Linux Academy

Hands-On Training

Configuring Proxy Protocol on an ELB and Nginx

Contents

Introduction

Elastic Load Balancers are an AWS service; these run overtop an operating system that runs the HAProxy software. Elastic Load Balancers work as a middleman for connections, intercepting and distributing connections among multiple EC2 instances. Elastic Load Balancers allow for high availability, fault-tolerant environments.

Proxy protocol enables the configuration of an additional header that contains client information on the remote machine accessing the Elastic Load Balancer. The web servers also need to be configured to accept the new header configurations.

In this lab, we learn how to set up proxy protocol on the Elastic Load Balancer, as well as Nginx.

Getting Started

Log into the **AWS console** with the given credentials, and navigate to your **EC2 instances**.

Copy the IP address of the web instance, and SSH into the machine with the user *linuxacademy* and the password *123456*. Remember, the SSH format looks like:

```
ssh user@ipaddress
```

Use `su` to log in as the root user. The password is, again, *123456*. Alternatively, this lab can be run as the *linuxacademy* user, with `sudo` prepended to the commands as needed.

Because we are configuring Nginx as part of this lab, we want to ensure that Nginx is running. To check, run:

```
systemctl status nginx
```

It should read that Nginx is **active (running)**. Additionally, if you navigate to your web instance's IP address in your browser, you should see the default Nginx installation page.

Now, should we review our Nginx access log:

```
tail -f /var/log/nginx/access.log
```

We can see that it is returning logs of the website being accessed by your IP address. This is because we hit the IP address of the web instance directly, so it did not pass through the load balancer.

If we, instead, navigate to the **Load Balancer** page in AWS and use the load balancer's *DNS name* to access the website, then view the tail output, you see that the traffic is reported from the load balancer IP, and not your own.

Configuring Command Line Interface

Additionally, we want to configure the AWS command line interface on our web instance server. Run:

```
aws configure
```

You can bypass the **AWS Access Key ID [None]:** and **AWS Secret Access Key [None]:** by pressing enter. The **Default region name [None]:** should be typed in as *us-east-1*. The **Default output format [None]:** can also remain as-is, by pressing enter a final time.

Configuring the ELB Using the AWS CLI

We are going to configure our Elastic Load Balancer to use proxy protocol as one of its policies. A policy is a configuration that is located on an individual ELB port.

To see what policies are available, run:

```
aws elb describe-load-balancer-policy-types
```

Here you see the **"PolicyTypeName": "ProxyProtocolPolicyType"** call.

Now we must create the policy itself. If you view the specific policy, you can see that it accepts an **AttrubuteName** and **AttributeType** (true or false since it is a boolean). We need to copy down our load balancer's name before continuing.

Back in the **AWS Console**, above the description, you can see your load balancer name. It should look something like *zrho-la-ElasticL-1N7ZC0FZ1OTI5*. Please note yours is different than the ones in this exercise's examples.

Creating the Policy

To create the policy, return to your command line, and enter:

```
aws elb create-load-balancer-policy --load-balancer-name oceg-la-ElasticL-1O51THARRZE40 --policy-name linuxacademy-protocol-policy --policy-type-name ProxyProtocolPolicyType --policy-attributes AttributeName=ProxyProtocol,AttributeValue=true
```

The **load-balancer-name** should be the unique name found on the Load Balancer page, whereas the **policy-name** can be anything; in this instance, we choose to call it *linuxacademy-protocol-policy*. The **policy-type-name** defines what type of policy we are making (*ProxyProtocolPolicyType*), while the **policy-attributes** are the attributes we need to input for the policy type. Please note all of this is case sensitive.

While the code above has created the policy, it did not assign it to the ELB. You should also note that any policies assigned to an instance port need to be included when this is assigned, or any policies not

included are removed.

To view the policies created, run:

```
aws elb describe-load-balancer-policies --load-balancer-name zrho-la-ElasticL-1N7ZC0FZ1OTI5
```

We only have the one we just input, and we now need to assign this to the instance port of the load balancer:

```
aws elb set-load-balancer-policies-for-backend-server --load-balancer-name oceg-la-ElasticL-1O51THARRZE4O --instance-port 80 --policy-names linuxacademy-protocol-policy
```

Remember to assign any policy you want on port 80 under policy-names, because any unassigned policies are removed.

To see if this is successful, we need to configure Nginx to use the proxy protocol header and insert the needed information into our log files, then test.

Configuring Nginx for Proxy Protocol

Navigate to the Nginx directory:

```
cd /etc/nginx
```

Open nginx.conf:

```
vim nginx.conf
```

Locate the section that denotes where the server is listening. It should look like:

```
server {
    listen       80 default_server;
    listen       [::]:80 default_server;
    server_name  _;
    root         /usr/share/nginx/html;
```

Change the **listen** values from *default_server* to *proxy_protocol*:

```
server {
    listen       80 proxy_protocol;
    listen       [::]:80 proxy-protocol;
    server_name  _;
```

```
    root        /usr/share/nginx/html;
```

Our ELB is making the connection to the backend servers, but with proxy protocol, we are allowing the parsing of different information. Because of this we need to define the **set_real_ip_from** to reference the ELB — in this instance, we use the *private CIDR block* for your virtual private cloud. Additionally, we need to set the **read_ip_header** to *proxy_protocol*.

```
server {
    listen       80 proxy_protocol;
    listen       [::]:80 proxy_protocol;
    set_real_ip_from 10.0.0.0/16;
    real_ip_header  proxy_protocol;
    server_name  _;
    root        /usr/share/nginx/html;
```

We now need to add this information to our access logs. We can alter this in the same file, in the **log_format** area. In this instance, we want to replace the *$remote_addr* variable with the *$proxy_protocol_addr* variable.

```
http {
    log_format  main  '$proxy_protocol_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';
```

Save and exit, then restart Nginx.

```
systemctl restart nginx
```

Testing

In your command line, run:

```
tail -f /var/log/nginx/access.log
```

Then return to the ELB's URL that you used before. Enter it in your browser, and view the updated access.log file in your terminal: You should see your IP address and not the ELB's.