



SCP Basics

Basic Syntax

```
scp [options] source_file target_file
```

Where `source_file` can be a local file or a remote file in the format `user@host:path`.

And `target_file` can be a local directory, a local file, or a remote directory/file in the format `user@host:path`.

Copying a Local File to a Remote System

Copying a single file:

```
scp local_file.txt user@remote_host:/remote/directory/
```

Copying multiple files:

```
scp file1.txt file2.txt user@remote_host:/remote/directory/
```

Copying a Remote File to a Local System

Copying a single file:

```
scp user@remote_host:/remote/path/remote_file.txt /local/directory/
```

Copying multiple files:

```
scp user@remote_host:/remote/path/file1.txt user@remote_host:/remote/path/file2.txt /local/directory/
```

Advanced SCP Options

Port Specification

```
scp -P port source target
```

Specifies the port to connect to on the remote host. Useful when the SSH server listens on a non-standard port.

Example:

```
scp -P 2222 local_file.txt user@remote_host:/remote/directory/
```

Preserving Modification Times and Modes

```
scp -p source target
```

Preserves modification times, access times, and modes from the original file.

Example:

```
scp -p local_file.txt user@remote_host:/remote/directory/
```

Limiting Bandwidth

```
scp -l limit source target
```

Limits the bandwidth used by SCP, specified in Kbit/s.

Example:

```
scp -l 100 local_file.txt user@remote_host:/remote/directory/
```

Recursive Copy

```
scp -r source_directory target_directory
```

Recursively copies entire directories.

Example:

```
scp -r local_directory user@remote_host:/remote/directory/
```

Using a Specific Cipher

```
scp -c cipher source target
```

Selects the cipher to use for encrypting the data transfer. Check available ciphers with `ssh -Q cipher`.

Example:

```
scp -c blowfish local_file.txt user@remote_host:/remote/directory/
```

Security Considerations

Verifying Host Identity

SCP relies on SSH for secure communication. Ensure you verify the host identity when connecting to a new server to avoid man-in-the-middle attacks.

Check the host key fingerprint against a known trusted source.

Using SSH Keys

Using SSH keys for authentication is more secure than password-based authentication. Generate an SSH key pair using `ssh-keygen`.

Copy the public key to the remote server using `ssh-copy-id user@remote_host`.

You can specify the identity file with the `-i` option:

```
scp -i ~/.ssh/id_rsa local_file.txt user@remote_host:/remote/directory/
```

Disabling Password Authentication

For increased security, disable password authentication on the SSH server after setting up SSH key authentication. Edit `/etc/ssh/sshd_config` and set `PasswordAuthentication no`.

Restart the SSH service after making changes: `sudo systemctl restart sshd`.

Practical Examples

Copying Files Between Two Remote Servers

To copy directly between two remote servers, you can use a local machine as an intermediary, or use SSH tunneling.

Copying from remote1 to remote2 via local:

```
scp user1@remote1:/path/file.txt /tmp/ scp /tmp/file.txt user2@remote2:/path/
```

Using SCP with Wildcards

Wildcards can be used to copy multiple files at once. Be careful to escape them properly to prevent local shell expansion.

Example:

```
scp user@remote_host:/remote/path/*.txt /local/directory/
```

SCP with Verbose Output

Use the `-v` option for verbose output, which can be useful for debugging.

Example:

```
scp -v local_file.txt user@remote_host:/remote/directory/
```