Learn AMAZON WEB SERVICES IN A MONTH OF LUNCHES

SAMPLE CHAPTER

DAVID CLINTON

MANNING

Learn Amazon Web Services in a Month of Lunches

by David Clinton

Chapter 13

brief contents

13 Keeping ahead of user demand

This is a bit of a transition chapter. Chapter 12 completed our survey of AWS's core deployment services. That means you're now familiar with the following:

- EC2 instances, Amazon Machine Images (AMIs), and the peripheral tools that support their deployment, such as security groups and EBS volumes
- Incorporating databases into applications, both on-instance and through the managed RDS service
- Using S3 buckets to deliver media files through your EC2 applications and for server backup storage
- Controlling access to your AWS resources with Identity and Access Management (IAM)
- Managing growing resource sets by intelligently applying tags
- Accessing resources using either the browser interface or the AWS command-line interface (CLI)

All of these things are represented in the schematic shown in figure 13.1. That alone would easily justify the time and energy you've invested in this book. Now we're going to shift our focus and explore some best practices for application optimization.

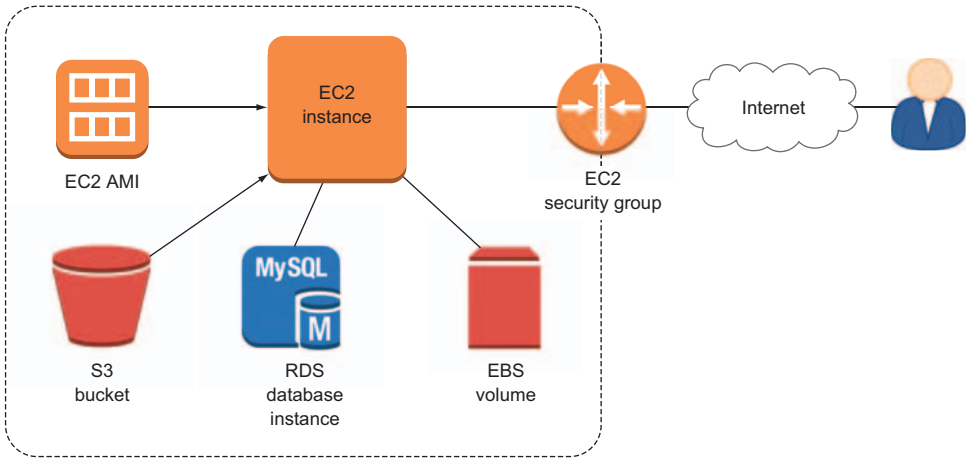

Figure 13.1 This is the kind of application infrastructure you should be able to build on your own, having read the first dozen chapters of this book.

But things are running nicely—who needs optimization? Well, as customer demand on your WordPress site continues to grow, you'll care, and in a big way. You see, for some reason—perhaps related to the fact that you discount the price of your product by 75% for half an hour each evening—most customers arrive in the early evening, local time. The single server you've been running is largely unused most of the day, but it melts under the pressure of thousands of visits squeezed into such a short stretch of time.

And then there's a question one of the guys in the office asked the other day: "Our entire business is running on a single web server. What happens if it goes down?" What indeed.

You could provision four or five extra servers and run them full time. That way, you'd be covered for the high-volume periods and for the failure of any one server. But that approach would involve colossal waste, because for much of each day you'd be paying for most of the instances to sit idle. Nor would it necessarily be much help in the event of a *network* failure, which would likely cut connectivity to *all* the servers at the same time.

You could address the customer demand issue by arranging for someone to be at the office every evening to manually fire up as many extra servers as needed; but you asked around, and no one volunteered. And

besides, the best way to ensure that a daily job won't get done is to assume that an admin will remember to do it.

13.1 Automating high availability

Alternatively, you could spend some time incorporating high availability capability into your setup and let software quietly and efficiently manage things. This will be the subject of the next few chapters. You'll learn to use AWS's geographically remote availability zones to make total application failure much less likely; load balancing to coordinate between parallel servers and monitor their health; and auto scaling to let AWS automatically respond to the peaks and valleys of changing demand by launching and shutting down instances according to need.

> **NOTE** *High availability* is any server resource configuration that allows a system to remain functioning and accessible for as close to 100% of the time as possible. The basic goal can be achieved through combinations that can include redundancy, replication, failover protocols, monitoring, and load balancing.

Figure 13.2 will help you visualize how all that infrastructure can be made highly available through the magic of network segmenting, auto scaling, and load balancing. Although you probably aren't familiar with many of the tools and relationships represented in the diagram, you should make a mental note of at least a few key points:

1. A virtual private cloud (VPC) encompasses all the AWS resources in your application deployment.
2. There are two kinds of subnets—private and public—that can be located in separate availability zones and are used to manage and, where needed, isolate resources.
3. Security-group rules control the movement of data between resources.
4. The EC2 AMI acts as a template for replicating precise OS environments.
5. The S3 bucket can store and deliver data, both for backup and for delivery to users.
6. The EBS volumes act as data volumes (like hard drives) for an instance.

7 The auto scaler permits automatic provisioning of more (or fewer) instances to meet changing demands on an application.
8 The load balancer routes traffic among multiple servers to ensure the smoothest and most efficient user experience.

As you've probably noticed, the *E* in many AWS service names (EC2, ECS, EFS, EMR, and so on) doesn't stand for *electronic* the way it does in the names of some older technologies, like *email*; rather, it stands for *elastic*. You can be excused for wondering just what about the AWS vision of cloud computing is so elastic.

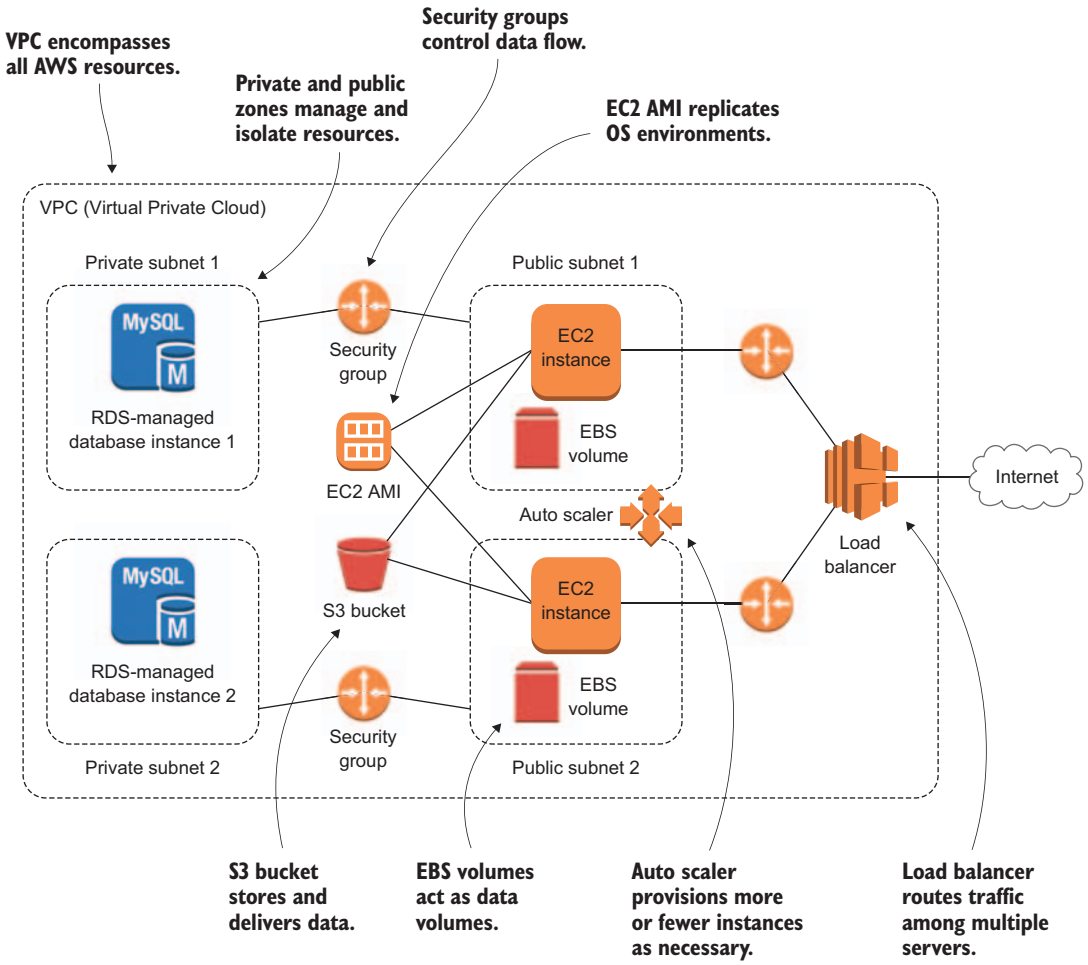

Figure 13.2 An illustration of how AWS data and security services work together to allow an EC2 instance to deliver its application

But before I answer that question, it may be useful to talk about cloud computing in general. Understanding what makes the cloud unique is essential for taking full advantage of all that it has to offer.

13.2 Cloud computing

The U.S. National Institute of Standards and Technology (NIST) defines *cloud computing* as services that offer users all of these five qualities:

- *On-demand self-service*—Customers can access public cloud resources whenever needed and without having to order them through a human representative.
- *Broad network access*—Cloud resources are accessible from any network-connected (that is, internet) location.
- *Resource pooling*—Cloud providers offer a multitenant model, whereby individual customers can safely share resources with each other; and dynamic resource assignment, through which resources can be allocated and deallocated according to customer demand.
- *Rapid elasticity*—Resource availability and performance can be automatically increased or decreased to meet changing customer demand.
- *Measured service*—Customers can consume services at varying levels through a single billing period and are charged only for those resources they actually use.

These five qualities describe a deeply flexible, highly automated system whose elements can be freely mixed and matched to provide the efficient, cost-effective service. But a great deal of what makes this possible is the existence of integrated systems that can dynamically adjust themselves based on what's going on around them. These adjustments are examples of elastic behavior.

13.3 Elasticity vs. scalability

Elasticity is a system's ability to monitor user demand and automatically increase and decrease deployed resources accordingly. *Scalability*, by contrast, is a system's ability to monitor user demand and automatically increase and decrease … wait, didn't I just say that about elasticity?

It's complicated. The two terms are sometimes used interchangeably, but I think it's worthwhile distinguishing between them. Bear in mind

that the way I explain the relationship between these two ideas is by no means the last word on the subject—look around, and you'll find some other approaches. But in the context of understanding how AWS works, my spin should be useful.

What makes an elastic band *elastic* is partly its ability to stretch under pressure, but also the way it quickly returns to its original size when the pressure is released. In AWS terms, that would mean the way, for instance, EC2 makes instances available to you when needed but lets you drop them when they're not, and charges you only for uptime (see figure 13.3).

Scalability describes the way a system is *designed* to meet changing demand. That might include the fact that you have 24-hour access to any resources you might need (which, of course, is an elastic feature), but it also means the *underlying design* supports rapid, unpredictable changes. As an example, software that's scalable can be easily picked up and dropped onto a new server—possibly in a new network environment—and run without any manual configuration. Similarly, as shown in figure 13.4, the composition of a scalable infrastructure can be quickly changed in a way that all the old bits and pieces immediately know how to work together with the new ones.

With that in mind, we can say that Amazon's EC2 is not only elastic but, because its elements—instances, storage volumes, security groups, and

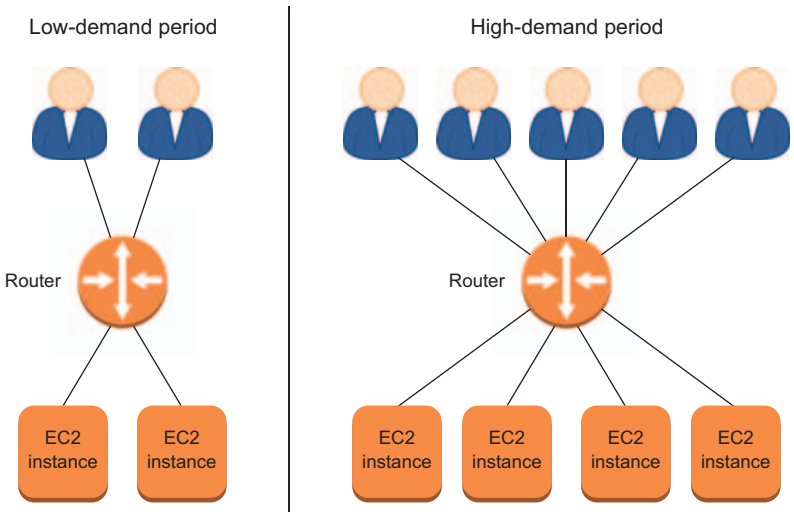

Figure 13.3 Elasticity allows for systems to dynamically add or remove resources to meet changing demand.

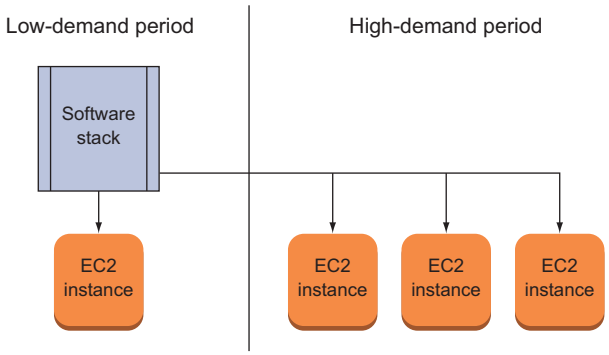

Figure 13.4 Scalable software can be easily copied for use in multiple servers deployed in multiple network environments.

so on—can be smoothly dropped into and out of running infrastructures, also very scalable. Ah, but what *kind* of scalable? There are two:

- *Horizontal scaling* is *scaling out*: you add more lightweight server nodes (or *instances*) to meet growing demand.
- *Vertical scaling* is *scaling up*: you move your application from a single lightweight server to one with greater compute capacity.

It's certainly possible to transfer AWS-based applications from lighter to heavier servers, and for some payloads—like many high-load transaction databases—it's preferred. But in an AWS context, if you hear a conjugation of the word *scale*, the odds are that it's referring to horizontal scaling.

If you want a more reliable, responsive, public-facing application (and who doesn't?), you should definitely stay tuned as we work through the book's second part, which focuses on optimizing your existing infrastructure.

Definitions

- *Cloud computing*—Networked services offering self-service, resource pooling, elasticity, and metered billing
- *Elasticity*—The ability to increase and decrease available compute resources to meet changing demands
- *Scalability*—The ability of software or infrastructure to adapt to changes in service volume
- *Scaling out*—The addition of new server nodes to handle increased demand (horizontal scaling)
- *Scaling up*—The adoption of a more powerful server node to handle increased demand (vertical scaling)

SOFTWARE ENGINEERING/CLOUD

Learn AMAZON WEB SERVICES IN A MONTH OF LUNCHES

DAVID CLINTON

Cloud computing has transformed the way we build and deliver software. With the Amazon Web Services cloud platform, you can trade expensive glass room hardware and custom infrastructure for virtual servers and easy-to-configure storage, security, and networking services. Better, because you don't own the hardware, you only pay for the computing power you need! Just learn a few key ideas and techniques, and you can have applications up and running in AWS in minutes.

Learn Amazon Web Services in a Month of Lunches gets you started with AWS fast. In just 21 bite-size lessons, you'll learn the concepts and practical techniques you need to deploy and manage applications. You'll learn by doing real-world labs that guide you from the core AWS tool set through setting up security and storage and planning for growth. You'll even deploy a public-facing application that's highly available, scalable, and load balanced.

WHAT'S INSIDE

- First steps with AWS—no experience required
- Deploy web apps using EC2, RDS, S3, and Route 53
- Cheap and fast system backups
- Setting up cloud automation

If you know your way around Windows or Linux and have a basic idea of how web applications work, you're ready to start using AWS.

David Clinton is a system administrator and Linux server professional with 15 years of teaching experience.

"An excellent book for those who have often thought about learning AWS but were overwhelmed by options and where to start."

—Conor Redmond
InComm Product Control

"An easy read and straight to the point. A great foundation for every AWS user."

—Alexey Galiullin, GlobalOrange

"A must-read for anyone who wants a robust and practical start with AWS."

—Peter Perlepes, Growth

"A friendly tutorial on what's essential to know in today's cloud/DevOps world."

—Miguel Paraz, Odecee

To download their free eBook in PDF, ePub, and Kindle formats, owners of this book should visit www.manning.com/books/learn-amazon-web-services-in-a-month-of-lunches

ISBN-13: 978-1-61729-444-0
ISBN-10: 1-61729-444-6
53999
9 781617 294440

MANNING $39.99 / Can $52.99 [INCLUDING eBOOK]