



What Is SSH?

SSH COMMANDS

CHEAT SHEET

SSH (short for "Secure Shell" or "Secure Socket Shell") is a network protocol for accessing network services securely over unsecured networks. It includes the suite of utilities implementing it, such as:

- ssh-keygen: for creating new authentication key pairs for SSH;
- SCP (Secure Copy Protocol): for copying files between hosts on a network;
- **SFTP** (Secure File Transfer Protocol): for sending and receiving files. It's an SSHsecured version of FTP (File Transfer Protocol), and it has replaced FTP and FTPS (FTP Secure) as the preferred mechanism for file sharing over the Internet.

An SSH server, by default, listens for connections on the standard Transmission Control Protocol (TCP) <u>port 22</u>. Your applications may listen for SSH connections on other ports.

SSH lets you securely manage remote systems and applications, such as logging in to another computer over a network, executing commands, and moving files from one computer to another. An advanced SSH functionality is the creation of secure tunnels to run other application protocols remotely.

Basic SSH Commands

The following are fundamental SSH commands. Commit as many to memory as you can.

Command	Description
ssh	Connect to a remote server
ssh pi@raspberry	Connect to the device raspberry on the
	default SSH port 22 as user pi



ssh pi@raspberry -p 3344	Connect to the device raspberry on a specific port 3344 as user pi
ssh -i /path/file.pem admin@192.168.1.1	Connect to root@192.168.1.1 via the key file /path/file.pem as user admin
ssh root@192.168.2.2 'ls -l'	Execute remote command ls -1 on 192.168.2.2 as user root
<pre>\$ ssh user@192.168.3.3 bash < script.sh</pre>	Invoke the script script.sh in the current working directory spawning the SSH session to 192.168.3.3 as user user
<pre>ssh friend@Best.local "tar cvzf - ~/ffmpeg" > output.tgz</pre>	Compress the ~/ffmpeg directory and download it from a server Best.local as user friend
ssh-keygen	Generate SSH keys (follow the prompts)
ssh-keygen -F [ip/hostname]	Search for some IP address or hostname from ~/.ssh/known_hosts (logged-in host)
ssh-keygen -R [ip/hostname]	Remove some IP address or hostname from ~/.ssh/known_hosts (logged-in host)
ssh-keygen -f ~/.ssh/filename	Specify file name
<pre>ssh-keygen -y -f private.key > public.pub</pre>	Generate public key from private key
<pre>ssh-keygen -c -f ~/.ssh/id_rsa</pre>	Change the comment of the key file ~/.ssh/id_rsa
<pre>ssh-keygen -p -f ~/.ssh/id_rsa</pre>	Change passphrase of private key ~/.ssh/id_rsa
ssh-keygen -t rsa -b 4096 -C "my@email.com"	Generate an RSA 4096-bit key with "my@email.com" as a comment: -t: Type of key (rsa, ed25519, dsa,
	-b: The number of bits in the key
son	Copy files securely between servers
<pre>scp user@server:/folder/file.ext dest/</pre>	Copy from remote to local destination dest/
<pre>scp dest/file.ext user@server:/folder</pre>	Copy from local to remote
<pre>scp user1@server1:/file.ext user2@server2:/folder</pre>	Copy between two different servers
<pre>scp user@server:/folder/* .</pre>	Copies from a server folder to the current folder on the local machine
scp -r	Recursively copy entire directories
scp -r user@server:/folder dest/	Copy the entire folder to the local destination dest/
<pre>scp user@server:/folder/* dest/</pre>	Copy all files from a folder to the local destination dest/
scp -C	Option to compress data
scp -v	Option to print verbose info



scp -p	Option to preserve the last modification timestamps of the transferred files
scp -P 8080	Option to connect to remote host port 8080
scp -B	Option for <u>batch mode</u> and prevent you
	from entering passwords or passphrases
sftp	Securely transfer files between servers
sftp -p	Option to preserve the last modification timestamps of the transferred files
sftp -P 8080	Option to connect to remote host port 8080
sftp -r	Recursively copy entire directories when uploading and downloading. SFTP doesn't follow symbolic links encountered in the tree traversal.

SSH Configurations and Options

Have you ever wondered how SSH remembers your login credentials for various machines? This section is a brief reference on how to do so.

Command	Description
man ssh_config	Open OpenSSH SSH client configuration files. This manual lists all the OpenSSH parameters you can change.
<pre>cat /etc/ssh/ssh_config less</pre>	View your OpenSSH client system-wide configuration file
<pre>cat /etc/ssh/sshd_config less</pre>	View your OpenSSH server system-wide configuration file; the "d" stands for the server "daemon"
<pre>cat ~/.ssh/config less</pre>	View your SSH client user-specific configuration file
<pre>cat ~/.ssh/id_{type} less</pre>	View your SSH client private key; type is any of rsa, ed25519, dsa, ecdsa.
<pre>cat ~/.ssh/id_{type}.pub less</pre>	View your SSH client public key; type is any of rsa, ed25519, dsa, ecdsa.
cat ~/.ssh/known hosts less	View your SSH client logged-in hosts
<pre>cat ~/.ssh/authorized_keys less</pre>	View your SSH client authorized login keys
ssh-agent	Hold private SSH keys used for public key authentication (RSA, DSA, ECDSA, Ed25519)
ssh-agent -E fingerprint_hash	Specify the hash algorithm used when displaying key fingerprints.
	Valid fingerprint_hash options are sha256 (default) and md5.
ssh-agent -t lifetime	Set up a maximum lifetime for identities/private keys, overwritable by the same setting in ssh-add.
	Examples of lifetime:



	 600 = 600 seconds (10 minutes) 23m = 23 minutes 1h45 = 1 hour 45 minutes 			
ssh-add	Add SSH keys to the ssh-agent			
ssh-add -1	List your private keys cached by ssh-agent			
ssh-add -t lifetime	<pre>Set up a maximum lifetime for identities/private keys. Examples of lifetime: 600 = 600 seconds (10 minutes) 23m = 23 minutes 1h45 = 1 hour 45 minutes</pre>			
ssh-add -L	List the public key parameters of all saved identities			
ssh-add -D	Delete all cached private keys			
ssh-copy-id	Copy, install, and configure SSH keys on a remote server			
ssh-copy-id user@server	Copy SSH keys to a server as a user			
ssh-copy-id server1	Copy to some alias server server1 with the default login			
<pre>ssh-copy-id -i ~/.ssh/id_rsa.pub user@server</pre>	Copy a specific key to a server as a user			

Remote Server Management

The operating systems of SSH servers are mostly Unix/Linux, so once you've logged in to a server via SSH, the following commands are largely the same as their counterparts in Unix/Linux. Check out our <u>Unix commands cheat sheet</u> and <u>Linux command line cheat sheet</u> for other file management commands applicable to SSH.

Command	Description				
cd	Change the current working directory				
kill	Stop a running process				
ls	List files and directories				
mkdir	Create a new directory				
mv	Move files or directories				
nano	Edit a file in the terminal using Nano				
ps	List running processes				
pwd	Display the current working directory				
tail	View the last few (10, by default) lines of a file				
top	Monitor system resources and processes				
touch	Create a new file or update the timestamp of an existing file				
vim	Edit a file in the terminal using Vim				
exit	Close the SSH session				



😕 Windows PowerShell							
Windows PowerShell							
Copyright (C) Microsoft Corporation. All rights reserved.							
Thy the new energy platform Dewenthell https://aka.ms/nacerof							
Try the new cross-practorm powersherr https://aka.ms/pscore6							
PS_C:\Windows\system32>_ssh_lglab@Moth.local							
Password:							
Last login: Mon Apr 24 13:49:50 2023 from 192.168.65.160							
The default interactive shell is now zsh.							
To update your account to use zsh, please run chsh -s /bin/zsh .							
For more details, please visit nttps://support.apple.com/kb/H1208050.							
molli.~ Iglabs IS -I total 35136							
drwx 3 lolab staff 96 Aug 3 2021 Applications							
drwx@ 32]glab staff 1024 Mar 1 08:28 Desktop							
drwx@ 13 lglab staff 416 Jan 31 2022 Documents							
drwxr-x@ 55 lglab staff 1760 Apr 1 10:23 Downloads							
drwx@ 10]g]ab staff 320 Apr 1 10:23 Dropbox							
drwx@19]g]ab staff 608 Apr 1 10:05 Dropbox (Old (1))							
drwx@ 5 Iglab staff 160 Jan 15 08:32 Dropbox (Old)							
drwx@ 80 Iglab Statt 2560 Jan 15 01:59 Library							
drwx+ 15 lglab staff 160 Apr 16 2021 Music							
drwx+ 7 lolab staff 224 Sen 3 2021 Pictures							
drwxr-xr-x+ 5 lglab staff 160 Jan 13 2021 Public							
-rwxr-xr-x 1 lglab staff 80 Jan 5 2021 aqweb.sh							
-rwxr-xr-x@ 1 lglab staff 16091160 Jan 5 2021 cloud_sql_proxy							
drwxr-xr-x@ 36]g]ab staff 1152 Apr 16 2021 ffmpeg							
-rw-rr 1 lglab staff 1886796 Jan 5 2021 get-pip.py							
drwxr-xr-x 21 Iglab staft 6/2 Jan 5 2021 google-cloud-sdk							
-rwxr-xr-x I Iglab Stall 40 Jan II 2021 rJS.Sn drwyr-yr-y 3 Jalab staff 96 Jul 28 2021 websites							
Moth'~ lalah\$ vim							
Moth:~ lglab\$ exit							
logout							
Connection to moth.local closed.							
PS C:\Windows\system32>							

Using PowerShell to access a lab account on a network computer via SSH on Windows 10

Advanced SSH Commands

This table lists some complex SSH utilities that can help with network administration tasks: SSH File System (SSHFS), data compression, and X11 forwarding.

To conduct X11 forwarding over SSH, do these three things:

1. Set up your client (~/.ssh/config) to forward X11 by setting these parameters: Host *

```
ForwardAgent yes
```

- ForwardX11 yes
- 2. Set up your server (/etc/ssh/sshd_config) to allow X11 by setting these parameters:

```
X11Forwarding yes
X11DisplayOffset 10
X11UseLocalhost no
```

3. Set up X11 authentication on your server by installing ${\tt xauth}.$



Command	Description
sshfs	Mount a remote server's file system on a local directory.
	Remember to install this program onto your machine before use. Example installation commands:
	 sudo apt install sshfs # Ubuntu/Debian
	 sudo yum install fuse-sshfs # CentOS
	Learn to install apps on various Linux distributions here.
ssh -C hostname	Compress SSH traffic to improve
	performance on slow connections.
	Alternatively, insert Compression yes into your SSH configuration files.
ssh -o "Compression yes" -v hostname	An alternative method to compress SSH traffic to improve performance on slow connections.
	This is the same as inserting
	Compression yes into your SSH
	configuration files
ssh -X user@server	Enable X11 forwarding over SSH: forward graphical applications from a remote server as a user to a local machine.
ssh -o ForwardX11=yes	Enable X11 forwarding over SSH: forward
user@server	graphical applications from a remote
ssh -x	Server as a user to a local machine.
ssh -Y	Enable trusted X11 forwarding This option
	is riskier than ssh -X as it forwards the
	entire display of the SSH server to the
	client.

Tunneling

These SSH command line options create secure tunnels.

Options	Description	Syntax / Example
-L	Local port forwarding:	ssh user@ssh_server -
	forward a port on the local	L
	machine (SSH client) to a	local_port:destinatio
	port on the remote machine	n:remote_port
	(ssh_server as user), the	
	traffic of which goes to a	# Example



	<pre>port on the destination machine. The parameters local_port and</pre>	ssh root@192.168.0.1 -L 2222:10.0.1.5:3333
	remote_port can match.	
-J	ProxyJump; ensure that traffic passing through the intermediate/bastion hosts is always encrypted end-to- end.	<pre>ssh -J proxy_host1 remote_host2 ssh -J user@proxy_host1 user@remote_host2</pre>
	bastion hosts to connect to a remote host with a single command.	<pre># Multiple bastion hosts/jumps ssh -J user@proxy_host1:port 1,user@proxy_host2:po rt2_user@remote_host3</pre>
-R	Remote port forwarding: forward a port remote_port on the remote machine (ssh_server as user) to a port on the local machine (SSH client), the traffic of which goes to a port destination_port on the destination machine. An empty remote means the remote SSH server will bind on all interfaces. Additional SSH options in the example: -N: don't execute remote commands; useful for dedicated port forwarding -f: run SSH in the background	<pre>ssh -R [remote:]remote_port: destination:destinati on_port [user@]ssh_server # Example ssh -R 8080:192.168.3.8:3030 -N -f user@remote.host</pre>
-D	Set up a SOCKS Proxy to tunnel traffic from a remote_host on which you're the user to a local_port_number. Additional SSH options in the example: -q: quiet mode; don't output anything locally -C: compress data in the tunnel, save bandwidth	<pre>ssh -D local_port_number user@remote_host # Example ssh -D 6677 -q -C -N -f me@192.168.5.5</pre>



-N: don't execute remote	
commands; useful for	
dedicated port forwarding	
-f: run SSH in the	
background.	

SSH Tunneling Demonstration

Let's show you two ways to pipe traffic from your router into Wireshark and monitor your network activity. The first demonstration involves installing programs onto a system used as a router; the second, without.

Using Django



SSH Tunneling Demo





As a demonstration, we're piping traffic from a router into <u>Wireshark</u>, so that we can monitor live web traffic through an SSH tunnel. (The router below is a macOS computer hosting a Kali Linux virtual machine using the Wireshark instance installed on the latter.)

The setup is as follows:

- On the router: Enable remote access via SSH. (NOTE: On the macOS system, go to System Preferences > Sharing > turn on Remote Login and note the login username and hostname. For your router setup, check your specific manufacturer's guide to enable remote access via SSH.)
- 2. On the router: Install Python Django and start up the Django template server on http://127.0.0.1:8000 using the Terminal command string django-admin startproject mysite; cd mysite; python manage.py runserver (or python3 manage.py runserver). Note the Django web app uses port 8000.
- 3. On Kali Linux: execute this command to listen on port 8080: ls | nc -l -p 8080
- 4. On Kali Linux: execute this command in a different Terminal tab/window. Below, 8000 is the router's Django port, 8080 is the Kali Linux listening port on localhost, and the command involves remote port forwarding (-R): sudo ssh -R 8000:localhost:8080 user@router_ip
- 5. On Kali Linux: start Wireshark and select the loopback interface (10) as the capture device. Wireshark should be sniffing packets on 10 now.
- 6. On the router: visit <u>http://127.0.0.1:8000</u> in a web browser. (Note localhost and 127.0.0.1 are equivalent.) The Django server wouldn't load; it freezes instead because of the rerouted traffic.
- 7. On Kali Linux: You should expect the following results:



Piping traffic of Django web app <u>http://127.0.0.1:8000</u> on the macOS router into the Wireshark instance on Kali Linux



📉 🐸 💼 📄 🏅	~ <u>1</u> 234	<u>ه</u> ا			. <u> </u>		🖸 11:48 🖴 G-
		piped router 202304	424.pcapng				
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>G</u> o <u>C</u> ap	oture <u>A</u> nalyze <u>S</u> tatistics	Telephony <u>W</u> ireless <u>T</u> ools	<u>H</u> elp				
📶 🗖 🔊 🛓 🖿	ो 🔀 🙆 ५ + →	n + + 📑 🔳 🖬	3 8 8) e			
Apply a display filter <c< td=""><td>Ctrl-/></td><td></td><td></td><td></td><td></td><td></td><td>+ ר</td></c<>	Ctrl-/>						+ ר
No. Time	Source	Destination	Protocol	Length Info			
2 0.000014840	::1	::1	TCP	74 8080 → 39318	[RST, ACK]	Seq=1 Ack	k=1 Win=0 l
3 0.000426778	127.0.0.1	127.0.0.1	TCP	74 58870 → 8080	[SYN] Seq=	0 Win=6549	95 Len=0 MS
4 0.000607455	127.0.0.1	127.0.0.1	TCP	74 8080 → 58870	[SYN, ACK]	Seq=0 Ack	<=1 Win=654
5 0.000652643	127.0.0.1	127.0.0.1	TCP	66 58870 → 8080	[ACK] Seq=	1 Ack=1 Wi	in=65536 Le
6 0.000945285	127.0.0.1	127.0.0.1	HTTP	278 Continuation			
7 0.001069538	127.0.0.1	127.0.0.1	TCP	66 58870 → 8080	[ACK] Seq=	1 Ack=213	Win=65408
8 0.025430365	127.0.0.1	127.0.0.1	HIIP	422 GET / HTTP/1	.1	040 4-1-05	7.110-005.00
9 0.025463892	127.0.0.1	127.0.0.1	ТСР	66 8080 → 58870	[ACK] Seq-	213 ACK-3	57 WIN-6528
<pre>> Ethernet II, Src: 00 > Internet Protocol V > Transmission Contro: + Hypertext Transfer , GET / HTTP/1.1\r\ Host: localhost:8 Upgrade-Insecure- Accept: text/html User-Agent: Mozil Accept-Language: Connection: keep- \r\n [Full request URI [HTTP request 1/1]</pre>	0:00:00_00:00:00 (00 ersion 4, Src: 127.0 l Protocol, Src Port Protocol n 000\r\n Requests: 1\r\n ,application/xhtml+x la/5.0 (Macintosh; I en-gb\r\n gzip, deflate\r\n alive\r\n : http://localhost:8]	:00:00:00:00:00), Dst: .0.1, Dst: 127.0.0.1 : 58870, Dst Port: 8080 ml,application/xml;q=0 ntel Mac OS X 10_15_7) 0000/]	00:00:00 D, Seq: 1 .9,*/*;q= AppleWeb	0_00:00:00 (00:00:00 1, Ack: 213, Len: 35 -0.8\r\n Kit/605.1.15 (KHTML	:00:00:00) 6 ., like Gecl	ko) Versio	n/14.1.1 Safa
0000 01 05 16 11 00300 02 00 0ff 8c 00 00 00 00 00 01 95 6d 15 16 11 00 00 00 00 02 00 0ff 8c 00 00 00 00 02 04 0 02 04 0 02 04 0 02 04 0 02 04 0 02 04 0 02<	00 01 01 00 01 01 03 03 2f e 20 2f 20 48 54 </td <td>0 08 00 45 00 0 00 01 7f 00 e 43 2a 80 18 b b0 44 2f eb 0 2f 31 2e 31 ,GET</td> <td>@ T {NC / I / HTTP/:</td> <td>• E • • • • • • • • • • • • • • • • • •</td> <td></td> <td></td> <td></td>	0 08 00 45 00 0 00 01 7f 00 e 43 2a 80 18 b b0 44 2f eb 0 2f 31 2e 31 ,GET	@ T {NC / I / HTTP/:	• E • • • • • • • • • • • • • • • • • •			
🗢 📓 🛛 Ready to load or cap	oture			Packets: 9 · Displayed: 9 (100.0%) · Dro	pped: 0 (0.0%	b) Profile: Default

Wireshark HTTP packet corresponding to the Django web app on the router

Using tcpdump





The following is an alternative method for capturing remote web traffic passing through a router.

In Kali Linux, you'll log in to your router via SSH, capture packets with the command-line packet capturing tool <u>tcpdump</u>, and pipe the traffic into Wireshark.

Here is the required command with the option flags explained:

```
ssh [username]@[hostname/ip] tcpdump -U -s 65525 -w - 'not port 22'
| wireshark -k -i -
```

• -U: No buffering. Produce real-time output.



- -s 65525: Grab 65525 bytes of data from each packet rather than the default of 262144 bytes. 65525 is the <u>maximum transmission unit of a Point-to-Point Protocol</u> <u>packet</u> that Wireshark can handle. Adjust this number as you see fit.
- -w: Write each packet to the output packet capture file on your local disk in Kali
 Linux. Combining -U and -w means tcpdump writes to your output file as the packets pour in, rather than until the memory buffer fills up.
- 'not port 22': This is to prevent topdump from echoing the SSH packets sent between your machine and the router.
- <u>-k -i -</u>: Start the capture immediately and use the command before the pipe character (|) as the capture interface.

Tue 19:33 1 ,2 kali:~/Documents# ssh root@192.168.1.1 tcpdump -U -s 65535 -w - 'not port 22' hark -k -i -

Example of piping router traffic to Wireshark via tcpdump

After executing the command above, Wireshark opens:







Wireshark triggered

Next, the SSH client will prompt you to input your router password. Pasting it suffices:



SSH login successful. Now, tcpdump packet capture begins:





Meanwhile, Wireshark receives the piped traffic from tcpdump:

+	Applications • Place	Averieshark •			100 13.34	
					Capturing from Standard input [Wireshark 1.12.6 (Git Rev Unknown from unknown)]	9 G Ø
File E	dit View Go Capture Analyze Statis	tics Telephony Tools I	Internals Help			
•	9 🖉 📕 🧶 🛅 🗋 × ၐ ۹	(+ + 교 후 호		ର୍ଷ୍) 👾 🖂 🔀 🕿 🚺	
Filter:		- Expr	ession Clear A	Apply Save		
No.	Time Source	Destination	Protocol	Length Ir	fo	
7	71 19.52427500 c2:ff:d4:8c:e4:d7	Netgear_8c:e4:d7	0x886c	194 E	hernet II	
7	72 19.53426800 c2:ff:d4:8c:e4:d7	Netgear_8c:e4:d7	0x886c	194 E	hernet II	
7	73 19.68420100 c2:ff:d4:8c:e4:d7	Netgear_Bc:e4:d7	0x886c	194 E	hernet II	
7	74 19.70417500 c2:ff:d4:8c:e4:d7	Netgear_8c:e4:d7	0x886c	194 E	hernet II	
7	75 19.73721300 192.168.1.8	17.253.84.253	NTP	90 N	P Version 4, client	
7	76 19.75417600 c2:ff:d4:8c:e4:d7	Netgear_Bc:e4:d7	0x886c	194 E	hernet II	
7	77 19.76416500 c2:ff:d4:8c:e4:d7	Netgear_8c:e4:d7	0x886c	194 E	hernet II	
7	78 19.96102600 17.253.84.253	192.168.1.8	NTP	90 N	P Version 4, server	
7	79 20.37382800 Netgear_Bc:e4:da	Apple_3c:ce:49	ARP	42 W	o has 192.168.1.8? Tell 192 <mark>.168.1</mark> .1	
5	30 20.37404800 Apple_3c:ce:49	Netgear_8c:e4:da	ARP	60 1	2.168.1.8 is at ac:87:a3:3c:ce:49	
6	81 24.19496800 192.168.1.8	54.175.223.74	TCP	66 [CP Retransmission] 57903-https [FIN, AOK] Seq=1 Ack=1 Win=4096 Len=0 TSval=1224346042 TSecr=	:440985131
8	32 25.02309400 Netgear_Bc:e4:da	Broadcast	ARP	42 W	o has 192.168.1.157 Tell 192.168.1.1	
8	33 25.04240000 Apple_41:tc:bc	Netgear_8c:e4:da	ARP	42 1	2.168.1.15 is at 9c:+3:87:41:+c:bc	
5	34 25.04244500 50.22.225.78	192.168.1.15	102	103 [CP segment of a reassembled POUL	
8	35 25.22190100 192.168.1.15	50.22.225.78	TOP	66 5	231-xmpp-ctient [Ack] sedel Ackess Winetise Leneo ISVales12320395 (SecHe1525/38196	
8	35 25.74257000 192.168.1.15	50.22.225.78	TOP	93 (CP segment of a reassembled PUUI	
	5/ 23.6312/100 30.22.223.76	192.100.1.15	10*	00 1	bb-crience25231 [MCV] 26d-26 MCV-56 MIII-214 Fei-0 12M8(-1252)/20052 126CL-215250315	
r Home Li Ha ujtao vilitite (advilla) za ujtao tapitate (advilla) vilitititite (di tita) (di tita) vilitititi (di tita) (di ti						
• Edement at, and requer_decenter(eder), but an address (fifitititit)						
· Auui	less liesocacton Procococ (reduesc)					
0000	ff ff ff ff ff ff c0 ff d4 8c e4	da 08 06 00 01				
0010	08 00 05 04 00 01 c0 ff d4 8c e4	da c0 a8 01 01				
0020	00 00 00 00 00 00 00 31 51		····· .Q			
0.**	Standard input: diveranture in progra-	erkets: 87 - Displayed: 87 (10)	0.030			THE CYBER SECURITY COMPANY
	standard nyon are captorent progress Pre-	erection on purpayed on troo	o lovely			

That's it.

Conclusion

We have covered SSH, SCP, SFTP, SSH configuration commands such as *ssh-agent*, *ssh-add*, and *ssh-copy-id*, and various SSH tunneling commands.

Here are some tips for using SSH more efficiently and securely:



- Disable X11 and TCP forwarding because attackers can use such weaknesses to access other systems on your network. Change the options on sshd_config to be AllowTcpForwarding no and X11Forwarding no.
- Change the default options on sshd_config, such as <u>changing the default port</u> from 22 to another number.
- Authenticate clients using SSH certificates created with ssh-keygen.
- Use a bastion host with the help of <u>tunneling</u> commands.
- Restrict SSH logins to specific IPs, such as adding user filtering with the AllowUsers option in sshd_config.

Thanks to its security measures and the ubiquity of networking tasks, SSH is indispensable for computer data communications. Hence every student and professional in IT and cyber security needs a working knowledge of SSH commands, and we hope this SSH cheat sheet is a good starter or refresher for you.

To learn more about SSH and secure network administration, check out the following courses from us:

- The Complete Cyber Security Course! Volume 3: Anonymous Browsing
 - <u>https://courses.stationx.net/p/the-complete-cyber-security-course-anonymous-browsing</u>
- Secure Shell Fundamentals Learn SSH By Configuring It
 - <u>https://courses.stationx.net/p/secure-shell-fundamentals-learn-ssh-by-</u> <u>configuring-it</u>
- Linux Security and Hardening, The Practical Security Guide
 - <u>https://courses.stationx.net/p/linux-security-and-hardening-the-practical-security-guide</u>

