Lab 5 – AWS – Gateway Load Balancer

**Goal** – Utilize the provisioned Gateway Load Balancer architecture to provide security for distributed ingress and centralized egress flows with FortiGate NGFW.

**Task** – Create VPC routes and FortiGate Policy objects allowing both flows of traffic.

**Validation** – Confirm connectivity to Public NLB1 and from Instance-B.

**Introduction**

In this scenario, there are multiple VPCs in the same region that have one instance each. Transit Gateway is configured with multiple Transit Gateway Route Tables, Gateway Load Balancer and endpoints are already configured as well. You will need to create the appropriate VPC routes to redirect traffic to Gateway Load Balancer via the deployed endpoints so the Active-Active FortiGates can inspect the traffic.

In this scenario these FortiGate NGFWs are working together in an Active-Active design to provide more capacity for bump in the wire inspection. This design is usable with workload VPCs that have a direct path to/from the Internet via an attached Internet Gateway (IGW), with or without a NAT Gateway (commonly referred to a distributed design). This design can also work in conjunction with Transit Gateway to offer centralized egress, ingress, and east/west inspection (commonly referred to a centralized design).

**Topology**

A screenshot of a computer

AI-generated content may be incorrect.

**Pre-Work:**

* Delete the Lab3 stacks by navigating to the CloudFormation console and the Lab-4 stack.
* **All resources from Lab 4 must be deleted prior to beginning the next lab**

**Lab 5:**

1 - Create a new stack by navigating to the CloudFormation console and importing the GWLB.json file:

* Click on **Create stack**
* Select **With new resources**

A screenshot of a computer

AI-generated content may be incorrect.

* Click **Choose an existing template**
* Choose **Upload a template file**
* Choose **GWLB.json** as the template file
* Click **Next**

Graphical user interface, text

AI-generated content may be incorrect.

* Stack name: **Lab-5**
* Click **Next**

A screenshot of a computer

AI-generated content may be incorrect.

* IAM Role: **ffirole**
* Behavior on provisioning failure: **Roll back all stack resources**
* Delete newly created resources during a rollback: **Delete all newly created resources**

A screenshot of a computer

AI-generated content may be incorrect.

**Acknowledge the capabilities** and then click **Next**

A screenshot of a computer

AI-generated content may be incorrect.

**Review the template parameters then click on Submit.**

It will take 10-15 minutes for the template to be fully deployed. Once it is finished the main/root stack will show as complete:

A screenshot of a computer

AI-generated content may be incorrect.

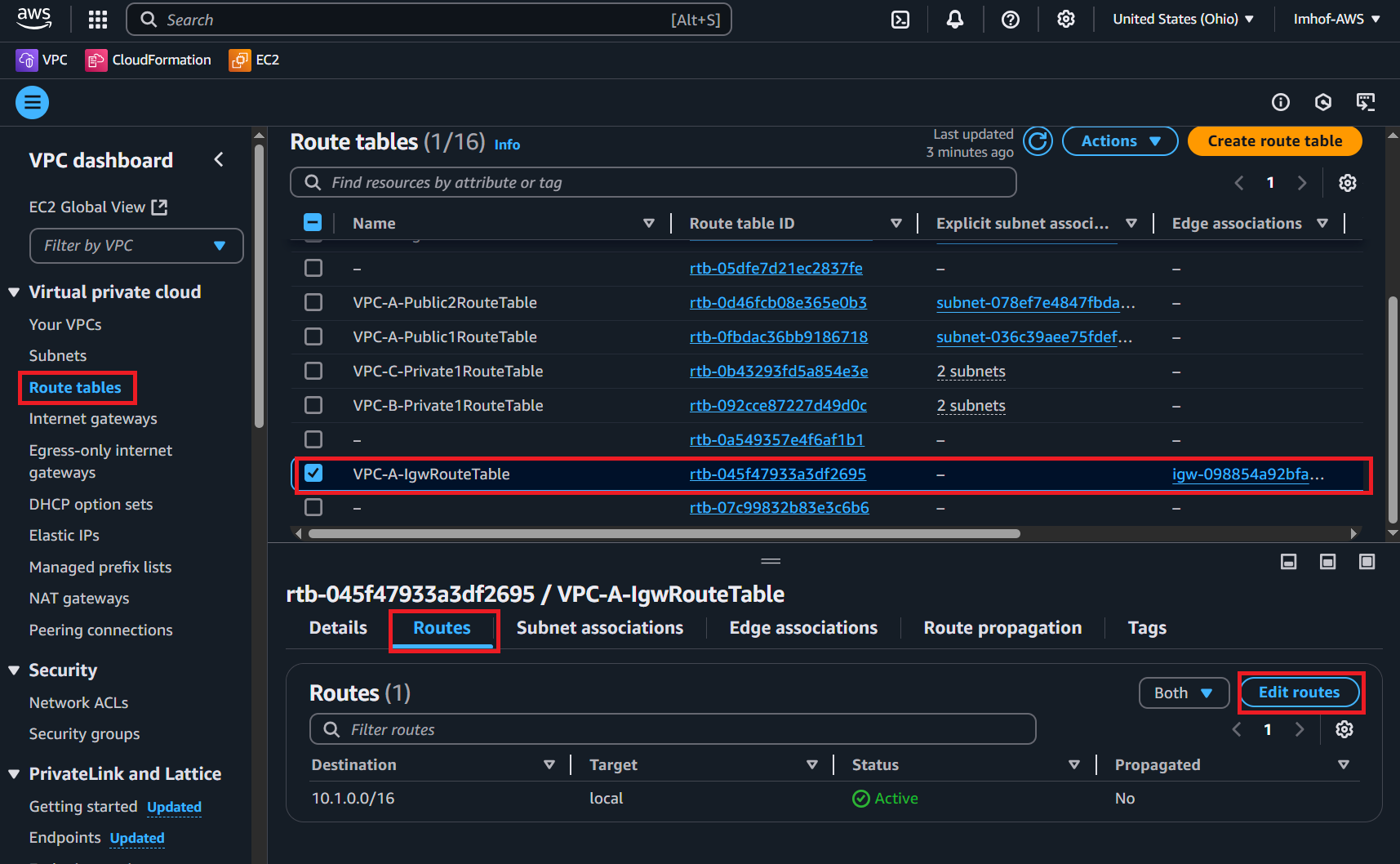
2 – Create VPC routes for ingress routing

* Navigate to the VPC console and go to the Endpoints page. You should see four endpoints, one for each AZ in both VPC-A and the NGFW VPC. **Make note of the Endpoint IDs** as these will be used in the next step.

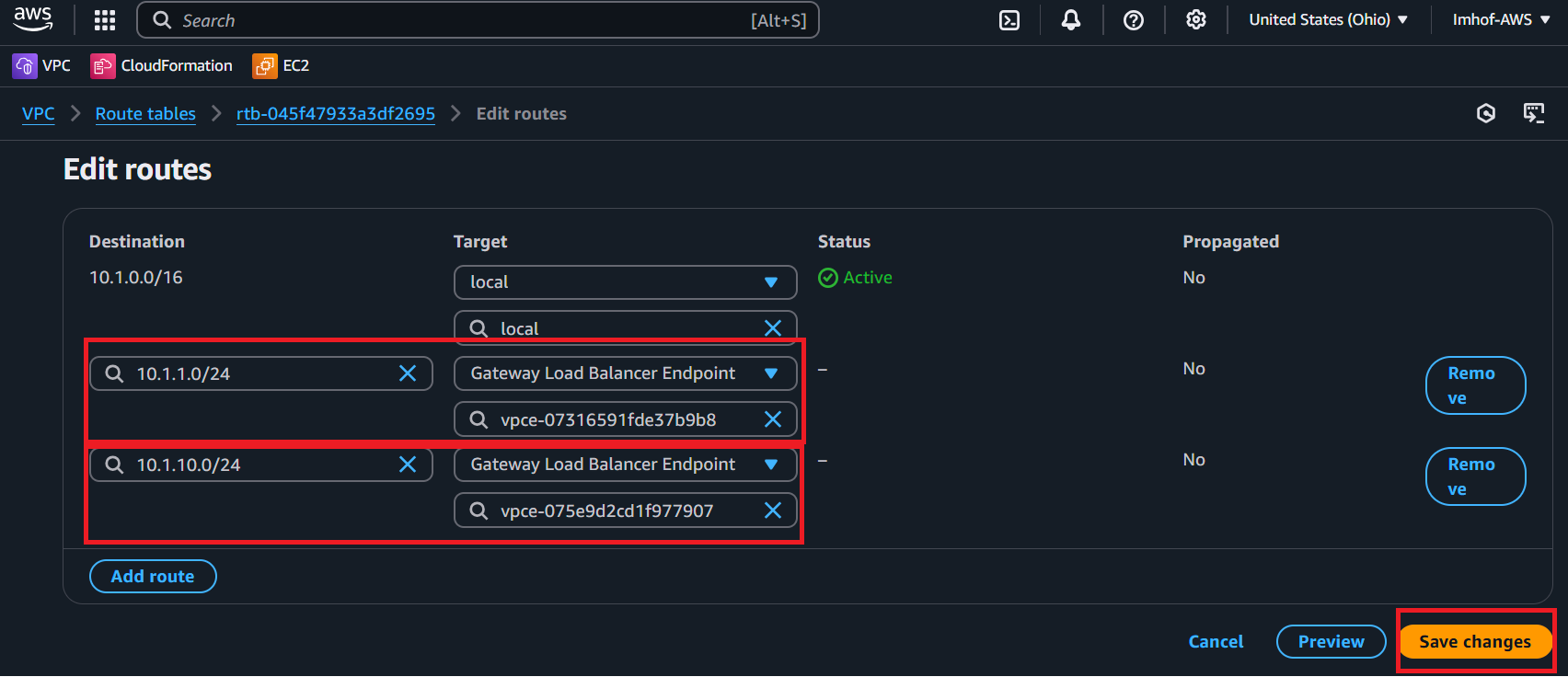
A screenshot of a computer

AI-generated content may be incorrect.

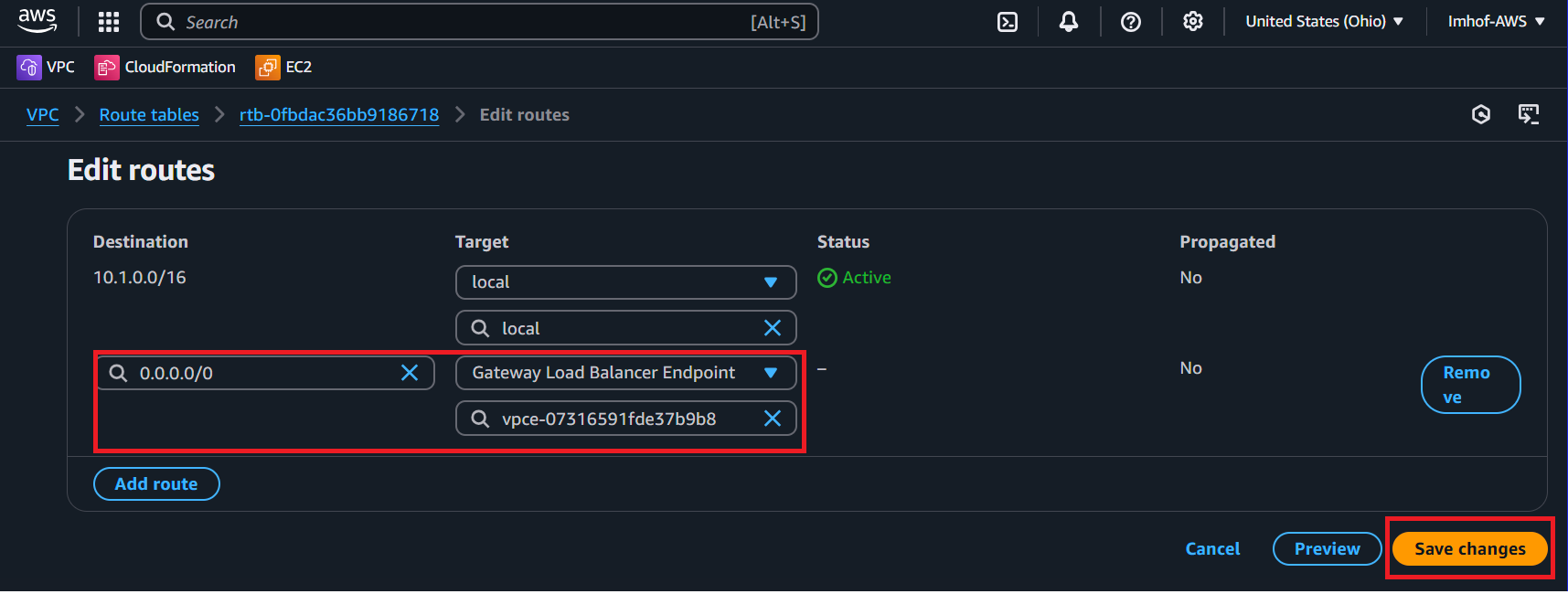
* Navigate to the route tables page, find the *VPC-A-IgwRouteTable* and edit the routes:



* Create a new route with the following parameters:
  + Destination: **10.1.1.0/24**
  + Target: **Gateway Load Balancer Endpoint**
  + Select **VPC-A-GWLB-VPCE-AZ1**
  + Click **Add route**
  + Destination: **10.1.10.0/24**
  + Target: **Gateway Load Balancer Endpoint**
  + Select **VPC-A-GWLB-VPCE-AZ2**
  + Click **Save changes**



* Navigate to the *VPC-A-Public1RouteTable* and add the following route:
  + Destination: **0.0.0.0/0**
  + Target: **Gateway Load Balancer Endpoint**
  + Select **VPC-A-GWLB-VPCE-AZ1**
  + Click **Save changes**



* Navigate to the *VPC-A-Public2RouteTable* and add the following route:
  + Destination: **0.0.0.0/0**
  + Target: **Gateway Load Balancer Endpoint**
  + Select **VPC-A-GWLB-VPCE-AZ2**
  + Click **Save changes**

A screenshot of a computer

AI-generated content may be incorrect.

3 – Create an ingress FW policy using a dynamic address object for Public NLB1 in VPC-A on both FortiGate’s

* Navigate to the CloudFormation console, select Lab-5, and note the outputs for **FGT1LoginURL**, **Username**, and **Password**.
* Log into FGT1 then navigate to **Policy & Objects -> Addresses** and clicking on **Create new**

A screenshot of a computer

AI-generated content may be incorrect.

* Create a dynamic object using the following parameters:
  + Name: **sdn-publicnlb1**
  + Interface: **any**
  + Type: **Dynamic**
  + Sub Type: **Fabric Connector Address**
  + SDN Connector: **aws-instance-role**
  + Addresses to collect: **Private**
  + Filter: **Description=ELB net/PublicNLB1**

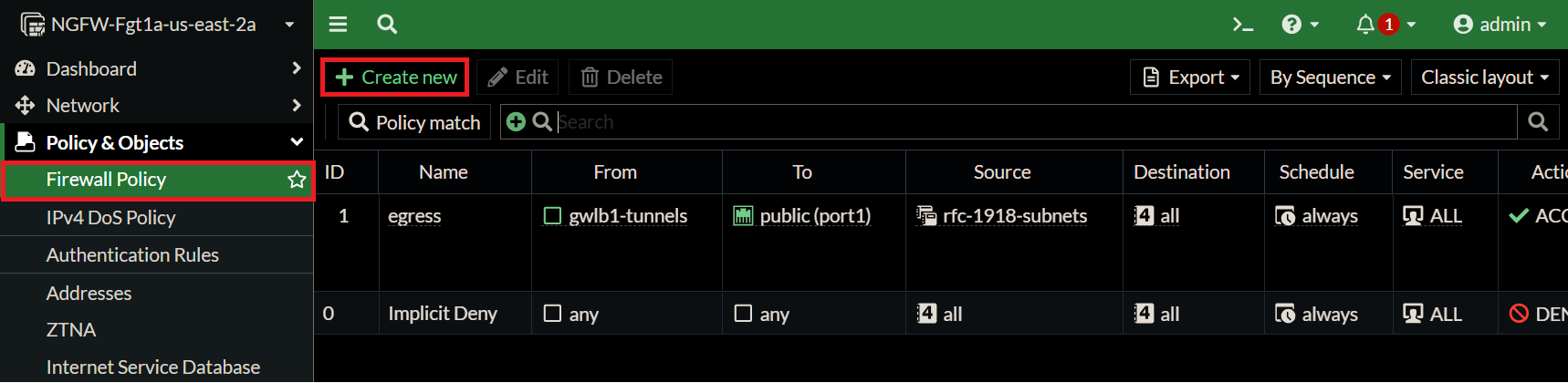
A screenshot of a computer

AI-generated content may be incorrect.

Text

AI-generated content may be incorrect.

* Navigate to **Policy & Objects -> Firewall Policy** and click **Create new**



* Create a new policy with the following parameters:
  + Name: **ingress**
  + Type: **Standard**
  + Incoming interface: **gwlb1-tunnels**
  + Outgoing interface: **gwlb1-tunnels**
  + Source: **United States**
  + Destination: **sdn-publiclb1**
  + Schedule: **always**
  + Service: **HTTP**
  + Action: **Accept**
  + Inspection Mode: **Flow-based**
  + NAT: **Disabled**
  + Application control: **default**
  + IPS: **default**

A screenshot of a computer

AI-generated content may be incorrect.

* + SSL inspection: **certificate-inspection**
  + Log allowed traffic: **Enabled, All sessions**
  + Generate logs when session starts: **Enabled**
  + Click **OK**

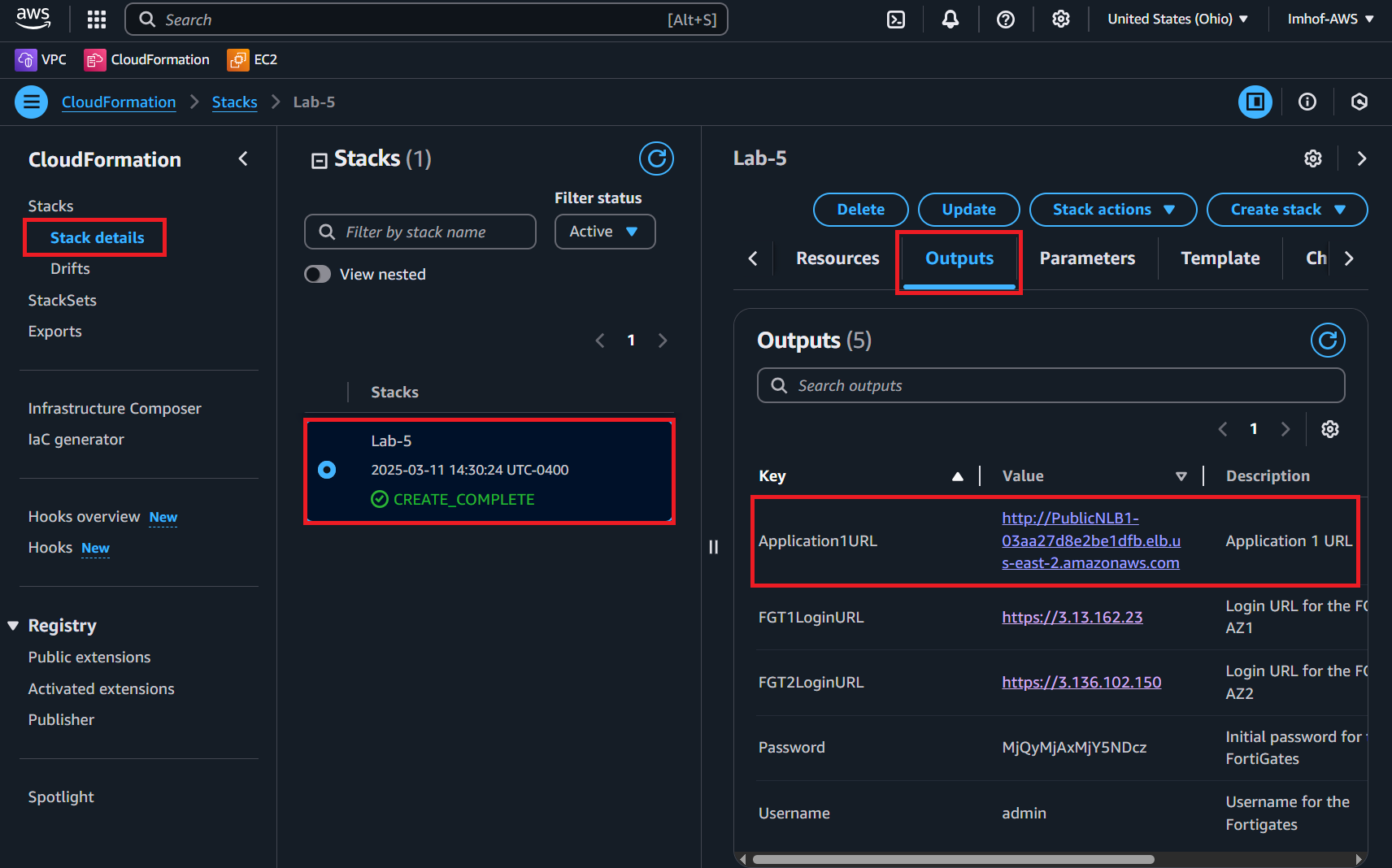
A screenshot of a computer

AI-generated content may be incorrect.

**Repeat creating the dynamic object and firewall policy on FGT2 as this is an active/active design**

4 – Test distributed ingress to Public NLB1 (and Instance-A) through GWLB and FGTs

* Navigate to the CloudFormation console
* Select the Lab-5 main template and select the **Outputs** tab.
* **Copy the URL from Application1URL**.



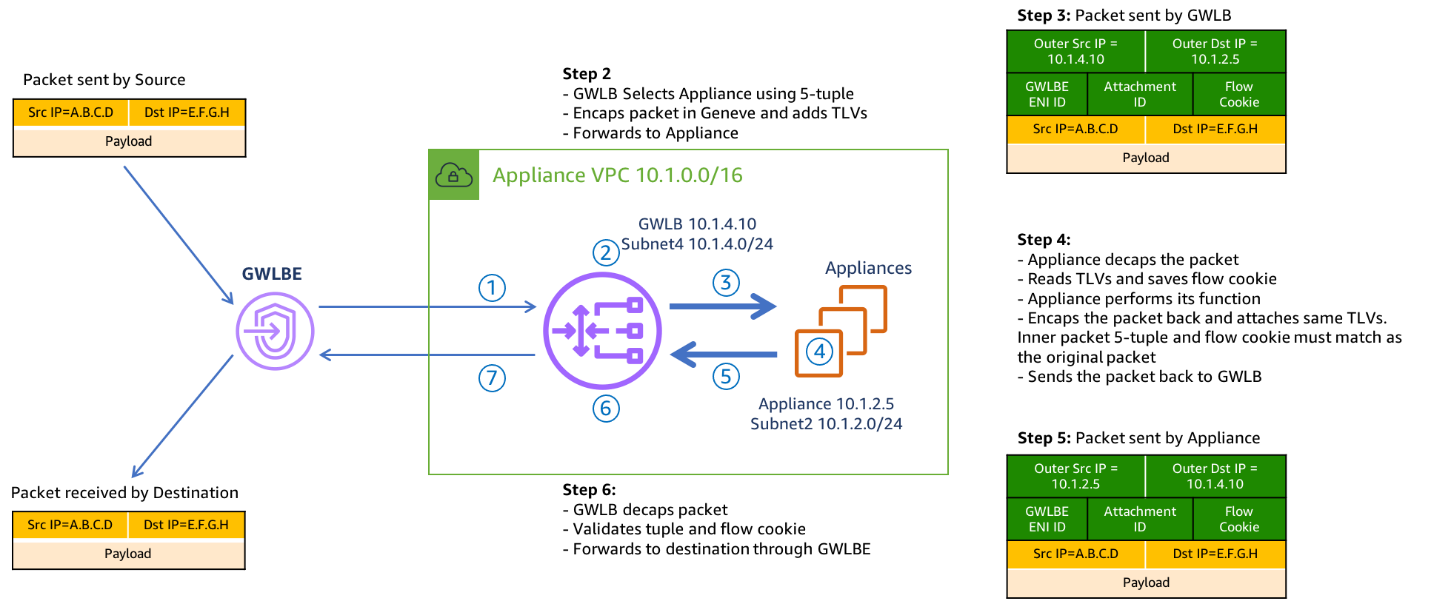
* Navigate to the **Application1URL** using your web browser.

Text

AI-generated content may be incorrect.

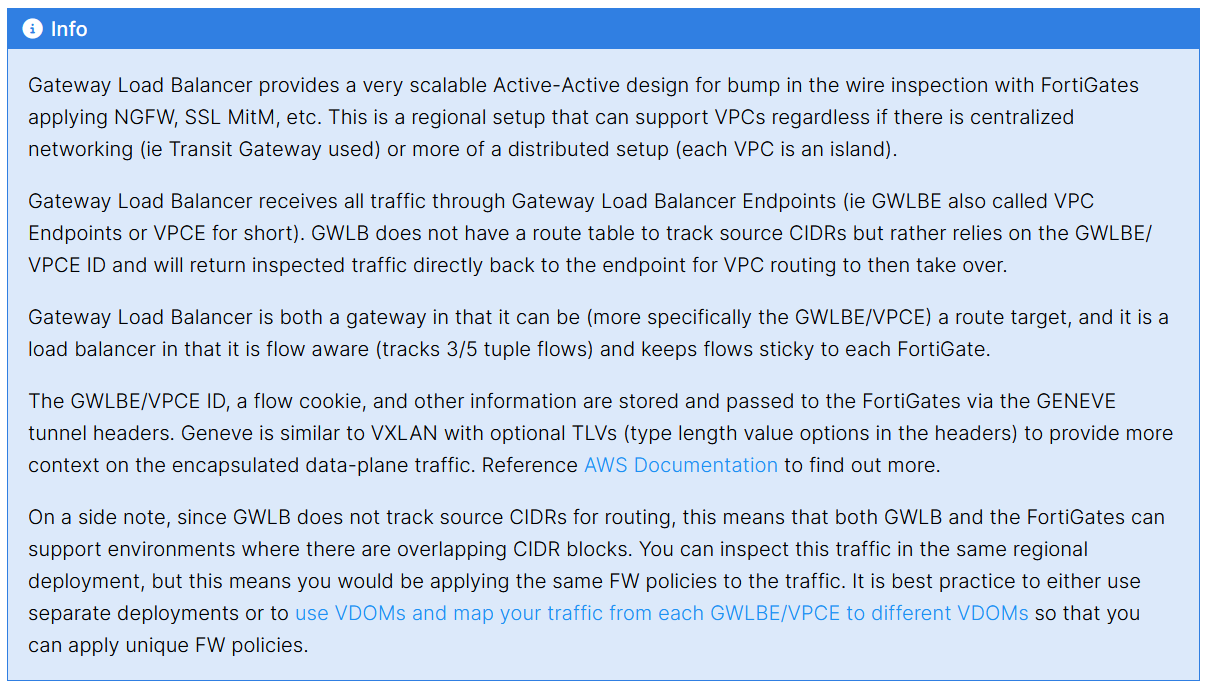
**How is all this working?**

Here is an example of how GWKB routing works including what information is tracked and what is included in the GENEVE tunnel headers:



Text

AI-generated content may be incorrect.



Below is a step by step breakdown of the packet handling for ingress web traffic to Public NLB1 in VPC-A:

Text, application, table

AI-generated content may be incorrect.

Graphical user interface

AI-generated content may be incorrect.

5 – Test centralized egress from **Instance-B** through GWLB and the FGTs

* Navigate to the EC2 console and **connect to** **Instance-B**.
  + username: **admin**
  + password: **FORTInet123!**
* Once connected, run the following connectivity tests:
  + **ping 8.8.8.8**
  + **curl ipinfo.io**

A screenshot of a computer

AI-generated content may be incorrect.

Below is a step by step breakdown of packet handling for centralized egress from Instance-B:

Table

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**6 – Delete this lab by navigating to the CloudFormation console and deleting Lab-5.**

**Discussion Points**

* GWLB is a regional service that is both a gateway (VPC route target) and a flow aware load balancer.
  + This allows very scalable active-active inspection for all directions of traffic.
  + No SNAT requirement to keep flows sticky to the same FGT as GWLB is flow aware.
* GWLB and FortiGates supports one and two arm mode (distributed vs centralized egress access & NAT GW replacement).
* Jumbo frames (8500 bytes) are supported.
* Inspection VPC handles FortiGate NGFW inspection for any traffic flow (Inbound, Outbound, East/West) and for any network design (distributed vs centralized).
  + [**Appliance Mode**](https://docs.aws.amazon.com/vpc/latest/tgw/transit-gateway-appliance-scenario.html) is required for this design to keep flows sticky to the correct availability zone which in turn means the correct GWLB endpoint.
  + Advanced architectures for all of these scenarios can be [**found here**](https://github.com/FortinetCloudCSE/.github/blob/main/profile/AWS/README.md).

**Additional Information**

**One-Arm Model**

GWLB supports two different models of firewall deployments, one-arm and two-arm where a firewall appliance can also perform NAT.

In the one-arm model, the FortiGates will inspect traffic and forward this back to GWLB where Internet bound traffic is has NAT applied by a NAT GW. Typically, the NAT GW will be in a workload VPC in a distributed design. Distributed designs have GWLB endpoints in each workload VPC requiring an attached Internet Gateway (IGW) and public load balancer or NAT GW. A centralized design can use NAT GW in an inspection VPC for centralized egress and have GWLB endpoints only deployed in the inspection VPC (no need for GWLB endpoints in each workload VPC).

Diagram

AI-generated content may be incorrect.

We can use static and policy routes like below to support this setup. In a 2 AZ deployment there are two static routes using [**priority setting**](https://community.fortinet.com/t5/FortiGate/Technical-Note-Routing-behavior-depending-on-distance-and/ta-p/198221) to bypass the reverse path filtering check when receiving data plane traffic over the GENEVE tunnels. The static routes are default routes to simplify the config, but you could also specify a route for each spoke VPC for each GENEVE tunnel. Also, there are two policy routes to hairpin traffic received over each GENEVE tunnel, back to the same one.

Graphical user interface

AI-generated content may be incorrect.

**Two-Arm Model**

In the two-arm model, the FortiGates will inspect traffic and forward & SNAT traffic out port1 (public interface) to act as a NAT GW. This removes the need for deploying NAT GWs in each AZ of each workload VPC. This is typically used in a centralized design where the data plane traffic used TGW to reach the GWLB endpoints in the inspection/security VPC and be inspected by the FortiGates. In summary centralized vs distributed designs is in reference to the GWLB endpoint placement which impacts how traffic is routed to these for inspection of traffic for different directions (ingress, egress, and east/west).

Diagram

AI-generated content may be incorrect.

We can use static and policy routes like below to support this setup. In a 2 AZ deployment there are two static routes using [**priority setting**](https://community.fortinet.com/t5/FortiGate/Technical-Note-Routing-behavior-depending-on-distance-and/ta-p/198221) to bypass the reverse path filtering check when receiving data plane traffic over the GENEVE tunnels. The static routes are default routes to simplify the config, but you could also specify a route for each spoke VPC for each GENEVE tunnel. Also, there are two policy routes to hairpin traffic received over each GENEVE tunnel, back to the same one. These policy routes will only hairpin traffic destined RFC1918 addresses and will route internet bound traffic out port1.

Graphical user interface, text, application

AI-generated content may be incorrect.

**Supporting Both Models**

In a single region, you can have one deployment of FGTs & GWLB support both distributed and centralized designs. This all comes down to implementing the appropriate routing at the VPC & TGW route tables and FortiGate routes. For examples on the VPC & TGW routes for different designs, reference [**common architecture patterns**](https://fortinetcloudcse.github.io/FortiCNF/2_moduletwo/23_awscommonarchitecturepatterns.html).

Here is an example of the static & policy routes to support a distributed spoke1 VPC (CIDR 10.1.0.0/16) and centralized spoke2 VPC.

Graphical user interface, text, application

AI-generated content may be incorrect.