

Lab 10: An Exploration into Heterogeneity

East Tennessee State University

CSCI 4417/5417: Introduction to System Administration

Spring 2016

Pramod Nepal

Purpose

This lab experienced into Heterogeneity of mixing different operating systems to play with the windows Active Directory domain. A Red Hat Linux instance was used to join to the lastname.loc domain created during previous labs.

Materials

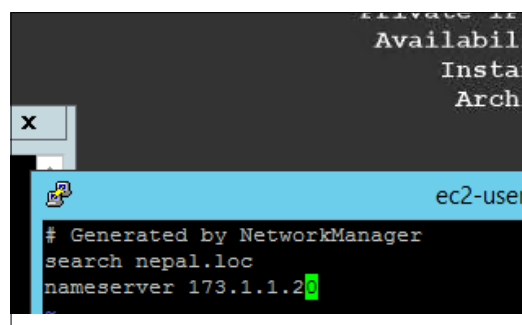
- Lab Instructions
- PuTTY
- Web browser
- AD Domain Server

Procedure and Results

First the domain controller instance was started. Then a new Linux instance was created. Unlike previous labs AWS Red Hat Linux instance was used. VPC used throughout the labs was used. A private IP address of 173.1.1.27 was set for the instance. This would allow the instance to be connected from the domain controller for configuring it as Active Directory member. The instance was named lastnameRHL-ADDS-01. Once the instance was created and started it was connected using Putty using the private IP address. The login user name for the instance was 'ec2-user'.

First the host of the Linux instance was configured to make it point to the Active Directory Domain Controller. This was done by adding the Domain Controller's hostname in /etc/hosts. After this setup, the root user of this Linux instance was given a password. File resolv.conf was configured to search for the Domain Name Server, which in this case was lastname.loc (see Fig. 1).

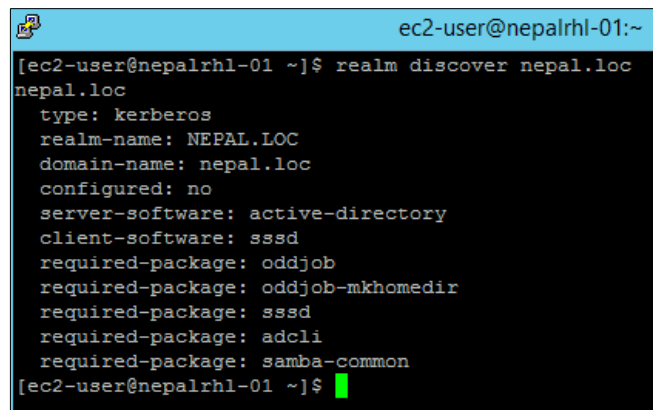
Figure 1: Editing resolv.conf



The name server's IP address was set as 173.1.1.20. If there were multiple name servers their IP addresses would be added in the 'nameserver' line. In order to make sure /etc/resolv.conf did not get reset during boot, chattr command was used to restrict the change. Even a root user would not be able to change the settings of the file. To make sure the hostname was read from /etc/hostname hostname setup configuration parameters were commented out in /etc/cloud/cloud.cfg. Similarly the hostname was set up as lastnamemerhl-01 in /etc/hostname file. This would give a recognizable name for the host in Active Directory. The instance was restarted.

To make the instance search for the Domain Name Server few packages needed to be installed. One of those packages is called realmd. First it was made sure the domain controller could be reached using ping. Then the domain discovery was done using the realm command (see Fig. 2).

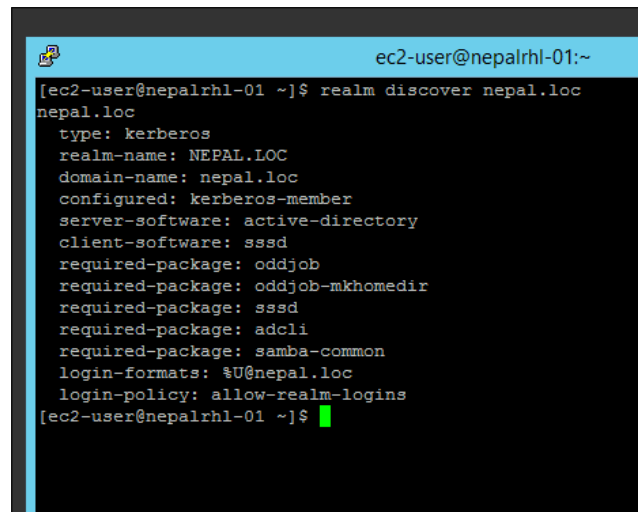
Figure 2: Discover domain

A terminal window with a blue title bar showing 'ec2-user@nepalrhl-01:~'. The command '[ec2-user@nepalrhl-01 ~]\$ realm discover nepal.loc' has been executed. The output is as follows:

```
nepal.loc
type: kerberos
realm-name: NEPAL.LOC
domain-name: nepal.loc
configured: no
server-software: active-directory
client-software: sssd
required-package: oddjob
required-package: oddjob-mkhomedir
required-package: sssd
required-package: adcli
required-package: samba-common
[ec2-user@nepalrhl-01 ~]$
```

The output specified several packages needed to be installed. Once the dependencies were installed the domain was joined using 'realm join lastname.loc' command as an admin. When the domain discovery command was run again it showed the information about the configuration. Few of those settings had changed like configured, and login parameters (see Fig. 3).

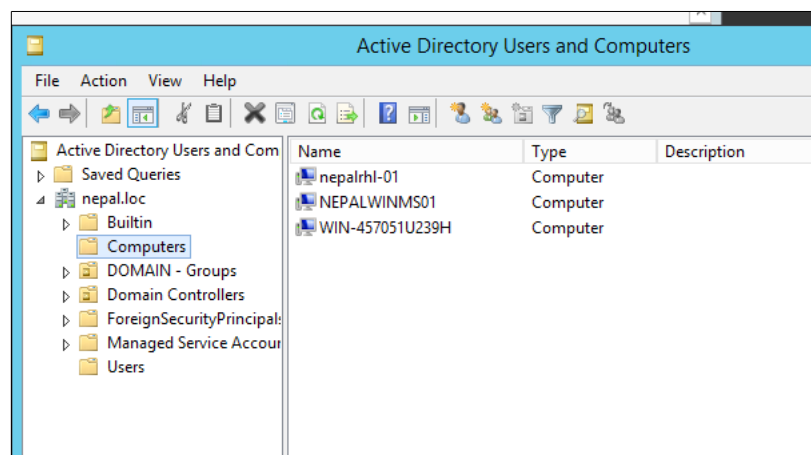
Figure 3: Running discover domain after joining the domain



```
ec2-user@nepalrh1-01:~  
[ec2-user@nepalrh1-01 ~]$ realm discover nepal.loc  
nepal.loc  
type: kerberos  
realm-name: NEPAL.LOC  
domain-name: nepal.loc  
configured: kerberos-member  
server-software: active-directory  
client-software: sssd  
required-package: oddjob  
required-package: oddjob-mkhomedir  
required-package: sssd  
required-package: adcli  
required-package: samba-common  
login-formats: %U@nepal.loc  
login-policy: allow-realm-logins  
[ec2-user@nepalrh1-01 ~]$
```

Next realm login was added for all users using 'sudo realm permit --realm lastname.loc --all' command. This would allow AD users to connect to this instance. After above configuration the instance could be viewed in the AD Users and Computer tool (see Fig. 4).

Figure 4: Joined instance in Active Directory



To allow password authentication login for the user several authentication options needed to be enabled in the Linux instance. Ssh configuration file /etc/ssh/sshd_config was edited and the authentication options were enabled. Two noticeable options were for Kerberos authentication and GSSAPI authentication. The ssh daemon was restarted for these configurations to take effect. In addition, Domain Admins were also added in sudoers file (see Fig 5.). This would allow users like Bob Smith to run Administrative commands using sudo.

Figure 5: Adding Domain Admins to sudoers

```
## Read drop-in files from /etc/sudoers.d (the #  
#includedir /etc/sudoers.d  
ec2-user        ALL=(ALL)        NOPASSWD: ALL  
  
Domain\ Admins@nepal.loc ALL=(ALL) ALL
```

After completing these instructions the domain was connected using Active Directory Administrator (see Fig.

6).

Figure 6: Login as Administrator

```
administrator@nepal.loc@nepalrhl-01:~  
login as: Administrator@nepal.loc  
Administrator@nepal.loc@173.1.1.27's password:  
Creating home directory for Administrator@nepal.loc.  
[administrator@nepal.loc@nepalrhl-01 ~]$
```

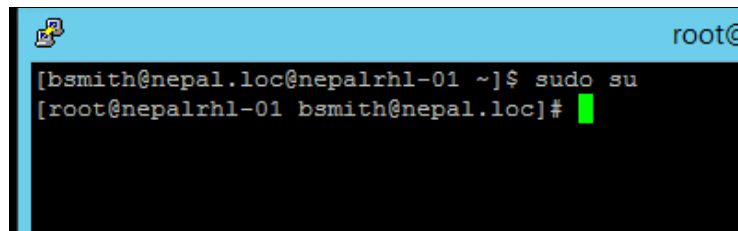
The user could go to the newly created home directory and execute different commands in the shell. Next a non-admin user account named 'agoodall' was used to login. When the user tried to execute admin commands using sudo the console showed a login attempt reporting message. That login attempt was reported to /var/log/secure log file, which could be read using admin login (see Fig. 7).

Figure 7: Admin access attempt reported in log from non Admin user

```
[administrator@nepal.loc@nepalrhl-01 ~]$ sudo su  
We trust you have received the usual lecture from the local System  
Administrator. It usually boils down to these three things:  
  
#1) Respect the privacy of others.  
#2) Think before you type.  
#3) With great power comes great responsibility.  
  
[sudo] password for administrator@nepal.loc:  
[root@nepalrhl-01 administrator@nepal.loc]# sudo tail /var/log/secure  
Apr 19 15:14:14 nepalrhl-01 sshd[12767]: Invalid user agoodall from 173.1.1.20  
Apr 19 15:14:14 nepalrhl-01 sshd[12767]: input_userauth_request: invalid user agoodall [preauth]  
Apr 19 15:14:21 nepalrhl-01 sshd[12767]: Failed password for invalid user agoodall from 173.1.1.20 port 51396 ssh2  
Apr 19 15:14:34 nepalrhl-01 sshd[12767]: Connection closed by 173.1.1.20 [preauth]  
Apr 19 15:14:50 nepalrhl-01 sshd[12769]: pam_sss(sshd:auth): authentication success; logname= uid=0 euid=0 tty=ssh ruse  
rhost=nepal.loc user=agoodall@nepal.loc  
Apr 19 15:14:50 nepalrhl-01 sshd[12769]: Accepted password for agoodall@nepal.loc from 173.1.1.20 port 51396 ssh2  
Apr 19 15:14:50 nepalrhl-01 sshd[12769]: pam_unix(sshd:session): session opened for user agoodall@nepal.loc by (uid=0)  
Apr 19 15:15:07 nepalrhl-01 sudo: pam_sss(sudo:auth): authentication success; logname=agoodall@nepal.loc uid=1745001128  
uid=0 tty=/dev/pts/3 ruser=agoodall@nepal.loc rhost= user=agoodall@nepal.loc  
Apr 19 15:15:07 nepalrhl-01 sudo: agoodall@nepal.loc : user NOT in sudoers ; TTY=pts/3 ; PWD=/home/agoodall@nepal.loc ;  
SER=root ; COMMAND=/bin/su  
Apr 19 15:15:26 nepalrhl-01 sudo: root : TTY=pts/2 ; PWD=/home/administrator@nepal.loc ; USER=root ; COMMAND=/bin/ta  
/var/log/secure  
[root@nepalrhl-01 administrator@nepal.loc]#
```

Another admin user named Bob Smith logged in with bsmith username. Bob could execute admin commands using sudo (see Fig. 8)

Figure 8: Domain Admin accessing root account

A terminal window with a blue title bar. The title bar contains a small icon on the left and the text 'root@' on the right. The terminal text shows a user 'bsmith' at a prompt '[bsmith@nepal.loc@nepalrhl-01 ~]' typing 'sudo su'. The prompt changes to '[root@nepalrhl-01 bsmith@nepal.loc]#' and a green cursor is visible at the end of the line.

```
[bsmith@nepal.loc@nepalrhl-01 ~]$ sudo su
[root@nepalrhl-01 bsmith@nepal.loc]#
```

Observations

Focus of the lab was to enable heterogeneous machines work together in Active Directory. This would allow the IT shops that have different Operating Systems across different departments to be able to talk with each other and access resources. A System Administrator would be able to transparently provide authentication and authorization of resources across different Operating Systems. Several IT shops have embraced the concept of Bring Your Own Device (BYOD). Workers typically bring their own laptops and mobile devices. These devices may contain Linux or Unix based Operating Systems like FreeBSD, Apple Mac, Ubuntu and Android. Since workers pay for their devices, it helps mid size business cut costs (“Pros and Cons of Bringing Your Own Device”). However, the company loses control over how these hardware are used. If not setup properly in the computer network and policy not set for resource access it can risk the security of the organization. Employees download many third-party applications. These applications require (demand) access to different resources on the device. This could give third-party companies access to organization secrets (Greenwald, 2015). As seen in this lab the authentication is done by Kerberos. Kerberos uses encryption and a trusted third party known as Key Distribution Center (KDC) to authenticate users to network services. Once authenticated KDC sends a ticket for that session, which the services look for validating that the user is authenticated and can access the service (“How Kerberos Works”). In this lab we used the public security group in which the source is set as 0.0.0.0/0. Since the instance gets a public IP it can be connected from computers outside the domain. This security group also enables the computer to accept connection from outside. However, the private

security group only accepts connections from 173.1.0.0/16 subnet. This would mean that the instance would only accept requests from within the domain and not outside. After using the private security group, if we had tried to connect to the instance from outside the domain it would reject. For the instance to work the security group should incorporate public IP as source as well.

References

"Pros and Cons of Bringing Your Own Device to Work." *PCWorld*. Web. 25 Apr. 2016.

Greenwald, Adam. "Is BYOD Worth the Security Risk for Your Startup?" *The Huffington Post*. TheHuffingtonPost.com, 03 Feb. 2015. Web. 25 Apr. 2016.

"18.3. How Kerberos Works." *How Kerberos Works*. Web. 25 Apr. 2016.