Amazon introduced a computing platform that has changed the face of computing in the last decade.

First, it installed a powerful computing infrastructure to sustain its core business, e-commerce, selling

a variety of goods ranging from books and CDs to gourmet foods and home appliances. Then Amazon

discovered that this infrastructure could be further extended to provide affordable and easy-to-use

resources for enterprise computing as well as computing for the masses.

In mid-2000 Amazon introduced Amazon Web Services (AWS), based on the IaaS delivery model.

In this model the cloud service provider offers an infrastructure consisting of compute and storage

servers interconnected by high-speed networks that support a set of services to access these resources.

An application developer is responsible for installing applications on a platform of his or her choice

and managing the resources provided by Amazon.

It is reported that in 2012, businesses in 200 countries used the AWS, demonstrating the international

appeal of this computing paradigm. A significant number of large corporations as well as start-ups take

advantage of computing services supported by the AWS infrastructure. For example, one start-up reports

that its monthly computing bills at Amazon are in the range of $100,000, whereas it would spend more

than $2,000,000 to compute using its own infrastructure, without benefit of the speed and flexibility

offered by AWS. The start-up employs 10 engineers rather than the 60 it would need to support its own

computing infrastructure (“Active in cloud, Amazon reshapes computing,” New York Times, August

28, 2012).

Amazon Web Services. Amazon was the first provider of cloud computing; it announced a limited

public beta release of its Elastic Computing platform called EC2 in August 2006. Figure 3.1 shows the

palette of AWS services accessible via the Management Console in late 2011 [13–18].

Elastic Compute Cloud (EC2)1 is a Web service with a simple interface for launching instances of

an application under several operating systems, such as several Linux distributions, Microsoft Windows

Server 2003 and 2008, OpenSolaris, FreeBSD, and NetBSD.

An instance is created either from a predefined Amazon Machine Image (AMI) digitally signed

and stored in S3 or from a user-defined image. The image includes the operating system, the run-time

environment, the libraries, and the application desired by the user. AMI images create an exact copy of

the original image but without configuration-dependent information such as the hostname or the MAC

address. A user can: (i) Launch an instance from an existingAMI and terminate an instance; (ii) start and

stop an instance; (iii) create a new image; (iv) add tags to identify an image; and (v) reboot an instance.

EC2 is based on the Xen virtualization strategy discussed in detail in Section 5.8. In EC2 each virtual

machine or instance functions as a virtual private server. An instance specifies the maximum amount of

resources available to an application, the interface for that instance, and the cost per hour.

A user can interact with EC2 using a set of SOAP messages (see Section 4.3) and can list available

AMI images, boot an instance from an image, terminate an image, display the running instances of a

user, display console output, and so on. The user has root access to each instance in the elastic and

secure computing environment of EC2. The instances can be placed in multiple locations in different

regions and availability zones.

EC2 allows the import of virtual machine images from the user environment to an instance through

a facility called VM import. It also automatically distributes the incoming application traffic among

multiple instances using the elastic load-balancing facility. EC2 associates an elastic IP address with

an account; this mechanism allows a user to mask the failure of an instance and remap a public IP

address to any instance of the account without the need to interact with the software support team.

Simple Storage System (S3) is a storage service designed to store large objects. It supports a minimal

set of functions: write, read, and delete.

S3 allows an application to handle an unlimited number of objects ranging in size from one byte

to five terabytes. An object is stored in a bucket and retrieved via a unique developer-assigned key. A

bucket can be stored in a region selected by the user. S3 maintains the name, modification time, an

access control list, and up to four kilobytes of user-defined metadata for each object. The object names are global. Authentication mechanisms ensure that data is kept secure; objects can be made public, and

rights can be granted to other users.

S3 supports PUT, GET, and DELETE primitives to manipulate objects but does not support primitives

to copy, rename, or move an object from one bucket to another. Appending to an object requires a read

followed by a write of the entire object.

S3 computes the MD52 of every object written and returns it in a field called ETag. A user is expected

to compute the MD5 of an object stored or written and compare this with the ETag; if the two values

do not match, then the object was corrupted during transmission or storage.

The Amazon S3 SLA guarantees reliability. S3 uses standards-based REST and SOAP interfaces (see

Section 4.3); the default download protocol is HTTP, but BitTorrent3 protocol interface is also provided

to lower costs for high-scale distribution.

Elastic Block Store (EBS) provides persistent block-level storage volumes for use with Amazon EC2

instances. A volume appears to an application as a raw, unformatted, and reliable physical disk; the size

of the storage volumes ranges from one gigabyte to one terabyte. The volumes are grouped together in

availability zones and are automatically replicated in each zone. An EC2 instance may mount multiple

volumes, but a volume cannot be shared among multiple instances. The EBS supports the creation of

snapshots of the volumes attached to an instance and then uses them to restart an instance. The storage

strategy provided by EBS is suitable for database applications, file systems, and applications using raw

data devices.

Simple DB is a nonrelational data store that allows developers to store and query data items via

Web services requests. It supports store-and-query functions traditionally provided only by relational

databases. Simple DB creates multiple geographically distributed copies of each data item and supports

high-performanceWeb applications; at the same time, it automatically manages infrastructure provisioning,

hardware and software maintenance, replication and indexing of data items, and performance tuning.

Simple Queue Service (SQS) is a hosted message queue. SQS is a system for supporting automated

workflows; it allows multiple Amazon EC2 instances to coordinate their activities by sending and

receiving SQS messages. Any computer connected to the Internet can add or read messages without

any installed software or special firewall configurations.

Applications using SQS can run independently and asynchronously and do not need to be developed

with the same technologies. A received message is “locked” during processing; if processing fails, the

lock expires and the message is available again. The time-out for locking can be changed dynamically

via the ChangeMessageVisibility operation. Developers can access SQS through standards-based SOAP

and Query interfaces. Queues can be shared with other AWS accounts and anonymously; queue sharing

can also be restricted by IP address and time-of-day. An example showing the use of message queues

is presented in Section 4.7.

CloudWatch is a monitoring infrastructure used by application developers, users, and system administrators

to collect and track metrics important for optimizing the performance of applications and

for increasing the efficiency of resource utilization.Without installing any software, a user can monitor

approximately a dozen preselected metrics and then view graphs and statistics for these

metrics.

When launching an Amazon Machine Image (AMI), a user can start the CloudWatch and specify the

type of monitoring. Basic Monitoring is free of charge and collects data at five-minute intervals for up to 10 metrics; Detailed Monitoring is subject to a charge and collects data at one-minute intervals. This

service can also be used to monitor the latency of access to EBS volumes, the available storage space for

RDS DB instances, the number of messages in SQS, and other parameters of interest for applications.

Virtual Private Cloud (VPC) provides a bridge between the existing IT infrastructure of an organization

and the AWS cloud. The existing infrastructure is connected via a virtual private network (VPN) to

a set of isolated AWS compute resources. VPC allows existing management capabilities such as security

services, firewalls, and intrusion detection systems to operate seamlessly within the cloud.

Auto Scaling exploits cloud elasticity and provides automatic scaling of EC2 instances. The service

supports grouping of instances,monitoring of the instances in a group, and defining triggers and pairs of

CloudWatch alarms and policies, which allow the size of the group to be scaled up or down. Typically,

a maximum, a minimum, and a regular size for the group are specified.

An Auto Scaling group consists of a set of instances described in a static fashion by launch configurations.

When the group scales up, new instances are started using the parameters for the runInstances

EC2 call provided by the launch configuration. When the group scales down, the instances with older

launch configurations are terminated first. The monitoring function of the Auto Scaling service carries

out health checks to enforce the specified policies; for example, a user may specify a health check for

elastic load balancing and then Auto Scaling will terminate an instance exhibiting a low performance

and start a new one. Triggers use CloudWatch alarms to detect events and then initiate specific actions;

for example, a trigger could detect when the CPU utilization of the instances in the group goes above

90% and then scale up the group by starting new instances. Typically, triggers to scale up and down are

specified for a group.

Several new AWS services were introduced in 2012; some of them are in a beta stage at the time of

this writing. Among the new services we note: Route 53, a low-latency DNS service used to manage

user’s DNS public records; Elastic MapReduce (EMR), a service supporting processing of large amounts

of data using a hosted Hadoop running on EC2 and based on the MapReduce paradigm discussed in

Section 4.6; Simple Workflow Service (SWF), which supports workflow management (see Section 4.4)

and allows scheduling, management of dependencies, and coordination of multiple EC2 instances;

ElastiCache, a service enabling Web applications to retrieve data from a managed in-memory caching

system rather than a much slower disk-based database; DynamoDB, a scalable and low-latency fully

managed NoSQL database service; CloudFront, a Web service for content delivery; and Elastic Load

Balancer, a cloud service to automatically distribute the incoming requests across multiple instances of

the application. Two new services, the Elastic Beanstalk and the CloudFormation, are discussed next.

Elastic Beanstalk, a service that interacts with other AWS services, including EC2, S3, SNS, Elastic

Load Balance, and Auto Scaling, automatically handles the deployment, capacity provisioning, load

balancing, Auto Scaling, and application monitoring functions [356]. The service automatically scales

the resources as required by the application, either up, or down based on default Auto Scaling settings.

Some of the management functions provided by the service are: (i) deployment of a new application

version (or rollback to a previous version); (ii) access to the results reported by CloudWatch monitoring

service; (iii) email notifications when application status changes or application servers are added or

removed; and (iv) access to server login files without needing to login to the application servers.

The Elastic Beanstalk service is available to developers using a Java platform, the PHP server-side

description language, or .NET framework. For example, a Java developer can create the application using any Integrated Development Environment (IDE) such as Eclipse and package the code into a Java

Web Application Archive (a file of type “.war”) file. The “.war” file should then be uploaded to the

Elastic Beanstalk using the Management Console and then deployed, and in a short time the application

will be accessible via a URL.

CloudFormation allows the creation of a stack describing the infrastructure for an application. The

user creates a template, a text file formatted as in Javascript Object Notation (JSON), describing the

resources, the configuration values, and the interconnection among these resources. The template can

be parameterized to allow customization at run time, (e.g., to specify the types of instances, database

port numbers, or RDS size). A template for the creation of an EC2 instance follows:

{

"Description" : "Create instance running Ubuntu Server 12.04 LTS

64 bit AMI

"Parameters" : {

"KeyPair" : {

"Description" : "Key Pair to allow SSH access to the instance",

"Type" : "String"

}

},

"Resources" : {

"Ec2Instance" : {

"Type" : "AWS::EC2::Instance",

"Properties" : {

"KeyName" : {"Ref" : "KeyPair"},

"ImageId" : "aki-004ec330"

}

}

},

"Outputs" : {

"InstanceId" : {

"Description" : "The InstanceId of the newly created

instance",

"Value" : { "Ref" : "Ec2InstDCM"}

}

},

"AWSTemplateFormatVersion" : "2012-03-09"

}

The Amazon Web Services Licensing Agreement (AWSLA) allows the cloud service provider to

terminate service to any customer at any time for any reason and contains a covenant not to sue Amazon

or its affiliates for any damages that might arise out of the use of AWS. As noted in [133], the AWSLA

prohibits the use of “other information obtained through AWS for the purpose of direct marketing,

spamming, contacting sellers or customers.” It prohibits AWS from being used to store any content that is “obscene, libelous, defamatory or otherwise malicious or harmful to any person or entity.” It

also prohibits S3 from being used “in any way that is otherwise illegal or promotes illegal activities,

including without limitation in any manner that might be discriminatory based on race, sex, religion,

nationality, disability, sexual orientation, or age.”

Users have several choices for interacting with and managing AWS resources from either a Web

browser or from a system running Linux or Microsoft Windows:

1. The AWSWebManagement Console, available at http://aws.amazon.com/console/; this is the

easiest way to access all services, but not all options may be available in this mode.

2. Command-line tools; see http://aws.amazon.com/developertools.

3. AWS SDK libraries and toolkits provided for several programming languages, including Java, PHP,4

C#, and Obj C.

4. Raw REST requests (see Section 4.3 for a discussion of architectural styles for cloud applications).

Regions and Availability Zones. Today Amazon offers cloud services through a network of data

centers on several continents, (see Table 3.15). In each region there are several availability zones

interconnected by high-speed networks; regions communicate through the Internet and do not share

resources.

An availability zone is a data center consisting of a large number of servers.Aservermay runmultiple

virtual machines or instances, started by one or more users; an instance may use storage services, S3,

EBS), and Simple DB, as well as other services provided by AWS (see Figure 3.2). A cloud interconnect

allows all systems in an availability zone to communicate with one another and with systems in other

availability zones of the same region.

Storage is automatically replicated within a region; S3 buckets are replicated within an availability

zone and between the availability zones of a region, whereas EBS volumes are replicated only within the same availability zone. Critical applications are advised to replicate important information in multiple

regions to be able to function when the servers in one region are unavailable due to catastrophic events.

A user can request virtual servers and storage located in one of the regions. The user can also request

virtual servers in one of the availability zones of that region. The Elastic Compute Cloud (EC2) service

allows a user to interact and to manage the virtual servers.

The billing rates in each region are determined by the components of the operating costs, including

energy, communication, and maintenance costs. Thus, the choice of the region ismotivated by the desire

to minimize costs, reduce communication latency, and increase reliability and security.

An instance is a virtual server. The user chooses the region and the availability zone where this virtual

server should be placed and selects from a limited menu of instance types: the one that provides the

resources, CPU cycles, main memory, secondary storage, communication, and I/O bandwidth needed

by the application.

When launched, an instance is provided with a DNS name. This name maps to a private IP address

for internal communication within the internal EC2 communication network and a public IP address

for communication outside the internal Amazon network, (e.g., for communication with the user that

launched the instance).Network Address Translation (NAT) maps external IP addresses to internal ones.

The public IP address is assigned for the lifetime of an instance and it is returned to the pool of

available public IP addresses when the instance is either stopped or terminated. An instance can request

an elastic IP address, rather than a public IP address. The elastic IP address is a static public IP address

allocated to an instance from the available pool of the availability zone. An elastic IP address is not

released when the instance is stopped or terminated and must be released when no longer needed.

The Charges for Amazon Web Services. Amazon charges a fee for EC2 instances, EBS storage,

data transfer, and several other services. The charges differ from one region to another and depend on

the pricing model; see http://aws.amazon.com/ec2/pricing for the current pricing structure.

There are three pricing models for EC2 instances: on-demand, reserved, and spot. On-demand

instances use a flat hourly rate, and the user is charged for the time an instance is running; no reservation

is required for this most popular model. For reserved instances a user pays a one-time fee to lock in a

typically lower hourly rate. This model is advantageous when a user anticipates that the application will

require a substantial number of CPU cycles and this amount is known in advance. Additional capacity

is available at the larger standard rate. In case of spot instances, users bid on unused capacity and their

instances are launched when the market price reaches a threshold specified by the user.

The EC2 system offers several instance types:

• Standard instances. Micro (StdM), small (StdS), large (StdL), extra large (StdXL); small is the

default.

• High memory instances. High-memory extra-large (HmXL), high-memory double extra-large

(Hm2XL), and high-memory quadruple extra-large (Hm4XL).

• High CPU instances. High-CPU extra-large (HcpuXL).

• Cluster computing. Cluster computing quadruple extra-large (Cl4XL).

Table 3.2 summarizes the features and the amount of resources supported by each instance. The

resources supported by each configuration are main memory, virtual computers (VCs) with a 32- or 64-bit

architecture, instance memory (I-memory) on persistent storage, and I/O performance at two levels: moderate

(M) or high (H). The computing power of a virtual core is measured in EC2 compute units (CUs).

A main attraction of Amazon cloud computing is the low cost. The dollar amounts charged for one

hour of running Amazon’s services under Linux or Unix and Microsoft Windows in mid-2012 are summarized

in Table 3.3. There are no charges for data transfers from the user’s site to the Amazon network

or within the Amazon network; the charges for data transfer from the AWS network to the outside world

depend on the region. For example, the charges for the USWest (Oregon) region are shown in Table 3.4.

An Evaluation of Amazon Web Services. In 2007 Garfinkel reported the results of an early evaluation

of Amazon Web Services [133]. The paper reports that EC2 instances are fast, responsive, and very reliable; a new instance could be started in less than two minutes. During the year of testing, one

unscheduled reboot and one instance freeze were experienced. No data was lost during the reboot, but

no data could be recovered from the virtual disks of the frozen instance.

To test the S3 service, a bucket was created and loaded with objects in sizes of 1 byte, 1 KB, 1MB,

16 MB, and 100 MB. The measured throughput for the 1-byte objects reflected the transaction speed

of S3 because the testing program required that each transaction be successfully resolved before the

next was initiated. The measurements showed that a user could execute at most 50 non-overlapping

S3 transactions. The 100 MB probes measured the maximum data throughput that the S3 system could

deliver to a single client thread. From the measurements, the author concluded that the data throughput

for large objects was considerably larger than for small objects due to a high transaction overhead. The

write bandwidth for 1 MB data was roughly 5 MB/s, whereas the read bandwidth was five times lower

at 1MB/s.

Another test was designed to see if concurrent requests could improve the throughput of S3. The

experiment involved two virtual machines running on two different clusters and accessing the same

bucket with repeated 100 MB GET and PUT operations. The virtual machines were coordinated, with

each one executing one to six threads for 10 min and then repeating the pattern for 11 h. As the number

of threads increased from one to six, the bandwidth received by each thread was roughly cut in half and

the aggregate bandwidth of the six threads was 30MB/s, about three times the aggregate bandwidth of

one thread. In 107,556 tests of EC2, each one consisting of multiple read and write probes, only six

write retries, three write errors, and four read retries were encountered.