LINUX Notes By bongoDev

bongo Dev

Linux Advanced

LINUX Notes By bongoDev

Users and groups Management

Add user: $ sudo useradd -m user_name
Check users: $ cat /etc/passwd

Set user password: $ sudo passwd user_name
Switch User: $ su user_name

Delete user: $ sudo userdel -r user_name

Note: When we create an user, a group is already created by the same name

Groups Management

Create a group: $ sudo groupadd group_name
Check groups: $ cat /etc/group

Add single user in a group: $ sudo usermod -aG group_name user
Check users in group: $ cat /etc/group

Add Multiple users in a group at one go:
$ sudo gpasswd -M user1,user2,user3 group_name

Give access to a group (single group) to a file:
$ sudo chgrp group_name file_name

Check what happened:
$ ls -l index.txt > -rw-rw-r-- 1 ubuntu devops 1250 Nov 30 15:35 index.txt

Give multiple group access to a file/directory

Through Access Control Lists (ACLs): is a feature in Linux and other operating systems that provides more fine-grained file and directory permission management than the traditional owner-group-others permission model.

Key Features of ACL

1. **Fine-Grained Access Control:** ACLs let you grant specific permissions (e.g., read, write, execute) to multiple users and groups, which is not possible with traditional permissions.
2. **Default ACLs for Directories:** You can define default ACLs for directories so that new files inherit the specified permissions automatically.
3. **Backward Compatibility:** ACLs coexist with traditional permission mechanisms. If a file has no ACL, traditional permissions are used.

Example:

- **u:alice:rwx** User alice has read, write, and execute permissions.
- **g:developers:rw-** Group developers has read and write permissions.

ACL Commands

For group: $ setfacl -m g:groupname:permissions filename
$setfacl -m g:mygroup:rw config.yaml
For user: $ setfacl -m u:username:permissions filename

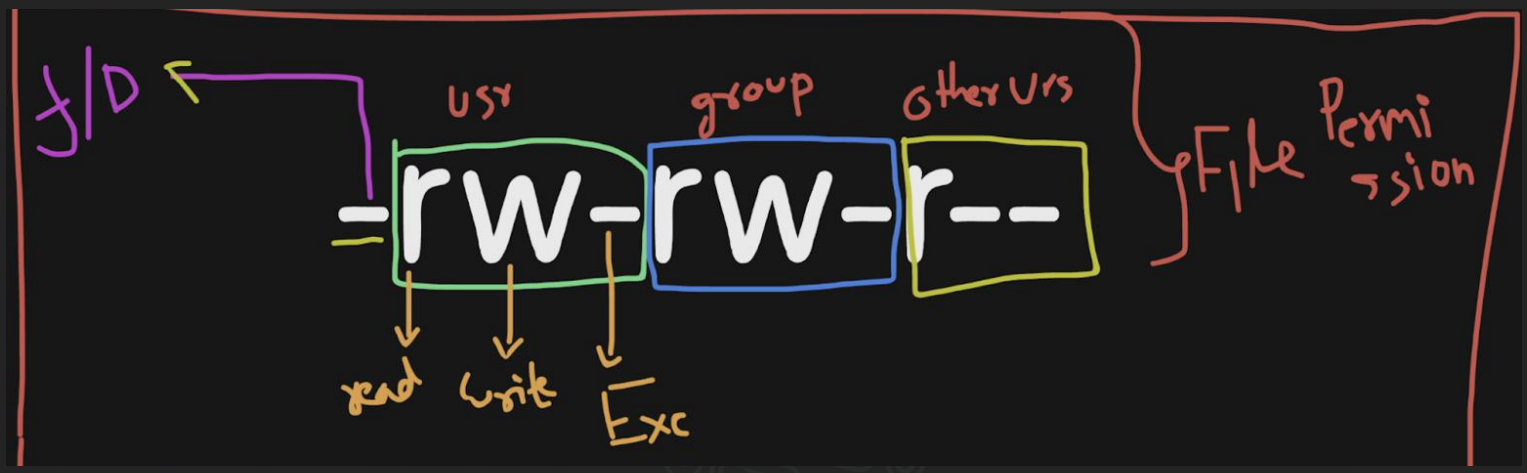
If ACL doesn't support, run this, $ sudo apt install acl
Check file ACL: $ getfacl config.yaml
Remove ACLs from a File or Directory: $ setfacl -b filename
Remove ACL From Computer: $ sudo apt remove acl

LINUX Notes By bongoDev

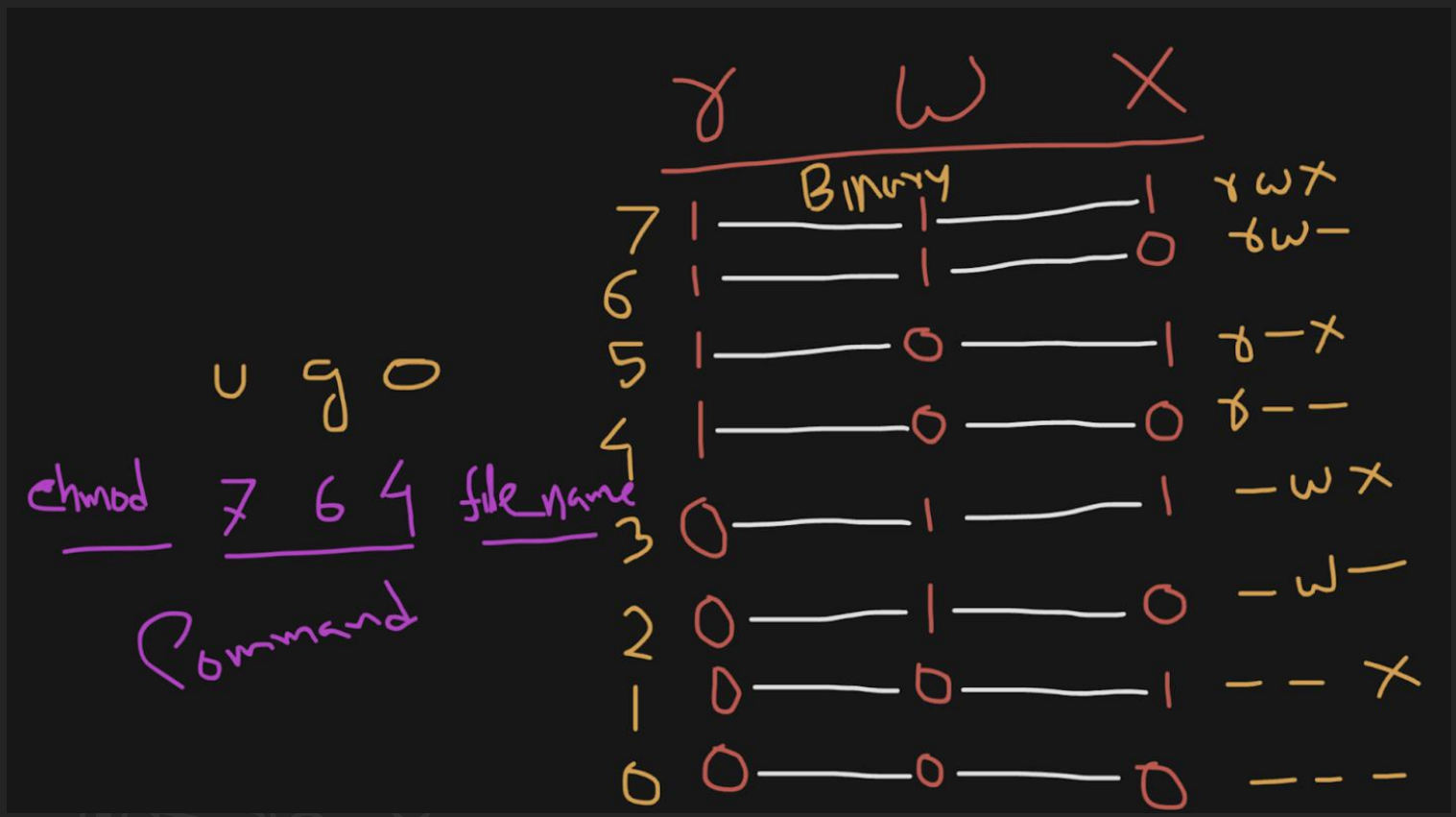
bongo Dev

File Permission

Everything in linux is either a file or a directory



chmod Command Reference



File Permission

File permission command: $ chmod permission file_name

```
ubuntu@ip-172-31-3-232:~$ ls -l
total 8
-rw-rw-r-- 1 ubuntu devops 1250 Nov 30 15:35 index.txt
drwxrwxr-x 2 ubuntu ubuntu 4096 Nov 30 16:04 linux2
-rw-rw-r-- 1 ubuntu ubuntu    0 Nov 30 15:26 log
ubuntu@ip-172-31-3-232:~$ chmod 770 log
ubuntu@ip-172-31-3-232:~$ ls -l log
-rwxrwx--- 1 ubuntu ubuntu 0 Nov 30 15:26 log
ubuntu@ip-172-31-3-232:~$
```

Now, I want only my users (ubuntu) will have all permissions but no group and other users will have permission for the file - log:

```
ubuntu@ip-172-31-3-232:~$ chmod 700 log
ubuntu@ip-172-31-3-232:~$ ls -l log
-rwx------ 1 ubuntu ubuntu 0 Nov 30 15:26 log
ubuntu@ip-172-31-3-232:~$
```

SSH: How To SSH On AWS Machine

SSH: Secure Shell, is a cryptographic network protocol used to securely access and manage devices over an unsecured network. It provides a secure channel for communication between two devices, such as a client and a server, by encrypting the data exchanged between them. SSH is commonly used for remote login, command execution, and secure file transfer.

How SSH Works:

1. Handshake: When you initiate an SSH connection, the client and server exchange keys and establish a secure channel.
2. Authentication: The server authenticates the client (e.g., using a password or SSH key).
3. Secure Communication: Once authenticated, all communication between the client and server is encrypted.

SSH typically runs on port 22 by default and is widely used in system administration, development, and DevOps practices.

Go to terminal and type: ssh

To connect from local (your PC) to remote (cloud computer), or one remote to another remote server, you need ssh client is installed on your computer

When you launch an EC2 machine (AWS Instance/computer), it provided you a .pem file (private_key) and got downloaded automatically to your Downloads folder/directory

LINUX Notes By bongoDev

Check .pem file content: cat <file_name>

What do I need to SSH: two files

1. **public_key:** should be available in the computer where you want to connect/access/send files. Example: you have two computers/servers - A & B. You would like to get connected from A to B, you should have Public_key in B.
2. **private_key:** should be available in your PC or from the computer/server you would like to connect/access other computer/server. Example: you have two computers/servers - A & B. You would like to get connected from A to B, you should have Private_key in A.

SSH: How To Generate SSH Keys: PUBLIC and Private Pair

Command: $ ssh-keygen

Press enter, no need to set password, but set passwoedd if you want

```
ubuntu@ip-172-31-3-232:~$ ls -a
.   .bash_history  .bashrc  .env      .profile  .sudo_as_admin_successful  app.log     friend.txt
..  .bash_logout   .cache   .lesshst  .ssh      .viminfo                   config.yaml  hall
ubuntu@ip-172-31-3-232:~$ ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_ed25519
Your public key has been saved in /home/ubuntu/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:t1As0fhP3+HG+3aMk5Bjpqs++WYXR1ngrO/1iFZhfCc ubuntu@ip-172-31-3-232
The key's randomart image is:
+--[ED25519 256]--+
|        .o    .. |
|        .o.  o  .|
|        ..o  .oo |
|         o. ..E.o|
|        S .o.=o=o|
|         o .O.++.|
|          ..+ *o+o|
|        o + oo++=|
|       .o*o+. oo=|
+----[SHA256]-----+
ubuntu@ip-172-31-3-232:~$ cd .ssh
ubuntu@ip-172-31-3-232:~/.ssh$ ls
authorized_keys  id_ed25519  id_ed25519.pub
ubuntu@ip-172-31-3-232:~/.ssh$
```

Is your private key

Is your public key

LINUX Notes By bongoDev

How to copy local file (config.yaml) to a server

```
scp -i linux2.pem config.yaml
ubuntu@ec2-18-191-205-7.us-east-2.compute.amazonaws.com:/home/ubuntu
```

How to download file from a Server to Your Computer

$ scp -i linux2.pem ubuntu@ec2-18-191-205-7.us-east-2
.compute.amazonaws.com:/home/ubuntu/config.yaml .

System CTL (systemd control):

Many processes and services running in the server and check their status, start, stop, restart, enable, disable via systemctl

systemctl is a command-line utility used to interact with the systemd service manager in Linux-based operating systems. It provides a centralized way to manage system and service processes, control system startup, and monitor services.

Key Features of systemctl:

1. **Service Management:** Start, stop, restart, enable, disable, and check the status of services.
2. **System Boot Management:** Configure services to start at boot and analyze the boot process.
3. **Unit Management:** Manage systemd "units," which include services, sockets, timers, and more.
4. **Monitoring:** Check system performance, resource usage, and log output.

LINUX Notes By bongoDev

bongo Dev

System CTL Commands

List all services: $systemctl list-units

Status of a service: $systemctl status ssh
$systemctl status nginx

Start a service: $sudo systemctl start nginx

Stop a service: $sudo systemctl stop nginx

Check log of a system: $ sudo journalctl -fu nginx.service

Restart a service: $sudo systemctl restart <service_name>

Reload a service's configuration without stopping:
$sudo systemctl reload <service_name>

Enable a service to start at boot:
$sudo systemctl enable <service_name>

Disable a service from starting at boot:
$sudo systemctl disable <service_name>

GREP AWK, FIND, and SED

The common purpose of GREP, AWK, FIND, and SED is text and file processing. They are essential utilities in UNIX/Linux systems, are designed to handle data retrieval, processing, and manipulation, making them invaluable for system administrators, developers, and data analysts working

Common Use Cases:

1. Searching for Data:
   - grep: Search for patterns or strings within files.
   - find: Locate files and directories based on attributes like name, size, or date.
2. Filtering and Processing Text:
   - awk: Extract, process, and analyze structured data (e.g., columns in a file).
   - grep: Extract lines matching specific patterns.
   - sed: Filter and transform text.
3. Modifying Data:
   - sed: Perform substitutions, deletions, and transformations on text streams.
   - awk: Perform calculations, reformat, or transform data on the fly.
4. Automating File Operations:
   - find: Locate and take actions on files (e.g., delete, move, or process them in bulk).
   - sed and awk: Automate editing tasks directly in scripts.
5. Combining with Other Tools:
   - These tools are often combined with others like sort, uniq, or cut in pipelines to perform complex tasks in shell scripts.

LINUX Notes By bongoDev

Comparison

Tool	Primary Function	Best Use Case
`grep`	Search for text patterns	Find specific lines matching a pattern
`awk`	Text processing and field manipulation	Extract and format columns of data
`sed`	Stream editing	Modify text in files or streams
`find`	File and directory search	Locate files/directories in a filesystem

Find log data: https://github.com/logpai/loghub

GREP (Global Regular Expression Print)

used to search for specific patterns or strings in files or input. It's powerful with regular expressions

Note: GREP is case sensitive while parsing text/string data

Commands:

Find lines containing "error" in a log file:
$ grep "error" file_name

Find lines containing "error" (case insensitive):
$ grep -i "error" file_name

Find lines matching "error" with line numbers:
$ grep -n "error" file_name

AWK (Aho, Weinberger, and Kernighan)

awk is a text processing tool used for pattern scanning and processing. It's particularly useful for analyzing structured data.

Commands:

Print the second column of a file:
```
$ awk '{print $2}' app.log
```

Find lines where the value in the second column is greater than 50:
```
$ awk '$2 > 50' app.log
```

Calculate the sum of the second column:
```
$ awk '{sum += $2} END {print sum}' app.log
```

Find lines for a particular date duration:
```
$ awk '$1>="2015-07-29" && $1<="2015-07-30" {print $1, $4, $7}' app.log
```

FIND

is used to search for files and directories in a directory hierarchy based on various criteria.

Commands:

Find all .txt files in the current directory and its subdirectories:

```
$ find . -name "*.txt"
```

Find files modified in the last 7 days:

```
$ find .  -mtime  -7
```

Find and delete empty files:

```
$ find .  -type f  -empty  -delete
```

SED (Stream Editor)

SED is a stream editor used to perform basic text transformations on an input stream (file or input piped to it).

Commands:

Replace all occurrences of "cat" with "monkey" in a file:
$ sed 's/cat/monkey/g' file.txt (just prints, no action, temporary,)
$ sed -i 's/cat/monkey/g' file.txt (actually writes/changes to the file)

Delete lines containing "error":
$ sed '/error/d' file.txt

Print only lines 3 to 5:
$ sed -n '3,5p' file.txt

Stream text from one file (local.env) to another (prod.env):
$ $ sed 's/locat@123/prod@123/g' local.env > prod.env