2 A simple example: WordPress in 15 minutes

This chapter covers

- Creating a cloud infrastructure for WordPress
- Exploring a cloud infrastructure consisting of a load balancer, virtual machines, a database, and a network filesystem
- Estimating costs of a cloud infrastructure
- Shutting down a cloud infrastructure

Having looked at why AWS is such a great choice to run web applications in the cloud, in this chapter, you'll explore migrating a simple web application to AWS by setting up a sample cloud infrastructure within 15 minutes. Over the course of the book, we will revisit the WordPress example to understand the concepts in more detail. For example, we will take a look at the relational database in chapter 10 and learn how to add and remove virtual machines based on the current load in chapter 17.

NOTE The example in this chapter is totally covered by the Free Tier (see section 1.4.1 for details). As long as you don't run this example longer than a few days, you won't pay anything for it. Keep in mind that this applies only if you created a fresh AWS account for this book and there is nothing else going on in your AWS account. Try to complete the chapter within a few days, because you'll clean up your account at the end of the chapter.

Imagine you work for a mid-sized company that runs a blog to attract new software and operations engineers and uses WordPress as the content management system. Around 1,000 people visit the blog daily. You are paying $250 per month for the on-premises infrastructure. This seems expensive to you, particularly because at the moment, the blog suffers from several outages per month.

To leave a good impression on potential candidates, the infrastructure should be highly available, which is defined as an uptime of 99.99%. Therefore, you are evaluating new options to operate WordPress reliably.

AWS seems to be a good fit. As a proof of concept, you want to evaluate whether a migration is possible, so you need to do the following:

- Set up a highly available infrastructure for WordPress.
- Estimate the monthly costs of the infrastructure.
- Come to a decision and delete the infrastructure afterward.

WordPress is written in PHP and uses a MySQL database to store data. Besides that, data like user uploads is stored on disk. Apache is used as the web server to serve the pages. With this information in mind, it's time to map your requirements to AWS services.

.1 Creating your infrastructure

You'll use the following five AWS services to copy the old infrastructure to AWS:

- *Elastic Load Balancing (ELB)*—AWS offers a load balancer as a service. The load balancer distributes traffic to a bunch of virtual machines and is highly available by default. Requests are routed to virtual machines as long as their health check succeeds. You'll use the Application Load Balancer (ALB), which operates on layer 7 (HTTP and HTTPS).
- *Elastic Compute Cloud (EC2)*—The EC2 service provides virtual machines. You'll use a Linux machine with an optimized distribution called Amazon Linux to install Apache, PHP, and WordPress. You aren't limited to Amazon Linux; you could also choose Ubuntu, Debian, Red Hat, or Windows. Virtual machines can fail, so you need at least two of them. The load balancer will distribute the traffic between them. In case a virtual machine fails, the load balancer will stop sending traffic to the failed VM, and the remaining VM will need to handle all requests until the failed VM is replaced.
- *Relational Database Service (RDS) for MySQL*—WordPress relies on the popular MySQL database. AWS provides MySQL with its RDS. You choose the database size (storage, CPU, RAM), and RDS takes over operating tasks like creating backups and installing patches and updates. RDS can also provide a highly available MySQL database using replication.

- *Elastic File System (EFS)*—WordPress itself consists of PHP and other application files. User uploads—for example, images added to an article—are stored as files as well. By using a network filesystem, your virtual machines can access these files. EFS provides a scalable, highly available, and durable network filesystem using the NFSv4.1 protocol.
- *Security groups*—Control incoming and outgoing traffic to your virtual machine, your database, or your load balancer with a firewall. For example, use a security group allowing incoming HTTP traffic from the internet to port 80 of the load balancer. Or restrict network access to your database on port 3306 to the virtual machines running your web servers.

Figure 2.1 shows all the parts of the infrastructure in action. Sounds like a lot of stuff to set up, so let's get started!

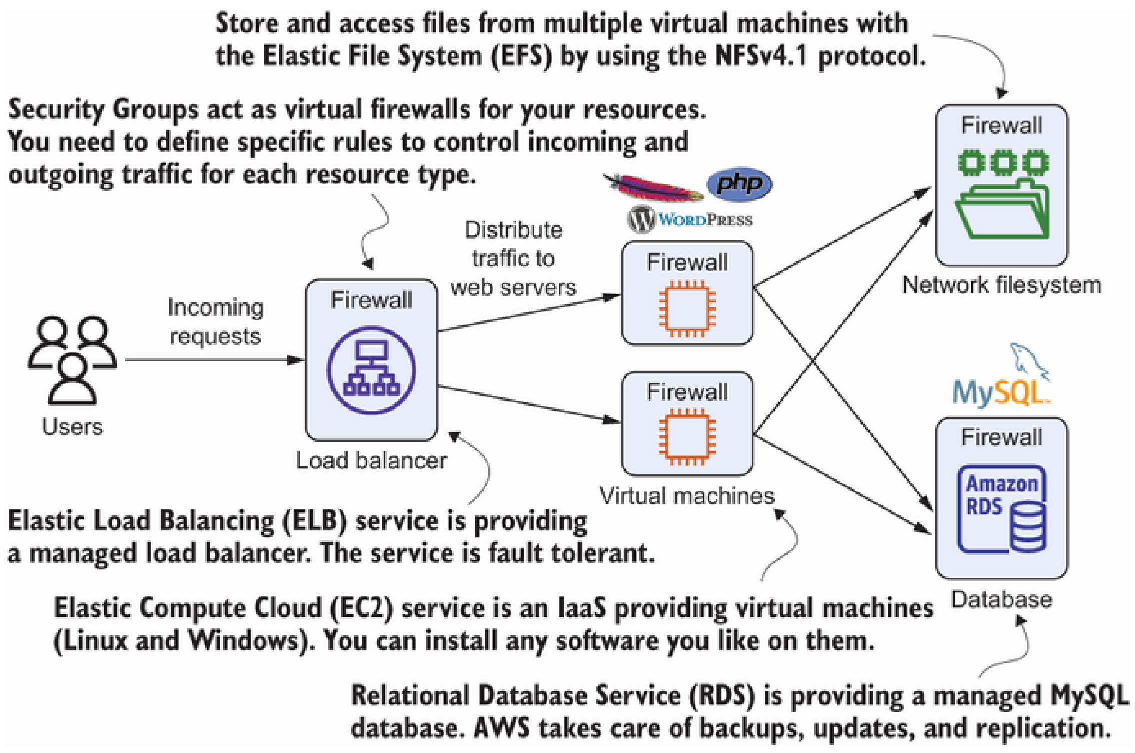

Store and access files from multiple virtual machines with the Elastic File System (EFS) by using the NFSv4.1 protocol.
Security Groups act as virtual firewalls for your resources. You need to define specific rules to control incoming and outgoing traffic for each resource type.
Firewall
Network filesystem
Distribute traffic to web servers
Firewall
Incoming requests
Firewall
Users
Load balancer
Firewall
Virtual machines
MySQL
Firewall
Amazon RDS
Database
Elastic Load Balancing (ELB) service is providing a managed load balancer. The service is fault tolerant.
Elastic Compute Cloud (EC2) service is an IaaS providing virtual machines (Linux and Windows). You can install any software you like on them.
Relational Database Service (RDS) is providing a managed MySQL database. AWS takes care of backups, updates, and replication.

Figure 2.1 The company's blogging infrastructure consists of two load-balanced web servers running WordPress, a network filesystem, and a MySQL database server.

If you expect pages of instructions, you'll be happy to know that you can create all of this with just a few clicks using a service called AWS CloudFormation, which you will learn about in detail in chapter 4. AWS CloudFormation does all of the following automatically in the background:

1. Creates a load balancer (ELB)
2. Creates a MySQL database (RDS)

3. Creates a network filesystem (EFS)
4. Creates and attaches firewall rules (security groups)
5. Creates two virtual machines running web servers:
    1. Creates two virtual machines (EC2)
    2. Mounts the network filesystem (EFS)
    3. Installs Apache and PHP
    4. Downloads and extracts the 4.8 release of WordPress
    5. Configures WordPress to use the created MySQL database (RDS)
    6. Starts the Apache web server

To create the infrastructure for your proof of concept, open the AWS Management Console at **https://console.aws.amazon.com**. Click Services in the navigation bar, and select CloudFormation. You can use the search function to find CloudFormation more easily.

DEFAULT REGION FOR EXAMPLES All examples in this book use N. Virginia (also called us-east-1) as the default region. Exceptions are indicated. Please make sure you switch to the region N. Virginia before starting to work on an example. When using the AWS Management Console, you can check and switch the region on the right side of the main navigation bar at the top.

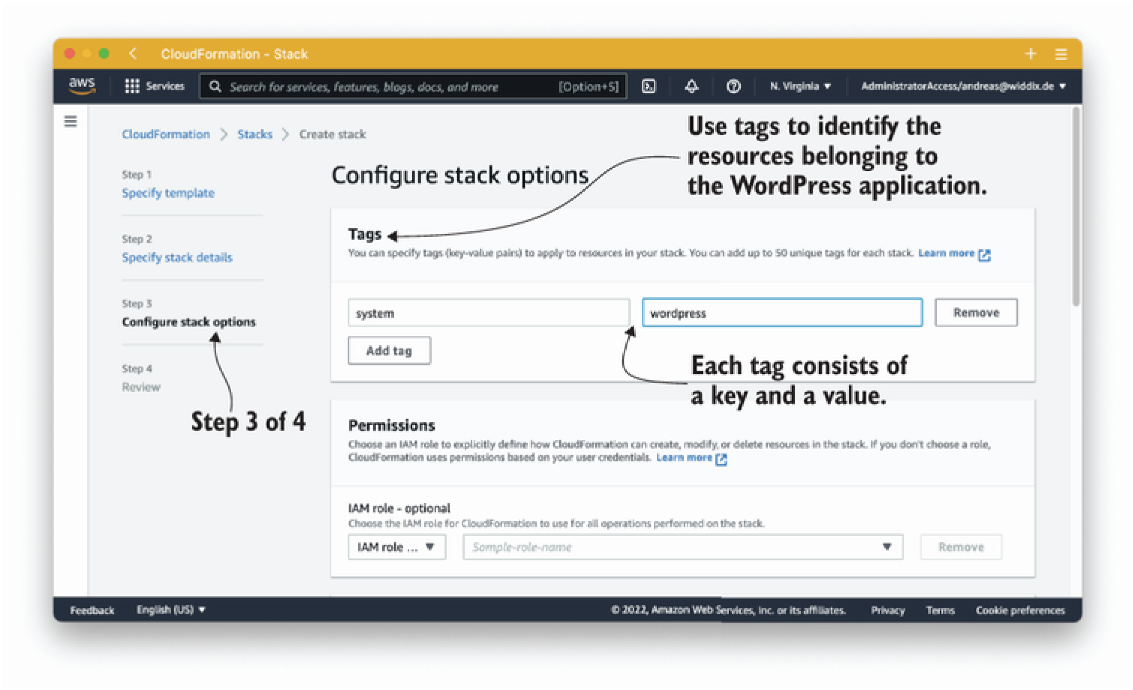
Figure 2.2 Creating the stack for the proof of concept: Step 1 of 4

Click Create Stack, and choose With New Resources (Standard) to start the four-step wizard, as shown in figure 2.2. Choose Template Is Ready, and enter the Amazon S3 URL **https://s3.amazonaws.com/awsinaction-code3/chapter02/template.yaml** to use the template prepared for this chapter. Proceed with the next step of the wizard.

Specify wordpress as the Stack name and, in the Parameters section, set the WordpressAdminPassword to a password of your choice that you are not using somewhere else, as shown in figure 2.3.

Figure 2.3 Creating the stack for the proof of concept: Step 2 of 4

The next step is to specify tags for your infrastructure, as illustrated in figure 2.4. A *tag* consists of a key and a value and can be used to add metadata to all parts of your infrastructure. You can use tags to differentiate between testing and production resources, add a cost center to easily track costs in your organization, or mark resources that belong to a certain application if you host multiple applications in the same AWS account.



You can specify tags (key-value pairs) to apply to resources in your stack. You can add up to 50 unique tags for each stack. Learn more

system | wordpress

Choose an IAM role to explicitly define how CloudFormation can create, modify, or delete resources in the stack. If you don't choose a role, CloudFormation uses permissions based on your user credentials. Learn more

Choose the IAM role for CloudFormation to use for all operations performed on the stack.

Figure 2.4 Creating the stack for the proof of concept: Step 3 of 4

Figure 2.4 shows how to configure the tag. In this example, you'll use a tag to mark all resources that belong to the wordpress system. This will help you to easily find all the parts of your infrastructure later. Use a custom tag consisting of the key—system—and the value—wordpress. Afterward, press the Next button to proceed to the next step.

You can define your own tags as long as the key name is less than 128 characters and the value is less than 256 characters.

Additional CloudFormation stack options

It is possible to define specific permissions used to manage resources, as well as to set up notifications and other advanced options. You won't need these options for 99% of the use cases, so we don't cover them in our book. Have a look at the CloudFormation User Guide (http://mng.bz/deqv) if you're interested in the details.

Figure 2.5 illustrates that all you need to do is to acknowledge that CloudFormation will create IAM resources and click Create Stack.

Figure 2.5 Creating the stack for the proof of concept: Step 4 of 4

Your infrastructure will now be created. Figure 2.6 shows that wordpress is in the state of `CREATE_IN_PROGRESS`. It’s a good time to take a break; come back in 15 minutes, and you’ll be surprised.

Figure 2.6 CloudFormation is creating the resources needed for WordPress.

After all the needed resources have been created, the status will change to `CREATE _COMPLETE`. Be patient and click the refresh button from time to time if your status continues to show `CREATE_IN_PROGRESS`.

Select the check box at the beginning of the row containing your word-press stack. Switch to the Outputs tab, as shown in figure 2.7. There you’ll find the URL to your WordPress installation; click the link to open it in your browser.

Figure 2.7 The blogging infrastructure has been created successfully.

You may ask yourself, how does this work? The answer is *automation*.

Automation references

One of the key concepts of AWS is automation. You can automate everything. In the background, your cloud infrastructure was created based on a blueprint. You'll learn more about blueprints and the concept of programming your infrastructure in chapter 4. You'll learn to automate the deployment of software in chapter 15.

You'll explore the blogging infrastructure in the next section to get a better understanding of the services you're using.

2.2 Exploring your infrastructure

Now that you've created your blogging infrastructure, let's take a closer look at it. Your infrastructure consists of the following:

- Web servers running on virtual machines
- Load balancer
- MySQL database
- Network filesystem

2.2.1 Virtual machines

First, use the navigation bar to open the EC2 service as shown in figure 2.8. Next, select Instances from the subnavigation options. A list showing two virtual machines named wordpress shows up. When you select one of those instances, details about the virtual machine appear below.

Figure 2.8 The virtual machines are running a web server to deliver WordPress.

You're now looking at the details of your virtual machine, also called an EC2 instance. Some interesting details follow:

- *Instance ID*—The ID of the virtual machine.
- *Instance type*—The size of the virtual machine (CPU and memory).
- *IPv4 Public IP*—The IP address that is reachable over the internet.
- *AMI ID*—Remember that you used the Amazon Linux OS. If you click the AMI ID, you'll see the version number of the OS, among other things.

Select the Monitoring tab to see how your virtual machine is used. This tab is essential if you really want to know how your infrastructure is doing. AWS collects some metrics and shows them here. For example, if the CPU is using more than 80% of its capacity, it might be a good time to add another virtual machine to prevent increasing response times. You will learn more about monitoring virtual machines in section 3.2.

2.2.2 Load balancer

Next, have a look at the load balancer, shown in figure 2.9, which is part of the EC2 service as well. There is no need to switch to a different service in the Management Console—just click Load Balancer in the subnavigation options on the left-hand side of the screen. Select your load balancer from the list to show more details. Your *internet-facing* load balancer is accessible from the internet via an automatically generated DNS name.

Figure 2.9 Get to know the load balancer.

The load balancer forwards an incoming request to one of your virtual machines. A target group is used to define the targets for a load balancer. You'll find your target group after switching to Target Groups through the subnavigation options of the EC2 service, as shown in figure 2.10.

Figure 2.10 Details of target group belonging to the load balancer

The load balancer performs health checks to ensure requests are routed only to healthy targets. Two virtual machines are listed as targets for the target group. As you can see in the figure, the status of both virtual machines is healthy.

As before, on the Monitoring tab you can find interesting metrics that you should watch in production. If the traffic pattern changes suddenly, this indicates a potential problem with your system. You'll also find metrics indicating the number of HTTP errors, which will help you to monitor and debug your system.

2.2.3 MySQL database

The MySQL database is an important part of your infrastructure; you'll look at it next. To do so, open the Relational Database Service (RDS) via the main navigation. Afterward, select Databases from the subnavigation options. Select the database using the engine called MySQL community, as illustrated in figure 2.11.

Figure 2.11 Finding the MySQL database

WordPress requires a MySQL database, so you have launched a database instance with the MySQL engine as noted in figure 2.12. Your blog receives a low amount of traffic, so the database doesn't need to be very powerful. A small instance class with a single virtual CPU and 1 GB memory is sufficient. Instead of using SSD storage, you are using magnetic disks, which is cheaper and sufficient for a web application with around 1,000 visitors per day.

Figure 2.12 Details of the MySQL database storing data for the blogging infrastructure

As you'll see in chapter 10, other database engines, such as PostgreSQL or Oracle Database, are available, as well as more powerful instance classes, offering up to 96 cores with 768 GB memory, for example.

Common web applications use a database to store and query data. That is true for WordPress as well. The content management system (CMS) stores blog posts, comments, and more within a MySQL database.

WordPress also stores data outside the database on disk. For example, if an author uploads an image for their blog post, the file is stored on disk. The same is true when you are installing plug-ins and themes as an administrator.

2.2.4 Network filesystem

The Elastic File System (EFS) is used to store files and access them from multiple virtual machines. EFS is a storage service accessible through the

NFS protocol. To keep things simple, all files that belong to WordPress, including PHP, HTML, CSS, and PNG files, are stored on EFS so they can be accessed from all virtual machines.

Open the EFS service via the main navigation as shown in figure 2.13. Next, select the filesystem whose name starts with “wordpress”.

Figure 2.13 The NFS used to store the WordPress application and user uploads

Figure 2.14 shows the details of the filesystem. For example, the throughput mode *bursting* is used, to allow high I/O throughput for small periods of time during the day at low cost.

Figure 2.14 The details of the EFS

To mount the Elastic File System from a virtual machine, mount targets are needed. You should use two mount targets for fault tolerance. The network filesystem is accessible using a DNS name for the virtual machines.

Now it’s time to evaluate costs. You’ll analyze the costs of your blogging infrastructure in the next section.

2.3 How much does it cost?

Part of evaluating AWS is estimating costs. We recommend using the AWS Pricing Calculator to do so. We created a cost estimation including the load balancer, the virtual machines, the database, and the network file system. Go to **http://mng.bz/VyEW** to check out the results, which are also shown in figure 2.15. We estimate costs of about $75 for hosting WordPress on AWS in a highly available manner, which means all components are distributed among at least two different data centers. Don’t worry—the example in this chapter is covered by the Free Tier. Our cost estimation does not consider the Free Tier, because it applies for only the first 12 months.

Figure 2.15 Blogging infrastructure cost calculation

Table 2.1 summarizes the results from the AWS Pricing Calculator.

Table 2.1 More detailed cost calculation for blogging infrastructure

AWS service	Infrastructure	Pricing	Monthly cost
EC2	Virtual machines	2 * 730 hours * $0.0116 (t2.micro)	$16.94
EC2	Storage	2 * 8 GB * $0.10 per month	$1.60
Application Load Balancer	Load balancer	730 hours * $0.0225 (load bal-ancer hour) 200 GB per month	$18.03
Application Load Balancer	Outgoing traffic	100 GB * $0.00 (first 100 GB) 100 GB * $0.09 (up to 10 TB)	$9.00
RDS	MySQL database instance (primary + standby)	732.5 hours * $0.017 * 2	$24.82
RDS	Storage	5 GB * $0.115 * 2	$2.30
EFS	Storage	5 GB * $0.3	$1.50
			$74.19

Keep in mind that this is only an estimate. You're billed based on actual use at the end of the month. Everything is on demand and usually billed by seconds or gigabyte of usage. The following factors might influence how much you actually use this infrastructure:

- *Traffic processed by the load balancer*—Expect costs to go down in December and in the summer when people are on vacation and not looking at your blog.
- *Storage needed for the database*—If your company increases the amount of content in your blog, the database will grow, so the cost of storage will increase.
- *Storage needed on the NFS*—User uploads, plug-ins, and themes increase the amount of storage needed on the NFS, which will also increase the cost.
- *Number of virtual machines needed*—Virtual machines are billed by seconds of usage. If two virtual machines aren't enough to handle all the traffic during the day, you may need a third machine. In that case, you'll consume more seconds of virtual machines.

Estimating the cost of your infrastructure is a complicated task, but that is also true if your infrastructure doesn't run in AWS. The benefit of using AWS is that it's flexible. If your estimated number of virtual machines is too high, you can get rid of a machine and stop paying for it. You will learn more about the pricing model of the different AWS services throughout the course of this book.

You have completed the proof of concept for migrating your company's blog to AWS. It's time to shut down the infrastructure and complete your migration evaluation.

.4 Deleting your infrastructure

Your evaluation has confirmed that you can migrate the infrastructure needed for the company's blog to AWS from a technical standpoint. You have estimated that a load balancer, virtual machines, and a MySQL database, as well as a NFS capable of serving 1,000 people visiting the blog per day, will cost you around $75 per month on AWS. That is all you need to come to a decision.

Because the infrastructure does not contain any important data and you have finished your evaluation, you can delete all the resources and stop paying for them.

Go to the CloudFormation service in the Management Console, and take the following steps, as shown in figure 2.16:

1. Select your wordpress stack.
2. Click the Delete button.

Figure 2.16 Deleting your blogging infrastructure

After you confirm the deletion of the infrastructure, as shown in figure 2.17, it takes a few minutes for AWS to delete all of the infrastructure's dependencies.

Figure 2.17 Confirming deletion of your blogging infrastructure

This is an efficient way to manage your infrastructure. Just as the infrastructure's creation was automated, its deletion is also. You can create and delete infrastructure on demand whenever you like. You pay for infrastructure only when you create and run it.

ummary

- Creating a cloud infrastructure for WordPress and any other application can be fully automated.
- AWS CloudFormation is a tool provided by AWS for free. It allows you to automate the managing of your cloud infrastructure.
- The infrastructure for a web application like WordPress can be created at any time on demand, without any up-front commitment for how long you'll use it.
- You pay for your infrastructure based on usage. For example, you pay for a virtual machine per second of usage.
- The infrastructure required to run WordPress consists of several parts, such as virtual machines, load balancers, databases, and network filesystems.
- The whole infrastructure can be deleted with one click. The process is powered by automation.