Proxying FiveM Connections using Nginx

As of server version 2377, support was added to FiveM to run a custom streaming proxy similar to cfx.re, but proxying the data connection (UDP) as well. This can be accomplished very easily with nginx acting as a proxy, and you can combine the caching proxy with it.

Requirements

- Nginx 1.17+ (1.16 absolute minimum, guide tested on 1.17)
- Server with reasonable network bandwidth (500mbps uplink **strongly** suggested)
- FXserver version 2377 or later
- SSL certificate for the domain you're using (highly recommended)

Steps

Prerequisites Setup

Some older linux distributions do not have the required version of nginx packaged. You can install the latest via the official nginx repositories if required. More information can be found here: https://docs.nginx.com/nginx/admin-guide/installing-nginx/installing-nginx-open-source/#installing-prebuilt-debian-packages

Nginx Configuration

Below is an example configuration of implementing a combined proxy for both cache and the game server itself. You would place this file inside `/etc/nginx/sites-available` and use it just like any other website.

`proxy-web.conf`

```
upstream backend {
	server your.fivem.server.ip:30120;
}

proxy_cache_path /srv/cache levels=1:2 keys_zone=assets:48m max_size=20g inactive=2h;

server {
    listen 443 ssl;
    listen [::]:443 ssl;

    server_name yourhost.yourdomain.com;

    # SSL is highly encouraged but optional. If not using SSL, comment the below and change the listen blocks above.
    ssl_certificate /path/to/certificate.pem;
    ssl_certificate_key /path/to/privkey.pem;



    location / {
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_pass_request_headers on;
        proxy_http_version 1.1;
        proxy_pass http://backend;
    }

    # if you do not wish to use the caching proxy, remove the below block
    location /files/ {
        proxy_pass http://backend$request_uri;
        add_header X-Cache-Status $upstream_cache_status;
        proxy_cache_lock on;
        proxy_cache assets;
        proxy_cache_valid 1y;
        proxy_cache_key $request_uri$is_args$args;
        proxy_cache_revalidate on;
        proxy_cache_min_uses 1;
    }
}
```

The next file should be placed *outside* this directory (i.e, in `/etc/nginx`):

`stream-proxy.conf`

```
stream {
    upstream backend{
        server your.fivem.server.ip:30120;
    }
    server {
        listen 30120;
        proxy_pass backend;
    }
    server {
        listen 30120 udp reuseport;
        proxy_pass backend;
    }
}
```

`nginx.conf`

Add the following within the `http {}` block, preserving everything else. You can configure the options based on your needs.

```
proxy_cache_path /srv/cache levels=1:2 keys_zone=assets:48m max_size=20g inactive=2h;
```

Add the following **after** the close of your `http {}` block:

```
include /etc/nginx/stream-proxy.conf
```

Use `service nginx reload` (varies by distribution, you just need to reload nginx via your favorite service manager) to apply your changes. You need to do the FXServer steps next before it will be usable.

FXServer Config

This configuration is pretty straightforward.

```
sv_forceIndirectListing true
sv_listingHostOverride yourhost.yourdomain.com
sv_listingIpOverride "your.proxy.ip.address"
sv_proxyIPPranges "your.proxy.ip.address/32"
```

```
# below is optional if you are not using the caching proxy
adhesive_cdnKey "randomKeyHere"
fileserver_add ".*" "https://yourhost.yourdomain.com/files"
```

Place this in your `server.cfg` and restart the server.

Testing

First, test your setup by trying to browse to https://yourhost.yourdomain.com/info.json. If you can see your server's information page, good job! If not, review logs for troubleshooting steps and check your configuration.

Next, attempt to connect to the server via the server list. If connection succeeds, you've successfully set up the server.

You can also connect to your server by using `connect "https://yourhost.yourdomain.com"`, similar to using the cfx.re proxy you get with Nucleus.

Troubleshooting

HTTP errors

Review nginx logs (example, in `/var/log/nginx/error.log`) for clues. Double check your `upstream {}` entries as well.

"Failed to getinfo after 3 attempts"

This error is caused by your client not receiving UDP packets from the proxy. Review upstream settings to ensure you're actually passing these to the server. For this issue, you may have to use wireshark/tcpdump to trace where the packets get lost. Occasionally, the proxy server's host may be to blame.

"Failed to connect after 3 attempts"

Check your nginx version with `nginx -t`. This guide has been tested with 1.17.10 specifically, but any meeting the requirements above should work.

Revision #2
Created 16 August 2024 12:25:53 by alexwilliam
Updated 16 August 2024 12:26:59 by alexwilliam