User Authentication – Overview Sheet

Users and Authentication

Users are the representation of each person who can be identified by the computer, each user has a user identifier (uid) number that must be unique. Having a user doesn't necessaryly mean you can access anything, but it does mean that the computer knows who you are and has verified who you are through passwords or other security barriers. When you log in to an Ubuntu computer, you are identifying yourself with your password.

The act of identification and verification of a user is called “Authentication”

Groups and Authorisation

Once the computer knows who you are, it will want to know what you are allowed to do on the computer. This is done through groups. Each group has a unique group identifier (gid) and is an effective permission to access or change certain things and each user is a member of a number of different groups. For instance, if you are not in the games group on Ubuntu, you shouldn't be allowed to play any of the games that are correctly set to that permission.

The act of giving permission though various mechanisms is called “Authorisation”

ACLs, Access Control Lists

ACLs as a security model are more refined ways of specifying what a user can access. By default the Ubuntu system doesn't expose it's ACL system to users, but it's worth understanding what they are in case you come across them.

The ACL is a simple list which mentions each of the objects (for instance files, hardware devices or services) and documents the access control for each authenticated user. Each time the object is accessed the system will check the relevant ACL to confirm that the action is permitted before continuing.

PAM (Access Management)

Because there are different ways to authenticate users, there is a service on the modern computer called PAM and each method of authentication is called a pam module. For instance the normal logon procedure uses the 'passwd' pam module and you can see this and other authentication modules configured in the /etc/pam.d/ directory. Each file is a configuration for a different kind of authentication and more can be installed.

This allows your organisation's network to allow people to log in via different methods other than creating a single user on every single computer in the company. It allows permissions and access to be managed centrally and for flexibility of set up in case you need other verification such as thumb prints (biometric) or some other method.

Adding and Removing Local Users

There is a list of local users configured in /etc/passwd, this contains all the users who can log in and what their settings are. (name, default shell, uid etc), but it's very rare that we would ever have to modify this file directly and instead use the `useradd`, `userdel`, `usermod` and `users` commands on the command line.

```
useradd [username]
userdel [username]
```

Adding and Removing of Groups

There is a list of local groups configured in /etc/group and like users, we don't want to modify this file directly. Instead we want to use the groupadd and groupdel commands. As always, do read the full man page for all options and syntax as these tools are often very powerful.

```
groupadd [group-name]
groupdel [group-name]
```

File Permissions

Each file and directory has a set of permissions that allows read, write and execute/search access to different users and groups. To look up the existing file permission of a file, use the `ls -l` command:

```
ls -l [filename]
```

This will show you the filename or a list of files in the given directory like so:

```
drwxr-xr-x  2 teacher teachers     16384 2009-08-18 20:00 Desktop
```

The first section is the permissions, type (d=directory), user permissions (rwx), group permissions(r-x), other permissions(r-x). Where there is a letter such as x, that means the file or directory has that permission. Where there is a dash instead of the expected letter, that means the file does not have that permission. The type flag is normally a dash for files, d for directory, s for symbolic link and so on.

You can use `chmod` to change these permissions as shown in these examples:

```
chmod 755 [filename]   # Set the permission exactly to rwxr-xr-x
chmod o-rw [filename]  # Remove the read and write permissions from other
chmod a+x [filename]   # Add execute perm to all (user, group and other)
chmod u+X [filename]   # Add search perm only if a directory to user.
```

You can also change the owner and group of the file or directory using 'chown' as shown in the command line basics class in this course.

Kerberos

Kerberos was created by MIT and is an example of a centrally managed service which manages user authentication to the wider system. It's sometimes called “Single Logon” since it can be used for logging people on to many website services at the same times as logging onto their desktop workstations.

The configuration and setup of Kerberos is not yet easy or simple to set up, so this class will not cover step by step procedure. Although when your setting up a kerberos network you have to make sure that all the computers have the same date and time, so be sure to use the NTP protocol (Network Time) to synchronise all the clocks.

You will also need the krb5-kdc, krb5-config, krb5-user, krb5-clients, and krb5-rsh-server packages installed in the correct places and /etc/krb5.conf and /etc/krb5kdc/kdc.conf correctly configured before you can start adding users. Please check out one of the many good internet guides to setting up kereros for full information.

OpenLDAP

LDAP (Lightweight Directory Access Protocol) is a directory information service for users. Once a user has been identified then they can have a set of information applied to them. This includes permissions, telephone numbers, photographs and other information. OpenLDAP can thus be used for authorisation data, allowing the centralisation of what users are allowed to do on the network no matter what computer they log in on or even what different system they use.

Each block of information is configured and linked together by a definition similar to a relational database. These blocks can define interconnected sections of the directory, such as companies, contacts, calendar events and so on. This allows LDAP to be used as a shared address book as well as a useful way to centralise authorisation data.

Installation and Configuration

Be sure that before you begin that your computer has a domain name that can be used for the directory services that all clients will be able to see. The directory uses the domain name to structure the data access and by default the domain is example.com. For testing, you can configure example.com to resolve to localhost.

You will need to install the slapd and ldap-utils packages on your server computer using apt-get.

Using the supplied example.ldif file, we are going to add this information about a user, group and base configuration to our new OpenLDAP directory making sure to stop the sldapd daemon service first:

```
sudo /etc/init.d/slapd stop
sudo slapadd -l example.ldif
sudo /etc/init.d/slapd start
```

We will then test to see if the directory information has been entered correctly using this search command from the slapd-utils package:

```
ldapsearch -xLLL -b "dc=example,dc=com" uid=john sn givenName cn
```

You should get back a directory entry for our example John Doe.

PAM and OpenLDAP

It is possible to now use OpenLDAP and the PAM module “pam_ldap” to configure your client network to both authenticate and authorise based on your networked directory services. Simply install libpam-ldap and configure the module to connect to your OpenLDAP server.