

Introducción a Terraform y Salt Project

# Unidad 07. Terraform y LocalStack para simular Amazon Web Services

---



Autor: Sergi García

Actualizado Noviembre 2025

## Licencia




**Reconocimiento - No comercial - CompartirIgual (BY-NC-SA):** No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se ha de hacer con una licencia igual a la que regula la obra original.

## Nomenclatura







A lo largo de este tema se utilizarán diferentes símbolos para distinguir elementos importantes dentro del contenido. Estos símbolos son:

 **Importante**

 **Atención**

 **Interesante**

## ÍNDICE

1.  Introducción: simulando la nube en local	3
2.  ¿Qué servicios de AWS emula LocalStack?	3
3.  Instalación de LocalStack	4
4.  Integración de Terraform con LocalStack	5
5.  Beneficios prácticos de LocalStack para IaC y DevOps	6
6.  Webgrafía	6

## UNIDAD 07. TERRAFORM Y LOCALSTACK PARA SIMULAR AMAZON WEB SERVICES

### 1. INTRODUCCIÓN: SIMULANDO LA NUBE EN LOCAL

En entornos de aprendizaje o desarrollo, trabajar directamente con **Amazon Web Services (AWS)** puede resultar complejo o costoso:

- Es necesario registrar una cuenta y configurar credenciales.
- Algunos servicios tienen costes de uso reales.
- Los despliegues tardan más debido a la latencia de la nube.

Para solventar esto, surge **LocalStack**, una herramienta que permite **emular los servicios de AWS localmente** mediante contenedores Docker.


LocalStack actúa como un **entorno local compatible con las APIs oficiales de AWS**.

Esto significa que cualquier aplicación, script o código Terraform que funcione en AWS, también puede ejecutarse contra LocalStack **sin modificaciones**, simplemente redirigiendo el endpoint a <http://localhost:4566>.

### 2. ¿QUÉ SERVICIOS DE AWS EMULA LOCALSTACK?

LocalStack puede ejecutar decenas de servicios de AWS de forma simulada, entre los más comunes:

Categoría	Servicios disponibles
Almacenamiento	S3, EBS, SQS, SNS
Bases de datos	DynamoDB, RDS
Cómputo	Lambda, ECS, EC2
Infraestructura	CloudFormation, IAM, CloudWatch
API Gateway	REST y HTTP APIs simuladas
Otros	Secrets Manager, KMS, Step Functions

 En su versión **gratuita (Community)**, LocalStack incluye los servicios más usados como S3, Lambda y DynamoDB. La versión **Pro** amplía la compatibilidad con más de 60 servicios AWS.

Sus principales casos de uso son:

- Laboratorios educativos de Terraform y AWS.
- Testing de infraestructura como código (IaC) sin costes.
- Validación de plantillas antes de desplegarlas en producción.
- Ejecución de pipelines CI/CD en entornos sin conexión a Internet.

### 3. INSTALACIÓN DE LOCALSTACK

#### ♦ Requisitos previos

Asegúrate de tener instalados:

- **Python 3.8+** (para instalar la CLI de LocalStack)
- **Docker** (LocalStack usa contenedores)
- **Terraform** (para los despliegues IaC)

Verifica las instalaciones:

```
python3 --version
docker --version
terraform version
```

#### ♦ Instalación de LocalStack (vía pip)

LocalStack se instala fácilmente desde **pip**, el gestor de paquetes de Python:

```
pip install localstack
```

Para instalar la interfaz de línea de comandos adicional (localstack-cli), ejecuta:

```
pip install localstack-client
```

#### ♦ Arrancar LocalStack

Una vez instalado, inícialo con:

```
localstack start
```

Durante unos segundos verás cómo se inician los contenedores Docker. Cuando aparezca el mensaje: "Ready." LocalStack estará operativo y escuchando en el puerto **4566**, que es su gateway principal para todos los servicios AWS simulados.

Puedes comprobar el estado de los servicios con:

```
curl http://localhost:4566/health
```

Deberías obtener una salida como:

```
{
  "services": {
    "s3": "running",
    "dynamodb": "running",
    "lambda": "running"
  }
}
```

### ♦ (Opcional) Ejecutar LocalStack con Docker Compose

También puedes correr LocalStack directamente como contenedor sin instalarlo globalmente:

```
version: "3.8"
services:
  localstack:
    image: localstack/localstack:latest
    ports:
      - "4566:4566"
      - "8080:8080"
    environment:
      - SERVICES=s3,dynamodb,lambda
      - DEBUG=1
      - DATA_DIR=/tmp/localstack/data
    volumes:
      - "/var/run/docker.sock:/var/run/docker.sock"
```

Levántalo con:

```
docker compose up -d
```

Esto levantará LocalStack automáticamente con los servicios indicados.

## 4. INTEGRACIÓN DE TERRAFORM CON LOCALSTACK

Terraform se comunica con AWS mediante el provider hashicorp/aws. Este provider, por defecto, usa las APIs reales de AWS, pero podemos redirigir sus peticiones a LocalStack indicando un endpoint local (<http://localhost:4566>).

De esta forma, Terraform **crea que está desplegando en AWS**, pero en realidad está creando los recursos dentro de LocalStack, dentro de tu máquina local.






### Entendiendo el flujo de integración Terraform ↔ LocalStack

Cuando ejecutas “terraform apply” en este contexto, ocurre el siguiente flujo:

1. Terraform analiza los recursos definidos (aws\_s3\_bucket, aws\_s3\_object).
2. El **provider AWS** interpreta esas configuraciones.
3. En lugar de enviar las peticiones a las API reales de AWS, las redirige hacia **LocalStack**, que actúa como un servidor local en localhost:4566.
4. LocalStack, a su vez, ejecuta servicios simulados dentro de contenedores Docker (por ejemplo, un contenedor que simula S3).
5. Terraform recibe la respuesta de LocalStack como si proviniera de AWS real, y actualiza su archivo de estado (terraform.tfstate) con la información del recurso creado.

## 5. BENEFICIOS PRÁCTICOS DE LOCALSTACK PARA IAC Y DEVOPS

LocalStack no solo es útil para aprender Terraform, sino que también tiene **usos reales en entornos de desarrollo y testing**:

Beneficio	Descripción
 <b>Testing seguro y rápido</b>	Permite probar código Terraform sin afectar entornos reales.
 <b>Cero costes en la nube</b>	Ideal para formaciones o laboratorios sin cuentas AWS.
 <b>CI/CD local</b>	Se puede integrar en pipelines (GitHub Actions, Jenkins) para pruebas de despliegue.
 <b>Entornos aislados</b>	No requiere credenciales AWS; ideal para entornos con políticas de seguridad estrictas.
 <b>Compatibilidad 100% IaC</b>	Usa los mismos módulos y providers que AWS real.

## 6. WEBGRAFÍA

-  **Documentación oficial LocalStack**
  - Guía completa sobre instalación, configuración, servicios soportados y ejemplos de uso.
  - <https://docs.localstack.cloud>
-  **Terraform AWS Provider**
  - Detalle de todos los recursos, parámetros y ejemplos de configuración del provider de AWS.
  - <https://registry.terraform.io/providers/hashicorp/aws/latest/docs>
-  **Repositorio oficial Terraform + LocalStack**
  - Ejemplos prácticos del uso conjunto de Terraform con LocalStack, incluyendo múltiples servicios AWS simulados.
  - <https://github.com/localstack/terraform-localstack>