

# HTTP/1.1 vs HTTP/2 vs HTTP/3

## Bevezetés

Ez a gyakorlati jegyzet a HTTP/1.1, HTTP/2 és HTTP/3 protokollok működését, különbségeit és teljesítménybeli jellemzőit bemutató előadáshoz kapcsolódik.

## Szükséges eszközök

- Google Chrome vagy Firefox böngésző
- Wireshark (legfrissebb verzió ajánlott)
- curl parancssori eszköz
- Internet kapcsolat

A **curl** parancs Windows verziója letölthető innen: <https://curl.se/windows/>

Megjegyzés: A gyakorlat Windows, Linux és macOS rendszeren is elvégezhető.

## HTTP/1.1 vs HTTP/2

### 1. feladat – HTTP verzió felismerése (curl és böngésző)

Cél: Megállapítani, hogy egy adott weboldal milyen HTTP protokollt használ.

1. Nyiss meg egy parancssort és lépj be a **curl** telepített könyvtárában a “bin” alkönyvtárba.
2. Add ki az alábbi parancsot:

```
curl.exe -I https://httpbin.org
```

3. Figyeld meg a válasz első sorát (pl. HTTP/2 200 vagy HTTP/1.1 200).

4. Kényszerítsd a HTTP/1.1 használatát:

```
curl.exe --http1.1 -I https://httpbin.org
```

5. Kényszerítsd a HTTP/2 használatát:

```
curl.exe --http2 -I https://httpbin.org
```

Magyarázat:

A kliens (curl vagy böngésző) és a szerver közösen dönti el, melyik HTTP verziót használják. A HTTPS kapcsolat esetén ez az ALPN (Application-Layer Protocol Negotiation) során történik.

## 2. feladat – HTTP/1.1 Head-of-Line Blocking megfigyelése

Cél: A HTTP/1.1 soros működésének és a HOL blocking jelenségnek a megfigyelése.

1. Nyisd meg a Chrome DevTools-t (F12).
2. Network fül → Throttling: 'Slow 4G' vagy 3G.
3. Nyiss meg egy weboldalt.

Ez a website HTTP/1.1 protokollt használ: <https://mysupport.enghouse.com/andtek-manuals/>

4. Figyeld meg a Waterfall diagramot.

Megfigyelés:

- HTTP/1.1 esetén több erőforrás várakozik egymásra.
- Egy lassú válasz késlelteti a mögötte lévőket.

Magyarázat:

Ez a head-of-line blocking jelenség, amely a HTTP/1.1 egyik legnagyobb teljesítményproblémája.

## 3. feladat – HTTP/2 multiplexing vizsgálata Wiresharkkal

Cél: Megfigyelni, hogy a HTTP/2 hogyan használ egyetlen TCP kapcsolatot több kéréshez.

1. Indítsd el a Wiresharkot.
2. Válaszd ki az aktív hálózati interfészt.
3. Indíts rögzítést.
4. Nyiss meg egy HTTP/2-t használó oldalt.

`curl.exe -v --http2 http://nghttp2.org/`

5. Állítsd le a rögzítést.

Hasznos Wireshark szűrők:

```
http2  
tcp.port == 80
```

Megfigyelés:

- Egy TCP kapcsolat
- Több HTTP/2 stream (Stream ID-k)

Magyarázat:

A HTTP/2 multiplexing lehetővé teszi, hogy több kérés és válasz párhuzamosan haladjon egyetlen TCP kapcsolaton.

## HTTP/3 és QUIC

### 4. feladat – HTTP/3 forgalom felismerése

Cél: HTTP/3 és QUIC forgalom azonosítása.

1. Wiresharkban indíts új rögzítést.
2. Nyiss meg egy HTTP/3-at támogató oldalt (pl. <https://cloudflare-quic.com>).
3. Állítsd le a rögzítést.

Wireshark szűrők:

```
quic  
udp.port == 443
```

Megfigyelés:

- UDP csomagok
- QUIC protokoll

Magyarázat:

A HTTP/3 nem TCP felett működik, hanem a **QUIC** protokollt használja **UDP** felett.

### 5. feladat – QUIC stream-ek megfigyelése

Cél: A QUIC működési modelljének megértése.

1. Figyeld meg a QUIC csomag típusokat és a kapcsolat felépítésének lépéseit.
2. Figyeld meg, hogy a QUIC forgalom **UDP felett** zajlik, TCP kapcsolat és TCP handshake nélkül.
3. Figyeld meg, hogy a QUIC forgalom **kötelezően titkosított**, a belső stream-ek és stream azonosítók (stream ID-k) nem láthatók visszafejtés nélkül.
4. Figyeld meg, hogy nincs különálló TLS kapcsolatfelépítés; ez a QUIC protokoll része.
5. Figyeld meg a kapcsolat végpontjai által használt logikai azonosítókat: **SCID** és **DCID**.

Magyarázat:

Ez a HTTP/3 + QUIC egyik legfontosabb tulajdonsága a HTTP/2 + TCP-hez képest: A QUIC stream alapú működése miatt nem jelentkezik transport-szintű head-of-line blocking, egy adatfolyam problémája nem blokkolja a többi.