

1) See screenshot.

```
ryan@localhost:/etc/httpd/conf — sudo nano httpd.conf
GNU nano 5.6.1 httpd.conf
# Do not add a slash at the end of the directory path.  If you point
# ServerRoot at a non-local disk, be sure to specify a local disk on the
# Mutex directive, if file-based mutexes are used.  If you wish to share the
# same ServerRoot for multiple httpd daemons, you will need to change at
# least PidFile.
#
ServerRoot "/etc/httpd"
#
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, instead of the default.  See also the <VirtualHost>
# directive.
#
# Change this to Listen on a specific IP address, but note that if
# httpd.service is enabled to run at boot time, the address may not be
# available when the service starts.  See the httpd.service(8) man
# page for more information.
#
#Listen 12.34.56.78:80
Listen 0.0.0.0:8080
```

2) See screenshots.

```
11 29.119169973 10.0.1.107 10.0.1.185 TCP 68 8080 - 43978 [ACK] Seq=1 Ack=423 Win=64768 Len=0 TSval=1056646987 TSecr=1196745602
19 29.119343911 10.0.1.185 10.0.1.107 TCP 68 8080 - 43978 [ACK] Seq=1 Ack=423 Win=64768 Len=0 TSval=1056646987 TSecr=1196745601
20 29.119882137 10.0.1.185 10.0.1.107 HTTP 534 HTTP/1.1 200 OK (text/html)
21 29.119892238 10.0.1.107 10.0.1.185 TCP 68 43978 - 8080 [ACK] Seq=423 Ack=467 Win=64128 Len=0 TSval=1196745602 TSecr=1056646987
22 34.117396850 10.0.1.185 10.0.1.107 TCP 68 8080 - 43978 [FIN,ACK] Seq=407 Ack=423 Win=64768 Len=0 TSval=1056651988 TSecr=1196745602

Frame 18: 490 bytes on wire (3920 bits), 490 bytes captured (3920 bits) on interface any, id 0
Linux cooked capture v1
Internet Protocol Version 4, Src: 10.0.1.107, Dst: 10.0.1.185
Transmission Control Protocol, Src Port: 43978, Dst Port: 8080, Seq: 1, Ack: 1, Len: 422
Hypertext Transfer Protocol
  GET /GetExample.php?firstNum=12&secondNum=7 HTTP/1.1\r\n
    [Expert Info (Chat/Sequence): GET /GetExample.php?firstNum=12&secondNum=7 HTTP/1.1\r\n]
    [GET /GetExample.php?firstNum=12&secondNum=7 HTTP/1.1\r\n]
    [Severity level: Chat]
    [Group: Sequence]
    Request Method: GET
    Request URI: /GetExample.php?firstNum=12&secondNum=7
    Request URI Path: /GetExample.php
    Request URI Query: firstNum=12&secondNum=7
    Request URI Query Parameter: firstNum=12
    Request URI Query Parameter: secondNum=7
    Request Version: HTTP/1.1
    Host: 10.0.1.185:8080\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8\r\n
    Accept-Language: en-US,en;q=0.5\r\n
    Accept-Encoding: gzip, deflate\r\n
    Connection: keep-alive\r\n
    Referer: http://10.0.1.185:8080/GetForm.php\r\n
    Upgrade-Insecure-Requests: 1\r\n
    \r\n
    [Full request URI: http://10.0.1.185:8080/GetExample.php?firstNum=12&secondNum=7]
    [HTTP request 1/1]
    [Response in frame: 20]

Link-layer address type: Ethernet (1)
Link-layer address length: 6
Source: vnet0000 10:00:00:00:00:00 (00:00:00:00:00:00)
Unused: 0000
Protocol: IPv4 (0x0000)
Internet Protocol Version 4, Src: 10.0.1.107, Dst: 10.0.1.185
Transmission Control Protocol, Src Port: 59026, Dst Port: 8080, Seq: 352, Ack: 799, Len: 525
Hypertext Transfer Protocol
  POST /PostExample.php HTTP/1.1\r\n
    [Expert Info (Chat/Sequence): POST /PostExample.php HTTP/1.1\r\n]
    [POST /PostExample.php HTTP/1.1\r\n]
    [Severity level: Chat]
    [Group: Sequence]
    Request Method: POST
    Request URI: /PostExample.php
    Request Version: HTTP/1.1
    Host: 10.0.1.185:8080\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8\r\n
    Accept-Language: en-US,en;q=0.5\r\n
    Accept-Encoding: gzip, deflate\r\n
    Content-Type: application/x-www-form-urlencoded\r\n
    Content-Length: 23\r\n
    Origin: http://10.0.1.185:8080\r\n
    Connection: keep-alive\r\n
    Referer: http://10.0.1.185:8080/PostForm.php\r\n
    Upgrade-Insecure-Requests: 1\r\n
    \r\n
    [Full request URI: http://10.0.1.185:8080/PostExample.php]
    [HTTP request 2/2]
    [Prev request in frame: 38]
    [Response in frame: 44]

File Data: 23 bytes
HTML Form URL Encoded: application/x-www-form-urlencoded
  Form item: "firstNum" = "4"
    Key: firstNum
    Value: 4
  Form item: "secondNum" = "12"
    Key: secondNum
    Value: 12
```

- 3) The GET request encodes the parameters in the GET portion of the URL. The POST request encodes them as part of the HTML form, not the URL.
- 4) The web server knows that the client is using a Mozilla browser on Linux, Firefox v102.0. It knows that the browser supports webp, xhtml, html, xml, avif, and is English language based. The web server also knows the remote IP of the client. Additionally, the web server knows that the client wants gzip or deflate compressed files.

5) See screenshots.

The first screenshot shows a packet capture of an HTTP 200 OK response. The packet list shows a frame of 518 bytes captured on interface any, id 0. The packet details pane shows the following information:

- Linux cooked capture v1
- Internet Protocol Version 4, Src: 10.0.1.185, Dst: 10.0.1.107
- Transmission Control Protocol, Src Port: 8080, Dst Port: 59020, Seq: 799, Ack: 877, Len: 450
- Hypertext Transfer Protocol
 - HTTP/1.1 200 OK
 - [Expert Info (Chat/Sequence): HTTP/1.1 200 OK]
 - [HTTP/1.1 200 OK]
 - [Severity level: Chat]
 - [Group: Sequence]
 - Response Version: HTTP/1.1
 - Status Code: 200
 - [Status Code Description: OK]
 - Response Phrase: OK
 - Date: Thu, 07 Sep 2023 22:19:01 GMT
 - Server: Apache/2.4.53 (Rocky Linux)
 - X-Powered-By: PHP/8.0.27
 - Keep-Alive: timeout=5, max=99
 - Connection: Keep-Alive
 - Transfer-Encoding: chunked
 - Content-Type: text/html; charset=UTF-8

The packet bytes pane shows the following HTML content:

```
<!DOCTYPE html>
<html>
<head>
<title>Addition Using PHP (POST)</title>
</head>
<body>
<h1>Simple Addition Using PHP (POST)</h1>
<p>The result of 4 + 12 is: 16</p>
</body>
</html>
```

The second screenshot shows a packet capture of an HTTP 200 OK response. The packet list shows a frame of 534 bytes captured on interface any, id 0. The packet details pane shows the following information:

- Linux cooked capture v1
- Internet Protocol Version 4, Src: 10.0.1.185, Dst: 10.0.1.107
- Transmission Control Protocol, Src Port: 8080, Dst Port: 43978, Seq: 1, Ack: 423, Len: 466
- Hypertext Transfer Protocol
 - HTTP/1.1 200 OK
 - [Expert Info (Chat/Sequence): HTTP/1.1 200 OK]
 - [HTTP/1.1 200 OK]
 - [Severity level: Chat]
 - [Group: Sequence]
 - Response Version: HTTP/1.1
 - Status Code: 200
 - [Status Code Description: OK]
 - Response Phrase: OK
 - Date: Thu, 07 Sep 2023 22:18:09 GMT
 - Server: Apache/2.4.53 (Rocky Linux)
 - X-Powered-By: PHP/8.0.27
 - Keep-Alive: timeout=5, max=100
 - Connection: Keep-Alive
 - Transfer-Encoding: chunked
 - Content-Type: text/html; charset=UTF-8

The packet bytes pane shows the following HTML content:

```
<!DOCTYPE html>
<html>
<head>
<title>Addition Using PHP (Without Form)</title>
</head>
<body>
<h1>Simple Addition Using PHP (Without Form)</h1>
<p>The result of 12 + 7 is: 19</p>
</body>
```

- 6) The only major difference is in the HTML content provided back to the client – one says GET and the other says POST. The clients know that this is an apache 2.4.53 server running on Rocky Linux 9, running PHP 8.0.27. The client also knows the server's keep-

alive TCP connection timers, default character set of utf-8, and that it wants to use chunked responses for large amounts of data.

- 7) 5.5 KB were exchanged throughout the course of the two transactions. This includes a 404 or two when I messed up an address.
- 8) There were two requests made to the web server VM.

Process Name	Sent Bytes	Rcvd...	Sent Packe...	Rcvd Pack...
Python	310 bytes	836 bytes	2	4

- 9) There was 836 bytes of inbound data from said web server.
- 10) There is not much information sitting on the web server with no external references and only a couple internal references.
- 11) There were 4 requests made to the wordpress VM.

Process Name	Sent Bytes	Rcvd...	Sent Packe...	Rcvd Pack...
Python	598 bytes	7 KB	4	8

- 12) There was only 7KB of traffic transferred from the wordpress VM. The amount of data transferred is much larger than the web server because wordpress pages are javascript and php, not just basic HTML.
- 13) There is no data in my Wordpress web site, so there is almost no data to find except the home page.

Depth 1:

Process Name	Sent Bytes	Rcvd...	Sent Packe...	Rcvd Pack...
Python	1 KB	94 KB	6	72

Depth 2:

Process Name	Sent Bytes	Rcvd...	Sent Packe...	Rcvd Pack...
Python	53 KB	7.6 MB	203	5,773

Depth 3:

Process Name	Sent Bytes	Rcvd...	Sent Packe...	Rcvd Pack...
Python	712 KB	80.1 MB	2,756	61,607

- 14) Depth 1: 6
Depth 2: 203
Depth 3: 2756
- 15) Depth 1: ~94 KB

Depth 2: ~7.6 MB
Depth 3: ~80.1 MB

- 16) I would be concerned about being DDOSed. My internet connection is only 500Mb/s, and that could easily be overwhelmed by a single host sending rapid connections.
- 17) I would use a DNS provider like Cloudflare to prevent DDOS attacks. Cloudflare acts as a proxy and rate-limits web traffic to a reasonable limit. Additionally, setting up a proxy server in AWS and redirecting all my traffic to pass through that.