LE01
Portfolio Two

Containers and scripting

Synopsis

In order to protect a bash script that fetches random numbers, I want you to create two unprivileged containers and enable networking between them and the host.

The first container should be available to the outside and run a web-server. The other container should run a service which provides random numbers to the first container.

How you implement it is up to you. The following slides contain references and hints that I have tested.

Enable bridged networking

- Follow the instructions below to create unprivileged containers and enable networking:
    - https://help.ubuntu.com/lts/serverguide/lxc.html
- More detailed information on the bridge can be found here:
    - https://angristan.xyz/setup-network-bridge-lxc-net/
- Note that you need to enable DNAT with **iptables** to expose the web-server to the outside.

Installing and starting lighttpd webserver in the first container

- $ **lxc-create -n C1 -t download -- -d alpine -r 3.4 -a armhf**
- $ **lxc-start -n C1**
- $ **lxc-attach -n C1**
- Update package list and install needed packages
- $ **lxc-attach -n C1 -- apk update**
- $ **lxc-attach -n C1 -- apk add lighttpd php5 php5-cgi php5-curl php5-fpm**
- Uncomment the *include "mod_fastcgi.conf"* line in */etc/lighttpd/lighttpd.conf*
- Start the lighttpd service
- $ **rc-update add lighttpd default**
- $ **openrc**

Configure /var/www/localhost/htdocs/index.php

- Create */var/www/localhost/htdocs/index.php* and add the following:

```
<!DOCTYPE html>
<html><body><pre>
<?php
        // create curl resource
        $ch = curl_init();
        // set url
        curl_setopt($ch, CURLOPT_URL, "C2:8080");
        //return the transfer as a string
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
        // $output contains the output string
        $output = curl_exec($ch);
        // close curl resource to free up system resources
        curl_close($ch);
        print $output;
?>
</body></html>
```

Installing the randomness service into the second container

- Use the same process as above to create **C2**.
- Install **socat**.
- Copy the following script to the container:

```
#!/bin/bash

dd if=/dev/random bs=4 count=16 status=none | od -A none -t u4
```

- Serve the script with **socat**, e.g.:

```
socat -v -v tcp-listen:8080,fork,reuseaddr exec:/bin/rng.sh
```

Notes and hints

- Alpine is a minimal linux distribution and uses the **apk** package manager:
    - https://wiki.alpinelinux.org/wiki/Alpine_Linux_package_management
- The init system of Alpine uses **openrc** instead of **systemd**
    - https://www.cyberciti.biz/faq/how-to-enable-and-start-services-on-alpine-linux/
- While debugging the setup use **journalctl** to monitor the logs:
    - https://www.cheatography.com/airlove/cheat-sheets/journalctl/
- While debugging the container, run it in the foreground:
    - **lxc-start -F -n C1**
- You can refer to the containers by name if you add the bridge as a nameserver, i.e. add **10.0.3.1** to the *prepend domain-name-servers* line in */etc/dhcp/dhclient.conf*

Optional tasks

- Automate the whole process.
- Add the **socat** service as an openrc script and enable it at boot.
- Any other embellishment you can think of.

Hand-in and evaluation

- A short **README** of how you implemented your solution including relevant commands used.
- The **README** and any scripts used should be committed to a github repository.
- Send a link to the repository to *thor@mmmi.sdu.dk*
- Your solution will be evaluated as a live demo in class on December 7