## Linux Server Project

### Introduction

In most modern IT infrastructure, we find both Linux and Windows services, which must be integrated. One popular use for Linux (among many others) is the Apache web server, which can be used with PHP (a powerful web programming language used by many web applications). However, most networks rely on services such as Microsoft’s Active Directory directory services for authentication and authorization. Since we would not want to have silo’d user accounts on our Linux infrastructure, it is important to understand how Linux could use services such as LDAP (which is supported by Active Directory) to facilitate user management. This would allow a “single sign-on” environment, whereby user’s need only be concerned about one user account/password, which allows them access to all of their resources. This also simplifies network and server administration. This project is intended to demonstrate this concept, and provide hands-on practice with many of the objectives explored throughout this course.

**This project uses elements from various Linux tutorials available on the Internet:**

* <http://php.net/manual/en/ldap.examples-basic.php>
* <http://www.thegeekstuff.com/2015/02/openldap-add-users-groups/>
* <http://stackoverflow.com/questions/171519/authenticating-in-php-using-ldap-through-active-directory>
* <http://www.server-world.info/en/note?os=Fedora_23&p=openldap>

### Background

In this project, you will create an **application server**, which uses **Apache** and **LDAP** to provide some service to user’s on the network. You will set up **VSFTP** in order for a user to copy their files to the server, then Apache will be configured to work with an individual user’s home directory for the web application. Authentication for this application will rely on LDAP, which would be typical in a **single-sign-on** environment (for which we will use **OpenLDAP** to simulate).

**This practical assignment demonstrates concepts in authentication and authorization in section 2, Linux OS security features discussed in section 3, and application services discussed in section 4.**

**Pre-requisite:** *Fedora 23 installed in a VMWare virtual machine (GUI optional) with at least one user named “student” (password “student” recommended). VMWare should be hosted on a Windows 7/8/10 workstation. You should ensure that you are able to connect using TCP/IP from your Windows host to your Linux guest running in VMWare (this may require certain settings in VMWare under the network tab).* **If needed, elevate your permissions (use su), and complete all steps as root (unless otherwise noted)**

### Procedure

1. **Configure the network and host security options for this lab:**
   1. Find your IP Address and your interface name for the interface connecting you to the network (be sure you are not using the loopback interface); **note the IP Address and interface name for later use**:

|  |
| --- |
| # ifconfig |

* 1. Check your firewall status using the firewall-cmd

|  |
| --- |
| # firewall-cmd --state |

* 1. If your firewall is not “running”, enable and start it:

|  |
| --- |
| # systemctl enable firewalld.service  # systemctl start firewalld.service  # firewall-cmd --state |

* 1. Check which firewall zone is being used by the adapter you noted in step 1a by running the following command, and **make a note of the zone name**:

|  |
| --- |
| # firewall-cmd –get-active-zones |

* 1. View services currently enabled in your firewall for your NIC’s zone:

|  |
| --- |
| # firewall-cmd --zone=FedoraWorkstation --list-service |

* 1. For this lab, we will need to allow TCP for FTP, HTTP & LDAP for the zone you noted above (*change in the command below if necessary*):

|  |
| --- |
| # firewall-cmd --zone=FedoraWorkstation --add-service=http  # firewall-cmd --zone=FedoraWorkstation --add-service=ftp  # firewall-cmd --zone=FedoraWorkstation --add-service=ldap  # firewall-cmd --zone=FedoraWorkstation --add-service=ssh |

* 1. You may also want to run these commands with the –permanent option, which will ensure the setting will work after a reboot:

|  |
| --- |
| # firewall-cmd --zone=FedoraWorkstation –permanent --add-service=http |

* 1. To learn more about configuring your firewall, you can read this helpful tutorial: <https://www.digitalocean.com/community/tutorials/how-to-set-up-a-firewall-using-firewalld-on-centos-7>
  2. Next, we will need to set some options for SELinux, to make sure the various services required will work. Start by verifying SELinux is enabled:

|  |
| --- |
| # getenforce |

* 1. If the status is “Enforcing”, you do not need to change the status… however, if you need to set SELinux to ensure it is running and enforcing restrictions, use this command:

|  |
| --- |
| # setenforce 1 |

* 1. We will need to set two settings in SELinux for this lab… first, we need to make sure users are permitted to access their home directories using FTP, and we will need to make sure that LDAP access is permitted from Apache. We can use the **semanage** command to see all the options available in SELinux, and pipe the list into grep to see only those settings referencing HTTP and FTP:

|  |
| --- |
| # getsebool -a | grep 'ftp\|http' |

* 1. The settings we will need are “httpd\_enable\_homedirs” (allows the configuration of Apache to use home directories for web content), “ftp\_home\_dir”, and “httpd\_can\_connect\_ldap”

|  |
| --- |
| # setsebool -P httpd\_enable\_homedirs on  # setsebool -P httpd\_can\_connect\_ldap on  # setsebool -P ftp\_home\_dir on |

* 1. Save the output from the following command as “**Deliverable1m**”

|  |
| --- |
| # getsebool -a | grep 'httpd\_can\_connect\_ldap\|ftp\_home'; firewall-cmd --list-services |

1. **Install/Configure vsftpd**
   1. Use **dnf** to install **vsftpd**

|  |
| --- |
| # dnf install vsftpd  # systemctl enable vsftpd.service |

* 1. Edit vsftpd’s configuration file using “vi” (**# vi /etc/vsftpd/vsftpd.conf**) to allow local users to log in by ensuring this line is set, and not commented with a “#”:

|  |
| --- |
| Vsftpd.conf |
| local\_enable=YES |

* 1. Start **or** Restart your VSFTP service:

|  |
| --- |
| # systemctl start vsftpd.service  # systemctl restart vsftpd.service |

1. **Test FTP from your Windows host**
   1. Open Windows Explorer (explorer.exe). Note, this is *NOT* Internet Explorer (iexplorer.exe). *If you are unsure how to start Windows Explorer, this is the same as “file explorer”, the application used to browse files on your computer.*
   2. In the address bar (where you would normally find the path to a file), type **ftp:\\userid:password@ipaddress** (replace userid with your user account, password with your password, and ipaddress should be the IP of your Fedora system running in the VM guest)
   3. You should see your user’s home directory, and you should be able to create a test file. Once you copy a test file, use the ls command to show a listing of the contents of the user’s home directory (from your terminal on your Linux system) and save this output as “**Deliverable3C**”.
2. **Install/Configure Apache**
   1. Use **dnf** to install **httpd**

|  |
| --- |
| # dnf install httpd  # systemctl enable httpd.service  # systemctl start httpd.service |

* 1. Use httpd’s config files to allow the userDir option, so when users upload files to **/home/student/public\_html/** they will be accessible from <http://server_address/~student>. To make this change, use **vi** to edit the **/etc/httpd/conf.d/userdir.conf** configuration file, so the lines appear as below:

|  |
| --- |
| ***userdir.conf*** |
| <IfModule mod\_userdir.c>  #  # UserDir is disabled by default since it can confirm the presence  # of a username on the system (depending on home directory  # permissions).  #  #UserDir disabled  #  # To enable requests to /~user/ to serve the user's public\_html  # directory, remove the "UserDir disabled" line above, and uncomment  # the following line instead:  #  UserDir public\_html  </IfModule> |

* 1. Create the **public\_html** directory in your home, and ensure that you are using the correct file permissions on **/home**, **/home/student**, and /**home/student/public\_html.** You can use the commands below to complete this task; be sure to substitute “userid” with the user id you are using with your Linux machine:

|  |
| --- |
| # mkdir /home/userid/public\_html  # chmod 755 /home  # chmod 755 /home/userid  # chmod 755 /home/userid/public\_html  # chown userid:userid /home/userid/public\_html |

* 1. Restart Apache using the following command:

|  |
| --- |
| # systemctl restart httpd.service |

* 1. Using “notepad” on the host PC, create a file called “**index.html**” with the following contents:

|  |
| --- |
| ***Index***.***html*** |
| <html>  <body>  Test Page (**your full name here**)  </body>  </html> |

* 1. Upload the file using FTP to your user’s **public\_html** directory*. Recall the steps from section 3 to use FTP with Windows Explorer.*
  2. Browse the appropriate web address to view the test page, and save the screen as “**deliverable4G**” (http://*server\_IP\_Address*/~userid/index.html)

1. **Install LDAP**
   1. Use **dnf** to install **openldap-servers** and **openldap-cleints**

|  |
| --- |
| # dnf install openldap-servers openldap-clients |

* 1. Use the following commands to copy the example config files, set permissions, and start LDAP:

|  |
| --- |
| # cp /usr/share/openldap-servers/DB\_CONFIG.example /var/lib/ldap/DB\_CONFIG  # [chown](http://www.server-world.info/en/command/html/chown.html) ldap. /var/lib/ldap/DB\_CONFIG  # [systemctl](http://www.server-world.info/en/command/html/systemctl.html) start slapd  # [systemctl](http://www.server-world.info/en/command/html/systemctl.html) enable slapd |

* 1. Now that LDAP has started, we can import some schemas to get started... use the following commands to do this:

|  |
| --- |
| # ldapadd -Y EXTERNAL -H ldapi:/// -f /etc/openldap/schema/cosine.ldif  # ldapadd -Y EXTERNAL -H ldapi:/// -f /etc/openldap/schema/nis.ldif  # ldapadd -Y EXTERNAL -H ldapi:/// -f /etc/openldap/schema/inetorgperson.ldif |

* 1. Next, use the **slappasswd** command to get a password hash. (password “student” is recommended) Be sure to save the hash output generated when you run this command, for later use (highlighted below).

|  |
| --- |
| # slappasswd  New password: student  Re-enter new password: student  {SSHA}sxVR97KcOsyVmFpmfKMT34lhDBXwyWOW |

* 1. Next, we need to use LDAP to set some parameters… on your Windows host, create a file called **chdomain.ldif** with the contents below, and use FTP to upload to your home directory on the Linux guest VM.

|  |
| --- |
| dhdomain.***ldif*** |
| dn: olcDatabase={1}monitor,cn=config  changetype: modify  replace: olcAccess  olcAccess: {0}to \* by dn.base="gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth"  read by dn.base="cn=Manager,dc=localhost,dc=localdomain" read by \* none  dn: olcDatabase={2}mdb,cn=config  changetype: modify  replace: olcSuffix  olcSuffix: dc=localhost,dc=localdomain  dn: olcDatabase={2}mdb,cn=config  changetype: modify  replace: olcRootDN  olcRootDN: cn=Manager,dc=localhost,dc=localdomain  dn: olcDatabase={2}mdb,cn=config  changetype: modify  add: olcRootPW  olcRootPW: {SSHA}sxVR97KcOsyVmFpmfKMT34lhDBXwyWOW  dn: olcDatabase={2}mdb,cn=config  changetype: modify  add: olcAccess  olcAccess: {0}to attrs=userPassword,shadowLastChange by  dn="cn=Manager,dc=localhost,dc=localdomain " write by anonymous auth by self write by \* none  olcAccess: {1}to dn.base="" by \* read  olcAccess: {2}to \* by dn="cn=Manager,dc=localhost,dc=localdomain " write by \* read |

* 1. Next, run the command to import these changes to your LDAP server:

|  |
| --- |
| # ldapmodify -Y EXTERNAL -H ldapi:/// -f /home/userid/chdomain.ldif |

* 1. You will also need to set your base domain; create a file called **basedomain.ldif** with the following contents, and upload to your home directory use FTP:

|  |
| --- |
| Basedomain.***ldif*** |
| dn: dc=localhost,dc=localdomain  objectClass: top  objectClass: dcObject  objectclass: organization  o: Localdomain Localhost  dc: localhost  dn: cn=Manager,dc=localhost,dc=localdomain  objectClass: organizationalRole  cn: Manager  description: Directory Manager  dn: ou=People,dc=localhost,dc=localdomain  objectClass: organizationalUnit  ou: People  dn: ou=Group,dc=localhost,dc=localdomain  objectClass: organizationalUnit  ou: Group |

* 1. Use the following command to add these records to LDAP:

|  |
| --- |
| # ldapadd -x -D cn=Manager, dc=localhost,dc=localdomain -W -f /home/userid/basedomain.ldif |

* 1. Finally, add some new users to your server by creating the two LDIF files below, which you will use to create new “entries” in your LDAP directory:
     1. Optionally, you can change “adam” and “eve” to any user names you like.
     2. The password can be the same hash you used before; this will set that user’s password to the same value you used for the “Manager” account you created previously in the initial configuration for LDAP at step 5d-5e. However, you can use the slappasswd command to get different passwords for these users.
     3. Be sure that one user’s description is “Directory Manage” and the other is “Directory User”.
     4. Once you create these files, use FTP to upload them to your home on your Linux guest.

|  |
| --- |
| ***User1.ldif:*** |
| dn: uid=eve,ou=People,dc=localhost,dc=localdomain  objectClass: top  objectClass: account  objectClass: posixAccount  objectClass: shadowAccount  cn: eve  uid: eve  uidNumber: 16859  gidNumber: 100  homeDirectory: /home/eve  loginShell: /bin/bash  gecos: eve  userPassword: {SSHA}sxVR97KcOsyVmFpmfKMT34lhDBXwyWOW  shadowLastChange: 0  shadowMax: 0  shadowWarning: 0  description: Directory Manager |
| ***User2.ldif*** |
| dn: uid=adam,ou=People,dc=localhost,dc=localdomain  objectClass: top  objectClass: account  objectClass: posixAccount  objectClass: shadowAccount  cn: eve  uid: eve  uidNumber: 16859  gidNumber: 100  homeDirectory: /home/adam  loginShell: /bin/bash  gecos: adam  userPassword: {SSHA}sxVR97KcOsyVmFpmfKMT34lhDBXwyWOW  shadowLastChange: 0  shadowMax: 0  shadowWarning: 0  description: Directory User |

* 1. To import each file, use the following command:

|  |
| --- |
| # ldapadd -x -D cn=Manager, dc=localhost,dc=localdomain -W -f /home/userid/user1.ldif  # ldapadd -x -D cn=Manager, dc=localhost,dc=localdomain -W -f /home/userid/user2.ldif |

* 1. To test LDAP, download and run the LDAPAdmin tool for Windows, which will let you browse the records in LDAP from your Windows host (<http://www.ldapadmin.org/download/ldapadmin.html>)
     1. To browse with “anonymous” mode, use the Base “**dc=localhost,dc=localdomain**”
     2. To browse using the admin account we created, the username will be: **cn=Manager,dc=localhost,dc=localdomain**, and the password should be “student” (*unless you elected to use a different password at steps 5d-5e*)
  2. Once you have been able to successfully browse LDAP, take a screen image showing your users, and save as “deliverable5L”

1. **Use PHP to create a sample application that uses LDAP**
   1. Install **php** and the **php-ldap** module using **dnf,** then restart Apache:

|  |
| --- |
| # dnf install php php-ldap  # systemctl restart httpd |

* 1. Verify that PHP is working:
     1. Create a file called “**phptest.php**”, and upload to your user’s **public\_html** directory using FTP:

|  |
| --- |
| ***phptest.php*** |
| <?php phpinfo() ?> |

* + 1. Use your browser, and test the PHP page (http://Server\_IP\_Address/~userid/phptest.php)
    2. Verify that LDAP is one of the installed modules for PHP
    3. Save the screen showing your PHP test page as “deliverable6B”.
  1. Next, test LDAP from PHP:
     1. Create a file called “**ldap.php**” in your **public\_html** directory, with the following contents. (If you elected to use a different password that “student” for the “manager” account, be sure to substitute the correct password.)

|  |
| --- |
| ***ldap.php*** |
| <?php  error\_reporting(E\_ALL);  ini\_set('display\_errors', 1);  echo "<h3>LDAP query test</h3>";  echo "Connecting ...";  $ds=ldap\_connect("localhost"); // must be a valid LDAP server!  echo "connect result is " . $ds . "<br />";  ldap\_set\_option($ds, LDAP\_OPT\_PROTOCOL\_VERSION, 3);  if ($ds) {  echo "Binding ...";  $r=ldap\_bind($ds,"cn=Manager,dc=localhost,dc=localdomain","student"); // this is an "anonymous" bind, typically  // read-only access  echo "Bind result is " . $r . "<br />";  echo "Searching for (sn=\*) ...";  // Search surname entry  $sr=ldap\_search($ds, "dc=localhost,dc=localdomain","cn=\*");  echo "Search result is " . $sr . "<br />";  echo "Number of entries returned is " . ldap\_count\_entries($ds, $sr) . "<br />";  echo "Getting entries ...<p>";  $info = ldap\_get\_entries($ds, $sr);  echo "Data for " . $info["count"] . " items returned:<p>";  echo '<hr />';  for ($i=0; $i<$info["count"]; $i++) {  echo "dn is: " . $info[$i]["dn"] . "<br />";    echo '<pre>';  //print\_r($info[$i]);  foreach ($info[$i] as $key=>$val) {  if (!is\_numeric($key)) {  //echo $key."\n";  if (is\_array($val)) {  foreach ($val as $k=>$v) {  if (is\_numeric($k)) {  echo $key.": ";  echo $v."\n";  }  }  } else {  echo $val."\n";  }  }  }  echo '</pre>';  echo '<hr />';  }  echo "Closing connection";  ldap\_close($ds);  } else {  echo "<h4>Unable to connect to LDAP server</h4>";  }  ?> |

* + 1. Check that this works by browsing to <http://server_address/~student/ldap.php>
    2. Save the screen as “deliverable6C”
  1. Next, create your basic PHP app that uses LDAP by creating a file called “**ldap\_app.php**” with the following contents, and upload to your **public\_html** directory on the Linux guest VM:

|  |
| --- |
| ***ldap\_app.php*** |
| <?php  error\_reporting(E\_ALL);  ini\_set('display\_errors', 1);  ?>  <form method="post" action="" name="ldapform">  <br />UserID: <input type="text" value="" name="uid" /> <br />  Password: <input type="password" value="" name="pass" /> <br />  <input type="submit" name="submit" value="submit" /> <br />  </form>  <?php  echo "<h3>LDAP Test App</h3>";  echo "Connecting ...";  $ds=ldap\_connect("localhost"); // must be a valid LDAP server!  echo "connect result is " . $ds . "<br />";  ldap\_set\_option($ds, LDAP\_OPT\_PROTOCOL\_VERSION, 3);  if ($ds) {  echo "Binding ...";  if ($r=ldap\_bind($ds,"uid=".$\_POST['uid'].",ou=People,dc=localhost,dc=localdomain",$\_POST['pass'])) {  echo "Bind result is " . $r . "<br />";  echo "Searching for (uid=".$\_POST['uid'].") ...";    $sr=ldap\_search($ds, "dc=localhost,dc=localdomain","uid=".$\_POST['uid']);  echo "Search result is " . $sr . "<br />";  echo "Number of entries returned is " . ldap\_count\_entries($ds, $sr) . "<br />";  echo "Getting entries ...<p>";  $info = ldap\_get\_entries($ds, $sr);  echo "Data for " . $info["count"] . " items returned:<p>";  echo '<hr />';  for ($i=0; $i<$info["count"]; $i++) {  echo "dn is: " . $info[$i]["dn"] . "<br />";  if ($info[$i]['description'][0] == 'Directory Manager') {  echo "<strong>You are an admin, you are special.</strong>";  } else {  echo "<strong>You are nobody. You are not special.</strong>";  }  echo '<pre>';  //print\_r($info[$i]);  foreach ($info[$i] as $key=>$val) {  if (!is\_numeric($key)) {  //echo $key."\n";  if (is\_array($val)) {  foreach ($val as $k=>$v) {  if (is\_numeric($k)) {  echo $key.": ";  echo $v."\n";  }  }  } else {  echo $val."\n";  }  }  }  echo '</pre>';  echo '<hr />';  }  } else {  echo '<h2>Authentication Failed</h2>';  }  echo "Closing connection";  ldap\_close($ds);  } else {  echo "<h4>Unable to connect to LDAP server</h4>";  }  ?> |

* 1. Test your simple PHP Web Application that uses LDAP for user authentication by using your browser on the host VM to browse to the web page you created. Note, that you will be able to login using the credentials “eve” and “adam” using their password (“student”, unless you elected to use a different password at step 5.i.iv). Once logged in, you will see the user’s information, and a message based on their “description”. Note how this information could be used in a real world application that uses LDAP for authentication/authorization. Save a screen showing this working web application and save as “deliverable6E”.