Pulumi

1. Pulumi là gì?

Pulumi là một công cụ "Infrastructure as Code" (IaC) hiện đại, cho phép bạn định nghĩa, triển khai và quản lý hạ tầng trên cloud (như AWS, Azure, Google Cloud, Kubernetes) bằng các ngôn ngữ lập trình phổ biến như C#, TypeScript, Python, và Go.

Thay vì phải học một ngôn ngữ chuyên biệt (như HCL của Terraform) hoặc viết file YAML/JSON dài dòng, bạn có thể sử dụng logic, vòng lặp, hàm, và các tính năng của ngôn ngữ lập trình bạn đã quen thuộc để xây dựng hạ tầng một cách linh hoạt và mạnh mẽ.

2. Cách hoạt động

- **Desired State (Trạng thái mong muốn):** Bạn viết code (ví dụ C#) để mô tả "trạng thái mong muốn" của hạ tầng. Ví dụ: "Tôi muốn có một EKS cluster với 3 node".
- **Pulumi Engine:** Khi bạn chạy một lệnh Pulumi, "Pulumi Engine" sẽ thực thi code của bạn để hiểu rõ trạng thái mong muốn.
- **Comparison (So sánh):** Pulumi so sánh trạng thái mong muốn này với trạng thái thực tế của hạ tầng đang chạy trên cloud. Nó biết được "trạng thái đã biết" thông qua một tệp trạng thái (state file) được lưu ở "backend".
- **Execution Plan (Kế hoạch thực thi):** Dựa trên sự khác biệt, Pulumi tạo ra một kế hoạch chi tiết: tài nguyên nào cần được **Tạo** (Create), **Cập nhật** (Update), hay **Xóa** (Delete).
- **Deployment (Triển khai):** Nếu bạn đồng ý với kế hoạch, Pulumi sẽ gọi các API cần thiết tới nhà cung cấp cloud (AWS, Azure...) để biến hạ tầng thực tế khớp với trạng thái bạn đã định nghĩa trong code.

3. Luồng làm việc (Workflow) cơ bản

Luồng làm việc của Pulumi rất đơn giản và an toàn:

1. **Viết Code:** Bạn định nghĩa các tài nguyên hạ tầng trong file `.cs` (với C#).
2. **Xem trước thay đổi (Preview):** Chạy lệnh `pulumi preview`. Pulumi sẽ cho bạn thấy một bản kế hoạch chi tiết về những gì sẽ thay đổi mà *không* thực sự

thay đổi gì cả. Đây là bước an toàn để kiểm tra lại.

3. **Triển khai (Deploy):** Chạy lệnh `pulumi up`. Pulumi sẽ hiển thị lại kế hoạch và yêu cầu xác nhận. Nếu bạn đồng ý (`yes`), nó sẽ tiến hành tạo/cập nhật hạ tầng.
4. **Hủy hạ tầng (Destroy):** Khi không cần nữa, chạy `pulumi destroy` để xóa tất cả các tài nguyên đã được tạo ra bởi Pulumi một cách an toàn.

4. Các lệnh Pulumi thường dùng

Ngoài 3 lệnh workflow chính ở trên, đây là các lệnh bạn sẽ sử dụng hàng ngày:

- **`pulumi new`**
  - **Công dụng:** Khởi tạo một dự án Pulumi mới từ một template (ví dụ: `aws-csharp`, `kubernetes-typescript`).
- **`pulumi stack ...`**
  - **Công dụng:** Quản lý các môi trường (stacks). Một "stack" thường tương ứng với một môi trường triển khai (ví dụ: `dev`, `staging`, `production`).
  - `pulumi stack init [tên-stack]`: Tạo một stack mới.
  - `pulumi stack ls`: Liệt kê tất cả các stack.
  - `pulumi stack select [tên-stack]`: Chuyển sang làm việc với một stack khác.
- **`pulumi config ...`**
  - **Công dụng:** Quản lý các biến cấu hình cho từng stack (ví dụ: số lượng node, kích thước máy ảo, API key...).
  - `pulumi config set [tên-biến] [giá-trị]`: Đặt một giá trị cấu hình.
  - `pulumi config set --secret [tên-biến] [giá-trị]`: Đặt một giá trị bí mật (như mật khẩu, key). Pulumi sẽ tự động mã hóa nó.
- **`pulumi login ...`**
  - **Công dụng:** Chỉ định nơi lưu "state file". Đây là lệnh quan trọng đầu tiên khi thiết lập.
  - `pulumi login`: (Mặc định) Đăng nhập vào Pulumi Cloud Service (khuyên dùng cho nhóm).
  - `pulumi login --local`: Lưu state file trên máy tính local (chỉ dùng để test).

    - `pulumi login s3://[tên-bucket]`: Tự host state file trên một cloud (ví dụ: AWS S3, Azure Blob, Google GCS).
- **`pulumi refresh`**
    - **Công dụng:** Đồng bộ state file với trạng thái thực tế trên cloud. Rất hữu ích khi ai đó thay đổi hạ tầng bằng tay (ví dụ: xóa tài nguyên trên giao diện AWS) và bạn muốn Pulumi "biết" về sự thay đổi đó.