HTTP in Action

Experience HTTP

The Exercises

1. See what a web *browser* <u>really</u> sends.
   - Use ncat as HTTP server to receive requests

2. See what a web *server* <u>really</u> sends.
   - Use ncat as web client. Send a request to a web server.

3. Redirect a web browser using HTTP response codes and Location header.

ncat - tool to send & receive TCP

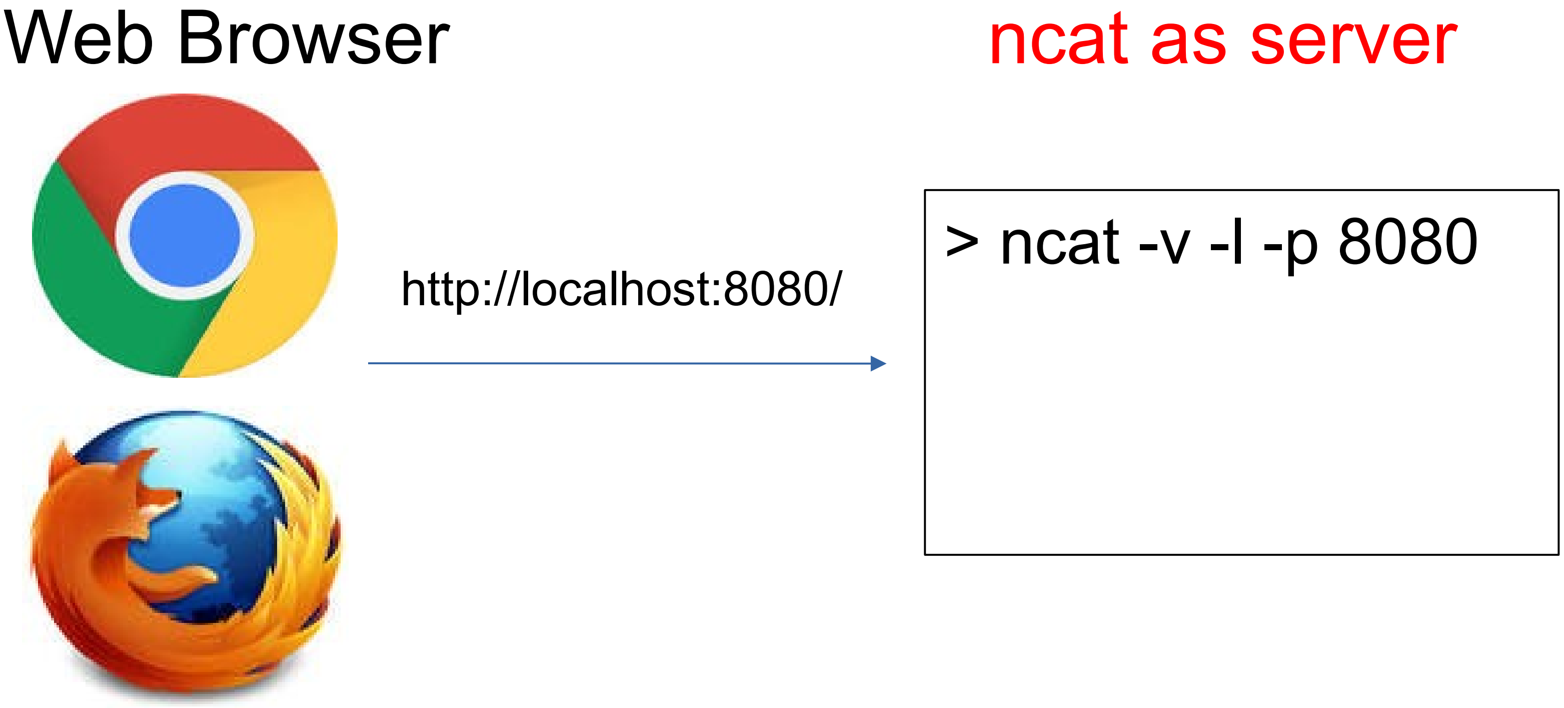
`ncat` is a tool that lets you:

- manually send and receive TCP packets
- be a server that accepts client connections

- "netcat" (nc) - older tool, part of Linux and Mac OSX.
- "ncat" is newer implementation that supports SSL/TLS. Available for Mac, Linux, and Windows.
- Home: `https://nmap.org/ncat`
- Download ncat as part of the nmap package: `https://nmap.org/download.html`

Exercise 1: ncat as a HTTP server

See what a request from a web browser really looks like.



Web Browser

ncat as server

http://localhost:8080/

```
> ncat -v -l -p 8080
```

Exercise 1: Use ncat as a server

1.1 In a terminal window. run ncat (or netcat) as a TCP server listening on port 8080

```
ncat -v -l -p 8080
```

- `-l` means listen for connections, `-v` means verbose
- You can use any free port number 1024 - 65535. (You must be *root* or *admin* to use ports 1-1023.)
- To receive a request from a **different** host, make sure there is no firewall blocking tcp port 8080 (or whatever).

What is my browser sending?

1.2 Open a **web browser** and send a request to ncat:

```
http://localhost:8080/make-my-day
```

(8080 is the port number ncat is listening on)

Use `http:` not `https` (encrypted http).

If you use https, the request shown in ncat window will look like gibberish.

What did the ncat server receive?

1.3 The ncat console should print something like this:

```
GET /make-my-day HTTP/1.1
Host: localhost
Connection: keep-alive
User-Agent: Mozilla/5.0 (X11; Linux x86_64)
  AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/69.0.3497.81 Safari/537.36
Accept:
  text/html,application/xhtml+xml,application/xml;
  q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,th;q=0.8
```

The Browser is Waiting for a Reply

You can see the browser is waiting for a reply.

We will use netcat to send a reply.

You are a human web server!

Send a Reply using HTTP protocol

1.4 In the ncat window, type a reply using HTTP.

First line **must** be "`HTTP/1.1 200 OK`"

```
HTTP/1.1 200 OK
Content-type: text/html
                           <-- blank line
<html><body>
<h1>Hello, Web Surfer</h1> <-- anything
                               you like
```

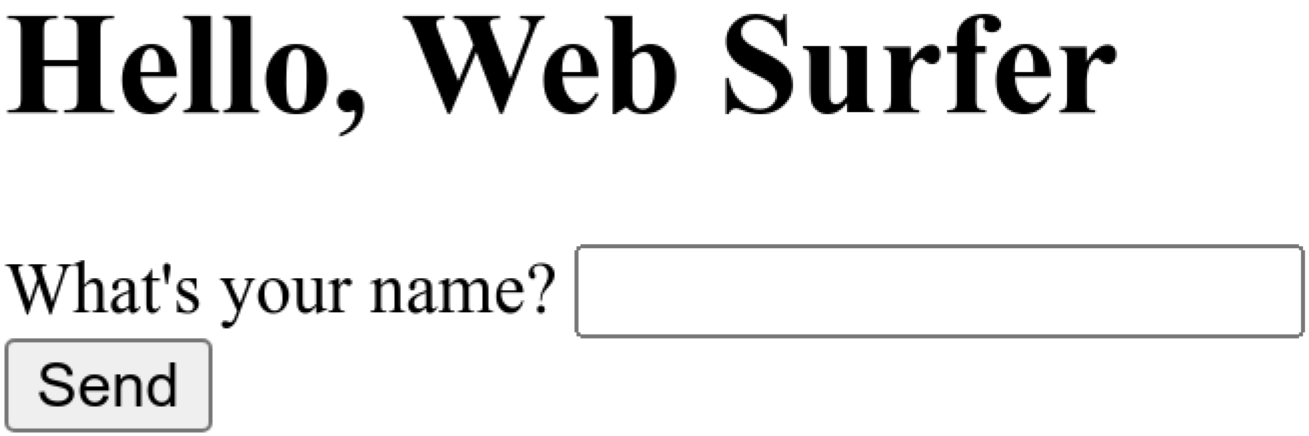
Let's Send a Form, too!

After the <h1>Hello Web Surfer</h1>, let's send a form:

```
<form method="POST">
<label>What's your name?</label>
<input type="text" name="username" />
<br/>
<button type="submit">Send</button>
</form>
```

<---- End the transmission by pressing
CTRL-D or CTRL-Z (Windows) or CTRL-C

Do You See the Form?



Hello, Web Surfer

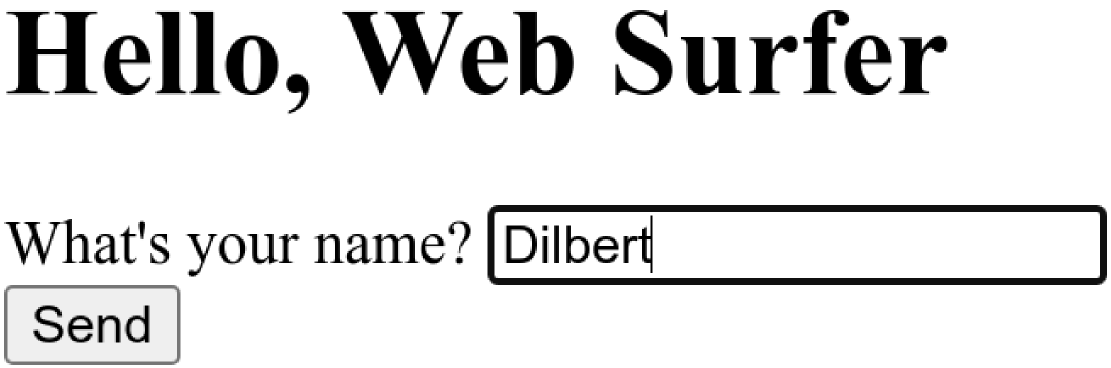
What's your name?

Don't press "Send" yet!

You need to restart ncat to listen for the response:

```
ncat  -v -l -p 8080
```

Restart ncat, then Send a Reply



Hello, Web Surfer

What's your name? Dilbert

When you press "Send", what does ncat show?

Form sends "POST" request with body

Unlike "GET", "POST" request can have a *body* with *data*.

```
POST /makemyday HTTP/1.1
Host: localhost:8080
Connection: keep-alive
Content-Length: 16
Content-Type: application/x-www-form-urlencoded
Referer: http://localhost:8080/makemyday

username=Dilbert        <------ form data
```

Your Turn

Use ncat to send a personal greeting to the web browser.

1. Send an HTTP 200 or 201 response code.
2. In the "body", send a greeting with the user's name.
3. Press Ctrl-D or Ctrl-Z (windows) or Ctrl-C to end the transmission (otherwise, browser will wait for more data).

Exercise 2

What does a *real* web server send?

Now we know what a request from a web browser looks like.

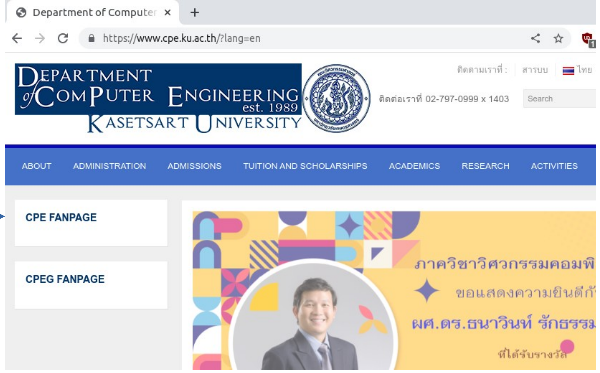
What does a ***real*** reply from a ***real*** web server look like?

ncat as web client

www.cpe.ku.ac.th

```
> ncat -v hostname 80

GET /   HTTP/1.1
```



Exercise 2: Use ncat to send http request

2.1 Send an HTTP request to http://www.cpe.ku.ac.th/

You must enter the HTTP request yourself !

```
cmd> ncat -v www.cpe.ku.ac.th 80
GET / HTTP/1.1
Host: www.cpe.ku.ac.th
(enter a blank line)
```

- -v means verbose: ncat prints status messages.
- Another way: **`curl -v http://www.cpe.ku.ac.th/`**
  curl can also use https.

Exercise 2: Use ncat to send http request

2.1 Send an HTTP request to http://www.cpe.ku.ac.th/

You must enter the HTTP request yourself !

```
cmd> ncat -v www.cpe.ku.ac.th 80
Ncat: Connected to 158.108.215.144
GET / HTTP/1.1
Host: www.cpe.ku.ac.th
 (enter a blank line)
```

-v means verbose. Ncat prints a message when you are connected ("`Connected to 158.108.215.144`")

What does server's reply mean?

What does the reply mean?

```
HTTP/1.1 301 Moved Permanently
Server: nginx
Location: https://cpe.ku.ac.th/
Content-Length: 178
Content-Type: text/html
... (more header and body)
```

- Status codes 301, 302, and 303 are **redirects**
- A web browser will automatically go to the new URL.
- Server "leaked" some info: *what did you learn?*

2.2 Follow the Redirect & Use https

2.2 Stop ncat (CTRL-C) and go to cpe.ku.ac.th

We must use TLS (**`--ssl`** flag) and port 443

```
cmd> ncat -v --ssl cpe.ku.ac.th 443
Ncat: SSL connection to 158.108.215.144
GET / HTTP/1.1
Host: cpe.ku.ac.th
(enter a blank line)
```

Do you get a web page in response?

Plain "http" sites are hard to find

Entire web is moving to https only.

Most http requests are redirected to an https URL.

Try: `www.rd.go.th`

```
cmd> ncat -v www.rd.go.th 80
GET / HTTP/1.1
Host: www.rd.go.th
```

(*worked in 2021*)

What does the response say?

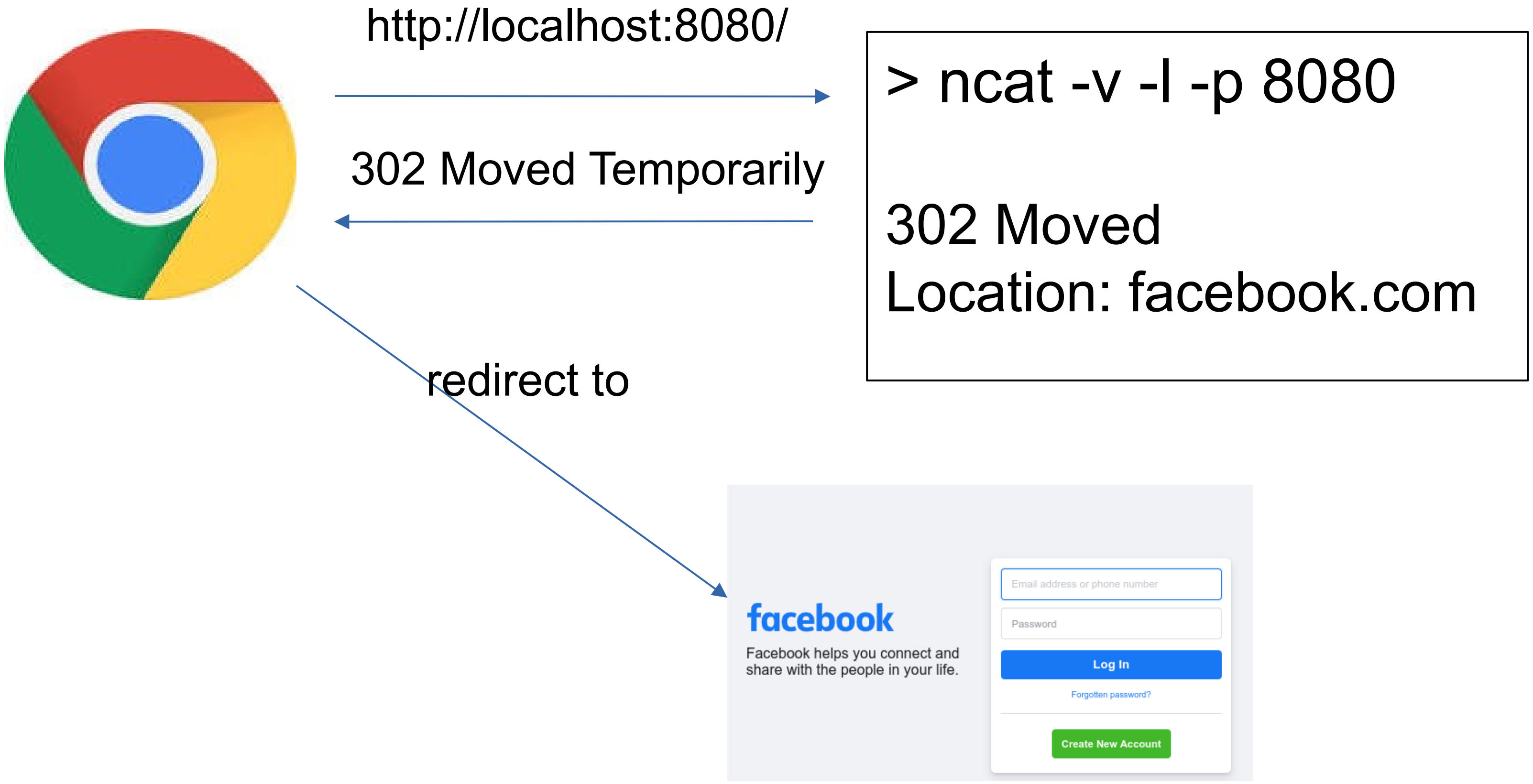
Whose web site is this?

Can we *redirect* a web browser?

See what a request from a web browser really looks like.

Web Browser

ncat as server



http://localhost:8080/

302 Moved Temporarily

```
> ncat -v -l -p 8080

302 Moved
Location: facebook.com
```

redirect to

facebook

Facebook helps you connect and share with the people in your life.

Can We Redirect a Browser?

Exercise: Use ncat to redirect web requests to Facebook.

1. Start ncat in listening (server) mode:

```
cmd> ncat -v -l -p 8080
```

2. Use a web browser, goto `http://localhost:8080`

Redirect the Browser

3. Redirect the browser to Facebook (or anyplace).
   Send status code 302 Moved Temporarily (**not** 301).

```
cmd> ncat -v -l -p 8080
Listening on 0.0.0.0 (family 0, port 80)
Connection from localhost 44240 received!
HTTP/1.1 302 Go away
Location: https://facebook.com

 (blank line)
```

*If you send status code 301 (Moved Permanently) the web browser will **always** go to Facebook instead of localhost.*

Did the Browser obey your redirect?

Browser should follow 302 Redirect to new Location.

You can also send a message in the response body.
In case the browser doesn't follow the redirect.

```
HTTP/1.1 302 Go away
Location: https://facebook.com
Content-type: text/plain

Only AIs allowed.      <---- optional body
Try https://facebook.com instead. :-)
```

Optional Exercises

1. Redirect a friend's web browser.
2. Redirect from inside a web page.
3. How many requests on a page?
4. View page-load statistics using Chrome or Firefox Developer Tools.
    - see how much stuff is downloaded for a single page!

Can you Redirect your Friend's Browser?

Can you get a friend to connect to your ncat server,

and redirect his browser to facebook.com?

Some issues:

1. Friend needs to know your IP address.
   Type `ifconfig` or `ipconfig` to view it.

2. Your TCP port must not be blocked by firewall running on your computer. Windows: use Control Panel to create an exception .

3. KUWIN may block wifi-to-wifi connections (called Wifi isolation)

Redirect inside a Web Page?

In some situations you may want to use redirect a single web page.

You can also add a delay and display a text message:

```
<html>
<head>
<meta http-equiv="refresh"
   content="5; URL='https://facebook.com'" />
</head>
<body>
<h2>You will be redirected in 5 seconds.</h2>
</body>
</html>
```

One page, many http requests

How many HTTP requests are needed to show this page?

```
<HTML>
<link rel="stylesheet" href="stylesheet.css">
<!-- Bootstrap makes my page look cool. -->
<link rel="stylesheet"
  href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
<BODY>
<h1>My vacation</h1>
<p>
For vacation we went to <a
  href=”http://www.unseen.com/bangkok”>Bangkok</a>.
We visited <em>Wat Phra Kaeo</em>, and took this photo:
<br/>
<IMG src="images/watprakaew.jpeg" alt="Wat Phra Keao"/>
```

How Many Requests to Load a Page?

Use web developer tools to see requests, size, & time.

In Chrome:

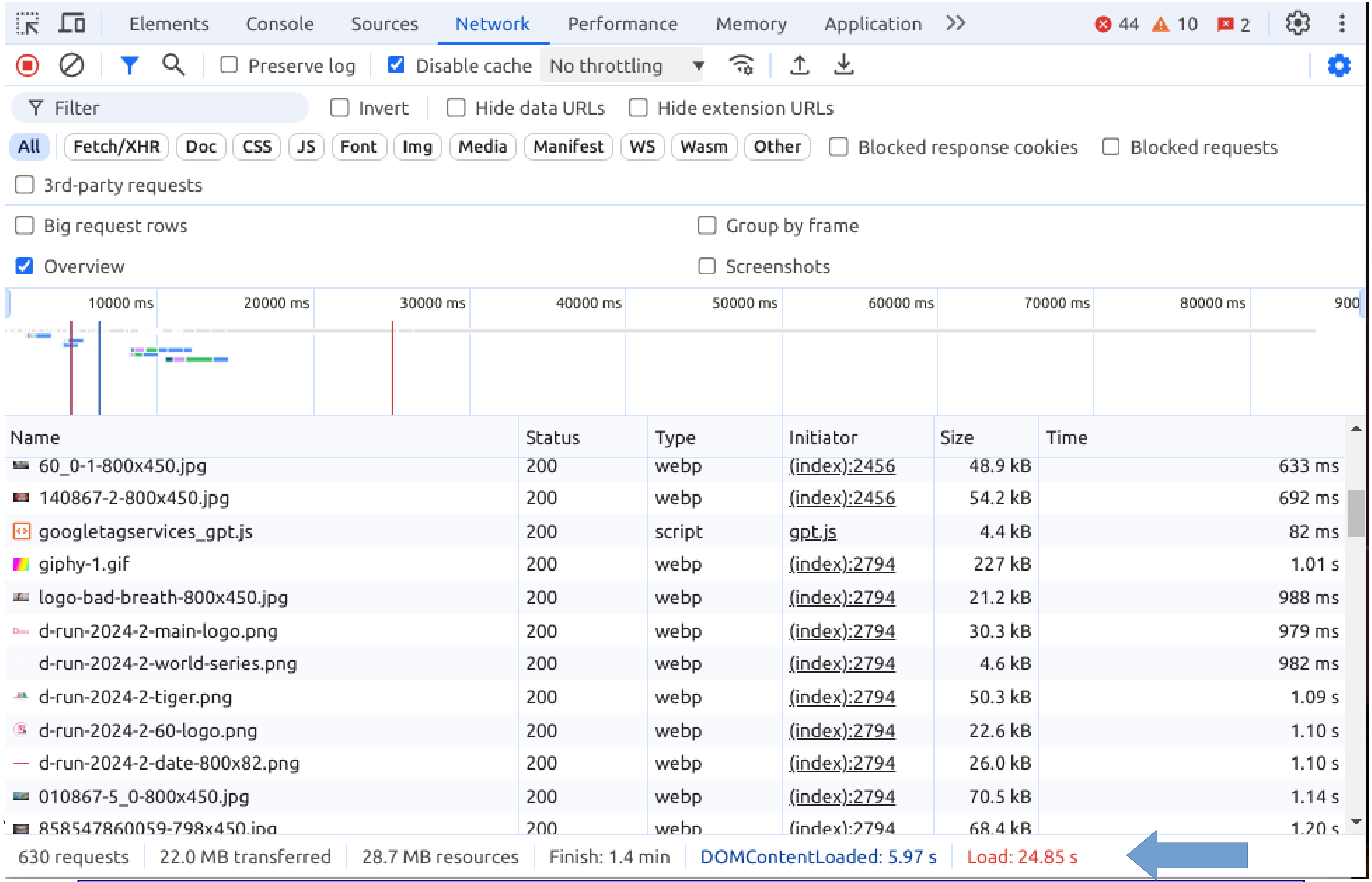
1. From "dots" menu choose More Tools -> Developer Tools
2. In "Developer Tools" window, choose Network tab.
3. Check the box: [x] Disable cache

4. In Chrome, enter a URL (such as dailynews.co.th)

How many requests?   How many MB?

For just one web page!

network stats for dailynews.co.th



□ Preserve log ☑ Disable cache

□ Invert □ Hide data URLs □ Hide extension URLs

□ Blocked response cookies □ Blocked requests

□ 3rd-party requests

□ Big request rows □ Group by frame

☑ Overview □ Screenshots

Name	Status	Type	Initiator	Size	Time
60_0-1-800x450.jpg	200	webp	(index):2456	48.9 kB	633 ms
140867-2-800x450.jpg	200	webp	(index):2456	54.2 kB	692 ms
googletagservices_gpt.js	200	script	gpt.js	4.4 kB	82 ms
giphy-1.gif	200	webp	(index):2794	227 kB	1.01 s
logo-bad-breath-800x450.jpg	200	webp	(index):2794	21.2 kB	988 ms
d-run-2024-2-main-logo.png	200	webp	(index):2794	30.3 kB	979 ms
d-run-2024-2-world-series.png	200	webp	(index):2794	4.6 kB	982 ms
d-run-2024-2-tiger.png	200	webp	(index):2794	50.3 kB	1.09 s
d-run-2024-2-60-logo.png	200	webp	(index):2794	22.6 kB	1.10 s
d-run-2024-2-date-800x82.png	200	webp	(index):2794	26.0 kB	1.10 s
010867-5_0-800x450.jpg	200	webp	(index):2794	70.5 kB	1.14 s
858547860059-798x450.jpg	200	webp	(index):2794	68.4 kB	1.20 s

630 requests | 22.0 MB transferred | 28.7 MB resources | Finish: 1.4 min | DOMContentLoaded: 5.97 s | Load: 24.85 s

630 requests, 22 MB transferred, 28.7 MB resources, Load: 24.5 sec

More Useful HTTP Tools

`wget` - Get one or more files via http/https.

- Used by Zuckerberg in *The Social Network*

`curl` - Transfer data to/from a server using many different protocols, including HTTP & HTTPS

Browser Extensions - send custom HTTP requests and see the response. Good for web services.

I use "RESTED" extension in Firefox and Brave.