2024 Project: Part 1 guidelines

LINFO2142

Nicolas Rybowski, Thomas Wirtgen

September 2024

Pedagogical Objectives

The goal of this project is helping you to learn the following competences:

- Installing a computer from scratch
- Configuring a secure remote access (SSH, firewall, access restriction, ...)
- Basic networking concepts (VLAN, Bridge, Virtual Ethernet interfaces, ...)
- Basic networking tools (tcpdump, netstat, iproute2, drill, nftables, ...)
- Configuring routing protocols (IS-IS, BGP, ...)
- Deploying services on a computer (DNS, Web server, reverse proxy, ...)
- Ensuring reliability of your services (redudancy, load-balancing, anycast, ...)

Practical considerations

The project is performed in groups of two students. Each student receives a Raspberry Pi (RPI), which is a simple micro-computer, for the duration of the semester. Hence, each group will manage a cluster of two RPIs. After performing the basic configuration of the RPIs, they will be placed in a dedicated infrastructure, isolated from the Internet. This means that students won't have physical access to them during the duration of the project.

There is an oral evaluation during which the students of the group must be able to explain the architecture of their infrastructure and each part of the configuration that is required to run the infrastructure. The groups will have to provide beforehand all their configuration files in a zip archive on moodle. More details about the evaluation will be provided later in the semester.

Phase 1: Basic configuration of a Raspberry Pi

During the first phase of the project, you are required to perform a basic configuration of your RPI. In practice, you have to apply the following steps:

1. Install a **fresh**, Linux-based, Operating System (OS) on the SD card of the RPI, as it may still contain an old installation from a previous course.
2. Create a VLAN interface whose *id* is **254 + your group number**. This interface will be connected to a dedicated management network that will allow you to SSH into your RPI. You must configure a static IPv6 prefix on this interface, i.e., `fc00:2142:ffXX::Y/64` where *XX* is your group number in **hexadecimal** and *Y* is 2 or 3 according to the RPI. `fc00:2142::ffXX::1` is your gateway. See the */etc/network/interfaces* file to persistently configure your interface.
3. Configure a SSH server on the RPI. Make sure to create a new SSH key according to the guidelines on this INGInious task, then post it in the task.
4. Prepare your local SSH configuration. We strongly recommend you to configure your SSH access in a SSH config file. Add an entry toward `130.104.78.202` with the username and the key that you use to connect to your RPI. This is our SSH gateway that provides access to the infrastructure. Then, use a **ProxyJump** command to get remote access to your RPI.

At the end of this phase, we will place your RPIs in our infrastructure and you will only have remote access to them after that. When you are outside of the UCLouvain network, you will first have to jump on the `studssh.info.ucl.ac.be` gateway before accessing the course's gateway.

MAKE SURE THAT YOUR CONFIGURATION ARE RESILISENT, I.E., THEY SUPPORT A REBOOT / POWER LOSS.

Phase 2: Harderning your RPI

Now that you have configured your RPI to gain a secure remote access, you are asked to enforce security measures to protect your infrastructure. The bare minimum required is:

1. Restricting your SSH server to pubkey authentication.
2. Disable root access.
3. Deploying a firewall that blocks all entering traffic except SSH. **DOUBLE CHECK YOUR CONFIGURATION OTHERWISE YOU MAY LOSE ACCESS TO YOUR INFRASTRUCTURE.**
4. Protect your SSH server against DDoS attacks (e.g. with fail2ban).

You are free to deploy additional protections at your will.

At the end of this phase, your RPIs are ready to act like virtual Data Centers.

Phase 3: Enabling networking in a virtual infrastructure

You will deploy your virtual infrastructure using the Infrastructure as Code paradigm. To that end, we leverage the containerlab software. This is an overlay over Docker that allows you configuring services in containers and the network that interconnects them. There are many documentations and tutorials on the Internet, but there is a presententation on moodle that summarizes the main features that you are likely to use during the project.

Tip: Make sure to disable the management network in containerlab.

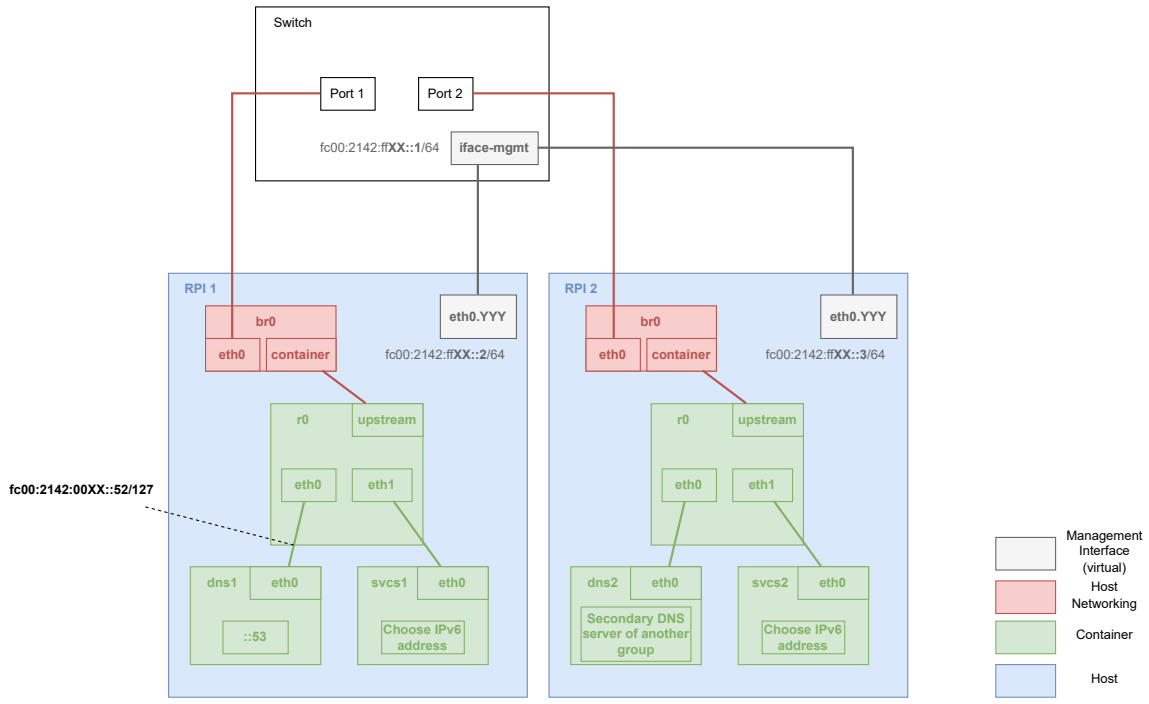

Figure 1: Network configuration of group's infrastructure.

Building specialized Docker containers

As mentionned previously, your infrastructure is not connected to the Internet, which means that you won't be able to build Docker containers on your RPIs. You will need to build your containers for the `aarch64` or `armv7l` architecture on your machine, depending on the OS installed in the RPI. This operation is called *cross-building* a container as the architecture of your own laptop is probably not `aarch64` (neither `armv7l`). To do so, we advise you to look at the `--platform` option of the `docker-build` command and at the `qemu-user-static-binfmt` binary. You can dump a container as a tar archive with the `docker-save` command, send it on your RPI with `scp` then load it in the Docker daemon of your RPI with the `docker-load` command.

Connecting your virtual infrastructure to the physical infrastructure

Now that you are able to build custom Docker containers, you are asked to package FRRouting in one of them and create a software router that will establish IS-IS sessions with the router managed by the TAs. Each group has been attributed a specific IPv6 prefix, i.e., `fc00:2142:XX::/48`, where *XX* is your group number in hexadecimal. In this phase of the project, you are asked to advertise your prefix and receive routes from your neighbors using IS-IS.

With containerlab, create a host link whose end interfaces are called `upstream` in the router container and `container` on the host. The `container` interface must be placed in a bridge along with the `eth0` of your RPI. Then, configure IS-IS in your router container to establish level-1, point-to-point, sessions with our router. In your router container (r0) you will have as many eth* interfaces as the number of service containers you are deploying. Figure 1 shows two interfaces, eth0 and eth1, one for the DNS service and the other for the web server. The DNS server should have a specific address, so you need to configure a prefix in the DNS IPv6 address space (we suggest you a `/127` prefix length). However, for the other eth* interfaces, you are free to set a different prefix of your choice. Enable a **passive IS-IS session** on all eth* interfaces, as you probably do not want to talk IS-IS on the DNS and web containers.

With this setup, your router will advertise the prefixes you have configured in the course's network. At that point, you should be able to ping addresses from other groups.

Tip: If you don't remember how to configure IS-IS sessions, there is a reminder on moodle.

Tip: Do not forget to enable IPv6 forwarding in your router.

Phase 4: Hosting services in your virtual infrastructure

At this point of the project, each group has two hypervisors connected to the course's network. Now you are asked to host services that are available to all the other groups.

The steps for adding new services are always the same:

1. Package a container with the software you need and load it on your hypervisor (RPI).
2. Configure your service with configuration files.
3. Update your containerlab configuration to deploy a new container hosting your service.
    1. Mount the configuration files from your RPI into your container.
    2. Add a link between the new container and your router.
    3. Put the container interface in the router bridge to give it access to the network.
    4. Add a static default route through your gateway in the service container.

Hosting DNS

Each group is responsible of the `grpXX.info2142.internal` domain name, where *XX* is the group number in decimal. The first step to easily deploy reachable services is deploying an authoritative DNS server, e.g. NSD, that will translate your domain names in IPv6 addresses from your prefix. TAs host a recursive DNS server at `fc00:2142:0:0:d0c4::53`. Your authoritative DNS server **must** use the `fc00:2142:XX::53` address, as the TAs recursive server is pre-configured to point to this address for the nameserver of each group. Your are free to choose the authoritative DNS server that suits you the best.

You may have noticed that the DNS container for RPI 2 in Figure 1 is a little unusual. The container you are deploying is not your second DNS server, but that of another group. You will therefore need

to coordinate with another group of your choice to transfer their DNS zone using AXFR or IXFR (if supported) in order to serve the group's DNS zone and thus have redundant DNS.

Hosting a Web service

You are asked to deploy the web server of your choice, e.g. nginx, Apache or Caddy, and host a wiki, i.e., a static website that summarizes the current state of your infrastructure. This website must be reachable through a domain name adverised by your DNS server, i.e. `wiki.grpXX.info2142.internal`. You can use a static website generator such as material for mkdocs to create the content of your site.

Tip: You can access your Web services thanks to a poor's man VPN called `sshuttle` *which is based on TCP forwarding over SSH. Take a look at the* `--dns` *and* `--to-ns` *options to enable DNS resolution of the course's TLD.*

Hosting an additional service

You are required to host *at least* an additional service in your infrastructure. Be creative! There are plenty of very nice open-source software that you can self-host. If you need some inspiration, take a look at Awesome Self-Hosted and Awesome Sysadmin.

The idea is that each group should propose a different service that can be used by the other groups. Document in your wiki how another group can use the service you provide.

Do not forget to update your DNS for each service you host.

Securing your infrastructure

TAs manage a Root TLS Certificate Authority (CA) for the Top Level Domain (TLD) `info2142.internal`. You are asked to manage your own Intermediate CA for your `grpXX.info2142.internal` domain. To that end, you will have to submit a Certificate Signature Request (CSR) to our CA which is hosted on INGInious. We recommend you to use openssl-ca to manage your CA.

Once you have configured your intermediate CA, you are asked to (i) protect each of your Web services with HTTPS thanks to a server TLS certificate you generate, and (ii) secure your DNS connections with the TAs' resolver thanks to DoT/DoH/DoQ.