SO Parcial 2: Cómo poner passkey y conectarse al servidor en la nube

---

1) Dónde poner `awskey.pem` en el Ubuntu nuevo (tu PC cliente)

Colócalo en tu carpeta `~/.ssh` (sitio estándar y seguro).

Desde tu Ubuntu (terminal):

```
# 1. ir al home
cd ~

# 2. crear .ssh si no existe y fijar permisos de la carpeta
mkdir -p ~/.ssh
chmod 700 ~/.ssh

# 3. copiar la llave desde donde la descargaste (ejemplo: ~/Descargas)
# ajusta la ruta si tu awskey.pem está en otra carpeta
cp ~/Descargas/awskey.pem ~/.ssh/awskey.pem

# 4. poner permisos correctos (obligatorio para SSH)
chmod 400 ~/.ssh/awskey.pem

# comprobar
ls -l ~/.ssh/awskey.pem
# salida esperada: -r-------- 1 tuusuario tuusuario ... awskey.pem
```

Si la llave ya está en `/home/tuusuario/Downloads` usa esa ruta en el `cp`.

---

2) Dónde poner `collector.c` en tu Ubuntu (antes de subirlo)

Crea una carpeta del proyecto y coloca ahí el archivo.

```
# ejemplo de carpeta proyecto
mkdir -p ~/so/parcial_2
cd ~/so/parcial_2

# Si collector.c está en Descargas:
cp ~/Descargas/collector.c .

# verificar
ls -l collector.c
```

(Alternativa: puedes editarlo con `nano` o `code .` en VSCode si hace falta.)

---

3) Subir `collector.c` al servidor AWS (con `scp`)

Sube `collector.c` a `/home/ubuntu` del EC2 (usuario `ubuntu` por defecto en AMIs Ubuntu):

```
# desde tu carpeta donde esté collector.c
scp -i ~/.ssh/awskey.pem collector.c ubuntu@13.59.14.144:/home/ubuntu/
```

Salida esperada: la barra de progreso y que termine sin error.
Si da `Permission denied (publickey)` vuelve a revisar `chmod 400 ~/.ssh/awskey.pem` y que estés usando la llave correcta.

---

4) Conectarte por SSH al servidor AWS

```
ssh -i ~/.ssh/awskey.pem ubuntu@13.59.14.144
```

Si todo ok verás el prompt `ubuntu@ip-...:~$` (estás dentro del servidor).

---

5) Compilar y ejecutar `collector` dentro del servidor AWS

Una vez conectado por SSH:

```
# (dentro del servidor)
ls -l collector.c      # confirmar que collector.c está en
/home/ubuntu
sudo apt update
sudo apt install -y build-essential   # sólo si gcc no está instalado

# compilar
gcc -std=c11 -Wall -Wextra -pthread -o collector collector.c

# verificar binario
ls -l collector
```

Ejecutar en primer plano (útil para ver la tabla en la consola):

```
./collector 9000
```

Verás la cabecera de la tabla y luego filas cuando lleguen agentes.

---

6) Ejecutar `collector` en segundo plano (para dejarlo corriendo después de cerrar SSH)

Opciones sencillas:

a) Con `nohup` (rápido y simple)

```
nohup ./collector 9000 > collector.log 2>&1 &
# ver salida:
tail -n 50 collector.log
# buscar proceso:
ps aux | grep collector
```

b) Con `screen` o `tmux` (más robusto -> te permite reconectar a la sesión)

Si quieres usar `screen`:

```
sudo apt install -y screen
screen -S collector
# dentro de screen:
./collector 9000
# para despegar: Ctrl-A then D  (detach)
# para volver a entrar:
screen -r collector
```

---

7) Verificar que el collector escucha en el puerto 9000 (en el servidor AWS)

```
# netstat (si no está, instala: sudo apt install -y net-tools)
sudo netstat -tulpen | grep 9000

# o con ss
ss -tuln | grep 9000
```

Deberías ver `LISTEN` en 0.0.0.0:9000 o en la IP pública interna.

IMPORTANTE: además de esto, en tu Security Group de AWS debe estar abierta la regla TCP 9000 desde la fuente que necesites (0.0.0.0/0 para pruebas).

---

8) Probar desde ese otro Ubuntu (ejecutar agentes y conectarse al collector AWS)

En tu otro computador Ubuntu (el que quieres usar para probar), compila los agentes (si tienes `agent_mem.c` y `agent_cpu.c`):

```
# suponer que los copiaste a ~/so/parcial_2 en ese PC
cd ~/so/parcial_2

gcc -std=c11 -Wall -Wextra -o agent_mem agent_mem.c
gcc -std=c11 -Wall -Wextra -o agent_cpu agent_cpu.c
```

Ejecutar y apuntar al collector en AWS:

```
# en terminal 1: agente memoria
./agent_mem 13.59.14.144 9000 MiPC-MEM

# en otra terminal: agente CPU
./agent_cpu 13.59.14.144 9000 MiPC-CPU
```

En la consola del **collector en AWS** (o en `collector.log` si usaste nohup) verás las filas actualizándose.

---

9) Comandos útiles para diagnosticar problemas

Si `scp` o `ssh` falla con `Permission denied (publickey)`:

```
ls -l ~/.ssh/awskey.pem
chmod 400 ~/.ssh/awskey.pem
```

-
- Si no llegan datos al collector:
    - Asegúrate de que el collector esté corriendo en AWS (ps/ss/netstat).
    - Asegura la regla en Security Group: TCP 9000 abierto a la IP origen (0.0.0.0/0 para pruebas).
    - Desde el PC cliente prueba `telnet 13.59.14.144 9000` o `nc -vz 13.59.14.144 9000` para verificar conexión TCP.

Ver logs del collector (si usaste nohup):

```
tail -f ~/collector.log
```

-

Parar el collector:

```
# encontrar PID
ps aux | grep collector
kill <PID>
# o si lo ejecutaste con nohup y background:
pkill -f ./collector
```

-

---

10) Resumen breve — comandos clave (copy/paste)

En tu Ubuntu local (cliente):

```
mkdir -p ~/.ssh && chmod 700 ~/.ssh
cp ~/Descargas/awskey.pem ~/.ssh/awskey.pem
chmod 400 ~/.ssh/awskey.pem
mkdir -p ~/so/parcial_2
cp ~/Descargas/collector.c ~/so/parcial_2/
cd ~/so/parcial_2
scp -i ~/.ssh/awskey.pem collector.c ubuntu@13.59.14.144:/home/ubuntu/
```

En el servidor AWS (por SSH):

```
ssh -i ~/.ssh/awskey.pem ubuntu@13.59.14.144
# una vez dentro:
sudo apt update && sudo apt install -y build-essential
gcc -std=c11 -Wall -Wextra -pthread -o collector collector.c
# ejecutar en foreground:
./collector 9000
# o en background:
nohup ./collector 9000 > collector.log 2>&1 &
```

En el cliente (para probar agentes):

```
gcc -o agent_mem agent_mem.c
gcc -o agent_cpu agent_cpu.c
./agent_mem 13.59.14.144 9000 MiPC-MEM
./agent_cpu 13.59.14.144 9000 MiPC-CPU
```

Perfecto manito, te explico **clarito y sin vueltas** qué hace este comando:

```
scp -i ~/.ssh/awskey.pem collector.c ubuntu@13.59.14.144:/home/ubuntu/
```

🧠 ¿Qué es `scp`?

`scp` significa **Secure Copy**
Sirve para **copiar archivos entre tu PC y un servidor remoto**, usando SSH.

Es como un **copiar/pegar**, pero hacia tu servidor de AWS.

🧩 Traducción completa en palabras humanas

> "Copia el archivo `collector.c` desde mi computadora **hasta** el servidor de AWS, a la carpeta `/home/ubuntu/`, usando mi llave privada para iniciar sesión."

Perfecto manito, te explico **qué hace este comando**, parte por parte:

```
ssh -i ~/.ssh/awskey.pem ubuntu@13.59.14.144
```

🧠 ¿Qué es `ssh`?

`ssh` significa **Secure SHell**
Es la herramienta que te permite:

✅ entrar a una máquina remota
🖥️ y usarla como si estuvieras sentado frente a ella

En este caso, tu máquina remota es **tu servidor de Amazon AWS EC2**.

🧩 Traducción a lenguaje humano

> "Entra a mi servidor AWS, usando la llave privada para identificarme."

Es como abrir la puerta de tu apartamento en la nube.
