**Hands-On Lab**

GitLab Pipelines (CI/CD)

**DevSecOps Theory to Practice**

**Contenido**

[1. Objetivo 3](#_Toc75297655)

[2. Instrucciones para levantar el entorno 3](#_Toc75297656)

[3. Codigo fuente de los laboratorios 5](#_Toc75297657)

[4. Ejercicio GitLab CI/CD Básico 5](#_Toc75297658)

# Objetivo

El objetivo de esta sección es indicar los pasos para llevar a cabo el Hands-on de GitLab en lo que respecta a un CI/CD básico, el objetivo es entender la sintaxis del archivo de configuración del pipeline y sus posibilidades.

# Instrucciones para levantar el entorno

Para iniciar el entorno deberás ingresar al portal de Microsoft Azure (<https://portal.azure.com>) con las credenciales que se te proporcionaron, tene en cuenta que la primera vez que inicies sesión se te solicitara un cambio de contraseña por razones de Seguridad.

Una vez dentro del portal veras una imagen como la siguiente:

A screenshot of a computer

Description automatically generated with medium confidence

Deberás ir al icono que se encuentra al lado de la barra de búsqueda:

A picture containing graphical user interface

Description automatically generated

Este icono te permite abrir una sesión de Cloud Shell, esto es una línea de comandos integrada 100 % en el navegador, lo que da como beneficio que no tengas que instalar absolutamente nada en tu computadora. Por ser la primera vez te va a aparecer una pantalla similar a la siguiente:

Graphical user interface, text, application

Description automatically generated

Le indican “Create Storage” y en cuestión de 2 a 3 minutos tendrán la consola lista para ejecutar comandos:

Text

Description automatically generated

Esta terminal la podrán iniciar las veces que sea necesario indicando el botón que se encuentra al lado del cuadro de búsqueda dentro del portal de Microsoft Azure.

# Codigo fuente de los laboratorios

Para facilitar la distribución de los laboratorios autoguiados los mismos han sido subidos a GitLab, con lo cual podrás clonarlos de ahí y hacerlos a medida que se vayan revisando a lo largo del curso. La URL del repositorio es:

<https://github.com/Cloud-Legion/DevSecOps-Theory-To-Practice.git>

La estructura es similar a la siguiente:

Graphical user interface, text, application, email

Description automatically generated

No te preocupes que los instructores te irán guiando con relación a que laboratorios hacer y estarán atentos a cualquier duda que tengas durante el desarrollo de los mismos😊

# Ejercicio GitLab CI/CD Básico

Llego la hora del ejercicio, habiendo hecho lo anterior estas listo para empezar a meter mano, para esto abrí una sesión de Cloud Shell y espera que se abra la ventana:

Graphical user interface, text, application

Description automatically generated

Dentro de la terminal ejecuta el siguiente comando para clonar el repositorio desde GitHub a tu espacio privado dentro de la terminal anteriormente configurada:

**git clone https://github.com/Cloud-Legion/DevSecOps-Theory-To-Practice.git**

Esto te lo descargara a tu espacio privado y con el comando **ls** podrás visualizar la carpeta de manera local:

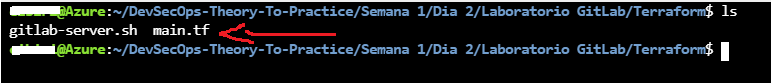
Text

Description automatically generated

Hecho esto debemos ingresar a la carpeta indicando la orden **cd DevSecOps-Theory-To-Practice**, podes tabular también en la primera palabra para que la terminal te lo autocomplete 😊, hecho eso veras algo como lo siguiente (que es lo mismo que verías en GitHub)



Deberás ir a la carpeta **/DevSecOps-Theory-To-Practice/Semana 1/Dia 2/Laboratorio GitLab/Terraform** con la orden **cd**, una vez ahí indica el comando ls para ver el contenido de la carpeta, deberías ver 2 archivos como los siguientes:

****

Estos son manifestos de Terraform y te permiten levantar cualquier tipo de infraestructura sin tener que configurar nada manualmente, para este caso puntual Terraform va a levantar una instancia 100 % configurada de GitLab CI / CD para que puedas desarrollar el laboratorio sin tener que hacer configuraciones de instalación, dentro de esta carpeta ejecuta el comando **terraform init**, al terminar la ejecucion deberías ver una pantalla como la siguiente indicando que se inicializo correctamente:

Text

Description automatically generated

Concluido esto ejecuta **terraform apply --auto-approve**, esto hará que se despliegue una máquina virtual dentro de tu misma suscripción de Azure con GitLab instalado, a esta maquina virtual vas a poder acceder a través de Internet con una IP publica provista automáticamente.

La provision de esta maquina virtual tardara de 5 a 7 minutos debido a configuraciones que deben realizarse, deberás ver una pantalla como la siguiente a lo largo del proceso:

Text

Description automatically generated

Terminada la ejecucion es probable que se sigan ejecutando tareas en segundo plano, pero en este punto deberías tener un output con la IP publica asignada:

Diagram

Description automatically generated with medium confidence

Esta IP publica luego de 5 minutos podrás usarla para iniciar sesión en GitLab a través de un navegador como Chrome, Firefox o Edge. Las credenciales para acceder son:

Usuario: **root**

Password: **Password1234!**

Graphical user interface, text, application

Description automatically generated

Una vez dentro veras una pantalla como la siguiente:

Graphical user interface, text, application, Teams

Description automatically generated

En este punto ya tenes el acceso para seguir desarrollando el laboratorio, lo próximo a realizar es entender cómo funciona la creación de pipelines en GitLab, para esto vamos a crear un proyecto nuevo dentro de GitLab llamado “Demo”

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Al crear el proyecto veras una pantalla como la siguiente:

Graphical user interface, text, application, email

Description automatically generated

GitLab utiliza para definir un pipeline un archivo llamado **.gitlab-ci.yml**, en él se indicarán todas las etapas e instrucciones para que el pipeline funcione y haga lo que nosotros queramos, este archivo tiene que estar definido en el root del repositorio. Vamos a crearlo en nuestro nuevo proyecto:

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Y vamos a copiar el contenido de este archivo:



Quedando algo como lo siguiente:

Graphical user interface, text, application

Description automatically generated

Al hacer commit de este archivo veras que se inicio el pipeline pero no avanza debido a que falta definir un Runner, esto es un componente que se instala en uno o varios Servidores y son los encargados de ejecutar los Jobs que nosotros indiquemos en el archivo **.gitlab-ci.yml**

Graphical user interface, application

Description automatically generated

Esto debido a que necesitamos un Runner, si vamos a la página de administración de GitLab dentro de Runner veremos que no tenemos ninguno:

Graphical user interface, text, application, email

Description automatically generated

Para registrar un runner debemos instalar un runner de acuerdo con el sistema operativo que estemos usando en el servidor destino para tal fin, revisar la siguiente documentación: <https://docs.gitlab.com/runner/install/index.html>, para el propósito de este laboratorio ya se encuentra instalado en el mismo Servidor de GitLab, solo debemos registrarlo, para eso iremos al Azure Cloud Shell y ejecutaremos el siguiente comando:

**ssh azureuser@xxx.xxx.xxx.xxx** (\*) Donde xxx.xxx.xxx.xxx es la IP Publica de tu GitLab.

Ingresar con los siguientes datos:

Usuario: **azureuser**

Password: **Password1234!**

**Text

Description automatically generated**

Ahora ejecutar la orden **sudo su** para poder tener derechos administrativos sobre la instancia. Hecho lo anterior ejecutaremos el comando **sudo gitlab-runner register** e indicaremos los datos que nos va solicitando, estos datos los obtendremos desde la pagina de runners de nuestro GitLab, de ahí necesitamos la URL y el Token.

**Graphical user interface, text, application

Description automatically generated**

Entonces con esto vamos completando los datos que nos pide el asistente:

**Text

Description automatically generated**

Hecho esto deberíamos ver el runner correctamente registrado dentro del GitLab en la sección de Runners:

**Graphical user interface, application

Description automatically generated**

Si vamos a la sección de pipelines del proyecto podemos ver que ahora si se ejecutó:

Graphical user interface, text, application, email

Description automatically generated

Si bien se ejecuto vemos que fallo esto fue deliberadamente a propósito, debido a que la etapa 2 tiene un exit code mayor a 0, lo cual hace que falle, si la removemos el job no falla.

Text

Description automatically generated

En este ejemplo podemos ver que estamos haciendo simplemente “echo” de archivos, pero en un escenario real (que mas adelante veremos) estos serian comandos reales que GitLab ejecutara por nosotros de manera automática.