Linux Commands for Professional Environment

Network Troubleshooting Commands

Practical Examples

Copyright Notice: Protection of Intellectual Property

DigiTalk

Reach us at digitalk.fmw@gmail.com
DigiTalk Channel: https://www.youtube.com/channel/UCCGTnI9vvF_ETMhGUXGdFWw
Playlists: https://www.youtube.com/@digitalk.middleware/playlists
Weblogic Server Architecture: https://youtu.be/gNqeIfLjUqw

DigiTalk Udemy Courses and Coupon Code

SOA Suite Administration
https://www.udemy.com/course/mastering-oracle-soa-suite-12c-administration/?couponCode=739A60915F86847014EB
Coupon Code: 739A60915F86847014EB

JBoss 8 Administration
https://www.udemy.com/course/mastering-jboss-eap-8-administration-from-intro-to-advanced/?couponCode=BF65EB008CFE16686BD2
Coupon Code:BF65EB008CFE16686BD2

OHS Administration
https://www.udemy.com/course/mastering-oracle-ohs-http-12c-web-server-administration/?couponCode=8E990556B21AF3E1A316
Coupon Code: 8E990556B21AF3E1A316

Weblogic Server Administration
https://www.udemy.com/course/oracle-weblogic-server-12c-and-14c-administration/?couponCode=87BC1314AC7690FD5294
Coupon Code:87BC1314AC7690FD5294

You can write us on digitalk.fmw@gmail.com if coupon code expired.

Ping

The ping command is a fundamental network utility used to test the reachability of a host on an IP network and to measure the round-trip time for messages sent from the originating host to a destination computer. Here's a comprehensive overview of ping with important options and practical examples.

Basic Syntax

```
ping [options] destination
```

Important Options

- -c count: Specifies the number of echo requests to send.
- -i interval: Sets the interval between sending each packet (in seconds).
- -t ttl: Sets the Time to Live for packets.
- -s packetsize: Specifies the number of data bytes to be sent.
- -w deadline: Sets a deadline for how long to run the command (in seconds).
- -q: Quiet output. Displays only the summary.
- -f: Flood ping. Sends packets as fast as possible.
- -l preload: Sends a specified number of packets before starting the normal ping process.
- -v: Verbose output.
- -D: Prints timestamp before each line.
- -I interface: Specifies the network interface to be used.

Practical Examples

Example 1: Basic Ping

```
ping google.com
```

This sends ICMP echo requests to google.com until you stop it with Ctrl+C.

Example 2: Ping with a Specified Number of Packets

```
ping -c 4 google.com
```

This sends exactly 4 ICMP echo requests to google.com.

Example 3: Ping with a Specific Interval

```
ping -i 2 google.com
```

This sends ICMP echo requests to google.com every 2 seconds.

Example 4: Ping with a Time to Live (TTL) Value

```
ping -t 10 google.com
```

This sends ICMP echo requests with a TTL of 10.

Example 5: Ping with a Specified Packet Size

```
ping -s 100 google.com
```

This sends ICMP echo requests with 100 bytes of data.

Example 6: Ping with a Deadline

```
ping -w 10 google.com
```

This sends ICMP echo requests to google.com for 10 seconds.

Example 7: Quiet Output

```
ping -c 4 -q google.com
```

This sends 4 ICMP echo requests to google.com and displays only the summary.

Example 8: Flood Ping (Use with Caution)

```
ping -f google.com
```

This sends packets as fast as possible. Requires root privileges and can generate a significant amount of traffic.

Example 9: Preload Packets

```
ping -l 10 google.com
```

This sends 10 ICMP echo requests immediately before starting the normal ping process.

Example 10: Verbose Output

```
ping -v google.com
```

This displays more detailed information about the ping process.

Example 11: Ping with Timestamp

```
ping -D google.com
```

This prints a timestamp before each line of output.

Example 12: Specify Network Interface

```
ping -I eth0 google.com
```

This sends ICMP echo requests using the eth0 network interface.

traceroute

The traceroute command is a network diagnostic tool used to trace the path that packets take from one network host to another. It helps identify routing issues and network bottlenecks by displaying each hop along the route and measuring the time taken for each hop.

Basic Syntax

traceroute [options] destination

Important Options

- -n: Do not resolve hostnames; display IP addresses only.
- -m max_ttl: Set the maximum number of hops (TTL).
- -q nqueries: Set the number of probe packets per hop.
- -w waittime: Set the time to wait for a response (in seconds).
- -I: Use ICMP ECHO instead of UDP datagrams.
- -T: Use TCP SYN for probing.
- -p port: Set the base UDP port number for probe packets.
- -f first_ttl: Set the initial TTL (time to live).
- -s source_addr: Set the source address to use.
- -4: Use IPv4 only.
- -6: Use IPv6 only.

Practical Examples

Example 1: Basic Traceroute

```
traceroute google.com
```

This traces the route packets take to google.com.

Example 2: Traceroute with IP Addresses Only

```
traceroute -n google.com
```

This traces the route and displays IP addresses without resolving hostnames.

Example 3: Set Maximum TTL

```
traceroute -m 15 google.com
```

This limits the traceroute to a maximum of 15 hops.

Example 4: Set Number of Probe Packets per Hop

```
traceroute -q 5 google.com
```

This sends 5 probe packets per hop.

Example 5: Set Wait Time for Responses

```
traceroute -w 3 google.com
```

This sets the wait time to 3 seconds for each response.

Example 6: Use ICMP ECHO Instead of UDP

```
traceroute -I google.com
```

This uses ICMP ECHO requests instead of the default UDP packets.

Example 7: Use TCP SYN for Probing

```
traceroute -T google.com
```

This uses TCP SYN packets for probing.

Example 8: Set Base UDP Port Number

```
traceroute -p 33434 google.com
```

This sets the base UDP port number to 33434 (default).

Example 9: Set Initial TTL

```
traceroute -f 5 google.com
```

This starts the traceroute with an initial TTL of 5.

Example 10: Specify Source Address

```
traceroute -s 192.168.1.1 google.com
```

This sets the source address to 192.168.1.1.

Example 11: Use IPv4 Only

```
traceroute -4 google.com
```

This forces the use of IPv4.

Example 12: Use IPv6 Only

```
traceroute -6 google.com
```

This forces the use of IPv6.

Practical Use Cases

Example 1: Diagnose Network Latency

```
traceroute google.com
```

Identify the hops with high latency to diagnose where delays are occurring.

Example 2: Detect Routing Loops

```
traceroute example.com
```

If the output shows the same IP addresses repeatedly, there might be a routing loop.

Example 3: Check Network Path Differences

Run traceroute from different locations or ISPs to compare network paths:

```
traceroute -I google.com
```

Compare results from different networks to see variations in routing.

nslookup

The nslookup command is a network administration tool used for querying the Domain Name System (DNS) to obtain domain name or IP address mapping, or other DNS records. It's commonly used to diagnose DNS-related issues.

Basic Syntax

nslookup [options] [hostname] [DNS server]

Important Options

- -querytype=type: Specifies the type of DNS record to be queried (e.g., A, AAAA, CNAME, MX, NS, PTR, SOA, SRV, TXT).
- -timeout=seconds: Specifies the time to wait for a response.
- -retries=number: Sets the number of retries before giving up.
- -port=port: Specifies the port number to use for the DNS query.
- -vc: Uses a virtual circuit (TCP instead of UDP).
- -debug: Prints debugging information.
- -type=type: Alias for -querytype.
- -server: Specifies the DNS server to use.
- -sil[ent]: Suppresses the printing of the initial message.
- -stats: Prints statistics of the query.
- -class=class: Specifies the class of the DNS record (IN, CH, HS).

Practical Examples

Example 1: Basic DNS Lookup

```
nslookup google.com
```

This queries the default DNS server to find the IP address for google.com.

Example 2: Query a Specific DNS Record Type

```
nslookup -querytype=MX google.com
```

This queries the mail exchange (MX) records for google.com.

Example 3: Specify a DNS Server

```
nslookup google.com 8.8.8.8
```

This queries the DNS server at 8.8.8.8 (Google Public DNS) for google.com.

Example 4: Debugging Information

```
nslookup -debug google.com
```

This provides detailed debugging information for the DNS query.

Example 5: Set Timeout

```
nslookup -timeout=10 google.com
```

This sets the query timeout to 10 seconds.

Example 6: Set Number of Retries

```
nslookup -retries=3 google.com
```

This sets the number of retries to 3.

Example 7: Query Using TCP

```
nslookup -vc google.com
```

This queries using a virtual circuit (TCP) instead of the default UDP.

Example 8: Query PTR Record

```
nslookup -querytype=PTR 8.8.8.8
```

This performs a reverse lookup to find the domain name associated with the IP address 8.8.8.8.

Interactive Mode

You can start nslookup in interactive mode by running the command without arguments. In this mode, you can execute multiple queries without exiting nslookup.

Example:

```
nslookup
```

```
> server 8.8.8.8
> google.com
> set type=MX
> google.com
> exit
```

Practical Use Cases

Example 1: Verify DNS Configuration

To verify that your DNS server is correctly resolving domain names, use:

```
nslookup google.com 8.8.8.8
```

Example 2: Check Mail Server Configuration

To check the mail servers for a domain:

```
nslookup -querytype=MX example.com
```

Example 3: Diagnose DNS Issues

If a website is not resolving correctly, you can use:

```
nslookup example.com
```

To see what IP address the default DNS server is returning.

ifconfig/ip

ifconfig Command

The ifconfig command is used to configure network interfaces. It’s part of the net-tools package and, while still available, it’s considered deprecated in favor of the ip command.

Syntax

ifconfig [interface] [options]

Commonly Used Options

- -a: Display all interfaces, both active and inactive.
- up: Activate the specified interface.
- down: Deactivate the specified interface.
- inet: Assign an IP address to the specified interface.
- netmask: Set the network mask for the specified interface.

- broadcast: Set the broadcast address for the specified interface.
- mtu: Set the Maximum Transmission Unit size for the specified interface.
- promisc: Enable promiscuous mode.

Examples

Display All Interfaces

```
ifconfig -a
```

This command shows the status of all network interfaces.

Display Specific Interface

```
ifconfig eth0
```

This command shows the configuration of the eth0 interface.

Assign IP Address to Interface

```
sudo ifconfig eth0 192.168.1.100 netmask 255.255.255.0
```

This command assigns the IP address 192.168.1.100 with the specified netmask to the eth0 interface.

Bring Interface Up

```
sudo ifconfig eth0 up
```

This command activates the eth0 interface.

Bring Interface Down

```
sudo ifconfig eth0 down
```

This command deactivates the eth0 interface.

Enable Promiscuous Mode

```
sudo ifconfig eth0 promisc
```

This command enables promiscuous mode on the eth0 interface, allowing it to capture all packets on the network.

Useful Commands in Combination with Other Commands

Display Interface Statistics

```
ifconfig eth0 | grep 'RX\|TX'
```

This command filters the output to show only the RX (receive) and TX (transmit) statistics for eth0.

ip Command

The ip command is a more modern and powerful tool for network interface management and configuration. It's part of the iproute2 package.

Syntax

ip [OPTIONS] OBJECT COMMAND

- link: Manage network interfaces.
- addr: Manage IP addresses.
- route: Manage routing table.
- neigh: Manage ARP or NDISC cache entries.
- rule: Manage routing rules.
- Commonly Used Options
- -s: Output more detailed information.
- -4: Display only IPv4 addresses.
- -6: Display only IPv6 addresses.
- show: Display information.

Examples

Display All Interfaces

```
ip link show
```

This command shows the status of all network interfaces.

Display IP Addresses

```
ip addr show
```

This command shows IP addresses assigned to all network interfaces.

Display Specific Interface

```
ip addr show dev eth0
```

This command shows IP addresses assigned to the eth0 interface.

Assign IP Address to Interface

```
sudo ip addr add 192.168.1.100/24 dev eth0
```

This command assigns the IP address 192.168.1.100 with a prefix length of 24 to the eth0 interface.

Bring Interface Up

```
sudo ip link set eth0 up
```

This command activates the eth0 interface.

Bring Interface Down

```
sudo ip link set eth0 down
```

This command deactivates the eth0 interface.

Add Default Gateway

```
sudo ip route add default via 192.168.1.1
```

This command adds a default gateway with IP address 192.168.1.1.

Useful Commands in Combination with Other Commands

Display Routing Table

```
ip route show
```

This command shows the routing table.

Display Interface Statistics

```
ip -s link show eth0
```

This command shows detailed statistics for the eth0 interface.

Monitor Network Changes

```
ip monitor all
```

This command continuously monitors changes in IP addresses, routes, and devices.

Check Neighbour (ARP) Table

```
ip neigh show
```

This command displays the ARP table.

netstat

The netstat command is a network utility that provides various statistics and information about network connections, routing tables, interface statistics, masquerade connections, and multicast memberships. It is commonly used for troubleshooting and monitoring network status.

Syntax

netstat [options]

Commonly Used Options

- -a: Show all sockets (both listening and non-listening).
- -t: Display TCP connections.
- -u: Display UDP connections.
- -n: Show numerical addresses instead of resolving hostnames.
- -l: Show only listening sockets.
- -p: Show the PID and name of the program to which each socket belongs (requires root).
- -r: Display the kernel routing table.
- -i: Show network interfaces.
- -s: Display summary statistics for each protocol.
- -c: Print the selected information every second continuously.

Examples

Display All Connections

```
netstat -a
```

This command shows all active connections and listening ports, both TCP and UDP.

Display TCP Connections

```
netstat -t
```

This command shows only active TCP connections.

Display UDP Connections

```
netstat -u
```

This command shows only active UDP connections.

Show Numerical Addresses

```
netstat -n
```

This command displays numerical addresses instead of resolving hostnames, useful for faster output and avoiding DNS lookups.

Show Listening Sockets

```
netstat -l
```

This command shows only the listening sockets (both TCP and UDP).

Show Program Names and PIDs

```
sudo netstat -p
```

This command displays the PID and name of the program to which each socket belongs. Note that this requires root privileges.

Display Kernel Routing Table

```
netstat -r
```

This command shows the kernel routing table, which is useful for understanding the routing decisions made by the system.

Show Network Interfaces

```
netstat -i
```

This command displays a table of network interfaces and their statistics.

Show Summary Statistics

```
netstat -s
```

This command provides a summary of statistics for each network protocol, including TCP, UDP, ICMP, and IP.

Continuous Output

```
netstat -c
```

This command continuously prints the selected information every second.

Combining Options

Display All Listening TCP Connections in Numerical Format

```
netstat -ltun
```

This command combines several options to show only listening TCP and UDP connections in numerical format.

Display Routing Table and Interface Statistics

```
netstat -ri
```

This command shows the routing table and interface statistics together.

Practical Example

Example1: Suppose you want to find out which process is using a specific port (e.g., port 8080). You could use the following command:

```
sudo netstat -tulpn | grep :8080
```

This command shows TCP and UDP listening ports, along with the PID and program name, and filters the results to show only those that include port 8080.

Example 2: Display Established TCP Connections with Process Information

```
sudo netstat -tnp | grep ESTABLISHED
```

This command displays all established TCP connections along with the PID and name of the process using those connections.

Example 3: Show Listening Ports with Human-Readable Process Information

```
sudo netstat -tulpn | less
```

This command shows all listening TCP and UDP ports with the corresponding PID and process name, and pipes the output through less for easy scrolling.

Example 4: Monitor Network Connections in Real-Time

```
watch -n 1 'netstat -an | grep ESTABLISHED'
```

This command uses watch to execute the netstat command every second, showing real-time updates of all established connections.

Example 5: Check Active Internet Connections and Routing Tables Together

```
netstat -atunr
```

This command displays active TCP and UDP connections (-a, -t, -u) along with the routing table (-r).

Example 6: Find the Number of Connections per IP Address

```
netstat -an | grep ESTABLISHED | awk '{print $5}' | cut -d: -f1 | sort | uniq -c | sort -nr
```

This command counts the number of established connections per IP address. It extracts the remote IP address from each connection, sorts them, counts the unique occurrences, and sorts the results numerically in descending order.

Example 7: Display Network Interface Statistics

```
netstat -i | column -t
```

This command shows the network interface statistics in a neatly formatted table using column -t.

Example 8: Continuous Monitoring of Network Statistics

```
watch -d 'netstat -i'
```

This command uses watch to display network interface statistics continuously, highlighting changes between updates (-d).

Example 9: Check Specific Port Usage

```
sudo netstat -tulpn | grep :80
```

This command displays information about processes using port 80, which is commonly used by HTTP servers.

Example 10: Combining netstat with grep for Specific Protocols

```
netstat -s | grep -A 5 'Tcp:'
```

This command shows TCP statistics by filtering the netstat -s output for lines starting with "Tcp:" and the following 5 lines.

Example 11: Display All Connections with Hostnames and Filter by State

```
netstat -at | grep TIME_WAIT
```

This command displays all TCP connections with hostnames and filters the output to show connections in the TIME_WAIT state.

Example 12: Check Listening Ports and Associated Processes

```
sudo netstat -plnt | awk '{print $1,$4,$7}' | column -t
```

This command displays only the protocol, local address, and PID/program name for listening ports, formatted into a table for easier reading.

SS

The ss (Socket Statics) command is a utility to investigate sockets and display detailed network statistics. It is often considered a faster and more modern replacement for netstat. Here's an explanation of the ss command with its options and useful combinations:

Syntax

ss [options]

Commonly Used Options

- -a: Show all sockets (both listening and non-listening).
- -t: Display TCP sockets.
- -u: Display UDP sockets.
- -l: Show only listening sockets.
- -p: Show the process using the socket.
- -n: Do not resolve service names or host names (show numerical addresses).
- -r: Resolve IP addresses to hostnames.
- -k: Display kernel memory used by sockets.
- -m: Display memory usage for socket buffers.
- -s: Display summary statistics.
- -4: Display only IPv4 sockets.
- -6: Display only IPv6 sockets.

- -o: Show timer information.

Examples

Display All Sockets

```
ss -a
```

This command shows all sockets (both listening and non-listening).

Display TCP Sockets

```
ss -t
```

This command shows only TCP sockets.

Display UDP Sockets

```
ss -u
```

This command shows only UDP sockets.

Show Listening Sockets

```
ss -l
```

This command shows only listening sockets.

Show Sockets with Process Information

```
sudo ss -p
```

This command displays the processes using the sockets. Note that it requires root privileges.

Show Numerical Addresses

```
ss -n
```

This command displays numerical addresses instead of resolving hostnames.

Show Kernel Memory Usage

```
ss -k
```

This command displays the kernel memory used by each socket.

Show Memory Usage for Socket Buffers

```
ss -m
```

This command shows the memory usage for socket buffers.

Display Summary Statistics

```
ss -s
```

This command provides a summary of socket statistics.

Display IPv4 Sockets Only

```
ss -4
```

This command shows only IPv4 sockets.

Display IPv6 Sockets Only

```
ss -6
```

This command shows only IPv6 sockets.

Show Timer Information

```
ss -o
```

This command shows timer information for each socket.

Useful Commands in Combination with Other Commands

Display Established TCP Connections with Process Information

```
sudo ss -tnp | grep ESTAB
```

This command shows all established TCP connections along with the PID and name of the process using those connections.

Show Listening Ports with Human-Readable Process Information

```
sudo ss -tulpn | less
```

This command shows all listening TCP and UDP ports with the corresponding PID and process name, and pipes the output through less for easy scrolling.

Monitor Network Connections in Real-Time

```
watch -n 1 'ss -an | grep ESTAB'
```

This command uses watch to execute the ss command every second, showing real-time updates of all established connections.

Check Active Internet Connections and Routing Tables Together

```
ss -atunr
```

This command displays active TCP and UDP connections (-a, -t, -u) along with the routing table (-r).

Find the Number of Connections per IP Address

```
ss -tan | grep ESTAB | awk '{print $5}' | cut -d: -f1 | sort | uniq -c | sort -nr
```

This command counts the number of established connections per IP address. It extracts the remote IP address from each connection, sorts them, counts the unique occurrences, and sorts the results numerically in descending order.

Display Network Interface Statistics

```
ss -i | column -t
```

This command shows the network interface statistics in a neatly formatted table using column -t.

Continuous Monitoring of Network Statistics

```
watch -d 'ss -i'
```

This command uses watch to display network interface statistics continuously, highlighting changes between updates (-d).

Check Specific Port Usage

```
sudo ss -tulpn | grep :80
```

This command displays information about processes using port 80, which is commonly used by HTTP servers.

Combining ss with grep for Specific Protocols

```
ss -s | grep -A 5 'Tcp:'
```

This command shows TCP statistics by filtering the ss -s output for lines starting with "Tcp:" and the following 5 lines.

Display All Connections with Hostnames and Filter by State

```
ss -at | grep TIME-WAIT
```

This command displays all TCP connections with hostnames and filters the output to show connections in the TIME-WAIT state.

Check Listening Ports and Associated Processes

```
sudo ss -plnt | awk '{print $1,$4,$6}' | column -t
```

This command displays only the protocol, local address, and PID/program name for listening ports, formatted into a table for easier reading.

tcpdump

To check available Ethernet interfaces and then capture traffic on a specific Ethernet interface using the tcpdump command, you can follow these steps:

Step 1: List All Network Interfaces

First, list all available network interfaces on your system. You can use the ifconfig or ip command to do this:

Using ifconfig:

```
ifconfig -a
```

This displays all network interfaces along with their details.

Using ip:

```
ip link show
```

This lists all network interfaces by name.

Step 2: Identify the Ethernet Interface

From the output of ifconfig -a or ip link show, identify the name of the Ethernet interface you want to capture traffic on. Common names for Ethernet interfaces include eth0, eth1, enp0s3, etc.

Step 3: Capture Traffic on the Ethernet Interface

Use tcpdump to capture traffic on the identified Ethernet interface. You'll need superuser privileges to run tcpdump, so use sudo.

Basic Capture:

```
sudo tcpdump -i eth0
```

Replace eth0 with the name of your Ethernet interface.

Capture and Display Packet Contents:

```
sudo tcpdump -i eth0 -vvv -X
```

This captures traffic on eth0, with -vvv for very verbose output and -X to display the packet contents in both hex and ASCII.

Save Captured Packets to a File:

```
sudo tcpdump -i eth0 -w capture.pcap
```

This saves the captured packets to a file named capture.pcap for later analysis.

Practical Examples

Example 1: Capture All Traffic on eth0

```
sudo tcpdump -i eth0
```

This captures all traffic on the eth0 interface and prints it to the screen.

Example 2: Capture Traffic on eth0 with Specific Filter

Capture only TCP traffic:

```
sudo tcpdump -i eth0 tcp
```

Capture traffic on port 80 (HTTP):

```
sudo tcpdump -i eth0 port 80
```

Example 3: Capture and Save to a File for Analysis

```
sudo tcpdump -i eth0 -w /path/to/save/capture.pcap
```

This captures all traffic on eth0 and saves it to /path/to/save/capture.pcap.

Example 4: Read Captured Packets from a File

To read and analyze a previously saved capture file:

```
sudo tcpdump -r /path/to/save/capture.pcap
```

Example 5: Capture with a Time Limit

Capture traffic for only 60 seconds:

```
sudo timeout 60 tcpdump -i eth0 -w capture.pcap
```

Filtering Traffic with tcpdump

You can use tcpdump filters to capture specific types of traffic. Here are some examples:

Capture HTTP traffic (port 80):

```
sudo tcpdump -i eth0 port 80
```

Capture HTTPS traffic (port 443):

```
sudo tcpdump -i eth0 port 443
```

Capture traffic from a specific IP address:

```
sudo tcpdump -i eth0 host 192.168.1.10
```

Capture traffic to a specific IP address:

```
sudo tcpdump -i eth0 dst 192.168.1.10
```

Capture traffic from and to a specific IP address:

```
sudo tcpdump -i eth0 host 192.168.1.10
```

Capture only TCP traffic:

```
sudo tcpdump -i eth0 tcp
```

Capture only UDP traffic:

```
sudo tcpdump -i eth0 udp
```

CURL Command (Checking the connectivity of a remote network host and port)

The curl command is a versatile tool used for transferring data from or to a server, using various protocols such as HTTP, HTTPS, FTP, and more. It is also useful for checking the connectivity of a remote network host and port.

Here's a detailed explanation of how to use curl to check connectivity using different protocols:

Basic Syntax

curl [options] [URL]

Checking HTTP/HTTPS Connectivity

To check if a remote host is reachable over HTTP or HTTPS, you can simply use:

HTTP:

```
curl http://example.com
```

This fetches the content of http://example.com and prints it to the screen.

HTTPS:

```
curl https://example.com
```

This fetches the content of https://example.com and prints it to the screen.

Checking Specific Port Connectivity

To check connectivity to a specific port, you can use the --connect-timeout option to set a timeout for the connection. This is useful for determining if a specific service is reachable.

Example:

```
curl --connect-timeout 5 http://example.com:8080
```

This tries to connect to http://example.com on port 8080 and times out if it takes longer than 5 seconds.

Checking FTP Connectivity

To check FTP connectivity, you can use the FTP protocol:

Example:

```
curl ftp://example.com
```

This tries to connect to an FTP server at example.com.

Checking SMTP Connectivity

While curl isn't primarily designed for SMTP, you can use it to check SMTP server connectivity by specifying the smtp protocol:

Example:

```
curl --url "smtp://mail.example.com:25"
```

This attempts to connect to an SMTP server at mail.example.com on port 25.

Checking Other Protocols

LDAP:

```
curl ldap://example.com
```

This tries to connect to an LDAP server at example.com.

IMAP:

```
curl imap://mail.example.com
```

This tries to connect to an IMAP server at mail.example.com.

Using curl with Verbose Output

To get more detailed information about the connection process, you can use the -v (verbose) option:

Example:

```
curl -v http://example.com
```

This prints detailed information about the request and response, including headers.

Practical Examples

Example 1: Check HTTP Connectivity

```
curl http://example.com
```

This fetches the webpage and prints it to the console.

Example 2: Check HTTPS Connectivity with Verbose Output

```
curl -v https://example.com
```

This fetches the webpage over HTTPS and provides detailed output.

Example 3: Check Connectivity to a Specific Port

```
curl --connect-timeout 5 http://example.com:8080
```

This checks if example.com is reachable on port 8080 within 5 seconds.

Example 4: Check FTP Connectivity

```
curl ftp://example.com
```

This checks if example.com is reachable via FTP.

Example 5: Check SMTP Connectivity

```
curl --url "smtp://mail.example.com:25"
```

This checks if mail.example.com is reachable on port 25 using SMTP.

Example 6: Check LDAP Connectivity

```
curl ldap://example.com
```

This checks if example.com is reachable via LDAP.

Example 7: Check IMAP Connectivity

```
curl imap://mail.example.com
```

This checks if mail.example.com is reachable via IMAP.

DISCLAIMER AND CONSENT

This document is being provided by DigiTalk as part of its effort to assist users in understanding and working with Linux. While every effort has been made to ensure the accuracy and reliability of the information presented in this document, there is a possibility of typographical errors or inaccuracies. DigiTalk does not guarantee the correctness or completeness of the content provided in this document.

Users of this document are encouraged to cross-reference the information presented here with official documentation available on their website or other authoritative sources. Any discrepancies or inaccuracies found in this document should be reported to us at digitalk.fmw@gmail.com.

By using this document, you acknowledge and consent to the following:

This document is not officially endorsed or verified by any other third party organization..

The Company makes no claims or guarantees about the accuracy or suitability of the information contained in this document.

Users are responsible for verifying and validating any information presented here for their specific use case.

DigiTalk disclaims any liability for any errors, omissions, or damages that may result from the use of this document.

If you discover any inaccuracies or errors in this document, please report them to digitalk.fmw@gmail.com, and the Company will endeavor to correct them as necessary.

This consent statement is provided to ensure transparency and understanding of the limitations of the information contained in this document. By using this document, you agree to abide by the terms and conditions outlined herein.