

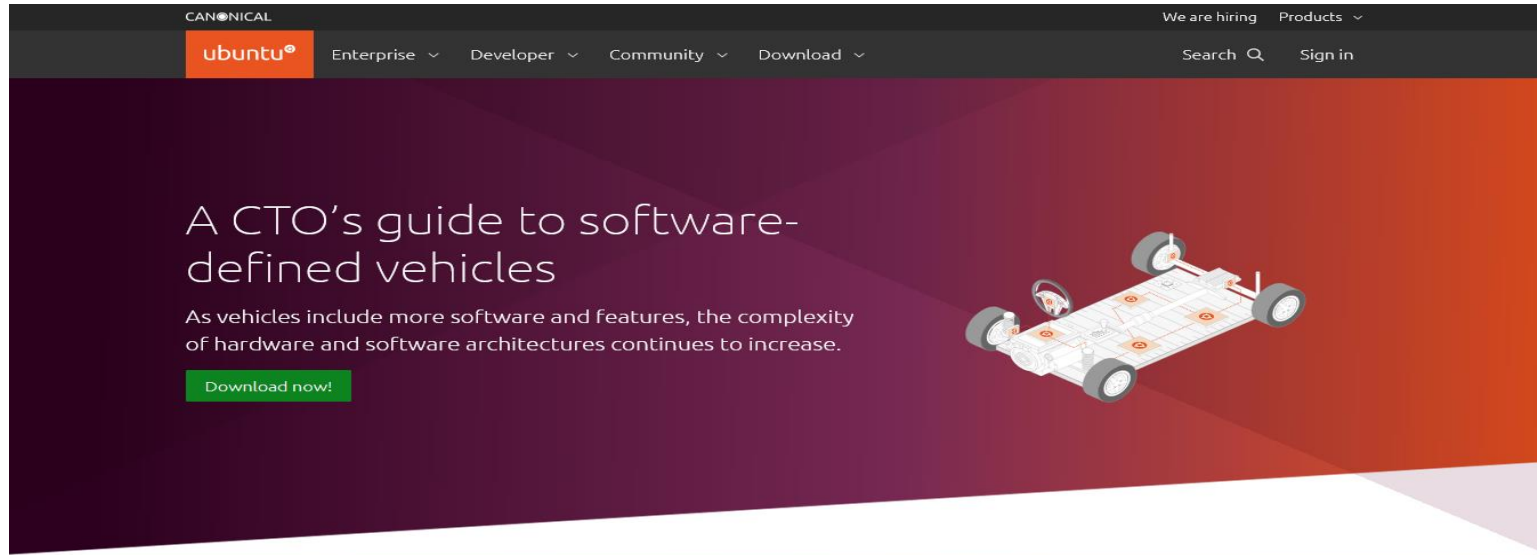
CYBER SECURITY AND NETWORK



Lochana koralage

Ubuntu – Overview

Ubuntu is a Linux-based operating system. It is designed for computers, smartphones, and network servers. The system is developed by a UK based company called Canonical Ltd. All the principles used to develop the Ubuntu software are based on the principles of Open-Source software development <https://ubuntu.com/> (https://en.wikipedia.org/wiki/List_of_Linux_distributions)



Features of Ubuntu

- ✓ The desktop version of Ubuntu supports all the normal software on Windows such as Firefox,
- ✓ Chrome, VLC, etc.
- ✓ It supports the office suite called LibreOffice.
- ✓ Ubuntu has an in-built email software called Thunderbird, which gives the user access to
- ✓ email such as Exchange, Gmail, Hotmail, etc.
- ✓ There are a host of free applications for users to view and edit photos. There are also applications to manage videos and it also allows the users to share videos.
- ✓ It is easy to find content on Ubuntu with the smart searching facility.
- ✓ The best feature is, it is a

Release Cycle of Ubuntu

Every year there are 2 releases of Ubuntu, one in April and one in October, from Canonical. The version number normally denotes the year in which the software was released. For example, version 14.04 specifies that it was released in the year 2014 and in the month of April. Similarly, the version 16.04 specifies that it was released in the year 2016 and in the month of April. The April build every year is the more stable build, while the October build does a lot of experimentation new features

Ubuntu – Flavors

Ubuntu Desktop

This is the operating system which can be used by regular users. This comes pre-built with software that help the users perform usual basic activities. Operations such as browsing, email and multimedia are also available in this edition. The latest version as of September 2016 is 16.04.01.

Ubuntu Server

The server version is used for hosting applications such as web servers and databases. Each server version is supported by Ubuntu for 5 years. These operating systems have support for cloud platforms such as AWS and Azure. The latest version as of September 2016 is 16.04.1

Requirement

- ✓ Memory 2GB RAM (recommended)
- ✓ 25GB of free hard disk spaces
- ✓ 2 GHz dual core processor or better

Kubuntu

The normal Ubuntu interface is based on a software called Unity. However, Kubuntu is based on a software called KDE Plasma desktop. This gives a different look and feel to the Ubuntu software.

Kubuntu has the same features and software availability as Ubuntu. The official site for Kubuntu is <http://www.kubuntu.org>

File Permission / Access Modes

- ✓ **Owner permissions** – The owner's permissions determine what actions the owner of the file can perform on the file.
- ✓ **Group permissions** – The group's permissions determine what actions a user, who is a member of the group that a file belongs to, can perform on the file.
- ✓ **Other (world) permissions** – The permissions for others indicate what action all other users can perform on the file.

\$ls -l /home/desktop

rw-r--r-- 1 desktop users 1024 Nov 2 00:10 lochanafile

drwxr-xr-x 1 desktop users 1024 Nov 2 00:10 lochfile

Linux Command

- ✓ Su
- ✓ ifconfig
- ✓ Whoami
- ✓ adduser :
- ✓ Chmod : You accomplish this by changing the rights or permissions for a file or directory. Here is the syntax for the chmod command
- ✓ ls -l
- ✓ Apt -get update
- ✓ Apt-get install
- ✓ Cd

Here, the first column represents different access modes, i.e., the permission associated with a file or a directory.

- ✓ The permissions are broken into groups of threes, and each position in the group denotes a specific permission, in this order: read (r), write (w), execute (x) –
- ✓ The first three characters (2-4) represent the permissions for the file's owner. For example, -rwxr-xr-- represents that the owner has read (r), write (w) and execute (x) permission.
- ✓ The second group of three characters (5-7) consists of the permissions for the group to which the file belongs. For example, -rwxr-xr-- represents that the group has read (r) and execute (x) permission, but no write permission.
- ✓ The last group of three characters (8-10) represents the permissions for everyone else. For example, -rwxr-xr-- represents that there is read (r) only permission.

File Access Modes

The permissions of a file are the first line of defense in the security of a Unix system. The basic building blocks of Unix permissions are the **read**, **write**, and **execute** permissions, which have been described below –

Read

Grants the capability to read, i.e., view the contents of the file.

Write

Grants the capability to modify or remove the content of the file.

Execute

User with execute permissions can run a file as a program.

BASH scripting

Bash Scripting is a powerful part of system administration and development used at an extreme level. It is used by the System Administrators, Network Engineers, Hacker Cyber security researcher, Developers, Scientists, and everyone who use Linux/Unix operating system. They use Bash for system administration, data crunching, web application deployment, automated backups, creating custom scripts for various pages

Hello World : cat command

```
lochana@att:~/Desktop/new$ cat hello.sh
#!/bin/bash

echo "helo lochana"
```

Hello world Rename (body)

```
GNU nano 6.2 hello.sh *
#!/bin/bash

echo "hello|lochana"
```

Permission issues

How to Fix ?

```
root@att:/home/lochana/Des x + -
```

```
GNU nano 6.2 hello.sh *  
#!/bin/bash  
  
echo "hello loch"  
  
[ Error writing hello.sh: Permission denied ]  
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute  
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify
```

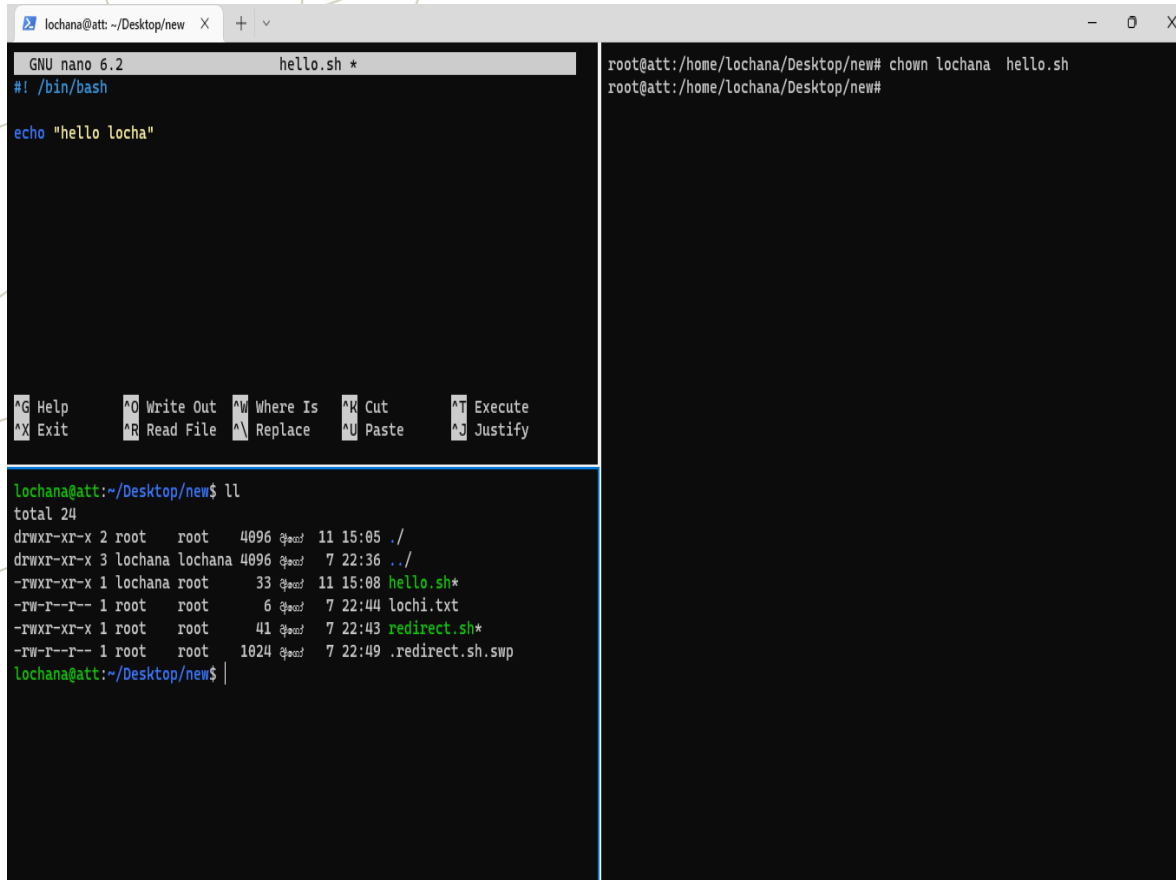
```
Windows PowerShell  
Copyright (C) Microsoft Corporation. All rights reserved.  
  
Try the new cross-platform PowerShell https://aka.ms/powershell  
  
PS C:\Users\Loch-PC>
```

```
GNU nano 6.2 hello.sh  
#!/bin/bash  
  
echo "hello loch"  
  
[ Read 5 lines ]  
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute  
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify
```



Solution

(chown : Linux chown command is used to change a file's ownership)



The screenshot shows a terminal window with two panes. The left pane is a nano editor editing a file named 'hello.sh'. The content of the file is 'echo "hello locha"'. The right pane shows a terminal session where the user 'lochana' is at the prompt 'lochana@att: ~/Desktop/new'. They run the command 'll' to list files, showing a directory listing with permissions, owners, and file names. Then, they run 'chown lochana hello.sh' as root, and the prompt returns to 'lochana@att: ~/Desktop/new\$'.

```
GNU nano 6.2      hello.sh *
#! /bin/bash

echo "hello locha"

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^_ Justify

lochana@att:~/Desktop/new$ ll
total 24
drwxr-xr-x 2 root  root  4096 @ead  11 15:05 ./
drwxr-xr-x 3 lochana lochana 4096 @ead  7 22:36 ../
-rwxr-xr-x 1 lochana root   33 @ead  11 15:08 hello.sh*
-rw-r--r-- 1 root   root    6 @ead  7 22:44 lochi.txt
-rwxr-xr-x 1 root   root   41 @ead  7 22:43 redirect.sh*
-rw-r--r-- 1 root   root 1024 @ead  7 22:49 .redirect.sh.swp
lochana@att:~/Desktop/new$
```

- ✓ The Linux system may have multiple users. Every user has a unique name and user ID. If only a user is available in the system, the user will be the owner of each file.
- ✓ The Linux system may have multiple users. Every user has a unique name and user ID. If only a user is available in the system, the user will be the owner of each file.
- ✓ Users can be listed in different groups. The group allows us to set permission on the group level instead of setting permission on an individual level.

Redirect file content

```
lochana@att: ~/Desktop/new$ cat lochi.txt
hello
lochana@att:~/Desktop/new$ |

root@att:/home/lochana/Desktop/new# cat redirect.sh
#!/bin/bash

echo "hello" >>lochi.txt
root@att:/home/lochana/Desktop/new#

lochana@att:~/Desktop/new$ ll
total 24
drwxr-xr-x 2 root root 4096 @aos 11 15:05 ./
drwxr-xr-x 3 lochana lochana 4096 @aos 7 22:36 ../
-rwxr-xr-x 1 lochana root 35 @aos 11 15:14 hello.sh*
-rw-r--r-- 1 root root 6 @aos 7 22:44 lochi.txt
-rwxr-xr-x 1 root root 41 @aos 7 22:43 redirect.s
h*
-rw-r--r-- 1 root root 1024 @aos 7 22:49 .redirect.
sh.swp
lochana@att:~/Desktop/new$
```

Live Redirect content

```
GNU nano 6.2 iit.txt
Lochana in IIT
lochana thank you for coming

[ File 'iit.txt' is unwritable ]
^G Help ^O Write Out ^W Where Is ^K Cut
^X Exit ^R Read File ^A Replace ^U Paste

lochana@att:~/Desktop/new$ ll
total 24
drwxr-xr-x 2 root root 4096 @aos 11 15:05 ./
drwxr-xr-x 3 lochana lochana 4096 @aos 7 22:36 ../
-rwxr-xr-x 1 lochana root 35 @aos 11 15:14 hello.sh*
-rw-r--r-- 1 root root 6 @aos 7 22:44 lochi.txt
-rwxr-xr-x 1 root root 41 @aos 7 22:43 redirect.s
h*
-rw-r--r-- 1 root root 1024 @aos 7 22:49 .redirect.
sh.swp
lochana@att:~/Desktop/new$ ls
hello.sh lochi.txt redirect.sh
lochana@att:~/Desktop/new$

root@att:/home/lochana/Desktop/new# cat re.sh
Lochana in IIT
Lochana thank you for coming
root@att:/home/lochana/Desktop/new# nano re.sh
root@att:/home/lochana/Desktop/new# ./re
root@att:/home/lochana/Desktop/new# cat re.sh
#!/bin/bash

echo "Lochana in IIT" >>iit.txt
root@att:/home/lochana/Desktop/new# nano re.sh
root@att:/home/lochana/Desktop/new# ./re
root@att:/home/lochana/Desktop/new# cat re.sh
Lochana thank you for coming
```

```
GNU nano 6.2 iit.txt
Lochana in IIT
lochana thank you for coming

[ File 'iit.txt' is unwritable ]
^G Help ^O Write Out ^W Where Is ^K Cut
^X Exit ^R Read File ^A Replace ^U Paste

lochana@att:~/Desktop/new$ ll
total 24
drwxr-xr-x 2 root root 4096 @aos 11 15:05 ./
drwxr-xr-x 3 lochana lochana 4096 @aos 7 22:36 ../
-rwxr-xr-x 1 lochana root 35 @aos 11 15:14 hello.sh*
-rw-r--r-- 1 root root 6 @aos 7 22:44 lochi.txt
-rwxr-xr-x 1 root root 41 @aos 7 22:43 redirect.s
h*
-rw-r--r-- 1 root root 1024 @aos 7 22:49 .redirect.
sh.swp
lochana@att:~/Desktop/new$ ls
hello.sh lochi.txt redirect.sh
lochana@att:~/Desktop/new$

root@att:/home/lochana/Desktop/new# ./re
root@att:/home/lochana/Desktop/new# cat re.sh
#!/bin/bash

echo "Lochana in IIT" >>iit.txt
root@att:/home/lochana/Desktop/new# nano re.sh
root@att:/home/lochana/Desktop/new# ./re
root@att:/home/lochana/Desktop/new# cat re.sh
Lochana thank you for coming
root@att:/home/lochana/Desktop/new# cat re.sh
#!/bin/bash

cat >> iit.txt
root@att:/home/lochana/Desktop/new# |
```

Comments

```
lochana@att: ~/Desktop/new  X + v
lochana@att:~/Desktop/new$ nano iit.text
lochana@att:~/Desktop/new$ ./comment.sh
Hello lochana
lochana@att:~/Desktop/new$ |

root@att:/home/lochana/Desktop/new# cat comment.sh
#!/bin/bash

echo "Hello lochana"

#echo " hello"
root@att:/home/lochana/Desktop/new#
```

Variables

```
root@att:/home/lochana/Des  X + v
lochana@att:~/Desktop/new$ cat variable.sh
#!/bin/bash

Name="Lochana"
echo $Name
lochana@att:~/Desktop/new$

root@att:/home/lochana/Desktop/new# ./variable.sh
Lochana
root@att:/home/lochana/Desktop/new# |
```

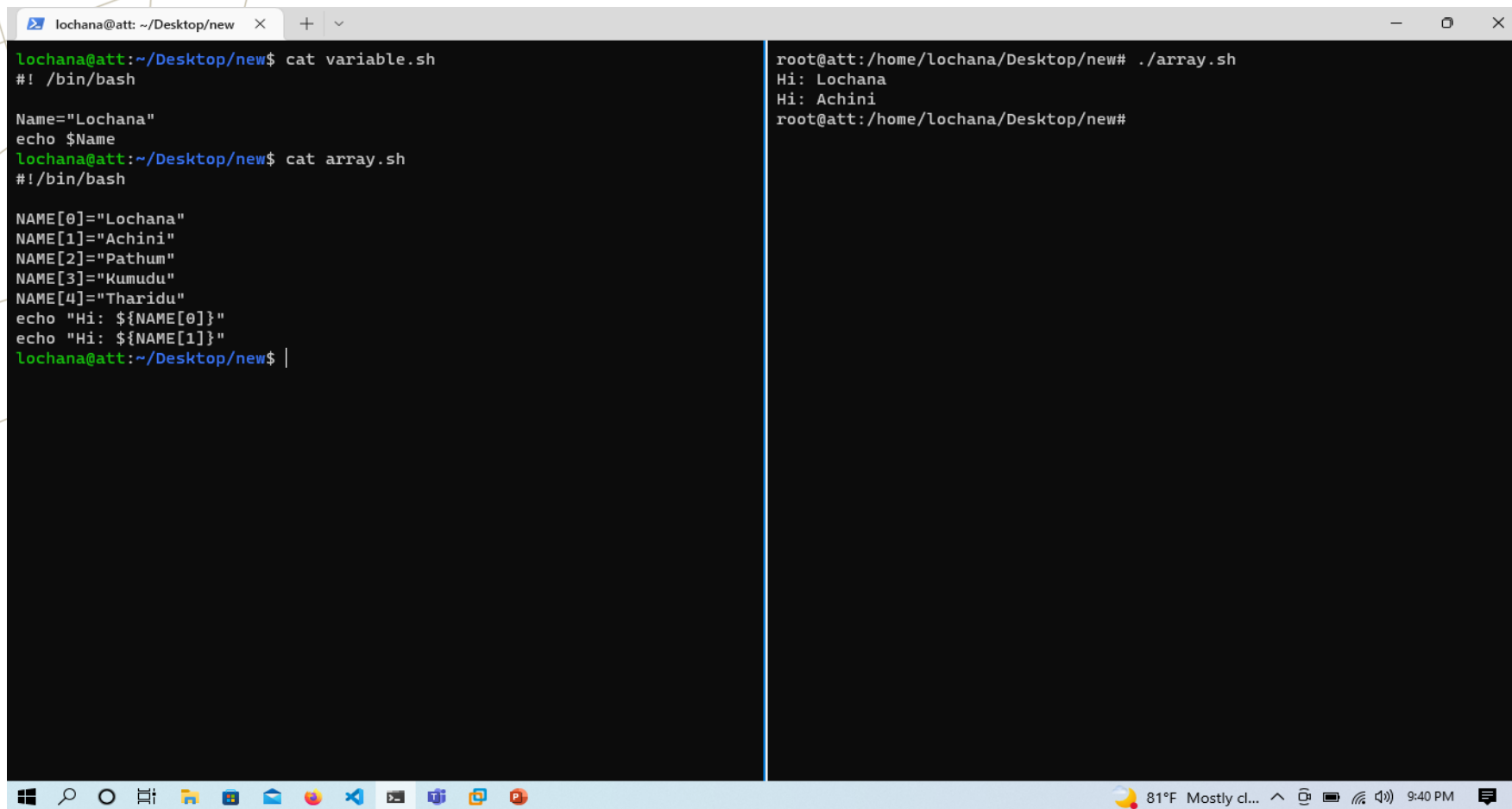
Arrays

```
lochana@att: ~/Desktop/new  x + v
lochana@att:~/Desktop/new$ cat variable.sh
#!/bin/bash

Name="Lochana"
echo $Name
lochana@att:~/Desktop/new$ cat array.sh
#!/bin/bash

NAME[0]="Lochana"
NAME[1]="Achini"
NAME[2]="Pathum"
NAME[3]="Kumudu"
NAME[4]="Tharidu"
echo "Hi: ${NAME[0]}"
echo "Hi: ${NAME[1]}"
lochana@att:~/Desktop/new$ |

root@att:/home/lochana/Desktop/new# ./array.sh
Hi: Lochana
Hi: Achini
root@att:/home/lochana/Desktop/new#
```



Basic Operators

- Arithmetic Operators
- Relational Operators
- Boolean Operators
- String Operators
- File Test Operators

Operator	Description	Example
+ (Addition)	Adds values on either side of the operator	`expr \$a + \$b` will give 30
- (Subtraction)	Subtracts right hand operand from left hand operand	`expr \$a - \$b` will give -10
* (Multiplication)	Multiplies values on either side of the operator	`expr \$a * \$b` will give 200
/ (Division)	Divides left hand operand by right hand operand	`expr \$b / \$a` will give 2
% (Modulus)	Divides left hand operand by right hand operand and returns remainder	`expr \$b % \$a` will give 0
= (Assignment)	Assigns right operand in left operand	a = \$b would assign value of b into a
== (Equality)	Compares two numbers, if both are same then returns true.	[\$a == \$b] would return false.
!= (Not Equality)	Compares two numbers, if both are different then returns true.	[\$a != \$b] would return true.

Relational Operators

Bourne Shell supports the following relational operators that are specific to numeric values. These operators do not work for string values unless their value is numeric.

Operator	Description	Example
-eq	Checks if the value of two operands are equal or not; if yes, then the condition becomes true.	[\$a -eq \$b] is not true.
-ne	Checks if the value of two operands are equal or not; if values are not equal, then the condition becomes true.	[\$a -ne \$b] is true.
-gt	Checks if the value of left operand is greater than the value of right operand; if yes, then the condition becomes true.	[\$a -gt \$b] is not true.
-lt	Checks if the value of left operand is less than the value of right operand; if yes, then the condition becomes true.	[\$a -lt \$b] is true.
-ge	Checks if the value of left operand is greater than or equal to the value of right operand; if yes, then the condition becomes true.	[\$a -ge \$b] is not true.
-le	Checks if the value of left operand is less than or equal to the value of right operand; if yes, then the condition becomes true.	[\$a -le \$b] is true.

Relational Operators Example

```
lochana@att: ~/Desktop/new
lochana@att:~/Desktop$ sudo su
[sudo] password for lochana:
root@att:/home/lochana/Desktop# cd new/
root@att:/home/lochana/Desktop/new# ls
array.sh  comment.sh  hello.sh  iit.text  lochi.txt  re2  redirect.sh  r
e.sh  variable.sh
root@att:/home/lochana/Desktop/new# nano operationre.sh
root@att:/home/lochana/Desktop/new# chmod +x operationre.sh
root@att:/home/lochana/Desktop/new# ./operationre.sh
10 -eq 20: a is not equal to b
10 -ne 20: a is not equal to b
10 -gt 20: a is not greater than b
10 -lt 20: a is less than b
10 -ge 20: a is not greater or equal to b
10 -le 20: a is less or equal to b
root@att:/home/lochana/Desktop/new# |

a=10
b=20

if [ $a -eq $b ]
then
    echo "$a -eq $b : a is equal to b"
else
    echo "$a -eq $b: a is not equal to b"
fi

if [ $a -ne $b ]
then
    echo "$a -ne $b: a is not equal to b"
else
    echo "$a -ne $b : a is equal to b"
fi

if [ $a -gt $b ]
then
    echo "$a -gt $b: a is greater than b"
else
    echo "$a -gt $b: a is not greater than b"
fi

if [ $a -lt $b ]
then
    echo "$a -lt $b: a is less than b"
else
    echo "$a -lt $b: a is not less than b"
fi

if [ $a -ge $b ]
then
    echo "$a -ge $b: a is greater or equal to b"
else
    echo "$a -ge $b: a is not greater or equal to b"
fi

if [ $a -le $b ]
then
    echo "$a -le $b: a is less or equal to b"
else
    echo "$a -le $b: a is not less or equal to b"
fi
lochana@att:~/Desktop/new$ |
```

Shell Decision Making

conditional statements which are used to perform different actions based on different conditions.

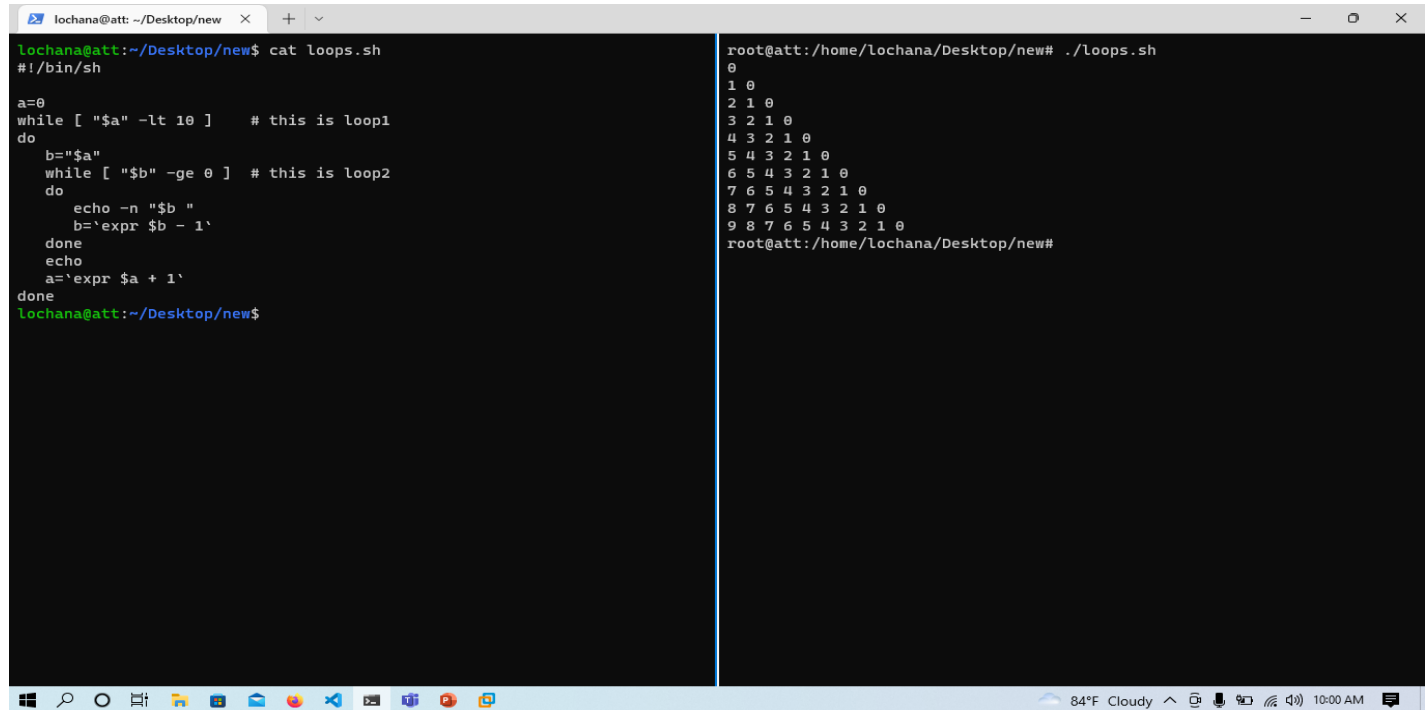
```
lochana@att: ~/Desktop/new
lochana@att:~/Desktop$ cd /home/lochana/Desktop/
lochana@att:~/Desktop$ sudo su
[sudo] password for lochana:
root@att:/home/lochana/Desktop# cd new/
root@att:/home/lochana/Desktop/new# ls
array.sh comment.sh hello.sh iit.text lochi.txt re2 redirect.sh r
e.sh variable.sh
root@att:/home/lochana/Desktop/new# nano operationre.sh
root@att:/home/lochana/Desktop/new# chmod +x operationre.sh
root@att:/home/lochana/Desktop/new# ./operationre.sh
10 -eq 20: a is not equal to b
10 -ne 20: a is not equal to b
10 -gt 20: a is not greater than b
10 -lt 20: a is less than b
10 -ge 20: a is not greater or equal to b
10 -le 20: a is less or equal to b
root@att:/home/lochana/Desktop/new# nano if.sh
root@att:/home/lochana/Desktop/new# nano if.sh
root@att:/home/lochana/Desktop/new# nano if.sh
root@att:/home/lochana/Desktop/new# la
array.sh if.sh operationre.sh re.sh
comment.sh iit.text re2 variable.sh
hello.sh lochi.txt redirect.sh
root@att:/home/lochana/Desktop/new# chmod +x if.sh
root@att:/home/lochana/Desktop/new# ./if.sh
His name is Lochana. It is true.
root@att:/home/lochana/Desktop/new#
```

```
lochana@att:~/Desktop/new$ cat if.sh
Name="Lochana"
if [ "$Name" = "Lochana" ]; then
    echo "His name is Lochana. It is true."
fi
lochana@att:~/Desktop/new$
```


Loop

A loop is a powerful programming tool that enables you to execute a set of commands repeatedly

- ✓ The while loop
- ✓ The for loop
- ✓ The until loop
- ✓ The select loop



The screenshot shows a terminal window with two panes. The left pane displays the contents of a file named `loops.sh`, which is a shell script. The script starts with `#!/bin/sh` and sets `a=0`. It then enters a `while` loop that runs as long as `a` is less than 10. Inside this loop, there is another `while` loop that runs as long as `b` is greater than or equal to 0. The inner loop prints `b` and then decrements it by 1. After the inner loop finishes, `a` is incremented by 1. The right pane shows the output of running the script, displaying a series of numbers from 0 to 9, each followed by a space-separated list of numbers from 0 to 9. The terminal window has a title bar that reads "lochana@att: ~/Desktop/new" and a status bar at the bottom showing "84°F Cloudy" and "10:00 AM".

```
lochana@att:~/Desktop/new$ cat loops.sh
#!/bin/sh
a=0
while [ "$a" -lt 10 ] # this is loop1
do
    b="$a"
    while [ "$b" -ge 0 ] # this is loop2
    do
        echo -n "$b "
        b=`expr $b - 1`
    done
    echo
    a=`expr $a + 1`
done
lochana@att:~/Desktop/new$
```

```
root@att:/home/lochana/Desktop/new# ./loops.sh
0
1 0
2 1 0
3 2 1 0
4 3 2 1 0
5 4 3 2 1 0
6 5 4 3 2 1 0
7 6 5 4 3 2 1 0
8 7 6 5 4 3 2 1 0
9 8 7 6 5 4 3 2 1 0
root@att:/home/lochana/Desktop/new#
```

What is Apache Web Server?

Apache HTTP Server is a free and open-source web server that delivers web content through the internet. It is commonly referred to as Apache and after development, it quickly became the most popular HTTP client on the web

Apache Web Application Architecture

Apache is just one component that is needed in a web application stack to deliver web content. One of the most common web application stacks involves LAMP, or Linux, Apache, MySQL, and PHP.

Linux is the operating system that handles the operations of the application. Apache is the web server that processes requests and serves web assets and content via HTTP. MySQL is the database that stores all your information in an easily queried format. PHP is the programming language that works with apache to help create dynamic web content

Web Server Landscape

Back in the late 90s and early 2000s, Apache's dominance was very strong, serving over 50% of the internet's active websites. Microsoft's IIS (Internet Information Services) was also an option but not nearly as popular

Apache still serves a large portion of the active websites but their share of the field has shrunk from 50% to just under 40% as of 2018 and [NGINX](#), a relatively new player to the web server playing field, is in second place with roughly 35% and Microsoft IIS hovering around 8-10%.

Why Apache Web Servers?

Apache is considered open-source software, which means the original source code is freely available for viewing and collaboration

Features of Apache Web Server

Handling of static files

Loadable dynamic modules

Auto-indexing

.htaccess

Compatible with IPv6

Supports HTTP/2

FTP connections

Gzip compression and decompression

Bandwidth throttling

Perl, PHP, Lua scripts

Load balancing

Session tracking

URL rewriting

Geolocation based on IP address

Apache functions as a way to communicate over networks from client to server using the TCP/IP protocol. Apache can be used for a wide variety of protocols, but the most common is HTTP/S.

HTTP/S or Hyper Text Transfer Protocol (S stands for Secure) is one of the main protocols on the web, and the one protocol Apache

Who Uses Apache Web Server?

Apache HTTP web servers are used by over 67% of all web servers in the world. Apache web servers are easy to customize environments, they're fast, reliable, and highly secure. This makes Apache web servers a common choice by best-in-class companies.

Alternatives for Apache HTTP Server

- Nginx
- Apache Tomcat
- Node.js
- Lighttpd
- Cherokee
- Microsoft IIS
- Appweb
- Hiawatha