Hands-on Infrastructure Automation with Terraform on AWS

Alexander Savchuk

Building a multi-tier environment

Section 4

What we will build

- Network layer - Virtual Private Cloud (VPC), Internet Gateway, public and private subnets, NAT Gateway, and a bastion host)
- Relational Database Service (RDS) instance running PostgreSQL
- Elastic Container Service (ECS) cluster to host a dockerised app

What we will build

- **Network layer** - Virtual Private Cloud (VPC), Internet Gateway, public and private subnets, NAT Gateway, and a bastion host)
- Relational Database Service (RDS) instance running PostgreSQL
- Elastic Container Service (ECS) cluster to host a dockerised app

What we will build

- Network layer - Virtual Private Cloud (VPC), Internet Gateway, public and private subnets, NAT Gateway, and a bastion host)
- **Relational Database Service** (RDS) instance running PostgreSQL
- Elastic Container Service (ECS) cluster to host a dockerised app

What we will build

- Network layer - Virtual Private Cloud (VPC), Internet Gateway, public and private subnets, NAT Gateway, and a bastion host)
- Relational Database Service (RDS) instance running PostgreSQL
- **Elastic Container Service** (ECS) cluster to host a dockerised app

Section overview

- Count meta-parameter and splat expressions
- Output variables
- Data sources
- Templates
- Resource graph

Section overview

- **Count meta-parameter and splat expressions**
- Output variables
- Data sources
- Templates
- Resource graph

Section overview

- Count meta-parameter and splat expressions
- **Output variables**
- Data sources
- Templates
- Resource graph

Section overview

- Count meta-parameter and splat expressions
- Output variables
- **Data sources**
- Templates
- Resource graph

Section overview

- Count meta-parameter and splat expressions
- Output variables
- Data sources
- **Templates**
- Resource graph

Section overview

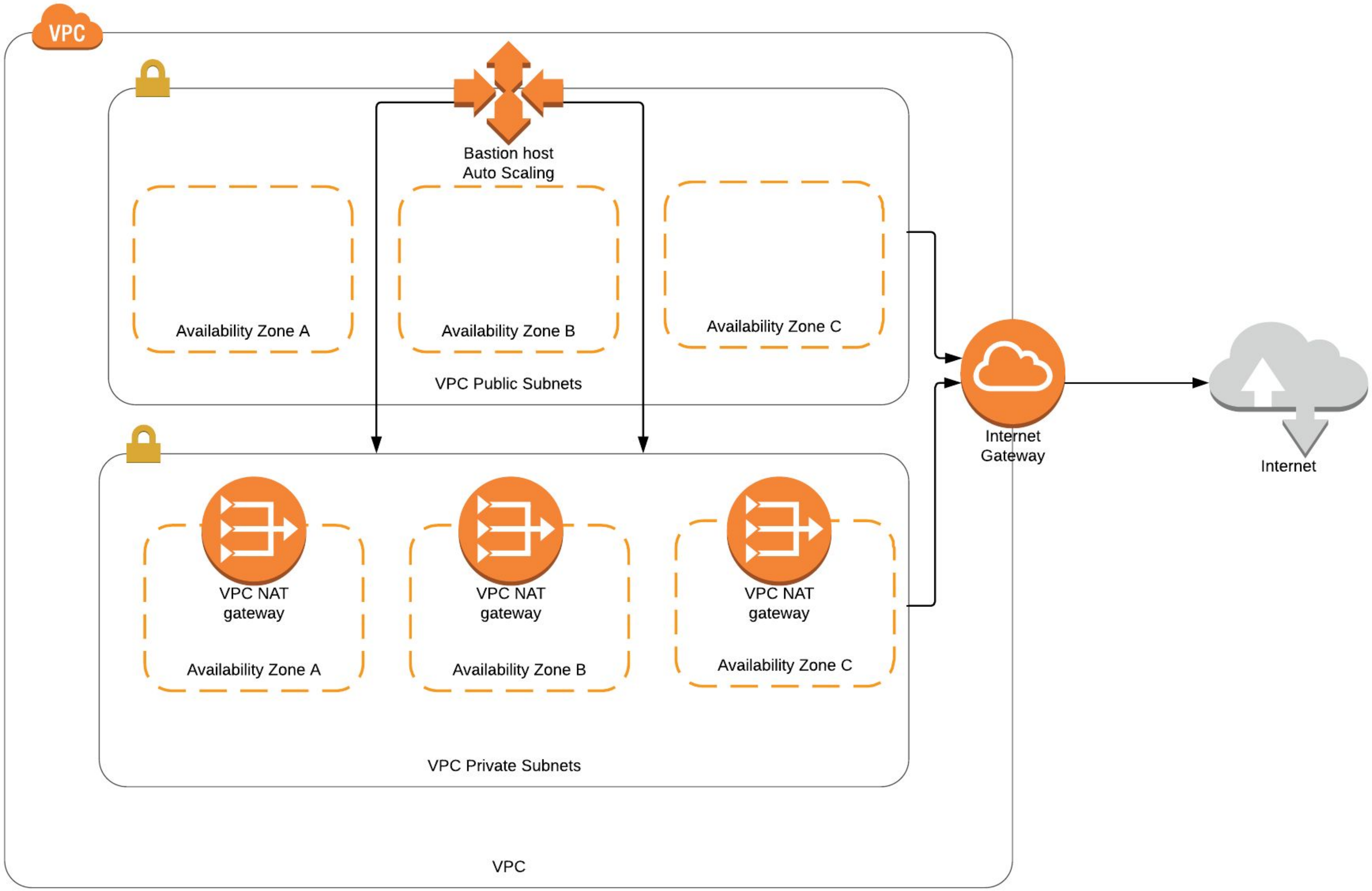
- Count meta-parameter and splat expressions
- Output variables
- Data sources
- Templates
- **Resource graph**

Basic understanding of AWS is helpful, but not required

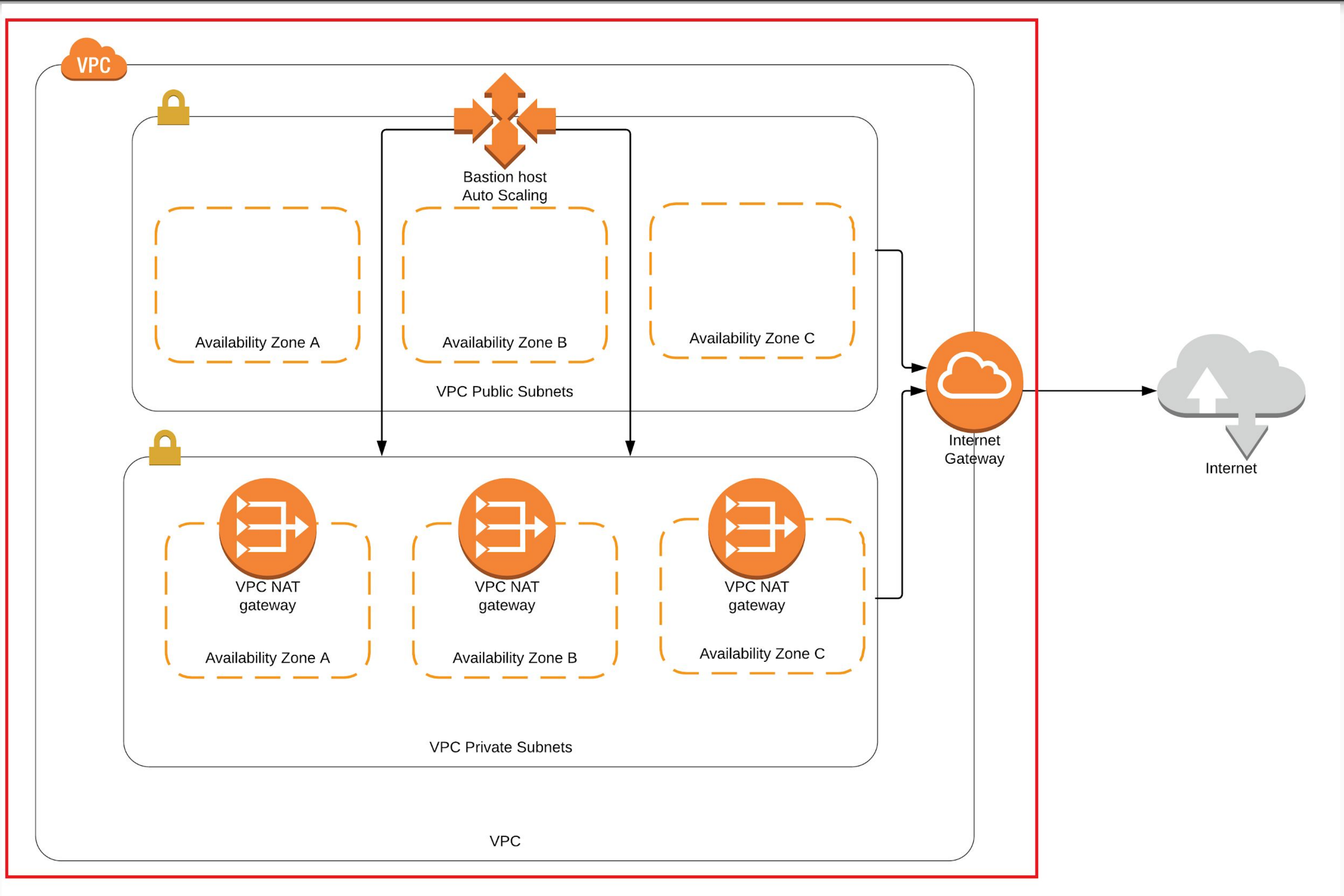
Some components will cost money

Starting to build a new environment

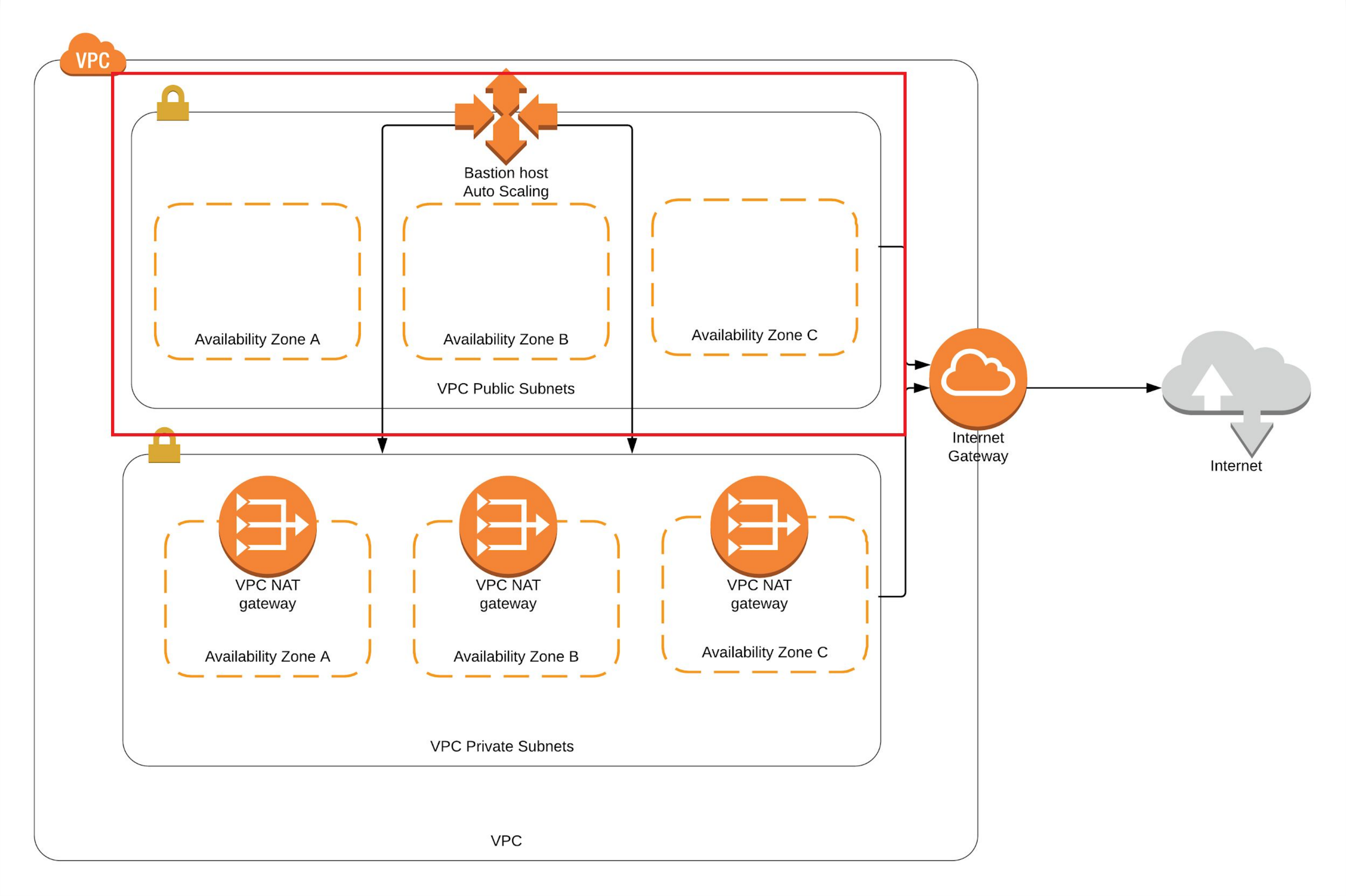
A whirlwind tour of the Virtual Private Cloud



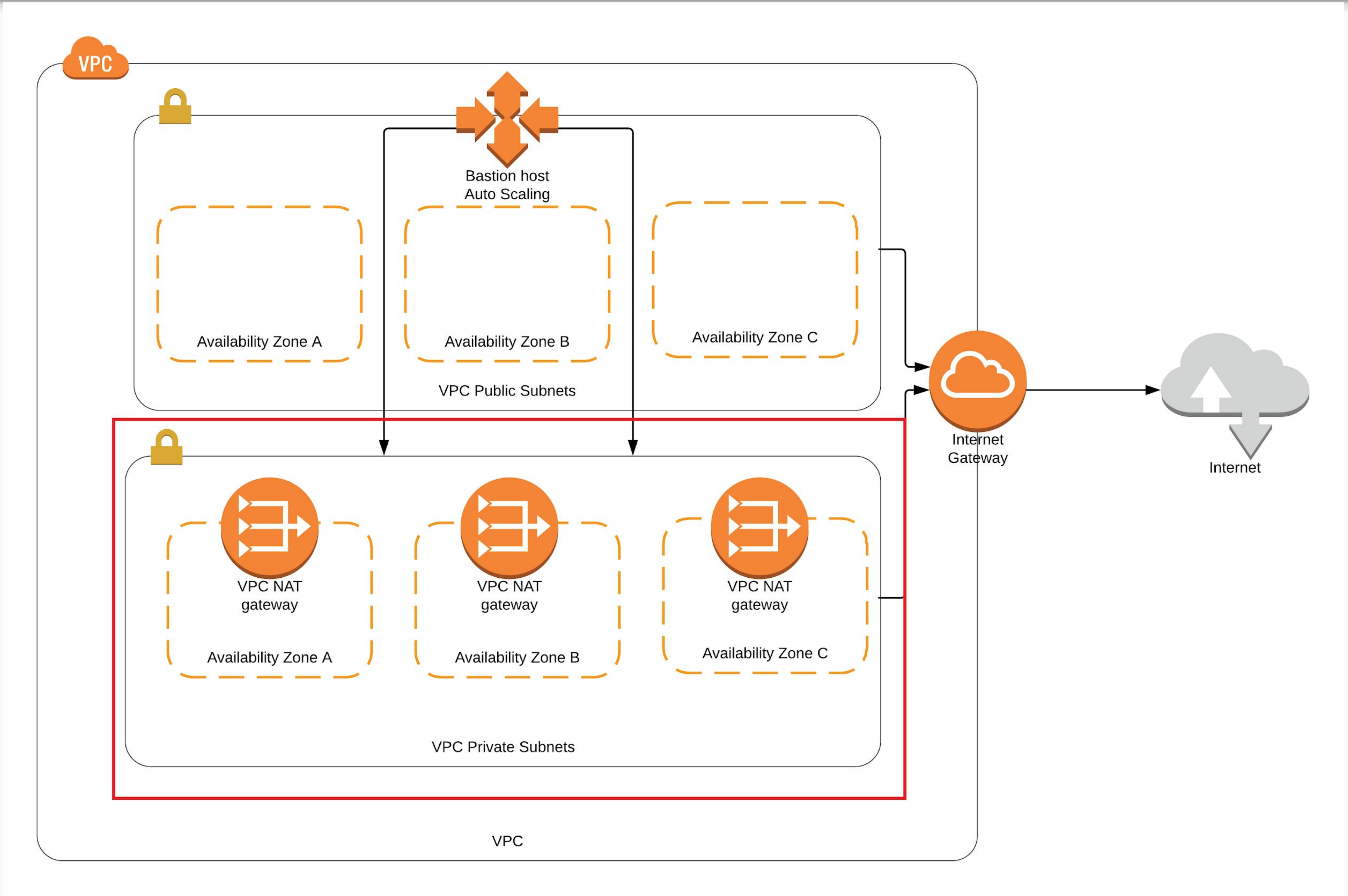
A whirlwind tour of the Virtual Private Cloud



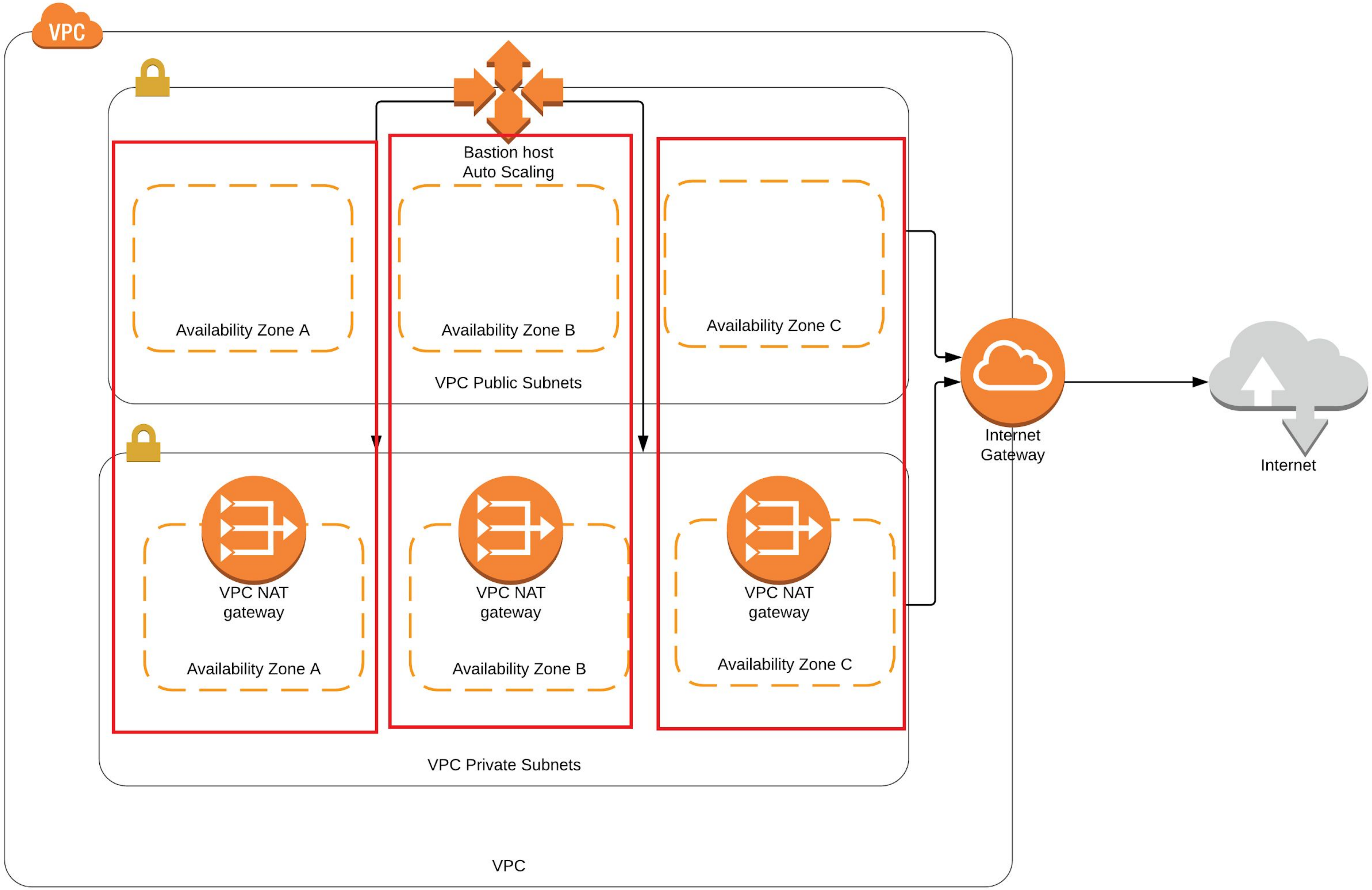
A whirlwind tour of the Virtual Private Cloud



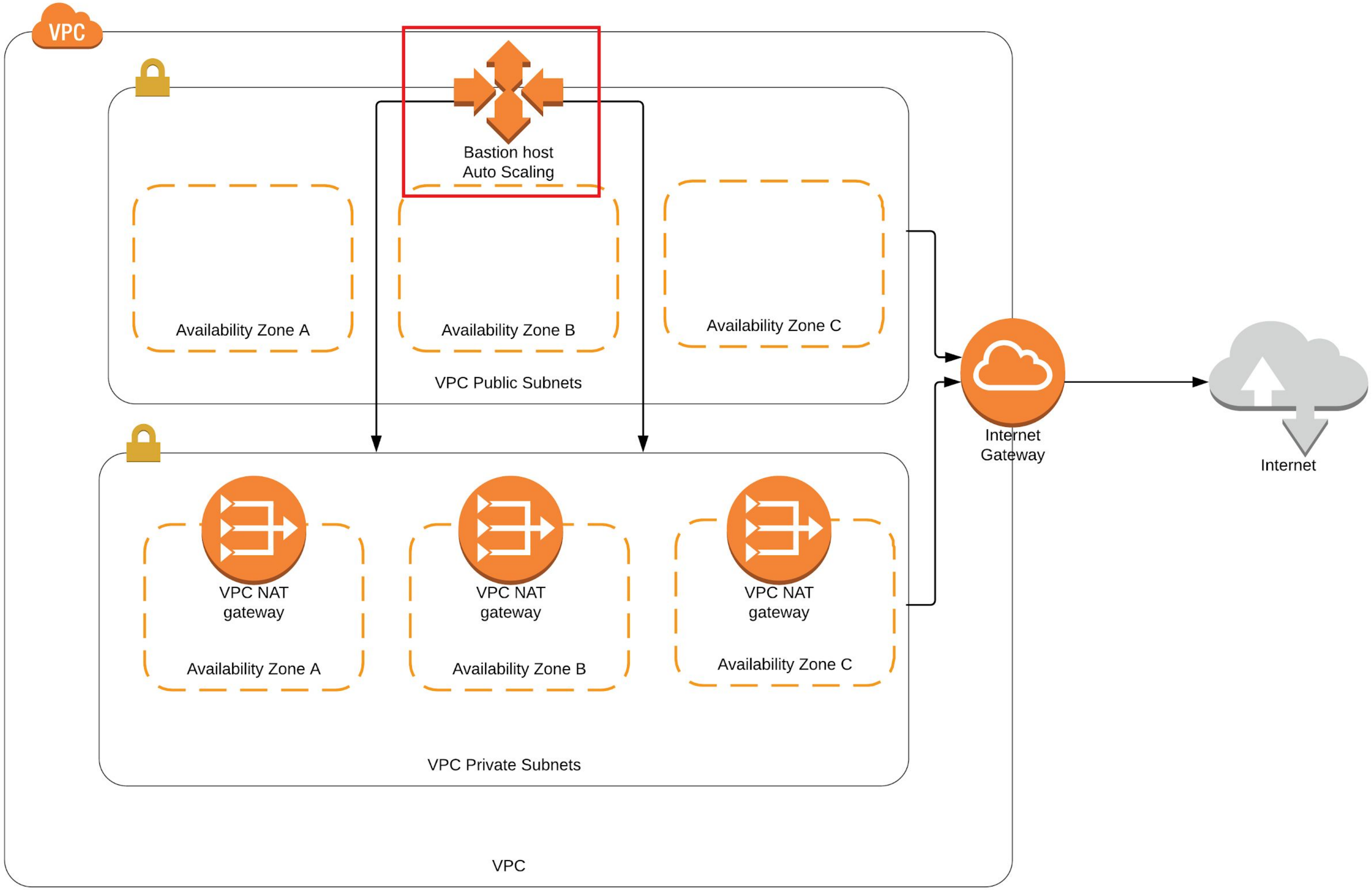
A whirlwind tour of the Virtual Private Cloud



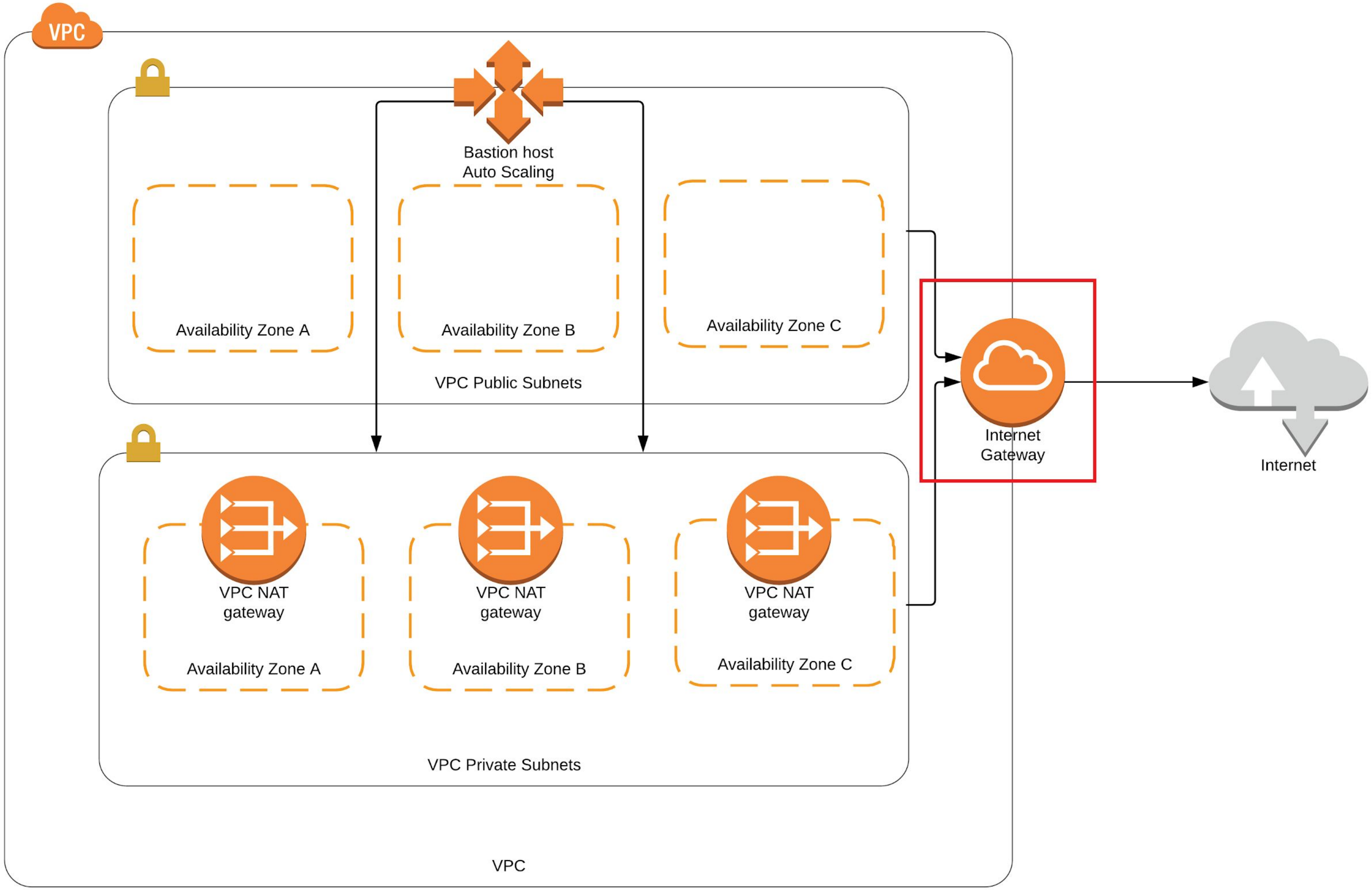
A whirlwind tour of the Virtual Private Cloud



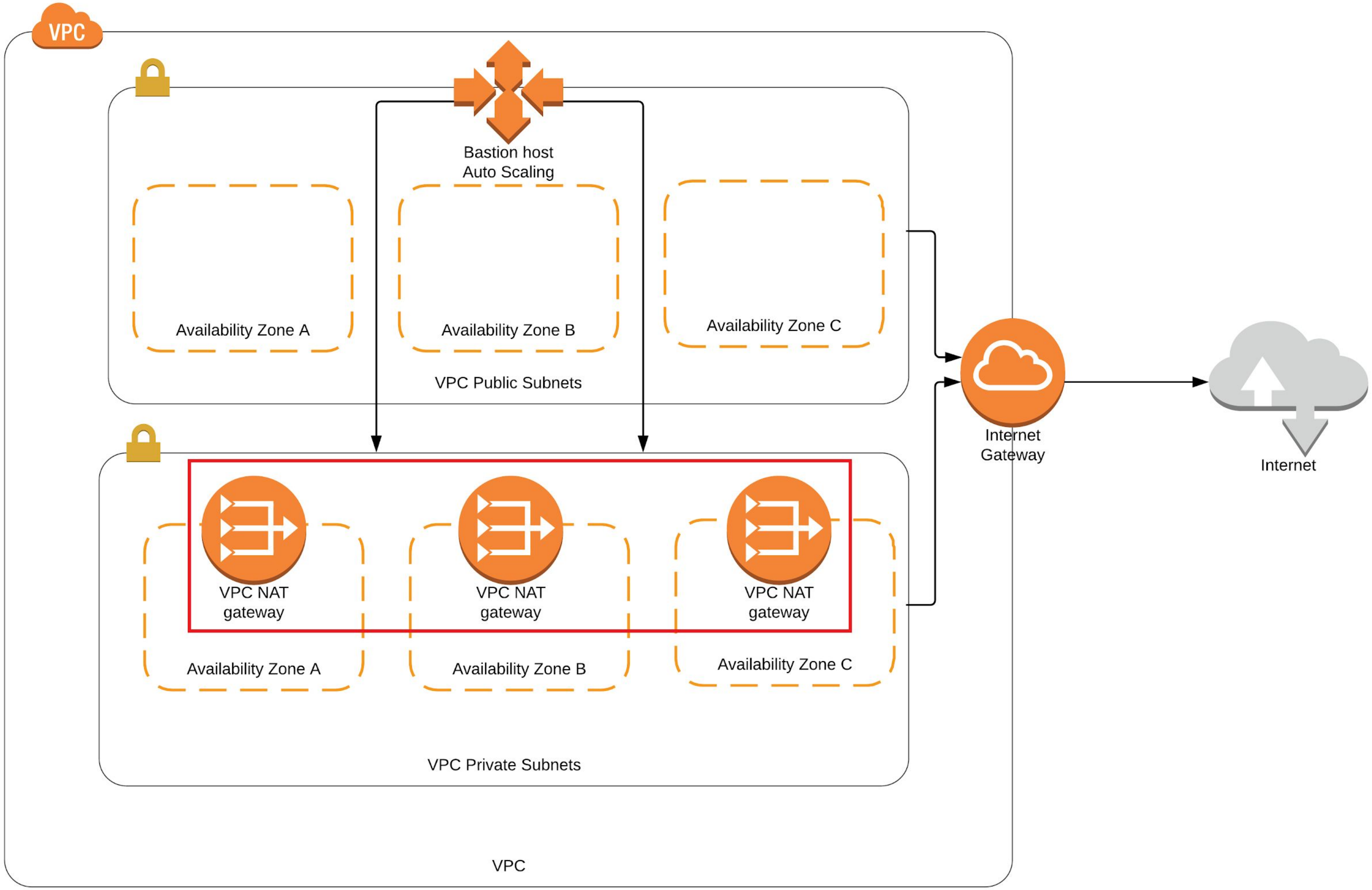
A whirlwind tour of the Virtual Private Cloud



A whirlwind tour of the Virtual Private Cloud



A whirlwind tour of the Virtual Private Cloud



Why a new project?

To keep the state files separate

- Keep resources with similar life cycles together
- Separate resources with different life cycles
- Manage risk of making changes to the configuration

Provider caching

- **terraform init** downloads providers separately for each project
- We can cache them by setting an environment variable

```
export TF_PLUGIN_CACHE_DIR="$HOME/.terraform.d/plugin-cache"
```

Count meta-parameter

- Available to all resources
- Allows creating multiple copies of a resource without repeating its configuration
- Helps keep your infrastructure code **DRY**

Splat expression

```
resource "aws_nat_gateway" "main" {
  count         = "${length(var.availability_zones)}"
  subnet_id     = "${element(aws_subnet.public.*.id, count.index)}"
  allocation_id = "${element(aws_eip.nat.*.id, count.index)}"
}
```

Organising data with output variables

Next video