Instituto Superior de Engenharia de Lisboa

BEng in Computer Science and Engineering

System Virtualization Techniques, Autumn/Winter 2024/2025

Third coursework assignment

For this assignment, we provide you with a minimal web application developed using Node.js, with data stored in Elasticsearch. The application supports running multiple instances simultaneously on different ports, anticipating the use of a reverse proxy with load balancing to distribute requests across these instances.

You will create a *systemd* service that receives control instructions on a Unix domain socket to:

- Launch one or more new instances of the Node.js application and add them to the reverse proxy.
- Terminate one or more instances of the Node.js application and remove them from the reverse proxy.
- Deactivate the application in the reverse proxy, stopping all instances while keeping the configuration.
- (Re)launch all application instances and (re)activate the application in the reverse proxy.

Setup

Ensure that solutions for the **first** assignment are stored in the **cw1** directory of your repository, and that solutions for the **second** assignment are in **cw2**. If not, move the files now, then commit and push the changes.

Place *tvs-2425-1_cw3.tgz* in the **cw3** directory of your repository and execute: *tar xzvf tvs-2425-1_cw3.tgz*

After the extraction, delete *tvs-2425-1_cw3.tgz*, then commit and push the extracted files and directories.

Before starting this assignment, install additional packages in your Ubuntu 24.04 environment, by executing:

```
sudo apt update
sudo apt install npm nodejs nginx
```

Finally, download and install `elasticsearch` (choose the appropriate version for your CPU):

```
x86-64 ⇒ wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-8.15.3-amd64.deb
 arm64 ⇒ wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-8.15.3-arm64.deb
         sudo dpkg -i elasticsearch-8.15.3-*.deb
         sudo nano /etc/elasticsearch/elasticsearch.yml
            xpack.security.enabled: false              <<< edit this configuration line
```

Preparation

i. Confirm the proper operation of the provided web application:

   a. Install the project's dependencies:
   ```
   cd cw3/tvsapp/app
   npm install
   ```
   b. Launch only the web application, without the database:
   ```
   NODE_PORT=43210 npm start
   ```
   c. Use the browser to access `http://localhost:43210`
   (you should see PORT: 43210 and *database unavailable*)

   d. Start `elasticsearch`:
   ```
   sudo systemctl start elasticsearch
   ```
   e. Use the browser again to access `http://localhost:43210`
   (you should see PORT: 43210 and a counter that increments with each new request made)

   f. Terminate the web application (Ctrl-C), but keep `elasticsearch` running

ii. Copy the web application from `cw3/tvsapp/app` to `/opt/isel/tvs/tvsapp/app` and set it up as a service. Then, start 4 instances in ports 43211, 43212, 43213, and 43214.
NOTE: see `cw3/tvsapp/etc/service/` and Service Templates.

iii. Add a configuration for a new site (`tvsapp`) to `/etc/nginx/sites-available` listening on port 44444 and operating as a load balancer for the 4 local instances on ports 43211 to 43214. Activate the new configuration in `/etc/nginx/sites-enabled` along with the existing one (`default`). Use the browser to access http://localhost:44444 and check the load balancer's operation (PORT changes on refresh). NOTE: see `cw3/tvsctl-srv/etc/nginx/sites-available/` and nginx Configuration Control

Exercises

1. Write the following bash scripts to manage the configuration and operation of the proposed solution:
   - `tvsapp-reset.sh` arguments: `scale` (default = 1), `base` (default = 35000)
     Force an initial stopped configuration with `scale` instances in consecutive ports starting at `base`
   - `tvsapp-inc.sh` arguments: `delta` (default = 1)
     Add `delta` instances of Node.js running the web application, using more consecutive ports
   - `tvsapp-dec.sh` arguments: `delta` (default = 1)
     Remove `delta` instances of Node.js with the highest ports in use, leaving at least 1
   - `tvsapp-stop.sh` arguments: `-db` (optional)
     Deactivate the site from `nginx` and stop web app instances. If `-db`, also stop `elasticsearch`.
   - `tvsapp-start.sh`
     Start `elasticsearch`, all web app instances and (re)activate the site in `nginx`
   - `tvsapp-status.sh`
     Write a summary with the solution status, with one line per element (`nginx`, web apps, db)

   You may use the following command to confirm that the load balancer is distributing the requests:

   ```
   seq 32 | xargs -I{} curl -s http://localhost:44444/ | grep "PORT" |
     sed "s/<\/\?[a-z]\+>//g" | sed "s/^[[:space:]]*//" | sort | uniq -c
   ```

   Tag this exercise on the GitHub repository with: **`CW3-1`** [Reference date: November 18th, 2024]

2. Using the C language, build a *systemd* service (`tvsctld`) to receive instructions on a Unix domain socket, to be placed in `/run/isel/tvsctld/request`, and a client program (`tvsctl`) with the following operations:
   - `tvsctl reset [`*`scale`* `[`*`base`*`]]`
   - `tvsctl inc [`*`delta`*`].`
   - `tvsctl dec [`*`delta`*`]`
   - `tvsctl stop [-db]`
   - `tvsctl start`
   - `tvsctl status`

   The `tvsctld` service is socket activated and runs with *root* privileges. Both the socket and the client (`tvsctl`) will only be available to *root* and members of group `tvsgrp`. to which user `isel` will belong.

   The client program (`tvsctl`) is non-interactive and its first instruction must be `close(0)`. For each execution of the client program, a request is sent to the service (`tvsctld`) through the socket. The service *daemon* will then invoke the appropriate script (from exercise 1) to perform the requested action.

   Write a script (`cw3/install-all.sh`) to copy all the files needed to run the solution from the repository folders to their final location (`/opt/isel/tvs/...`) and set up all permissions appropriately.

   Tag this exercise on the GitHub repository with: **`CW3-2`**

ISEL, November 10th, 2024

Submission last date: November 28th, 2024