Kathara networking exercises: BGP configuration

Introduction

These exercises aim to learn and train some of the basic concepts of network interdomain routing using the Border Gateway Protocol (BGP). For that you will use Kathara that is a network emulation system using Docker containers. To be able to do the exercises you must install the following tools:

- Python (version 3 min): https://www.python.org/downloads/
- Docker: https://www.docker.com/
- Kathara: https://github.com/KatharaFramework/Kathara/wiki/Installation-Guides/

After that you can run the exercises by starting Docker and executing the Python scripts:

python3 [exercise_name].py [seed]

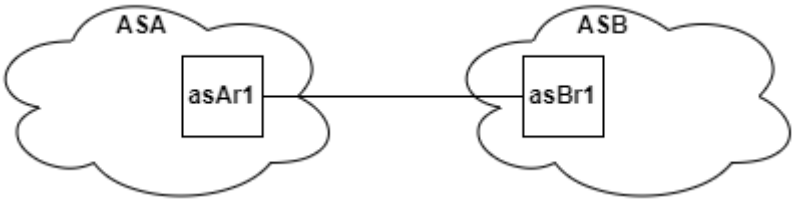
- **[exercise_name]:** Name of the exercise Python script (e.g. "**Kathara-BGP-Ex1-eBGP**").
- **[seed]** (optional): Seed that will be used for the random number generation, you can use your NOMA as seed.

Then you will have a client in your terminal where you can connect to the different routers and applying the configurations. When you have finished the exercise, you can enter "**exit**", and the evaluation will be made. After doing the modifications to the network, we recommend you wait a few seconds before testing or ending the exercise to let the network having the time to converge. The AS numbers and the IP addresses will be randomly generated and displayed in the terminal.

For the configurations you can directly edit the configuration files of the routers ("**/etc/frr/frr.conf**") or use the VTY shell. As you will use FRRouting as Internet routing protocol suite, you can find the different commands and options for the BGP configuration in the FRRouting documentation: https://docs.frrouting.org/en/latest/bgp.html.

Exercise 1: Enable eBGP

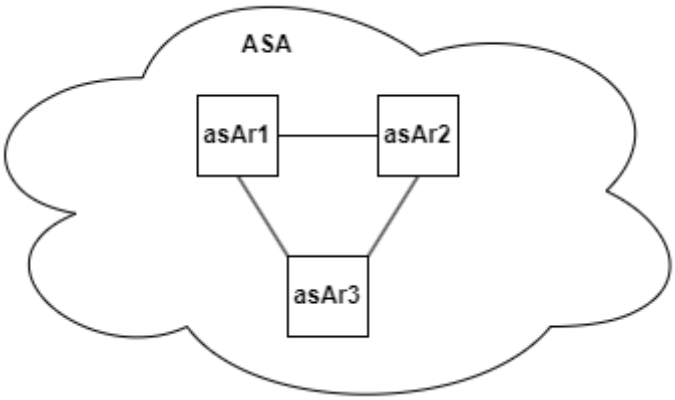
A network topology can be constituted by many Autonomous Systems (AS) and each AS can be constituted by many routers. For an AS to be able to communicate with another AS (interdomain routing), we need to establish an eBGP session between the routers that border the two AS. This exercise consists of doing that on the following topology.



Configure the eBGP session between asAr1 and asBr1. For that you need on both routers to announce the subnet prefix to the network and declare the direct neighbor. For this exercise and the others, you can use the command "**show ip bgp**" in **vtysh** to investigate the BGP table of a router and verify if the different routes are there and are correct.

Exercise 2: Enable iBGP

When router has leaned the routes to the other ASes through his eBGP sessions, it should want to advertise them to the other routers inside its AS in order for them to also be able to communicate with these other ASes. This is the purpose of the iBGP sessions. For this exercise you will set up the iBGP on the following topology.

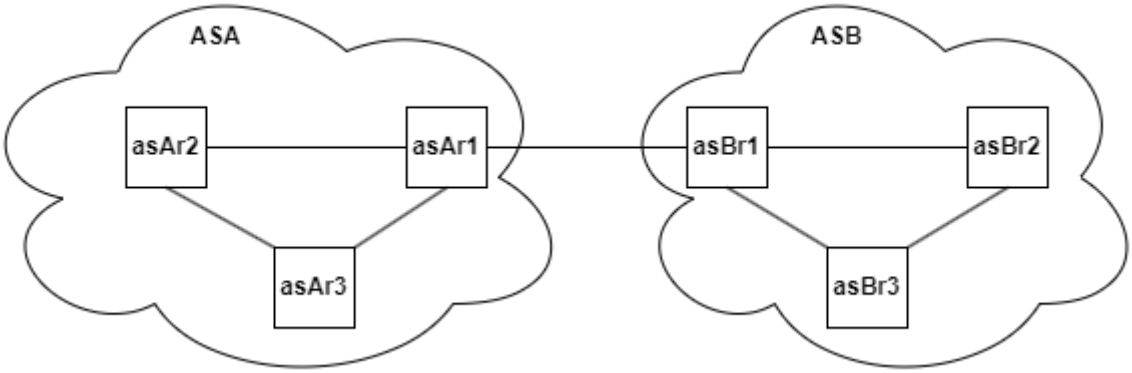


Configure the iBGP sessions between every router in ASA. The routers share the same AS number as they are in the same AS. We would like to draw your attention to the following points when configuring an iBGP session:

- As we are using the loopback addresses of the routers, you need to tell the router to use the loopback address as **source address** for the session.
- When a router advertises a route, we want it to declare itself as **next hop** for the route.

Exercise 3: Enable iBGP and eBGP

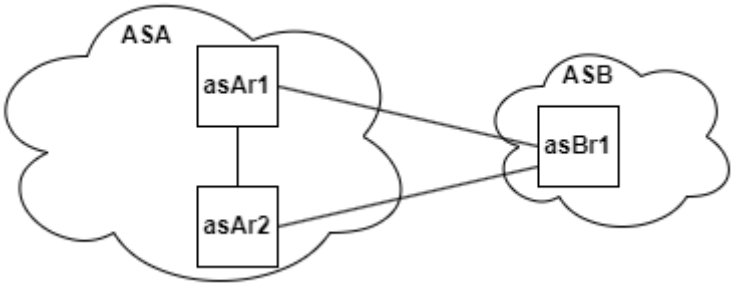
Now that you have learned how to setup iBGP and eBGP sessions, you will try to put them together on a network with the following topology.



Configure the iBGP and eBGP sessions in order for asAr2 and asAr3 to be able to communicate with asBr2 and asBr3 (and vice versa). BGP is the protocol used for the interdomain routing (EGP), but we also need a protocol for the intradomain routing (IGP). For these exercises we will use OSPF as Interior Gateway Protocol, but it will already be configured. Pay attention that the border routers must also be able to **redistribute** the routes that they have learned from OSPF to the eBGP neighbor.

Exercise 4: Multi-Exit Discriminator

When an AS has multiple exits to another AS, he might want to indicate which route he would prefer his neighbor to take to reach him. That's the purpose of the Multi-Exit Discriminator (or MED) attribute. For this exercise you have to configure this feature on a network with this topology (iBGP and eBGP sessions are already configured).

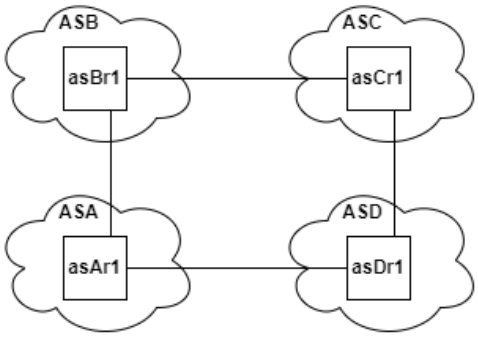


The MED can be configured using **route maps.** A route map is a tool that can be used to filter and apply policies to the routes (https://docs.frrouting.org/en/latest/routemap.html). By default, in BGP, a hidden route map rule is applied that refuse every route to be exchanged. So, for the exercise, you first need to create a route map that allow a route to be accepted and apply this route map to every route that are coming in or coming out.

In BGP, if every parameter is left by default, asBr1 would prefer the route to asAr1 to join ASA because it has the lowest router ID. For the exercise we want asBr1 to prefer the route to asAr2 instead. For that, you can create another route map that sets the MED of route and apply this route map on the route send by asAr1 to asBr1 (in asBr1 the lowest MED route will be chosen).

Exercise 5: Local preference

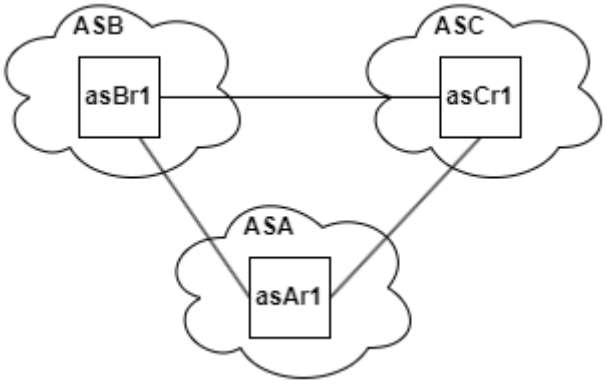
The local preference (or local pref) is another attribute that is used by the BGP routing decision process. If every other parameter is left by default, the route with the highest local pref is chosen. You must configure local pref on the following network (eBGP sessions and ACCEPT_ALL route maps are already configured and applied).



By default, the route from ASA to ASD is asAr1 -> asDr1. Change the local pref using the route maps in order for asAr1 to have the route asAr1 -> asBr1 -> asCr1 -> asDr1 as best route to ASD instead. Do the same for asDr1 that must have asDr1 -> asCr1 -> asBr1 -> asAr1 as best route to ASA.

Exercise 6: Route map - deny a prefix

Route maps are a powerful tool that can also be used to filter some unwanted routes with specific prefixes. We will consider the following network.

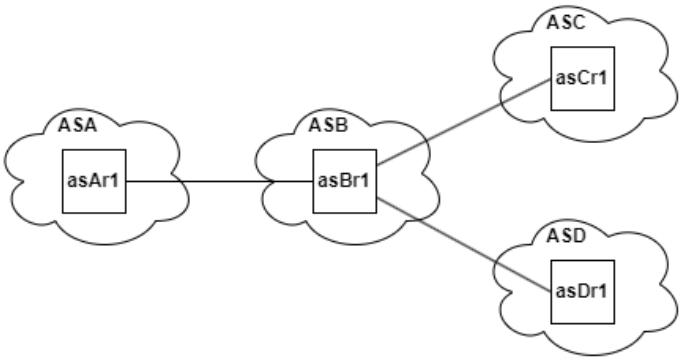


The eBGP sessions and the ACCEPT_ALL routes map are already configured so every AS can communicate with each other. However, ASA and ASB don't want to communicate with ASC anymore. You must configure route maps in order for asAr1 and asBr1 to deny every route coming in or out matching asCr1 prefixes (asAr1 and asBr1 must still be able to communicate with each other). This can be made by these three steps on the two routers:

- Create a prefix list (https://docs.frrouting.org/en/latest/filter.html) that matches the asCr1 prefix
- Create a route map that filter the routes matching the prefix list
- Apply this route map on the correct route. Be careful that this route map must be applied before the ACCEPT_ALL route map, otherwise it will not be applied (the first matching route map in sequence is executed).

Exercise 7: Community

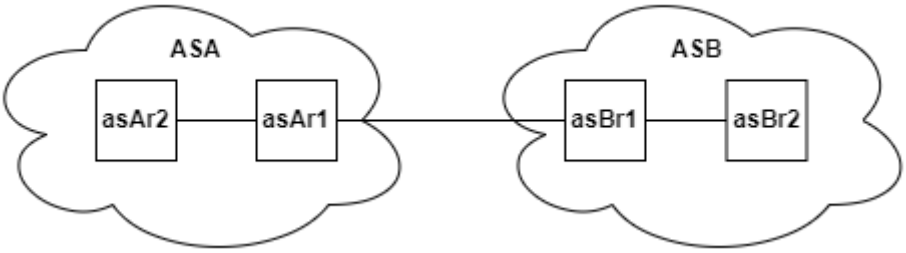
In BGP, a community is a tag or a label that we can put on a route that can be used to easily apply policies on specific routes. Consider the following topology.



This network is already configured. However, we want ASD to not have any route to ASA using a community. For that you can follow these steps:

- In asAr1: create a community using a route map and apply this route map to the prefix that he declares to the network.
- In asBr1: create a route map that matches the community created in asAr1 and apply this route map to the outgoing routes to asDr1

Exercise 8: Find the error



This network is already configured. Unfortunately, there is an error or a missing configuration. Inspect the configurations of the routers (iBGP, eBGP, OSPF, route maps, …), find the error and fix it so that asAr2 has the correct route to asBr2 (and vice versa).