Prompt

Teach me SCP for AWS EC2 and Linux administration.

Cover:

* Why SCP exists
* How SCP uses SSH
* Production use cases

Then teach:

1. Local → EC2
2. EC2 → Local
3. Directory transfer
4. Server → Server transfer (first we have to do passwordless authentication)

For each:

* Command breakdown
* Practical implementation
* Verification

Finally:

* SCP vs RSYNC
* Advantages & limitations
* 10 interview questions

Keep it concise, practical, and interview-focused.
Wait for my confirmation before moving ahead.

SCP

SCP (Secure Copy Protocol) — AWS EC2 & Linux Administration

We'll do this in parts and stop after the foundation. Once you're comfortable, we'll move to **Local ↔ EC2 transfers**.

---

1. Why SCP Exists

Before SCP, administrators commonly used:

- FTP
- RCP (Remote Copy)

Problem:

- Data transferred in plain text
- Credentials exposed
- No encryption

SCP was introduced to provide:

✅ Secure file transfer
✅ Authentication via SSH
✅ Encrypted data in transit
✅ Simple command-line usage

Think of SCP as: "cp command for remote systems."

Local copy:

cp file.txt /tmp/

Remote copy:

scp file.txt user@server:/tmp/

---

2. How SCP Uses SSH

This is the most important interview concept.

SCP does **not** create its own secure channel.

Instead:

SCP

SSH Connection

Encrypted Transfer

When you run:

scp -i mykey.pem file.txt ec2-user@54.x.x.x:/home/ec2-user/

SCP:

Step 1

Uses SSH to authenticate.

-i mykey.pem

Private key proves your identity.

Step 2

SSH session established.

Your Laptop

|

SSH

|

EC2

Step 3

File transferred through encrypted SSH tunnel.

No separate port required.

Network Requirement

Only:

TCP 22

must be accessible.

Security Group:

Inbound:

SSH (22)

Source: Your IP

No SCP-specific port exists.

Interview question:

> Which port does SCP use?

Answer:

Port 22 (SSH)

SCP Authentication Methods

Method 1: Password Authentication

Linux server:

scp file.txt user@server:/tmp/

Server asks:

password:

Common on on-prem servers.

Method 2: Key-Based Authentication

AWS standard approach.

scp -i mykey.pem file.txt ec2-user@server:/tmp/

Uses:

Private Key

+

authorized_keys

No password required.

Production Use Cases

1. Deploy Application Files

Upload application package:

Laptop

|

SCP

▼

EC2

Example:

app.zip

config.yaml

nginx.conf

---

2. Backup Retrieval

Download logs:

EC2

|

▼

Laptop

Example:

/var/log/messages

/application/logs

---

3. Configuration Migration

Move configs between servers:

Server A

  |

SCP

  ▼

Server B

Examples:

nginx.conf

httpd.conf

systemd files

---

4. SSL Certificate Transfer

Transfer:

.crt

.key

.pem

between administration servers.

---

5. Database Export Movement

Move:

mysqldump.sql

backup.tar.gz

to backup servers.

Key Points

- SCP depends on SSH.
- Port 22 must be reachable.
- Same key pair used for SSH can be used for SCP.
- Security Group must allow SSH access.

Interview Nuggets

Difference between SSH and SCP

SSH:

Remote login

SCP:

File transfer using SSH

Can SCP work without SSH?

No.

Is SCP encrypted?

Yes.

Entire transfer is encrypted by SSH.

Does SCP preserve permissions?

Basic file permissions are copied.

For preserving timestamps and metadata:

scp -p file.txt server:/tmp/

What you've learned so far

✔ Why SCP exists
✔ How SCP works internally
✔ SCP + SSH relationship
✔ Authentication methods
✔ Production use cases
✔ AWS networking requirement (Port 22)

Scenario 1: Local → EC2

Command

scp -i mykey.pem file.txt ec2-user@54.x.x.x:/home/ec2-user/

Command Breakdown

Part	Meaning
scp	Secure Copy
-i mykey.pem	SSH private key
file.txt	Source file
ec2-user	Remote user
54.x.x.x	Public IP/DNS
/home/ec2-user/	Destination

Practical Example

Local:

ls

Output:

file.txt

Upload: scp -i mykey.pem file.txt ec2-user@54.x.x.x:/home/ec2-user/

Verification

SSH into instance: ssh -i mykey.pem ec2-user@54.x.x.x

Check: ls -l /home/ec2-user/

Output: file.txt

Transfer successful.

```
PS C:\Users\Aditi Narang\Downloads> scp -i "narang.pem" docker.pdf ec2-user@13.233.236.235:/home/ec2-user/
The authenticity of host '13.233.236.235 (13.233.236.235)' can't be established.
ED25519 key fingerprint is SHA256:3Bx0dQGoSE6UMRn0K6LVCeR965EIdKhzNKyT2JPz0x0.
This host key is known by the following other names/addresses:
    C:\Users\Aditi Narang/.ssh/known_hosts:28: ec2-13-233-236-235.ap-south-1.compute.amazonaws.com
Are you sure you want to continue connecting (yes/no/[fingerprint])?
Please type 'yes', 'no' or the fingerprint:
Warning: Permanently added '13.233.236.235' (ED25519) to the list of known hosts.
docker.pdf                                                                    100% 1594KB   3.0MB/s   00:00
PS C:\Users\Aditi Narang\Downloads> ssh -i "narang.pem" ec2-user@ec2-13-233-236-235.ap-south-1.compute.amazonaws.com
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Sat Jun 20 10:14:59 2026 from 103.77.186.57
[ec2-user@ip-172-31-43-112 ~]$ whoami
ec2-user
[ec2-user@ip-172-31-43-112 ~]$ ls
docker.pdf
[ec2-user@ip-172-31-43-112 ~]$
```

Upload Multiple Files

scp -i mykey.pem file1.txt file2.txt file3.txt ec2-user@54.x.x.x:/home/ec2-user/

```
PS C:\Users\Aditi Narang\Downloads> scp -i "narang.pem" abcd.pdf AditiNarangPhoto.jpeg ec2-user@13.233.236.235:home/ec2-user/
C:\windows\System32\OpenSSH\scp.exe: remote mkdir "home/ec2-user/": No such file or directory
PS C:\Users\Aditi Narang\Downloads> scp -i "narang.pem" abcd.pdf AditiNarangPhoto.jpeg ec2-user@13.233.236.235:/home/ec2-user/
abcd.pdf                                                                      100%  151KB 363.1KB/s   00:00
AditiNarangPhoto.jpeg                                                         100%   67KB 344.3KB/s   00:00
PS C:\Users\Aditi Narang\Downloads> ssh -i "narang.pem" ec2-user@ec2-13-233-236-235.ap-south-1.compute.amazonaws.com
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Sat Jun 20 10:15:47 2026 from 103.77.186.57
[ec2-user@ip-172-31-43-112 ~]$ pwd
/home/ec2-user
[ec2-user@ip-172-31-43-112 ~]$ ls
AditiNarangPhoto.jpeg  abcd.pdf  docker.pdf
[ec2-user@ip-172-31-43-112 ~]$
```

Scenario 2: EC2 → Local

Download from server.

Command

scp -i mykey.pem ec2-user@54.x.x.x:/var/log/messages .

Breakdown

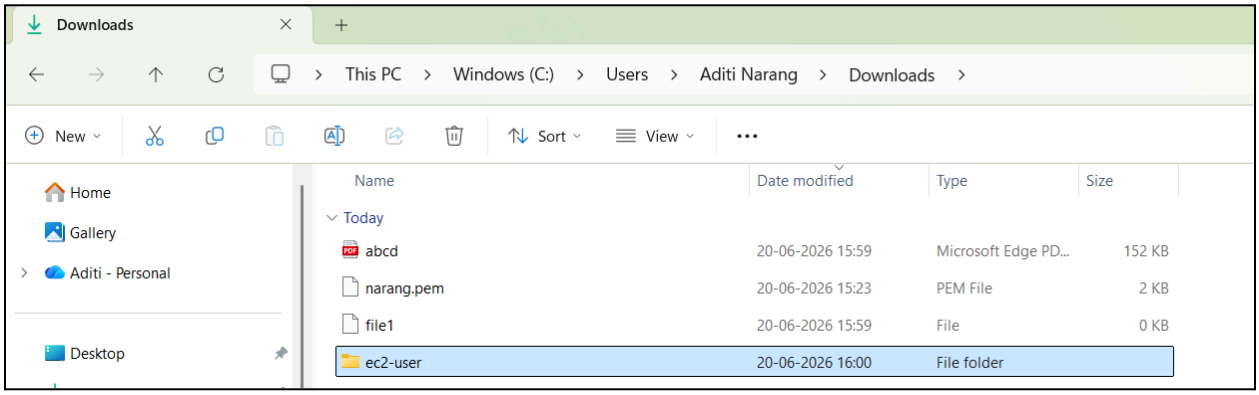
Part	Meaning
scp	Secure Copy
-i mykey.pem	SSH Key
ec2-user@54.x.x.x	Remote system
/var/log/messages	Source file
.	Current local directory

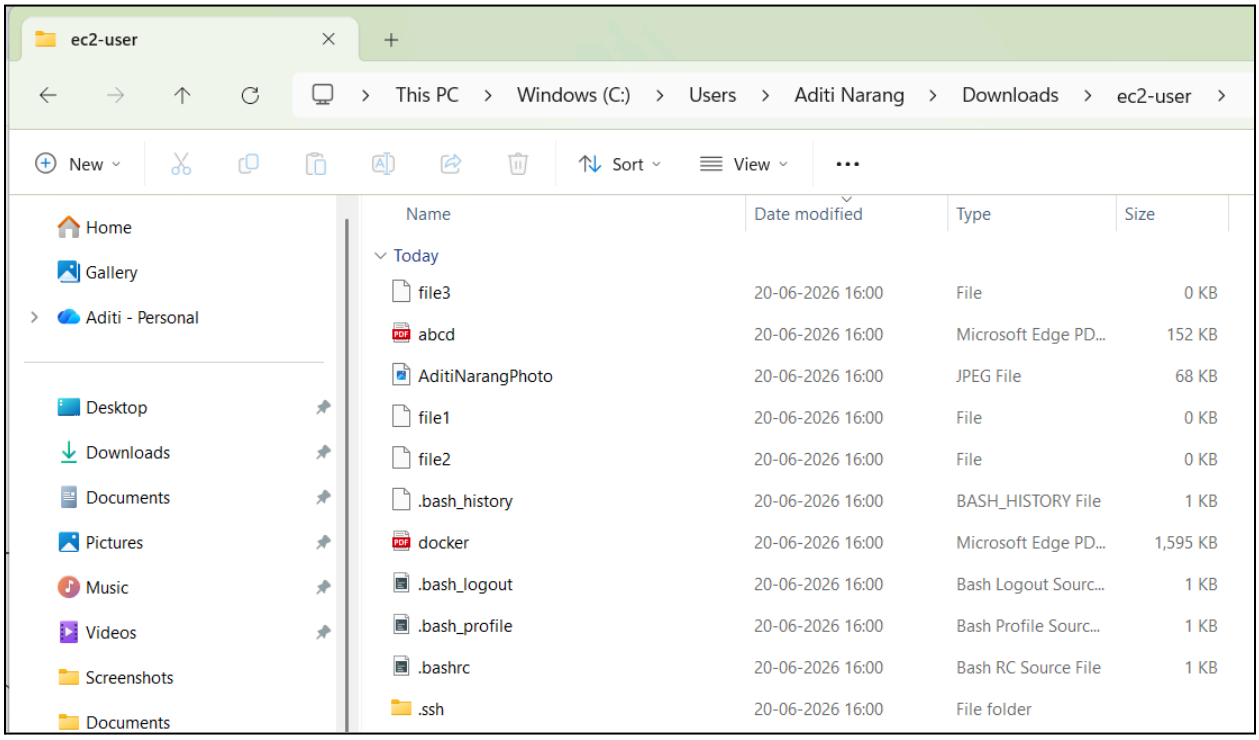
Practical Example

Download:

scp -i mykey.pem ec2-user@54.x.x.x:/etc/os-release .

```
PS C:\Users\Aditi Narang\Downloads> scp -i "narang.pem" ec2-user@13.233.236.235:/home/ec2-user/abcd.pdf .
abcd.pdf                                                                    100%  151KB 365.7KB/s   00:00
PS C:\Users\Aditi Narang\Downloads> ssh -i "narang.pem" ec2-user@ec2-13-233-236-235.ap-south-1.compute.amazonaws.com
       #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Sat Jun 20 10:23:17 2026 from 103.77.186.57
[ec2-user@ip-172-31-43-112 ~]$ ls
AditiNarangPhoto.jpeg  abcd.pdf  docker.pdf
[ec2-user@ip-172-31-43-112 ~]$ touch file1 file2 file3
[ec2-user@ip-172-31-43-112 ~]$ exit
logout
Connection to ec2-13-233-236-235.ap-south-1.compute.amazonaws.com closed.
PS C:\Users\Aditi Narang\Downloads> scp -i "narang.pem" ec2-user@13.233.236.235:/home/ec2-user/file1 .
PS C:\Users\Aditi Narang\Downloads> scp -r -i "narang.pem" ec2-user@13.233.236.235:/home/ec2-user/ .
.bash_logout                                                                100%   18     0.1KB/s   00:00
.bash_profile                                                               100%  141     0.7KB/s   00:00
.bashrc                                                                     100%  492     5.1KB/s   00:00
authorized_keys                                                             100%  388     2.1KB/s   00:00
.bash_history                                                               100%   79     0.4KB/s   00:00
docker.pdf                                                                  100% 1594KB   2.1MB/s   00:00
abcd.pdf                                                                    100%  151KB 694.5KB/s   00:00
AditiNarangPhoto.jpeg                                                       100%   67KB 381.5KB/s   00:00
PS C:\Users\Aditi Narang\Downloads>
```





Verification

Local:

cat os-release

File should exist locally.

Scenario 3: Directory Transfer

Without recursive option:

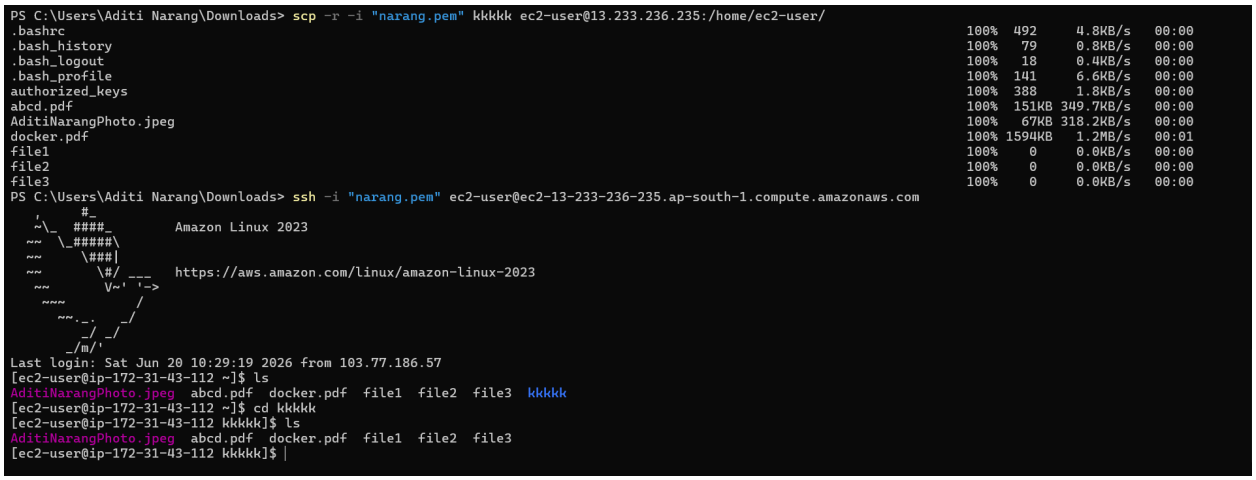
scp mydir ec2-user@server:/tmp/

Fails.

Because SCP copies files by default.

Recursive Copy

scp -r -i mykey.pem mydir ec2-user@54.x.x.x:/home/ec2-user/



Example

Local:

project/

├── app.py

├── config.yml

└── logs/

Transfer:

scp -r -i mykey.pem project ec2-user@54.x.x.x:/home/ec2-user/

Verification

EC2:

ls -R project

Entire directory should appear.

```
[ec2-user@ip-172-31-43-112 ~]$ ls -R kkkkk
kkkkk:
AditiNarangPhoto.jpeg  abcd.pdf  docker.pdf  file1  file2  file3
[ec2-user@ip-172-31-43-112 ~]$
```

Useful SCP Options

Preserve Timestamp & Permissions

scp -p file.txt ec2-user@server:/tmp/

Verbose Mode

Troubleshooting:

scp -v file.txt ec2-user@server:/tmp/

Shows SSH negotiation details.

Limit Bandwidth

scp -l 5000 file.txt ec2-user@server:/tmp/

Useful in production.

Scenario 4: Server → Server Transfer

This is a favorite interview topic.

Why Passwordless Authentication First?

Suppose:

Server A

Server B

A must authenticate to B.

If no passwordless setup exists:

scp file.txt ec2-user@serverB:/tmp/

will ask:

password:

Production automation cannot rely on manual password entry.

Therefore:

SSH Key-Based Authentication (This is done on Day 1, Following step1,2,3 are from day1)

is configured first.

Step 1: Generate Key on Server A

Login to Server A:

ssh-keygen

Files created:

~/.ssh/id_rsa

~/.ssh/id_rsa.pub

Step 2: Copy Public Key to Server B

View public key: cat ~/.ssh/id_rsa.pub

Append contents to: ~/.ssh/authorized_keys

on Server B.

Step 3: Verify Passwordless Login

From Server A:

ssh ec2-user@serverB

Expected:

Login succeeds

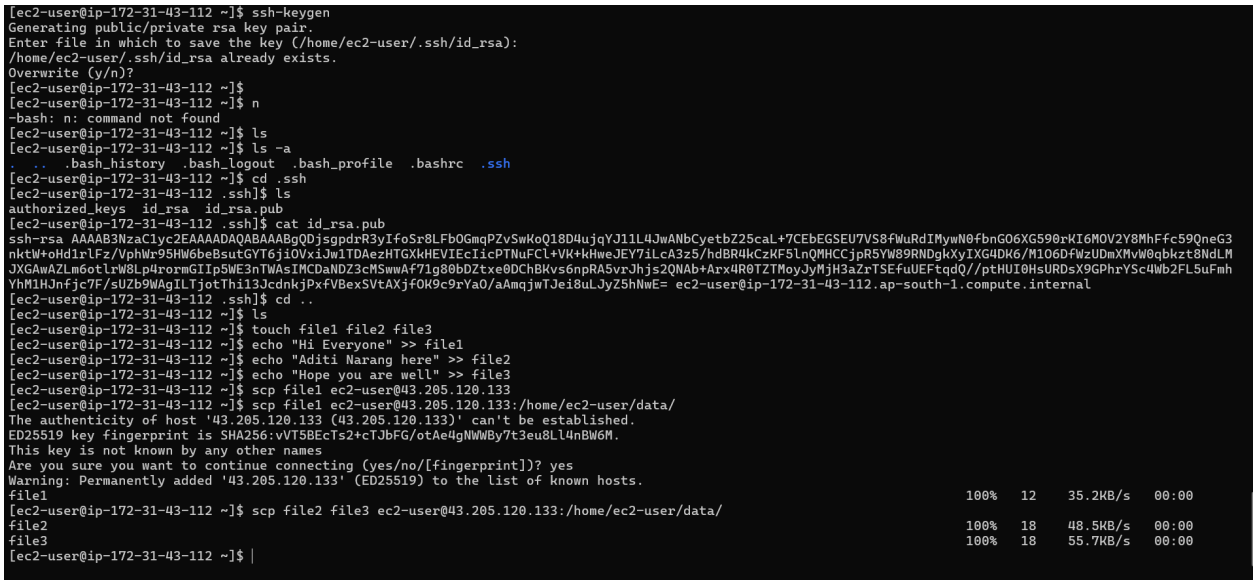
No password prompt

Step 4: SCP Server → Server

Now copy:

scp file2 file3 ec2-user@<server-2-ki-ip>:/home/ec2-user/data/

Server1:



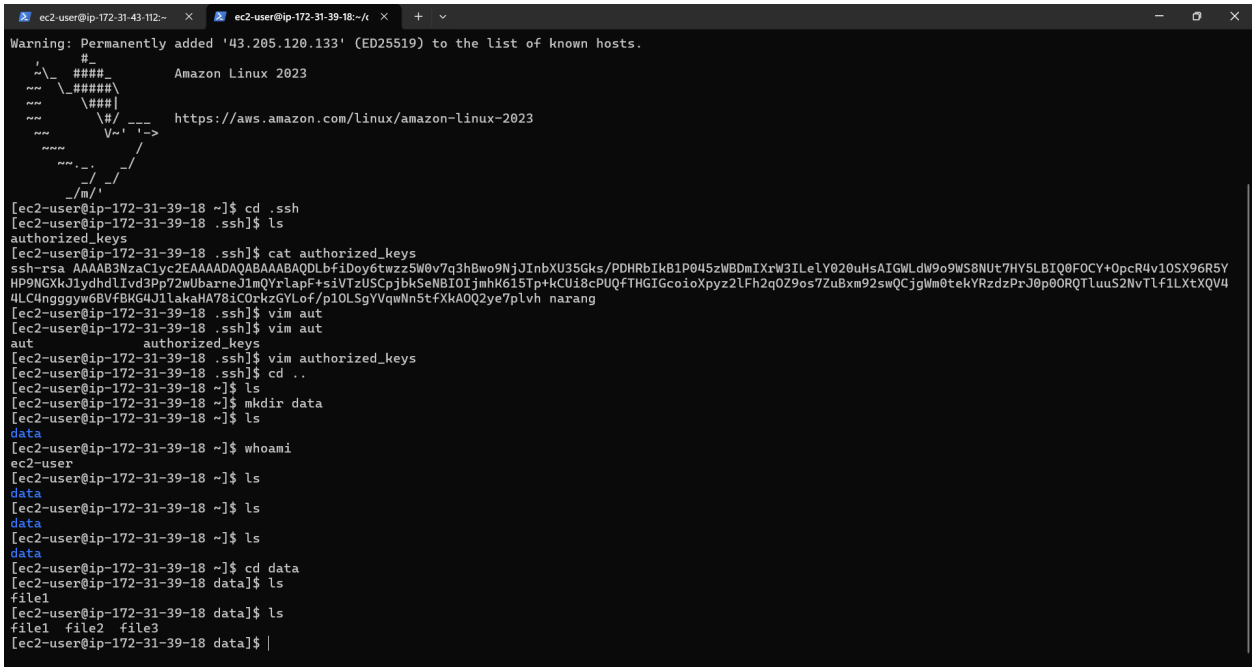
Verification

On Server B:

cd data
ls data

File should exist.

Server2:



Common Errors

Permission Denied

Permission denied (publickey)

Reasons:

- Wrong key
- Wrong username
- Missing authorized_keys entry
- Incorrect permissions

Connection Timed Out

ssh: connect to host ...

port 22: Connection timed out

Usually:

- Security Group issue
- NACL issue
- Route issue
- Instance not running

No Such File

scp: file.txt: No such file

Source path incorrect.

SCP vs RSYNC

Feature	SCP	RSYNC
Encryption	Yes (SSH)	Yes (SSH)
Simple	Yes	Moderate
Resume Transfer	No	Yes
Incremental Copy	No	Yes
Bandwidth Efficient	No	Yes

Large Data Sync	Poor	Excellent
Directory Sync	Basic	Advanced
Production Backup	Rare	Common

When to Use SCP

✅ Small file transfers

✅ Quick deployments

✅ Log downloads

✅ Configuration uploads

✅ One-time copy operations

When to Use RSYNC

✅ Large datasets

✅ Daily backups

✅ Synchronization jobs

✅ Repeated transfers

✅ Production backup systems

Advantages of SCP

Secure

Uses SSH encryption.

Easy

Single command.

Available Everywhere

Installed on almost all Linux systems.

No Additional Ports

Uses:

22/TCP

only.

Limitations of SCP

No Resume

If transfer stops at 99%:

Start again from 0%

No Incremental Sync

Copies everything every time.

Slow for Large Repeated Transfers

Compared with RSYNC.

Limited Features

Primarily designed for copying.

Not synchronization.

---

10 Interview Questions

1. What port does SCP use?

Port 22 (SSH).

---

2. Does SCP have its own authentication mechanism?

No. Uses SSH authentication.

---

3. Can SCP work without SSH?

No.

---

4. Difference between SSH and SCP?

SSH = Remote login

SCP = File transfer over SSH

---

5. How do you copy a directory using SCP?

scp -r directory user@server:/path

---

6. How do you preserve timestamps?

scp -p file.txt user@server:/path

---

7. Why is SCP not ideal for backups?

No resume and no incremental transfer.

---

8. SCP vs RSYNC?

RSYNC transfers only changed blocks and can resume interrupted transfers.

---

9. What causes `Permission denied (publickey)`?

Wrong key, username, permissions, or missing public key.

---

10. How do two EC2 instances perform SCP without passwords?

Configure SSH key-based authentication:

Server A public key

↓

Server B authorized_keys

Then SCP works non-interactively.

Interview Summary (Must Remember)

SCP = Secure Copy Protocol

Uses SSH

Uses Port 22

Local → EC2

EC2 → Local

Directory Copy (-r)

Server → Server :

Passwordless authentication required for automation

Good for:

- Small transfers
- Quick deployments

RSYNC preferred for:

- Backups
- Large data
- Synchronization