Bug, unidentified Elixir port, QLM server (Ubuntu): (2024-01-30) (1)

Problem

```
Error found in /var/log/nginx/error.log
upstream prematurely closed connection while reading response header fr
om upstream,
client: 102.16.82.18, server: focicom.re, request: "GET / HTTP/1.1",
upstream: "http://127.0.0.1:4010/", host: "focicom.re"
```

- `upstream prematurely closed connection while reading response header from upstream`: This part of the error message indicates there was a problem with the connection between Nginx and the upstream server (in this case, a server running at `http://127.0.0.1:4010/`\`). The connection was unexpectedly closed before the response header could be fully read.
- `client: 102.16.82.18`: The IP address of the client that made the request.
- `server: focicom.re`: The server name where the error occurred.
- `request: "GET / HTTP/1.1"`: The HTTP request type that was made. In this case, it is a GET request for the root path ("/") using HTTP/1.1.
- `upstream: "http://127.0.0.1:4010/"`: The upstream server to which Nginx forwards requests. In this case, it is a server running at `http://127.0.0.1:4010/`\`.
- `host: "focicom.re"`: The Host header of the client's request.

Causes

1. **Application crash:** If the upstream application crashes or encounters an error, it may close the connection unexpectedly.
2. **Resource exhaustion:** The upstream server may be running out of resources such as memory or file descriptors, leading to a premature connection close.

3. **Timeouts:** If timeout settings are in place, the connection can be closed if generating the response takes too long.
4. **Network issues:** Network problems between Nginx and the upstream server can cause premature connection closures.

Personal analysis:

Bug: It appears that the Elixir application (in this case the focicom site) using port 4010 crashed for a short time and then resumed normal operation, while the Elixir process {beam.smp} continued running on the same port.

As a result, when the site tried to run normally again on port 4010, Nginx could not route to it on that same port because the Elixir process {beam.smp} was already using it.

Solution

Operating flow of the sites: focicom, fracomex, bbmay, cheinmalt

> A client enters the site URL, for example: focicom.re → DNS: focicom.re → 135-125-87.eu:8006 → QLM server → Nginx (on the QLM server) → reverse proxy (ports 80 and 443 redirected to 4030) → local site

> 💡 Difference between proxy and reverse proxy
>
> - **Proxy:** Acts on behalf of the client, forwarding requests to the server.
> - **Reverse proxy:** Acts on behalf of the server, receiving requests and routing them to the appropriate backend server.

Reflex:

- In case of an error, check the Nginx log at /var/log/nginx/error.log
- Assign another port to the website (4030, the site works without any problem)
  - Edit the Nginx configuration file to change the port, file located at

    ```
    /etc/nginx/sites-available/mgbi.conf
    ```

    ```
    # UPSTREAMS

    ##### Lines to edit to change Nginx upstream ports
    ```

```
upstream cheinmalt{
       server 127.0.0.1:8069;
}

upstream bbmay{
        server 127.0.0.1:4000;
}

upstream focicom{
        server 127.0.0.1:4030;
}

upstream fracomex{
        server 127.0.0.1:4020;
}
```

  - Edit the configuration file of the site in question (in our case focicom.re) to change its startup port, file located at

    ```
    /home/mgbi/elixir/focicom/master/focicom/config/dev.exs
    ```

    ```
    ##### Line to modify to change the port for focicom.re
      http: [ip: {0, 0, 0, 0}, port: 4030],
      check_origin: false,
      code_reloader: true,
      debug_errors: true,
      secret_key_base: "a47qN/Cgy6ikhBek4f55bQWq42gjbWE30sVVx2PgLzbsn9lYlFXUrq9/HL0Dq9r8",
      watchers: [
        # Start the esbuild watcher by calling Esbuild.install_and_run(:default, args)
        esbuild: {Esbuild, :install_and_run, [:default, ~w(--sourcemap=inline --watch --loader:.woff=file --loader:.woff2=file --loader:.eot=file --loader:.woff2=file --loader:.svg=file --loader:.ttf=file)]}
      ]
    ```

- Kill the process using port 4010, then restart the site

```
# List open ports and the processes (by PID: PROCESS ID) using them
netstat -tulpn

# List all active processes
ps -aux
# List processes matching a specific name
ps -aux | grep "<process_name or PID>"

# Kill a process by PID
kill <PID>

# Force kill a process
kill -9 <PID>

# Verify the process is gone
After the kill command, run echo $?
# Or check if the PID is still bound to the port
netstat -tulpn
# Or list processes again to confirm
ps -aux
```