TACACS+ and RADIUS in AWS and Azure

Purpose

The purpose of this lab was for us to set up TACACS+ and Radius authentication through the cloud with Azure and AWS. Through this lab, we learned how to secure our routers with a user which we stored in the cloud. Rather than just setting up Radius and TACACS+ on a virtual machine or switch, we set it up on the cloud which is more secure and a more useful skill to know as we now have experience in the cloud.

Background Information

In this lab, we used Radius and TACACS+ to remotely authenticate users in a router. Radius and TACACS+ are both forms of authentication used to permit users into a router. We setup Radius and TACACS+ on a remote Linux and Windows Server for this lab. We set these up through the cloud using Microsoft’s Azure and Amazon’s AWS services. We also set up AAA in the routers, which is a protocol used to secure access to a cisco network device. It stands for Authentication, Authorization, and Accounting. In order to get it working properly, we connected our computers to the router and to the Internet through a bridge, which allowed us to connect the PC’s ethernet connection to its Internet connection.

Lab Summary

In this lab, we configured TACACS+ and Radius through a Linux server and Radius server on the cloud. We used the services Azure and AWS with a free tier to get basic access to the cloud. We started by getting our AWS and Azure accounts and then creating four servers, one Linux and one Windows on each. Then, we started with TACACS+ on AWS and set it up. We used the terminal to install TACACS+ and then used its config file to set up the groups and users. After this, we then repeated the steps on Azure, and set up TACACS+ there too. Next, we created security groups on Azure and AWS to allow traffic from all ports. We then created a bridge to the internet on the PC, setup the router with AAA, and we were able to get it working. Next, we repeated the steps with Radius. We started with AWS and configured Radius by installing roles and features, creating a security group and user, setting up the policies, connection requests, and radius client, and then we continued to Azure with the same steps. After creating security groups within AWS and Azure to allow traffic on all ports, we setup our Router with AAA for Radius and we were able to get it working.

Lab Commands

In this lab, we used the AAA commands we previously learned, and a few new commands to work with the server. We needed different commands for Radius and TACACS+. For both of them, we used the command “aaa new-model” which created a new AAA model. For TACACS+, we first used the command “aaa authentication login default group tacacs+ local” which tells the router to check any login attempts against the TACACS+ server, and if it isn’t available to check the local user database. Next, we used the commands “aaa authorization config-commands,” “aaa authorization commands 0 default group tacacs+ local none,” and “aaa authorization commands 15 default group tacacs+ local none.” These three commands tell the router to contact the TACACS+ server and make sure if the user is allowed to run commands at that privilege level. The next three commands are “aaa accounting exec default start-stop group tacacs+,” and “aaa accounting network default start-stop group tacacs+.” These two commands tell the router to log activity into the TACACS+ server. Finally, for TACACS+, there is the “tacacs-server host …” command, the “tacacs-server key …” command, the “ip tacacs source-interface” command, and the “tacacs-server directed-request” command. The host command lets the router know where the host for the TACACS+ server is, the key command sets the TACACS+ encryption key, the “ip tacacs source-interface” command let you identify which interface was connected to the server, and the “tacacs-server directed-request” command is used to specify a specific TACACS IP for authentication. Finally, the “ip tacacs source-interface” command is used to specify the interface for TACACS to go through.

Next, for Radius, we used only five commands, “aaa new-model,” “aaa authentication login default group radius local,” “aaa authorization exec default group radius if-authenticated,” “radius-server host … key …,” and “ip radius source-interface.” The first command is used to create a new AAA model. The next command tells the router to check for all login attempts against the Radius server first, and then check the locally configured user database if the server isn’t available. The next command is used to contact the Radius server to determine if the user is allowed, and then lets them into the router. The next command lets the router know the IP and secret key of the Radius server so they can connect. The final command specifies which port RADIUS needs to go through.

Problems

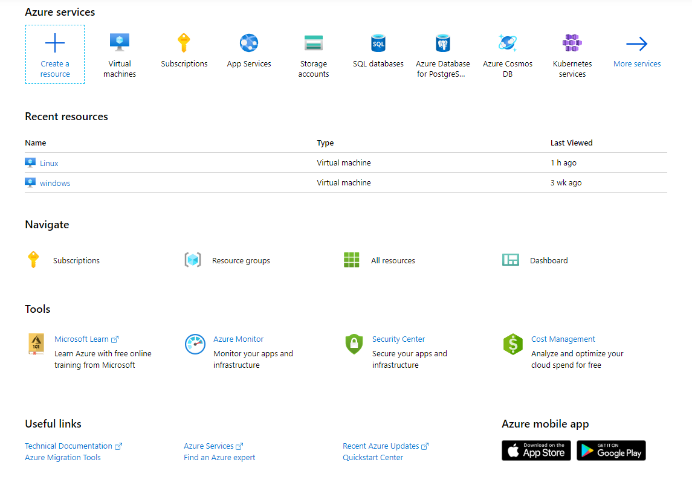
As we had never worked with the cloud before, we ran into a multitude of problems while setting this up. One of our first problems after setting up TACACS+ was to figure out how to connect the PC to the internet and to the router. After some research, we figured out how to create a network bridge from the control panel and connected the ethernet connection to the router to the internet. After we figured this out, we still had another problem with TACACS. We decided to research it more and we found out that we needed to add the command “tacacs-server directed-request” to our router config. After adding this line, we were able to get TACACS+ on Azure to work, but TACACS+ wasn’t working on AWS even though the commands were identical. We eventually found out through some help from others that in AWS our security groups were set up wrong, and it was simply because we gave them the wrong name. After correcting this mistake, we successfully setup TACACS+ on AWS and we were ready to move onto Radius.

Similarly with TACACS, setting up Radius came with its own problems, but we were able to solve them quicker than we did with TACACS. Our main problem with Radius at first turned out to be a simple one. After setting it up correctly, the Router was not connecting to the server. We tried changing a lot of settings within Radius to fix it, but it turned out that wasn’t the problem. After checking through our router config, I noticed that we had used the “ip radius source-interface” command with the wrong interface by accident. I changed this mistake, but there still was one more problem. We continued looking through Radius and eventually we found that we had not registered the server in the active directory, so it could not access the users we created. We fixed this by simply pressing the button that said, “register server in active directory.”

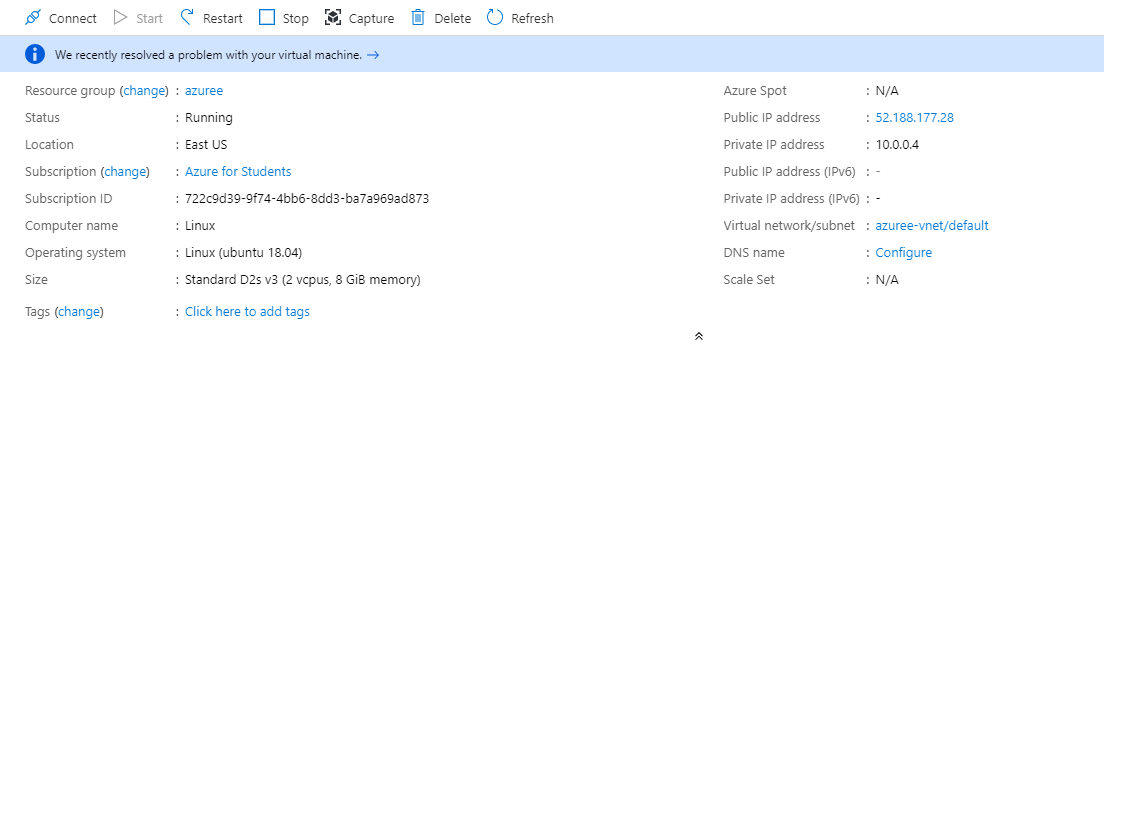
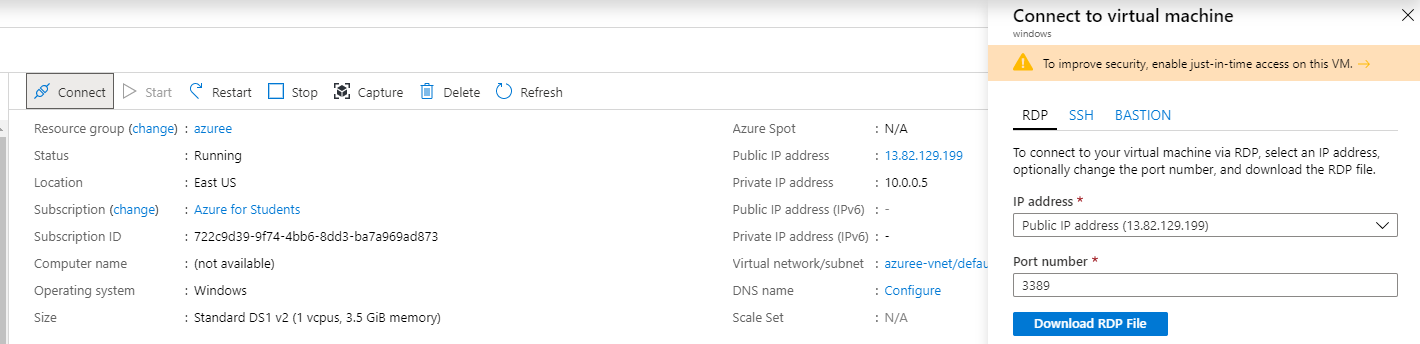
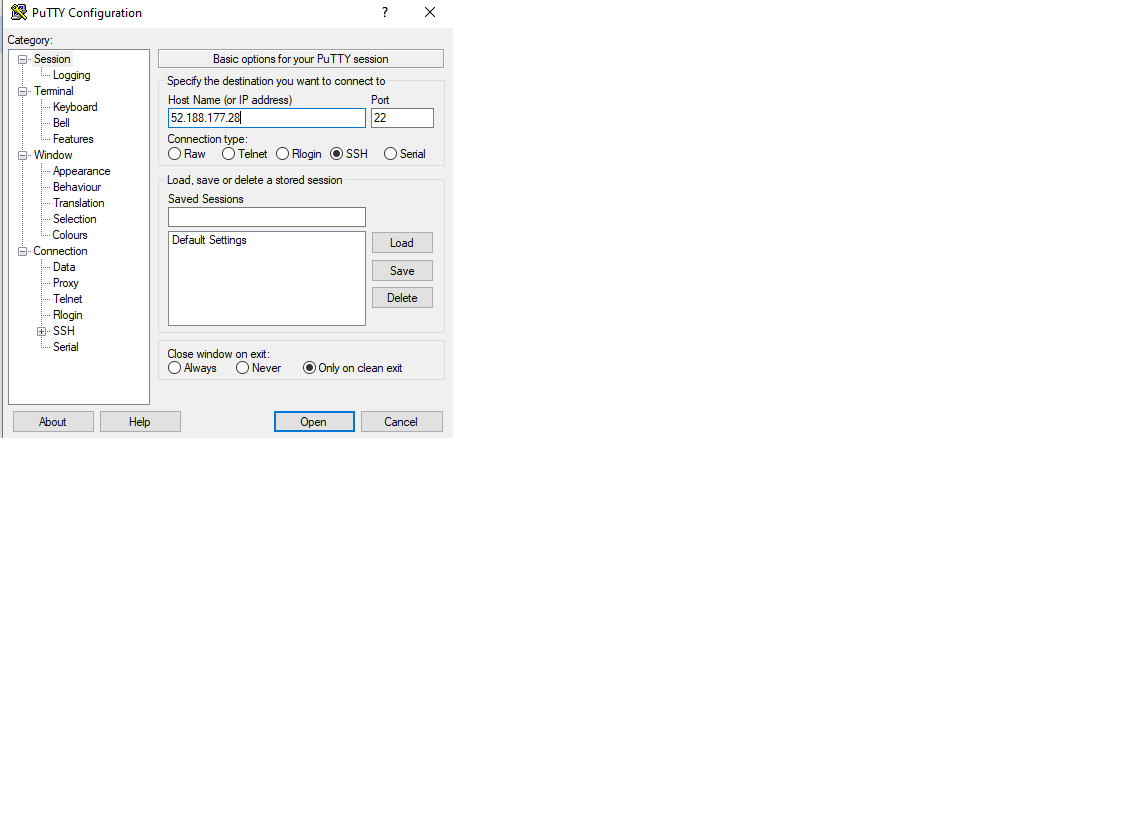
Conclusion

Overall, I learned a lot during this lab about the cloud and setting up Radius and TACACS+ through it. After we had a multitude of problems such as figuring out how to make a bridge, missing security groups, and forgetting to start the NPS service, we were able to successfully secure our router through the cloud and gain experience in using it. In the end, we learned how to set up TACACS+ and Radius on the cloud using Azure and AWS, and it gave us experience in a useful topic.

How to Setup a Resource in Azure

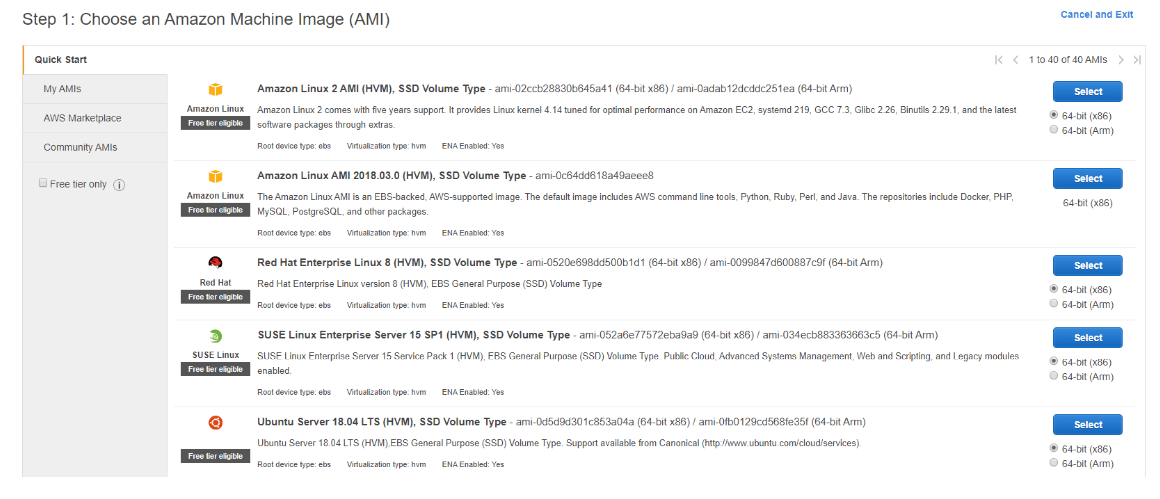




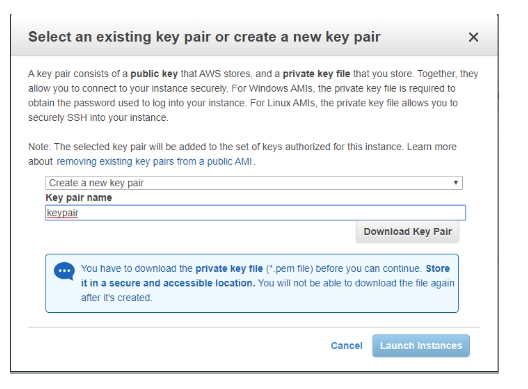
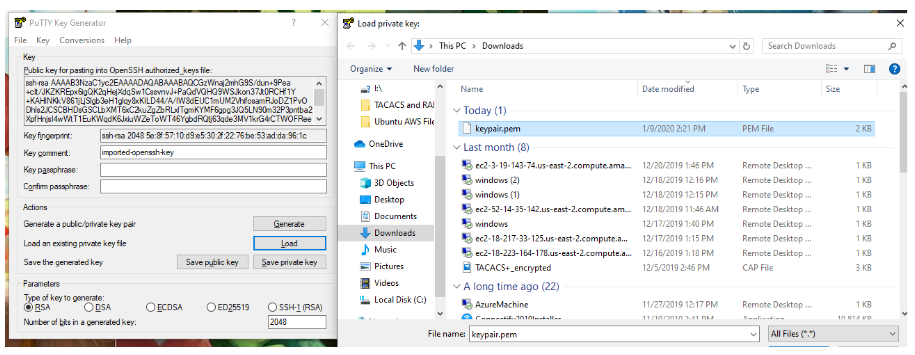




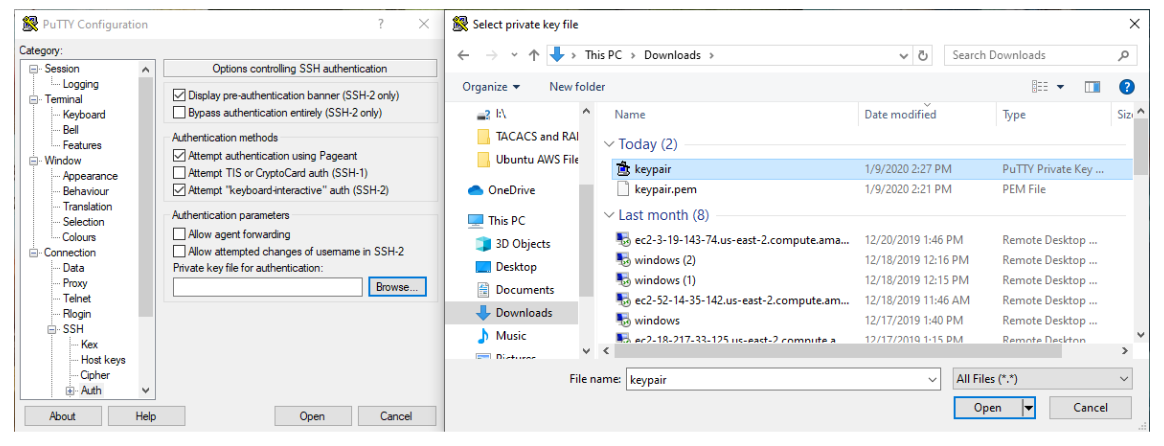
How to Setup an Amazon Machine Image (AMI) in AWS



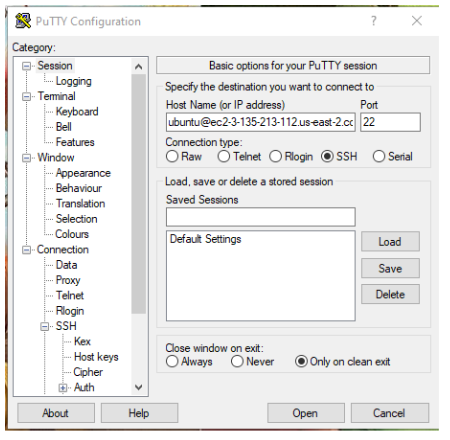




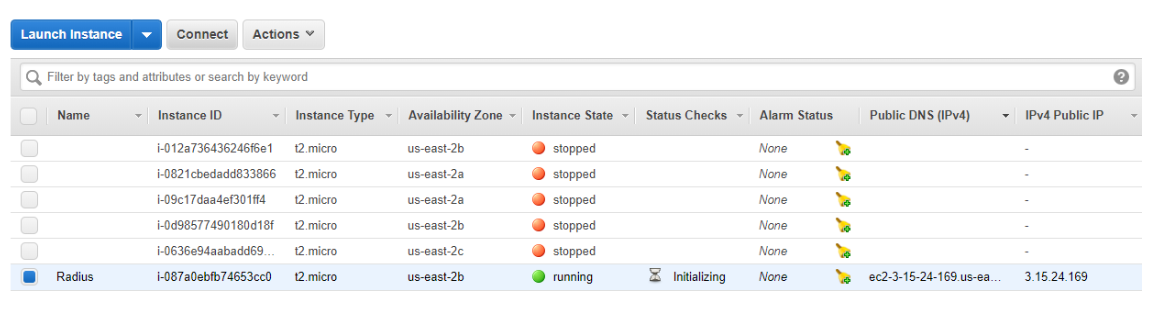




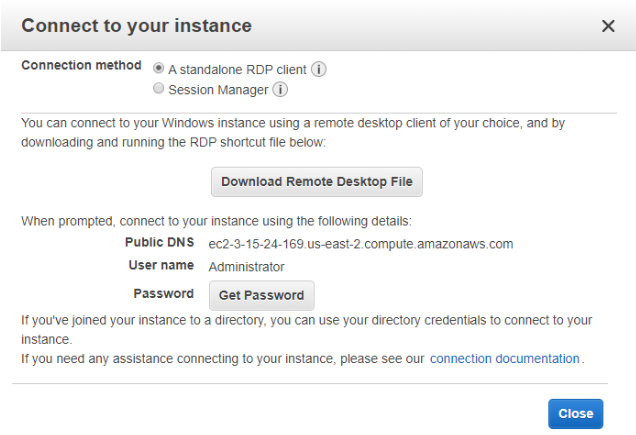




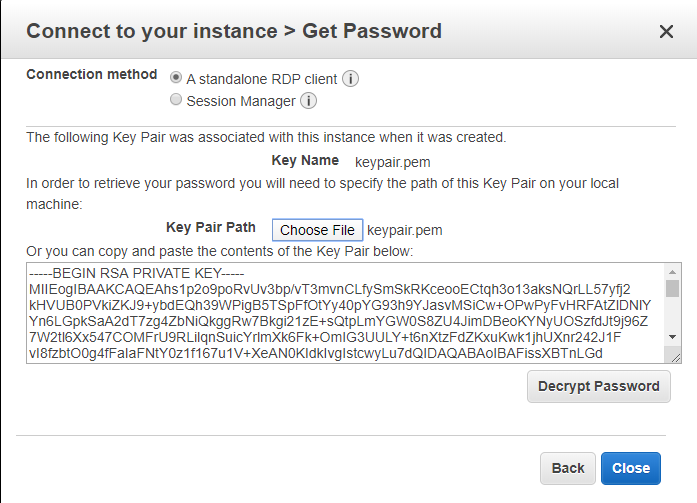


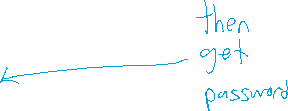
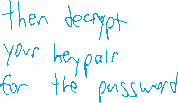




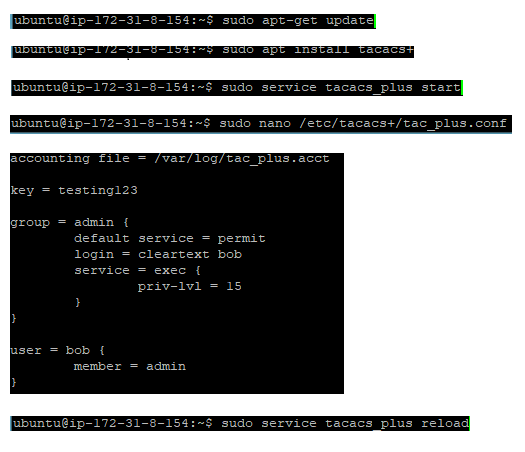






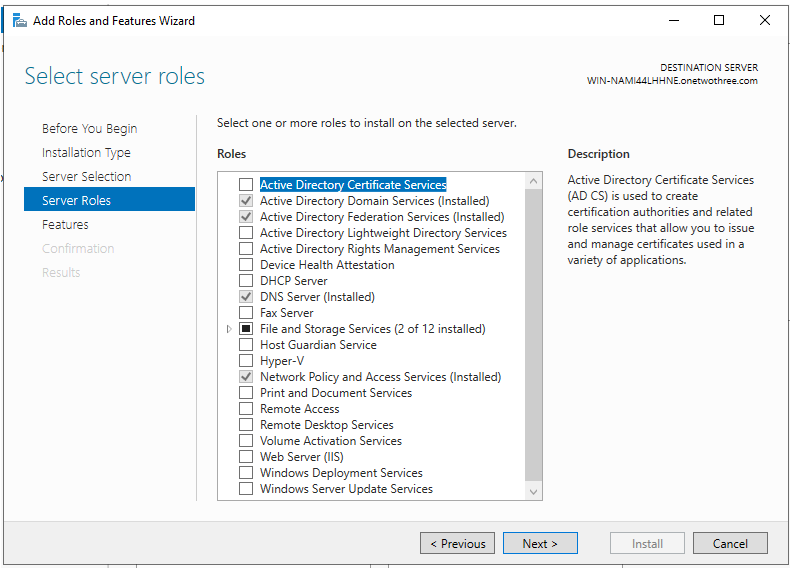


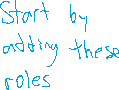
How to Configure TACACS+

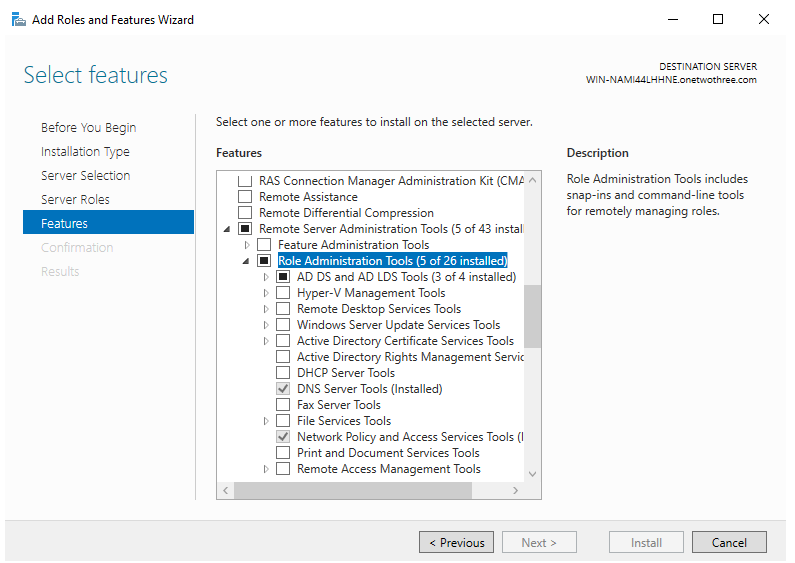




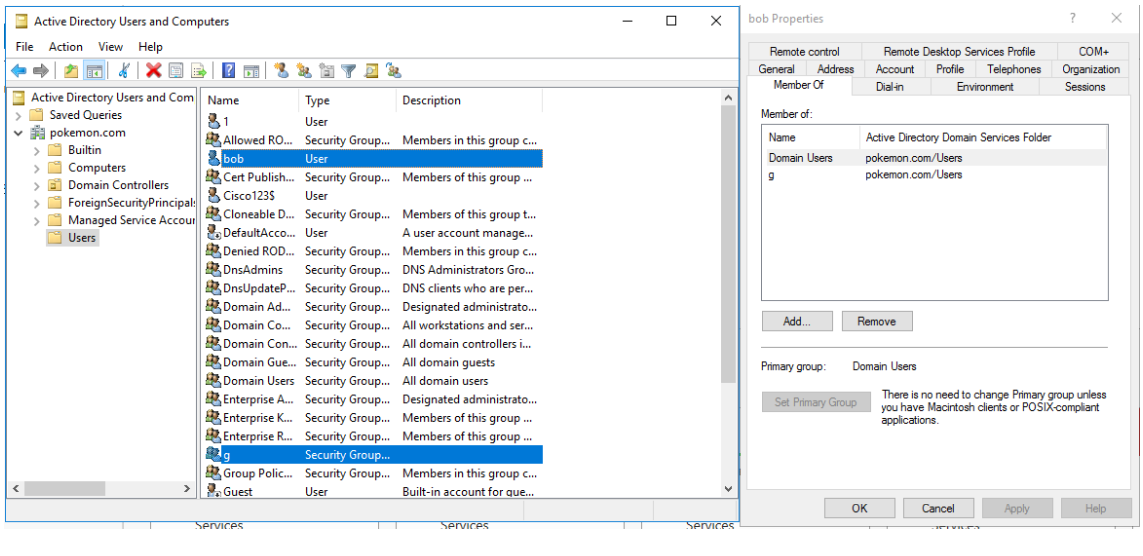
How to Configure RADIUS



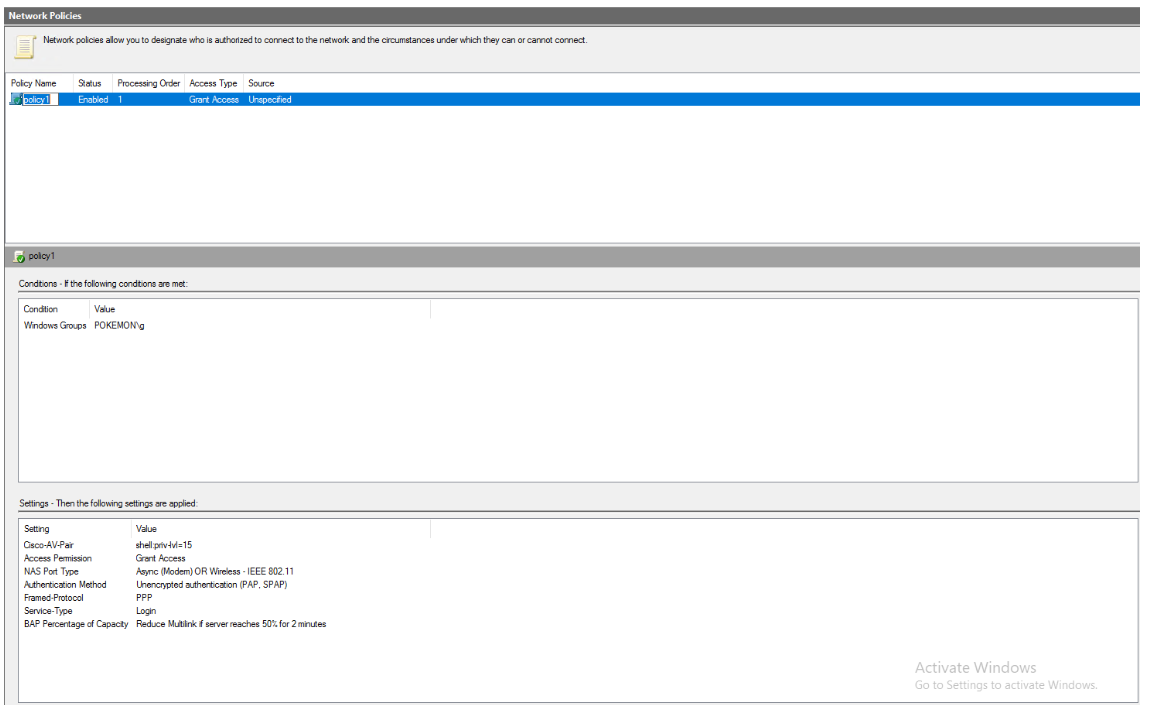




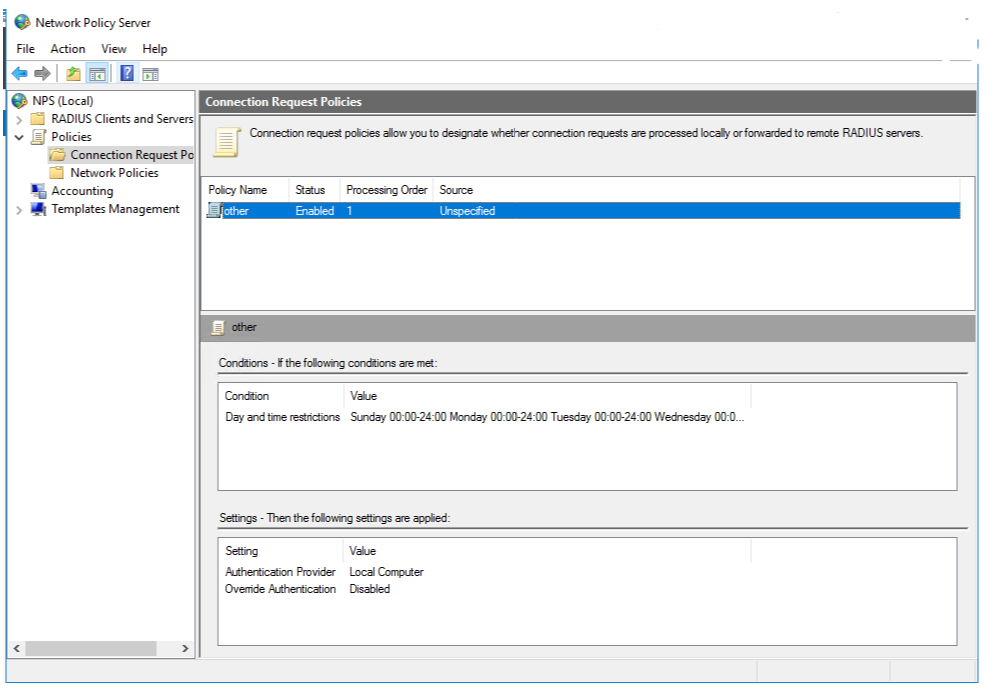


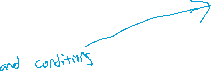


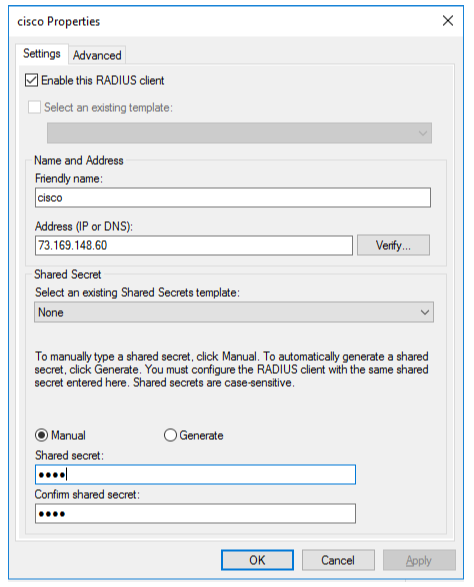




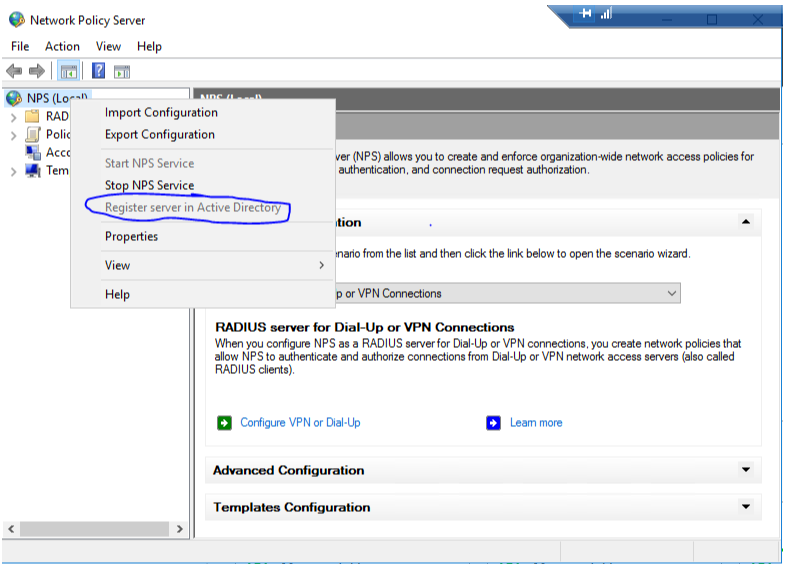


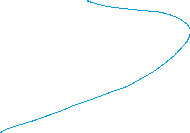
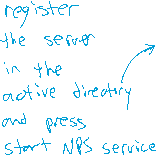












-------------

R1 TACACS+

-------------

Last configuration change at 14:42:41 UTC Wed Nov 27 2002

version 15.2

service timestamps debug datetime msec

service timestamps log datetime msec

no service password-encryption

hostname Tacacs

boot-start-marker

boot-end-marker

aaa new-model

aaa authentication login default group tacacs+ local

aaa authorization config-commands

aaa authorization commands 0 default group tacacs+ none

aaa authorization commands 15 default group tacacs+ none

aaa accounting send stop-record authentication failure

aaa accounting update newinfo periodic 5

aaa accounting exec default start-stop group tacacs+

aaa accounting network default start-stop group tacacs+

aaa session-id common

memory-size iomem 10

ip cef

no ipv6 cef

multilink bundle-name authenticated

voice-card 0

license udi pid CISCO2901/K9 sn FTX1520806V

license accept end user agreement

license boot module c2900 technology-package securityk9

license boot module c2900 technology-package uck9

vtp domain cisco

vtp mode transparent

username ethan privilege 15 password 0 ethan

redundancy

interface Embedded-Service-Engine0/0

no ip address

shutdown

interface GigabitEthernet0/0

ip address dhcp

duplex auto

speed auto

no shut

interface GigabitEthernet0/1

ip address dhcp

duplex auto

speed auto

shut

interface Serial0/0/0

no ip address

shutdown

clock rate 2000000

interface Serial0/0/1

no ip address

shutdown

clock rate 2000000

interface GigabitEthernet0/1/0

no ip address

shutdown

duplex auto

speed auto

ip forward-protocol nd

no ip http server

no ip http secure-server

ip tacacs source-interface GigabitEthernet0/0

tacacs-server host 3.135.213.112

tacacs-server directed-request

tacacs-server key testing123

control-plane

mgcp profile default

gatekeeper

shutdown

line con 0

line aux 0

line 2

no activation-character

no exec

transport preferred none

transport input all

transport output lat pad telnet rlogin lapb-ta mop udptn v120 ssh

stopbits 1

line vty 0 4

exec-timeout 0 0

transport input all

scheduler allocate 20000 1000

end

-------------

R1 RADIUS

-------------

Last configuration change at 22:15:45 UTC Mon Oct 28 2019

version 15.2

service timestamps debug datetime msec

service timestamps log datetime msec

no service password-encryption

hostname Radius

boot-start-marker

boot-end-marker

aaa new-model

aaa authentication login default group radius local

aaa authorization exec default group radius if-authenticated

aaa session-id common

memory-size iomem 10

ip cef

no ipv6 cef

multilink bundle-name authenticated

voice-card 0

license udi pid CISCO2901/K9 sn FTX1704Y038

license accept end user agreement

license boot module c2900 technology-package securityk9

license boot module c2900 technology-package uck9

vtp domain cisco

vtp mode transparent

username ethan privilege 15 password 0 ethan

redundancy

interface Embedded-Service-Engine0/0

no ip address

shutdown

interface GigabitEthernet0/0

ip address dhcp

no shutdown

duplex auto

speed auto

interface GigabitEthernet0/1

no ip add

duplex auto

speed auto

shut

interface Serial0/0/0

no ip address

shutdown

clock rate 2000000

interface Serial0/0/1

no ip address

shutdown

clock rate 2000000

ip forward-protocol nd

no ip http server

no ip http secure-server

ip radius source-interface g0/0

radius-server host 13.82.129.199 key 1234

control-plane

mgcp profile default

gatekeeper

shutdown

line con 0

line aux 0

line 2

no activation-character

no exec

transport preferred none

transport output pad telnet rlogin lapb-ta mop udptn v120 ssh

stopbits 1

line vty 0 4

transport input all

scheduler allocate 20000 1000

end