**Computer Security Lab 3**

In this lab we are going to talk about binary numbers, decimal numbers, hexadecimal numbers, how the Linux file system operates and how to create users.

**Preface Decimal, Binary and hexadecimal Introduction**

**Decimal & Binary Introduction**

Before we even get into conversation about memory let’s briefly talk about binary. Binary is a number system composed of 0’s and 1’s that no number greater than 1 and less than 0. Binary is also read from left to right but computed right to left. In order to solve for binary we apply the following formula: some number n starting at 0 with some integer. In real life we use the decimal number system (also known as the base 10 number system numbers 0-9). I understand that this might seem confusing at first but once we do a couple of examples I can assure you it won’t be confusing anymore:

For this example let’s prove that the number 5053 is actually 5053. Earlier on, I stated that in today’s society we use the decimal number system to represent integers, and so they are base 10 In order to prove the number [5053] is 5053, we can dissect it down into powers of 10. Refer to the steps below:

We can write [5053] by breaking it down into the thousands, hundreds, tens and one columns.

We can also write it as the following:

Thus we are able to prove that these are equal because when we expressed 5053 in polynomial form we get that they are equal:

Notice how we started at on the most right integer and multiplied it by 3 which is the right most value in 5053.

We do the same thing with the next value we increment the 0 to 1 so now we have which is then multiplied by 5 and we have 50. Thirdly we do the same thing we increment the digit from 1 to 2 so now we have and multiply that by 0 which is well 0. Lastly we do the same thing with our 2 becomes a 3 and we multiply it by 5 and we get 5000. So when we add up all the digits we get the result of 5053.

The above example was the decimal number system also known as base 10 as I described. We won’t be going into too much more detail about it but it is good to know. We are really focused on the **Binary Number System**. The binary number system is the most important and fundamental number system in the realm of digital and computer based electronics. Binary number systems as I described work only in powers of 2 and are only comprised of 0’s and 1’s

In case you aren’t sure why we perform our computation from left to right we do this because in the world of computing we call the right most bit (a bit is a binary digit) the Least Significant Bit or LSB and the left hand most bit being called the Most Significant Bit or the MSB and we can visually see this in the table below:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| MSB | Binary Digit | Binary Digit | Binary Digit | Binary Digit | Binary Digit | Binary Digit | Binary Digit | Binary Digit | LSB |
|  |  |  |  |  |  |  |  |  |  |
| 512 | 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |

Let’s work through an example of figuring out a binary number in decimal. The example is illustrated below:

Notice how we had the binary number “1001” and we set it equal to the powers of 2 starting from right to left. The least significant bit correlates to the power which in our example above is a 1 so we multiply, then we proceed onto the next bit which is. After we do this we then repeat the following pattern for the next digits. The second bit is this means that the 2nd bit is a 0. Then lastly the third bit is this means that 3rd bit is 8. So if we add up the following values we get the value which is in base 10. Hence the binary value 1001 base 2 equals 9 base 10.

It is perfectly understandable for binary to be a little intimidating especially if this is your first time trying to understand it. But we are going to do another example below. But first the key takeaways from this are that: (1) binary is base 2 which means all values start from (where n is the right most significant bit), (2) we have to remember to multiply our current value of by the given bit in the binary stream whether it’s a 1 or a 0(in our case x is just some value of the bit’s location). Let’s go through one more example to really get our hands dirty with binary:

In this example we are going to deal with the binary number and convert it into a decimal number (or base 10). The first thing to do is start from the Least Significant Bit and multiply it by the power of. This pattern as we know will continue to the most significant bit:

In order to understand the following example above I did what did for the first example laying out the powers of 2 which correspond to each bit. In order to continue with the theme of binary we will start from the right most significant bit which happens to be the least significant bit (and yes it is the least significant bit regardless of what the next bit is) so we have: , next , next, next and lastly . Then we add the following values and get the value 29 in base 10. This means that the binary value “11101” is the same as 29 base 10.

I know what you are probably thinking why are we learning this weird number system? Well the answer is pretty simple computers use binary and hexadecimal numbers in order to get values from memory to the user and also allow us to type values from the keyboard to the monitor. That is a brief overview of where it is used for a more in-depth look feel free to do more research on the topic! (Because if you look you will see binary is used everywhere in the realm of digital electronics and computers). Below you will see a couple questions look through what we previously discussed and try the next four questions:

The last thing I want to talk about in regards to binary & decimal is converting decimal to binary. I understand that fact that we can just try and put the numbers together and get the value 23 but what if we needed a computer program to figure it out? We can’t just tell the computer verbally hey look for then binary values which make the number 23 up. Instead when binary was developed someone invented an algorithm which allows us to do this.

The algorithm to convert decimal to binary is to (1) divide the number by 2, (2) get the integer quotient for the next iteration of the algorithm,(3) get the remainder of the binary digit (it will be a 1 or a 0 because in the remainder column is where our binary digit is) and lastly (4) repeat the steps until the quotient is 0. This might seem confusing but I promise you it isn’t too bad. In order to fully understand this let’s take an example below and walk through it:

Let’s covert the value to binary:

|  |  |  |  |
| --- | --- | --- | --- |
| Value/2 | Quotient | Remainder | Bit number |
| 25/2 | 12 | 1 | 0 |
| 12/2 | 6 | 0 | 1 |
| 6/2 | 3 | 0 | 2 |
| 3/2 | 1 | 1 | 3 |
| 1/2 | 0  | 1 | 4 |

The result is starting from the bottom up within the remainder column hence the value is: 11001

In the first row we took the initial decimal value “25” divided it by 2 and we get the value 12.5 well that .5 becomes a remainder within the realm of binary so we mark a 1 within in our Remainder column for that value. Then we put the quotient of 12 in the quotient column (this quotient will be our new value) and lastly we label the bit number so it’s easier to visualize the Least Significant Bit and the Most Significant Bit in this case it is bit number 0 so it is our right most digit or the Least Significant Bit.

In the second row we took the former quotient which was 12 and divided it by 2 and we got the new value 6 which has no remainder so we mark a “0” within the remainder column and write in the quotient is 6. Then we put that is bit number 1 within that row.

In third row we took the former quotient 6 and divided it by 2 and we got a new quotient which is 3 and the remainder doesn’t exist hence we have a 0 there and this is bit number 2.

In the fourth row we took the former quotient which is a 3 and divided it by 2 which gives a result of 1.5 so that means there is a remainder of 1 and the quotient is 1 and this bit number is 3.

In the fifth row we use the prior quotient which is 1 and divide it by 2 and we get the value of 0.5 which means there is no quotient and only a remainder and this is bit number 4.

Hence the final value starting from bottom to top the MSB to LSB is: 11001 which is equal to 25 and we can prove it by the following:

As you can see we were able to go through the entire process from decimal to binary and then back to decimal thus proving to us that

This conversion process isn’t too difficult if you take your time read and understand and there is absolutely no shame in going back and reading through the labs and asking questions.

Let’s convert the number to binary:

|  |  |  |  |
| --- | --- | --- | --- |
| Value/2 | Quotient | Remainder | Bit number |
| 180/2 | 90 | 0 | 0 |
| 90/2 | 45 | 0 | 1 |
| 45/2 | 22 | 1 | 2 |
| 22/2 | 11 | 0 | 3 |
| 11/2 | 5 | 1 | 4 |
| 5/2 | 2 | 1 | 5 |
| 2/2 | 1 | 0 | 6 |
| 1/2 | 0 | 1 | 7 |

In the first row we take the value 180 which is our decimal value and divide it by 2 to get 90 with no remainder (so no .5) then that bit number is 0.

In the second row we take the new quotient value which is 90 and divide it by 2 and get the new quotient value which is 45 with a remainder of 0 since it has no “.5” and then that bit number is 1.

In the third row we use the new quotient value 45 and divide it by 2 and we get 22.5 and since the .5 is present we know that we have a remainder of 1 and then our new quotient is 22. Then this is bit number 2 because it is our second bit.

In the fourth row we use the new quotient 22 and divide it by 2 and get the new quotient 11 and since this has no remainder it is a 0 in that column. Then the bit number has a 3 in that column.

In the fifth row we use the new quotient 11 and divide it by 2 and we get 5.5 and since there is a “.5” in there we get a 1 in the remainder column and 5 as the new quotient value. Then the bit number is 4 for this row.

In the sixth row we use 5/2 and we get the value 2.5 which has a .5 in it so we know there is a remainder for it and we know the new quotient value is 2. This is bit number 5

In the seventh row we use 2/2 and we get the value 1. This means there is no remainder and a quotient value of 1 within bit number 6.

In the eighth row we use 1/2 which is .5 which leads to a value of 1 for the remainder and 0 for the quotient. This is bit number 7.

Then after this is done we start from the bottom and list our remainder column and get the following: 10110100 which is 180 in binary.

**You’ve been reading a lot take a 2 minute break**

**Hexadecimal**

Hexadecimal is just a fancy word for base 16.Hexadecimal use 16 digits they are 0-9 and then the letters A,B,C,D,E and F these letters take the place of numbers 10-15 respectively.(A=10, B=11, C=12, D=13, E=14, F=15). The same rule applies where we start from the right most value and as we move to the left we get bigger (in this case we get 16 times bigger). All the general concepts stay the same except instead of “2” we are using “16”. In order to see how this works let’s go through a given example:

In the first example we are going to convert “8E” to decimal.

We first need to figure out what E is we know from looking above that it is 14 because A is 10, B is 11, C is 12, D is 13, E is 14 and F is 15. After we have the value E we then can set up our hexadecimal to decimal formula:

In the example above we used the least significant bit “E” which is 14 and multiplied it by since that is that least bit’s spot value hence why we have .After this we used the 8 which is the most significant bit spot value which is and we multiplied it by 8 hence why we have.

In the second example we are going to convert the value “5EC2” from hexadecimal to decimal:

(That is how we do hexadecimal it is pretty easy exactly like binary except instead of 2’s we use 16 and in hexadecimal we use A-F to represent 10-15.)

The right most value utilizes the 2 value from the hexadecimal value and multiplies it by since it is the 0th bit spot for the hexadecimal value, in the first bit spot we use which is based off of C equaling 12 and multiplying it by the next bit spot, in the second bit spot we use E which is 14 and multiply it by where 2 is the location bit spot and lastly the most significant bit the 5 is where the most significant spot is and it is the 3rd location bit for the point hence we get . Then when we add this all up we get the value 24,258 which is the decimal value of 5EC2.

**File System Architecture and how it works**

In Ubuntu and almost every Linux distro there is something called the Filesystem Hierarchy Standard (FHS) which defines the directory structure and directories in the Linux distributions. The organization of these files is in a hierarchical tree matter where relationships are thought of in terms of parents, children and leaves.

The parent directory is considered to be the top directory which in our case of the FHS is the root directory. This means the root directory controls all the other directories which allows Ubuntu to function correctly you can think of this directory as the stump of a tree or the parent directory to all the other directories.

Well now that we understand the main stump or parent directory it is time to expand more into the different directories which allow Ubuntu to function correctly. In this next part of the lab below we will go through 16 directories which allow Ubuntu to function. After we go through this we will go through a couple sub directories within the directories and lastly after I will draw a picture for you all to visualize what I’m saying.

The **first** directory is the /bin/ directory within this directory we store user commands such as the ls command, cp, mv and etc…

The **second** directory is the /boot/ directory in this directory there are files which are used for booting the operating system. The more technical description is that the /boot/ directory includes the Linux kernel, a RAM disk image and a GRUB bootloader which boot Ubuntu. (A GRUB bootloader is a piece of software that loads the Linux Kernel and is typically the first software that is started up at boot time)

The **third** directory is the /dev/ directory within this directory we have device files! Device files are files which refer to your hardware devices on the system such as hard drives, USB devices and CD-ROMS (Note\* Device files **aren’t** regular files. Device files are an interface (or a map) for a device driver that allows the device to operate. Device drivers are a piece of software that does the actual controlling of the device.)

The **fourth** directoryis the /etc/ directory which contains configuration files which impact the systems behavior for all users. Still confused? Think of the /etc/ directory as the nerve center of the system it contains configuration files which impact the system itself.

The **fifth** directory is the /home/ directory this is where the different user directories are located. For instance multiple users might be here such as: /home/ComputerSecurity/, /home/Joseph/, /home/Linux/, etc…

The **sixth** directory is the /lib/ directory which contains libraries and kernel modules which allows Ubuntu to function (basically a whole bunch of C code)

The **seventh** directory is the /media/ directory which is used to mount (mount or set up) removable media devices such as CD-ROM, floppy drives, CD recorder and etc…

The **eighth** directory is the /mnt/ directory which is used when you want to mount media devices manually this command should typically only be used by system administrator.

The **ninth** directory is /opt/ directory which is used to store additional software on the current system which is independent of the Operating System. (This doesn’t include software packages such as libraries it simply just means add on applications)

The **tenth** directory /sbin/ contains administrative commands that are generally only utilized by using the superuser do command (commands such as init, route and ifconfig (we won’t be getting into these))

The **eleventh** directory is the /srv/ directory which contains data directories in regards to services such as HTTP, FTP, www, rsync and many more.

The **twelfth** directory is the /tmp/ directory this is where anything that is temporary is stored.

The **thirteenth** directory is the /usr/ directory this contains the majority of the user applications, libraries and binaries (binaries are executables they are similar to the .exe things that we download) which in turn makes this the largest folder in the Linux directory.

The **fourteenth** directory is the/var/ directory which is where all the variable data is stored such as logs, databases, websites, emails and other stuff. To access the log files we usually go /var/log/ directory.

The **fifteenth** directory is the /proc/ directory which is where the virtual file system provides a mechanism to the kernel to send and process information.

The **sixteenth** directory is the /root/ directory which is the home directory for the root user.

Still confused from my little descriptions understandable try to look at the tree diagram image below to understand the parent child relationship and see if this helps. If you are confused about the descriptions check out the “man 7 hier” command and it will explain the directories more in depth. All together those 16 directories + sub directories (which we didn’t discuss) make up the Ubuntu operating system kind of crazy if you think about it. Feel free to browse around within the files just don’t mess anything up.



**User Permissions**

In this section we are going to talk about how to manage local users and groups. This is a fairly straight forward process once you have done it a couple times. The first part we are going to do is adding and deleting users/groups without further ado let’s jump right into it.

**Adding and Deleting Users/Groups**

The most basic action that you need to learn on a Linux system is how to add/delete users and groups and as we go through this you might have questions just go back and re read the part you are confused about. If you still aren’t sure ask a question because asking questions is the best thing you can do because odds are if you ask it someone else is/was thinking the same thing.

In Ubuntu in order to add a user account we use the following command. Once the command is executed it will prompt us for a few credentials about the user. Let’s check it out below:

sudo adduser username

In the command above we are using the sudo keyword which is our administrative command we need this command in order to add users.

In the command above we also use the adduser keyword this well allows us to add a user and it takes in a parameter. The parameter is the next keyword which is “username” in reality it should be the name of your user in my case I am using the value username which means a user named username will be added. Check out the image below:



**Figure 3.1:** In the figure above we are illustrating how to make a user on a ubuntu machine.

In order to be completely sure you understand the output and what is happening we will go through it line by line the first line being where I’m prompted for my password.

The first line “[sudo] password for computersecurity:” as we know is just prompting me for my password for the given account because it is a new terminal session.

In the 2nd line Ubuntu is creating the user named “username”

In the 3rd line Ubuntu is creating a group named username and as we can see it has a number (1001) this number is the group ID number

In the 4th line Ubuntu is just telling you again that it is a adding a new user “username” with a group “username” and the group ID is 1001.

In the 5th line Ubuntu is creating a home directory for the new user which allows said new user to be able to work.

In the 6th line there are some files being copied from the /etc directory this is to ensure that you have the correct configuration has the System administrator.

In the 7th and 8th line we enter a new UNIX password and retype it in this password is important to remember! (It will allow you to log on to your new account)

In the 9th- 14th lines I showed what happens if a password failed and how we try again and reenter the UNIX passwords.

In the 15th line it tells us that the passwords are updated successfully.

In the 16th line we enter the user information

In the 17th -22nd line we enter the new values for what they prompt us for Full name, room number, etc…

In the 23rd line it prompts us to identify if the information we entered is correct.

After we do this and it is successful we can check out our new user just remember that we need our UNIX password. Follow the steps below to access your new user via GUI

1. Click on the following icon which is on the top right:



**Figure 3.2:** Click on the tree icon on the top right

1. After we do this then we will click on the lock button:



**Figure 3.3:** Click on the lock symbol to access the user login screen

1. After we do this then we will go to the log in screen and click Log in as another user.



**Figure 3.4:** Going to login as a different user.

1. Then after we do this we will see a bunch of other options for users which we are able to click on or not and if you click on a new user just remember to enter the new UNIX password you made for that user.



**Figure 3.5:** Listing out the different users

For our purposes of the class we won’t be going into the users (it is the same thing except they don’t have sudo privileges)

I know what you’re thinking most System administrators don’t have access to GUI’s or screens that they can button click on so how do they view the users well they use this command called “getent passwd” this is a very powerful command which displays users and groups and a lot of other properties we are going to break down a couple lines within this output because once you have gone through a couple lines you can figure everything else out.

The command is below:



**Figure 3.6:** Using a Linux System administrator command getent passwd to display the users and groups within the Ubuntu System.

Please enter in the above command into your current terminal (not the user you created your main account). Once this command is entered you will get a very large output (I know what you’re thinking this is a large and scary output but have no fear because I will break it down for you):



**Figure 3.7:** The log files for all the users and groups within the system.

Let’s go through the root user the screen shot of it is below it should be on the top of your command output:



Home Directory /root

Account user root

The second 0 is the primary group ID number

The first 0 means that the **User ID** is 0 (which makes sense because this is root)

The “x” means that their password is encrypted

The User name is root

/bin/bash is the login shell for the user (this means the user always needs to run this script with bash

**Figure 3.8:** Explaining what the values mean in user above.

That is a high level breakdown of how this works the output is pretty consistent for all these usernames. Some people might be looking at this output and thinking I never made 90 % of these users and you are correct you didn’t Ubuntu did. A lot of these processes in this output are called **daemons.** Daemonsare systems process and resources which are running in the background.

Let’s go through another example of what this output is actually saying. In the output below we are going to break down the user named “username”



The second number is the primary Group ID.

Account User Joe

The first number as I said is the **User ID** this is an identifier for you and the Systems Administrator

The “x” means that their password is encrypted

The username is “username”

See paragraph below about the “x”

This is the login shell that is executed every time you log in.

This is the home directory for this user.

The next 4 things are just the basic information that you entered when creating the new user.

**Figure 3.9:** In this figure we are going through a more detailed example of a user

One thing to note if the “x” is not there then there is **NO** authentication required. But when we have the lower case x then the password is encrypted and stored in the /etc/shadow directory (check out man 5 shadow for more information). If this still doesn’t make too much sense check out “man 5 passwd” it will give a more detailed explanation (don’t forget man stands for manual which is a helpful page about most commands within Ubuntu).

As we can see adding users aren’t that bad and neither are deleting users in the next line we enter we will delete a user. Deleting users aren’t very complicated either in the next couple of lines we will go through how to delete them:

In the command below “sudo deluser username” we are telling the Ubuntu shell that we want to delete a user called username. So in short we are using our administrative privileges and deleting the user called username. Below the image I will provide a breakdown of what I’m exactly doing here:



**Figure 3.10:** This figure illustrates how we can delete a user off the Ubuntu System.

In the above image I have the command “sudo deluser username”. In here we are using administrative privileges like I said and deleting a user called username.

In the first line we entered in the following administrative command “sudo deluser username”. In the next 3 lines we will break this down:

sudo: is the administrative command which we need to add/ delete users.

deluser: This is the command used to delete a user feel free to do “man deluser” for more information

“username”: this is the user parameter in our case the user is username.

After we enter this command we will enter in our password for “sudo” (these are the next 3 lines)

The next line we remove the user “username” and then it prompts us for a warning that there is no members in the group username (since we just deleted the only user in that group)

Then the last shows “Done” and it was success we officially deleted the user “username”.

After we finish deleting the user we can check out our getent passwd command and this will verify to us that the user is deleted check it out below:



**Figure 3.11:** Verifying that the user is deleted off the system.

In order to add users to a group we are going to need to learn another new command. This command being **usermod**. The usermod command is used to change and modify any attributes to an already created user via the command line. The usermod command has a lot of flags which can be used with it which make it a very simple yet powerful command. When I enter the next commands for the groups I will have assumed you have entered in “man usermod” and have familiarized yourself with the flags.

In the commands below we will create a group called software\_developers (this is where all the software developers within the company would be). After we create the group we will add our user Joe to the software\_developers group and then verify that he is in there.

In the screenshot below we will go through the lines which we wrote to create a group and add a user to the group:



**Figure 3.12:** Viewing the different groups made in the Ubuntu system.

In the first line above “sudo groupadd software\_developers” we are creating a group called software\_developers

sudo: this is the administrative command which we need to use to make groups. This is because we are dealing with the /etc directory. Normally when dealing with this directory we need to use the sudo command.

groupadd: is the command that we use to create a group and it takes in a parameter which is the name of the group we are making and in this case it is “software\_developers”

software\_developers: this is the name of the group which we made from the gorupadd command.

The next command is the “sudo usermod –a –G software\_developers joe” command. In this command we are adding the user joe to the software\_developers group. I understand that this command has a lot going on so allow me to break it down.

sudo: this is the administrative command which we need again to change user attributes.

usermod: is the command we are using to change the attributes of an existing user

-a flag: this flag is used to append a user to a secondary group.

-G flag: This flag and –a flag are typically used hand and hand the –G flag is used to add a supplementary group.

software\_developers: this is the group name which we are adding the user to.

joe: this is the user which we want to add to the group.

The third command we are going into the /etc directory which is where we have our groups, users, configuration files and much more this is a very important directory which we will get into later.

cd: current directory

/etc: the etc directory

The fourth command nano group allows us to view the file group which is where the different groups are located /etc/group directory. Once inside the file you should see something like this:



**Figure 3.13:** Viewing a snippet of the group file.

If we scroll to the bottom of this we will see our group software\_developers which says that it has a group id of 1004 and the only user in the following group is joe:



**Figure 3.14:** Created a new group called software\_developers verified.

In the next lab we are going to discuss further about the following: user management and password policies, memory management/memory addresses and start off learning a couple more advanced Linux commands.