**Course content**

**https://www.udemy.com/course/prometheus/**

1 section • 25 lectures • 3h 13m total length

Introduction25 lectures • 3hr 13min

* Install Prometheus

We will setup a dedicated Prometheus server.

Before you start, you will need a Linux server. Preferably an unrestricted Ubuntu 20.04 LTS Server with root access, since all the commands demonstrated in this course were executed on Ubuntu 20.04 LTS Server.

You can use other operating systems, such as Centos, but all commands in the course are prepared for Ubuntu 20, so you will experience some differences in syntax or equivalent commands which you may need to research yourself if I can't help you.

Once you have an Ubuntu 20.04 LTS server ready, you can start.

SSH onto your server, on windows I use Putty as my SSH client.

# sudo apt install prometheus

This will have installed 2 services being Prometheus and the Prometheus Node Exporter. You can verify there status using the commands. (Press Ctrl-C to exit the status log)

# sudo service prometheus status

# sudo service prometheus-node-exporter status

The install also created a user called Prometheus. You can see which processes it is running by using the command,

# ps -u prometheus

If Prometheus has started successfully, you can visit it at

You can visit it at http://[your ip address]:9090

Preview08:16

* Pointing A Domain Name

Note that this is optional, but it is useful if your Prometheus server is accessible from the internet, you want it to look more professional to clients and you want to have less problems sending emails from it.

I have gone onto my domain name provider, and added an A Name record that points to the IP address of my new Prometheus server.

Example,

prometheus.sbcode.net. IN A 134.209.224.39

Your domain and IP will be different, and note that it may take some time for the DNS record to propagate across the internet.

02:53

* Reverse Proxy Prometheus with Nginx

One option to help secure our Prometheus server is to put it behind a reverse proxy so that we can later add SSL and an Authentication layer over the default unrestricted Prometheus web interface.

We can use Nginx.

# sudo apt install nginx

CD to the Nginx *sites-enabled* folder

# cd /etc/nginx/sites-enabled

Create a new Nginx configuration for Prometheus

# sudo nano prometheus

And copy/paste the example below

-------------------------------

server {  
    listen 80;  
    listen [::]:80;  
    server\_name  YOUR-DOMAIN-NAME;  
  
    location / {  
        proxy\_pass           http://localhost:9090/;  
    }  
}

----------------------------

Save and test the new configuration has no errors

# nginx -t

Restart Nginx

# sudo service nginx restart

# sudo service nginx status

Test it by visiting again

http://**YOUR-DOMAIN-NAME**

03:36

* Add SSL to Prometheus Reverse Proxy

We will now add transport encryption to the Prometheus web user interface.

Since I have already set up the domain name, I can get a free certificate using Certbot.

Certbot will install a LetsEncrypt SSL certificate for free.

Ensure your domain name has propagated before running CertBot.

Your domain and IP will be different than mine, and note that it may take some time for the DNS record to propagate across the internet.

On my server, I will run

# sudo snap install --classic certbot

Now we can run CertBot.

# sudo certbot --nginx

Follow the prompts and select the domain name I want to secure.

Next open the Nginx Prometheus config file we created earlier to see the changes.

# sudo nano /etc/nginx/sites-enabled/prometheus

04:21

* Add Basic Authentication to the Prometheus User Interface

Everything is great so far, but anybody in the world with the internet access and the URL can visit my Prometheus server and see my data.

To solve this problem, we will add user authentication.

We will use Basic Authentication.

SSH onto your server and CD into your /etc/nginx folder.

# cd /etc/nginx

Then install apache2-utils (on ubuntu) or httpd-tools (on centos)

# //on ubuntu

# sudo apt install apache2-utils

# // on centos

# sudo yum install httpd-tools

Now we can create a password file. In the command below, I am creating a user called 'admin'.

# htpasswd -c /etc/nginx/.htpasswd admin

I then enter a password for the user.

Next open the Nginx Prometheus config file we created.

# sudo nano /etc/nginx/sites-enabled/prometheus

And add the two authentication properties in the examples below to the existing Nginx configuration file we have already created.

-------------------

server {  
    ...  
    #addition authentication properties  
    auth\_basic  "Protected Area";  
    auth\_basic\_user\_file /etc/nginx/.htpasswd;  
    location / {  
        proxy\_pass           http://localhost:9090/;  
    }  
    ...  
}

-------------------------------

Save and test the new configuration has no errors

# nginx -t

Restart Nginx

# sudo service nginx restart

# sudo service nginx status

06:01

* Scrape Target Basics

When you install Prometheus using

# apt install prometheus

It sets up two metrics endpoints.

* + Prometheus : http:127.0.0.1:9090/metrics
  + Node Exporter : http:127.0.0.1:9100/metrics

In this video, I show where the settings are configured for these metrics endpoints, how to enable them, change them and show some of the properties that can be retrieved in the graph expressions field.

11:38

* Install an External Node Exporter

Now we will install an external Prometheus Node Exporter on a different server.

# apt install prometheus-node-exporter

Now check the node exporter is running.

# sudo service node-exporter status

You can stop, start or restart a node exporter using

# sudo service node-exporter stop

# sudo service node-exporter start

# sudo service node-exporter restart

Node exporter will now be running on http://[your domain or ip]:9100/metrics

You can now block port 9100 externally, but leave it open internally for localhost.

And optionally, you can also allow a specific ip address or domain on the internet to access the port.

13:02

* Deleting a Time Series

There may be a time when you want to delete data from the Prometheus TSDB database.

Data will be automatically deleted after the storage retention time has passed. By default it is 15 days.

If you want to delete specific data earlier, then you are able.

You need to enable the admin api in Prometheus before you can.

# sudo nano /etc/default/prometheus

Add --web.enable-admin-api to the ARGS="" variable. eg,

ARGS="--web.enable-admin-api"

Restart Prometheus and check status

# sudo service prometheus restart

# sudo service prometheus status

You can now make calls to the admin api.

In my example I want to delete all time series for the **instance="sbcode.net:9100"**

So I run the **delete\_series** api endpoint providing the value to match. eg,

# curl -X POST -g 'http://localhost:9090/api/v1/admin/tsdb/delete\_series?match[]={instance="sbcode.net:9100"}'

When I re execute the Prometheus query, the time series I wanted deleted no longer exists.

04:47

* PromQL Example Queries

Now that we have at least 2 scape targets, we can begin to run some more interesting queries that involve multiple scrape targets.

We will try basic time series queries, queries with regular expressions, compare the Instant and Range Vector data types, use Functions, Aggregates and Sub Queries.

Preview14:06

* Recording Rules

We create some Recording rules, for the more complicated common queries we may want to create time series data for.

08:27

* Alerting Rules

Alerting rules are created in Prometheus very similar to how we created recording rules. We can use the same *prometheus\_rules.yml* or, if you wish, create a different file but remember to add the reference to it in the *rule\_files* section in *prometheus.yml.*

08:53

* Install the Prometheus Alert Manager

Install the Prometheus Alert Manager

# sudo apt install prometheus-alertmanager

It has started a new service called prometheus-alertmanager

# sudo service prometheus-alertmanager status

It is also managed by the user prometheus

# ps -u prometheus

Note that the service is running on port 9093

Visit http://[your domain name or ip]:9093/

06:10

* Install a Send Only SMTP Server for the Prometheus Alert Manager

We will configure the alert manager to use a local SMTP server for sending email alerts.

In this lecture, we create a very quick and simple local SMTP server which can only send emails from localhost.

The commands are

# sudo apt install mailutils

A postfix config screen should appear,

Choose **internet site** from the choices

enter the mail servers name, I will use my existing Prometheus domain i've already setup.

Then configure postfix by calling

# sudo nano /etc/postfix/main.cf

scroll to the bottom, and set

inet\_interfaces = loopback-only

inet\_protocols = ipv4

restart postfix

# sudo systemctl restart postfix

try sending an email from the command line,

# echo "This is the body" | mail -s "This is the subject" -a "FROM:root@your-domain.tld" your@email-address

06:08

* Configure Alert Manager and Promethues to Send Email Alerts

We now configure the Prometheus and Alert Manager processes to communicate with each other, and to send alerts when the alerting rules fire.

07:56

* Add the Prometheus Alert Manager UI

When installing the Prometheus Alertmanager using apt install prometheus-alertmanager, the version at the time of creating this document, didn't include a user interface. The message on the default UI says "The Debian package of the alertmanager does not include a web application."

The prebuilt binary version that you can download from GitHub does contain a UI.

In this lesson I will set up the prometheus-alertmanager from binary which can be downloaded from GitHub.

12:28

* Install Grafana

I set up a new minimum spec Ubuntu 20.04 LTS server for the purpose of demonstrating install Grafana.

Once you have connected to your new server, make sure your package lists are updated.

# sudo apt update

Then ensure that the dependencies for Grafana are installed.

# sudo apt-get install -y adduser libfontconfig1

Now to download the binary, and run the debian package manager.

# wget https://dl.grafana.com/oss/release/grafana\_7.2.0\_amd64.deb

# sudo dpkg -i grafana\_7.2.0\_amd64.deb

The install has now completed. You can now start the Grafana service

# sudo service grafana-server start

**Check the status**

# sudo service grafana-server status

Your Grafana server will be hosted at

http://[your Grafana server ip]:3000

The default Grafana login is

Username : \*\*admin\*\*

Password : \*\*admin\*\*

You have the option to update your password upon first login and then be presented with the option to add a new data source and create dashboards and users.

02:18

* Setup the Prometheus Datasource

I create a new Prometheus Datasource using the Grafana user interface.

I connect to my Prometheus url, which also uses SSL and has basic auth configured.

02:20

* Setup Prometheus Dashboards

Lets enable some of the default Dashboards provided with the Prometheus Data Source, and download one from the community specifically for the node exporters .

03:18

* SNMP in Prometheus

SNMP stands for Simple Network Management Protocol.

The main reason to use SNMP in Prometheus is in the situations were you cannot install a Node Exporter on the device you want to monitor, but the device can still support SNMP.

Common devices that support SNMP are routers, switches, printers, servers, workstations and other devices found on IP networks. Not every network device supports SNMP, or has it enabled, and there is a good chance you don't have an SNMP enabled device available on your network that you can use in this lesson. So we can set up a local SNMP daemon running on the server that we can test with.

Later on, we will install another SNMPD on an external server, and I will also demonstrate connecting to an existing CISCO switch.

05:47

* Install the SNMP Exporter

Now to manually install the SNMP Exporter.

I am going to manually install the latest version of the SNMP Exporter.

To see the latest versions, you can visit, https://github.com/prometheus/snmp\_exporter/releases

Select the file appropriate for your operating system.

18:46

* Configure a Second External SNMP Daemon

I will install an SNMPD on a different external server and configure the prometheus.yml with the new settings.

I do not need the SNMP tools (snnmpget, snmpwalk, etc) in this case, so I am only installing the bare minimum which is the SNMP Daemon.

09:34

* Setup SNMP for a CISCO Switch

I am now going to configure Prometheus to query SNMP from my Cisco Switch.

My Cisco Switch is a 24 Port Cisco Catalyst 2950.

After doing a factory reset of the Switch, It has

* + SNMP enabled
  + The default gateway set to the IP address of my main network router that is connected to the internet

On my internet firewall/router, I have created a new forwarding rule external port 161 to internal port 161 IP address 192.168.1.1

I then add my internet routers external IP address to the *prometheus.yml* configuration for the SNMP scrape target, and Prometheus is now able to query the SNMP properties of my switch through the SNMP exporter process.

I can verify that i am seeing data in Prometheus by querying the ifInOctets metric. This shows a new metric for every ethernet interface on my Switch.

Since I have Grafana installed from an earlier lesson. I will also use the SNMP Stats dashboard from the Grafana community.

This dashboard was written specifically for the *if\_mib* module that is configured in the *snmp.yml* and referenced in the *prometheus.yml* for the SNMP job used by the SNMP Exporter.

09:40

* SNMP Exporter Configuration Generator

We can generate custom modules that the SNMP Exporter can use.

We do this by generating a customised snmp.yml using a program called the SNMP Exporter Config Generator.

After generating a new snmp.yml, we can configure the existing SNMP Exporter to use the new customised snmp.yml.

Organise a dedicated server for this process so that you don't break your existing Prometheus server or any other production servers.

Note : Ensure that the version of the SNMP Exporter Config Generator matches the version of the SNMP Exporter that you are generating the snmp.yml for.

In my SNMP Exporter video, I installed version 0.19 so I need to make sure that the version of the SNMP Exporter Config Generator is from the same repository version.

09:27

* Generate HUAWEI SNMP Exporter Module

Now the we have the SNMP Exporter Configuration Generator setup, we can try and create custom modules to use in the SNMP Exporter.

For this example, I will create a module for a generic HUAWEI device.

I will need to download a HUAWEI MIB and research which OIDs I should walk/get.

12:02