Cloud Development

Best practices on developing in AWS.

Intro

Think of Amazon Web Services (AWS) like a massive digital warehouse full of building blocks for creating technology projects. Just like you might rent an apartment instead of buying a house, AWS lets you rent computing resources instead of buying physical servers and equipment. This means you can build websites, run applications, store data, and do complex calculations without needing to own any of the physical hardware

At Blueprint, we mainly AWS to run our staging environments, tho we also leverage some of it's other functionalities when we're working with various NPOs.

EC2

Imagine your laptop or desktop computer - it has a processor, memory (RAM), storage, and an operating system. An EC2 instance is very similar, but it exists in AWS's data centers instead of physically sitting on your desk. Here's how it relates to what you already know:

1. Virtual Machines vs EC2:

- If you've ever used VirtualBox or VMware on your computer to run a different operating system (like running Windows on a Mac), you're already familiar with virtual machines
- An EC2 instance is basically the same thing, but instead of running on your computer, it runs on Amazon's powerful servers
- Just like a VM on your computer, you can choose what operating system to run (Windows, Linux, etc.)
- You can connect to it remotely, just as if it were a computer sitting in front of you

The key differences from a regular VM on your computer:

- EC2 instances can be much more powerful than your personal computer
- You can access them from anywhere with an internet connection
- You only pay for what you use (if you turn it off, you don't pay)
- You can easily make it more powerful (add more CPU/RAM) without buying new hardware

Security Groups and SSH keys

Security Groups (The Firewall):

Think of a security group like a bouncer at a club who checks IDs. It controls what traffic can reach your EC2 instance and what traffic can leave it. For example:

- Want to allow SSH access (so you can connect to your instance)? You set a rule to allow traffic on port 22
- Need to host a website? Allow traffic on port 80 (HTTP) or 443 (HTTPS)
- By default, nothing is allowed in (inbound rules) - you have to explicitly permit it
- All traffic is allowed out (outbound rules) by default

SSH Keys (Your Digital Key):

Just like you need a physical key to enter your house, you need a digital key to connect securely to your EC2 instance.

Windows: https://www.purdue.edu/science/scienceit/ssh-keys-windows.html

Linux/macOS: open the terminal and type "ssh-keygen -t ed25519", follow the prompts and it will tell you where it saved the keys.

The file ending in .pub is the publickey, and the other one is the private key.

VPC

A VPC is like having your own private section of AWS's cloud, similar to having your own isolated network. Think of it like a university campus:

Main VPC Concepts:

- It's your private network in the cloud
- You control the IP address range (like 10.0.0.0/16)
- You can divide it into subnets (like different buildings on campus)
- You decide what can go in and out through Internet Gateways

Subnets:

- These are smaller sections of your VPC (like different floors in a building)
- Public Subnets: Can access the internet directly (like the university library that anyone can enter)
- Private Subnets: No direct internet access (like secure research labs that need special access)

Network Components:

- Internet Gateway: Your connection to the internet (like the university's main entrance)
- Route Tables: Directions for network traffic (like campus signs telling you how to get places)
- NAT Gateway: Lets private resources access the internet while staying private (like having a security guard fetch something from outside for you)