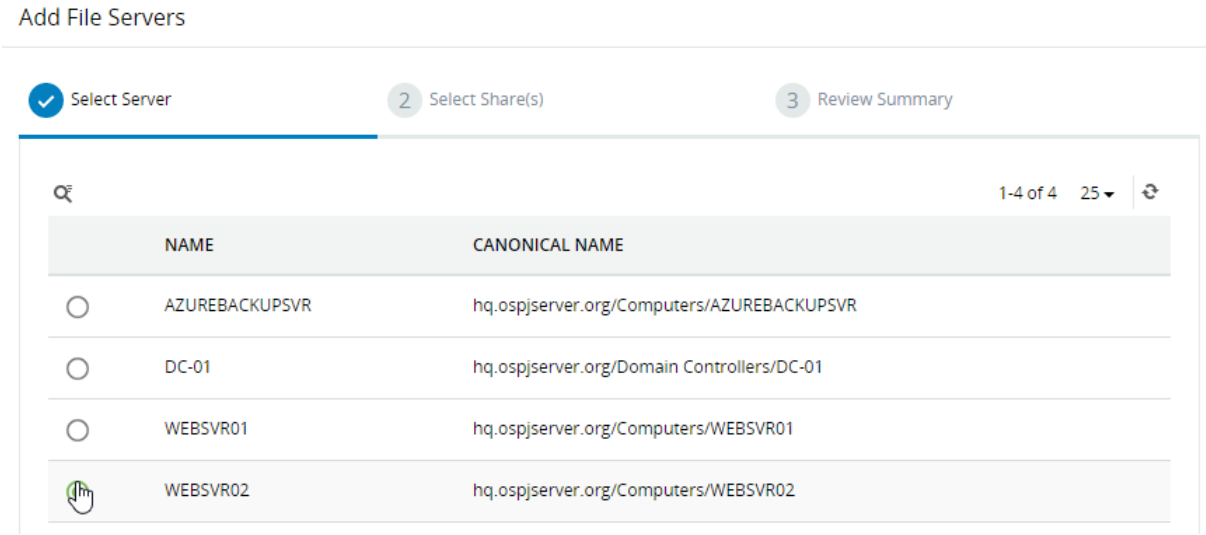
- DC-01 (Active Directory Domain Controller)
- WEBSVR01 (Victim)
- WEBSVR02 (Backup Server)

Part 1 – Add the three servers into AD Audit Plus

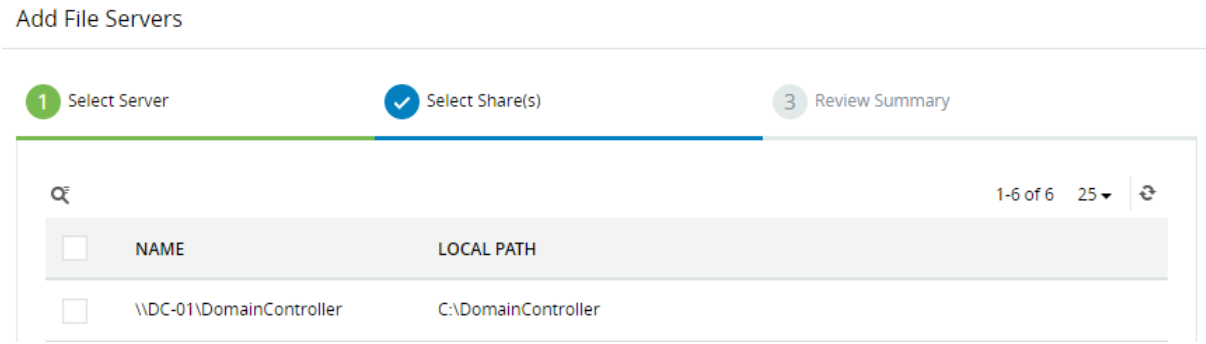
1. In AD Audit Plus, add the relevant servers.

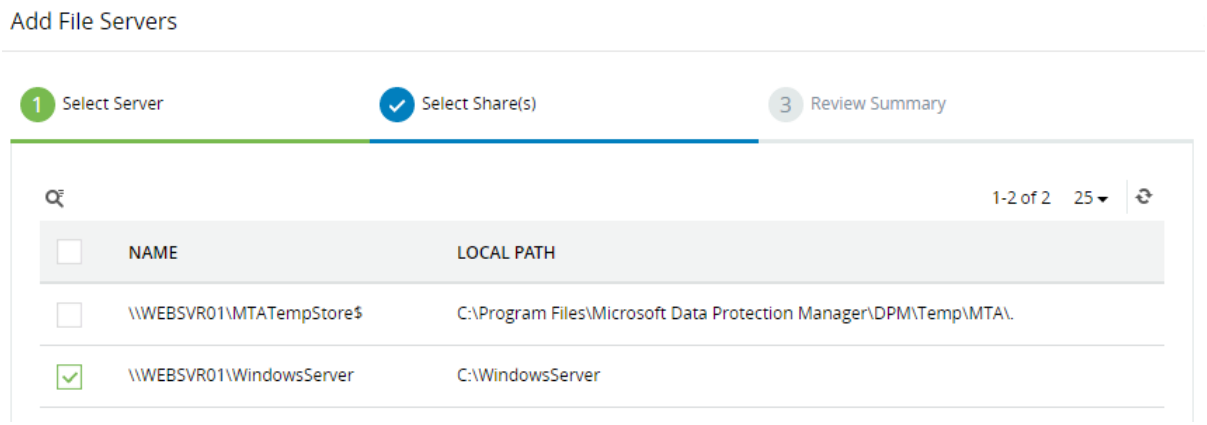


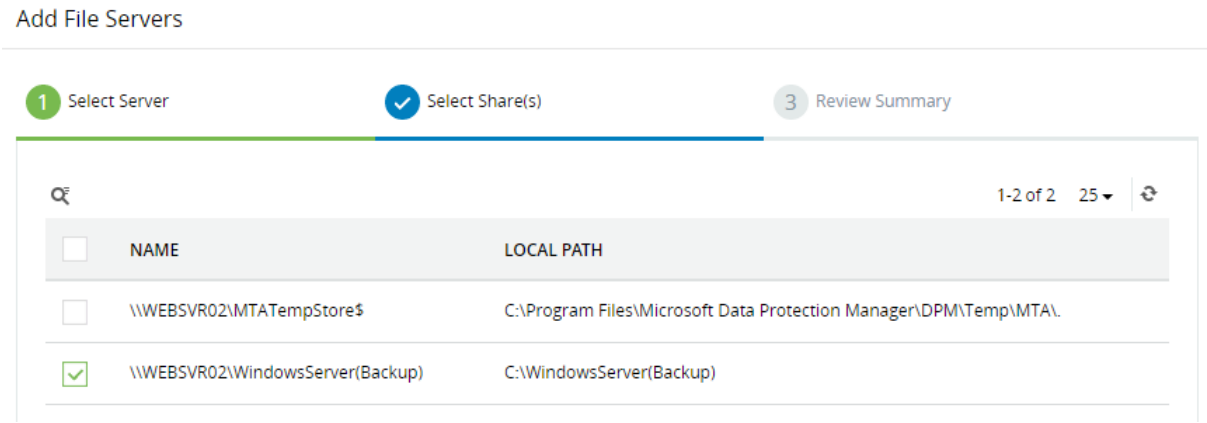




2. Next in each of the servers, add the relevant shares.





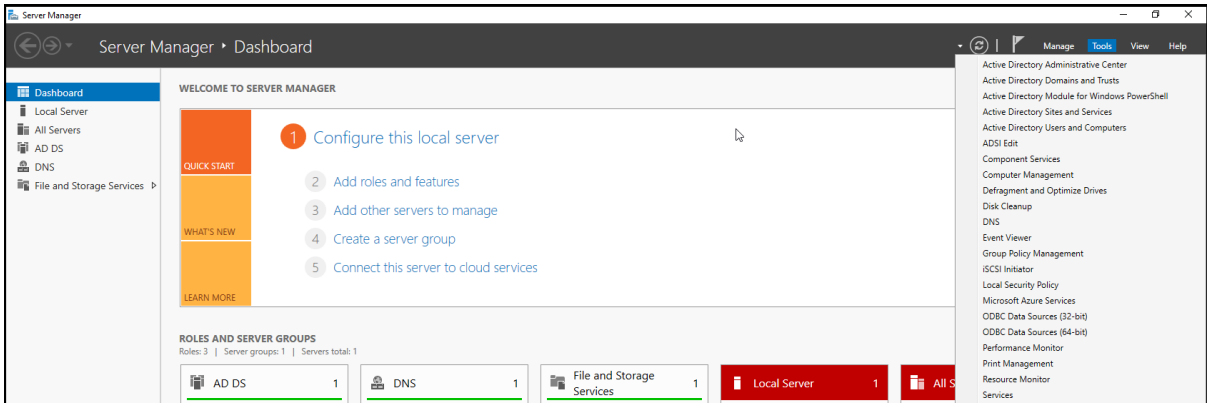


We have completed adding the three servers into AD Audit Plus. However, at this moment, AD Audit Plus is unable to detect any changes made to the protected servers.

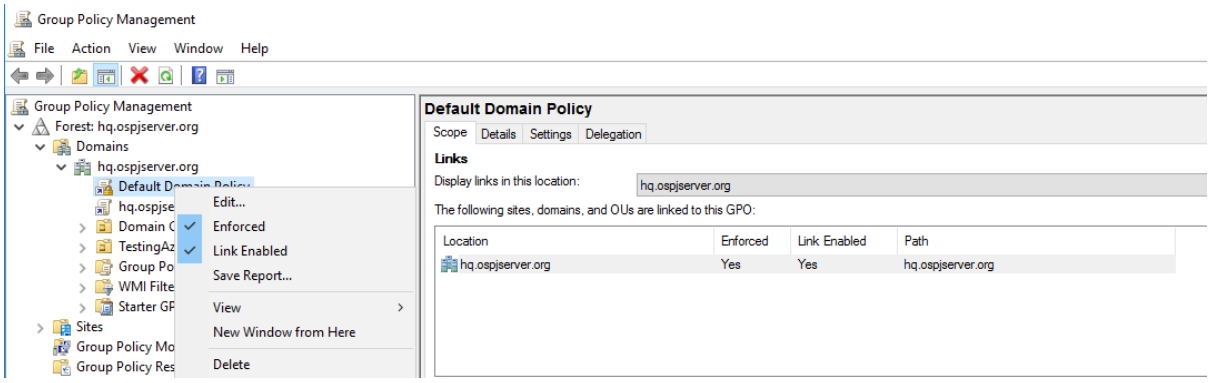
In the following steps, we are going to configure each of the three servers to allow AD Audit Plus to detect changes made to them.

For each of the three servers, repeat the following steps 3-6 for each of them.

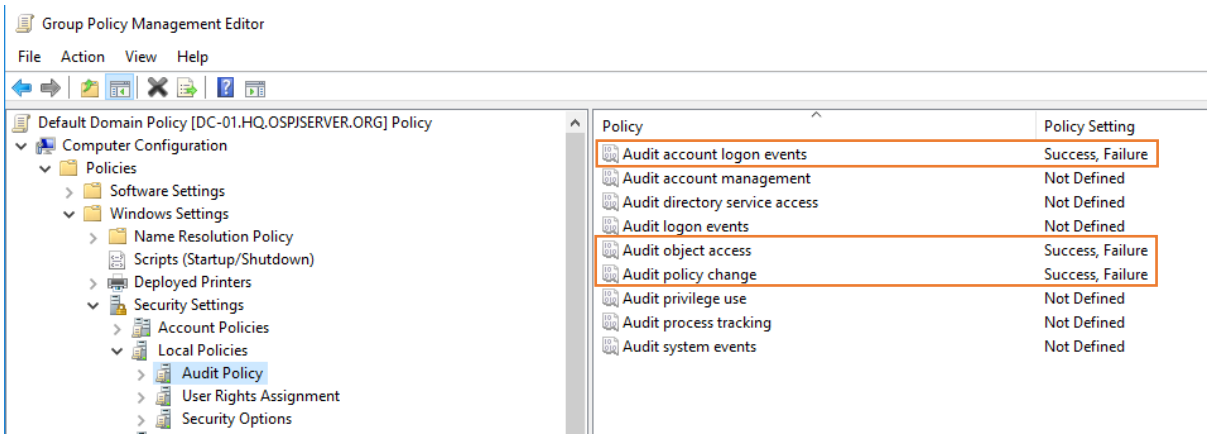
3. At the "Server Manager" click on "Group Policy Management".



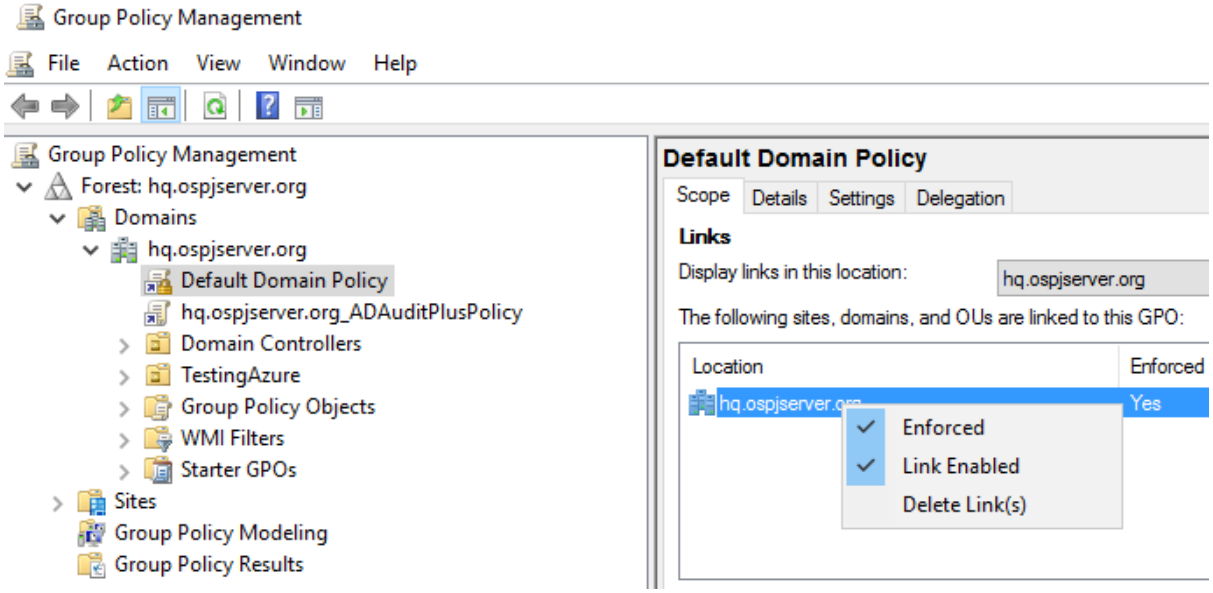
4. **At "Group Policy Management", right click on "Default Domain Policy" and "Edit" to open up the "Group Policy Management Editor" window.**



5. **At the "Group Policy Management Editor", expand "Computer Configuration -> Policies -> Windows Settings -> Security Settings -> Local Policies -> Audit Policy" and enable "Audit account logon events", "Audit object access" and "Audit policy change" for both "Success" and "Failure".**

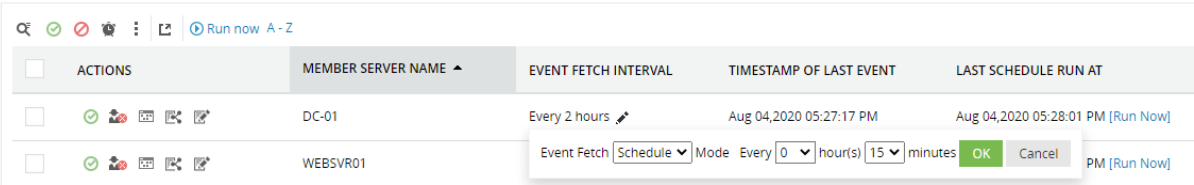


6. **Once done, close the "Group Policy Management Editor" window and we are brought back to the "Group Policy Management" window. At "Group Policy Management", right-click on the domain name and check "Enforced" to enforce the policy settings that we just made.**



We have finished configuring each of the three servers to allow AD Audit Plus to detect changes in them. Now, in the final step, we want to configure AD Audit Plus to automatically retrieve events (i.e. changes) in the protected systems every 15 minutes.

7. **To configure AD Audit Plus to automatically retrieve events, we can use the scheduler included in AD Audit Plus. In the following screenshot, it is set for AD Audit Plus to automatically fetch events every 15 minutes.**



Conclusion

1. Configuring AD Audit Plus to work with the systems in our organization.
2. Configuring Group Policy rulesets to enable change detection by AD Audit Plus.
3. Configuring AD Audit Plus to automatically retrieve events (e.g. changes) in protected systems.

References

1. AD Audit Plus – File Auditing
   https://www.youtube.com/watch?v=SvCSRfWa0kc

2. How to configure audit policies for servers with AD Audit Plus
   https://www.youtube.com/watch?v=vnIAA-xR3SA

3. How to configure Audit Policies on Windows Server 2016
   https://www.youtube.com/watch?v=JPqLYbtbic4

Integration with Splunk

Configuration Documentation – Ansible Configuration

1. In /etc/ansible/ansible.cfg, uncomment the following lines to enable logging

```
# logging is off by default unless this path is defined
# if so defined, consider logrotate
#log path = /var/log/ansible.log
```

2. To enable integration with Splunk, we change the following from

```
# enable callback plugins, they can output to stdout but cannot be 'stdout' type.
#callback_whitelist = timer, mail
```

to

```
# enable callback plugins, they can output to stdout but cannot be 'stdout' type.
callback_whitelist = splunk
```

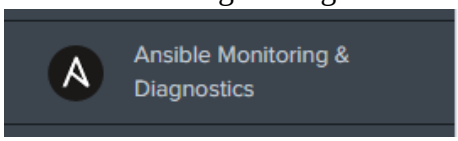
3. Add the following lines into ansible.cfg file

```
# enable callback plugins, they can output to stdout but cannot be 'stdout' type.
callback_whitelist = splunk
export SPLUNK_URL=http://192.168.0.128:8088/services/collector
export SPLUNK_AUTHTOKEN=401ec4a6-8205-4f18-a960-f3e6006dcfa4
```

```

[callback_splunk]
url = http://192.168.0.128:8088/services/collector
authtoken = 401ec4a6-8205-4f18-a960-f3e6006dcfa4
```

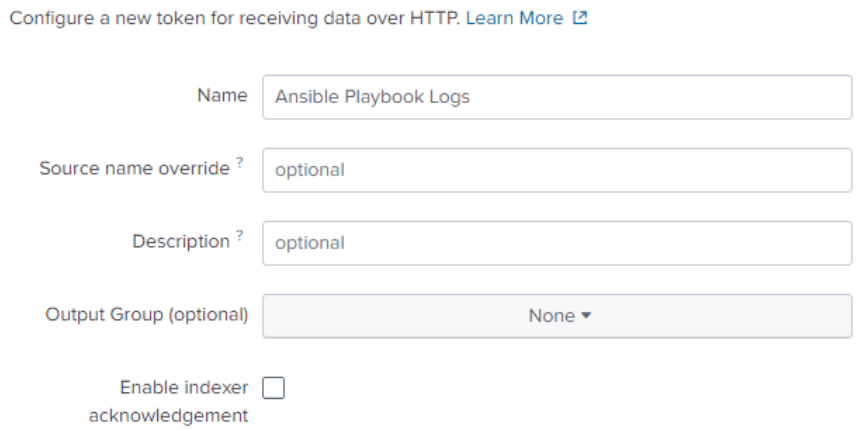
Configuration Documentation – Splunk Configuration

1. Download Splunk App Ansible Monitoring & Diagnostics to ingest the Logs

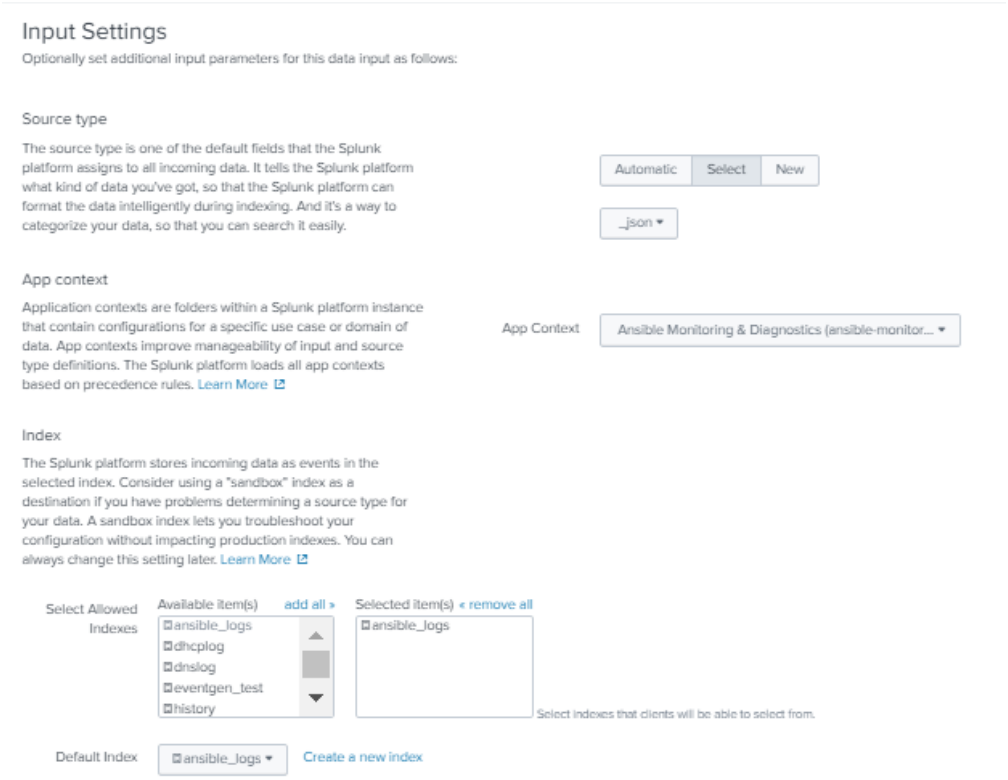


Ansible Monitoring & Diagnostics	ansible-monitoring-and-diagnostics_122	1.2.2	Yes	Yes	App | Permissions	Enabled | Disable

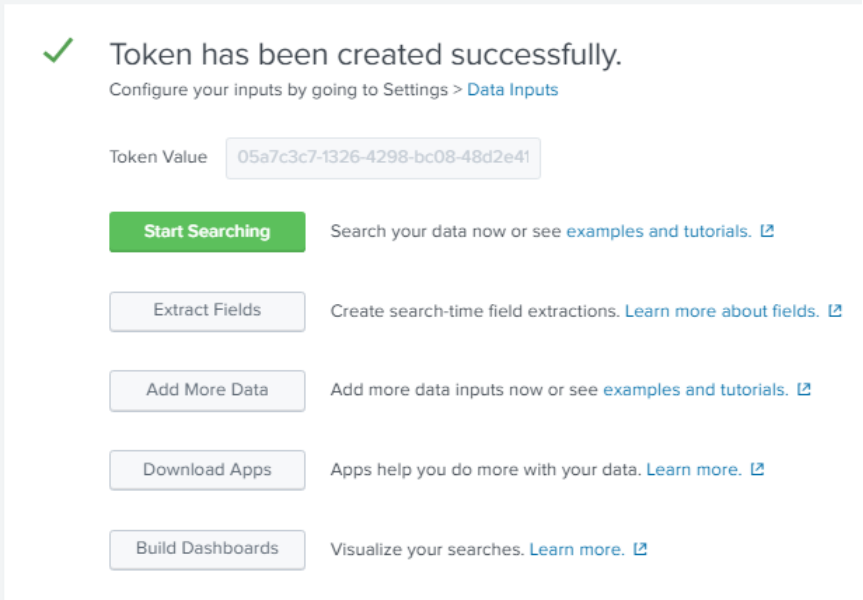
Create HTTP Event Collector to ingest the Logs



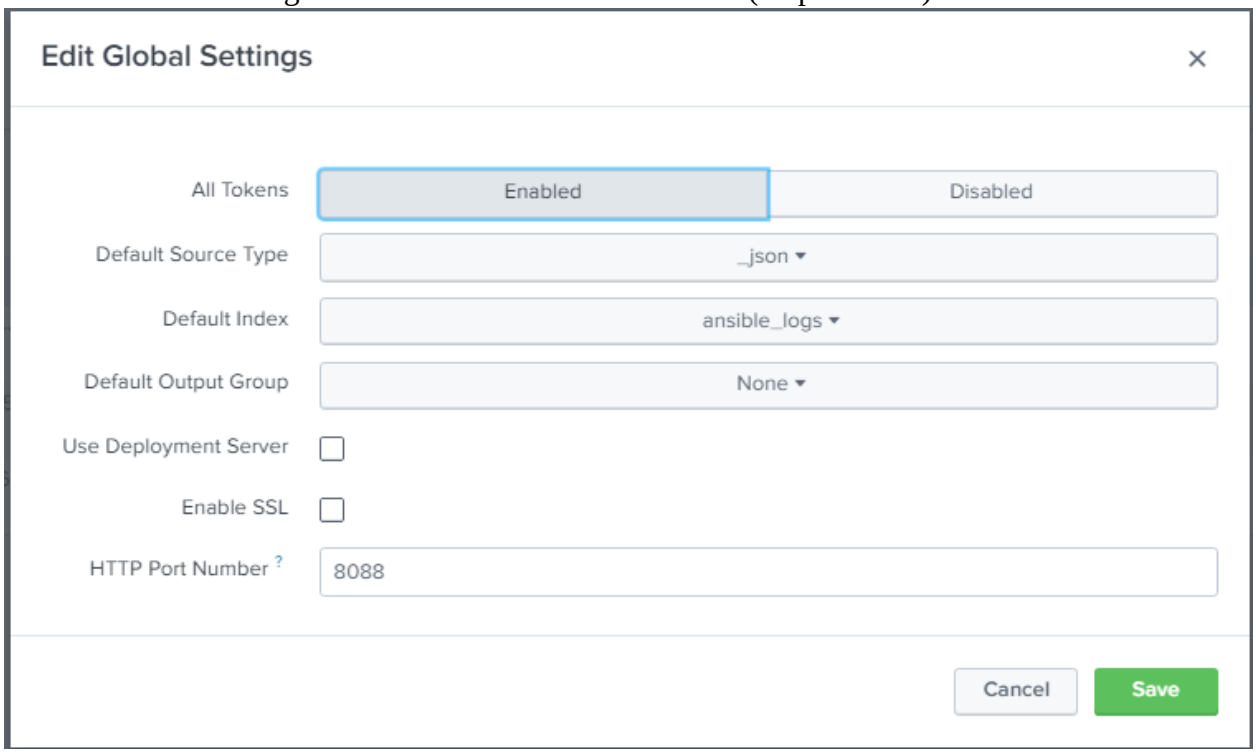
2. Set up input settings for the HTTP Event Collector



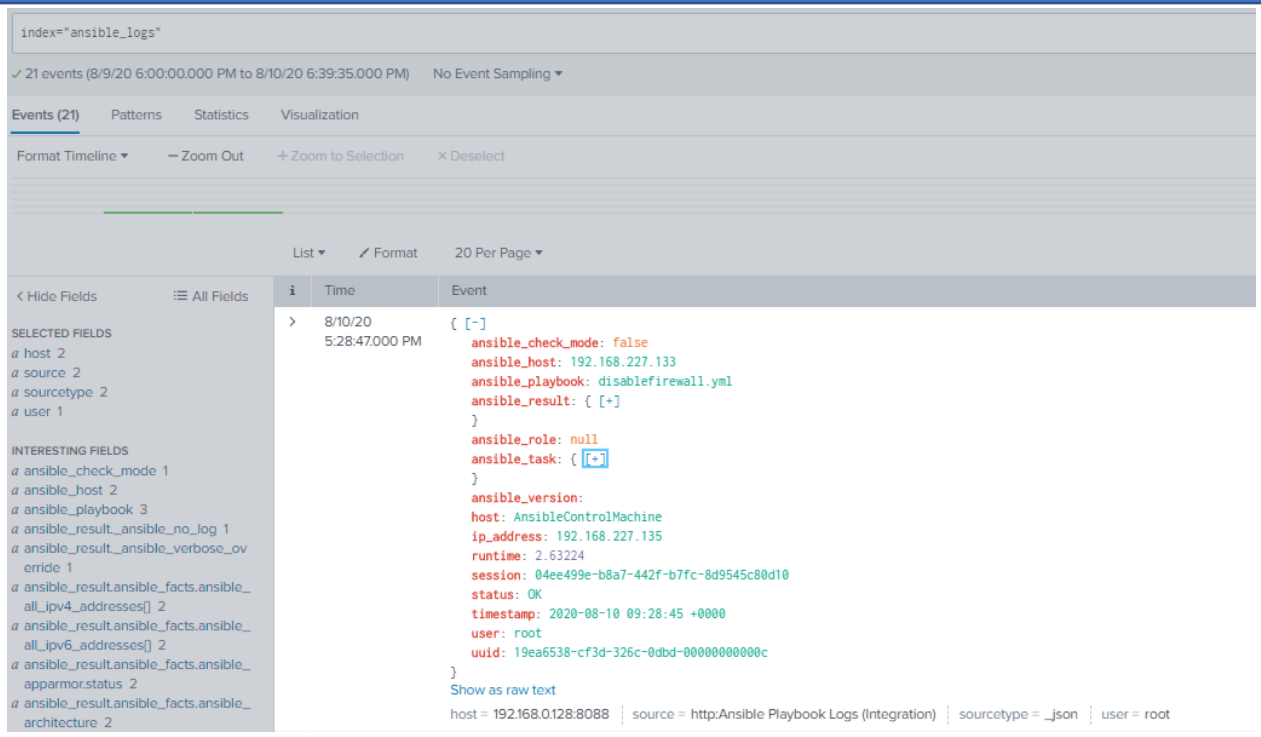
Token Created Successfully!



3. Edit Global Settings to Enable HTTP Event Collector (on port 8088)



<u>Sample Output (index="ansible_logs")</u>



We have finish configuring Ansible and Splunk to work together

Integration with pfSense

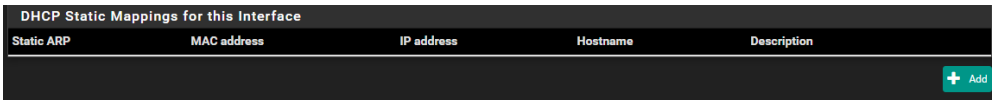
Configuration Documentation – Configuring DHCP Server

Firewall (pfSense) DHCP Static Mapping & allow port rule for Ansible

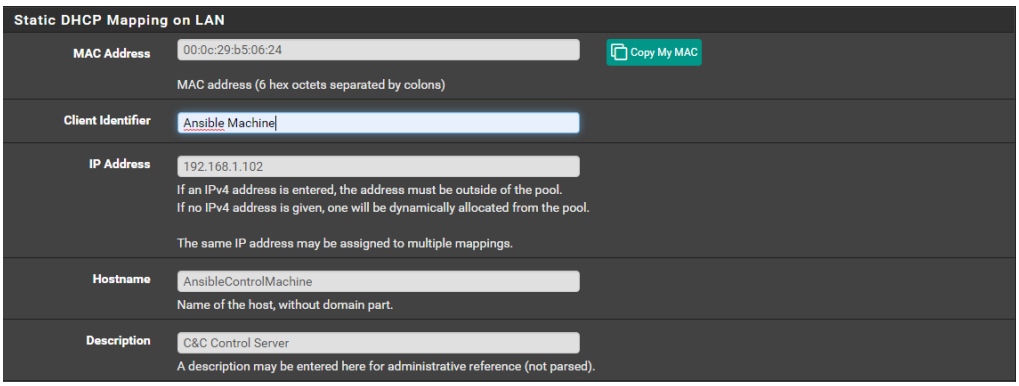
Retrieve the MAC address of the NIC from the Ansible Machine. (00:0c:29:ba:05:cb)

```
root@AnsibleControlMachine:/home/student# ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.13  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::5750:b5c2:fb7b:3b28  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:b5:06:24  txqueuelen 1000  (Ethernet)
        RX packets 909  bytes 262140 (262.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 164  bytes 21589 (21.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

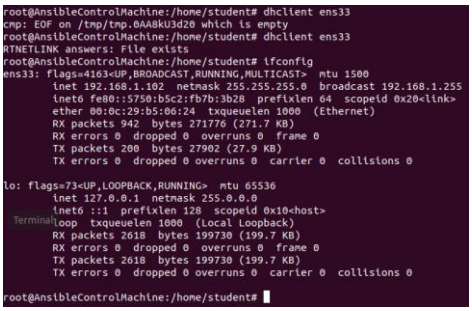
Navigate to Service → DHCP Server → LAN → Add



Specify the MAC address corresponding to the NIC, specify the IP address that needs to be outside

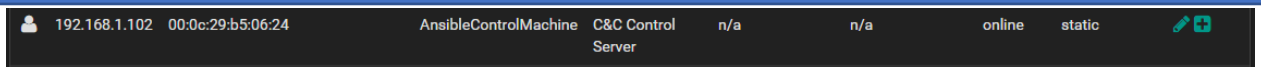


Reacquire IP address - Reacquire a new IP address from the pfSense DHCP server



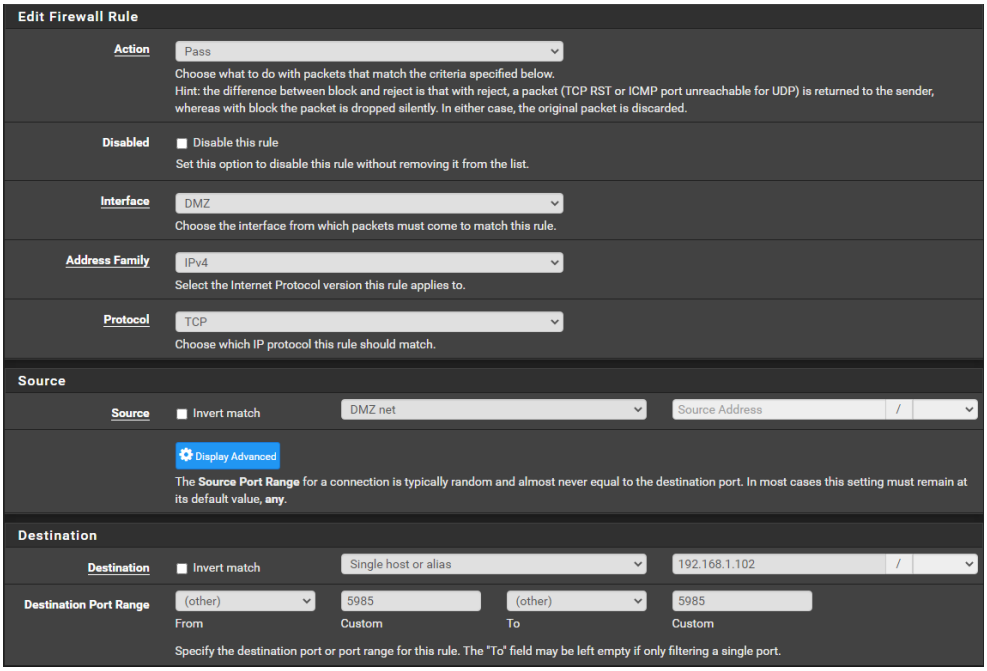
```
root@AnsibleControlMachine:/home/student# dhclient ens33
cmp: EOF on /tmp/tmp.0AA8kU3d20 which is empty
root@AnsibleControlMachine:/home/student# dhclient ens33
RTNETLINK answers: File exists
root@AnsibleControlMachine:/home/student# ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.102  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::5750:b5c2:fb7b:3b28  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:b5:06:24  txqueuelen 1000  (Ethernet)
        RX packets 942  bytes 271776 (271.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 200  bytes 27902 (27.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 2618  bytes 199730 (199.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2618  bytes 199730 (199.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@AnsibleControlMachine:/home/student#
```

Lease Window



Creating a rule to Allow WinRM service port 5986 (DMZ → LAN)

Firewall → Rules → DMZ → Add



Integration with Automated Backup

Configuration Documentation – Ubuntu Configuration

1. Open Terminal and enter apt-get install cifs-utils

```
root@AnsibleControlMachine:/home/student# apt-get install cifs-utils
Reading package lists... Done
Building dependency tree
Reading state information... Done
cifs-utils is already the newest version (2:6.9-1).
The following packages were automatically installed and are no longer required:
  linux-headers-5.4.0-37 linux-headers-5.4.0-37-generic linux-headers-5.4.0-39
  linux-headers-5.4.0-39-generic linux-image-5.4.0-37-generic
  linux-image-5.4.0-39-generic linux-modules-5.4.0-37-generic
  linux-modules-5.4.0-39-generic linux-modules-extra-5.4.0-37-generic
  linux-modules-extra-5.4.0-39-generic
Use 'apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 228 not upgraded.
```

2. Create a mounting point at /mnt/backup by entering mkdir /mnt/backup

```
root@AnsibleControlMachine:/home/student# mkdir /mnt/backup
```

3. Mount the mounting point using this command
   mount -t cifs //192.168.0.115/UbuntuBackup/ /mnt/backup -osec=ntlmv2,domain=hq.ospjserver.org,username=nigeltan,password=P@ssw0rd

```
root@AnsibleControlMachine:/home/student# mount -t cifs //192.168.0.115/UbuntuBackup/ /mnt/backu
p -osec=ntlmv2,domain=hq.ospjserver.org,username=nigeltan,passwo[redacted]rd=P@ssw0rd
```

4. Install pscp into Azure Backup Server
   Link: https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html

5. To see all the mounted partitions on the server type df -h

```
root@AnsibleControlMachine:/home/student/Desktop# df -h
Filesystem                    Size  Used Avail Use% Mounted on
udev                          1.9G     0  1.9G   0% /dev
tmpfs                         391M  1.8M  390M   1% /run
/dev/sda5                      20G  8.7G  9.5G  48% /
tmpfs                         2.0G     0  2.0G   0% /dev/shm
tmpfs                         5.0M  4.0K  5.0M   1% /run/lock
tmpfs                         2.0G     0  2.0G   0% /sys/fs/cgroup
/dev/loop1                     56M   56M     0 100% /snap/core18/1885
/dev/loop0                     55M   55M     0 100% /snap/core18/1880
/dev/loop2                    256M  256M     0 100% /snap/gnome-3-34-1804/33
/dev/loop3                    256M  256M     0 100% /snap/gnome-3-34-1804/36
/dev/loop8                     30M   30M     0 100% /snap/snapd/8790
/dev/loop6                     50M   50M     0 100% /snap/snap-store/454
/dev/loop7                     30M   30M     0 100% /snap/snapd/8542
/dev/loop5                     50M   50M     0 100% /snap/snap-store/467
/dev/loop4                     63M   63M     0 100% /snap/gtk-common-themes/1506
/dev/sda1                     511M  4.0K  511M   1% /boot/efi
tmpfs                         391M   24K  391M   1% /run/user/1000
//192.168.0.115/UbuntuBackup   60G   34G   26G  57% /mnt/backup
```

6. This is the script to backup

backup.sh
~/Desktop

```
#!/bin/sh
####################################
#
# Backup to NFS mount script.
#
####################################

# What to backup.
backup_files="/home/student/Desktop/testing123"

# Where to backup to.
dest="/mnt/backup"

# Create archive filename.
day=$(date +%A)
hostname=$(hostname -s)
archive_file="$hostname-$day.tgz"

echo "Backing up $backup_files to $dest/$archive_file"
date
echo


#rsync -aAXv --exclude={"/dev/*","/proc/*","/sys/*","/tmp/*","/run/*","/mnt/*","/media/*","/-lost+found"} / /mnt/backingup

# Print start status message.
echo "Backing up $backup_files to $dest/$archive_file"

# Backup the files using tar.
tar czf $dest/$archive_file $backup_files

# Print end status message.
echo
echo "Backup finished"
date
```

7. To automate the backup, we need to setup the cronjob. Login as student and type sudo crontab -e. You can choose whichever editor you prefer .

```
student@AnsibleControlMachine:~/Desktop$ sudo crontab -e
[sudo] password for student:
no crontab for root - using an empty one

Select an editor.  To change later, run 'select-editor'.
  1. /bin/nano        <---- easiest
  2. /usr/bin/vim.tiny
  3. /bin/ed

Choose 1-3 [1]: 1
```

Enter following line to the file and save it. This cronjob will run the backup file everyday at 7pm.

```
00 19 * * * bash /home/student/Desktop/backup.sh
```

Link: https://ittutorials.net/open-source/linux/how-to-backup-ubuntu-to-a-windows-share/