Web Server on LAN

Team 23
10.134.178.23

Authors - Tymur Slesarenko, Aman Singh, Willem Schillemans

Task 1: User accounts for team members

1. Create accounts for all team members **2.** Add them to the secondary group

Script was used to create users and add them into a group

```
#!/bin/bash

# Define the users to create
users=("aman" "tymur" "ilias" "willem" "emre")

# Loop through the array and create each user
for user in "${users[@]}"; do
  sudo useradd $user
done

# Add the users to the developers group
sudo usermod -aG developers "${users[@]}"
```

This script first defines the user list at the top and uses a for loop to iterate through the list and adds each user. Then adds each user to the group “developers”

3. Grant sudo rights to the group members. Do so by adding a config file in the configuration "drop-in" directory sudoers.d

```
  GNU nano 5.6.1                                students
tymur ALL=(ALL:ALL) ALL
aman ALL=(ALL:ALL) ALL
willem ALL=(ALL:ALL) ALL
emre ALL=(ALL:ALL) ALL
ilias ALL=(ALL:ALL) ALL
```

4. Change the permissions of this file so that only the owner root can read/write the config file.

```
root@team23int2: /etc/sudoers.d
# chmod 600 students
```

5. Remove the initial account when tasks have been completed successfully.

```
root@team23int2: ~
# userdel team
```

Task 2: Passwordless ssh access

1. Create an ed25519 key pair

```
[aman@localhost ~]$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/aman/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/aman/.ssh/id_ed25519
Your public key has been saved in /home/aman/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:bxe3BWlSUAeixvJsapzxha43pNILhEpQV9E/9a/IfZM aman@localhost.localdomain
The key's randomart image is:
+--[ED25519 256]--+
|  . ..oo    oo+..|
| . .    .. ..o o |
|.       ..+...+  |
| .   .   =o. o.. |
|  . . . S =.o ...|
| . . . . O.. o o.|
|  .   ..=o=..o...|
|      .oooo.o oE.|
|       .oo .   ..|
+----[SHA256]-----+
```

2. Transfer the public key to your user account on the team server and connect to the server over ssh with the command ssh user@10.134.178.23.

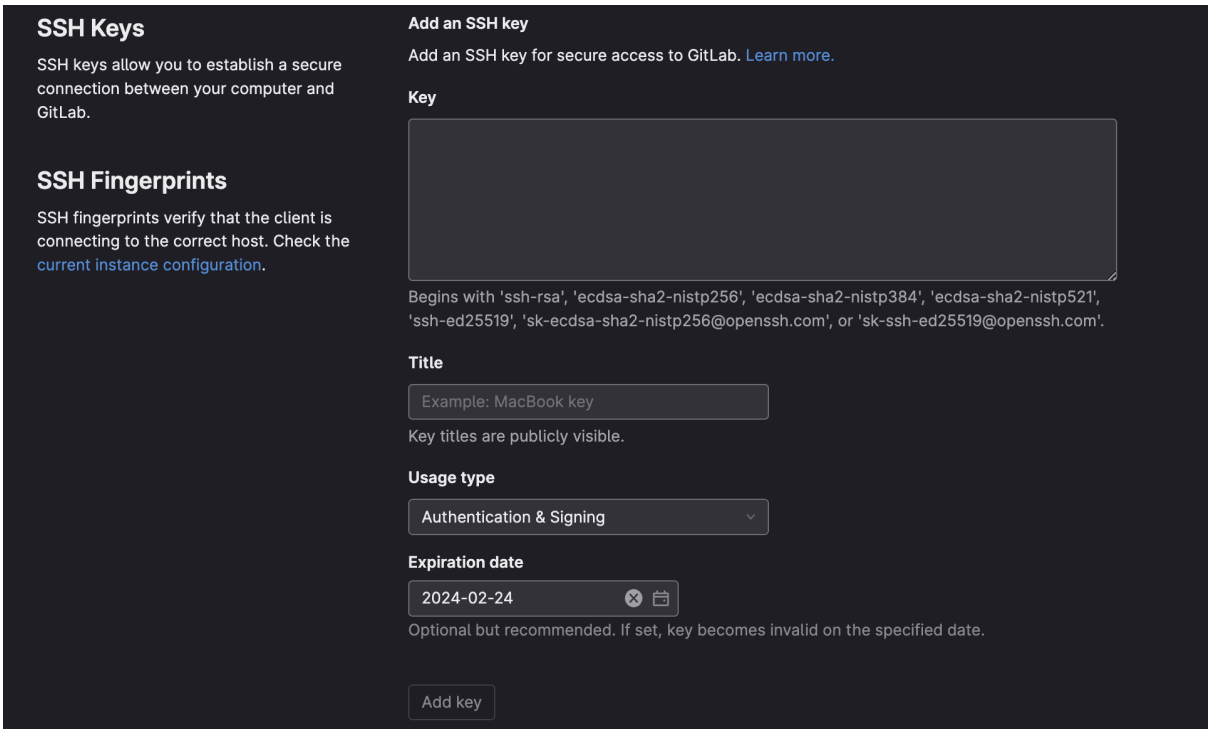
```
[aman@localhost ~]$ ssh-copy-id aman@10.134.178.23
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompt
ed now it is to install the new keys

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'aman@10.134.178.23'"
and check to make sure that only the key(s) you wanted were added.

[aman@localhost ~]$ ssh aman@10.134.178.23
Last login: Wed Feb 15 00:28:50 2023

aman@team23int2: ~
$
```

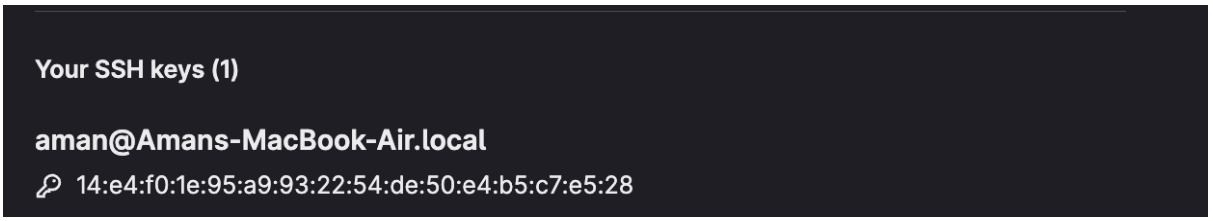
3. Passwordless authentication with gitlab

- First, generated a key pair in /Users/aman/.ssh/key_gitlab

```
aman@Amans-MacBook-Air JS % ssh-keygen -f /Users/aman/.ssh/key_gitlab
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /Users/aman/.ssh/key_gitlab
Your public key has been saved in /Users/aman/.ssh/key_gitlab.pub
The key fingerprint is:
SHA256:YK0bwuVOXpERvHAEERYX2GqONZqiP2lHIp7h5dHWYCk aman@Amans-MacBook-Air.local
The key's randomart image is:
+---[RSA 3072]----+
|      *X*o       |
|     .oo+o       |
|      =++.       |
|   E B=o..       |
|    *BB.S        |
|..oo=B.=         |
|oo==o +          |
|.++..            |
| o.o             |
+----[SHA256]-----+
aman@Amans-MacBook-Air JS %
```

- I copied the public key's content to my gitlab account using GUI.





- This can be done in terminal too with the command ssh-copy-id -i {path to the public key} user@remothost

The ssh-copy-id command copies the public key of the SSH keypair to the destination system. If you omit the path to the public key file while running ssh-copy-id,it uses the default **/home/user/.ssh/id_rsa.pub** file.

After the public key is successfully transferred to a remote system, you can authenticate to the remote system using the corresponding private key while logging in to the remote system over SSH.

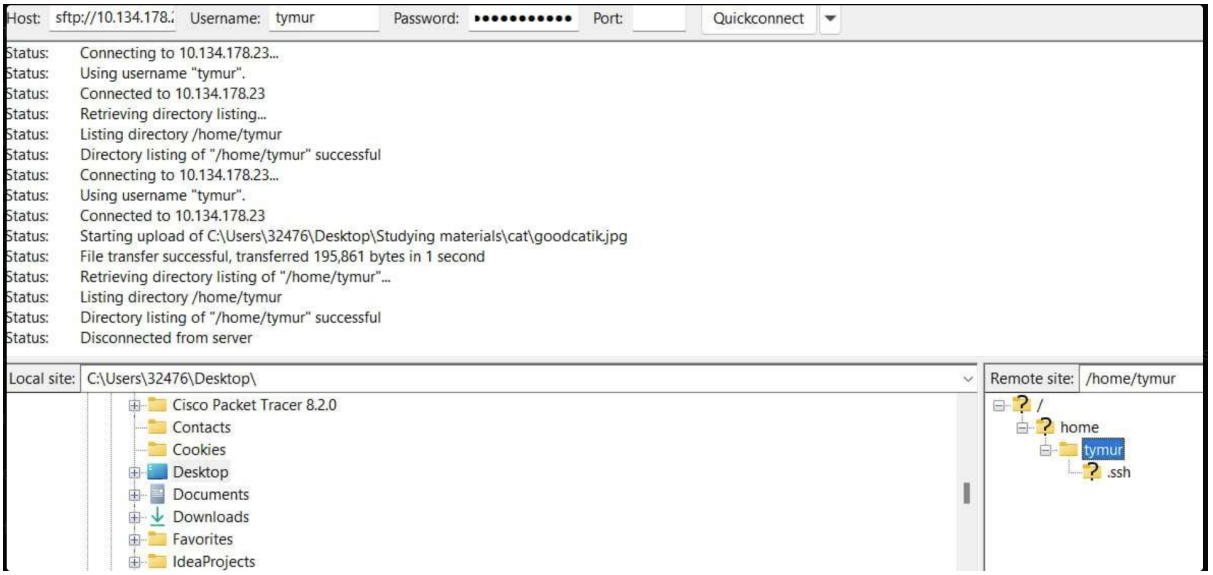
```
aman@Amans-MacBook-Air ~ % ssh git@gitlab.com
PTY allocation request failed on channel 0
Welcome to GitLab, @aman.singh11!
Connection to gitlab.com closed.
```

Task 3: Additional network configuration

1. Adding a new connection “instructors” which uses ens19 with a static IPv4 address 192.168.1.23 using /24 subnet mask and bringing it up using nmcli con up command

```
root@team23int2: ~
# nmcli con add con-name instructors ifname ens19 type ethernet ipv4.address 192
.168.1.23/24 ipv4.gateway ''
Connection 'instructors' (15b4dbde-570e-432f-b0ba-01321c8f112b) successfully add
ed.

root@team23int2: ~
# nmcli con up instructors
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveC
onnection/3)

root@team23int2: ~
# nmcli con show
NAME                UUID                                  TYPE      DEVICE
ens18               4b465f2d-9419-39c8-805b-a738fa695b3c  ethernet  ens18
instructors         15b4dbde-570e-432f-b0ba-01321c8f112b  ethernet  ens19
Wired connection 1  2626f79e-76f7-3054-a609-316be7d11078  ethernet  --
```

2. Transfering a jpg image to the server



3. Putting the image to the instructors server 192.168.1.250 using SFTP

```
tymur@team23int2: ~
$ sftp team@192.168.1.250
The authenticity of host '192.168.1.250 (192.168.1.250)' can't be established.
ED25519 key fingerprint is SHA256:G4GIDu8PFJcT7ESIzVPQ2ZSZRL7Dh5XVbq/IiMaywEw.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:1: 10.134.178.23
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.250' (ED25519) to the list of known hosts.
team@192.168.1.250's password:
Connected to 192.168.1.250.
sftp> p
progress  put       pwd
sftp> put team23.jpg
Uploading team23.jpg to /home/team/team23.jpg
team23.jpg                                            100%  191KB  54.6MB/s   00:00
sftp> ls
IN_HERE_GOES_YOUR_TEAM_PICTURE                check_upload.sh
new_upload_check                              team07.jpg
team12.jpg                                    team13.JPG
team15.jpg                                    team16.jpg
team18.jpg                                    team2.jpg
team21.jpg                                    team22.JPG
team22.jpg                                    team23.jpg
team28.jpg                                    team3.jpg
team30.jpg                                    team6.jpg
sftp>
```

Task 4: Install PostgreSQL

1. Updating our Linux system:

```
root@team23int2: ~
# sudo dnf update
```

2. Install package postgresql-server

```
root@team23int2: ~
# sudo dnf install postgresql-server -y
```

3. Initialize the postgresql service with:

```
root@team23int2: ~
# sudo /usr/bin/postgresql-setup --initdb
 * Initializing database in '/var/lib/pgsql/data'
 * Initialized, logs are in /var/lib/pgsql/initdb_postgresql.log
```

4. Check if postgresql service is running

```
root@team23int2: ~
# systemctl status postgresql.service
○ postgresql.service - PostgreSQL database server
     Loaded: loaded (/usr/lib/systemd/system/postgresql.service; disabled; vend
     Active: inactive (dead)
```

5. Start the service

```
root@team23int2: ~
# systemctl start postgresql.service
```

6. Enable the service

```
root@team23int2: ~
# systemctl enable postgresql.service
Created symlink /etc/systemd/system/multi-user.target.wants/postgresql.service -
 /usr/lib/systemd/system/postgresql.service.
```

7. Check the status of the service

```
root@team23int2: ~
# systemctl status postgresql.service
● postgresql.service - PostgreSQL database server
     Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; ven
     Active: active (running) since Mon 2023-03-13 11:45:52 CET; 9s ago
   Main PID: 98661 (postmaster)
      Tasks: 8 (limit: 11078)
     Memory: 16.9M
        CPU: 68ms
     CGroup: /system.slice/postgresql.service
             ├─98661 /usr/bin/postmaster -D /var/lib/pgsql/data
             ├─98662 "postgres: logger "
             ├─98664 "postgres: checkpointer "
             ├─98665 "postgres: background writer "
             ├─98666 "postgres: walwriter "
             ├─98667 "postgres: autovacuum launcher "
             ├─98668 "postgres: stats collector "
             └─98669 "postgres: logical replication launcher "
```

8. Edit the configuration file /var/lib/pgsql/data/postgresql.conf

```
root@team23int2: ~
# nano /var/lib/pgsql/data/postgresql.conf
```

9. Add the listen address (IP address of the team server)

```
# CONNECTIONS AND AUTHENTICATION
#------------------------------------------------------------------------------

# - Connection Settings -

listen_addresses = '10.134.178.23'            # what IP address(es) to list
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost'; use '*' fo
                                        # (change requires restart)
#port = 5432                            # (change requires restart)
max_connections = 100                   # (change requires restart)
#superuser_reserved_connections = 3     # (change requires restart)
#unix_socket_directories = '/var/run/postgresql, /tmp'  # comma-separated lis
                                        # (change requires restart)
#unix_socket_group = ''                 # (change requires restart)
#unix_socket_permissions = 0777         # begin with 0 to use octal notation
                                        # (change requires restart)
```

10. Restart the server

```
root@team23int2: ~
# systemctl restart postgresql.service
```

11. Edit the configuration file /var/lib/pgsql/data/pg_hba.conf

```
root@team23int2: ~
# nano /var/lib/pgsql/data/pg_hba.conf
```

12. Add the line to permit all hosts from A class subnet 10.0.0.0 to access the database

```
# configuration parameter, or via the -i or -h command line switches.



# TYPE  DATABASE        USER            ADDRESS                 METHOD

# "local" is for Unix domain socket connections only
local   all             all                                     peer
# IPv4 local connections:
host    all             all             127.0.0.1/32            ident
# IPv6 local connections:
host    all             all             ::1/128                 ident
# Allow replication connections from localhost, by a user with the
# replication privilege.
local   replication     all                                     peer
host    replication     all             127.0.0.1/32            ident
host    replication     all             ::1/128                 ident
host    all             all             10.0.0.0/8              md5
```

13.

14. Become the linux user “postgres”, create a new user "game" with password '7sur7' and grant it all permissions to a new database "game".

```
root@team23int2: ~
# sudo -u postgres psql
could not change directory to "/root": Permission denied
psql (13.7)
Type "help" for help.

postgres=# CREATE DATABASE game;
CREATE DATABASE
postgres=# CREATE USER game WITH PASSWORD '7sur7';
CREATE ROLE
postgres=# GRANT ALL PRIVILEGES ON DATABASE game TO game;
GRANT
postgres=# \q
```

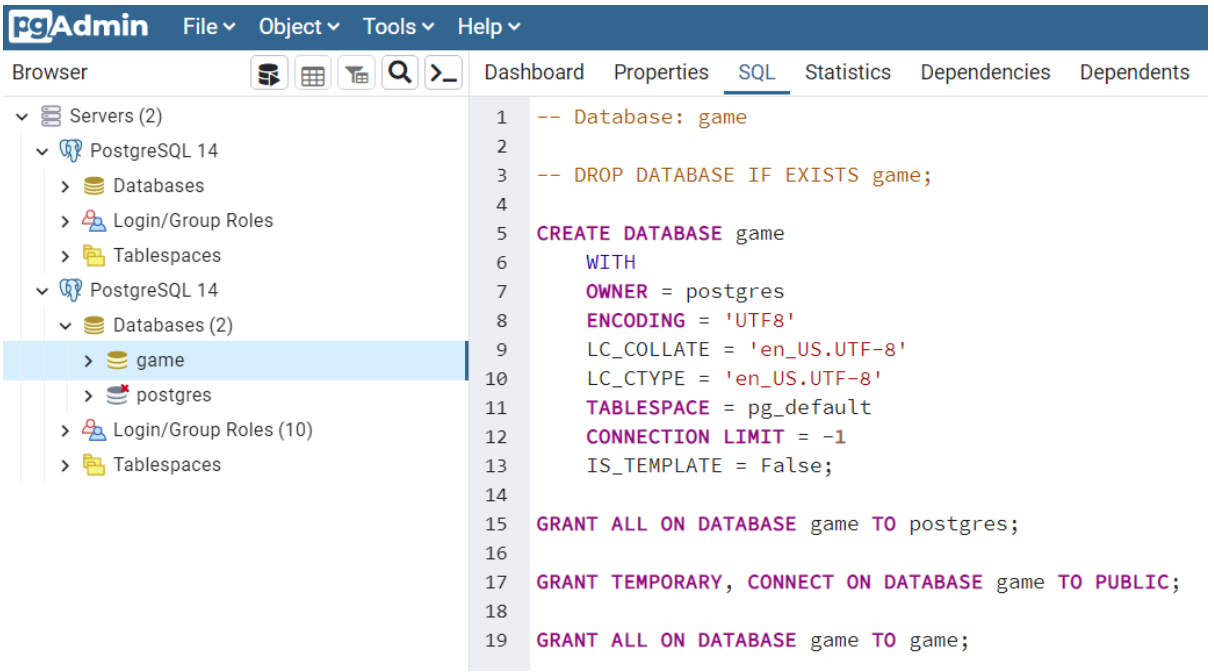
15. Configure the

```
root@team23int2: /opt/scripts
# firewall-cmd --state
running

root@team23int2: /opt/scripts
# firewall-cmd --add-service=postgresql --permanent
success

root@team23int2: /opt/scripts
# firewall-cmd --reload
success

root@team23int2: /opt/scripts
#
```

16. Find out on which port postgres clients connect to the server

```
root@team23int2: ~
# ss -tln
State      Recv-Q     Send-Q         Local Address:Port         Peer Address:Port    Process
LISTEN     0          128                  0.0.0.0:22                0.0.0.0:*
LISTEN     0          244            10.134.178.23:5432              0.0.0.0:*
LISTEN     0          511                        *:80                      *:*
LISTEN     0          128                     [::]:22                   [::]:*
```

17. Successfully connected to the database



PS. You can access the database on the server using command:
$ psql -U game -d game -h 10.134.178.23

Task 5: Webserver

1. Check the status of httpd service, start and enable it

```
root@team23int2: ~
# systemctl status httpd
o httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor pr>
     Active: inactive (dead)
       Docs: man:httpd.service(8)


root@team23int2: ~
# systemctl start httpd

root@team23int2: ~
# systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr
/lib/systemd/system/httpd.service.
```

Check the status again

```
root@team23int2: ~
# systemctl status httpd
● httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor pre>
     Active: active (running) since Mon 2023-03-13 13:58:14 CET; 9s ago
       Docs: man:httpd.service(8)
   Main PID: 99455 (httpd)
     Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes>
      Tasks: 213 (limit: 11078)
     Memory: 33.2M
        CPU: 143ms
     CGroup: /system.slice/httpd.service
             ├─99455 /usr/sbin/httpd -DFOREGROUND
```

2. Check if the web server is accepting connections on port 80

```
root@team23int2: ~
# telnet 10.134.178.23 80
Trying 10.134.178.23...
Connected to 10.134.178.23.
```

3. Creating a test file in /var/www/html

```
root@team23int2: /var/www/html
# nano testfile.html
```

```
  GNU nano 5.6.1
<h1>It is working!</h1>
```

4. Configure the firewall

```
root@team23int2: /var/www/html
# firewall-cmd --zone=public --permanent --add-service=h
ttp
success

root@team23int2: /var/www/html
# firewall-cmd --reload
success
```

5. It is working!

Niet beveiligd | 10.134.178.23/testfile.html

It is working!

Task 6: RSYSLOG configuration

1. Open the rsyslog.d file and add the following line to the file to configure the local6 facility to log messages to /var/log/team23.log

```
root@team23int2: /etc/rsyslog.d
# sudo nano /etc/rsyslog.d/team23.conf
```

```
local6.*    /var/log/team23.log
```

2. Restart the rsyslog service

```
root@team23int2: ~
# sudo service rsyslog restart
Redirecting to /bin/systemctl restart rsyslog.service
```

3. Test the syslog facility by running logger command

```
root@team23int2: ~
# logger -p local6.info "This is a test message for local6 facility"
```

4. View the contents of the file

```
root@team23int2: /etc/rsyslog.d
# cat /var/log/team23.log
Mar 13 15:27:25 team23int2 root[100685]: This is a test message for local6 facility
```

5. Configure the journal

```
root@team23int2: /etc/rsyslog.d
# sudo systemd-tmpfiles --create --prefix /var/log/team23.log
```

```
root@team23int2: /etc/rsyslog.d
# sudo nano /etc/systemd/journald.conf
```

6. Add the following line to the [Journal] section

```
[Journal]
Storage=persistent
#Compress=yes
#Seal=yes
#SplitMode=uid
#SyncIntervalSec=5m
#RateLimitIntervalSec=30s
#RateLimitBurst=10000
#SystemMaxUse=
#SystemKeepFree=
```

7. Restart the systemd-jornald service