



RADIUS/ TACACS In The Cloud

CCNP Lab 5

2018-2019

Nipun Chhajer

Cisco CCNP - Hoffman and Mason - Periods 6 and 7

RADIUS/TACACS - In the Cloud Lab 5

Purpose

The purpose of this lab was to take what we learned from the previous lab, about TACACS+ and RADIUS authentication, and apply it to cloud based infrastructures, like Amazon Web Services and Microsoft Azure.

Background Information

Cloud based services have become increasingly more prevalent in today's world. As more and more startups and companies start growing and expanding their resources, they must have an efficient way to expand their networking infrastructures and server related projects as well. This can be easily done with the power of cloud services such as Amazon Web Services and Microsoft Azure. In a couple of minutes, a company could more than double the amount of server space and increase their server's processing power ten fold than they currently have, and easily enlarge their networking infrastructures as well. Services like AWS and Azure, especially, are gaining more and more popularity extremely fast, due to their reliability and easy of accessibility. Plus, you can combine the power of the cloud with local machines such as routers and switches to create a local networking infrastructure, with cloud implemented to handle any sort of tasks that may be intensive for routers and switches. That is what this lab is about, simply

building off of the last lab, we are configuring TACACS+ and RADIUS in the cloud and connecting them to a local router to use for authentication.

Lab Summary

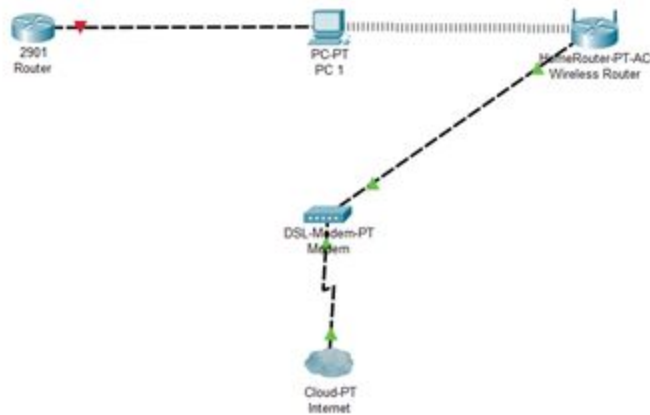
In this lab we set up two virtual machine: one Linux and one Windows Server. We then set up the respective servers' security protocol (TACACS+ on Linux and RADIUS on Windows).. Next, using aaa commands we configured routers that allowed for authentication using TACACS and RADIUS. Finally we tested the configurations, by logging in to the routers using the username and passwords we set up on each of the Virtual Machine's security protocols

Lab Commands

aaa new model	Creates a new aaa model for the router to use when people try to login to the router - allows for RADIUS or TACACS+ to be configured.
tacacs-server host [ip address]	Defines the host TACACS+ sever, where the TACACS+ configuration was done, so that the router can get the information from it.
tacacs-server directed-request	To allow users to send authentication requests to a specific TACACS+ server when logging in
tacacs-server key [key]	Sets the key for the tacacs-server on the router, must match the key defined in the tac_plus.conf file on linux.
aaa authentication login default group [tacacs+ or radius] local	Sets the default security protocol to be used for logging into the router.
aaa authentication enable default group tacacs+ enable	Sets the default security protocol to be used for enabling the user to

	login to the router
aaa authorization config-commands	Configures default authorization commands for USER EXEC commands
radius-server host [address] key [key]	Allows for router to connect to the RADIUS server, and retrieve information for it
aaa authorization commands [level] default group tacacs+ none	Sets the level of authority the client has, meaning which changes they are allowed to make and which information they are allowed to access. Levels go from 0 to 15, where 15 is highest.
Aaa group server radius [name]	Defines the radius server's name that is to be used.
aaa session-id common	To ensure the session ID is maintained across all authentication, authorization, and accounting packets in a session
ping [ip-address]	Shows the connectivity between the device and another one - works for IPv4 and IPv6
ip address [address]	Sets up the IPv4 address in an interface

Network Diagram



Device	IP Address
Router G0/0	DHCP
PC 1	DHCP
Wireless Router	192.168.40.1/23

Configurations

Tacacs Router Configuration:

```
ip address dhcp
Interface GigabitEthernet0/0
    no shut
aaa authentication login default group tacacs+ local none
aaa authorization exec default group tacacs+ local none
aaa authorization commands 0 default group tacacs+ local none
tacacs-server host 40.78.42.250
tacacs-server key AzureTacacs
```

Radius Router Configuration:

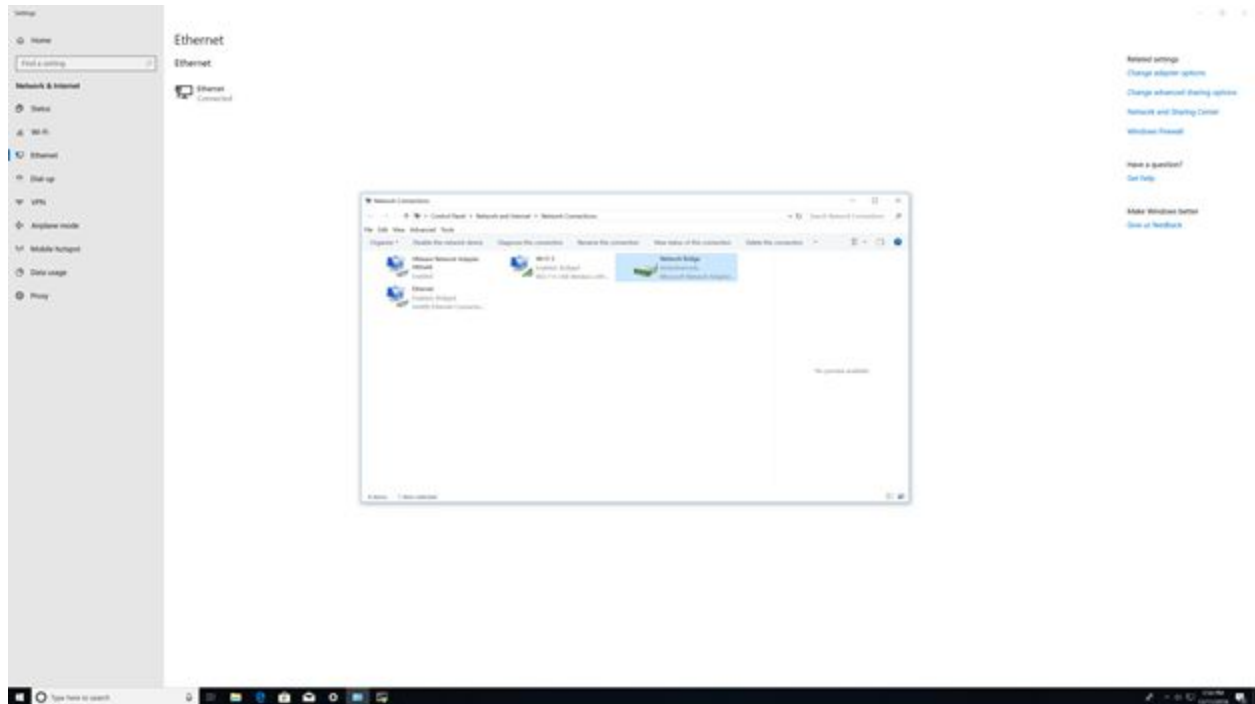
```
ip address dhcp
Interface GigabitEthernet0/0
    no shut
aaa new-model
aaa authentication login default group radius local
aaa authorization exec default group radius if-authenticated
```

radius-server host 40.78.6.150 key AzureRadius

Screenshots

Configurations

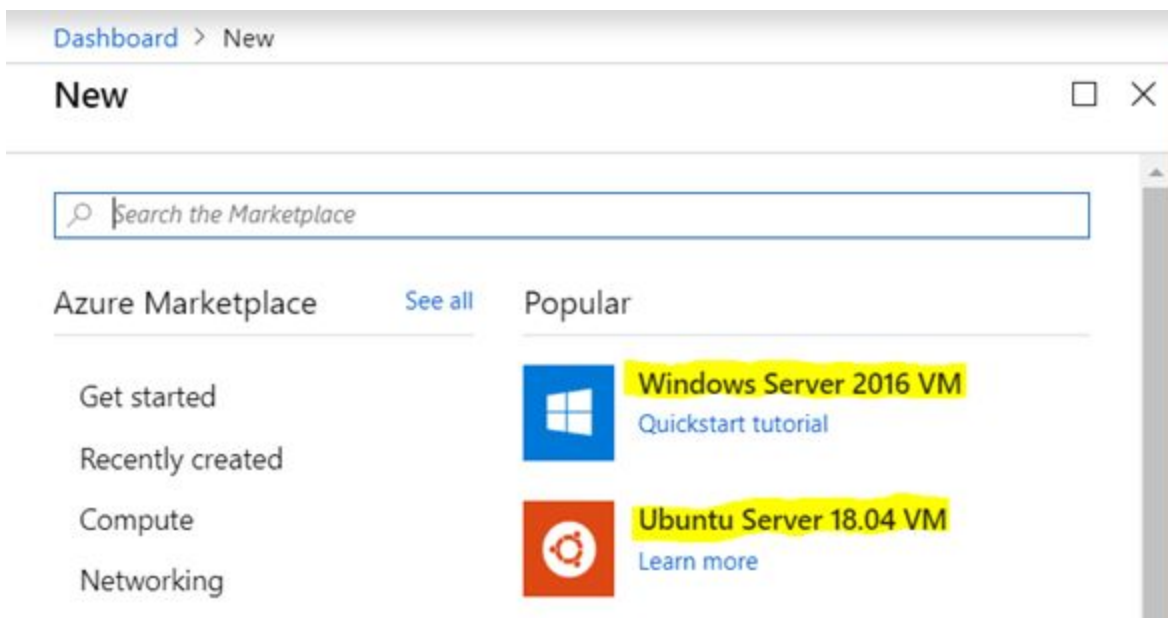
Step 1.



Connect a router to the PC in the local network, and link to the PC to the internet, in this scenario, we utilized a wireless connection to a home router which was further connected to the Internet. After confirming that the two connection are recognized by the local machine, bridge both the connections in order for all the devices in the local network to attain an internet connection.



Step 2.



Advance to Microsoft Azure and after a successful login attempt to your Azure account, create one of the two highlighted virtual servers on the cloud with your desired preferences.

Step 3.



Prior to heading to the next step, you should check the public IP address from a trusted source on the internet. One should also ensure that the ISP is using PAT with just a single public IP Address, by checking the public IP address on other devices connected to the same wi-fi.

Step 4.

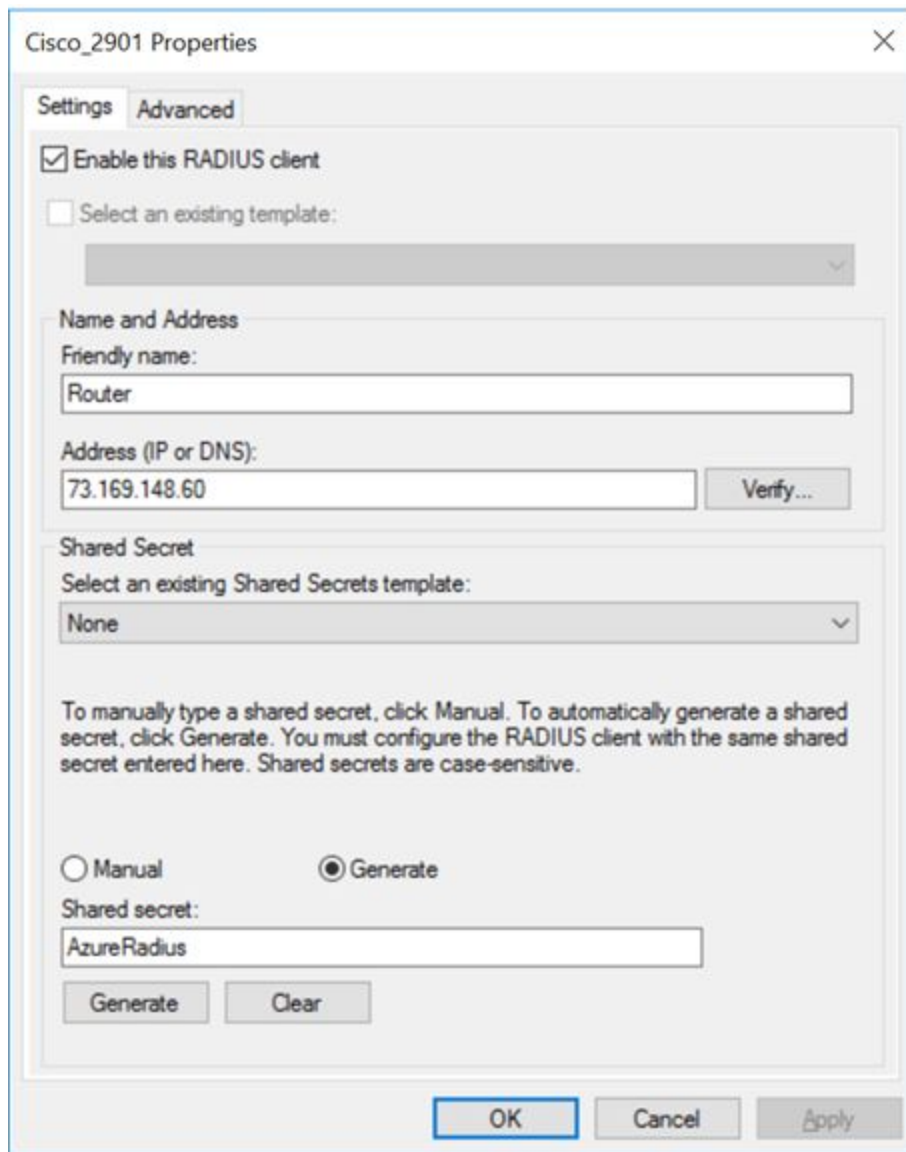
Size
Standard F1 (1 vcpus, 2 GB memory)
Public IP address
40.78.6.150
Virtual network/subnet
TanishkAzure-vnet/default

Before starting the lab on the Windows Server 2016, note the public IP address of the server shown on Azure and utilize the in-built application called Remote Desktop Management on the Windows PC in the local machine to remotely manage the Windows Server.

Step 5.



Perform the similar processes from the “RADIUS and TACACS+ Lab” to complete the RADIUS configurations on the server except for the following :-

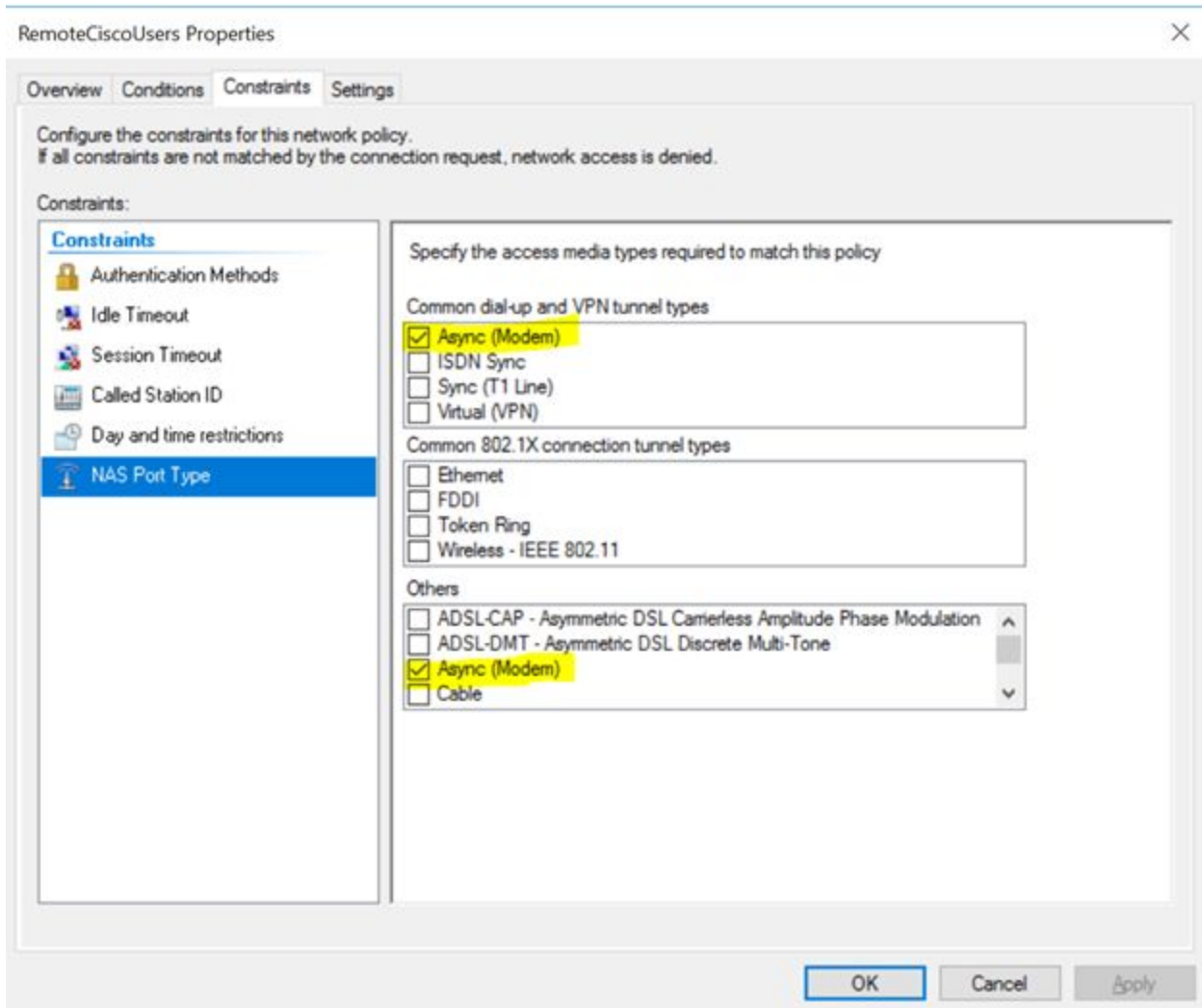


The image shows a Windows-style dialog box titled "Cisco_2901 Properties" with a close button (X) in the top right corner. It has two tabs: "Settings" and "Advanced", with "Advanced" currently selected. The "Advanced" tab contains the following elements:

- A checked checkbox labeled "Enable this RADIUS client".
- An unchecked checkbox labeled "Select an existing template:" followed by a dropdown menu.
- A section titled "Name and Address" containing:
 - A label "Friendly name:" followed by a text box containing "Router".
 - A label "Address (IP or DNS):" followed by a text box containing "73.169.148.60" and a "Verify..." button.
- A section titled "Shared Secret" containing:
 - A label "Select an existing Shared Secrets template:" followed by a dropdown menu showing "None".
 - A paragraph of text: "To manually type a shared secret, click Manual. To automatically generate a shared secret, click Generate. You must configure the RADIUS client with the same shared secret entered here. Shared secrets are case-sensitive."
 - Two radio buttons: "Manual" (unselected) and "Generate" (selected).
 - A label "Shared secret:" followed by a text box containing "AzureRadius".
 - "Generate" and "Clear" buttons.

At the bottom of the dialog are three buttons: "OK", "Cancel", and "Apply".

Under the “Clients” section, instead of the private IP address of the router on the local network, which will act as the RADIUS client in this lab, fill the public IP address that was learnt by checking on the internet, and :-



In the Network Policies, under the NAS Port Type Section, which happens to be under the constraints section, check the boxes which show: “Async (Modem)”

Step 6.

At this point, the RADIUS configuration on the server is complete. Progress to the router in the LAN and copy the configurations from above.

If all the procedures are finely followed, the user should have a successful login attempt.

Step 7.

For completing the TACACS+ process, create the Ubuntu server, and instead of using RDP to remotely manage the server, head to the Serial Console section in the cloud serve that is being used and type in the following commands :-

sudo apt-get install tacacs+

service tacacs_plus start

nano /etc/tacacs+/tac_plus.conf

Step 8.

After executing the third command from the given set of commands above creat a user, group and a TACACS+ key that can be used by the router in the local network.

```
accounting file = /var/log/tac_plus.acct

# This is the key that clients have to use to access Tacacs+
key = "AzureTacacs"

# Use /etc/passwd file to do authentication
```

```
# We also can define local users and specify a file where data is stored.
# That file may be filled using tac_pwd
#user = test1 {
#    name = Bob
#    member = RemoteCiscoUsers
#    login = 
#}
```

```
# We can also specify rules valid per group of users.
#group = RemoteCiscoUsers {
#    default serice = permit
#    service = exec {
#        priv-lvl = 15
#    }
#}
```

Ensure that everything except for the name of the group remains similar, and username must be a part of the new group created.

Step 9.

We are left with a minor step before moving to the router for the completion of the lab. There are two commands that are essential for the completion that must be deployed on the server :-

chmod o-r /etc/tacacs+/tac_plus.conf

service tacacs_plus reload

Step 10.

Finally, proceed to the router. If you are using the same router as used in the configuration of RADIUS, make certain to reload the router and erase the startup-config file if anything was saved there. Copy and paste the router configurations for TACACS+ from above.

Step 11.

Repeat the same steps on AWS for concluding the lab.

Problems

During this lab, we encountered many problems. The main problem was getting the configuration on the Virtual Machine itself correct. If the TACACS+ or RADIUS configuration wasn't correct on the Virtual Machines, then the authentication on the routers would fail. We would constantly research how to configure the files correctly, and how to get authentication working. After countless failed attempts, we cracked it and got authentication on both TACACS+ and RADIUS working. Another major problem, was getting the Virtual Machines to connect to internet and ethernet. Ethernet connectivity was a problem for both, since it took time to figure out how to get the Virtual Machines on AWS and Radius connected. After a while, we figured out that

we just had to bridge the connection between the Virtual Machines and their host computers, and finally, they connected.

Conclusion

In conclusion, we set up four cloud based Virtual Machines, AWS Linux and Windows Server, and Azure Linux and Windows Server. This was a very challenging lab, and once we got it to work, it was a very fulfilling moment, as we encountered many problems where one small error could cause the whole thing to fail. I now feel very confident in implementing these two technologies.