Naresh Kumar Sehgal
Pramod Chandra P. Bhatt

Cloud Computing

Concepts and Practices

Springer

Cloud Computing

Naresh Kumar Sehgal · Pramod Chandra P. Bhatt

Cloud Computing

Concepts and Practices

Springer

Naresh Kumar Sehgal
Santa Clara, CA
USA

Pramod Chandra P. Bhatt
Bangalore
India

ISBN 978-3-319-77838-9 ISBN 978-3-319-77839-6 (eBook)
https://doi.org/10.1007/978-3-319-77839-6

Library of Congress Control Number: 2018934927

Printed on acid-free paper

This Springer imprint is published by the registered company Springer International Publishing AG part of Springer Nature
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Pramod would like to dedicate this book to Sir Timothy John Berners-Lee, inventor of World Wide Web, which revolutionized the way we communicate information and share knowledge.

Naresh wants to dedicate this book to his late parents, Sh. Pawan Kumar Sehgal and Mrs. Pushpa Sehgal, who inspired him to always be curious and not to give up until a task was done.

Foreword by Diane M. Bryant

The Cloud is a massive advancement in computer architecture. It is transforming the way all digital services are delivered and consumed, accelerating the digitization of all industries. With its high efficiency and scale, Cloud Computing enables the democratization of technology worldwide.

With breakneck pace of technology innovation, broad education of the Cloud lags the adoption. This book drives to close that gap and broaden the knowledge base on what is a critical and foundational capability.

For all those looking to contribute to the digital economy, this book is a highly valuable source of information.

Santa Clara, USA
November 2017

Diane M. Bryant
Group President, Intel Corporation

Foreword by Professor V. Rajaraman

A Google search of "Cloud Computing" resulted in 90 million hits in December 2017. It is thus apparent that Cloud Computing is one of the hottest terms in computing. It is a metaphor for computers connected to the Internet, which are enclosed in a cloud-like boundary in figures that are used in PowerPoint presentations. According to a Wikipedia entry (since removed), the term was first used academically in a paper entitled "Intermediaries on Cloud Computing" by Ramnath Chellappa in 1997. A start-up called Net Centric (now defunct) applied for a trademark for "Cloud Computing" in 1999 for educational services that was rejected. The terms became popular after it was used in 2006 by Eric Schmidt (then the CEO of Google). The idea of Cloud computing as a computing utility, that is, "pay for what you use" computing, was foreseen in 1961 by John McCarthy. In 1957 (the mainframe era), MIT had one of the most powerful computers at that time, an IBM 704, that was used in a batch mode with punched cards as input. It was time-consuming to write complex programs and make them work on this computer. John McCarthy, who had joined the faculty of MIT in 1959, found it very difficult to use this computer for his research. He wrote, in frustration, a note to the Director of the Computer Centre suggesting that teletypewriters from the offices of faculty members be connected to the IBM 704 permitting interactive time-shared use of the computer by several persons simultaneously. Both hardware and software technologies were not available to implement this idea. Fernando Corbató, who was then the Associate Director of the MIT Computer Center, took up the challenge and led a team that designed the hardware and the software of what is known as the Compatible Time-Sharing System (CTSS) on an IBM 709, the successor of IBM 704. The system became operational in 1961 that happened to be the centennial year of MIT. John McCarthy in a lecture during the centennial year celebration said that time-shared computers someday in the future will be organized as a public utility just as a telephone system is a public utility and that such a computing utility could become the basis of a new and important industry. It took over five decades to realize the dream of John McCarthy, and now, Cloud Computing is an important industry that provides not only as much computing and storage as one wishes from one's desk but also application programs "on tap," on a pay only for what you use

basis. A number of technologies had to mature to provide what we now call Cloud architecture. Among these are the emergence of the high-speed Internet, rapid development of high bandwidth communication at competitive rates, availability of huge storage at throwaway prices, and accessibility of powerful and highly reliable computers at low cost. In addition, developments in software had to mature that include advances in operating systems, powerful high-level language compilers, service-oriented architecture, and cryptography. The major features of the Cloud model of computing are service orientation, on-demand provisioning, virtualization, multi-tenancy, self-healing, and SLA driven. Pay as you use is an added business model, which is the main feature of a utility.

The architecture model of Cloud Computing is a paradigm shift and will be the way data centers will be architected in the future. In the curriculum of undergraduates in computer science and engineering, a number of discrete subjects such as programming, computer logic design, computer architecture, computer networks, Internet architecture, operating systems, World Wide Web software, and object-oriented design are taught. A synthesis of many of the concepts learnt in these courses is required to design Cloud architecture. Cloud Computing will become a very important course taken by students in their final year of the undergraduate curriculum, and textbooks are needed to teach such a course. A number of books on Cloud Computing have appeared since 2013. Many of them are addressed to working professionals and are not comprehensive. This book by Dr. Naresh Sehgal and Prof. P. C. P. Bhatt is written primarily as a textbook. A look at the table of contents shows that the coverage of the book is comprehensive. It starts from basic principles of Cloud Computing, describes its historical evolution, and outlines various models of the Cloud, types of services, how to characterize workloads, customers' expectations, management of Clouds, security issues, problems in migrating applications to a Cloud, analytics, economics, and future trends. A uniform feature of this book is a nice combination of a scholarly exposition with a practical orientation. I particularly appreciated their treatment of workload characterization, management, and monitoring. The authors have conducted experiments on the performance of the providers of Cloud Services using diverse workloads and have analyzed the results. They have devoted considerable effort to explain the problems of Cloud security. Every chapter has a large number of references to original sources that will provide the students enough information to explore beyond what is in the text. Exercises are well thought out, and there is a project orientation in these so that students can learn by doing.

Dr. Naresh Sehgal has a wealth of experience in the design and development of Intel products and has been training professionals in the industry. This has resulted in the practical orientation of this book. Prof. P. C. P. Bhatt has over 40 years of experience in teaching students in reputed academic institutions internationally and also to professionals in industries. He is one of the pioneer teachers of India. This book has gained a lot by this admirable collaboration of an experienced professional with a reputed academic. I thoroughly enjoyed reading this book and learnt a lot

form it. I am confident that both students and working professionals will gain a lot of in-depth knowledge on Cloud Computing by reading this book. It is a valuable addition to the literature on Cloud Computing, and I congratulate the authors on their fine effort.

Bangalore, India

V. Rajaraman
Supercomputer Education and Research Centre
Indian Institute of Science

Preface

The idea for this book started with a call from Prof. P. C. P. Bhatt, whom Naresh has known for over a decade. Prof. Bhatt is the author of a widely used Operating Systems book, which is in its 5th edition in India. He asked Naresh to collaborate on his next edition, but Naresh's expertise is in Cloud Computing, on which he had taught a graduate class at Santa Clara University in California. Thus, a new book was conceived.

Cloud has evolved as the next natural step after Internet, connecting mainframe like centralized computing capabilities in a data center with many handheld and distributed devices on the other end. Thus, computing has taken the form of exercise dumbbell equipment, with Servers on one end and Clients on the other, joined by the fat pipes of communication in between. Furthermore, it has transformed the way computing happens, which for both authors started by carrying decks of JCL (Job Control Language) cards for feeding an IBM mainframe, culminated with collaboration for this book using the Cloud. This is amazing, especially considering that even a smartphone has more memory and compute than was present on the entire Saturn V rocket that carried astronauts to the Moon and back. This implies that more such miracles are on the way, enabled by widely and cheaply available compute power made possible by Cloud Computing.

Objective of this book is to give an exposure to the shift in paradigm caused by Cloud Computing and unravel the mystery surrounding basic concepts including performance and security. This book is aimed at teaching IT students, and enabling new practitioners with hands-on projects, such as setting a Web site and a blog. Focus areas include Best Practices for using Dynamic Cloud Infrastructure, Cloud Operations Management, and Security.

As a textbook, our target audiences are students in computer science, information technology, and MBA students pursuing technology-based management studies. This book assumes some background and understanding of basic programming languages, operating systems, and some aspects of software engineering. To that extent, it would suit senior undergraduates or graduates in their early semesters. As a technical manuscript, this book has enough in-depth coverage to

interest IT managers and SW developers who need to take a call on migrating to Cloud.

In the initial four chapters, this book introduces relevant terms, definitions, concepts, historical evolution, and underlying architecture. Midway, we cover taxonomy based on the Cloud Service models. In particular, we cover the service options that are available as infrastructure and software services. This closes with explanation of the features of public, private, and hybrid Cloud Service options. Chapters 5–9 dive deeper to explain underlying workload characterization, security, and performance-related issues. To explain the key question of migration, we have chosen EDA (Electronics Design Automation). EDA presents a typical dilemma of how much and what could, or should, go into Public Cloud. Chapter 10 onward we expose the readers to business considerations from effective cost–benefit and business perspective. Also presented is a key area of analytics to reinforce the utility of Cloud Computing. Finally, to keep the user interested, we offer glimpses of hands-on operations and also project ideas that can be carried forward. To encourage further reading and challenge, we have included a few advance topics toward the end of this book.

In our opinion, as a textbook the student's interest would be best served if there is an accompanying weekly laboratory exercise using some Public Cloud or captive university Cloud platform. Public Cloud Services are available from Google, Microsoft, Amazon, and Oracle under their education support programs.

We recognize that a book has a limited shelf life so would like to plan ahead for an update. Thus, we invite the readers to send their inputs, comments, and any feedback to cloudbookauthors@gmail.com. These will be incorporated in the next edition. Many thanks!!

Santa Clara, USA — Naresh Kumar Sehgal
Bangalore, India — Pramod Chandra P. Bhatt

Acknowledgements

Pramod would like to acknowledge indirect contribution of his numerous students. They shared their understanding and thoughts. As a result, he benefited and grew professionally.

Naresh wants to acknowledge help from several experts, starting with Tejpal Chadha, who had invited Naresh to teach the SCU class in 2015, and his colleagues at Intel who taught him Cloud. Thanks to Naresh's teaching assistant Vijayalakshmi Gowri Shanker for sharing her project assignments. Naresh's students used megaApp and Scouts tools developed by Shiv Shankar, to do performance measurements of various Public Cloud servers. We appreciate help from Mrittika Ganguli, Shesha Krishnapura, Vipul Lal, Ty Tang, and Raghu Yeluri for reviewing this book and offering their technical feedback(s) for improvement. Vikas Kapur gave his valuable time to suggest many changes, which have improved the quality of this book. Concepts and ideas from several previously published papers with Prof. John M. Acken, Prof. Sohum Sohoni, and their students were used to compile this book, for which we are very grateful. We appreciate excellent support from Charles Glaser and Brian Halm, along with the staff of Springer Publications to bring this book in your hands. Various trademarks used in this book are the respective ownership of their companies. An approval from Naresh's Intel managers, Chris Hotchkiss and Stephen L. Smith, was instrumental in bringing this book out for a publication.

Finally, Naresh wants to thank his wife Sunita, and children Hetesh, Gauri and Garima for supporting this labor of love.

Contents

1 Introduction 1
- 1.1 Motivation 1
- 1.2 Cloud Computing Definitions 2
- 1.3 Cloud Computing Operational Characteristics 3
  - 1.3.1 Cloud Computing Benefits 6
  - 1.3.2 Cloud Computing Potential Risks 6
- 1.4 Cloud Computing Trends 6
  - 1.4.1 Trend #1: Abstraction of Network, Storage, Database, Security, and Computing Infrastructure 6
  - 1.4.2 Trend #2: A Pricing Model that Is Retail in Its Conception 7
  - 1.4.3 Trend #3: Service-Level Agreements (SLAs) 7
- 1.5 Cloud Computing Needs 8
- 1.6 Points to Ponder 9

References 10

2 Foundations of Cloud Computing 11
- 2.1 Historical Evolution 11
- 2.2 Different Network Protocols 13
- 2.3 Evolution of Enterprise IT 16
- 2.4 Evolution of Web Services 17
- 2.5 Server Operations in a Data-Center 20
- 2.6 Server-Based Web Services 25
- 2.7 Service-Oriented Architecture 26
- 2.8 Building an Enterprise SOA Solution 28
- 2.9 Top-Down Versus Bottom-Up Approach 30
- 2.10 Enterprise Service Bus (ESB) 31
- 2.11 Enterprise Implementation on Private Clouds 33
- 2.12 Enterprise Implementation on Hybrid Clouds 35
- 2.13 Web Threat Models 35

2.14 Open Web Application Security Project . . . 37
2.15 Summary . . . 39
2.16 Points to Ponder . . . 39
References . . . 40

3 Cloud Computing Pyramid . . . 41
3.1 Roots of Cloud Computing . . . 41
3.2 Essential Characteristics of Cloud Computing . . . 44
3.3 Cloud Players and Their Concerns . . . 45
3.4 Considerations for Cloud Data Centers . . . 47
3.4.1 Migration . . . 48
3.4.2 Performance . . . 48
3.5 Points to Ponder . . . 49
References . . . 49

4 Features of Private and Public Clouds . . . 51
4.1 Customer Expectations of Cloud Computing . . . 51
4.2 Interoperability of Cloud Computing . . . 53
4.3 Reliability of Cloud Computing . . . 53
4.4 Performance of Cloud Computing . . . 55
4.5 A Sample Study . . . 56
4.6 Summary . . . 59
4.7 Points to Ponder . . . 59
References . . . 60

5 Cloud Workload Characterization . . . 61
5.1 Motivation . . . 61
5.2 Some Background on Workload Characterization . . . 62
5.3 Top-Level Cloud Workload Categorization . . . 65
5.4 Cloud Workload Categories . . . 66
5.5 Computing Resources . . . 70
5.5.1 Data Buses Between Servers . . . 71
5.6 Example Workload Categorizations . . . 72
5.7 Temporal Variability of Workloads . . . 72
5.8 Low-Level or Hardware Metrics of Computer Utilization . . . 76
5.9 Dynamic Monitoring and Cloud Resource Allocation . . . 77
5.10 Benefits to Cloud Service Providers . . . 78
5.11 Summary . . . 80
5.12 Points to Ponder . . . 81
References . . . 81

6 Cloud Management and Monitoring . . . 85
6.1 Motivation . . . 85
6.2 Introduction to Cloud Setup and Basic Tools . . . 85
6.3 Noisy Neighbors in a Cloud . . . 86
6.4 Cloud Management Requirements . . . 87

6.5 Essentials of Monitoring . . . 88
6.6 Some Example of Monitoring Tools . . . 89
6.7 Future Work . . . 91
6.8 Points to Ponder . . . 92
References . . . 92

7 Cloud Computing and Information Security . . . 93
7.1 Background and Definitions . . . 93
7.2 Security Concerns of Cloud Operating Models . . . 95
7.3 Identity Authentication . . . 96
7.4 Secure Transmissions . . . 100
7.5 Secure Storage and Computation . . . 100
7.6 The Security Players . . . 101
7.7 Traditional Versus Internet Security Issues . . . 102
7.8 Variations and Special Cases for Security Issues with Cloud Computing . . . 105
7.8.1 The Players . . . 105
7.8.2 Secure Communication . . . 106
7.8.3 An Example Security Scenario for Cloud Computing . . . 107
7.9 A Few Key Challenges Related to Cloud Computing and Virtualization . . . 108
7.10 Some Suggested Security Practices for Cloud Computing . . . 110
7.11 Summary . . . 111
7.12 Points to Ponder . . . 112
References . . . 112

8 Migrating to Cloud . . . 115
8.1 Cloud Business Models . . . 115
8.2 A Case Study: B2C . . . 116
8.3 A Case Study: B2B . . . 117
8.4 A Case Study: C2C . . . 119
8.5 Summary . . . 120
8.6 Points to Ponder . . . 120
References . . . 121

9 Migrating a Complex Industry to Cloud . . . 123
9.1 Background . . . 123
9.2 Introduction to EDA . . . 124
9.3 A Brief History of EDA Tools and Flows . . . 125
9.3.1 The Nascent Years of the 70s . . . 125
9.3.2 The Roaring 80s . . . 126
9.3.3 Growing up in the 90s . . . 126
9.3.4 Maturing into the First Decade of Twenty-First Century . . . 127
9.3.5 From 2010s till Now, EDA Stable . . . 127

9.4 EDA Flow Steps Mapping to Cloud . . . 128
9.5 Considerations for Cloud Computing Adoption . . . 133
9.6 Summary . . . 136
9.7 Points to Ponder . . . 137
References . . . 138

10 Costing and Billing Practices in Cloud . . . 141
10.1 Cloud as a Service (CaaS): The Billing Imperatives . . . 141
10.1.1 Billing and Best Practices . . . 141
10.2 Pay as You Go . . . 142
10.3 Amazon EC2 Motivations and Setup . . . 143
10.3.1 Amazon's On-Demand Instances . . . 144
10.3.2 Amazon Spot Instances . . . 144
10.3.3 Amazon Reserved Instances . . . 145
10.3.4 Amazon Dedicated Instances and Dedicated Hosts . . . 145
10.4 Motivation and Methods for Right Sizing Customer VMs . . . 146
10.4.1 Elastic IP . . . 146
10.4.2 Elastic Load Balancing . . . 147
10.4.3 Auto-Scaling . . . 148
10.5 Cost Minimization . . . 149
10.6 Capacity Forecasting . . . 152
10.7 Optimizations Across Clouds . . . 153
10.8 Types of Cloud Service-Level Agreements . . . 155
10.9 Summary . . . 157
10.10 Points to Ponder . . . 157
References . . . 158

11 Analytics in the Cloud . . . 159
11.1 Background and Problem Statement . . . 159
11.2 Introduction to MapReduce . . . 162
11.3 Introduction to Hadoop . . . 162
11.4 Usage of Amazon's MapReduce . . . 165
11.5 Twitter Sentimental Analysis Using Cloud . . . 167
11.6 Future Possibilities . . . 168
11.7 Points to Ponder . . . 169
References . . . 169

12 Future Trends in Cloud Computing . . . 171
12.1 Revisiting History of Computing . . . 171
12.2 Current Limitations of Cloud Computing . . . 171
12.3 Emergence of Internet of Things (IoT) . . . 173
12.4 Emergence of Machine Learning . . . 174
12.5 Emergence of Edge Computing . . . 176
12.6 Security Issues in Edge Computing . . . 177

12.7 Security Considerations for Edge Computing 178
12.8 Future Work Needed 180
12.9 Example of an IoT-Based Cloud Service 181
12.10 Summary 182
12.11 Points to Ponder 183
References 183

13 A Quick Test of Your Cloud Fundamentals Grasp 185

14 Hands-On Project to Use Cloud Service Provider 195
14.1 Project 1: Install Lamp Stack on Amazon EC2 195
14.1.1 Installing Lamp Web Server on AWS via EC2 195
14.1.2 Installing Wordpress 205
14.1.3 Wordpress URL 217
14.2 Project 2: Install PHP on Your AWS Instance 217
14.3 Project 3: Enhance Security of Your AWS Instance 219
14.4 Project 4: Setup a Load Balancer for Your AWS Instance 221
14.4.1 Elastic Load Balancer Setup 221
14.4.2 Unique Features of AWS Load Balancer 226
14.5 Project 5: Use Elastic IP for Your AWS Instance 227
14.5.1 How to Make an Instance Elastic 227
14.5.2 Extra: Elastic IP 227
14.6 Bonus 229
14.7 Points to Ponder 239

Appendix A 241

Appendix B: Additional Considerations for Cloud Computing 259

Appendix C: Suggested List of Additional Cloud Projects 265

Index 267

About the Authors

Naresh Kumar Sehgal has been working at Intel since 1988, prior to that completed his B.E. from Punjab Engineering College, M.S. and Ph.D. from Syracuse University. Naresh has taught a Cloud Computing class at Santa Clara University, where he also earned a MBA.

Pramod Chandra P. Bhatt started his teaching career in 1965 at IIT Kanpur, and then moved to IIT Delhi in 1969, and retired from IIT Delhi in 1996. Prof Bhatt also worked as a visiting professor at the University of Ottawa, McGill University, Montreal (Canada), Universities of Dortmund, Paderborn and Bochum (Germany), and Kochi University of Technology (Japan). Prof. Bhatt has a M.E. from Calcutta University and a PhD from IIT Kanpur. He has also been a Konrad Zuse Fellow at the University of Dortmund.

Abbreviations

AMI	Amazon Machine Image
AWS	Amazon Web Service
AZ	Availability zone
B2B	Business-to-business
B2C	Business-to-consumer
C2C	Consumer-to-consumer
CAD	Computer-aided design
CapEx	Capital expenditure
CSRF	Cross-site request forgery
CV	Coefficient of variation
DARPA	Defense Advanced Research Projects Agency
DBMS	Database management system
DOS	Denial of service
DR	Disaster recovery
EC2	Elastic Compute Cloud
EMR	Elastic MapReduce
ESB	Enterprise service bus
HPC	High-Performance Computing
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	HTTP Secure
IaaS	Infrastructure as a Service
IB	In-Band
IoT	Internet of Things
IP	Intellectual Property
IP	Internet Protocol
LAN	Local area network
ML	Machine learning
NIST	National Institute of Standards and Technology
OEM	Original Equipment Manufacturer

OOB	Out of band
OpEx	Operational expenditure
OWASP	Open Web Application Security Project
PaaS	Platform as a Service
QoS	Quality of Service
REST	Representational state transfer
SaaS	Software as a Service
SLA	Service-level agreement
SMB	Small and medium business
SNIA	Storage Networking Industry Association
SOA	Service-oriented architecture
SOAP	Simple Object Access Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
UDDI	Universal Description, Discovery, and Integration
VLSI	Very-large-scale integration
VM	Virtual machine
VMM	Virtual machine monitor
VPC	Virtual Private Cloud
VPN	Virtual private network
WAN	Wide area network
WSDL	Web Services Description Language
WWW	World Wide Web
XML	eXtensible Markup Language
XSS	Cross-site scripting

Definitions

1. **Auto-Scaling:** Auto-scaling is used by AWS to automatically add new servers to support an existing service, when more customers are using it and service may be slowing down.
2. **Cache:** A cache is a high-speed memory included in a CPU (central processing unit) package, for storing data items that are frequently accessed, to speed up a program. There may be multiple levels of cache memories.
3. **Check Pointing:** Check pointing is a practice of taking frequent snapshots of a database or virtual machine memory, to restore operations in case of a machine crash or failure.
4. **Clickjacking:** Clickjacking is a security attack to hijack a user's click on a Web site or link and direct it to an unintended target, to trick them into giving away private information.
5. **Client-Server:** Client-Server is a computing model where a single server provides services to many client devices.
6. **Cloud Bursting:** Cloud Bursting a process to request and access resources beyond an enterprise's boundary, typically reaching into a Public Cloud from a Private Cloud, when user load increases.
7. **Cloud Watch:** Cloud Watch is an AWS service to monitor performance of a user's virtual machine. It typically reports CPU utilization, and if that exceeds beyond a preset limit, then it may indicate a slowdown, thus starting an action such as auto-scaling (see #2 above).
8. **Cluster:** Cluster refers to a group of interconnected servers, with in a rack or combining a few racks, to perform a certain function such as supporting a large database or to support a large workload.
9. **Container:** A Container refers to packaged software that contains everything needed to run it, e.g., code, runtime executables, system tools, system libraries, environmental settings etc. It is convenient to deploy and upgrade. Containers can be deployed for both Linux and Windows operating systems.
10. **Controllability:** Controllability is the measure that indicates the ease with which state of a system can be controlled or regulated.

11. **Cookies:** Cookie is a small piece of data sent by a Web site and stored locally on a user's computer. It is used by Web sites to remember state information, such as items added in the shopping cart in an online store.
12. **CSRF:** Cross-site request forgery, also known as one-click attack or session riding and abbreviated as CSRF (sometimes pronounced sea-surf) or XSRF, is a type of malicious exploit of a Web site where unauthorized commands are transmitted from a user that the Web application trusts, to perform unauthorized actions such as money transfer on a bank's Web site.
13. **Edge Computing:** Edge Computing refers to the notion of having compute ability at the edge of a network with local storage and decision-making abilities. Edge denotes the end point of a network.
14. **Elasticity:** Elasticity is a property of computing and storage facilities in a Cloud to expand in view of a growing need and shrink when the need goes away.
15. **Enterprise Service Bus (ESB):** ESB is a communications system between mutually interacting software applications in a service-oriented architecture (SOA).
16. **Fabs:** Fabs refer to fabrication plants for manufacturing Silicon Chips.
17. **Fault Tolerance:** Fault tolerance is the property of a computer system to keep functioning properly in the presence of a hardware or software fault.
18. **Frame:** Frame is a portion of Web browser's window.
19. **Frame Busting:** Frame busting is a technique used by Web applications to prevent their Web pages from being displayed within a frame, to prevent malicious attacks.
20. **Hadoop:** Apache Hadoop is an open-source software framework used for distributed storage and processing of dataset of big data using the MapReduce programming.
21. **Hybrid Cloud:** Hybrid Cloud is a Cloud Computing environment, which uses a mix of on-premises, Private Cloud and third-party, Public Cloud Services with orchestration between the two platforms.
22. **Interoperability:** Interoperability is the ability of different information technology systems and software applications to communicate, exchange data, and use the information that has been exchanged.
23. **Latency:** Latency refers time delay in transmission of network packets or in accessing data from a memory.
24. **Machine Learning:** Machine learning refers to the ability of computers to learn without being explicitly programmed.
25. **MapReduce:** MapReduce is a programming model and implementation for processing and generating big datasets with a parallel, distributed algorithm on a cluster.
26. **Noisy Neighbors:** Noisy neighbor is a phrase used to describe a Cloud Computing infrastructure co-tenant that monopolizes bandwidth, disk I/O, CPU, and other resources and can negatively affect other users' Cloud performance.
27. **Observability:** Observability is a measure of how well internal states of a system can be inferred from knowledge of its external outputs.
28. **Open Stack:** OpenStack is a Cloud operating system that controls large pools of compute, storage, and networking resources throughout a data center.

29. **Optimizations:** In computing, optimization is the process of modifying a system to make some features of it work more efficiently or use fewer resources.
30. **Private Clouds:** Private Cloud is dedicated to a single organization.
31. **Public Clouds:** A Public Cloud offers its services to a full range of customers. The computing environment is shared with multiple tenants, on a free or pay-per-usage model.
32. **Reliability:** Reliability is the quality of performing consistently well.
33. **Self-Service:** Self-service Cloud Computing is a facility where customers can provision servers, storage and launch applications without going through an IT person.
34. **Streaming:** Streaming is a technique for transferring data so that it can be processed as a steady and continuous stream. Streaming technologies are becoming important because most users do not have fast enough access to download large datasets. With streaming, the client browser or plug-in can start displaying the data before the entire file has been transmitted.
35. **Thick and Thin Clients:** Thick client is a full-featured computer that may be connected to a network. A thin client is a lightweight computer built to connect to a server from a remote location. All features typically found on a thick client, such as software applications, data, and memory, are stored on a remote data center server when using a thin client.
36. **Threat Model:** Threat modeling is an approach for analyzing the security of an application. It is a structured approach to identify, quantify, and address the security risks associated with an application.
37. **Verification:** Verification is the process of establishing the truth, accuracy, or validity of something.
38. **Virtual Machines:** A virtual machine (VM) is an emulation of a computer system.
39. **Virtual Machine Monitor:** A virtual machine Monitor (VMM) enables users to simultaneously run different operating systems, each in a different VM, on a server.
40. **Virtual Private Cloud:** A Virtual Private Cloud (VPC) is an on-demand configurable pool of shared computing resources allocated within a Public Cloud environment, providing a certain level of isolation between the different organizations.
41. **Vulnerability:** Vulnerability refers to the inability of a system to withstand the effects of a hostile environment. A window of vulnerability (WoV) is a time frame within which defensive measures are diminished and security of the system is compromised.
42. **Web Service:** Web Service is a standardized way of integrating and providing Web-based applications using the XML, SOAP, WSDL, and UDDI open standards over an Internet Protocol backbone.
43. **Workload:** In enterprise and Cloud Computing, workload is the amount of work that the computer system has been given to do at a given time. Different types of tasks may stress different parts of a system, e.g., CPU-bound or memory-bound workloads.

Chapter 1
Introduction

1.1 Motivation

The Internet has brought revolutionary changes in the way we use computers and computing services. The paradigm mostly used is a Client–Server architecture supported over Internet. Client refers to computers that are generally used by individuals and include desktops, laptops, handheld smartphones, or other similar devices. A server refers to bigger computers that can simultaneously support multiple users, typically using many multi-core processors, larger memories, and bigger storage capacity. Historically, mainframes fit this description and lately smaller compute servers have been pooled together to meet the needs of several users at once. These servers are placed in a rack, also known as a rack of servers networked together, and housed in a building called a data-center (DC) along with storage devices. The power, cooling, and physical access to a data-center are tightly controlled for operational and security reasons.

Recently, networking and storage technologies have increased in complexity. As a consequence, the task to manage them has become harder. Information Technology (IT) professionals manage server infrastructure management in a DC to ensure that hardware and software are working properly. Hence, a fully operational DC requires large amounts of capital and operational expenditures, but may not be fully utilized at all hours of a day, over a week or a year. This has led some people to consider sharing the resources of a DC with other users from other organizations, while maintaining data security, application-level protection, and user-isolation. This has been enabled by operating systems in the past, and most recently with hardware-based virtualization. The overall net effect of such technologies has been to eliminate the direct capital and operational expenditure of a DC for users on a shared basis, and for the DC operator to spread their costs over many users. This is akin to airlines industry, where buying a commercial ticket for occasional travel is much cheaper than the cost of owning an airplane and hiring staff to operate it.

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6_1

1.2 Cloud Computing Definitions

Cloud Computing refers to providing IT services, applications, and data using dynamically scalable pool(s), possibly residing remotely, such that users do not need to consider the physical location of server or storage that supports their needs. According to NIST, the definition of Cloud Computing is still evolving [1]. Their current definition for Cloud Computing is "a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." Armbrust provides another, similar, definition of Cloud Computing as "the applications delivered as services over Internet and the hardware and system in the data centers that provide these services. The services themselves have long been referred to as Software as a Service (Saas)" [2]. The Cloud can include Infrastructure as a Service (IaaS) and a platform as a Service (PaaS). These service models are defined, starting from the top of a symbolic pyramid, as follows:

1. **Software as a Service (Saas)**: is focused on end users of Cloud, to provide them with application access, such that multiple users can execute the same application binary in their own virtual machine or server instance. These application sessions may be running on the same or different underlying hardware, and SaaS enables application providers to upgrade or patch their binaries in a seamless manner. Examples of SaaS providers are Salesforce.com providing Customer Relationship Management (CRM) , Google.com serving docs, gmail, etc., all of which are hosted in Cloud.
2. **Platform as a Service (PaaS)**: is focused on application developers with varying computing needs according to their project stages. These are met by servers that can vary in number of CPU cores, memory, and storage at will. Such servers are called elastic servers. Their services can auto-scale, i.e., new virtual machines can start for load balancing with a minimal administrative overhead. Examples of PaaS providers are Google's AppEngine, Microsoft's Azure, Red Hat's Makara, Amazon Web Services (AWS) Elastic Beanstalk, and AWS Cloud Formation, etc. These Cloud Service Providers (CSPs) have the capability to support different operating systems on the same physical server.
3. **Infrastructure as a Service (IaaS)**: is the bottommost layer in a Cloud stack, providing with direct access to virtualized or containerized hardware. In this model, servers with given specification of CPUs', memory and storage are made available over a network. Examples of IaaS providers are AWS EC2 (Elastic Compute Cloud), OpenStack, Eucalyptus, Rackspace's CloudFiles, etc.

Different types of Cloud Computing providers [2, 3] defined as below:

1. **Public Clouds: A Public Cloud offers its services to** a full range of customers. The public nature of this model of a Cloud is similar to Internet, i.e., users and services can be anywhere on the World Wide Web. The computing environment is shared with multiple tenants, on a free or pay-per-usage model.

2. **Private Clouds**: Private Clouds restrict their users to a select subset, usually to a specific organization within a given company. The Private Cloud is similar to an intranet, i.e., their services are provided internally via an organization's internal network.
3. **Hybrid Clouds**: Hybrid Clouds providers offer their services to a narrowly defined range of private users, which if needed, can expand to reside on a Public Cloud infrastructure. Alternatively, a public service provider can remotely manage part of the infrastructure in a private organization and use Cloud for backups.

An example of a Hybrid Cloud is Microsoft Azure Stack, deployed in an enterprise but is managed externally and if the computing requirements increase, then some tasks are migrated to an external Public Cloud. This process is often called Cloud Bursting. Clearly, Public Clouds pose the greatest security challenges due to their wider open access. In general, when a particular model is not specified, the term Cloud Computing refers to Public Clouds. In all models, a client's usage is independent of the source and location of the service provider.

Cloud customers often choose a Cloud Service on the basis of their needs:

(1) Workload Attributes:

   a. Performance;
   b. Security;
   c. Integration complexity;
   d. Data size.

(2) Business Needs:

   a. Time to deployment;
   b. Compliance regulatory;
   c. Geographical reach;
   d. Service-level agreements (SLA).

(3) Ecosystem available:

   a. Maturity of SaaS offerings;
   b. Viability of alternate services;
   c. Availability of resellers/system integrators.

1.3 Cloud Computing Operational Characteristics

In the extreme case, Cloud Computing can be defined as a service-oriented architecture (SOA) that provides any type of computing component [3]. An example is Internet-based e-mail providers, where the content for each user resides in the Cloud and the primary interface is via a browser. A user who may be traveling can access her e-mail from any location in the world by simply typing a URL, without

caring about the location of a service provider's database. However, the response time may vary depending on the physical distance and latency between the user and service provider's locations. To overcome this limitation, some international service providers use a distributed database replicating each user's e-mail information among multiple data centers in different locations. One of these is picked by intelligent routers to provide the shortest response time. At any given time, the user does not know which physical location is providing her e-mail service and thus considers it to be residing in a Cloud. Another advantage of Cloud Computing is the economy of scale. The utilization of servers inside an enterprise, a small–medium business, or even on a home computer can vary widely but rarely reaching a near 100% level at all times. In fact, averaged over 24 h, and 7 days a week, an IEEE study recently showed that most servers will show CPU utilization between 10 and 15%, and the same is true of network bandwidth [4]. Storage is also underutilized. Combining the usage of several such customers served from a single large data-center enables the operators to spread their capital investment and operational costs over a large set of customers. The sharing of resources can drive the utilization higher. Such higher utilization meets the need to fill the installed capacity with new users by allowing a flexible Cloud environment. Amazon uses this concept for their Elastic Compute Cloud [5]. This allows them to rent their servers with a reduced total cost of ownership (TCO) for their customers. This is the broadest definition of Cloud Computing, that information is stored and processed on computers somewhere else—"in the Cloud" and results are then brought back to the end-users' computer [6].

The trend toward Cloud Computing continues due to financial benefits accrued to both the users and providers. As the trend continues, Cloud-specific security issues are added to the existing security issues [7–9]. In a Cloud, Services are delivered to the end users via the public Internet, or via enterprise networks in case of a Private Cloud, without any user intervention. Private Clouds are deployed behind a firewall for an organization's internal use and enable IT capabilities to be delivered as a service. Companies often resort to using Private Clouds as they do not want to trust the Public Cloud with their confidential or proprietary information. Some key characteristics [10] of a Public or Private Cloud include:

- **Automated provisioning and data migration**: Cloud Computing eliminates the users' need to worry about buying a new server, loading an OS, and copying data to or from it when scalability is needed. Advance capacity planning is important in a traditional environment: users need to forecast their needs in advance, outlay capital budget to procure necessary hardware and software, and then wait for days and weeks before systems are brought online. In case a user under-forecasts her needs, their applications will not be available or may run slow, while over-forecasting results in wasted money. In contrast to the traditional computing environments, with Cloud Computing, users can order and get new capacity almost immediately. An analogy for fluctuating user demand and remotely installed capacity is with water and electricity utility companies.

- **Seamless scaling**: Pay as you go, instead of making an upfront investment in hardware and software, some of may be partially used, above two features allow customers to get on-demand applications and services and then pay for only what they use [5]. This is similar to households paying for electricity and water utility bills based on their monthly consumption. In fact, several services on the public Internet Cloud are available free to the end users, such as e-mail and search capabilities, and a fee is charged to the advertisers. In this model, each user is promised some space to store their mail on the Cloud but multiple users share storage and computers at the same time, often referred to as multi-tenancy on a server.
- **Increased multi-tenancy**: With the advent of multi-core computers and virtualization, it is possible for several customers to share a single server [11] with each being isolated in a separate virtual machine (VM). However, the topic of virtualization is orthogonal to Cloud Computing as some leading operators are currently not using virtualization in their data centers, but used together the benefits of these two technologies can multiply. Virtualization is used by some Cloud operators to provide immediate capacity by creating a new VM. Multiple VMs are consolidated on a single physical server to improve HW utilization. When demand decreases, any unused servers are shut down to save electricity and air-conditioning costs in a data center.

Cloud Computing is based on many prior innovations, e.g., the ability to do task consolidation and VM migration using virtualization for dynamic load balancing. It enables a service provider to optimize their datacenter usage by booting new servers and migrating some of the existing users to new machines to meet the expected response time. However, this also opens up potential security threats as mission-critical applications and confidential data from different users is colocated on the same hardware. Current methods of disk encryption are no longer deemed sufficient if one can get physical access to a competitor's servers in a third-party data-center. Physical access controls are placed on the employees working in such a data-center but there is no line of defense if one can gain access to the contents of main memory on a server. An example is the launch plan for next-generation product kept behind the firewalls, let some competitors can access it, while sharing the same server in a Public Cloud. Someone with physical access to a shared server can simply copy the files, and in a virtualized environment, even though the VMs are isolated, their run-time images are backed up at regular intervals to enable recovery or migration of a session. Hence, the backup images also need to be protected.

The benefits and some potential risks of Cloud Computing may be classified as follows:

1.3.1 Cloud Computing Benefits

1. Shared infrastructure reduces cost;
2. Pay as you go or only pay for what you use;
3. On-demand elasticity, from a large pool of available resources;
4. Increased focus on the application layer;
5. Let someone else worry about the hardware.

1.3.2 Cloud Computing Potential Risks

1. You lose direct knowledge and control of underlying hardware;
2. Noisy neighbors or other VMs sharing the same hardware can affect your performance;
3. Hard to diagnose performance issues, due to limited visibility and virtualization;
4. Potential security risks of placing your mission critical data on remote servers;
5. Vendor lock-in means getting stuck with a Cloud provider who has your data.

1.4 Cloud Computing Trends

Cloud Computing is rapidly evolving from providing end-user services (e.g., search, e-mail, social networking) to support mission-critical business and commercial grade applications as shown in Fig. 1.1. Some services for some companies are already based entirely in Cloud, e.g., Salesforce.com, and coming years will see new emerging trends, such as:

1.4.1 Trend #1: Abstraction of Network, Storage, Database, Security, and Computing Infrastructure

– This helps with software applications and data migration between Clouds.
– Offering image of on-demand, virtual data center with flexibility implied in scalability and agility.

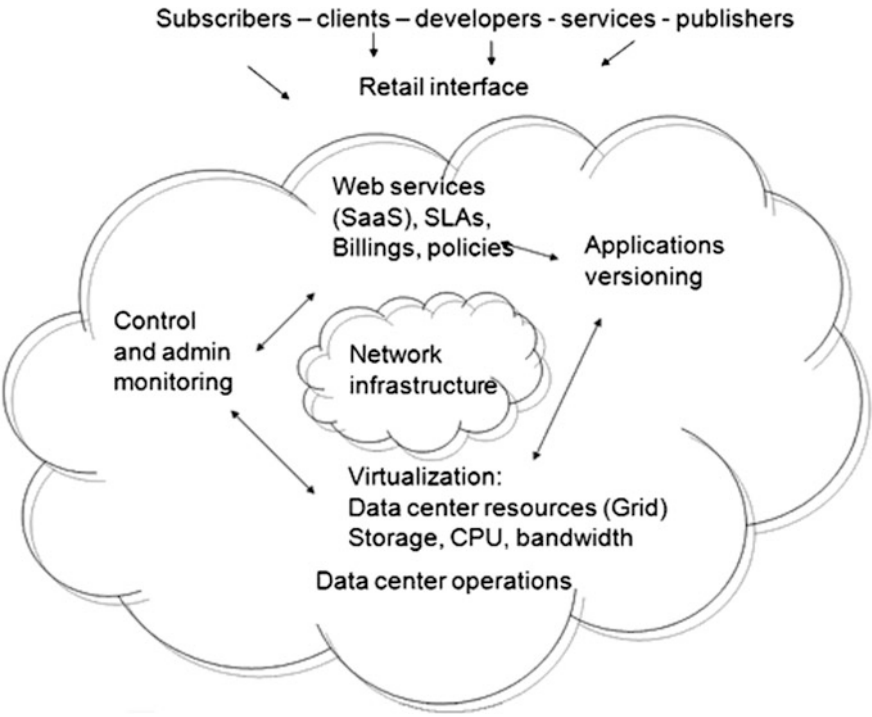


Fig. 1.1 Interaction of Cloud Computing components to provide business services

1.4.2 Trend #2: A Pricing Model that Is Retail in Its Conception

– For example, pennies per gigabyte, massive CPU cycles and bandwidth, which in turn will make computing more affordable beyond Moore's or Metcalfe's law can predict. This is due to a higher utilization of infrastructure in a Cloud data center.

1.4.3 Trend #3: Service-Level Agreements (SLAs)

Increasingly, SLAs are being used for the following purposes:

– Data persistence, system reliability, and business continuity as individual consumers may be patient to wait for their search or e-mail results but businesses need predictability to meet their goals and deliver products in a timely manner.

- SLAs imply that Cloud Service Providers will need systems in place to ensure redundancy and security of their customers' information.
- SLAs also need to cover performance aspects of the Cloud Services being provided in terms of the required computing speed and network bandwidth.

1.5 Cloud Computing Needs

Cloud Computing is still in its nascent stages during the first two decades of the twenty-first century. With the present implementations, two Clouds can be quite different from each other. A company's cluster of servers and switches can be termed as a Private Cloud, accessible only to the people inside an enterprise. Some companies have started to outsource their IT services to external Cloud providers for economic reasons, which gives rise to the expansion of Hybrid Clouds. A company can also offer both internal and external computing services. As an example, Google has internal infrastructure for its e-mail and search products and also has a Public Cloud offering. The following areas need to be addressed in order for Cloud Services to be competitive with internally dedicated servers:

- **Network Latency**: Customers are demanding high bandwidth to overcome the latency issues but in a data-flow chain, delay is determined by the weakest link. Thus, service providers are often using local caching near a customer's location. This further expands the attack surface, as multiple copies exist with the same confidential information.
- **Fine-grained Migration and Provisioning**: Solutions are needed to avoid copying gigabytes of data when a physical server needs to be brought down for planned or un-planned maintenance, e.g., to change its memory or fans. In this case, regular snapshots are taken of a VM to help with quick migration but one needs to ensure that these memory snapshots are protected to avoid compromising a user's information in the Cloud.
- **Standards and Open-source solutions for Cloud SW** infrastructure support [12] are needed as currently most commercial grade Cloud providers use their internal data representations. In case, a provider declares bankruptcy or the customer wishes to migrate her data to another provider; it is important that the information is stored in a manner that can be read by others with necessary security keys.
- **Offline versus Online synchronization of data** is needed as Internet or electric service is rarely guaranteed 24×7 in the emerging markets. Thus, users want to use devices that go beyond Internet browsers running on thin clients, which can store local copies of application and data that can run using an un-interrupted power supply and upon restoration of public utility service can sync up with the Cloud providers' database. An example of this is an accountant's office which wants to continue work on client's tax return even if the Cloud Service is not available. However, they will also want the customers' data to be protected.

Cloud Computing has found widespread adoption among public consumers with personal e-mails and photo storage. However, enterprises have hesitated to move their internal e-mail services, financial data, or product design databases to external Clouds due to real or perceived security concerns. This has given rise to internal or Private Clouds within enterprises, tucked behind corporate firewalls. For Cloud Computing to become truly universal, these real or perceived security concerns need to be outlined as described in this chapter. Until then, economic benefits of broadly shared Cloud Computing infrastructure will be slow to reduce the IT costs of enterprises.

This book is organized in 14 Chapters and 3 Appendices. In the first chapter, we began with a brief overview of the Cloud Computing domain, and then second one delves deeper in the technologies that laid down the foundations of Cloud over the preceding half-century. Then next two Chaps. 3 and 4, review the taxonomy of Cloud, followed by a classification of computing workloads that are suitable for Cloud Computing in Chap. 5. We wrap up the infrastructural topics in Chaps. 6 and 7 with a review of Cloud monitoring and security technologies. Chapter 8 starts by looking at industries that have well adapted the Clouds, while Chap. 9 studies an industry that has struggled with the concept of Cloud Computing. It should serve as a case study for any new and complex flow users who want to experiment with the Clouds. No discussion of Cloud can be complete without the economic aspects, so we visit the billing topics in Chap. 10, followed by new usage models and future trends in Chaps. 11 and 12. Finally, we wrap up the book with a short quiz, some project ideas, and appendices to cover topics that supplement the main text of this book. This includes Linux Containers, the usage of which is rising rapidly in the Cloud Computing.

Our recommended way to read this book is serially starting with this chapter onwards, but if some readers are already familiar with the basics of the Cloud, then they can take the test of Chap. 13 to judge their proficiency. After basics are understood then Chaps. 5, 6, and 7 can be reviewed to appreciate the performance and security issues related to Cloud. If an IT manager is looking to migrate complex workloads to Cloud, then they will benefit from considerations of Chap. 9. New research ideas and directions can be found in Chap. 12 and hands-on exercises in Chap. 14 will give an appreciation of actual usage scenarios. Our earnest hope is that you, our readers, will enjoy reading this book as much as we enjoyed writing it.

1.6 Points to Ponder

1. **Cloud Computing has enjoyed double-digit yearly growth over the last decade and driven mainly by economics of Public Clouds versus enterprise IT operations. Can you think of three economic reasons why Public Cloud Computing is cheaper than an enterprise DC?**

2. **Concentration of data from various sources in a few data centers may increase the risk of data corruption or security breaches. How can this possibility be minimized?**
3. **Under what circumstances an enterprise should opt to use a Public Cloud versus its own Private Cloud facilities?**
4. **Why does a Public Cloud provider want to support a Hybrid Cloud, instead of asking enterprises to use its public facilities?**
5. **Is there a scenario when Private Cloud may be cheaper than the Public Cloud?**
6. **Can you think of scenarios where a Cloud user favors (a) PaaS over IaaS PaaS and (b) SaaS over PaaS?**

References

1. Mell P, Grance T (2009) The NIST definition of Cloud computing, version 15 ed. Available: http://csrc.nist.gov/groups/SNS/Cloud-computing/Cloud-def-v15.doc
2. Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I, Zaharia M (2010) A view of cloud computing. ACM Commun 53:50–58
3. Ramgovind S, Eloff MM, Smith E (2010) The management of security in cloud computing. In: Information security for South Africa (ISSA), pp 1–7
4. Vasan A, Sivasubramaniam A, Shimpi V, Sivabalan T, Subbiah R (2010) Worth their Watts? —An empirical study of datacenter servers. In: IEEE 16th international symposium on high performance computer architecture (HPCA), pp 1–10
5. OProfile. Available: http://oprofile.sourceforge.net
6. Fowler GA, Worthen B (2009) Internet industry is on a cloud—Whatever that may mean. Available: http://online.wsj.com/article/SB123802623665542725.html
7. Kaufman LM (2010) Can public-cloud security meet its unique challenges? IEEE Secur Priv 8:55–57
8. Anthes G (2010) Security in the cloud. ACM Commun 53:16–18
9. Christodorescu M, Sailer R, Schales DL, Sgandurra D, Zamboni D (2009) Cloud security is not (just) virtualization security: a short chapter. In: Proceedings of the 2009 ACM workshop on cloud computing security, Chicago, Illinois, USA, pp 97–102
10. Soundararajan G, Amza C (2005) Online data migration for autonomic provisioning of databases in dynamic content web servers. In: Proceedings of the 2005 conference of the centre for advanced studies on collaborative research, Toranto, Ontario, Canada, pp 268–282
11. Nicolas P. Cloud multi-tenancy. Available: http://www.slideshare.net/pnicolas/Cloudmulti-tenancy
12. Bun FS (2009) Introduction to cloud computing. Presented at the Grid Asia

Chapter 2
Foundations of Cloud Computing

2.1 Historical Evolution

During the past half-century, computing technologies have evolved in several phases as described below. There appears to be a cyclic relationship between them, but in reality computing has grown as an outward going spiral.

(1) **Phase 1**: was the era of large mainframe systems in the backrooms connected to multiple users via dumb terminals. These terminals were electronics, or electromechanical hardware devices, used separate devices for entering data and displaying data. They had no local data processing capabilities. Even before the keyboard or display, capabilities were card-punching systems with JCL (Job Control Language), which was an early form of scripting language to give instructions to mainframe computers [1].
The main takeaway of this era is the concept of multiple users sharing the same large machine in a backroom and each unaware of other users. At an abstract level, this is similar to Cloud Computing with users on thin clients connected to racks of servers in the backend data centers.

(2) **Phase 2**: was the era starting in the 1980s with personal computers (PCs), many of which were stand-alone, or connected via slow modems [2]. Each user interacted with a PC on a 1-on-1 basis, with a keyboard, mouse, and a display terminal. All the storage, computing power, and memory were contained within a PC box. Any needed software was installed via floppy disks with limited storage capacity to run on the PCs. These PCs evolved in the early 90s to laptops with integration of display, keyboard, mouse, and computing in a single unit. There was also an attempt in the early 90s to create a network computer, which was diskless and connected to more powerful computers in the backend, but perhaps the idea was before its time as networks were still slow with 28.8 kbit/s dialup modems [3]. A more practical solution in terms of Client–Server solution [4], which enabled remote devices with a little computing power to connect with servers over corporate Ethernet networks, became prevalent.

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6_2

The main takeaway of this era was the birth of an online desktop to emulate the desk of working professionals. It represented multiple tasks that were simultaneously kept in an open state on the PC. With GUI and OS utilities, it was possible to create the notion of a desktop on a computer, to go from a single user–single job model to single user–multiple jobs running simultaneously. This caused user interactions to move from command prompts to mouse-driven clicks.

(3) **Phase 3**: in the mid 90s was the era of Web browsers [5], which is a software application to retrieve, present, and traverse information resources on the World Wide Web (WWW). These came out of a research project, but became popular with everyday computer users to access information located on other computers and servers. Information often contained hyperlinks, clicking which enabled users to traverse to other Web sites and locations on the WWW. These browsers needed full-fledged PCs to run on, which were more than dumb terminals so the evolution cycle was not yet complete.
The main takeaway of this era was the birth of WWW, with PC forming a gateway for connecting users to the Internet. It used Internet infrastructure to connect with other devices through TCP/IP protocol, for accessing a large set of unreliable resources and get a reliable output.

(4) **Phase 4**: New century heralded an era of full-fledged Internet browsing with PCs and a mobility revolution with cell phones abound. It was inevitable that the twains shall meet launching innovative mobile applications running on cell phones. Cell phones created a yet another gateway to Cloud. This enabled users to book hotels, rent rooms, and buy goods on the move.
This main takeaway of this era was the birth of mobile clients, similar to Client–Server model, except with limited compute power in small form factors. Thousands of powerful servers were located in large, far-flung data centers. It may be noted that this represented one revolution of spiral akin to mainframes, as depicted in Fig. 2.1.

(5) **Phase 5**: When companies discovered that number of smartphones and mobile devices they can sell is limited by the population of the world, they started to look for new business opportunities. These came in the form of Internet of Things (IoT), which enables everyday common objects such as a television, a refrigerator, or even a lightbulb to have an IP (Internet Protocol) address. This gives rise to new usage models, such as to conserve energy, IoT objects can be remotely monitored, turned on or off for energy conservation, etc. This also includes consumer services for transportation and a utilitarian phase for user interactions with appliances, leading to higher productivity and improved quality of life. Computing reach extended to a better-informed decision-making and into social relationships.
This era completes the virtual cycle of computing evolution, with many front-end dumb devices connected to powerful servers on the backend, as shown in Fig. 2.1.

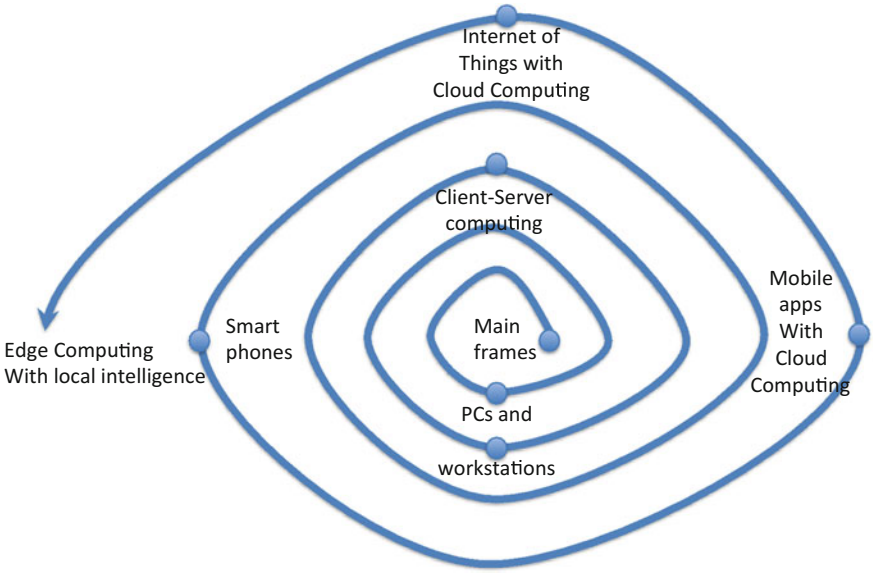


Fig. 2.1 Evolution of computing models in a spiral of growing computing needs

2.2 Different Network Protocols

Two models prevailed in the networking domain: peer-to-peer and Client–Server. In the former, each computer that wants to talk to another computer needs a dedicated line, so *N* machines will need *N*^2 connections. This while being fast was obviously not scalable; also when no data is being transferred, then network capacity is un-utilized. Meanwhile, the Client–Server model with a central server supporting multiple clients was economical and scalable for the enterprises, as data and files could be easily shared between multiple users as shown in Fig. 2.2.

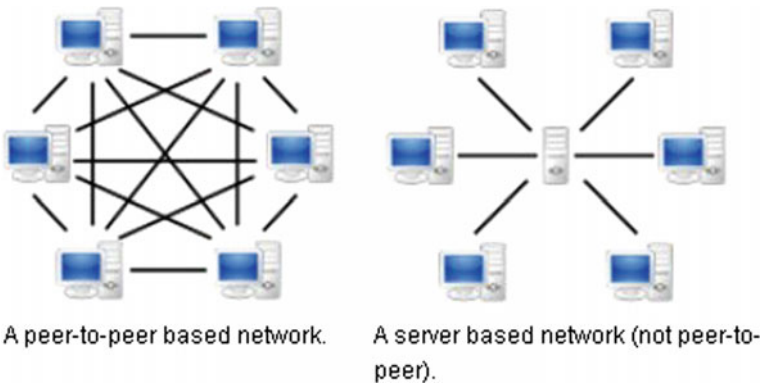


Fig. 2.2 Two networking models

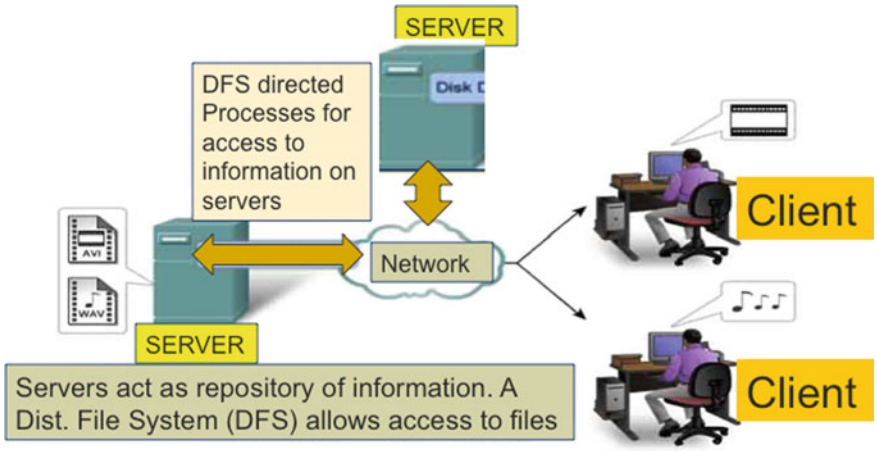


Fig. 2.3 The networking model

On networked systems with a Client-Server model, a computer system acting as a Client makes request for a service, while another computer acting as a Server provides a response when a request is received, as shown in Fig. 2.3. The greatest advantage of the networking model was its scalability; i.e., one could add any number of servers and clients. It offered an opportunity to create open-source software that offered interoperability. The network protocol was based on TCP/IP, and going forward, this growth led to several interconnecting networks, eventually into Internet using http protocol. The protocol-based control meant that one could operate without a vendor lock-in, i.e., not be dependent on any single supplier. Networks within an enterprise grew as LAN (local area network) and WAN (wide area network). LAN connected computers in confined areas, such as within a single office building, or there could be several LANs, e.g., one for each floor. WAN connected computers spread over larger geographical areas, such as across the cities. Several LAN Standards were developed over time, such as IEEE 802.2, Ethernet or IEEE 802.3, IEEE 802.3u for faster Ethernet, IEEE 802.3z and 802.3ab for Gigabit Ethernet, IEEE 802.5 for token rings, and IEEE 802.12 for 100VG on any LAN. The Internet in turn is simply a net of networks, using TCP/IP suite of protocols. These were mostly packet switching protocols, with each packet having a structure offering a virtual connection, and no single dedicated path committed to maintain the connection between a source and its destination. Paths were temporarily created to maintain connections as needed between users and/or machines. Sometimes, time division multiplexing was used to share the same physical line between different pairs of users, and every pair was given a short time slot. As an example, if a time unit has 10 slots, then 10 pairs can communicate simultaneously, oblivious of the presence of others. Connections are made over a network of switched nodes, such that nodes are not concerned with content of data. The end devices on these networks are stations such as a computer terminal, a phone, or any

other communicating device. Circuit switching operates in three phases, to establish a connection, transfer data, and then disconnect. Networking routers used an intelligent algorithm to determine optimal path that takes the least time, or to minimize cost to establish a path of communication.

TCP/IP is a fault-tolerant protocol, so if a packet is lost or corrupted, then the connection is retried and packet is sent again. It was originally designed by DARPA (Defense Advanced Research Projects Agency) during the Cold War era to survive a nuclear attack. TCP/IP is a layered protocol where TCP provides reliability of a connection by attempting to send the same packet again if the previous transmission failed, while IP provides routability between the communicating nodes by finding an optimal path.

When computing needs exceed beyond what can be reasonably supported by a single server, an effort is made to share the workload among multiple servers. Note that in sharing a workload, the latencies on servers that are loosely connected shall be determined by TCP/IP suite of protocols. The concern often is that these latencies would be in far excess of what may be acceptable. Therefore, an alternate approach is to use server clusters. The latency in tightly bound server clusters is far less than the networked servers. One of the techniques for clustering is by sharing I/O switches between the servers [1]. The switches connect individual class of controllers such as Ethernet controller in an enterprise network, as shown in Fig. 2.4.

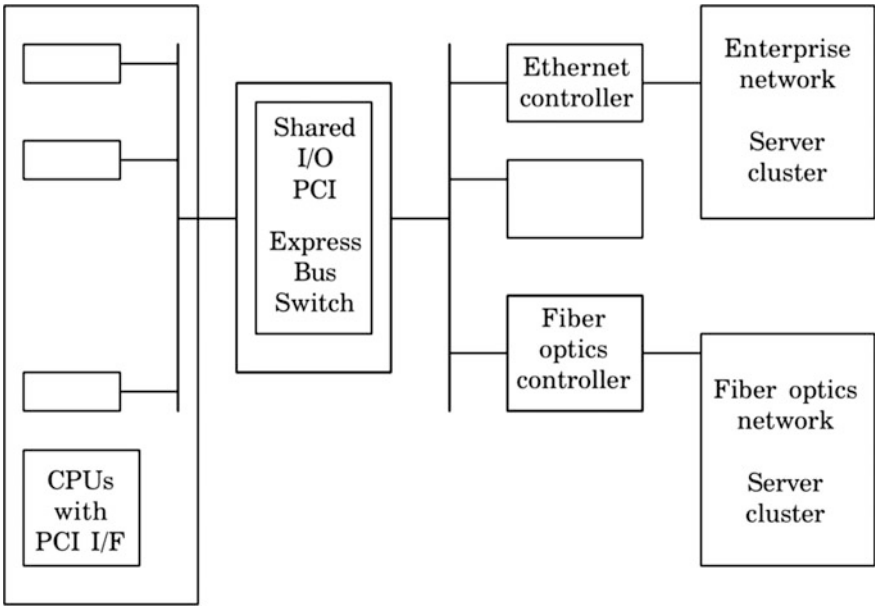


Fig. 2.4 Connecting multiple servers in a cluster

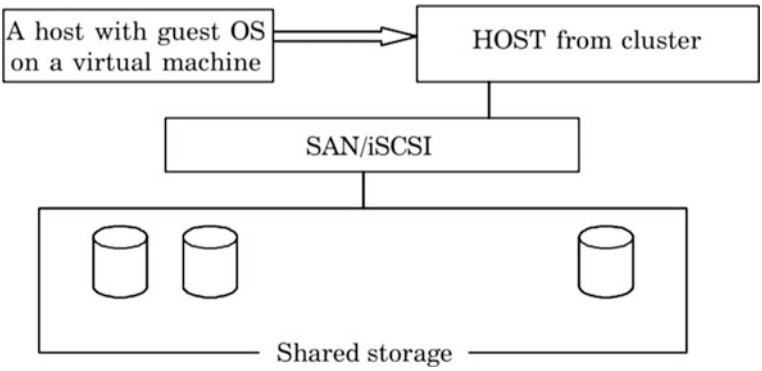


Fig. 2.5 Connecting multiple storage elements in a cluster

Clustering ensures that large applications and services are available whenever customers and employees need them. This enables IT managers to achieve high availability and scalability for mission-critical applications, such as corporate databases, e-mail, Web-based services and support external facing retail Web sites. Clustering forms the backbone of enterprise IT's servers and storage elements. Clustering may operate with SAN (storage area network) or iSCSI (Internet Small Computer System Interface) architectures [1] as shown in Fig. 2.5.

2.3 Evolution of Enterprise IT

Enterprise computing refers to business-oriented information technology that is critical to a company's day-to-day operations. This includes various types of activities, such as product engineering, accounting, human resources, and customer services using software for database management and other service-specific modules. One key aspect of enterprise computing is its high availability, as crash of hardware or software component can lead to loss of revenue and customer base. Thus, multiple redundant computers exist for mission-critical applications, as well as regular data backups are taken to ensure that there is no single point of failure within an enterprise.

IT has evolved over the decades to closely align with business units (BUs) within a large corporation. A BU is typically responsible for providing a product or service to the external world, whereas the IT department has mostly internal interfaces to ensure a smooth flow of information and data. IT is the lifeblood of a successful enterprise. Any problems in IT can bring a corporation to halt, such as e-mail disruptions, but worse yet security vulnerability in an IT system can cause loss of valuable information. As an example, Target's security and payment system was hacked, leading to loss of Target's customers' information [6]. Hackers came through the HVAC (heating, ventilation, and air-conditioning)

system and installed malware, which targeted "40 million credit card numbers—and 70 million addresses, phone numbers, and other pieces of personal information." About six months before this happened, Target invested $1.6 million to install a malware detection tool made by FireEye; their security product is also used by the CIA. FireEye's tool is reasonably reliable software, and it spotted the malware, but it was not stopped at any level in Target's security department. Target's unresponsiveness to the alerts resulted in the exposure of confidential information of one in three US consumers.

Much less dramatic examples of other IT failures include lack of scalability. For example, in August 2016, a computer outage at the Delta Airlines headquarters in Atlanta prompted the airline to cancel about 2300 flights, delaying the journeys of hundreds of thousands of passengers and prompting three days of chaos [21]. Delta executives said a malfunction in an aging piece of equipment at its data center had caused a fire that knocked out its primary and backup systems. The carrier said the system failure reduced revenue by $100 million in the month of August 2016.

Examples abound in other key economic sectors also. At the NYSE (New York Stock Exchange), trading was suspended for nearly four hours on July 8, 2015, after a botched technology upgrade, freezing one of the world's biggest financial markets. The NYSE blamed a technical "configuration problem" that was subsequently fixed [20].

One way to deal with an unavoidable hardware or software failure is the rise of distributed computing. In such a system, components are located on networked computers and interact with each other to achieve a common goal. The goal is to provide Service with an acceptable level of reliability and timeliness. This became possible with the advent of service-oriented architecture in enterprises, which we will study in a subsequent section.

2.4 Evolution of Web Services

A Web Service is an interface described by some form of service from a remote provider. These are based on "Open" and "Standard" technologies, such as HTTP, HTTPS, XML, REST. These evolved from Client–Server and distributed computing concepts, to offer a "Web of Services," such that distributed applications can be assembled from a Web of software services in the same way that Web sites are assembled from a Web of HTML pages [7].

The advent of the Internet is based on a set of open standards, some of which are:

1. **TCP/IP**: Transmission Control Protocol/Internet Protocol, for network applications to exchange data packets in real time. Data is packed in byte packages, ranging up to 64 K (65,535 bytes). These are sent and acknowledged, and if not acknowledged, then sent again with multiple retries, until each packet arrives at the destination. These packets are then reassembled to create a complete copy of the information content in whole.

2. **RPC**: Remote procedure call allows functions written in C, Java, or any other procedural language to involve each other across a network, allowing software services to reach other servers across the network.
3. **HTTP**: Hypertext Transport Protocol, for sharing data between machines based on top of TCP/IP protocol.
4. **HTML**: Hypertext Markup Language, the format for representing data in a browser-friendly manner.
5. **XML**: stands for eXtensible Markup Language. It enables any data to be represented in a simple and portable way. Users can define their own customized markup language, to display documents on the Internet.
6. **SOAP**: Service-oriented architecture protocol, for connecting computers. It allows message passing between end points and may be used for RPC or documents transfer. These messages are represented using XML and can be sent over a transport layer, e.g., HTTP or SMTP. An example of communication using SOAP and XML is shown in Fig. 2.6.
7. **UDDI**: Universal Description, Discovery, and Integration is an XML-based registry for businesses to list themselves on the Internet. Its can streamline online transactions by enabling companies to find one another on the Web and make their systems interoperable for e-commerce.
8. **WSDL**: Web Services Description Language is used to describe a Web Service in a document. It uses an XML format for describing network services as a set of end points operating on messages. These can contain either document-oriented or procedure-oriented information. The operations and

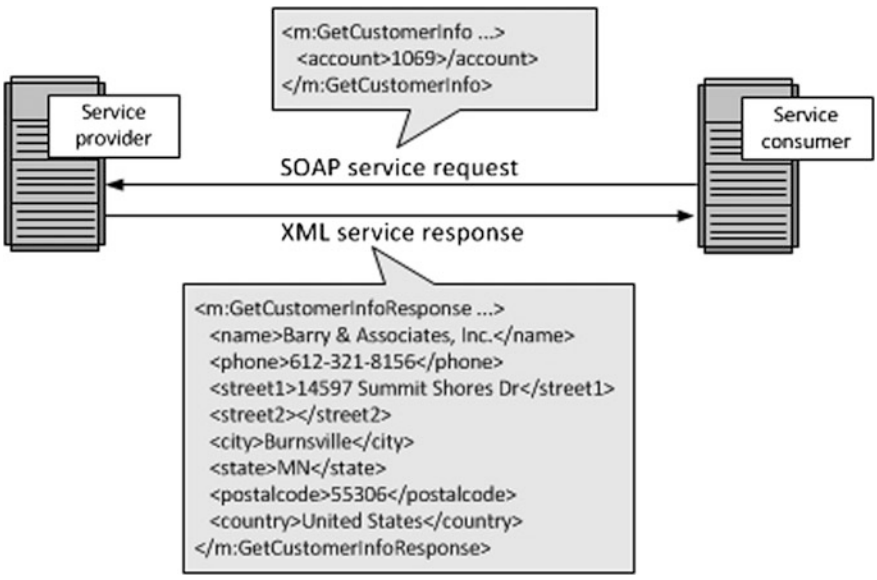


Fig. 2.6 Interacting using SOAP and XML [15]

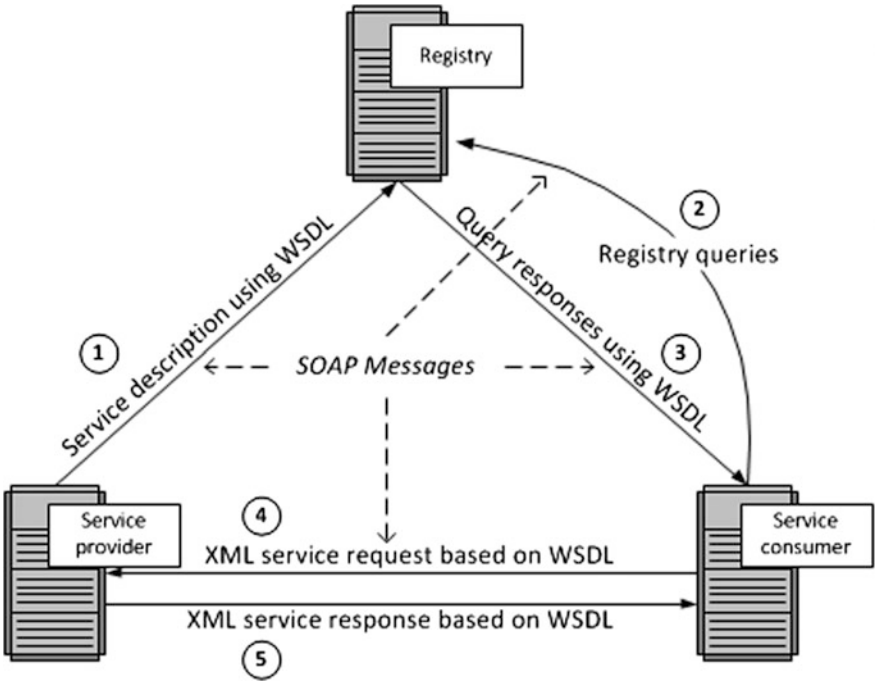


Fig. 2.7 An example of using WDSL [9]

messages are described abstractly and then bound to a concrete network protocol and message format to define an end point. The Web Services Description Language (WSDL) forms the basis for the original Web Services specification. Figure 2.7 illustrates the use of WSDL. At the left is a service provider. At the right is a service consumer. The steps involved in providing and consuming a service are described below:

I. A service provider describes its services using WSDL. This definition is published to a repository of services. The repository could use Universal Description, Discovery, and Integration (UDDI) as a way to publish and discover information about Web Services. UDDI is a SOAP-based protocol that defines how other UDDI clients communicate with registries. Other forms of directories could also be used.
II. A service consumer issues one or more queries to the repository to locate a service and determine how to communicate with that service.
III. Part of the WSDL provided by the service provider is passed to the service consumer. This tells the service consumer what the requests and responses are for the service provider.

Fig. 2.8 Interaction of two computers using REST [9]

IV. The service consumer uses the WSDL to send a request to the service provider.
V. The service provider provides the expected response to the service consumer.

9. **REST**: Representational state transfer is a protocol used to create and communicate with the Web Services. REST is language independent. Developers prefer REST due to a simpler style that makes it easier to use than SOAP. It is less verbose so less data wrappers are sent when communicating. An interaction is illustrated in Fig. 2.8.
10. **JSON**: JavaScript Object Notation uses a subset of JavaScript. An example is shown in Fig. 2.9. It uses name/value pairs and is similar to tags used by XML. Also, like XML, JSON provides resilience to changes and avoids the brittleness of fixed record formats. These pairs do not need to be any specific order.

2.5 Server Operations in a Data Center

The server stores or has access to a database, for which clients initiate queries from terminals. As an example, type of software called "DBMS" (database management system) manages the storage of information, often stored as a set of tables (files). Access to retrieve desired information is via a query processor (SQL queries). Using the previously explained networking technology and Client–Server model, following are some examples of servers' operations:

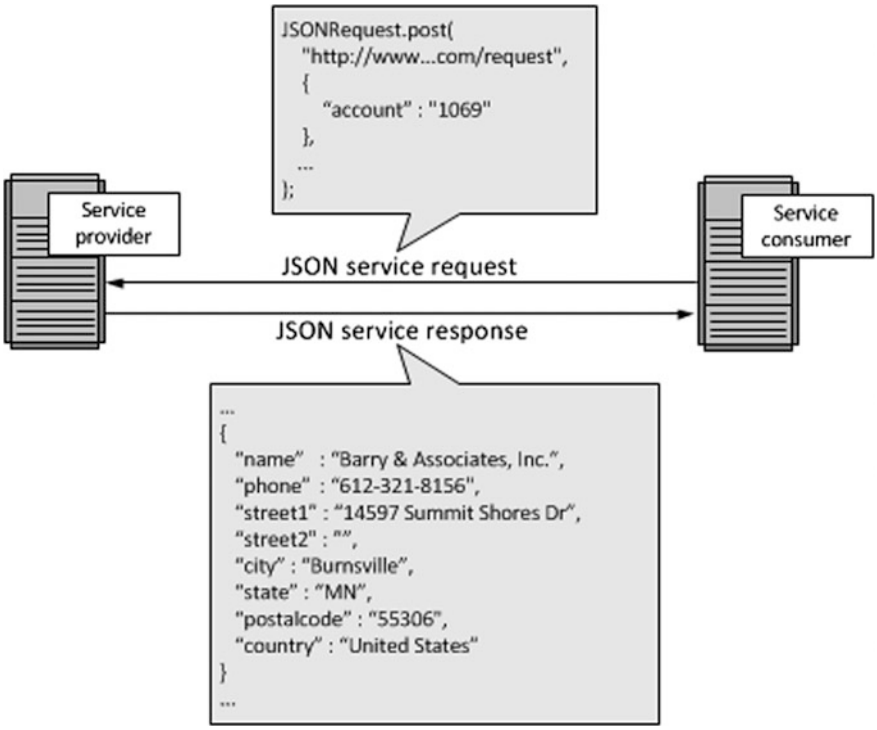


Fig. 2.9 Interaction of two computers using JSON [9]

- Print Servers
  - Over a LAN or remote printing, by different clients to one or more set of printers.
- FTP (File Transfer Protocol):
  - Client may download or upload,
  - 7-bit ASCII character set,
  - Text or images,
  - Commands and responses
- DNS (Domain Name Server): for resolving IP addresses.
- Mail: below are various protocol processing services offered by mail servers
  - SMTP
  - POP
  - IMAP
  - POP3

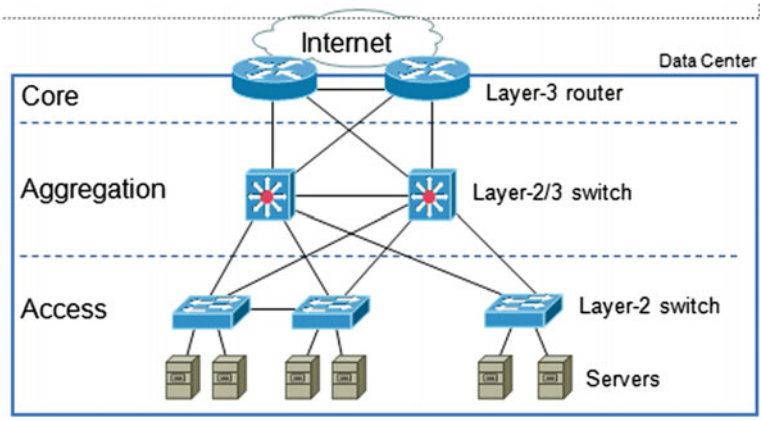


Fig. 2.10 Organization of a data center

- Media
  - Streaming and buffering
- Security Servers: for issuing and maintaining encryption keys, digital certificates, user transaction, and monitoring logs.

Development of GUI-based browsers using HTTP led to an infrastructure that we currently call World Wide Web (WWW). This led to Web Services and various applications, such as search engines, e-commerce, and banking Apps.

To support many servers in one place, in turn serving many users spread across a large areas, one needs a lot of servers, routers, and switches. These server clusters are also called data centers (DCs). The sheer density of equipment means the DCs need special cooling equipment to dissipate the heat generated by the equipment. Typically, $1 spent to power a server also needs another $1 to cool it. A data center can be hierarchically organized, as shown in Fig. 2.10.

Note that different layering of switches is done for efficiency reasons. At layer 2, one single tree spans the access of local server racks, to prevent looping, and ignores alternate paths. At layer 3, aggregation is done with shortest path routing between source and destination, with best-effort delivery of packets. A failure at this level can be catastrophic so backups are provided with the appropriate trade-off between cost and provisioning services.

These servers need to have massive storage, connectivity, and traffic management capability using switches. Of these, storage is a key capability required to provide large amounts of data needed by remote users, for which SNIA (Storage Networking Industry Association) has recommended storage architecture, as depicted in Fig. 2.11. It consists of a layered architecture, starting from bottom going upward as:

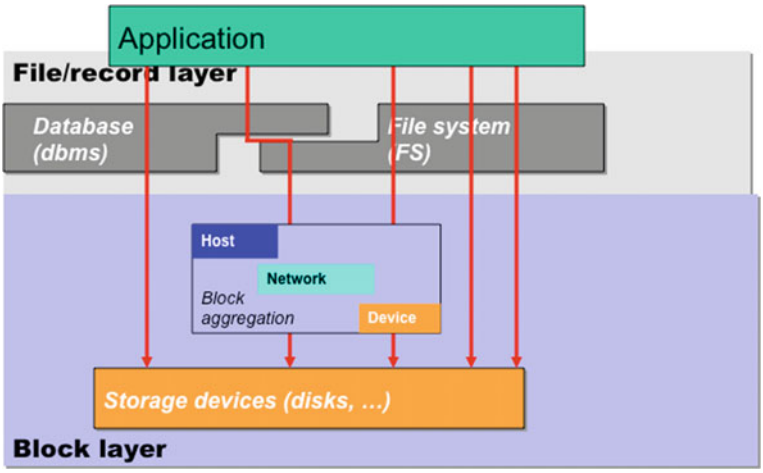


Fig. 2.11 SNIA recommended storage architecture for large data centers

1. Storage devices
2. Block aggregation layer with three functional placements:
   a. Device,
   b. Network,
   c. Host.
3. File/record layer
   a. File system,
   b. Database.
4. Applications at the top to use the storage.

Key advantages of the SNIA-proposed architecture include scalability and 24 × 7 connectivity. The former is needed to support easy addition of storage volumes as data needs grow, while the latter is needed to provide reliable, distributed data sharing with high performance and no single point of failure.

Web Services are provisioned from a client's browser to access applications and data resident on remote servers in a remote data center. Figure 2.12 represents this linkage with the following architecture, often using thin clients that:

- Offer reduced energy consumption;
- May not have any native SW application, other than a browser;
- May not support a file system;
- Hence, no SW distribution licenses are required on the client side;
- Applications running on the server side.

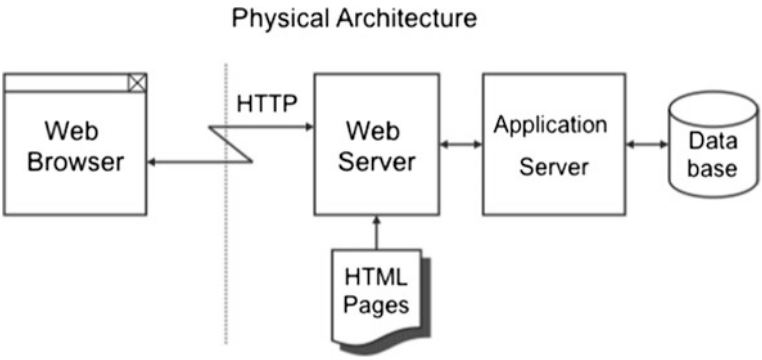


Fig. 2.12 A representation of how clients can communicate with servers

Cross-Platform compatibility is provided via browsers across OSes' such as Linux and Windows with

- Support for standard networking protocols
- HTTP (for browser communications);
- HTML (for rendering).

Clients select the type of information needed, access it remotely on servers, and then display it locally. The net result of this setup is that at an airport, one can get flight info, such as timing, reservation status using a smartphone application or browser. Similarly, terminals at public libraries allow patrons to browse services offered by the library. This is often done with thin clients, which are minimally networked computers using

- Thin Client Java Viewer,
- Browser,
- Embedded ActiveX Control.

However, a thin client has several limitations such as reduced computing, so a server will need to validate the inputs, slowing the server down. Also, with no native software, client-side GUI design limits most server interactions to be of pull nature and displays the incoming data as is, unable to do any computations or processing on the client-side. Lastly, HTTP is a packet-based connection-less protocol with limited reliability. Despite these limitations, thin clients' usages abound due to their cheaper costs, such as Chrome netbooks, mobile phones, and tablets. As we will see in a later chapter, these have also became the basis of even thinner clients in form of sensors and Internet of Things (IoT) to connect regular household or industrial devices directly to Internet.

2.6 Server-Based Web Services

An agent running on server hardware provides a service using a Web-based App and fits the basic paradigm of Client–Server computing, as reillustrated in Fig. 2.13. In short, a Web Service is an interface described by some form of service from a remote provider. The actual phrase, "Web Services," came from Microsoft in the year 2000 and was adopted by W3C. The idea was to offer services using "Open" and "Standard" technologies.

Various service components can be bound together to offer an integrated service, an example of which is a vacation planner. Given a date, desired location, and hotel requirements, the Web agent can search and come back with options for transportation and hotel arrangements. To enable this various airlines and hotels from different areas can register with the vacation planner, to offer an integrated services package.

A Web App is the software aimed at an end user and can be used from a remote client, providing a dedicated program-to-program communication-based service. It operates with formatted messages and instructions. Examples of Web applications include Internet banking or the Uber service App. These allow users to communicate from different operating systems such as Windows or Linux, using different devices such as a personal computer or a smartphone. The open standards include HTTP or HTTPS for transportation of HTML or XML messages. In these scenarios, the client side is running a Web browser, such as Chrome, whereas the server side is running a Web server such as Apache, enabling multiple users to interact with one or more servers in a remote data center. The size of the attack target in a Web Service is quite large, as the attacker can compromise the browser running on a client device, do a passive or active attack on the network in-between, or target the server in a data center. We will examine each of these in the next section.

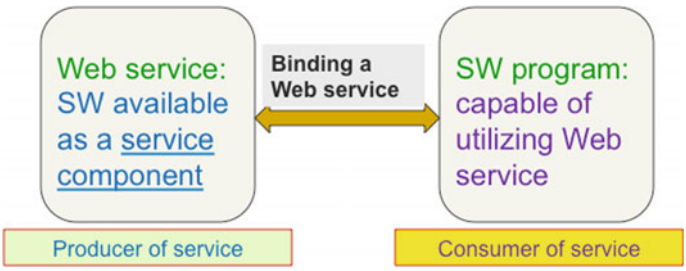


Fig. 2.13 Web Services based on a Client–Server paradigm

2.7 Service-Oriented Architecture

A service-oriented architecture (SOA) is a style of software design where services are provided to the other components by application components, through a communication protocol over a network [3]. The basic principles of service-oriented architecture are independent of vendors, products, and technologies. A service is a discrete unit of functionality that can be accessed remotely and acted upon and updated independently, such as retrieving a credit card statement online.

A service has four properties according to one of many definitions of SOA:

1. It logically represents a business activity with a specified outcome.
2. It is self-contained.
3. It is a black box for its consumers.
4. It may consist of other underlying services.

Different services can be used in conjunction to provide the functionality of a large software application. So far, the definition could be a definition of modular programming in the 1970s. Service-oriented architecture is about how to compose an application by integration of distributed, separately maintained, and deployed software components, as shown in Fig. 2.14. It is enabled by technologies and standards that make it easier for components to communicate and cooperate over a network.

Thus, SOA helps to organize discrete functions contained in enterprise applications into interoperable, standards-based services that can be combined and

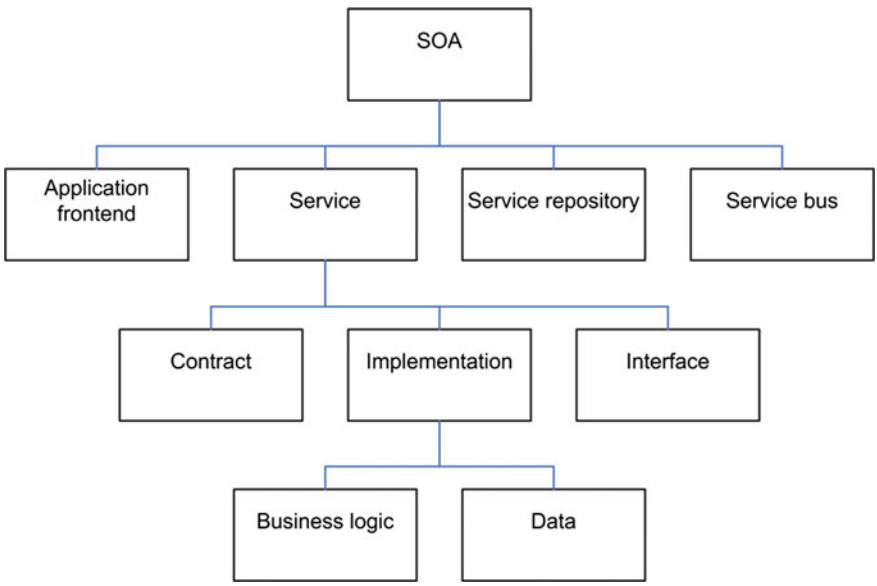


Fig. 2.14 Elements of service-oriented architecture

Fig. 2.15 The six SOA domains [8]

reused quickly to meet business needs [8]. By organizing enterprise IT around services instead of around applications, SOA provides following key benefits:

- Improves productivity, agility, and speed for both business and IT;
- Enables IT to deliver services faster and align closer with business;
- Allows the business to respond quicker and deliver optimal user experience.

Six SOA domains are depicted in Fig. 2.15 and enumerated below:

1. Business Strategy and Process: By linking an organization's business strategy with IT management and measurement, SOA enables continuous process improvements.
2. Architecture: By providing an IT environment based on standards, SOA enables integration of functionality at enterprise level.
3. Building blocks: By reusing implementation and core infrastructure, SOA provides consistency and repeatability of IT's success.
4. Projects and Applications: By cataloging and categorizing functionality of applications and systems across an enterprise, SOA drives out redundancy and promotes consistency in business execution.
5. Organization and Governance: By standardizing delivery of IT services, SOA ensures maximal reuse of any developed functionality.
6. Costs and benefits: By planning and reusing functionality, SOA ensures that existing IT investments are leveraged for a sustainable enterprise value.

Business objectives, such as a policy for retail banking, financing, individual loans, are achieved by SoA regardless of technology and infrastructure. The specific solution architecture may include:

1. Retail banking solution,
2. Financing solution,
3. Personal loans management system, etc.

Technical architecture for SoA would cover data abstraction, messaging services, process orchestration, and event management or monitoring. A service comprises of a stand-alone unit of functionality available only via a formally defined interface. A mature rollout of SOA effectively defines the API of an organization. Reasons for treating the implementation of services as separate projects from larger projects include:

1. Separation promotes the decoupling of services from consuming projects.
2. Encourages good design as the service is designed without knowing the consumers.
3. Agility fosters business innovations and speeds up time-to-market.
4. Documentation and test artifacts of the service are not embedded within the detail of the larger project.
5. Enables reuse of components later on.

SOA also promises to simplify testing indirectly. Services are autonomous, stateless, with fully documented interfaces and separate from the concerns of the implementation. A full set of regression tests, scripts, data, and responses is also captured for the service. The service can be tested as a "black box" using existing stubs corresponding to the services it calls. Each interface is fully documented with its own full set of regression test documentation. It becomes simple to identify problems in test services.

However, SOA suffers from drawbacks, as "state-full" services (i.e., where the memory state of previous transactions needs to be preserved) require both the consumer and the provider to share the same consumer-specific context. This could reduce the overall scalability of the service provider if the service provider needs to retain the shared context for each consumer. It also increases the coupling between a service provider and a consumer and makes switching service providers more difficult.

2.8 Building an Enterprise SOA Solution

Applications built using SOA style can deliver functionality as services. These can be used or reused when building applications or integrating services within the enterprise or trading partners. These services are the building blocks of business flows, as shown in Fig. 2.16. A service may be simple, such as "get me a person's

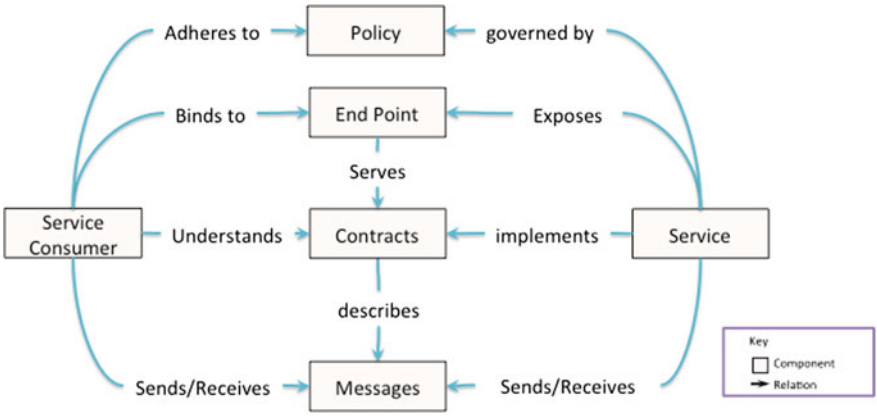


Fig. 2.16 An SOA service provisioning with contract

address," or complex as "process a cheque and make the payment," etc. It ensures that business function is executed consistently and within the quality of service parameters, to return predictable results.

SOA allows integration by using open standards, which ensure compatibility across enterprise boundaries. Consumer of data is expected to provide only the stated data on the interface definition, and service handles all processing, including exceptions, e.g., if an account has insufficient funds to pay in exchange of a cheque.

SOA services are stateless, e.g., they do not maintain any memory between invocations. Each service takes the input parameters, execute a previously identified process, and return the results. If a transaction is involved and executed properly, then data is saved to the database (e.g., checking amount is updated after a payment). Users of the service do not need to worry about the implementation details for accessing the service. This is all done with loosely coupled building blocks with well-defined integration points, such that any changes can be made to evolve the service. An example is to allow ATM and electronics transactions on a bank account in addition to paper check processing. SOA service is carried from end-to-end (E2E), and no context is maintained between different service invocations. Rollbacks are implemented if a particular transaction fails, e.g., once a check is deposited in an ATM, it is processed through the depositor's account and a message is sent to the originator's bank. Even if a temporary credit is given to the depositor, most of it is on hold for a few days until the check clears. If check writer does not have sufficient funds, then the check is returned and any temporary credit is rolled back. Similarly, a withdrawal process from an ATM machine first checks if the account holder has sufficient funds, deducts the amount, and then dispenses the cash. If for any inventory or mechanical reasons, cash cannot be given out, then the transaction rolls back and the account is credited resulting in no withdrawal. This ensures that each transaction is complete and all possible failure conditions are

accounted for, leaving the database (in this case, bank account) in a consistent and correct integrity state for the next transaction to take place.

2.9 Top-Down Versus Bottom-Up Approach

Just like buildings are planned with a top-down approach, a similar architectural style works well for SOA. Purpose is to avoid redundancy between services and constituent applications. Although, combining it with a bottom-up approach and with a reuse of existing structure is pragmatic. An example in Fig. 2.17 shows a layered solution. It describes business processes at the top layer, service interfaces at the next layer, and implementation points at the lowest layer to connect applications written in different languages, such as .NET, J2EE, and legacy C language. It supports automation of flexible, adaptable business processes by composing loosely coupled services. As we noted in a previous section, Web Services frameworks are evolving with numerous WS-* specifications that be composed using SOAP messaging.

This multilayered architecture demonstrates the decoupling of an end user on a front-end terminal from the backend servers in a Cloud. The presentation layer in the Cloud formulates and sends HTML messages to a browser on a client device, often without knowing the type or make of the client machine. This allows the

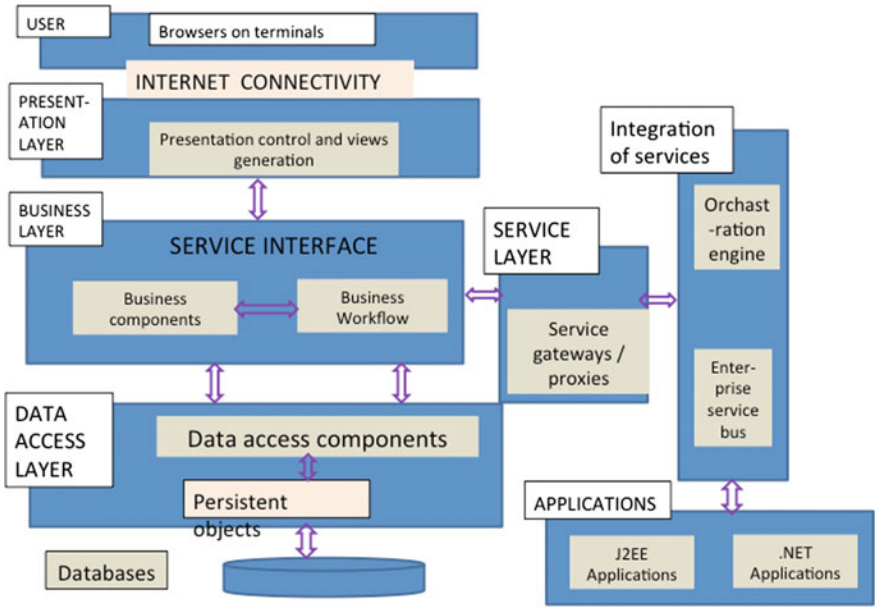


Fig. 2.17 A multilayered reference model for an enterprise

browser to update only specific frames, such as weather information or a share's price, while keeping other display parts constant such as the name of the cities or number of shares. It minimizes the traffic across distant Internet connection, while speeding up the necessary communications. On the Cloud side, the motivation to decouple presentation and business layers is to simplify the task of rendering and serving information, from the rules of who can access which kind of information. An example is need-to-know-based access, e.g., for medical records. A patient can read their own records but not modify them, whereas a caregiver can add but not delete any notes, while a doctor may be able to edit the medical records to alter the medical treatment direction. Data is often kept on a different server, with provisions to create backups and for access by multiple users at the same time. This is useful for serving content of common interest from one to many, such as news and movies. It enables scalability when number of users increases by adding more servers in the business logic or database layers. SOA defines the tasks for each layer and protocols between them as indicated by bidirectional buses.

A top-down design starts with inventory analysis. Just like for a house, the blueprint SOA is established. An SOA design can cross enterprise and Internet boundaries to provide compounded services, supported by well-defined service-level agreements (SLAs) between the organizations. Next step is the design of Service Contracts, which includes inputs and outputs between different entities involved. It is followed by Service Logic, which specifies tasks performed by each individual service block. Then, each service is further developed, and it may be composed of sub-services. An example is shown in Fig. 2.18, which describes orchestration of custom services for a travel agency's portal with multiple service providers and consumers.

It integrates services [9] from airlines and hotels, offering a one-stop shop to make travel and staying arrangements for a business or vacation trip. Input–output specification for a service block allows it to be tested in a stand-alone manner before integration in the SOA flow. After each block has been confirmed to work per its specification, the service can be deployed. Lastly, service governance is performed to ensure that overall SOA delivery is happening per the contractual terms. Complex business processes are defined using a Business Process Execution Language (BPEL) in a portable XML format. This describes how services interact to form complex business process, fault handling, and transaction rollback. A top-down architectural approach reduces the overall burden of subsequent service governance, because services were modeled as a part of an inventory.

2.10 Enterprise Service Bus (ESB)

In its simplest form, an ESB delivers a message from one point to another. It is middleware for connecting heterogeneous services and orchestrating them. This is done with loose coupling, a location transparency in a transport neutral manner. The messages can be XML document, sent using SOAP, as shown in Fig. 2.19.

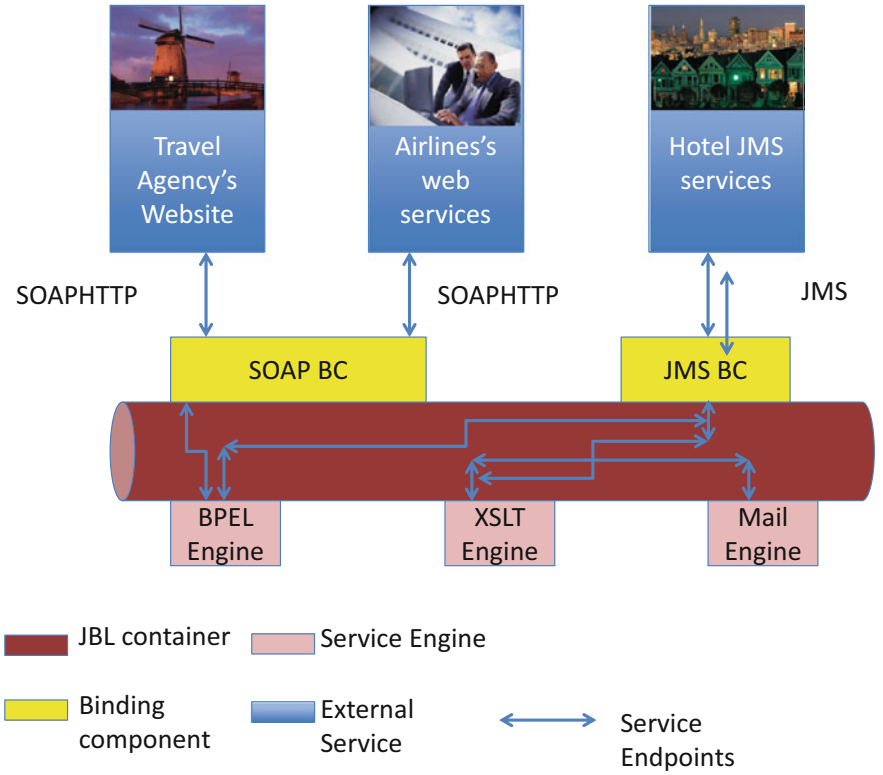


Fig. 2.18 A travel agency conceptual architecture

ESB can also be thought of as an abstraction layer on top of an enterprise messaging system, with the following key characteristics:

- Streamlines development;
- Supports multiple binding strategies;
- Performs data transformation;
- Intelligent routing;
- Real-time monitoring;
- Exception handling;
- Service security.

The key values of using an SOA-based solution are that it is not tightly coupled and has clean integration points and a flexible architecture, which is amenable to changes. An ESB allows for intelligent routing of messages, with real-time monitoring and service security.

Benefits of SOA initiatives should be measured over years rather than months or quarters. As shown in Fig. 2.20, initial investment may be higher than the traditional IT approaches to develop nonstandard applications [8]. These will payoff over time as the number of new capabilities using SOA building blocks grows due

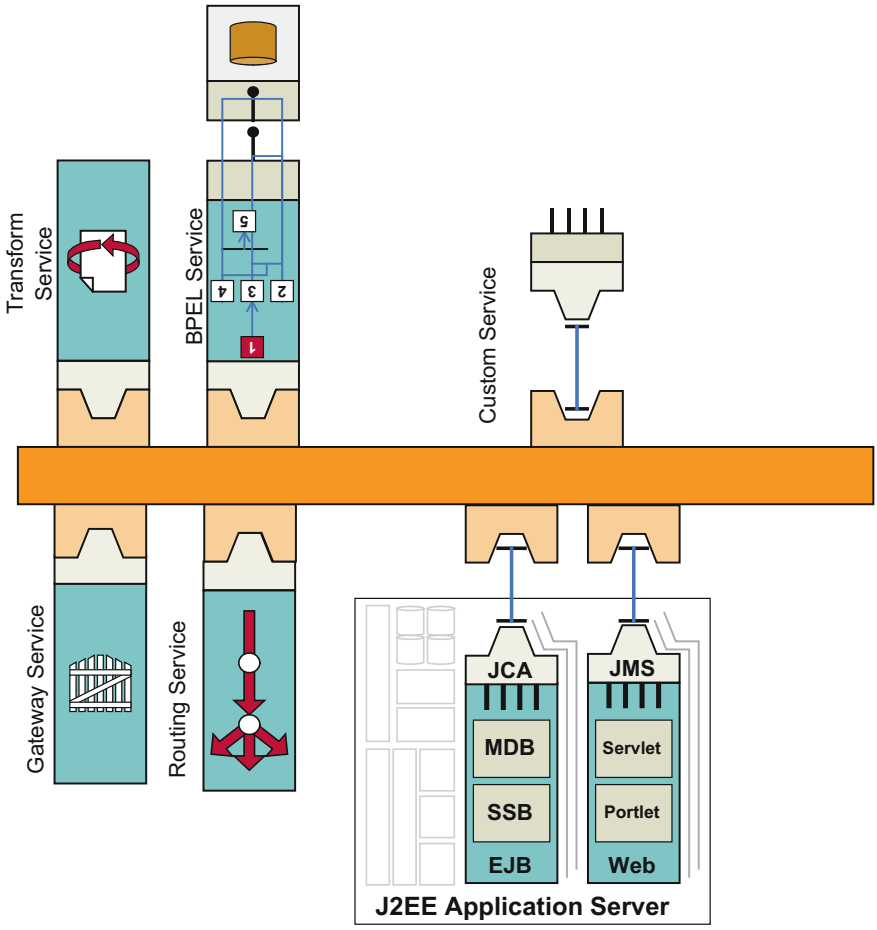


Fig. 2.19 An enterprise service bus with compounded applications

to reuse. A prime example of this is IT providing services to multiple business units in an enterprise through corporate Cloud, instead of each business unit maintaining its own IT infrastructures. A Public Cloud is a further generalization of SOA across various consumers from different enterprises and unrelated business entities, as long as their security concerns can be met.

2.11 Enterprise Implementation on Private Clouds

Increased adoption of Client–Server architecture and Internet computing infrastructure has resulted in a large number of servers scattered across different divisions of an enterprise. However, IT experts soon learnt that each organization works in a

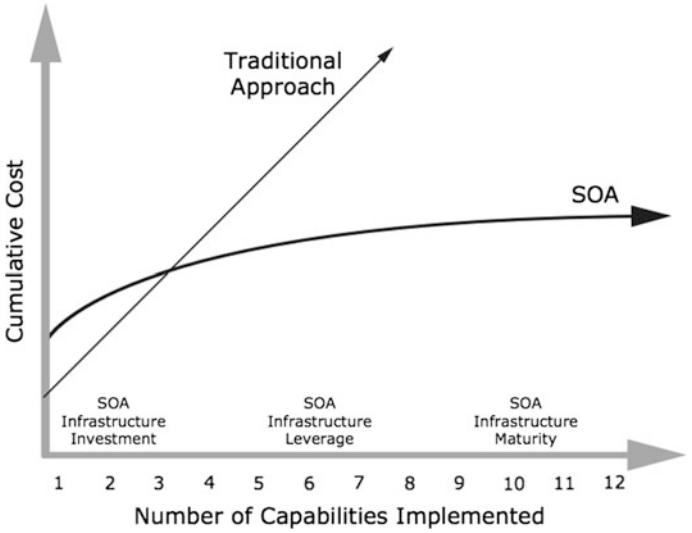


Fig. 2.20 SOA versus traditional delivery cost structure

silo and sharing of servers across groups was minimal. Furthermore, due to time zone differences, it is possible to share the hardware and software licenses across different divisions. It is further enabled by the use of virtual machines. Hence, the idea of creating an enterprise-wide data center, where jobs and tasks from different divisions can be scheduled to run, seemed very attractive for cost savings and run-time efficiencies as powerful servers can be purchased with pooled resources.

The need for a Private Cloud, which is a virtual data center backed by one or more physical data centers, was driven by the following factors:

- Need for rapid scaling of applications when additional capacity is required.
- Inability to predict capacity and performance trends.
- Organizations grow but applications do not have the scalability, reliability, and performance.
- Difficult to maintain client/server applications, especially on thousands of PCs.
- Unused capacity and associated costs.

The factors mentioned above motivated growth of grid computing [4] during the early 2000s, which has been largely subsumed by Cloud Computing in subsequent years. With grid computing, several inexpensive servers are combined to function as a large server. Thus, IT departments no longer had to acquire large and expensive servers to run existing or future workloads. Moreover, new capacity could be added to existing infrastructure by simply adding new servers and systems. These are generally stacked in racks of servers, and at the top of rack is a networking switch to monitor the loading of each server and load it appropriately. Each department contributes to the purchase and maintenance of servers, and run-time costs of these servers, based on the extent of its usage. This is a basic tenant of Cloud Computing,

except that the cost has to be paid upfront for IT department to buy the servers. Any financial adjustments are made during the lifetime of that server in terms of credits and debits. Furthermore, enterprise customers do not know exactly which servers their jobs are running on and have no direct control of these machines. However, they trust the IT managers who are the employees of the same enterprise to run their jobs in Private Clouds [7].

2.12 Enterprise Implementation on Hybrid Clouds

A general contrast between Amazon's AWS and Microsoft's Azure has been latter's ability to tap into enterprise segment. IT managers tend to be conservative and want to maintain a semblance of control on the servers used by their corporate customers. Thus, they are reluctant to adopt Public Clouds and migrate their corporate data into unknown server locations. This gave rise to computing environments, which are a mix of on-premises, Private Clouds and third-party, Public Cloud servers with orchestration between the two sets of platforms.

A hybrid Cloud [10] enables enterprise IT managers to combine one or more Public Cloud providers such as AWS and Google Cloud Platform (GCP), with a Private Cloud platform. Using encrypted connection technology, data and application can be ported between Private and Public Cloud components. One of the usage models is Cloud Bursting, i.e., when number of enterprise jobs exceed the Private Cloud capacity, some of these are seamlessly migrated to an external Public Cloud server. Later on, if and when the internal demand for computing decreases, external jobs can be brought back into run on the internal servers. This in summary is how Cloud Bursting works.

Microsoft has further extended this model with an Azure Stack [17] that enables IT managers to deliver Azure services from an enterprise data center. Azure Stack is an integrated system, ranging in size from 4 to 12 server nodes deployed on the premise and jointly supported by a hardware partner and Microsoft. Azure Stack uses Public Cloud for any bursting needs as well as backup of production workloads to assure 24×7 business continuity. Additional use cases are still evolving.

2.13 Web Threat Models

Client-side browsers may contain vulnerabilities that may enable remote code execution by Web sites. A Google study in 2007, "The Ghost in Browser" [11], found Trojan software on 300,000 Web pages and adware on 18,000 Web pages. The URLs (Uniform Resource Locator) or Web addresses for these sites were examined, to find issues of following types:

1. **HTML**: Attackers can identify Web applications with vulnerabilities and insert small pieces of HTML code in Web pages. This HTML code is then used as a vehicle to launch large collections of exploits against any user who visits the infected page [16].
2. **Rendering content**: Each browser has an execution model, with windows or frames to load content and render it. Rendering refers to processing HTML and scripts to display Web pages, which may involve images and sub-frames, etc. By having a transparent frame overlaid on a legitimate and expected frame, attacker is able to get the click data as well as information such as login passwords or credit card numbers, etc.
3. **Remote Scripting**: The goal here is to exchange data between a client-side App running in a browser and a server-side App without reloading pages. Examples include a Java applet, ActiveX controls, or Flash. A page can maintain bidirectional communication with browser, until a user closes the page or quits the Web App. By injecting scripting commands on behalf of an active user, the attacker can cause the server side to behave in an unexpected manner, e.g., deleting all files in the user's account.
4. **Cookies**: These are used to store state on a user's machine. Think of these as simple and small text files that are used by a Web browser to maintain persistent context across Web sessions. Cookies can contain information such as user authentication, personalization, and usage tracking. A browser may store typically 20 cookies/site, with 3 Kb of information per cookie. Cookies provide confidentiality against network attacker; e.g., a browser will only provide cookies over HTTPs, but not integrity protection; e.g., a network attacker can rewrite secure cookies and alter their content.
5. **Frames and frame busting**: A Web page may contain frames from different sources, which can be rigid or a floating inline iFrame, as shown in Fig. 2.21. Their purpose is to designate screen areas to display content from another source. The browser is supposed to provide isolation based on frames. A browser's security mechanism is based on such isolation, as each frame of a page has its own origin or URL. However, if frame policies are not set properly, then one frame can interfere with the data or cookies of another frame within its own hierarchy. An example of this is Clickjacking [18], which uses features of

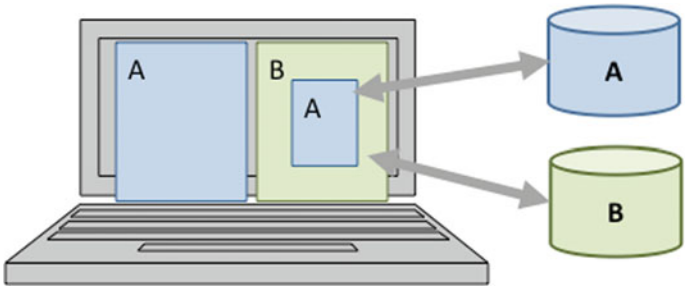


Fig. 2.21 Using browser frame isolation as a Web security mechanism

HTML and Javascript to trick the victim to perform undesirable actions, such as clicking a button that appears to perform a different operation. This is a "client-side" security issue that affects a variety of browsers and platforms, using page overlays. Once the victim is surfing on the fictitious Web page, he thinks that he is interacting with the visible user interface, but effectively he is performing actions on a hidden page. The attacker can deceive users into performing actions they never intended to perform through an "ad hoc" positioning of the elements in the Web page, such as typing their passwords or credit card numbers in a realistic appearing page overlaid on a hacker's page. This can be prevented by Frame Busting, which consists of a script in each page that should not be framed. The aim of this technique is to prevent a site from functioning when it is loaded inside a frame.

2.14 Open Web Application Security Project

In view of the threats presented in the previous sections, a not-for-profit foundation called Open Web Application Security Project (OWASP) [12] was established in April 2004 to conceive, develop, and maintain applications that can be trusted. All of the OWASP tools, documents, forums, and chapters are free and open to anyone interested in improving application security. An example of end-to-end threat model and mitigations espoused by OWASP is shown in Fig. 2.22.

Web application attackers can potentially use many different paths through applications to do harm to business or organization. Each of these paths represents a risk that may, or may not, be serious enough to warrant attention. Sometimes, these paths are trivial to find and exploit, and sometimes, they are extremely difficult. Similarly, the harm that is caused may range from nothing, all the way to shutting down a business. To determine the risk to an organization, one should evaluate the likelihood associated with each threat agent, attack vector, and security weakness and combine it with an estimate of the technical and business impact to the

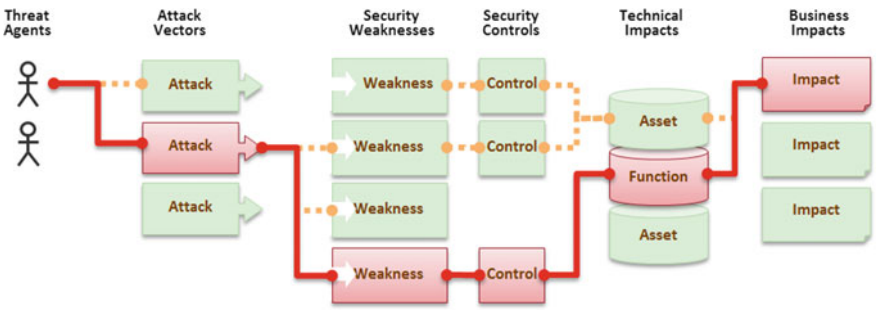


Fig. 2.22 An illustration of application security risks

organization. Together, these factors determine the overall risk. OWASP has identified [13] and listed top 10 risks in 2013, as enumerated below:

1. **Injection**: Injection flaws, such as SQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.
2. **Broken Authentication and Session Management**: Application functions related to authentication and session management are often not implemented correctly. This allows attackers to compromise passwords, keys, session tokens, or to exploit other implementation flaws, to assume other users' identities. Another way this can happen is through a playback attack, e.g., by storing and replaying a user's login session.
3. **Cross-Site Scripting (XSS)**: XSS flaws occur whenever an application takes untrusted data and sends it to a Web browser without proper validation. XSS allows attackers to execute scripts in the victim's browser, which can hijack user sessions, deface Web sites, or redirect the user to malicious sites.
4. **Insecure Direct Object References**: A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, or database key. Without an access control check or other protection, attackers can manipulate these references to access unauthorized data.
5. **Security Misconfiguration**: Good security requires having a secure configuration to be defined and deployed for the application, frameworks, application server, Web server, database server, and platform. Secure settings should be defined, implemented, and maintained, as defaults are often insecure. Additionally, software versions should be kept up to date with latest patches.
6. **Sensitive Data Exposure**: Many Web applications do not properly protect sensitive data, such as credit cards, tax IDs, and authentication credentials. Attackers may steal or modify any weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data deserves extra protection such as encryption at rest or in transit, as well as special precautions when exchanged with the browser.
7. **Missing Function Level Access Control**: Most Web applications verify function level access rights before making that functionality visible in the UI. However, applications need to perform the same access control checks on the server when each function is accessed. If incoming requests are not verified, attackers will be able to forge requests in order to access functionality without proper authorization.
8. **Cross-Site Request Forgery (CSRF)**: A CSRF attack forces a logged-on victim's browser to send a forged HTTP request, including the victim's session cookie and any other automatically included authentication information, to a vulnerable Web application. This allows the attacker to force the victim's browser to generate requests for vulnerable application, disguised as legitimate requests from the victim.

9. **Using Components with Known Vulnerabilities**: Components, such as libraries, frameworks, and other software modules, almost always run with full access privileges. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications using components with known vulnerabilities may undermine application defenses and enable a range of possible attacks and impacts. A recent example of such an attack is WannaCry Ransomware [19], which exploited vulnerability in Windows Server Message Block (SMB) v1 protocol. WannaCry used a combination of the RSA and AES algorithms to encrypt files and then demands money in Bitcoins to decrypt an infected user's files. Microsoft addressed this vulnerability with Security Bulletin [14].
10. **Un-validated Redirects and Forwards**: Web applications frequently redirect and forward users to other pages and Web sites and use untrusted data to determine the destination pages. Without proper validation, attackers can redirect victims to phishing or malware sites, or use forwards to access unauthorized pages.

2.15 Summary

Cloud Computing emerged only a decade ago. It is based on basic technologies that have been around and under development for more than half a century. These technologies were hardened in other environments, such as defense and personal computing. Note that as we advance into a Web application domain, it also expands the hacking attack surface. Threat agents can have multiple entry points on the client-side browsers, network in between, and server-side hardware or software. Each risk needs to be evaluated, and mitigation strategies developed in advance to prevent harm to the users of Web applications.

2.16 Points to Ponder

1. **What led to the desire of people with PCs to connect with one another?**
2. **What led to the growth of thin clients? How thick should thick clients be and how thin should thin clients be? Which use cases suit each category (e.g., an information panel at the airport versus an enterprise handheld computer)? Or, given a usage class which client they should use? Besides usage, software update frequency and security are also a consideration.**
3. **How has SOA helped with the evolution of Cloud Computing?**
4. **Note that Cloud Computing is a natural evolutionary step as communication links grew in capacity and reliability.**

5. **Even though cookies pose security risks, why do browsers allow them and what's the downside for a user to not accept cookies?**
6. **What's the minimal precaution a public Web site's user should take?**

References

1. Bhatt PCP (2014) An introduction to operating systems concepts and practice (GNU/Linux). PHI Learning Pvt Ltd, 1 Jan 2014
2. http://www.computerhistory.org/timeline/1980/
3. https://en.wikipedia.org/wiki/Modem
4. http://www.gridcomputing.com
5. http://www.informit.com/articles/article.aspx?p=27295
6. Riley M, Elgin B, Lawrence D, Matlack C (2014) Missed alarms and 40 million stolen credit card numbers: how target blew it. Bloomberg Businessweek. Retrieved from: http://www.businessweek.com/articles/2014-03-13/target-missed-alarms-in-epic-hack-of-credit-card-data
7. https://www.techopedia.com/2/29245/trends/cloud-computing/building-a-private-cloud
8. http://www.soablueprint.com/yahoo_site_admin/assets/docs/BEA_SOA_Domains_WP.290214359.pdf
9. https://www.javaworld.com/article/2077653/soa/build-an-soa-application-from-exisiting-services.html
10. http://www.zdnet.com/article/hybrid-cloud-what-it-is-why-it-matters/
11. https://www.usenix.org/legacy/event/hotbots07/tech/full_papers/provos/provos.pdf
12. https://www.owasp.org/index.php/About_The_Open_Web_Application_Security_Project#The_OWASP_Foundation
13. https://www.owasp.org/index.php/Top_10_2013-Top_10
14. https://technet.microsoft.com/en-us/library/security/ms17-010.aspx
15. https://www.service-architecture.com/articles/web-services/web_services_explained.html
16. https://iconewsblog.wordpress.com/2015/09/16/does-your-website-have-a-leak/
17. https://docs.microsoft.com/en-us/azure/azure-stack/
18. https://www.owasp.org/index.php/Testing_for_Clickjacking_(OTG-CLIENT-009)
19. https://www.secureworks.com/research/wcry-ransomware-analysis
20. http://www.datacenterknowledge.com/archives/2015/07/08/technical-issue-halts-trading-on-nyse
21. https://www.usatoday.com/story/news/2017/01/30/delta-outage-airline-technology-problems/97250834/

Chapter 3
Cloud Computing Pyramid

3.1 Roots of Cloud Computing

"Cloud Computing" is a recent term starting from about a decade ago. However, its roots extend at least half a century back when users sat in front of blinking terminals far away from mainframe computers connected via cables. Telecommunication engineers about a century ago used Cloud concepts. Historically, telecommunications companies only offered single dedicated point-to-point data connections [1]. In the 1990s, they started offering virtualized private network (VPN) connections, with the same Quality of Service (QoS) as their dedicated services but at a reduced cost. Instead of building out physical infrastructure to allow for more users to have their own connections, telecommunications companies were now able to provide users with shared access to the same physical infrastructure. Later on, notions of utility computing, server farms and corporate data-centers formed the foundation of current generation of large Cloud Data-centers. The same theme underlies the evolution of Cloud Computing, which started with mainframes and now has evolved to 24 × 7 services (24 h service available 7 days a week, i.e., no downtime). The following list briefly explains the evolution of Cloud Computing [2]:

- **Grid computing**: Solving large problems using parallelized solutions, in a server farm
- **Utility computing**: Computing resources offered as a metered service
- **SaaS**: Network-based subscriptions to applications
- **Cloud Computing**: "Anytime, anywhere" access to IT resources delivered dynamically as a service.

Server farms did not have Applications Programming Interface (APIs). Each user was made to think that he or she had full access and control of a server, but in reality time-sharing and virtual machine isolation kept each user's processes independent of others. Let us consider an example: assume a Pharmaceutical company called *P*. Also, assume there is a Hollywood movie production studio

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6_3

called *H*. *P* needs 1000 machines to run a drug trial on a continuous 24 × 7 basis for a whole month. *H* also needs 1000 machines on a weekend to render the scenes shot during the week. If only 1000 physical servers exist in the data-center, on an exclusive basis these can be loaned to one or the other user. In practice, if each user is running their respective tasks in a VM, then additional 1000 VMs can be spawned on the weekend, which may slow down *P's* jobs, but not impact their confidentiality or integrity. Each user thinks that he or she has a full access to 1000 server "Platforms." In this context, the term "Platform" is used to represent a collection of hardware and software components such as CPU, memory, and storage in an entity such as a server. Several servers are stacked to form a rack and racks of servers constitute a data-center that can be accessed via the Internet.

Each Platform hides the complexity and details of the underlying infrastructure from users and applications by providing a Web-based graphical interface or APIs. These server Platforms provide "always on-demand services" and are infrequently rebooted for planned maintenance such as HW upgrades or low-level BIOS patches. Meanwhile, load balancers and routers direct user traffic to other such servers, to give users an impression of "always-on" Cloud Services. The hardware- and software-based Cloud services are made available to general public, enterprises, and Small and Medium Businesses (SMBs) by the Cloud Service Providers (CSPs).

The National Institute of Standards and Technology (NIST) in USA has defined Cloud Computing [3] as: "Cloud Computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." This Cloud model is composed of five essential characteristics, three service models, and four deployment models. The three service models mentioned in the NIST definition are described as follows:

1. **Software as a Service (SaaS):** It allows the consumer to use the provider's applications running on a Cloud infrastructure. A user need not worry about the operating system, CPU types, memory, or any storage required for these applications. Also, the software developers are able to roll out new version and patches without worry about CDs or other distribution models, charging users on a subscription basis, much like cable TV services that charge for channel bundles based on number of television sets at home (Fig. 3.1).
2. **Platform as a Service (PaaS)**: It allows the consumer to deploy onto the Cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider. This is one level lower than SaaS, but user does not need to worry about the underlying hardware, such as the memory, storage, or CPU capabilities.
3. **Infrastructure as a Service (IaaS)**: It allows the consumer to provision CPU processing, storage, networks, and other computing resources at the lowest abstraction levels, with an ability to deploy and run software, which can include operating systems and applications.

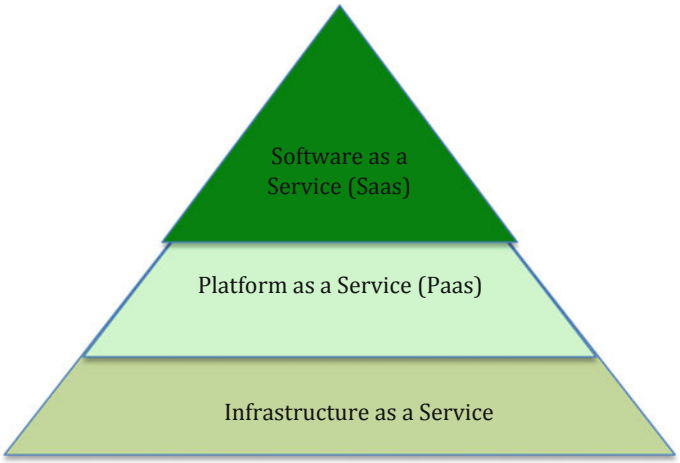


Fig. 3.1 Cloud Computing services pyramid

Business models differentiate Cloud providers in the following five deployment models:

1. **Public Clouds**: Cloud infrastructure is made available to the general public. An example is Amazon Web Services (AWS) offering Elastic Compute Cloud (EC2) to anyone with a credit card. Typically, server rental prices are charged on a per-hour rental basis.
2. **Private Clouds**: Cloud infrastructure is operated solely for an organization. An example is an Enterprise that offers Cloud-based services behind corporate firewalls to its employees located at different sites. Internal departments contribute or pay to build such a Cloud, typically maintained by the Corporate IT services.
3. **Hybrid Clouds**: Cloud infrastructure is composed of two or more Clouds that inter-operate or federate through technology. An example is a company with an internal data center, which further uses a Public Cloud for certain non-mission critical tasks, or during overage demand periods. Central IT service determines the policy on when to use internal versus external resources, generally based on the usage patterns, or confidentially of data being shared and task urgency.
4. **Community Clouds**: Cloud infrastructure is shared by several organizations to supporting a specific community. An example is a large housing complex with Cloud-based community services exclusively for its residents. All residents, similar to backup power generation or water facilities in a closed community, use and share the cost of such services.
5. **Virtual Private Clouds (VPC)**: This simulates a Private Cloud experience but located within a Public Cloud infrastructure cluster, such as a Health Maintenance Organization (HMO) offering private services to its members, but hosting their data and applications on segregated servers within AWS. Due to

the nature of data and tasks being handled, a VPC cluster may be physically isolated in terms of storage and networking, etc., which means that VPC users are charged a fixed cost and then a variable amount on the basis on their usage patterns.

From the above, hybrid Clouds are the fastest growing segment as it enables Enterprise ITs, and even SMB owners to keep onsite computing capabilities, to provide business continuity in case of external Internet disruptions, and allows access to outside Cloud resource for dealing with sudden compute demand surges. One motivation is to leverage existing capital investments in an already existing local computing infrastructure and alleviate wait times for additional resources on need basis.

3.2 Essential Characteristics of Cloud Computing

According to NIST, any Cloud must have the following five characteristics:

1. **Rapid Elasticity**: Elasticity is defined as the ability to scale resources both up and down as needed. To the consumers, the Cloud appears to be infinite, and they can purchase as much or as little computing power as they need. This is one of the essential characteristics of Cloud Computing in the NIST definition (Fig. 3.2).
2. **Measured Service**: In a measured service, aspects of the Cloud service are controlled and monitored by the Cloud provider. This is crucial for billing, access control, resource optimization, capacity planning, and other tasks.
3. **On-Demand Self-Service**: The on-demand and self-service aspects of Cloud Computing mean that a consumer can use Cloud services as needed without any human interaction with the Cloud provider.

Fig. 3.2 Five essential characteristics of Cloud Computing

4. **Ubiquitous Network Access**: Ubiquitous network access means that the Cloud provider's capabilities are available over the network and can be accessed through standard mechanisms by both thick and thin clients.
5. **Resource Pooling**: Resource pooling allows a Cloud provider to serve its consumers via a multi-tenant model. Physical and virtual resources are assigned and re-assigned according to consumers' demand. There is a sense of location independence in that the customers generally have no control or knowledge over the exact location of the provided resources but may be able to specify a geographical location (e.g., country, state, or data center).

3.3 Cloud Players and Their Concerns

Figure 3.3 shows a consumer–producer chain for Cloud Services, starting with the users requesting services from their accounts located in a Cloud portal, e.g., an e-mail service hosted in the Cloud. A Cloud user can start the request by opening a Web browser page and logging into the user account for which all the incoming mails are stored somewhere in the Cloud. Users may not care about the actual location of Cloud servers as long as they can get a reasonable response time and some assurance of secured, continuous service. The Cloud infrastructure may be located in a remote data-center or spread across many, depending on the location and mobility of users. The IT manager in charge of Cloud operations manages the (QoS) by monitoring several metrics, e.g., average wait and response times for users, as well as the application performance etc. Oftentimes, the Cloud service is hosted in a third-party data-center, and here, the hardware usage monitoring and maintenance is done by the IT service providers, whereas the application performance monitoring is done by the (CSP). In any event, Cloud users do not care about who is doing what as long as they get their service.

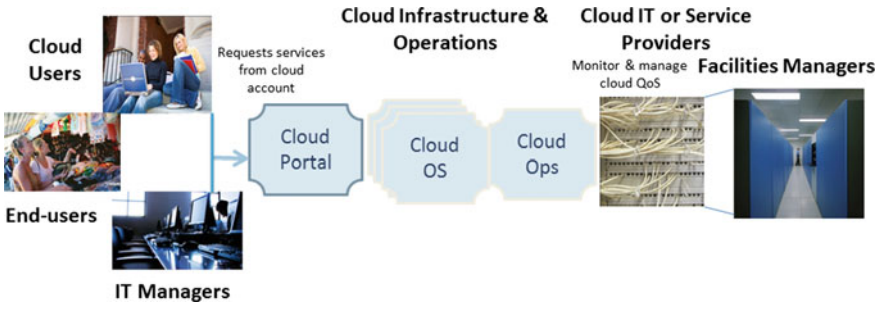


Fig. 3.3 Consumer–producer chain in the Cloud [11]

In a typical Public Cloud, the following actors are involved:

1. **Facility Managers**: They are responsible for the Cloud infrastructure and its operations, i.e., efficient operations of a Data-Center (DC), including the servers, storage, power supplies, and air-conditioning equipment. Their concern is to run DC with maximum uptime, and at the lowest possible operational cost.
2. **Cloud IT Managers or Service Providers**: They are responsible for scheduling tasks in one or more DCs. Their goal is to maximize utilization of equipment. This refers to maximal loading while leaving sufficient headroom for any usage spikes, monitoring infrastructure performance, planning future capacity growth, and compliance to ensure secure and efficient DC operations.
3. **Cloud Users**: They are the customers of Cloud services and may be different than the applications' end users, e.g., someone who is renting machines in a Cloud to host an application or provide a service, such that end users may not know where their jobs are running. A Cloud User's concern is to provide a Quality of Service that the end user expects at the lowest possible cost. An example is Netflix using Amazon's AWS to host movies.
4. **End users**: The people or businesses at the other end of a Cloud. They may be using mobile or other devices to consume services that are hosted in the Cloud. The end users' concern is the protection of their data and QoS.
5. **IT managers**: They are in-charge of computing infrastructure operation. If the Cloud user is a business, it typically has an IT manager for the internal or enterprise data-center, but also using a Cloud for backups, dealing with occasional workload spikes, or certain type of non-mission critical business services.

Figure 3.4 outlines pain points in Cloud, starting with Cloud Users who must specify the type of services they will need in advance. If their future requests exceed the forecast, then level of service will depend on the elasticity of service providers' capacity, e.g., since the Cloud hardware is shared between many tenants, it may be fully utilized at a given time. In this case, level of service will deteriorate for all users on a given server; e.g., if one of the users starts to do excessive I/O operations, it will slow down I/O requests from other users. This is referred to as a noisy neighbor virtual machine (VM) problem such as caused by placing a streaming media task next to the database-accessing job on the same server. A Cloud User

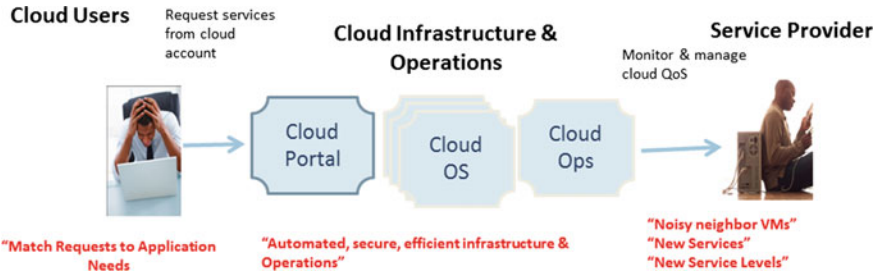


Fig. 3.4 Pain points in the Cloud

may care only for his or her own QoS, while an IT manager needs to worry about satisfying all users. The facilities manager needs to ensure that hardware is being fully utilized and the data-center is receiving adequate power and cooling, etc. There is no silver bullet to solve all Cloud actors' problems. An orchestration of actions is needed in response to dynamic monitoring of usage situation at various levels. This will be further explained in later chapters.

3.4 Considerations for Cloud Data Centers

A major consideration in selecting a Cloud Service Provider is the service-level agreements (SLA). Several researchers have targeted the problem of meeting SLA requirements and optimizing resource allocation for Cloud users, because SLAs define measurable considerations, such as listed below:

1. Response time
2. Output bandwidth
3. Number of active servers to monitor for SLA violations
4. Changes in the environment and
5. Responding appropriately to guarantee Quality of Service.

Appleby et al. [4] developed the Oceano framework to match Cloud resource allocation to users' requirements. Emeakaroha et al. [5] presented LoM2HiS, a framework that uses resource usage metrics such as network bandwidth availability, data transfer, and down/uptime to prevent SLA violations. Others perform statistical analysis and employ heuristics to improve resource allocation in autonomous compute environments such as a Cloud infrastructure. Ardagna et al. [5] employed heuristics with local-search algorithm to better match physical servers with application sets that are currently using them. Bennani and Menasce [6] used analytic performance models to better match application environments with physical servers.

These concepts of relating a workload to hardware utilization can be applied to the Cloud by monitoring the virtual machines (VMs) and utilizing a load balancing mechanism. An example of a simplistic approach is the switch hierarchy within a standard data center. A standard data center design involves a switch on top of each rack connecting to higher level switches forming a tree, with a load balancer at the rack level between the two lowest level aggregate switches. The load balancer simply re-allocates jobs when one VM, computer, or server gets too full. This is analogous to pour drinks to guests but only switching glasses when the liquid overflows out of the cup onto the table. According to the work done by Khan et al. [7], monitoring a single VM over time appears only as noise but observing a cluster of VMs over time reveals a pattern, showing that if a moderately accurate prediction about the workload could be made, the load balancing would be much more effective.

Box 3.1: What is Virtualization, and why is it needed?
Data center servers are powerful machines. No single application can possibly utilize their computational power on a 24 × 7 basis over long periods of time. Hence, there is a good business case to share a data center's resources among many different users. The enabling technology is called Virtualization, which creates a layer of abstraction between the applications and the underlying hardware. It makes each user feels that he or she has full access and control of the machine, whereas in reality, there is time-sharing between different users. With multi-cores, it is possible to allocate each user to a dedicated core, but still some I/O resources will need to be shared between the users. This can cause occasional performance bottlenecks.

3.4.1 Migration

Another challenge is migrating traditional workloads that require computer clusters such as High-Performance Computing (HPC) workloads from captive data-centers to a Cloud. The best-known example of commercial support for HPC in the Cloud is Amazon's Cluster Compute Instances (CCI). Amazon's CCI tries to address the upfront costs of running a cluster by allowing the customer to rent a scalable number of instances by the hour. For one-time HPC jobs or low utilization HPC needs, this solution appears cost-effective. However, for highly utilized clusters, the cost of running HPC jobs on Amazon CCI can be significantly higher. Carlyle et al. [8] show that current pricing on Amazon CCI is prohibitive when compared to the total cost incurred by their university maintained cluster. The relatively high utilization ratio on their university cluster resources amortizes the upfront and maintenance costs of owning the cluster.

3.4.2 Performance

Besides cost, the performance of HPC in a Cloud is of significant concern. Zhai et al. [9] evaluate Amazon CCI for tightly coupled Message Passing Interface (MPI)-based parallel jobs. They compare jobs run on CCI and an InfiniBand backed local cluster and found that performance in the Cloud is significantly reduced due to the lower bandwidth provided by the 10 gigabit network on CCI as compared to the InfiniBand network on their dedicated cluster. They also found significant performance variability with storage provided by Amazon's virtual storage infrastructure, often with periods of very low disk performance lasting for hours. Evangelinos and Hill [10] also reached the same finding and illustrated as much as two orders of

magnitude higher latency in inter-node communication as compared to dedicated high-performance clusters. Most recently, IBM has started to use Watson as their HPC solution for advanced analytics [12] and several real-life problems such as medicine, etc.

3.5 Points to Ponder

1. **SMB users lack financial muscle to negotiate individual SLAs with (CSP). So what are their options?**
2. **Some potential Cloud users are on the fence due to Vendor lock-in concerns. How can these be addressed?**
3. **What are the new business opportunities that enable movement of users and data between different Cloud providers?**
4. **What are some key difference between the concerns of a Private versus Public Cloud's IT managers?**
5. **Why is virtualization important for Public Clouds? Would it also help in Private Clouds?**
6. **Is workload migration a viable solution for a large Public Cloud Service Provider?**

References

1. Donovon and Magi. https://www.amazon.com/Operating-Systems-Computer-Science-Madnick/dp/0070394555
2. https://www.ibm.com/blogs/Cloud-computing/2014/03/a-brief-history-of-Cloud-computing-3/
3. Mell P, Grance T (2011) The NIST definition of cloud computing (draft). NIST Spec Publ 800:145
4. Appleby K, Fakhouri S, Fong L, Goldszmidt G, Kalantar M, Krishnakumar S, Pazel DP, Pershing J, Rochwerger B (2001) Oceano-SLA based management of a computing utility. In: Integrated network management proceedings, 2001 IEEE/IFIP international symposium on, pp 855–868
5. Emeakaroha VC, Brandic I, Maurer M, Dustdar S (2010) Low-level metrics to high-level SLAs-LoM2HiS framework: bridging the gap between monitored metrics and SLA parameters in cloud environments. In: High performance computing and simulation (HPCS), 2010 international conference on, 2010, pp 48–54
6. Bennani MN, Menasce DA (2005) Resource allocation for autonomic data centers using analytic performance models. In: Autonomic computing, 2005. ICAC 2005 proceedings second international conference on 2005, pp. 229–240
7. Khan A, Yan X, Tao S, Anerousis N (2012) Workload characterization and prediction in the cloud: a multiple time series approach. In: Network operations and management symposium (NOMS), 2012 IEEE, 2012, pp 1287–1294
8. Carlyle AG, Harrell SL, Smith PM (2010) Cost-effective HPC: the community or the Cloud? In: Cloud computing technology and science (CloudCom), 2010 IEEE second international conference on, 2010, pp 169–176

9. Zhai Y, Liu M, Zhai J, Ma X, Chen W (2011) Cloud versus in-house cluster: evaluating amazon cluster compute instances for running MPI applications. In: State of the practice reports, p. 11
10. Evangelinos C, Hill C (2008) Cloud computing for parallel Scientific HPC applications: feasibility of running coupled atmosphere-ocean climate models on amazon's EC2. Ratio 2:2–34
11. Mulia WD, Sehgal N, Sohoni S, Acken JM, Stanberry CL, Fritz DJ (2013) Cloud workload characterization. IETE Tech Rev 30(5):382–397. (Institution of Electronics and Telecommunication Engineers, India)
12. https://www.ibm.com/analytics/watson-analytics/us-en/

Chapter 4
Features of Private and Public Clouds

4.1 Customer Expectations of Cloud Computing

While economics is the main driver of Public Cloud Computing, its customers want the same performance and security aspects that they enjoy on a private server. This is a classic case of wanting to have your cake and eat it too. Specifically, Public Cloud users want full observability and controllability for their workloads. This refers to the applications and data they deploy on a remote server. They also expect a cloud server to be secure as if it was on their own premises behind a firewall. These requirements are well captured in NIST's 5 essential characteristics that we listed in the previous chapter. However, there are some ambiguities; for example, "On-demand self-service" requires that a consumer can unilaterally provision computing capabilities, without requiring human interaction with a Cloud Service Provider. This implies automation on the Cloud Service Provider's site, but does not specify anything on the user side. In reality, for a user to provision hundreds of jobs at a moment's notice or to monitor them requires some automation. Furthermore, if a particular application is not behaving well, then the user will need some diagnostics to identify the root cause of problems and be able to migrate the job to another Cloud server. This implies availability of suitable monitoring and alerting tools. Automation of actions is such that a user and Cloud provider's environments work in unison. This will help to realize the full potential of Cloud Computing. The reality lies somewhere in between, as shown in Fig. 4.1.

Most enterprises already own some servers and storage facilities in their private data centers. For customers with confidential or performance-sensitive workloads, Public Cloud providers offer a Virtual Private Cloud, which can be thought of as a hosted service, or a group of dedicated servers cordoned off for such a customer. These offer dedicated facility but at a higher cost since the Cloud Service Provider cannot share this infrastructure with other customers. This is preferable for some Cloud users who do not wish to maintain their own IT services but want the assurance of privacy.

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6_4

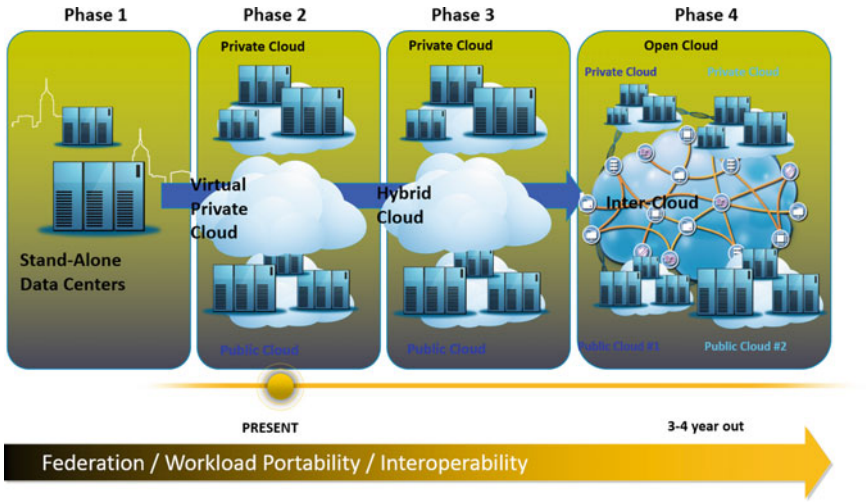


Fig. 4.1 A phased approach to adoption of Cloud Computing

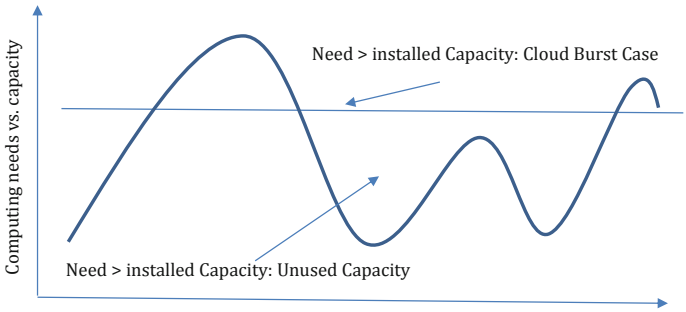


Fig. 4.2 Variation computing needs of an organization over time

Next phase is a hybrid Cloud, where users experience large variations in the Cloud consumption profiles, some of which exceeds their internally installed server base. Then they have 2 choices, either to buy more servers or let some computing demand remain unsatisfied. The former requires more capital investment and operational costs, as these servers may remain idle during off-peak times. The second choice has an implication of lost business opportunities as user tasks will need to wait in a queue, or worst yet customers will go somewhere else to meet their need. This is particularly true for online retailers who cannot respond to their shoppers in a timely manner. This dilemma is depicted in Fig. 4.2, where one way to meet the unsatisfied demand is via migration of computing tasks to a Public Cloud.

"Cloud Bursting" is a term used to define the jobs that were running in an internal data- centers, but are moved out to a Public Cloud at peak usage and then return back to a Private Cloud when the internal capacity is available. Since Public Clouds charge on a pay-per-use basis, this proposition is attractive. However, as we will see in later chapters, this is non-trivial because computing jobs typically require a setup, which includes an operating system environment, sometimes with a plethora of supporting tools and supporting, and often a large amount of data. Such jobs cannot be easily migrated between Private and Public Clouds at a moment's notice, which means that computing environments on both sides need to be kept in synchronization. One way to do this is by data mirroring between the data-centers, so only computing programs need to migrate while associated databases always stay in synchronization. However, this adds to the operational cost to keep the Public Cloud environment always ready to go at a moment's notice.

4.2 Interoperability of Cloud Computing

The last phase in computing evolution curve, as shown in Fig. 4.1, is interoperability, which refers to data formats, and interfaces, not just between private and Public Clouds, but among various Public Cloud players, so ideally users can decide when and where to go on need basis. This is far from the current reality, as each Cloud provider uses different set of tools, formats, and environments, so customers tend to get locked-in with a Public Cloud vendor, which hampers the growth and adoption of Clouds by mainstream computing users.

4.3 Reliability of Cloud Computing

Generally, each Cloud provider has a service-level agreement (SLA), which is a contract between a customer and the service provider. The service provider is required to execute service requests from the customer with negotiated quantity and quality ranges for a given price. Due to variable load, as shown in Fig. 4.2, dynamic provisioning of computing resources is needed to meet an SLA. Optimum resource utilization on the Cloud provider side is not an easy task. Hence, large Cloud Service Providers have fixed SLAs that are offered on a take-it-or-leave-it basis, unless the user is a large agency with high computing needs and ability to pay for these needs. In such cases, Amazon Web Service (AWS) is known to offer dedicated compute resources on an exclusive basis, which may solve the availability problem but only for large customers.

Availability is defined as the ratio of uptime/total time, where uptime is total time minus downtime. Availability ratio in percentage is measured by 9's, which is number of digits shown in Table 4.1. So 90% has one 9, while 99% has two 9s, etc.

Table 4.1 Example availability of various computing systems

9's	Availability (%)	Downtime per year	Examples
1	90.0	36 days 12 h	Personal computers
2	99.0	87 h 36 min	Entry-level business
3	99.9	8 h 45.6 min	ISPs, mainstream business
4	99.99	52 min 33.6 s	Data centers
5	99.999	5 min 15.4 s	Banking, medical
6	99.9999	31.5 s	Military defense

A good High Availability (HA) package, even with substandard hardware, can offer up to three 9's while enterprise class hardware with a stable Linux Kernel has 5 + nines.

An example is a system composed of two servers in series, such that each server has an expected availability of 99%, so the expected availability of such a system is 99% * 99% = 98.01%, because if either server fails will bring the system down. Now consider a 24 × 7 e-commerce site with lots of single points of failures, e.g., the following eight components with their respective availability:

	Component	Availability (%)
i.	Web server	85
ii.	Application	90
iii.	Database	99.9
iv.	DNS	98
v.	Firewall	85
vi.	Switch	99
vii.	Data center	99.99
viii.	ISP	95

Now the expected availability of such a site would be less than 60%, as calculated by simply multiplying the availability of each row, i.e., 85% * 90% * 99.9% * 98% * 85% * 99% * 99.99% * 95% = 59.87%. This is not acceptable, since the Web site may be down for half the time.

One way to improve this situation is by using individual components with higher reliability, but these are more expensive. Another way is to put two low availability systems in parallel, to improve the overall component-level reliability. An example is data stored on parallel disks, or two servers running in parallel, one of which can be used if the other is down. The availability of a parallel system with two components, each of which has an availability of Ac, can be computed as

$$\mathrm{As} = \mathrm{Ac} + ((1 - (\mathrm{Ac}/100)) * \mathrm{Ac})$$

Hence, two Web servers, each with only 85% availability, in parallel will have availability of 85 + ((1 − 85/100) * 85) = 97.75%. Another way to think is that Server A is down 15% of the time (with 85% reliability). Server B is also down

15% (with 85% reliability). Therefore, the chance of both Server A and Server B down at the SAME time is 0.15 × 0.15 = 0.0225 or 2.25%. So the uptime of A and B is 100 − 2.25 = 97.75%. Just this one change alone will improve the overall system-level reliability to nearly 80%, as shown below:

$$97.75\% * 90\% * 99.9\% * 98\% * 97.75\% * 99\% * 99.99\% * 95\% = 79.10\%.$$

This is better than 60% before, so let us add a second ISP to improve the overall reliability to 94.3%, which represents 500 h of downtime/year versus 3506 h/year with 60% downtime.

$$97.75\% * 99\% * 99.9999\% * 99.96\% * 97.75\% * 99.99\% * 99.99\% * 99.75\% = 94.3\%$$

However, improving it further requires more work and becomes expensive.

4.4 Performance of Cloud Computing

Cloud Computing combines the needs of several users, meeting them from one or more data-centers. This requires meeting the performance expectations of various workloads. The characterization of computer workloads has been extensively studied with many practical applications. Benchmark suites such as SPEC and PARSEC are often used to evaluate improvements to the microarchitecture such as cache designs. Jackson et al. [1] evaluated the performance of HPC applications on Amazon's EC2 to understand the trade-offs in migrating HPC workloads to Public Clouds. Their focus is on evaluating the performance of a suite of benchmarks to capture the entire spectrum of applications that comprise a workload for a typical HPC center. Through their Integrated Performance Monitoring (IPM) [2] framework, they provide an analysis of the underlying characteristics of the applications and quantitatively identify the major performance bottlenecks and resource constraints for the EC2 Cloud. Another study by Ferdman et al. [3] looks specifically at emerging scale-out workloads that require extensive amounts of computational resources. They introduce a benchmark suite, CloudSuite, to capture these workloads and characterize their performance via performance counters available on modern processors. Their high-level conclusion is that modern superscalar out-of-order processors are not well suited to the needs of scale-out workloads.

There is a 2 − 3X of performance difference observed between different Cloud vendors [4], or in the different data-centers owned by the same Cloud vendor. Even for the same Cloud data-center location, there is a 30–50% performance variation across the same class of Cloud servers. Such a wide variation poses serious problems for the customers of Public Clouds, especially when they host end user facing commercial applications or run mission-critical tasks. In the first case, end users expect real-time

response and may take their business elsewhere. In the second case, batch mode applications may take longer time impacting deadlines and violating service-level agreements.

4.5 A Sample Study

Performance tests were conducted with 24 users, 6 Public Cloud vendors, and the MegaApp application from Meghafind Inc. These experiments lasted for a full quarter during Oct–Dec 2015, with 351 samples collected and analyzed for the purpose of this book, as outlined in Table 4.2. A disclaimer is that this is a representative study done using the hardware and software available at that time. If any of the components have changed since then, the results will be different.

A user-level application (called "megaApp") collects the metrics, using a real-time clock provided by the operating system. The application creates measurement threads and schedules them to run on all CPU cores on a server. These scores are dependent on the CPU, memory speed and data access architecture, storage technology (speed, hard disk vs. SSD), operating system, threads scheduling and management, virtualization machine, virtualization guest OS, and any other applications running when the measurements are performed. The following components were measured:

1. **CPU Score** is a measure of how many seconds do certain thousands of CPU instructions take to complete. It is measured in real time by performing integer, floating point, and string operations. Lower duration scores are better than the higher duration scores.
2. **Memory Score** is a measure of how many seconds do certain thousands of memory instructions take to complete. It is measured in real time by allocating blocks of memory, reading from, and writing to the allocated memory blocks. Lower duration scores are better than the higher duration scores.
3. **Storage Score** is a measure of how many seconds do certain thousands of file creation, deletion, input and output operations take to complete. It is measured in real time by creating and deleting files, read, and writes to the created files. Lower duration scores are better than the higher duration scores.
4. **Total Score** is a simple addition of CPU, memory, and storage scores. A higher value of this score indicates that the server is running slower.

Table 4.2 Attributes of our study

Attribute	Value
Number of users	24
Number of samples	351
Number of Cloud providers	6
Number of guest operating systems	2 (Linux and Windows)
Number of VMMs or host operating systems	6

We have used Coefficient of Variation (CV) to compare performance of servers across various Clouds, and over different days of a week. CV quantifies the performance variability of the servers and incorporates the metrics described above. CV is computed as a ratio of standard deviation with average values of total score, i.e., $\text{CV} = \frac{\sigma(\textit{Total Score})}{\mu(\textit{Total Score})}$. Lower values of CV are desirable as these indicate minimal perturbations across the observed scores. For baseline comparison purposes, megaApp was run on a stand-alone machine, as shown in Fig. 4.3 over a period of 3 months. As expected, these show minimal CV if the loading conditions remain consistent.

However, megaApp results in Cloud recorded up to 14X higher Coefficient of Variation over the same period of time. This indicates large performance variability for Public Cloud users. As shown in Fig. 4.4, CPU scores had the least variations overall. This may be due to multi-cores servers being used by various Cloud providers. Since individual cores are allocated to different users, interference between users tends to be minimal. However, memory and storage read–write traffic

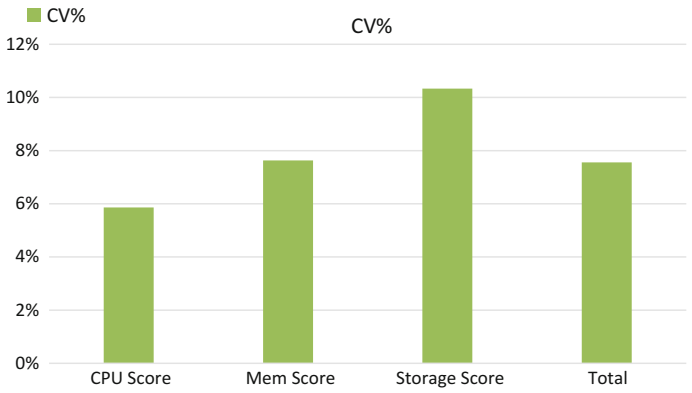


Fig. 4.3 Variations by type of scores on a stand-alone machine, lower is better

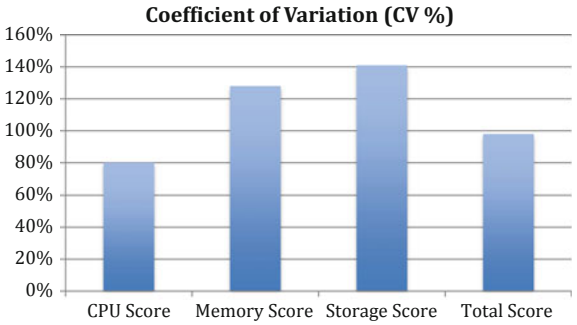


Fig. 4.4 Variations by type of scores in Public Clouds, lower is better

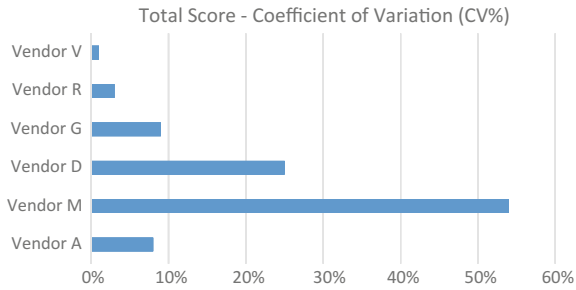


Fig. 4.5 Total score variation by Cloud vendor, lower is better

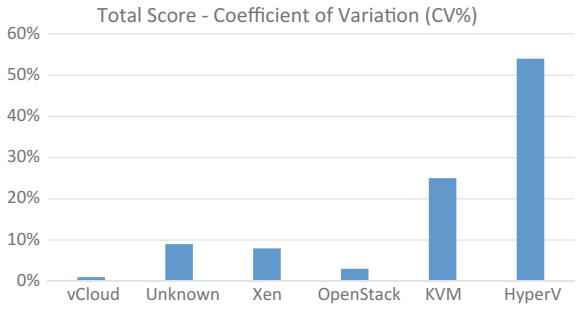


Fig. 4.6 Total score variation by VMM (i.e., Host OS), lower is better

for all users on a physical server passes through shared controllers thus giving opportunity for noisy neighbors [5].

Total score variation across different Cloud vendors is shown in Fig. 4.5, indicating that choice of a Cloud vendor is a key consideration if run-time consistency is desired.

Different vendors use varied server-sharing schemes, so it was interesting to observe wide gaps between normalized total score across different Virtual Machine Monitors (VMMs) in Fig. 4.6.

An IT manager in the Cloud data-center replaces only a fraction of her servers when a new processor is launched, to smoothen out Capital expenditure investments and minimize any perceived risks associated with any new hardware migrations. This results in a mix of various generations of servers and processors, with differing performance in a large data-center. This heterogeneity is masked by a virtualization layer, but underlying performance differences cannot be hidden and affect all user jobs running on this mix of servers. Furthermore, different users' jobs cause different loading conditions, resulting in differing performance, even on the same generation of hardware servers, also depends on Host and Guest OS combinations, as shown in Fig. 4.7.

"Day of the week" variations were surprising in that Saturday had the worst CPU and storage scores, Monday had the worst memory score. Overall, Thursday and Friday were most predictable days to get a consistent performance in the Cloud, as seen in Fig. 4.8.

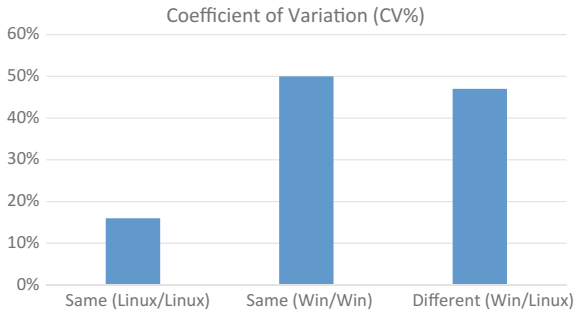


Fig. 4.7 Total score variation given a host/guest combination, lower is better

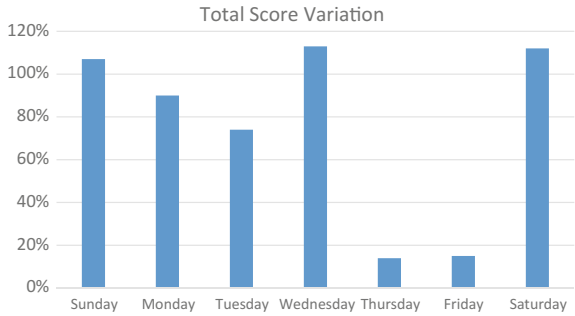


Fig. 4.8 Total score variation on a given Cloud server, lower is better

4.6 Summary

Cloud Computing performance has inherent variability. The study involved running megaApp scouts to take performance samples over time and observed large performance variations. More than 350 samples were collected over a quarter, on different days and times, to minimize the temporal dislocations. Wide variations were seen across the same type of machines in a Cloud for the same vendor, and even on the same machine over time. This study demonstrates how an end user can measure Cloud Computing performance, especially exposing the performance variability.

4.7 Points to Ponder

1. **Multi-tenancy drives load variations in a Public Cloud. Different users running different types of workloads on the same server can cause performance variations. How can you predict and minimize the undesirable effects of performance variability?**
2. **How can you predict and counter the undesirable effects of performance variability?**

3. **How is performance variability addressed in enterprise data-centers?**
4. **What is a good way to assess if a noisy neighbor problem exists?**
5. **How can a noisy neighbor problem be prevented?**
6. **How can understanding a Public Cloud's usage patterns help?**
7. **What operating system-related strategies can a user consider to optimize Cloud usage?**

References

1. Jackson KR et al (2010) Performance analysis of high performance computing applications on the amazon Web Services cloud. https://www.nersc.gov/assets/NERSC-Staff-Publications/2010/CloudCom.pdf
2. Skinner D (2005) Integrated performance monitoring: a portable profiling infrastructure for parallel application. In: Proceedings of ISC2005: international supercomputing conference, Heidelberg, Germany
3. Ferdman M et al (2012) Clearing the clouds. http://www.industry-academia.org/download/ASPLOS12_Clearing_the_Clouds.pdf
4. http://www.meghafind.com/blog/index.php/2015/12/01/secret-revealed-who-is-faster-aws-or-azure/
5. http://www.meghafind.com/blog/index.php/2015/10/17/why-my-aws-ec2-Cloud-server-suddenly-slows-down/

Chapter 5
Cloud Workload Characterization

5.1 Motivation

In previous chapters, we had elaborated several contexts that justify the use of Cloud Services for computational needs. Over time, Cloud Services have evolved to provide a variety of services. Each category of service poses a different workload challenge. The literature [1–5] is full of such characterization definitions. The definitions emanate from stakeholders that include Cloud Service providers, Cloud users, IT and Cloud facility managers, and hardware vendors. In this chapter, we will review these definitions and offer a comprehensive workload characterization with appropriate rationale.

Workload characterization employs analytical models [6, 7], and performance metric [8, 9]. Any workload characterization needs to account for resource utilization, job transitions, and such other concerns that require adhering to service-level agreements (SLAs). We will discover in this chapter that different players in the Cloud have different business and technical needs.

Various participants in Cloud Computing, including facility managers, Cloud IT or service providers, Cloud users, consumers, IT managers, and hardware vendors, do not have a common set of definitions of workloads categories. This variation in definitions leads to difficulties in matching customers' requirements with available resources. Several researchers have targeted the problem of meeting SLA requirements by optimizing resource allocation for Cloud users by using analytical models [6], high-level performance metrics [1, 2], and characterization [3–5]. Therefore, a key problem is variation in terminologies across various groups and viewpoints. This chapter specifies a set of common definitions between the participants. We will briefly describe the Cloud workload categories to enable their mapping to compute resource requirements. This will help to describe the metrics that can be used to distinguish job transitions between categories. These metrics can also be used to detect resource contention between the workloads and, in conjunction with the categories, minimize contention to improve resource allocation. Ultimately, the purpose

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6_5

of this chapter is that SLAs, capital purchase decisions, and computer architecture design decisions can be made based upon the workload categories.

Cloud Computing is utilized by a wide variety of applications. These applications have different requirements and characteristics. As inventor Charles F. Kettering said, "A problem well stated is a problem half solved." In order to better state the Cloud Computing workload characterization problem, let us decompose it in 4 steps, as follows:

1. The first step is to define a comprehensive list of the computer workload categories.
2. The second step is to identify resources required by Cloud Computing.
3. The third step is to map each Cloud Computing workload category to the required computer resources.
4. The fourth and last step is to correlate low-level hardware metrics with the Cloud Computing categories.

Above steps define a problem space, whose solutions include identifying real-time measurements that indicate a change in categories, relating these to SLAs, capital purchase decisions, architecture design decisions, software implementations, and pricing mechanisms.

In order to minimize the capital expenditure of upgrading all servers in a data center, IT managers tend to refresh only 20–25% of platforms each year. This results in a heterogeneous mix of 4–5 generations of platforms in a large data center, assuming that systems' Original Equipment Manufacturers (OEMs) release a new platform every year. These machines may have differing capabilities, such that same workload will perform differently depending on the generation of a hardware to which it will be assigned. Furthermore, since workloads in a Cloud are heterogeneous, it creates task-to-machine optimal mapping problems for the Cloud schedulers often resulting in unexpected performance issues.

5.2 Some Background on Workload Characterization

Characterization of computer workloads has been extensively studied with many practical applications. Several existing studies for workload characterization have used targeted benchmarks for resource utilization. Zhang et al. [10] created and used a set of fixed traces to create profiles of workloads based upon machine utilization and wait times. Others have done work on characterizing different types of applications [3, 4, 7, 11–15]. Workload characterization helps to drive the design of a wide range of computing-related systems such as microarchitectures, computing platforms, compute clusters, data centers, and support infrastructures. Benchmark suites such as SPEC CPU 2006 [16] and PARSEC [8] are often used to evaluate improvements to the microarchitecture such as cache designs. These benchmarks typically measure performance of specific computing resources rather than for a particular workload, with the assumption that measuring and optimizing the identified resource is equivalent to optimizing for the workload category.

This section provides references to the underlying needs of certain types of workloads. Our work aims to classify workloads at a higher level and then map different classes to compute resource requirements. Cloud users can use this mapping to better optimize their applications, and Cloud Service providers can use it to accommodate their users. Another aspect is the time-based usage patterns for a given workload, such as the following scenarios:

- Close to payday twice in a month, bank deposits, and withdrawals activities create a usage spike;
- Close to an event like elections or sports, lots of traffic on social media, e.g., Twitter;
- E-commerce activity close to festivals/holidays, e.g., online gift ordering;
- College admission time online traffic, e.g., related to airlines, and university Web sites.

Jackson et al. [9] evaluate the performance of HPC applications on Amazon's EC2 to understand the trade-offs in migrating HPC workloads to Public Clouds. Their focus is on evaluating the performance of a suite of benchmarks to capture the entire spectrum of application that comprises a workload for a typical HPC center. Through their Integrated Performance Monitoring (IPM) [17] framework, they provide an analysis of the underlying characteristics of the application and quantitatively identify the major performance bottlenecks and resource constraints for the EC2 Cloud. Compared to the slowest HPC machine at NERSC [18], the uncontended latency and bandwidth measurements of the EC2 gigabit Ethernet interconnect are more than 20 times worse. For the overall performance, they conclude that EC2 is 6 times slower than a mid-range Linux HPC cluster, and 20 times slower than a modern HPC cluster, with most of the slowdown and performance variability coming from the interconnect. They show that the more an application communicates between nodes, the worse it performs on EC2 compared to a dedicated HPC cluster. Some examples are given below:

- CPU-intensive activity (like matrix computations using floating point arrays);
- Network traffic-intensive activity (like social media, sports event);
- Memory look-up activity (like vector data transfer).

Xie and Loh [19] studied dynamic memory behavior of workloads in shared-cache environment. They classified workloads based on susceptibility of the activity of their neighbor and the manner of shared-cache usage. They grouped workloads into four classes:

1. Applications that do not make much use of the cache (turtle).
2. Applications that are not perturbed by other applications (sheep).
3. Applications that are sensitive and will perform better when they do not have to share the cache (rabbit).
4. Applications that do not benefit from occupying the cache and negatively impact other applications (devil).

Their proposal is not mix incompatible classes, such as mixing a devil with sheep, but may be okay to run devil class of applications with turtle, as the latter does not use the cache.

Younggyun et al. [20] studied hidden contention for resources in a virtualized environment. They ran two VMs at a time on the same physical host and measured the slowdown compared to running the applications alone. They considered a number of benchmark applications and real-world workloads; in addition to measuring performance based on completion time, they reported following key characteristics related to underlying resource consumption:

- **CPU utilization**: is an indication of instructions being executed at a given moment and how busy is the CPU resource; it is measured as a percentage between 0 and 100%.
- **Cache hits and misses**: is an indication of memory access patterns. A cache hit means that accesses are localized and required data is found in CPU's local memory, also known as cache, which is very fast. However, if a program requires data not already present in the CPU's local memory then it is a miss. Data that has to be fetched from a remote memory location takes more time, slowing down the overall execution speed.
- **VM switches per second**: with multiple virtual machines running, also known as multi-tenanted environment. These VMs are switched on a round-robin or some priority scheme to allow a fair share of CPU for all VMs. A switch can also be caused by an event such as a cache miss. CPU is not left in wait state while remote memory is being accessed, thus giving compute time to another waiting VM. Too much context switching is not desirable as it causes an overhead in saving and restoring states of CPU registers and flags.
- **I/O blocks read per second**: refers to the network I/O bandwidth comparison on a per VM basis. VMs share I/O devices so if one VM is consuming more bandwidth then less is available for the other VM, slowing it down.
- **Disk reads and writes per second:** Similar to network, storage resources are also shared between different VMs. If a VM performs excessive disk reads or writes per second, then storage access is reduced for other VMs.
- **Total disk read/write time per VM:** Measuring time spent in disk accesses from within a VM indicates the nature of program running in that VM. Other VMs waiting to access storage will experience a slowdown.

Koh et al's analysis further shows that applications differ in the amount of performance degradation, depending on their own and other co-scheduled applications' usage patterns. This is expected and is not a new finding. However, Koh et al. clustered applications based on their performance degradation and mapped them to the ten characteristics that they measured. This provides a categorization and means to predict performance degradation in advance.

5.3 Top-Level Cloud Workload Categorization

Cloud Computing users have a variety of needs and wants. Cloud Computing platform (or services) suppliers have resource choices for allocation, planning, purchasing, and pricing. As Khanna and Kumar say, "*Systems must handle complexity autonomously while hiding it from users*" [21]. This is true and especially difficult in a Cloud Computing environment than any other computing system. Reason is the unpredictable and varied nature of tasks being executed in a Cloud Computing data center. Essential first step to meet this need is a clear, broad, ubiquitous categorization of various workloads. Workload categories can be split in two ways:

1. **Static Architecture**: refers to the implementation of solution architecture such as Big Data Storage or a parallel computational setup, e.g., is the application distributed in nature using parallel resources or is it serial in nature? Also, what is the size of database that is needed to support this application? Knowing in advance the type of architecture (parallel or serial) or the size of a database is even more important for layered applications that may be split across different servers and VMs, etc. An incorrect task assignment will result in poor performance for the application. In Cloud, it results for a customer asking for a small or medium size of a virtual machine, e.g., 2 GB memory and 1 CPU core for a task that needs a large size VM.
2. **Dynamic Behavior**: refers to the run-time nature of resource usage, or stress that a workload places on the computational resources in a data center; e.g., how much of CPU time, I/O bandwidth, memory or disk read–write accesses are performed during the application's run-time? In a time-based access pattern, we are interested in both the peaks and valleys of usage curves as well as the area under such curves, as shown in Fig. 5.1. A small VM that a customer has requested may be heavily or lightly used depending on the workload that runs in that VM.

Ideally, one wants to know the computational resources required to support an application as well as its loading pattern to ensure that expected performance goals can be met. As an example, a streaming video on demand may be simply reading raw data from the disk and serving video frames through the network, imposing very little CPU loading. It could also be engaged in real-time decompression or encoding of data for a client device, showing a different usage pattern.

Another, orthogonal, way to split workload types is between interactive and batch-mode jobs, e.g., where a real-time response is needed with smaller accesses or computations vs. overall completion efficiency for the large computational tasks. It is important to know the nature of a workload and its users' expectations in advance to do appropriate task placements so any (SLAs) and QoS requirements can be met. There is a subtle difference between the two. SLA refers to a legal agreement regarding the job's completion time and its data-security, etc. For example, USA's Health Insurance Portability and Accountability Act (HIPPA) of 1996 requires that

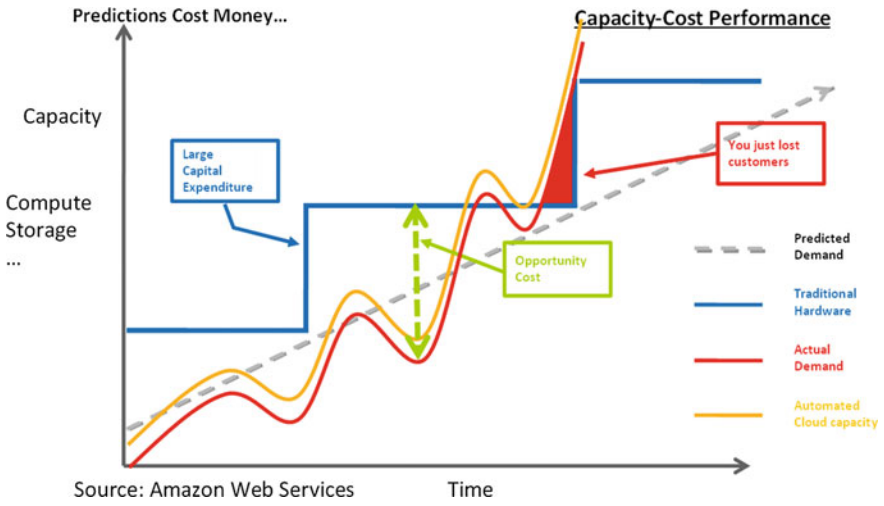


Fig. 5.1 Cloud Computing can be used depending on workload usage variations

any healthcare data must remain on the physical servers located within the USA. Thus, a Cloud company may have worldwide data centers, but it needs to ensure that any jobs from its US healthcare insurance customers remain within USA. Other SLAs may refer to the job needing to be done and results delivered within certain hours or days. QoS, on the other hand, refers to the run-time performance during the execution of a job. For example, a video being played on Cloud servers must be jitter-free and thus needs a certain network bandwidth assurance.

5.4 Cloud Workload Categories

This section lists and briefly describes the categories of workloads as differentiated by the customer and vendor viewpoint. These categories may have overlap, indeed might be identical, from other viewpoints. Some categories that are included are not technically Cloud Computing categories. These other categories are included for multiple reasons. The primary reason is to distinguish them for the Cloud Computing target of this chapter. A secondary reason is the theory that to an end user everything appears to be "in the computer or Cloud" whether it runs locally or remotely. History provides another reason for including non-Cloud workload categories in the table. In the late 90s, the concept of Application Service Provider (ASP) was proposed. In this model, the applications would run on servers connected via Internet to thin clients, which was called an Network Computer (NC) [22]. This model did not last. Therefore, SaaS on the Cloud may evolve; however, the categories described in this chapter will persist. Two key issues with computer workload categorization which are overcome in this chapter are based

upon perspective used in the definitions. The perspective in this chapter is to define the categories based upon fundamental concepts. For example, some categories are made of many tiny independent tasks as opposed to some single purpose huge tasks. One problem avoided by this approach is a categorization dependent upon the current state of the art implementation of computer resources. Our categories are definitely linked to and related to underlying computer resources; however, those resources do not define the categories. This means the categories remain even when the implementation hurdles are removed. Another categorization hurdle is the dependence upon a user's perspective. Our categories incorporate user's perspective; however, the categories still work with Internet, stand-alone computing, thin clients, cell phone, and automobile GPS systems. The final reason is for completeness. Any list of categories is open to questions about what is left out of the list. This final reason gives an opportunity to answer that question before it is asked.

To reiterate, these categories offer an operational view of Cloud Computing workloads. Although some of the related implementation resources (architectural, hardware, software, and infrastructure) concepts may be listed to help understand the categories, these implementation resource concepts are covered in another section. Although some low-level example hardware metrics, such as cache memory misses, may be cited for clarity in a particular category, those metrics are described in detail in another section of this chapter. The ultimate goal of the research in this area is to relate the ongoing measurement of the low-level metrics to the resource requirements. Secondary goals are to study how those resource requirements change at run-time for the purposes of SLAs, providing Cloud Services, hardware capital acquisition decisions, and future computer architectural design decisions.

The categories of computing workloads considered in this section include both Cloud computer workloads and workloads that are not commonly considered Cloud Computing issues. For example, high-end desktop graphics processing is not a Cloud Computing category. The categories in this section include: Big Streaming Data, big database calculation, big database access, Big Data Storage, Many Tiny Tasks (ants), tightly coupled calculation-intensive High-Performance Computing (HPC), separable calculation-intensive HPC, highly interactive single-person tasks such as video editing, Highly Interactive Multi-Person Jobs such as collaboration chats/discussions, Single-computer intensive jobs, Private Local Tasks, Slow Communication, Real-Time Local Tasks, Real-Time Geographically Dispersed tasks, and Access Control.

The **Big Streaming Data** workload category is characterized by an interactive initiation followed by long periods of huge amounts of data (usually video) sent to an end customer. The long periods of transfer may be interactively interrupted and restarted by the end user. This workload is usually measured by data throughput rather than latency; an example provider in this category is Netflix. The end user will not complain about, or even notice, a 100 ms delay at the beginning or in response to an interrupt. However, a 100 ms delay between delivered videos frames is obvious and irritating. Key underlying hardware resources are the storage server, streaming server, and the network speed to the end user.

The **Big Database Creation and Calculation** category is characterized by a large amount of data requiring simple yet massive computation efforts. For example, sorting and analyzing statistics for census data. Another example is the offline keying for search keys of databases. Example suppliers are the US Census Bureau, Yahoo, and Google.

The **Big Database Search and Access** workload category is characterized by repeated interactive requests or queries submitted to a very large database. The customer satisfaction for this category is dependent upon the response time or latency. The key resources that limit the latency are the database server software and organization, the disk storage, the network bandwidth from the storage to the database server, and the server load. Examples include Ancestry.com, the Mormon ancestry database, the US Census Bureau, Yahoo search, and Google search.

The **Big Data Storage** workload category is characterized by the integrity of large amounts of data which is periodically updated, usually by small increments, which occasionally requires a massive download or update. The end user sees this as an archiving or data backup resource. This workload category in the Cloud is multiuser as opposed to internal solutions that would have the IT department handle archiving. Example suppliers in this category are: Rackspace, Softlayer, Livedrive, Zip Cloud, Sugarsync, and MyPC. The success of this category is primarily availability, integrity, and security of the data. The speed (latency, throughput, storage, incremental processing, and complete restoration) is not the highest priority.

The **In-Memory Database** workload category is characterized by the integrity of large amounts of data that is rapidly accessed. The end user sees this as a real-time database. The limitation of this category is primarily size of the data. The speed (latency, throughput, storage, incremental processing, and complete restoration) is the highest priority. An example workload for this category is real-time business intelligence.

The **Many Tiny Tasks (Ants)** workload category is characterized by several very small tasks running independently. Very small is defined as completely fitting into cache with some room left over for other processes. This might be a lot of cache turtles (as defined by Xie and Loh [19]), not bothering any other process and never being flushed by other processes. These tasks may or may not have some interactive components. This category depends upon the ability to assign multiple processors. Each task is independent, so this can scale easily. Examples of this category are: small company monthly payroll, document translation, computer language syntax checkers, simple rendering or animation, format conversion, spelling and English syntax checkers.

The **Tightly Coupled-Intensive Calculation (HPC)** workload category is characterized by problems requiring teraflops of computing power. Notice that High-Performance Computing has multiple meanings. High grid point count matrices of partial differential equations are in this category. The base resource is raw compute power and large memory. However, implementing this with multiple processes adds the resource requirements of network speed and software partitioning. The tight coupling of these equations creates a dependency of the solution

at each point on the values at many other points. Classic examples of tightly coupled-intensive calculation HPC workloads are large numerical modeling simulations such as weather-pattern modeling, nuclear reaction simulation, and computational fluid dynamics.

The **Separable Calculation-intensive HPC** workload is characterized by large time-consuming quantities of calculations. This workload category is characterized by the ability to split up the calculations to be performed in parallel or small separable pieces. Examples of separable calculation-intensive HPC tasks are computer-aided engineering, molecular modeling, and genome analysis. For example, a section of a gnome slice can be analyzed completely separated from another section. In addition to these two workload categories, the term High-Performance Computing (HPC) is frequently used to identify a type of computing environment. In such cases, HPC is used to describe the computer systems capability rather than the workload running on the HPC.

The **Highly Interactive Single Person Tasks** workload category is characterized by a fast task with a single user. The key aspect here is response time latency. The examples for this category include online single-player gaming, online restaurant surveys, and online training courses.

The **Highly Interactive Multi-Person Jobs** workload category is characterized by the connectivity of jobs, such as collaborating chats/discussions. The resources include the server loading, the bandwidth between servers, and the bandwidth from server to end user. Another example of this category is online multiplayer gaming.

The **Single-Computer Intensive Jobs** workload category is characterized by high-speed large single user tasks that have significant user interaction. VLSI Integrated Circuit (IC) physical design (including timing analysis, simulation, IC layout, and design rule checking) is an example of this category. All computations for a designer are local, but database for the overall chip is on the network or Cloud.

The **Private Local Tasks** workload category is characterized by traditional single user tasks. Example tasks include word processing, spreadsheets, and data processing. These might be local for convenience or security. In the case of "thin" clients, these could be run in the Cloud.

The **Slow Communication** workload category is characterized by small amounts of information without a time limit for delivery. The best example of this is e-mail.

The **Real-Time Local Tasks** workload category is characterized by hardware measurements fed to a computer system. The key resources here are dedicated CPUs with interrupt-driven communications to the network. An example of this is a refinery or factory production monitoring system. Another example is a home security system. A third example is health monitoring. The nurses need immediate warning when the patient's heart monitor beeps. The hospitals Cloud database needs a continuous storage of each patient's data over the entire stay.

The **Location-Aware** workload category is characterized by utilizing auxiliary location input (such as GPS or telephone area code) data to small amounts of information without a time limit for delivery. The best example of this is following

a route on a map while walking or driving. Another example is using the location information as an additional security check for access while traveling.

The **Real-Time Geographically Dispersed Tasks** workload category is characterized by several dispersed hardware measurements systems feeding data to a network. The resource key here is dispersed dedicated CPUs with measurement and interrupt-driven communication to the network servers. An obvious example is weather reporting. Another example for this category is power grid management.

The **Access Control** workload category is characterized by user-initiated requests where the response is to another server authorizing more activity. The local example is the password on your computer. Most systems rely on passwords, and many security experts agree this is a very significant problem [23]. The local example is a bank ATM connected to a network (which may be a Cloud). A Cloud example is online purchasing. The resources required vary widely with the application and level of security desired. The Cloud is inherently open and flexible. A key issue with Access Control is the conflict with privacy. An example is Access Control to an e-mail server, where the administrator can observe or alter the accounts of various users, but each user can access his or her account only. The various Cloud providers need to verify the identity of individuals and systems using and providing services. As Bruce Schneier says, "Who controls our data controls our lives," [23] and so it is in the Cloud. Data for Access Control includes private data. Conglomerations of services (i.e., Cloud provider) must present a trusted environment meeting both security and privacy. The Access Control workload category must balance the trade-off between Cloud provider security and Cloud user privacy.

The **Voice or Video over IP** workload category is characterized by user-initiated requests where the response is through a server to each other. This is not presently a Cloud activity. The resources required are network bandwidth and user local compute power for data compression and decompression. An example is a platform using Chrome cast engaging in directing Netflix video traffic.

5.5 Computing Resources

This section lists the various resources that are used to provide Cloud Computing. Specifically, the resources considered will be the ones that have limitations set by SLAs, design decisions, capital purchases, software choices, and implementation choices. The relationship of these resources to lowest-level measurements is straightforward in many cases. The relationship of the resources to the higher-level categories is open to conjecture and ripe for extensive research. This chapter takes the first step toward connecting low-level resources to Cloud workload categories.

Persistent Storage is a relatively straightforward resource. The user estimates a need, gets an appropriate SLA, and uses until the resource needs an increase. While predicting future needs might be a bit difficult, other tasks are relatively easy. For

example, measuring usage, assigning drives, deciding on capital acquisition, measuring energy consumption, and storage system design are eminently solvable.

Compute Power/Computational Capability is measured by CPU time/cycles, number of cores available, number and type of computer nodes available, and the types and capabilities of CPUs assigned.

Network Bandwidth has several sub-categories, but we are only considering networks involving computers where bandwidth is a resource. For example, privately owned corporate entertainment is accessible only through subscription (monthly fees), requires possession of special hardware and/or software, and can be limited by connection type (DSL vs. dial-up modems). Examples include:

- Games: Playstation Network, Xbox Live, World of Warcraft, Eve;
- Movies: Netflix, Amazon Instant Video, Hulu;
- Radio: Pandora, Sirius, XM (last 2 are "satellite radio").

Broadcast Transmission receivers such as Global Positioning System (GPS) or radios. They require a special device added to the computer. For regular radio, it requires an additional receiver to plug into a port. For GPS, it also requires clear access to the sky to communicate with satellites and determine location, requires at least 4 satellites to be visible by the GPS (unless another positioning factor is known such as elevation), and is also limited by interference that creates false images. An example would be a computer using GPS on a real-time map-tracking program, which helps to navigate while one is driving around town.

Data Buses within a Server such as CPU to memory, cache to main memory, memory to disk, and backplanes on a card.

5.5.1 Data Buses Between Servers

Universal Serial Bus (USB): A full duplex system using a four-pin connector (power, data in, data out, and ground) to connect PCs with other devices at rates up to 5 Gb/s (version 3.0).

Network Types: A group of computers connected locally, not requiring a long distance carrier, typically using Ethernet over twisted pair; maximum data transfer rate is currently limited to 1 Gb/s using CAT6 cables.

- **Campus Network**: is larger than a LAN and smaller than Wide Area Network (WAN), this network is a collection of LANs in a localized area that does not need long haul (leased lines, long distance carrier, etc.) between computers but is usually connected with fiber optic cables using switches, routers, and multiplexers (MUXs).
- **Backbone Network**: is used to connect multiple LANs, Campus', or Metropolitan Networks by creating **multiple** paths from one computer to another over a different set of fiber.

- **Wide Area Network (WAN)**: connects different LANs, Campus', Cities, States, or Countries typically using fiber optic cable along with erbium-doped amplifiers, add-drop MUXs, regenerators, and digital cross-connects; current limit is 40 Gb/s using OC768 (each fiber).

Cache Memory requirements and utilization are important for planning, design, and job assignment to CPUs.

Software Capability determines what can be done and how efficiently it can be done. This includes operating systems, programming languages supported, and object code versions available.

Main Memory size is a straightforward resource in the sense of measurement, availability, and design. It is also amenable to real-time adjustments.

Cloud providers potentially use all metrics listed here. Efficiency of these metrics can be increased if a workload is known or a practical method of predicting its need is utilized. In some cases, bigger is better for computational resources. This may not be always feasible. Disk storage space can be increased by adding as much as needed but main memory, cache memory, or number of CPU cores requires either a design change or addition of new machines, IP addresses, racks, backplanes, etc. Often times, network bandwidth is capped out and cannot be increased further. This means that network traffic needs to be managed carefully by paying attention to heavy traffic trends and anticipating a change. This coincides with workload characterization and balancing as well as shifts in workload categories.

5.6 Example Workload Categorizations

Table 5.1 relates workload categories to computer system resources. Of course, all computing use many computing resources. However, some categories of jobs use particular resources to a much greater extent than others. This resource allocation affects all of the different groups. Service providers want to meet customer needs promised in a SLA without overspending on capacity. Customers do not want to purchase more capacity than needed. Computer manufacturers want to enhance hardware or software capabilities to improve performance due to any bottlenecks. The table includes computing jobs that are not Cloud related for two reasons. Primarily, the non-Cloud categories are included for completeness. And secondly, in the future, all computing may be in the Cloud.

5.7 Temporal Variability of Workloads

There are two different cases in which the workload category would change. The first case is when next step or phase of a job is a different category than the current category. Note that this happens in all jobs but only significant if the resource

Table 5.1 Characteristic computing resources for workload categories for the Cloud

Workload category	User view or example providers	Limiting resources	Level of Cloud relevance: "How Cloud heavy is this category?"
Big streaming data	Netflix	Network bandwidth	Heavy
Big Database Creation and Calculation	Google, US census	Persistent storage, computational capability, caching	Heavy
Big Database Search and Access	US census, Google, online shopping, online reservations	Persistent storage, network, caching	Heavy
Big Data Storage	Rackspace, Softlayer Livedrive, Zip Cloud Sugarsync, MyPC	Persistent storage, caching, bus speed	Heavy
In-Memory Database	Redis, SAP HANA, Oracle In-Memory DB	Main memory size, caching	Heavy
Many Tiny Tasks (Ants)	Simple games, word or phrase translators, dictionary	Network, many processors	Heavy
Tightly coupled calculation-intensive HPC	Large numerical modeling	Processor speed, processor-to-processor communication	Medium
Separable calculation-intensive HPC	CCI on Amazon Web Services, Cyclone™ (SGI) large simulations	Processor assignment and computational capability	Heavy
Highly interactive single-person	Terminal access, server administration, Web browsing, single-player online gaming	Network (latency)	Some
Highly Interactive Multi-Person Jobs	Collaborative online environment, e.g., Google Docs, Facebook, online forums, online multiplayer gaming	Network (latency), processor assignments (for VMs)	Medium
Single-computer intensive jobs	EDA tools (logic simulation, circuit simulation, board layout)	Computational capability	None
Private Local Tasks	Offline tasks	Persistent storage	None
Slow Communication	E-mail, blog	Network, cache (swapping jobs)	Some
Real-Time Local Tasks	Any home security system	Network	None
Location-Aware Computing	Travel guidance	Local input hardware ports	Varies

(continued)

Table 5.1 (continued)

Workload category	User view or example providers	Limiting resources	Level of Cloud relevance: "How Cloud heavy is this category?"
Real-Time Geographically Dispersed	Remote machinery or vehicle control	Network	Light now, but may change in the future
Access Control	PayPal	Network	Some, light
Voice or Video over IP	Skype, SIP, Google Hangout	Network	Varies

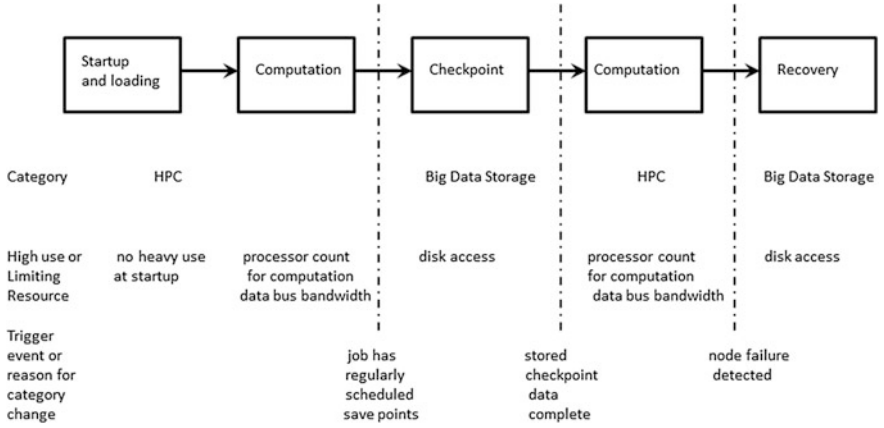


Fig. 5.2 Example of the changing Cloud workload categories for phases/step of an HPC job

requirements of the two categories are different. Consider the example of a very large HPC computing job, the stages or phases of which are shown in Fig. 5.2. The workload category is considered HPC and the primary resources are expected to be processors count for computational capability and node-to-node-to-memory data bandwidth. The first phase is the start-up and data-loading phase. This phase does not require significant resources to change the workload category. As the computation phase begins, the processors and data bandwidth resource availability are the limiting resources. Due to the high probability of a single node failure before the completion of a job, many HPC applications have intermittent checkpoints. As the job enters a checkpoint phase, the resource requirements change. Now, disk access becomes a limiting resource and the category changes to large data storage. After the check pointing is complete, the job returns to the computation phase. Should a node failure occur, the job enters the restore phase, which once again is a transition to a large data storage category with a high disk access requirement?

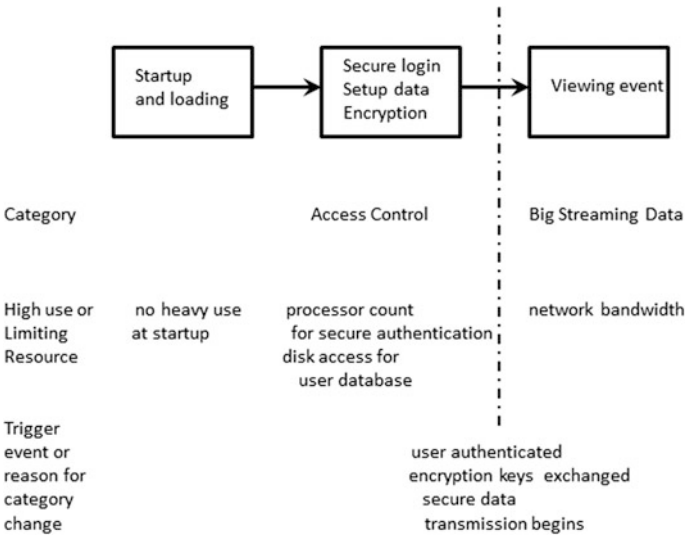


Fig. 5.3 Example of changing Cloud workload categories for phases for paid viewing of an event

Another example of a significant category change is paid subscription to online viewing of an event (such as a sports game or a band concert). Specifically, the initial phase is many users with high security login (that is payment) followed by a Big Streaming Data phase when the event starts. This example is shown in Fig. 5.3.

The second case is when the job is incorrectly categorized. The incorrect categorization can have an effect on billing; when for example, during the run it is found that a totally different resource is consumed, but the billing agreement set a very high (or low) rate for that resource. An example of this is shown in Fig. 5.4 (example A) where a job was initially characterized as a Big Database Creation and Calculation; however, during run-time it was found that the key usage was disk space, not calculations, so it is reclassified as Big Data Storage. Another example is shown, also in Fig. 5.4 (example B), where a job was initially categorized as Highly Interactive Multi-Person Jobs. So the resource to optimize for is I/O communication channels. However, upon measuring actual usage, it was found that the scarce resource was number of simple processing units, such as the Many Tiny Tasks (Ants) requires.

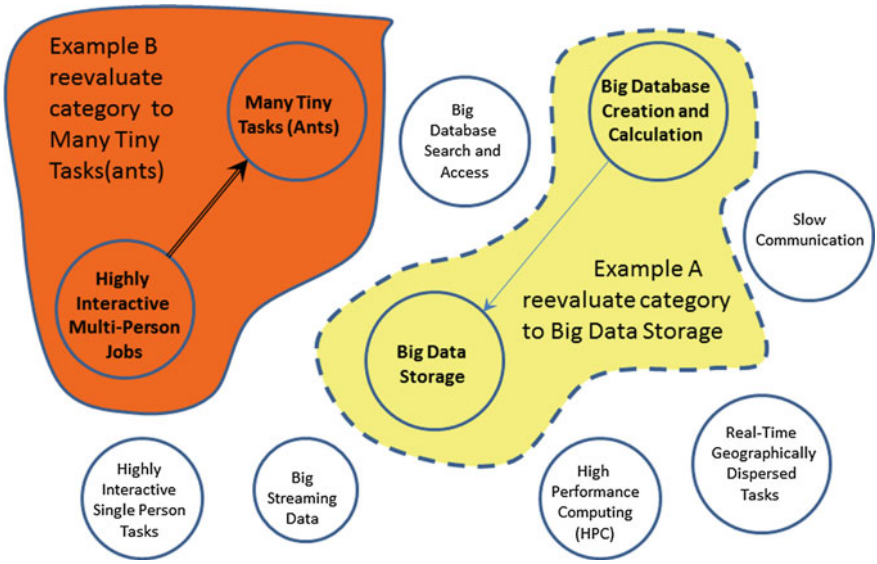


Fig. 5.4 Examples of mis-categorized workloads

5.8 Low-Level or Hardware Metrics of Computer Utilization

There are several performance counters available on most modern processors for monitoring hardware events. These can be sampled to gain insight on resource usage from the hardware perspective. For example, we can find whether floating point unit is used at all or, if it is used, and is it so heavily used that it ends up being a performance bottleneck. Below is a list of some metrics relevant to this study. A comprehensive list can be found in processor data sheets and optimization manuals. For Intel processors, Intel's VTune Amplifier XE [24] documentation provides a fair amount of details about performance counters.

Instruction per Cycle (IPC): The IPC metric is a good indicator of whether the CPU is being fully utilized or not. Although a coarse metric such as CPU usage may show 100% utilization, the CPU may not be running at its fullest potential due to microarchitecture bottlenecks such as cache misses, bus contention.

LLC misses: The LLC (Last Level Cache) misses are an important event to monitor as it gives an idea of how much memory traffic is being generated. The LLC misses counter will help quantify the extent by which the dataset surpasses the size of caches.

L1 Data Cache misses: The Level-1 cache is the fastest data provider to the processor and is thus crucial to maintaining a high throughput of instruction completion and a low latency. Monitoring L1 data misses allows us to gauge whether the core working set of an applications is captured in the first level cache.

Cycles for which Instruction Queue is full: The Instruction Queue is at the end of the fetch stage in a pipeline, or in the case of Intel processors, it is between the pre-decoder and the main decoder. If this counter is full, it indicates that the fetch engine is working very well. Depending on the instructions retired or the IPC, this could indicate that the pipeline is functioning at a high level of utilization if the IPC is high (which is good). On the other hand, if the IPC is low, it could indicate that the problem is not with the fetch engine but is caused by data dependencies or data cache misses, i.e., the backend of the pipeline.

L1 Instruction Cache misses: This is another indicator of front-end efficacy. If a large number of instruction cache misses are observed, it is either because the program sizes are too big for the I-cache, or because the program has several branches/jumps whose targets are not in the cache. In either case, a large number of misses indicate an issue with the front-end and the instruction fetch mechanism.

Cycles during which reservation stations are full: This would help us characterize the server application. For example, if the floating point unit (FPU) reservation station is full all the time, it is an indication that the application is probably a scientific application. By observing specific reservation stations, and with some knowledge about the typical usage characteristics of applications, we can identify the applications running on a particular server.

Number of lines fetched from memory: As servers have a large workload and the main memory is much larger than for desktop processors, this metric would give us an estimate of the pressure on the main memory. This is another indicator of the memory footprint of the application.

Dynamic, low-overhead low-level monitoring will also help to meet elastic demands in the Cloud. Animoto experienced a demand surge that required the number of servers grow from 50 to 3500 in three days [25]. Although this is an extreme case, low-overhead dynamic monitoring will help in assigning resources to varying demands much more quickly than depending solely on high-level measurements such as application response times. The goal of this is to make the changes seamlessly, without noticeable inconvenience to the end user.

5.9 Dynamic Monitoring and Cloud Resource Allocation

The categories presented in this chapter can be used in conjunction with low-level metric measurements to provide a more robust real-time, dynamic resource allocation. Previous works depend mostly on historical data and metrics such as application response times and high-level resource utilization measurements [1, 2, 26]. Others performed studies using performance counters to guide scheduling algorithms in single host systems avoid resource contention [20, 27–30]. Future work using the categories will better capture the nuances of workloads and their variability during the run, especially (but not limited to) in a Cloud environment where proper allocation of computing resources to the applications are essential and resource contention between customers' applications can be detrimental to the

overall performance of the data center. Research questions that must be answered include:

1. What is the extent of resource contention in various shared cluster setting (e.g., HPC applications running in the Cloud)?
2. Is there enough contention to warrant dynamic real-time monitoring with low-level measurements?
3. How often should these measurements be taken to maximize benefit over their drawbacks (sampling overhead, network bandwidth used for data aggregation, etc.)?
4. How will these measurements help to drive smarter resource allocation, e.g., how do low-level metrics measurements relate to and indicate which resource is the limiting one?

As an example, an HPC cluster augmented with low-level measurements and a robust aggregation framework will be able to answer all of these questions in the domain of HPC clusters. The local measurement of each host can be used to determine the extent of intra- and inter-node resource contention, and the aggregation of this data can be used to determine the efficacy of doing such measurements. Further study can also be done to determine the optimal frequency of measurement to capture the variability of the workload in time while keeping the overhead to an acceptable level.

As the VM density increases, co-scheduling VMs that are least destructive to each other on the same physical cores is crucial. Low-overhead dynamic monitoring of virtual machines allows prediction of the resources that a VM will require in the next time slice, enabling co-scheduling to minimize resource contention and power consumption. This can identify system-wide effects of an application's behavior; gaining insight on lightweight schemes of classifying applications or VMs at run-time; providing a framework for real-time monitoring, prediction, and optimization of system-wide power consumption.

5.10 Benefits to Cloud Service Providers

Since a Cloud Service Provider may not be able to anticipate the workloads that customers will be running in near future, it is not possible to plan ahead on placement of current jobs at hand, while leaving sufficient head room for new jobs. Similarly, an option to do live VM migration may not be available in the Cloud due to sheer number of servers and jobs in real-time. The best option is to build a profile of jobs run by a customer and use historic data to anticipate future workloads, and similarly build a time-based usage pattern to predict the near future usage. Another idea is for customers to build a small sample program to mimic the type of applications they will be running in the Cloud. They can then run this sample on a new Cloud machine, guess its capability, and use it to anticipate how their particular

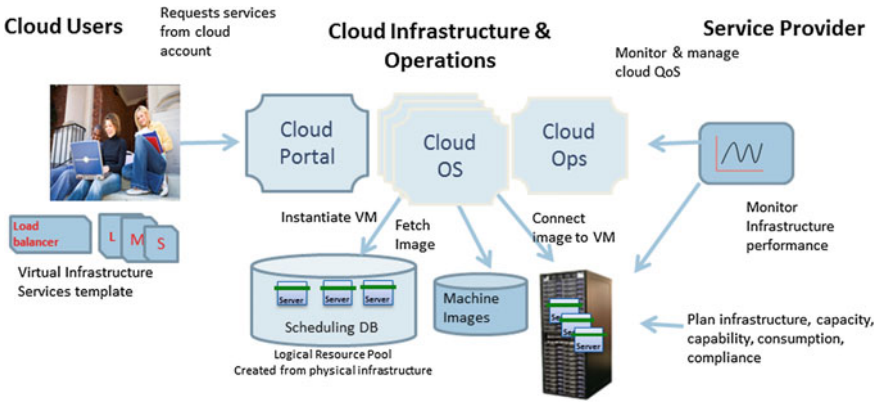


Fig. 5.5 SLA and QoS in IaaS Cloud. Cloud OS can benefit from fine-grained platform resource monitoring and controls to assure predictable performance, efficient and secure operations

job will perform relative to the actual application. Besides security, lack of predictability is the second largest reason why many enterprise customers are shy to adopt Public Clouds for running their mission-critical applications. Therefore, a mechanism that prospective users can use to calculate how their application will perform in Cloud will help them to migrate their applications to the Cloud.

As shown below in Fig. 5.5, if Cloud users can anticipate their needs in advance and forecast their demand parameters to the Cloud Service Providers, their jobs can be matched correctly to the available hardware. It will also identify a shared machine while avoiding heavy-duty users with fluctuating workloads. However, sometimes this is not possible so service provider must monitor the infrastructure and application performance to take corrective actions, such as moving jobs around or use the past information to do a better placement in future.

Frequently, a real-life workload is not composed of just 1 VM, but a combination of 2–3 VMs, such as to provide a storefront or other interactive service built on business logic and a database in the background. An example would be an airline's Web site, where users can do a search for flights, look at alternatives and their prices, select one, and apply their frequent flier miles or use a credit card to complete the purchase, at which time the seat in the flight's database will show as occupied.

This needs a 3-layer solution as shown in Fig. 5.6, with presentation layer at the top, followed by middle layer comprising of business logic such as how much of a flight has been sold and dynamic pricing based on what the competition is offering, and lastly, a database showing the seats availability. This represents a complex workload with the characteristics of an interactive workload, some computation in the middle and a database read–write access at the bottom layer. Furthermore, this workload's needs will vary in time depending on how many customers are browsing vs. buying at a given time.

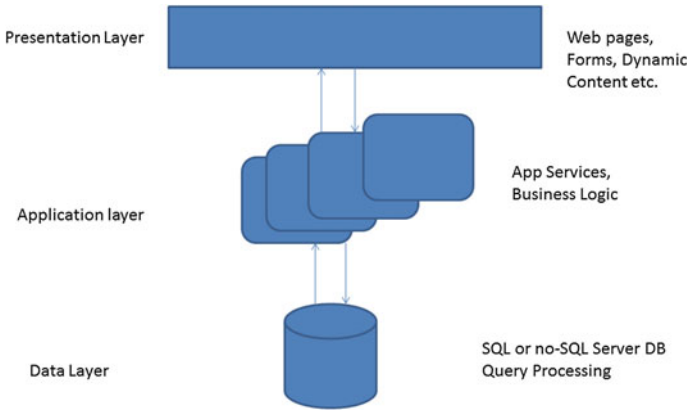


Fig. 5.6 3-tier Web Services architecture

Ideally, each VM can be placed on a different server optimized for the respective task and then inter-VM communications will bridge the gaps to give an appearance of a single Web-based elastic task running in the Cloud to the end users. However, any performance problems must be narrowed down and diagnosed in each layer for the whole system to work smoothly. Often time, users in Cloud do not know the hardware their VMs will get placed on, thus have to use hit or trial methods to find how many users can be supported by a VM at each layer.

Energy costs are the fastest-rising cost element of a data center. Power consumption is one of major concerns of these facilities, Clouds, and large IT enterprises. According to Senior Analyst Eric Woods, "Servers use 60% of their maximum power while doing nothing at all." There are also challenges in managing power at the appropriate times. One solution to monitor and control power consumption of servers is Intel's Data Center Manager (DCM) SDK [31], which enables IT managers to set power consumption policies and allocate desired power to different servers in a cluster based on their workloads and usage at a given time.

5.11 Summary

In this chapter, we described various Cloud workloads and optimization issues from the points of view of various players involved in Cloud Computing. A comprehensive categorization of various types of diverse workloads is proposed and nature of stress that each of these places on the resources in a data-center is described. These categorizations extend beyond the Cloud for completeness. The Cloud workload categories proposed in this chapter are: Big Streaming Data, Big Database Creation and Calculation, Big Database Search and Access, Big Data Storage, In-Memory Database, Many Tiny Tasks (Ants), High-Performance

Computing (HPC), highly interactive single-person, Highly Interactive Multi-Person Jobs, Single-computer intensive jobs, Private Local Tasks, Slow Communication, Real-Time Local Tasks, Location-Aware Computing, Real-Time Geographically Dispersed, Access Control, and Voice or Video over IP. We evaluate causes of resource contention in a multi-tenanted data-center and conclude by suggesting several remedial measures that both a Cloud Service Provider and Cloud customers can undertake to minimize their pain points. This chapter identifies the relationship of critical computer resources to various workload categories. Low-level hardware measurements can be used to distinguish job transitions between categories and within phases of categories. This relationship with the categories allows a technical basis for SLAs, capital purchase decisions, and future computer architecture design decisions. A better understanding of these pain points, underlying causes, and suggested remedies will help IT managers to make intelligent decisions about moving their mission-critical or enterprise-class jobs into Public Clouds.

5.12 Points to Ponder

1. **Many applications combine different workloads, e.g., an online Maps system to support mobile GPS, reading a large file first, building its graph and then doing I/O.**
2. **Different applications in a Public Cloud can offer opportunities to load balance and optimize resource usage, as compared to private data-centers serving only one type of load, e.g., video playbacks.**
3. **What is the best way to improve fault tolerance with multi-tier architectures?**
4. **What is the advantage of characterizing your workload in a Cloud?**
5. **From a Cloud Service Provider's perspective, how the knowledge of a workload's characterization may helps?**
6. **Can machine learning play a role to improve the efficiency of a Cloud data-center?**

References

1. Appleby K, Fakhouri S, Fong L, Goldszmidt G, Kalantar M, Krishnakumar S, Pazel DP, Pershing J, Rochwerger B (2001) Oceano-SLA based management of a computing utility. In: IEEE/IFIP international symposium on integrated network management proceedings, pp 855–868
2. Ardagna D, Trubian M, Zhang L (2007) SLA based resource allocation policies in autonomic environments. J Parallel Distrib Comput 67(3):259–270

3. Alarm S, Barrett RF, Kuehn JA, Roth PC, Vetter JS (2006) Characterization of scientific workloads on systems with multi-core processors. In: 2006 IEEE international symposium on workload characterization, pp 225–236
4. Ersoz D, Yousif MS, Das CR (2007) Characterizing network traffic in a cluster-based, multi-tier data center. In: Distributed computing systems, 2007 ICDCS'07. 27th international conference on 2007, p 59
5. Khan A, Yan X, Tao S, Anerousis N (2012) Workload characterization and prediction in the cloud: a multiple time series approach. In: Network operations and management symposium (NOMS), 2012 IEEE, pp 1287–1294
6. Bennani MN, Menasce DA (2005) Resource allocation for autonomic data centers using analytic performance models. In: Second international conference on proceedings on autonomic computing, ICAC 2005, pp 229–240
7. Bodnarchuk R, Bunt R (1991) A synthetic workload model for a distributed system file server. In: ACM SIGMETRICS performance evaluation review 1991, pp 50–59
8. Bienia C, Kumar S, Singh JP, Li K (2008) The PARSEC benchmark suite: characterization and architectural implications. In: Presented at the proceedings of the 17th international conference on parallel architectures and compilation techniques, Toronto, Ontario, Canada, 2008
9. Jackson KR, Ramakrishnan L, Muriki K, Canon S, Cholia S, Shalf J, Wasserman HJ, Wright NJ (2010) Performance analysis of high performance computing applications on the amazon web services Cloud. In: 2010 IEEE Second international conference on cloud computing technology and science (CloudCom), pp 159–168
10. Zhang Q, Hellerstein JL, Boutaba R (2011) Characterizing task usage shapes in Google's compute clusters. In: Proceedings of Large-Scale Distributed Systems and Middleware (LADIS 2011)
11. Arlitt MF, Williamson CL (1997) Internet web servers: workload characterization and performance implications. IEEE/ACM Trans Network (ToN) 5(5):631–645
12. Chesire M, Wolman A, Voelker G, Levy H (2001) Measurement and analysis of a streaming-media workload. In: Proceedings of the 2001 USENIX symposium on internet technologies and systems
13. Maxiaguine A, Künzli S, Thiele L (2004) Workload characterization model for tasks with variable execution demand. In: Proceedings of the conference on design, automation and test in Europe, vol 2, p 21040
14. Yu PS, Chen MS, Heiss HU, Lee S (1992) On workload characterization of relational database environments. IEEE Trans Softw Eng 18:347–355
15. Calzarossa M, Serazzi G (1985) A characterization of the variation in time of workload arrival patterns. IEEE Trans Comput 100:156–162
16. Standard Performance Evaluation Corporation (2006) SPEC CPU2006. Available: http://www.spec.org/cpu2006/, 8 Nov 2013
17. Skinner D (2005) Integrated performance monitoring: a portable profiling infrastructure for parallel applications. In: Proceedings of ISC2005: international supercomputing conference, Heidelberg, Germany
18. National Energy Research Scientific Computing Center (2013) NERSC. Available: www.nersc.gov, 8 Nov 2013
19. Xie Y, Loh G (2008) Dynamic classification of program memory behaviors in CMPs. The 2nd workshop on chip multiprocessor memory systems and interconnects
20. Younggyun K, Knauerhase R, Brett P, Bowman M, Zhihua W, Pu C (2007). An analysis of performance interference effects in virtual environments. In: ISPASS 2007 IEEE international symposium on performance analysis of systems and software, pp 200–209
21. Khanna R, Kumar MJ (2011) A vision for platform autonomy. Publisher Intel Press
22. Chapman MRR (2006) In search of stupidity: over twenty years of high tech marketing disasters. Publisher Apress
23. Schneier B (2009) Schneier on security. Publisher Wiley

24. Intel Corporation (2013) VTune Amplifier XE. Available: http://software.intel.com/en-us/intel-vtune-amplifier-xe, 8 Nov 2013
25. Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I (2010) A view of cloud computing. Commun ACM 53:50–58
26. Bacigalupo DA, van Hemert J, Usmani A, Dillenberger DN, Wills GB, Jarvis SA (2010) Resource management of enterprise cloud systems using layered queuing and historical performance models. In IEEE international symposium on parallel & distributed processing, workshops and Ph.D. forum (IPDPSW), 2010, pp 1–8
27. Knauerhase R, Brett P, Hohlt B, Li T, Hahn S (2008) Using OS observations to improve performance in multicore systems. IEEE Micro 28:54–66
28. Fedorova A, Blagodurov S, Zhuravlev S (2010) Managing contention for shared resources on multicore processors. Commun ACM 53:49–57
29. Fedorova A, Seltzer M, Smith MD (2007) Improving performance isolation on chip multiprocessors via an operating system scheduler. In: Presented at the proceedings of the 16th international conference on parallel architecture and compilation techniques
30. Nesbit KJ, Moreto M, Cazorla FJ, Ramirez A, Valero M, Smith JE (2008) Multicore resource management. IEEE Micro 28:6–16
31. Intel Corporation (2013) Intel Data Center Manager(TM). Available: www.intel.com/DataCenterManager, 8 Nov 2013

Chapter 6
Cloud Management and Monitoring

6.1 Motivation

Managing a Cloud operation is similar to managing any other shared resource. Imagine checking into a hotel after a long flight, hoping to catch a good night's sleep before next day's business meetings. But suddenly your next-door neighbor decides to watch a loud TV program. Its sound will awaken you for sure, but what's the recourse? Not much, as it turns out other than calling the hotel manager and making a request to noisy neighbor to lower the TV's volume. A similar situation can happen in a Cloud data center with multiple virtual machines (VMs) from different customers sharing the same physical server, or any set of resources. Your neighbor in this case may be another Cloud user, running a noisy job with too many memory or disk interactions. Obviously, this may cause a slowdown of other jobs running on the same-shared hardware.

6.2 Introduction to Cloud Setup and Basic Tools

In this section, we will introduce some Cloud Computing management terms, with reference to Amazon Web Services (AWSs). It is a large Public Cloud owned by Amazon, and the following terms are reproduced for readers' convenience. Detailed glossary is available at: http://docs.aws.amazon.com/general/latest/gr/glos-chap.html

(1) **Availability Zones**:
   I. A distinct location within a Region that is insulated from failures in other availability zones.
   II. It provides inexpensive, low-latency network connectivity to other availability zones in the same Region.

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6_6

III. A simple example can be a different data center (DC), or a separate floor in the same DC, or an electrically isolated area on the same floor.

(2) **Regions**:

I. A named set of AWS resources in the same geographical area.

II. A Region comprises at least two availability zones.

(3) **Instance**: A copy of an Amazon Machine Image (AMI) running as a virtual server in the AWS Cloud.

(4) **Instance Class**: A general instance-type grouping using either storage or CPU capacity.

(5) I**nstance Type**: A specification that defines the memory, CPU, storage capacity, and hourly cost for an instance. Some instance types are designed for standard applications, whereas others are designed for CPU-intensive, memory-intensive applications, and so on.

(6) **Server Performance**: Performance of a server depends on a full set of HW and SW components, as well as any other jobs running on that machine.

(7) **Application Performance**: An end-user application consumes CPU, memory or storage resources, and reports performance in transactional terms as ops/sec, or as latency to complete a task.

(8) **Performance Comparison**: Server performance is compared by ratio of ops/sec on any two servers.

(9) **Elastic Load Balancer**: Automatically distributes incoming application traffic across multiple Amazon Elastic Compute Cloud (EC2) instances. It enables you to achieve fault tolerance in your applications, seamlessly providing the required amount of load balancing capacity needed to route application traffic.

(10) **Load Balancers**: Elastic Load Balancer offers two types of load balancers that both feature high availability, automatic scaling, and robust security. These include the Classic Load Balancer that routes traffic based on either application or network-level information, and the Application Load Balancer that routes traffic based on advanced application-level information that includes the content of the request. The Classic Load Balancer is ideal for simple load balancing of traffic across multiple EC2 instances, while the Application Load Balancer is ideal for applications needing advanced routing capabilities, microservices, and container-based architectures. Application Load Balancer offers ability to route traffic to multiple services or load balance across multiple ports on the same EC2 instance.

6.3 Noisy Neighbors in a Cloud

One of the biggest challenges for an IT administrator in a Private or Public data center is to ensure a fair usage of resources between different VMs. If a particular VM does excessive I/O or memory accesses, then other VMs running on that same

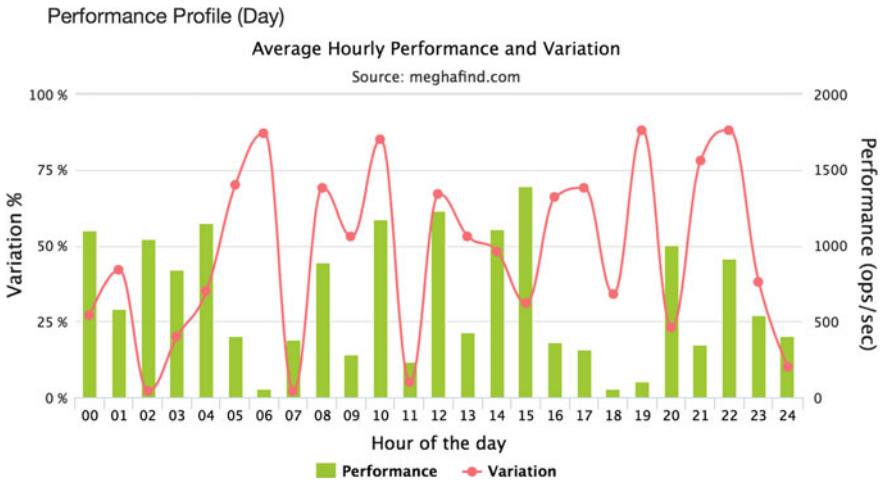


Fig. 6.1 Daily variation of a VM's performance on a server in a Public Cloud

server will experience a slowdown in their access of the same physical resource. This results in a performance variation experienced by a VM user over time, as shown in Fig. 6.1.

It turns out that while memory and CPU cores can be divided by a VMM (Virtual Machine Monitor) for different VMs on a physical server; there are components such as a memory, network, or disk controller that must be shared by all VMs. If one VM gets burdened with additional workload or malfunction, it will generate excessive I/O traffic saturating the shared controller. Just like a city highway, where an entry or exit ramp-up can become a bottleneck for new vehicles, a new request on the shared controller must wait for its turn. This will result in a slowdown of jobs or Web site responses. One way to minimize this is proactive measurements, before your customers notice a slowdown and start complaining. This is accomplished by an agent task for the sole purpose of monitoring a VM's performance over time.

6.4 Cloud Management Requirements

Cloud management tools can be of two types: in-band (IB) or out-of-band (OOB). In-band refers to an agent that typically runs in an OS or VM, collects data, and reports back for monitoring purposes. However, it may interfere with other processes running in that VM, slowing it down or creating additional resource contentions. OOB refers to monitoring tools that typically use a baseboard management controller with own processor and memory system, for observing the main server's health metrics. These do not tend to place additional CPU or memory load on the

server being monitored, but OOB cannot report detailed data as an IB agent due to their isolated nature. Rest of our discussion will not focus on OOB methods, which are mainly used to monitor the electrical health of a server by the data center operators, e.g., to check the voltage levels and frequency of a CPU, fan speeds inside the server box to ensure the proper cooling We will focus more on IB agents, which can report performance-related characteristics of a virtual machine or OS running on a physical server.

One way to limit the load placed by IB agents is to limit their measurement and reporting frequency, e.g., once every 5 min. Below is an example of the data types that are observed by an IB monitoring agent.

- CPUUtilization
- DiskReadBytes
- DiskReadOps
- DiskWriteBytes
- DiskWriteOps
- NetworkIn
- NetworkOut

These can be monitored in an alerting mode, e.g., raise an alarm when CPU utilization goes about 80% as CPU has a risk of saturation and slowing down other programs. Once the IT administrator gets such an alert, he or she has an option to move the offending application to another server, or kill it based of priority of other applications.

6.5 Essentials of Monitoring

Any Public or Private Cloud data center has a mix of servers, with different generations of processors, memories, networks, and storage equipment. These combined with the prevalent practice of multi-tenancy, where multiple users share the same physical infrastructure, result in different performances and varied user experiences even on the same server. These machines are often classified as small, medium, or large VMs for the end users to help decide what they need. However, two virtual machines may reside on different server hardware giving different throughput. Even if the underlying servers are similar, performance for two virtual machines of the same type will vary depending on which other tasks or jobs are running on their host servers. Furthermore, data center traffic conditions may change on different days of a week, or different hours of the same day.

All of the above factors affect run-time of your applications, resulting in different cost to finish a task. Monitoring and alerting data can also come from an application or business activity from which one can collect performance data, not just Cloud IT administrator provided solutions. A description of data types in the latter category is shown in Table 6.1.

Table 6.1 Nature of metrics collected and their units

Metrics name	Description	Units
CPUUtilization	The percentage of allocated compute units	Percent
DiskReadOps	Completed read operations from all disks available to the VM	Count
DiskWriteOps	Completed write operations to all disks available to the VM	Count
DiskReadBytes	Bytes read from all disks available to the VM	Bytes
DiskWriteBytes	Bytes written to all disks available to the VM	Bytes
NetworkIn	The number of bytes received on all network interfaces by the VM	Bytes
NetworkOut	The number of bytes sent out on all network interfaces by the VM	Bytes

Table 6.2 Frequency of performance metrics collection in a typical Public Cloud

Monitored resources	Frequency
VM instance (basic)	every 5 min
VM instance (detail)	every 1 min
Storage volumes	every 5 min
Load balancers	every 5 min
DB instance	every 1 min
SQS queues	every 5 min
Network queues	every 5 min

In order to limit the compute load placed by an IB agent on a server, its measurement frequencies are limited, but then short-term spikes may get ignored. An example of measured resources in AWS is shown in Table 6.2.

More importantly, an IT manager or Cloud user needs to worry about what to do with the data being measured. For example, if a VM is slowing down then it can be migrated to another server, or if not mission critical can be run at a later time. This may be possible if its output is not desired immediately, as some jobs can be run in the off-peak hours, e.g., late night hours or over the weekend.

6.6 Some Example of Monitoring Tools

Following are three contemporary and popular tools available for Cloud Computing users, first of which is provided for free by AWS:

(1) **Cloud Watch**: Amazon CloudWatch [1] is a monitoring service for AWS Cloud resources and the applications running on AWS. Amazon CloudWatch can be used to collect and track metrics, collect and monitor log files, set alarms, and automatically react to changes in AWS resources, such as starting new servers when CPU Utilization goes above a certain threshold. Amazon CloudWatch can monitor AWS resources such as Amazon EC2

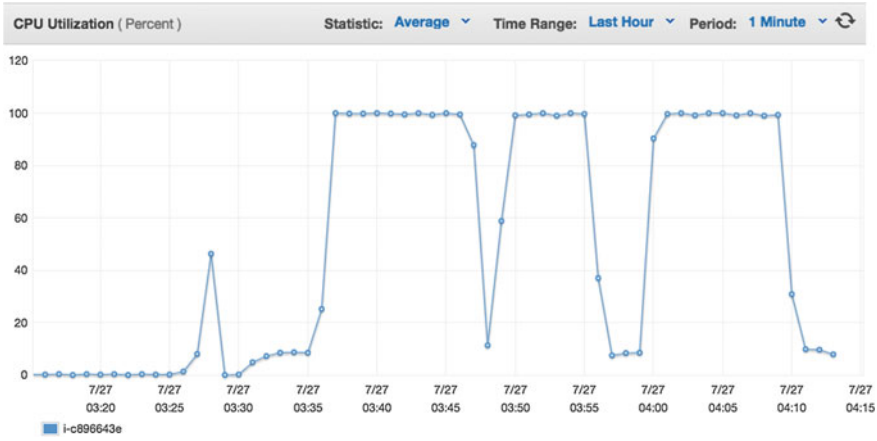


Fig. 6.2 CPU variations as observed in AWS Cloud Watch

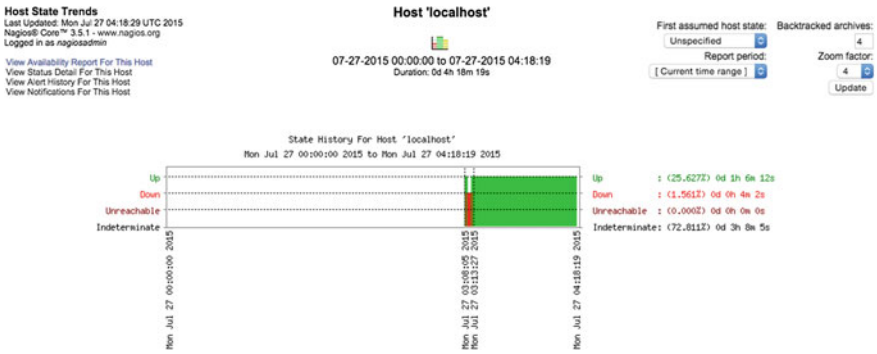


Fig. 6.3 An example of CPU host crossing a threshold and then becoming normal in Nagios

instances, Amazon DynamoDB tables, and Amazon RDS DB instances, as well as custom metrics generated by user applications and services, and any log files. One can use Amazon CloudWatch to gain system-wide visibility into resource utilization, application performance, and operational health. These insights can be used for smoother operation of user applications (Fig. 6.2).

(2) **Nagios**: is a free and open-source tool [2] to monitor computer systems, networks, and infrastructure. Nagios also offers alerting services for servers, switches, and user applications. It alerts users when observed quantities cross a threshold, as shown in Fig. 6.3 and alerts when the problem is resolved.

(3) **New Relic**: is an application performance monitoring (APM) solution [3] that uses agents placed in a VM, say in a Cloud or local server, to monitor how that application is behaving, as shown in Fig. 6.4. It is able to predict a slowdown in services before a user experiences the slow response.

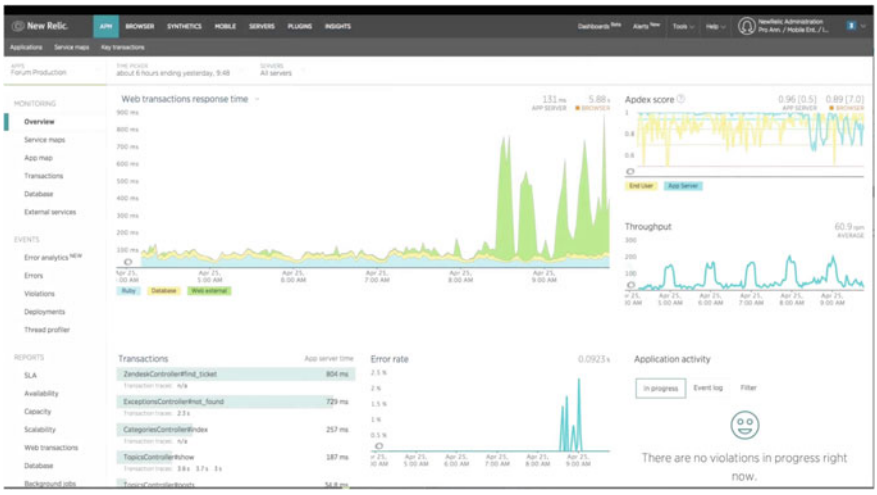


Fig. 6.4 An example of application performance monitoring (APM) in New Relic

6.7 Future Work

Any Public or Private Cloud data center has a mix of servers, with different generations of processor, memory, storage, and network equipment. These combined with the prevalent practice of multi-tenancy, where multiple users share the same physical infrastructure, result in different performances and varied user experiences even on the same server. These machines are often classified as small, medium, or large for the end users to help them decide what they may need. However, two virtual machines may reside on different server hardware giving different throughput. Even if the underlying servers are similar, performance for two virtual machines of the same type will vary depending on which other tasks or jobs are running on the same physical server. Furthermore, data center traffic conditions may change on different days of a week, or different hours of the same day.

All of the above factors impact run-times of users' applications, and result in different cost to finish a task. A Cloud user may want to maximize their performance or minimize their costs. However, there is no effective tool to predict this before selecting a machine.

We need a new solution to help unveil the physical characteristics of a Cloud machine and see what performance a user can expect in terms of CPU, memory, and storage operations. This will enable a user to model their application by selecting different proportion of these computational elements, or give a performance range, to see how different instance types will perform. Such a solution will also need to take into account applications that are multi-tiered, and use multiple Cloud servers, as it is the composite performance that an end user cares about. An example is a travel agency's Web site that upon a query goes in turn to query

various airlines and hotels' Web servers, to give a price comparison back to the user. In this scenario, not everything will be in the travel agency's control, but they can place a service-level agreement (SLA) to their data providers, or simply timeout and only list the components, which respond in time to avoid losing their customers. In a marketplace, where multiple vendors provide indiscernible services, price and performance are the key to winning.

6.8 Points to Ponder

1. **Distance causes network latency, so AWS has setup various regional data centers to be close to its customers and reduce delays. This is one way to mitigate network latency. Are there other methods you can think of?**
2. **In an AWS Region, there are different availability zones (AZs) to isolate any electrical fault or unintentional downtime. These could be different buildings or different floors within a data center building. What are other ways to build failure safe strategy?**
3. **What kind of metrics need be monitored to ensure the health of a data center?**
4. **What are the trade-offs of frequent versus in-frequent monitoring of a server's performance?**
5. **How long the monitoring data should be kept and for what purposes?**
6. **What's the role of monitoring if a SLA is in place?**

References

1. https://aws.amazon.com/Cloudwatch/
2. https://www.nagios.org
3. https://newrelic.com/application-monitoring

Chapter 7
Cloud Computing and Information Security

7.1 Background and Definitions

Information security can be viewed as including three functions, namely **access control, secure communications**, and **protection of private data.** Access control includes both the initial entrance by a participant and the reentry of that participant, or the access of additional participants. Note that a participant can be an individual or some computer process. The secure communication includes any transfer of information among any of the participants. The protection of private data includes storage devices, processing units, and even Cache memory.

The first function encountered is **access control**, i.e., who can rightfully access a computer system or data. The access control can be resolved at a hardware level with a special access device such as a dongle connected to the USB port or built-in security keys. Access control is usually addressed at the operating system level with a login step. An example of access control at the application level is the setting of cookies by a browser.

After access control is granted, **secure communication** is the next function, which requires encryption. The most commonly recognized function of a secure system is the encryption algorithm. The most commonly recognized problem in a secure system is the encryption key management. At the hardware level, the communication encryption device can be implemented at the I/O port. At the operating system level, encrypted communications can be implemented in secure driver software. At the application level, the encryption algorithm is implemented in any routine performing secure communication.

Some of the other functions and issues for security systems are hashing (for checking data integrity), identity authentication (for allowing access), electronic signatures (for preventing revocation of legitimate transactions), information labeling (for tracing location and times for transactions), and monitors (for identifying potential attacks on the system). Each of these functions affects the overall

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6_7

security and performance of a system. The weakest security element for any function at any level limits the overall security and risk. Weakest function is determined by technical issues (such as length of passwords) and also by user acceptance (such as blood samples in an extreme case if so required to authenticate a rare access).

In addition to accepting the security process, a user may have concerns regarding **protection of private data.** Specifically, the more information is required to ensure proper access, and the more private information is available to the security checking system. The level of security required is not universal. Ease of access is more important for low-security activities, such as reading advertisements. More difficult access is required for medium security such as bank accounts. High security is required for high-value corporate proprietary computations, such as design data for a next-generation product. Very strict and cumbersome access procedures are expected for nuclear weapons applications. These examples provide a clue to security in a Cloud Computing environment with shared resources. Specifically, in the same computing environment applications are running at a variety of security levels. Security solutions must also consider the trade-offs of security versus performance. Some straightforward increases in the security cause inordinate degradation of performance. As described previously, the security implementations can be done at multiple levels for each of the functions. Because security is a multi-function multi-level problem, high-level security operations need access to low-level security measurements. This is true of monitoring both performance and security.

Three environmental factors directly affect the evolution of information security: computing power available, growing user base, and sharing of information resources. First factor has been and continues to be the computer power available to both sides of the information security battle. Computing power continues to follow Moore's law with increasing capacity and speeds increasing exponentially with time. Therefore, while the breaking of a security system with brute force may take many years with the present computer technology, in only a few years the computer capacity may be available to achieve the same break-in within real time. Another environmental factor is the growing number of people needing information security. The world has changed from a relatively modest number of financial, governmental, business, and medical institutions having to secure information to nearly every business and modern human needing a support for information security. The sheer number of different people needing information security has increased the importance of different levels of security. The third environmental change that has a significant impact on the security is the sharing of information resources. That is the crux of this chapter.

Specifically, this chapter describes the information security structural changes caused by the spread of Cloud Computing, and more people across the world accessing Internet not just through PCs and browsers, but a myriad of mobile devices and applications. This has dramatically increased the risks and scale of potential damage caused by realization of a security threat on Internet.

7.2 Security Concerns of Cloud Operating Models

Security considerations vary depending on the users' interaction levels at the Cloud. Let us revisit the three operating models for Cloud Computing described below. We shall also discuss key requirements at each of these levels with security boundaries as shown in Fig. 7.1:

(1) **Software as a Service (SaaS)**: is the highest layer of abstraction focused on end users of Cloud, to provide them application access, such that multiple users can share the same application binary in their own virtual machines or server instances. Users want to ensure that their data is protected from other users, besides intruders, in the Cloud. Each user's persistent data may need to be encrypted to avoid un-authorized access. An application provider wants to ensure that users have a read-only access to the binary, and all patches are updated in a timely manner. This may require a restart of the application instance, if the binary version on disk changes.

(2) **Platform as a Service (PaaS)**: is focused on application developers, providing them access to elastic servers that can stretch in CPU cores, memory, and storage on need basis. Users want to ensure that platform services, such as various utilities can be trusted and there is no man-in-the-middle attack, such that user data is compromised. The PaaS requires strong middleware security. PaaS permits integrations of services so the security perimeters need a greater degree of access control. An example is machine learning and Hadoop clusters that are being offered by several Public Cloud providers. These require user jobs and data to be copied across several machines, so the protocols between software layers and services between various machines on a server cluster need to be secured.

(3) **Infrastructure as a Service (IaaS)**: is the bottom-most layer in a Cloud stack, providing direct access to virtualized or containerized hardware. Users want to ensure that hardware level services, such as various drivers and ports are protected from other processes running on the same physical server. An

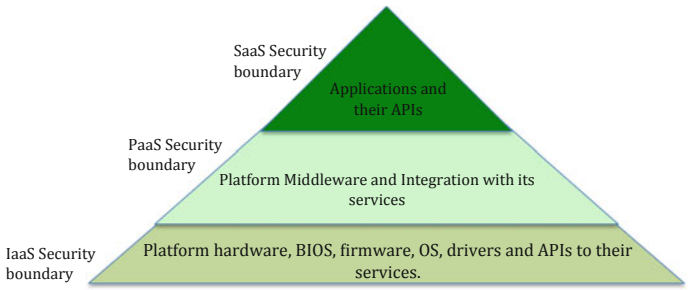


Fig. 7.1 Key requirements and security boundary at each level of Cloud Services

example is a virtual machine (VM) using the local disk to store data. After that VM's lifecycle is over, another user with a different VM then occupies the same disk space. Now the first user wants to be assured that the storage is zeroed out, and second user cannot access any residual data. This can be accomplished by per user encryption of shared services on a platform, such as storage, networking, and memory but that will cause run-time penalty every time such a resource is accessed. Another simpler method is to initialize the shared persistent state before the users are switched.

The previous paragraphs described several information security functions and proposed a relationship between the levels and details of implementation in a computer system. The levels and functions for security described throughout this chapter are summarized in Table 7.1.

7.3 Identity Authentication

Traditionally, identity authentication is applied when an individual requests access to a system. For this situation, the three elements or items used for identity authentication are what you have, what you know, and what you are [5]. An example of what you have is a house key. The use of the house key to unlock the door is sufficient identity authentication for the person to enter through the door. Examples of what you know are passwords, PINs, and your birthday. Examples of what you are refer to biometric information. These are pieces of information that you give to confirm your identity usually in conjunction with an item you have such as a bankcard or a check.

The use of passwords is a standard access control method. Passwords have the added value of being accepted by the user community. However, passwords are susceptible to brute force hacking. Therefore, some policies are implemented to increase the size and complexity of passwords. This leads to the practice of users writing down the long complex passwords on a post-it note affixed to the terminal screen or kept in an office drawer. This puts a human limit on the passwords, which leads to the direct prediction of "Password Exhaustion" [8]. The processing power available can readily hack passwords of a practical size. The conclusion is that the end of password usefulness is at hand. This conclusion is true if a system relies solely on password protection. However, passwords as part of a multi-pronged identity authentication provide a low cost, user acceptable increase in security. Passwords can be compared to the locks on car doors. The locks (or passwords) cannot protect against the directed high-powered attempts to gain access. However, they do provide adequate security for the lesser or casual attempts. A damaged car lock or a failed password warning provides a signal that a break-in was attempted. This allows intervention before any additional security steps are circumvented.

What you are is a physical characteristic. For example, a picture id or driver's license is frequently requested to authenticate your identity. A physical

Table 7.1 Information security functions and digital computer system levels

		Access control	Secure communications	Data protection	Monitoring
Software	User application	Some login usually relies on lower levels [1]	Usually relies on lower levels of implementation	Encrypt or disguise data [2]	Access logs
	Operating system (OS)	Login [1]	In-memory transactions	[1, 2]	Special processes as watchdogs
	Virtual Machine layer (VM)	[1, 2]		[1, 2]	[1]
	Hypervisor layer	[1]		[1]	[1]
	Software drivers	From OS	Encryption, security handshake	Encrypt data	
	BIOS/ FW-based system management layer	Privileged execution		Privileged access to certain memory locations	Log files
Hardware	CPU	From OS	Port and bus encryption, secure Caches	Separate secure registers and memory	
	Memory Cache/main RAM	Encrypted buses, hash checking tables	Data encryption	Partitioning and encryption	Interrupt logs
	Memory disk	Hash, checking tables [2–5]	USB data encryption	Encrypt disk storage, removable devices	Error logs [6, 7]
	I/O	Verify access id, such as Internet IP address	Encrypt transmissions; trust keyboard, mouse, and audio.	Security handshake, coding, encryption	Watchdog processes in hardware and software

characteristic that is measurable is a biometric. Example of biometrics includes height, weight, voice matching, face recognition, iris differentiation, and fingerprints. Measuring the individual characteristic and comparing that measurement to the value of authorized individuals can check biometrics. This is a problem in Internet world where the biometric measurement device is not under strict control, for example fingerprints [9, 10] or voice [11, 12]. Cloud Computing introduces a

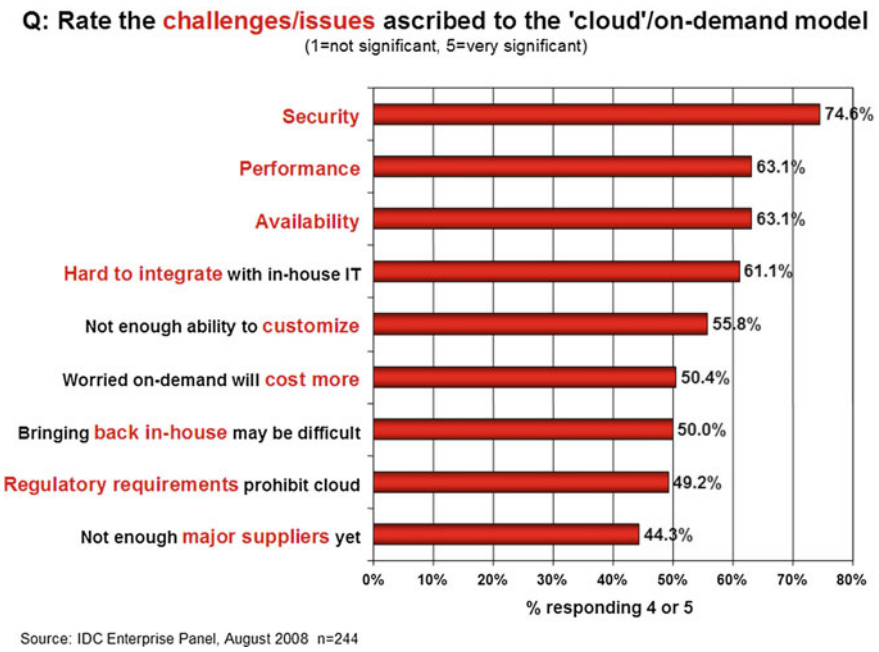


Fig. 7.2 Security is IT professionals' #1 concern in Cloud Computing

whole new challenge for identity authentication. As seen in a recent survey of IT professionals in Fig. 7.2, Security is their #1 concern.

An extensive discussion on the definition of Cloud Computing was in the previous chapters. For an identity authentication example, consider that when a program running in Cloud needs to access some data stored in the Cloud, then what you have and what you are criteria are irrelevant. However, the context of access request is relevant and can be used as described in an Amazon Web Services (AWS) Security Best Practices White Paper [6]. It follows an Information Security Manager System (ISMS), which is a collection of information security policies and processes for your organization's assets on AWS. For more information on ISMS, see ISO 27001 at http://www.27000.org/iso-27001.htm. AWS uses a shared responsibility and model, which requires AWS and customers to work together toward their security objectives.

AWS provides secure infrastructure and services, while its customers are responsible for securing operating systems, applications, and data. To ensure a secured global infrastructure, AWS configures infrastructure components. It provides services and features that customers can use to enhance security, such as Identity and Access Management (IAM) service, which can be used to manage users and user permissions in a subset of AWS services. To ensure secure services, AWS offers shared responsibility models for different type of service such as

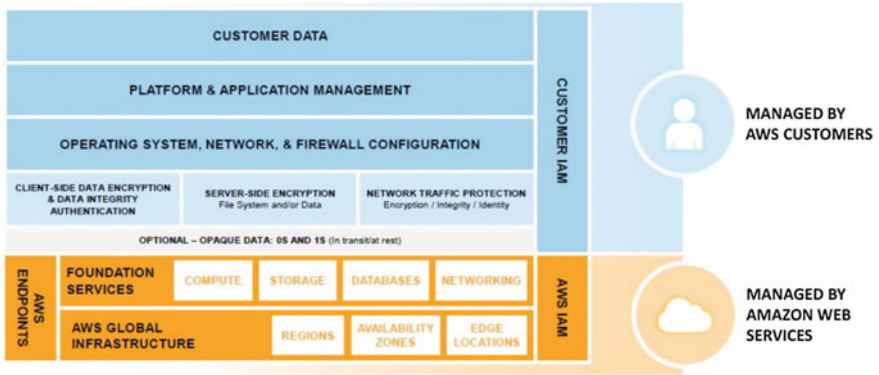


Fig. 7.3 Shared responsibility model for infrastructure services

- Infrastructure services;
- Container services;
- Abstracted services.

The shared responsibility model for infrastructure services, such as Amazon Elastic Compute Cloud (EC2), specifies that AWS manages security of the following assets:

- Facilities;
- Physical security of hardware;
- Network infrastructure;
- Virtualization infrastructure.

In this Amazon EC2 example, IaaS customer is responsible for the security of the following assets. Figure 7.3 depicts the building blocks for the shared responsibility model for infrastructure services.

- Amazon Machine Images (AMIs);
- Operating systems;
- Applications;
- Data in transit;
- Data at rest;
- Data stores;
- Credentials;
- Policies and configuration.

7.4 Secure Transmissions

Secure transmissions are required when data transfer is required between the islands of security via Internet. The model assumes a secure source environment and secure destination environment exists. The communication channel between the secure environments is open or un-secure. To protect the information to be transferred, it is encrypted whenever it is in an un-secure environment. The primary approach is to encrypt the information whenever it is "outside" of a secure environment. The primary tool for secure communication is encryption. Many aspects limit the security of an encryption system, such as key management, the strength of the specific encryption algorithm used, the size of the key, the time and effort required to crack an encrypted message. Much of the security research and security policy decisions are based on the time and effort required to crack an encrypted message. When implementing a secure system, a trade-off decision must be made between the security of the encrypted information (time and effort to crack the cipher) and the efficiency of the system (time and effort to encrypt the message). The trade-off is evaluated within certain boundary conditions, such as the lifetime and value of the information. A message about a transaction occurring tomorrow only needs to stay secured for two days, which limits the time and effort that will be available to crack the cipher. An example of this might be a major diplomatic announcement scheduled for the next day, whereas information that has years of lifetime value gives ample time for very significant effort to break the message. An example of this might be a detailed description of a long-term investment for a new drug discovery and its trial posing a competitive threat to other market players.

7.5 Secure Storage and Computation

The trade-offs between efficiency and security described for transmission also apply to the storage and computation. A common solution for security and integrity checking of networked storage environments is encrypted data file systems. The Cryptographic File Systems (CFSs) are a significant performance burden. Using a CFS is especially needed when the data storage is farmed out to un-trusted storage sub-providers [4]. A big difference is the wide range of storage lifetime. For storage such as copyrighted movies on DVD, there is a longtime value (several months or even years); however, for storage such as main memory there is a short-time value (perhaps micro seconds.) The emphasis in the main memory security should be on read–write efficiency. A small loss of time here has a huge impact on the performance of a computer system due to repeated operations.

Hence lighter and faster encryption schemes can be applied to data of ephemeral value, such as being held in a system memory, which will be short lived. For long-term data, companies such as financial institutions have invested in Hardware Security Modules (HSM). A HSM is a physical computing device acting as a vault

to hold and manage digital keys for strong authentication and provides crypto-processing services. This can be a plug-in card or an external device, attached directly to a network server. These HSMs are certified as per the internationally accepted standards, such as Federal Information Processing Standard (FIPS) in the USA to provide users with a security assurance.

7.6 The Security Players

As the previous sections demonstrate, the description of a secure system and its participants can get confusing. A common procedure for describing the participants in secure systems, scenarios, and security protocols is to use some nicknamed participants. In various descriptions, key personnel will maintain the same role. For example, person A will always represent the initiator or first participant in the given scenario. Not every scenario has every type of role. In fact, most scenarios have only a few of the roles. For convenience, the parties represented are given nicknames in addition to fixed roles. Several different publications have used this nicknaming technique when describing security systems and protocols [5, 8, 9]. Although these participants do not need to be physical human beings, for this chapter we are adding some virtual entities, as shown in Fig. 7.4. Security attacks at different layers of computer system are shown below in Table 7.2.

Cloud Computing presents a case with different classes of users using the same processes. That is, users requiring very high security are using the same operating system as the users requiring low security and therefore gain the same access. Also, the users requiring high security and the users requiring low security can run the same applications thereby having passed the same access required to run the application software. The implementation of typical real-world multi-level security

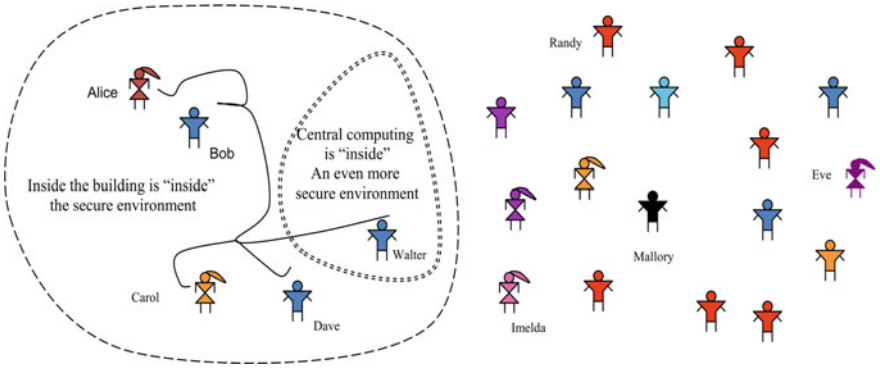


Fig. 7.4 Traditional computing security depends upon physical security

Table 7.2 Security attacks and challenges for digital computer system levels

		Unauthorized data or program changes (malicious by Mallory and accidental by Randy)	Unauthorized observation and copying (intentional eavesdropping by Eve, accidental leaks to Randy)	Denial-of-service attacks (intentional by Imelda and accidental by Randy)
Software	User application	Fake login or indirect access [10]	Usually relies on lower levels of implementation	Encrypt or disguise data [13]
	Operating system (OS)	Fake login, low-level instruction [1]	In-memory transactions	
	Virtual Machine layer (VM)	VM to VM communication [1]	Information leaks [2]	[13]
	Hypervisor layer			
	Software drivers	From OS	Encryption, security handshake	Encrypt data
	BIOS/ FW-based system management layer	Time date stamps [10]	Secure memory locations	Authentication for execution
Hardware	CPU	Information leaks [12]	Information leaks [12]	
	Memory Cache/main RAM		Information leaks [6, 12]	[11]
	Memory disk	Access privileges [6, 10]	Access privileges	

(MLS) is too rigid for most resource sharing activates (such as Internet) [14] and completely inadequate to meet the Cloud Computing challenge. The players described in this section have the same assignments in both traditional, Internet and Cloud environments.

7.7 Traditional Versus Internet Security Issues

The traditional approach to information security relied upon physical barriers. In this case, there is a clear physical boundary of what is inside the security perimeter and what is outside. The computing center access is available only to key personnel. The hardware and network connectivity to the main computers are physically

identified and can be checked for any taps. Individuals with a higher level of security clearance, such as system administrators, can have special hardware access lines. All of the individual users have a terminal in a controlled environment, such as in their offices or a terminal room.

Humans and devices can both be controlled for monitored access and activity. Devices to control access include card readers and key locks on doors. Human security guards can be posted at key points. Cameras are the primary devices to monitor activity. The most common human support for monitoring activity is the other workers. They can see people they know and what they are doing. People can also look for strangers by observing a person's uniform or badge to determine whether they seem to be doing an appropriate activity or not. The casual monitoring by coworkers is not only effective at catching intruders but it is also a deterrent to casual attempts at security breaches. This physical barrier provides two natural increases in security: limit repeated attempts and time available to perform a breach. In the case of repeated attempts, it becomes obvious to the most casual observer when the same unauthorized person tries to get in multiple times. And it is also obvious when a stranger keeps carrying files or disks out the door. As for the time, it takes minutes to gain access or gather information as the stranger walks around looking at computer screens on workers' desks.

As is shown in Fig. 7.5, the definition of "inside" the security perimeter and "outside" the perimeter is clear. When Alice wishes to get information to Bob, she can just walk over and stay inside the secure perimeter. Eavesdropping by Eve requires a much greater sophistication than the effort to secure the communication. For example, Alice may post documents on her bulletin board facing a clear window open to Eve's apartment across the street, providing her a way to view remotely. Even in the traditional environment information security leaks occur.

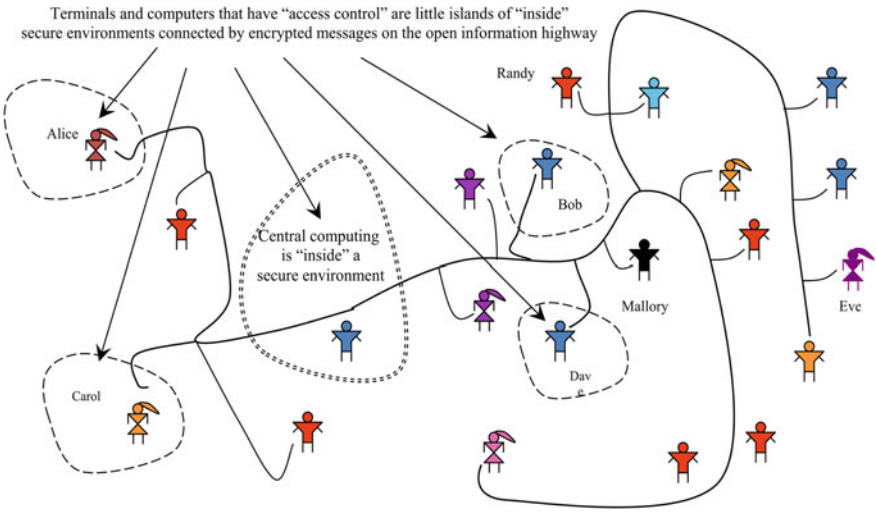


Fig. 7.5 Information security on Internet

Phone conversations, documents, and removable storage media did provide opportunities for information from within the secure environment to get to unauthorized individuals. However, usually enforcing policies based upon the physical boundary was sufficient to provide information security.

Internet created a new issue—that is connecting secure islands of information via open channels. The computing center with large computing and storage resources has controlled access and activities. This creates a large island of security where major resources still be controlled and monitored with humans and devices. However, now system operators have access via uncontrolled hardware lines, identical to regular user access lines. Unlike the traditional case, there is no casual monitoring by coworkers. As has been said in the cartoon world, "On Internet nobody knows your real identity." Also, the Internet provides an intruder with unlimited tries to gain access. After an intruder has failed to gain access, the failed attempts cannot be attached to the intruder. Each attempt may appear as a first attempt as IP addresses can be spoofed. Time to attempt repeated access is greatly reduced. Procedures to stop these intruders impact legitimate users. For example, one can stop the high number of repeated access attacks by limiting the number of false attempts by locking an account after some number of false attempts. However, this can prevent a legitimate user from accessing her account. This can lead to another form of security attack called denial of service (DOS). The idea here is not to gain access, but to prevent legitimate users' access by hitting the access attempts' limit and thus locking out (or denying service) to legitimate users.

The value of resource sharing via a network for a collaborative work environment is clear and established. One approach to access control to the collaborative environment is based upon people tagging [14]. On the Web, or any shared storage, the physical monitoring of access and activity is greatly reduced. In fact, the case of casual monitoring is completely eliminated. The integrity of shared data can be verified with digital signatures [4]. Integrity checking is needed when access is difficult to control. As shown in Fig. 7.4, the adversaries have access to the open channel. In addition to the open channel, the island of security concept has a time component to the isolation of a resource. A computer connecting to the central resource could be a shared resource such as a computer in a terminal laboratory. The next user sits down and has access to information that should have been protected, but hasn't been erased by the previous user. The concept of attacking persistence in main memory [15] is to read memory containing passwords or other secret data that is not consciously erased between users.

7.8 Variations and Special Cases for Security Issues with Cloud Computing

The levels of implementation and functions of a computer system are same in all three types of Cloud Computing. The difference is in the access and monitoring the usage of each element. In the case of Cloud Computing, there is no guarantee that a particular element be tightly coupled to another element for a particular process. For example, a database access request cannot rely on the fact that the application program on machine A has sole access through the operating system to the I/O port on machine A, as the storage can be shared between several machines. In fact, sharing in Cloud Computing is the very essence of efficiency to increase resource utilization instead of isolated resources. The security of Cloud Computing varies with the models used. Public Cloud model is used as it poses the greatest security challenges.

7.8.1 The Players

Traditional computing environments had a clear delineation between "inside" and "outside." Physically, "inside" might be in Alice's office or "inside" the bank building. With the dawn of networks, and especially Internet, the networks were partitioned as "inside the firewall" and "outside" which could be anywhere. This is one of the differences between a Public Cloud and a Private Cloud. Secure communication was only needed when the communication went from "inside" to "outside." With Cloud Computing, "inside" is not clearly defined as computers in multiple data centers across different geographies are pooled together to appear as a large virtual pool. Figure 7.6 shows the introduction of Cloud Computing relative to information security. This is true for both Public and Private Clouds because a Private Cloud can use a Virtual Private Network (VPN) that uses the open Internet to connect services only to a restricted category of internal clients. Eve and Malory could have access to the same set of physical resources that public is using. Some people can more easily trust a Private Cloud. Remember that within a Private Cloud, eavesdropping Eve and malicious Malory could be one of your coworkers or someone external. Also, in a Cloud environment, the unauthorized access could be by some random Randy that inadvertently slipped through the access barrier. The monitoring and response for security purposes must not only consider the level of secrecy and impact of a breach but also the category of intruders as shown in Fig. 7.7.

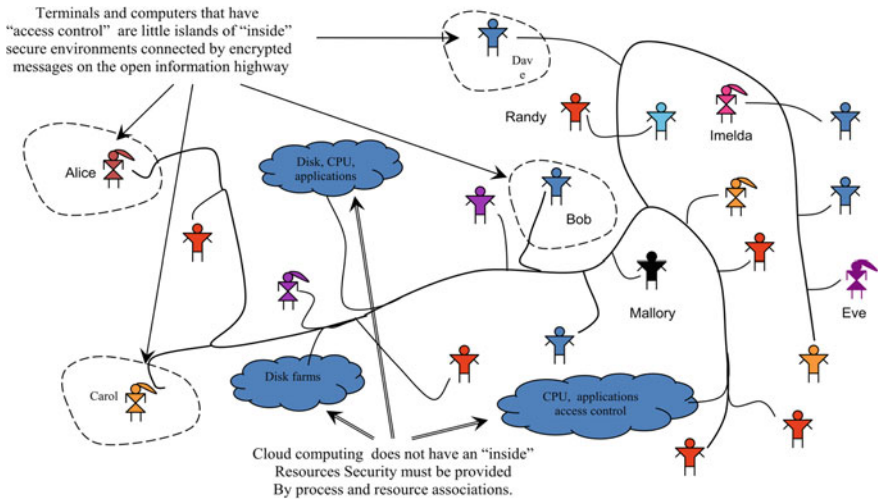


Fig. 7.6 Information security and Cloud Computing on the Internet

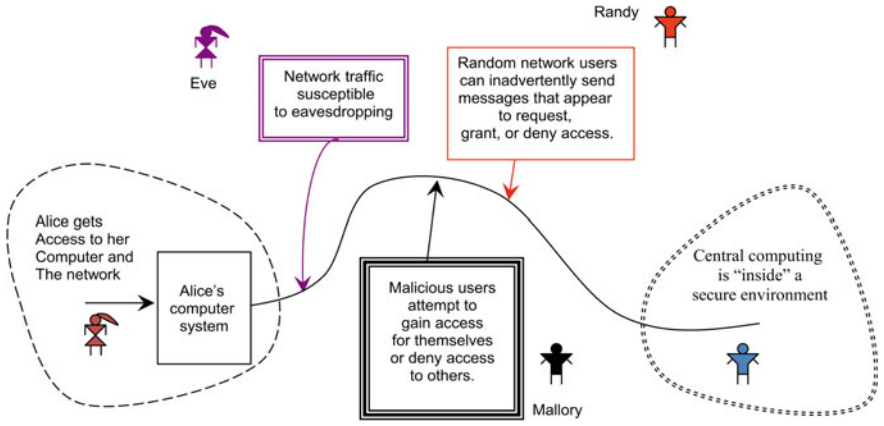


Fig. 7.7 Traditional Internet environment with communication between the islands of security

7.8.2 Secure Communication

The communication between islands of security through a sea of openness as described previously is solved by encryption all of the data in the open sea. This is shown in Fig. 7.8.

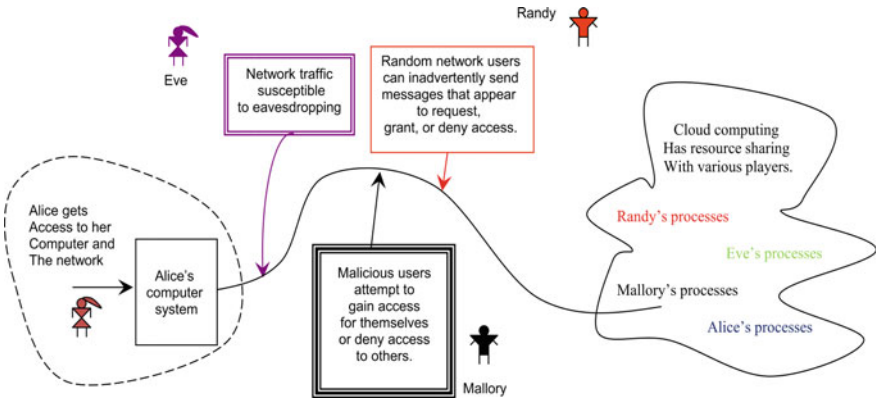


Fig. 7.8 Cloud Computing environment with no central island of security

7.8.3 An Example Security Scenario for Cloud Computing

As an example of security situation for Cloud Computing, consider a very large medical database. Alice represents the doctor's office administrator that must enter and retrieve patient information for a particular doctor. Bob represents an insurance company administrator that must enter and access patient information for a particular insurer. Carole represents a second medical database administrator with some overlap with the first database. Dave is the doctor at Alice's office and will need to access anything Alice does for reminders, changes, and for diagnostic tasks. The patients' pharmacy, hospital, or medical specialists may require additional accesses to the database. Consider that Eve is an eavesdropper for a private detective trying to illegally get health information of a patient. Encryption at the connection between two processors within the Cloud does not prevent Eve from using a process monitoring the operating system running the database application for a particular physician. Or, consider that Mallory maliciously accesses information about a patient in the files. He need not access it at the level of the doctor's office; he could access it as a potential insurance agency checking for future clients. Consider Imelda, an imposter who wants to pretend to be someone else. Imelda may have legitimate access as a drug store person checking on prescriptions. Imelda then uses the little bit of information she has on some patient to perform processes only allowed to Dr. Dave. By pretending to be in an emergency situation, the imposter may gain access to cause a security breach. For privacy concerns, the security breach could be simply gossip about a patient's treatment and condition. Imelda could also interfere with the insurance company by repeated unsuccessful attacks on their database by pretending to be Dr. Dave. Then as an imposter, she may cause a denial-of-service result such that the doctor's office cannot collect timely insurance payments. Even with a trusted arbitrator, Trent, on the job, each database would have a different basis for trust, so Mallory creates problems even if

just to get the databases to be inconsistent. So for security purposes, Wally the Warden monitors transactions. However, if Wally just guards the physical level of encryption, he will miss access problems (false identity authentication) as well as application-level breaches.

A security breach can occur at any computation level or for any security function. A very hard problem is for the Warden to distinguish between security attacks and Randy's random mistakes. In the Cloud Computing environment, errors by one person or process (either due to user input or weak security programming) can create the appearance of an attempted security breach. Overreaction to false breach attempts can cripple the performance of any large system. Strengthening the encryption implementation at the hardware level will not solve a weak access implementation at the application level. A simple verification of a user when logging in does not model the verification requirements of various processes, on various CPUs, accessing various storage devices in the Cloud Computing case. Thus, a more elaborate threat model is required for Cloud Computing. Victor the verifier must have a multi-level policy that is very fast and efficient. Both Wally the Warden and Victor the verifier need access to the virtual security service observer we introduced in the player's section. The virtual security observer Otto must have access to checkers at multiple levels of the computer architecture and across multiple security functions. Notice, that Otto does not implement the security solution. This would create the classic problem of, "Who is guarding the guardians?" Rather, Otto provides multi-level multi-function checking functions for both security policy implementations as well as performance trade-offs. Otto must have hardware level checking, communication checks, change or variation tracking, storage checksums, operating system messages, and application-level tags. Just as car door locks and login passwords provide an indicator mechanism for attempted break-ins, the data from Otto provides Wally with triggers to take further action.

7.9 A Few Key Challenges Related to Cloud Computing and Virtualization

In the scenario of Cloud Computing, people are concerned about the privacy of their data because they do not know where their data is being hosted. In a traditional computing center, anything inside the physical firewall boundaries is considered secure. However, a Cloud does not have clear boundaries because of its distributed features.

Customers need assurance that confidential data from their endpoint devices remains protected in a Cloud-based data center. In a computing system inside a data center, as shown in Fig. 7.9, various buses connect components. An adversary could physically tap those buses to gain information. This may be possible since an attacker could replace some modules or insert some components in a computer, for example, during the delivery phase. Hence, those buses are not trusted. It is hard for

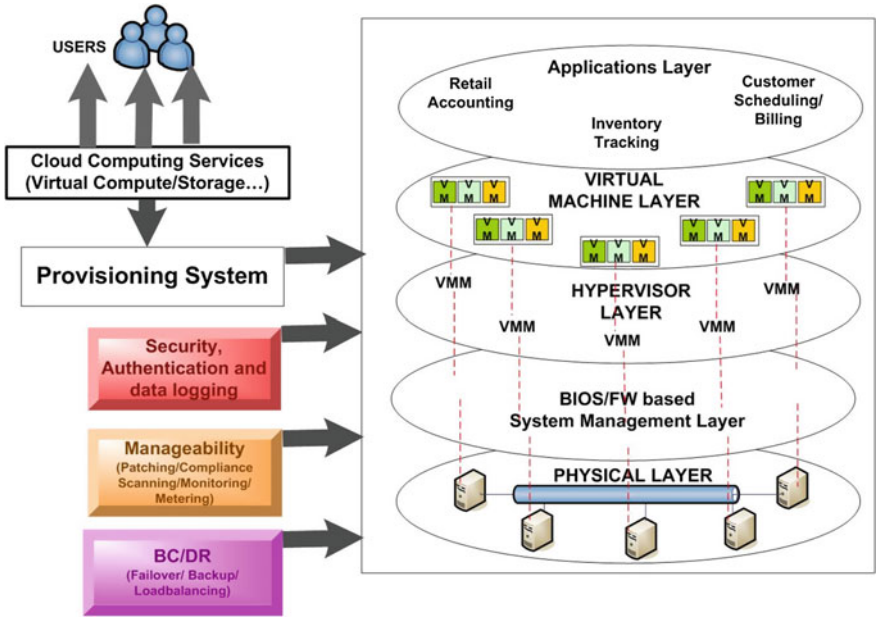


Fig. 7.9 Security inside a Cloud data center

each administrator or privileged personnel from a Cloud Service Provider to get a universal security clearance, especially when a third-party service provider is involved. The providers can use tools such as "QUIRC: A Quantitative Impact and Risk Assessment Framework for Cloud Security" to evaluate the security of their enterprise [16]. Cloud Service Providers need to gain their customers' confidence in the claims of data privacy and integrity.

Providers using Clouds face additional challenges due to dynamic provisioning of multiple virtual servers on shared physical machines. Such resource sharing is done to achieve the economy of scale. It implies that data from potentially competing sources can reside on the same disk or memory structure. Through an accident or by design, a computer process can violate the virtual boundary to access a competitor's data. Furthermore, a write or data-trashing activity may occur that can go un-noticed. Having secure backups, authentication and data logging for each user's activity inside a Cloud data center can contain such damage. Another specific problem of multiple virtual machines (VMs) is information leaks. These leaks have been used to extract RSA and AES encryption keys [17, 18]. However, it may create another problem of a large volume of logs storage, privacy issues of whom else can see a user's accesses.

7.10 Some Suggested Security Practices for Cloud Computing

While Cloud Computing and security practices continue to evolve, many users have already migrated their mission-critical applications and high-value data to the Cloud driven by the economic value and convenience factors. This has made both Public and Private Clouds attractive targets for security hackers. Hence, we propose following practices for the users and practitioners of Cloud Computing to ensure that their assets remain secure:

1. **Continuous Monitoring**: for any unexpected usage patterns or changes. You can't protect what you can't see.
2. **Attack Surface Management**: refers to access points that are exposed, and an organization needs to limit the devices or methods that can access its mission-critical data. Besides the obvious methods of using encryption, one needs to ensure that the devices that are authorized to access this data themselves are not vulnerable.
3. **No Residual Footprints**: Looking into bins for any trashed paperwork is an old spying practices. Online equivalent of this is to try reading the leftover bits in the memory or disk, after a target VM stops using these resources. By zeroing out the contents of memory and disk, a VM upon exit can ensure that next VM will have no residual data. This operation may cost some extra time and money, but well worth the trouble of avoiding your valuable data falling into wrong hands.
4. **Strong Access Control**: while obvious to any IT manager, many recent attacks took place through unexpected corners. Many companies use Internet-connected heating, ventilation, and air-conditioning (HVAC) systems without adequate security, giving hackers a potential gateway to key corporate systems. An example [19] shows how hackers stole login credentials belonging to a company that provides it HVAC services and used that access to gain a foothold on the target company's payment systems. A strong chain is as weak as its weakest link, so analyze your system and its access points, to find its most vulnerable spots.
5. **Damage Controls:** With always evolving sophisticated hacking techniques, no system is 100% hack proof and it is not a question of if, but when a security attack can happen on your Cloud infrastructure or data. Each organization and user needs a plan to minimize the damage in such cases. For an individual, it might be a matter of canceling their credit cards, changing banking passwords, or perhaps closing some online accounts if they are compromised. For an organization, mitigation strategies may be more complex involving an alternative control and command network, or quickly shutting down infected servers, etc. [20].

As Security Intelligence shows [21], available technological is not restricted to firewalls. Your Cloud solutions provider must protect the perimeter in the most effective way possible. As an example, any Cloud e-mail solution should have the following:

- Antivirus;
- Anti-spam;
- Information leakage control;
- The possibility to create specific rules for blocking, including attachments;
- E-mail monitoring.

While any Cloud application solution should have the following capabilities

- Intrusion detection tools;
- Application firewall;
- New generation firewall;
- Attack mitigation tools for DDoS attacks;
- Log correlation;
- Content delivery network.

Above requirements can be included in the service-level agreement (SLA) between the Cloud provider and users to ensure that both sides understand their roles and tools available.

7.11 Summary

Computer security issues exacerbate with growth of Internet as more people and computers join the Web, opening new ways to compromise an ever-increasing amount of information and potential for damages. However, an even bigger challenge to information security has been created with the implementation of Cloud Computing. This chapter gave a brief general description of information security issues and solutions. Some information security challenges that are specific to Cloud Computing have been described. Security solutions must make a trade-off between the amount of security and the level of performance cost. The key thesis of this chapter is that security solutions applied to Cloud Computing must span multiple levels and across functions. Our goal is spur further discussion on the evolving usage models for Cloud Computing and the increasing security cover these will need to address both the real and perceived issues, thus spurring new research in this area.

7.12 Points to Ponder

1. **Networking gear, such as Ethernet cards represent the first line of defense in case of a cyber attack, how can these be strengthened? (e.g., packet sniffing). Please see watchdog in appendix.**
2. **Recent storage crash in an AWS Public Cloud was caused by the scripting error of internal admin, how could it have been prevented?**
3. **Does having multiple customers in a Public Cloud increase its risk profile, as any failure will hit multiple businesses simultaneously? What operating system strategies you would recommend to prevent business impact?**
4. **Explain the role of attack surface management to minimize security risks? How would you reduce an attack surface?**
5. **How the regular monitoring of a server's usage activity may help to detect a security attack?**
6. **What are the security concerns due to a residual footprint?**

References

1. Christodorescu M, Sailer R, Schales DL, Sgandurra D, Zamboni D (2009) Cloud security is not (just) virtualization security: a short chapter. In Proceedings of the 2009 ACM workshop on cloud computing security, Chicago, Illinois, USA, pp 97–102
2. Ray E, Schultz E (2009) Virtualization security. In: Proceedings of the 5th annual workshop on cyber security and information intelligence research: cyber security and information intelligence challenges and strategies, Oak Ridge, Tennessee, pp 1–5
3. Naor M, Rothblum GN (2009) The complexity of online memory checking. J ACM 56:1–46
4. Cachin C, Keidar I, Shraer A (2009) Trusting the cloud. SIGACT News 40:81–86
5. Jain AK, Lin H, Pankanti S, Bolle R (1997) An identity-authentication system using fingerprints. In Proceedings of the IEEE, pp 1365–1388
6. AWS Security Best Practices, August 2016. http://aws.amazon.com/security
7. Juels A, Kaliski BS Jr (2007) PORS: proofs of retrievability for large files. In: Proceedings of the 14th ACM conference on computer and communications security, Alexandria, Virginia, USA, pp 584–597
8. Clair LS, Johansen L, Butler K, Enck W, Pirretti M, Traynor P, McDaniel P, Jaeger T (2007) Password exhaustion: predicting the end of password usefulness. Network and Security Research Center, Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA, USA. Technical Report NAS-TR-0030-2006
9. Gupta P, Ravi S, Raghunathan A, Jha NK (2005) Efficient fingerprint-based user authentication for embedded systems. In Proceedings of the 42nd annual design automation conference, Anaheim, California, USA, pp 244–247
10. Khan MK (2010) Fingerprint biometric based self-authentication and deniable authentication schemes for the electronic world. IETE Tech Rev 26:191–195
11. Shaver C, Acken JM (2010) Effects of equipment variation on speaker recognition error rates. In: Presented at the IEEE international conference on acoustics speech and signal processing, Dallas, Texas
12. Jayanna HS, Prasanna SRM (2009) Analysis, feature extraction, modeling and testing techniques for speaker recognition. IETE Tech Rev 26:181–190

13. Bun FS (2009) Introduction to cloud computing. Presented at the Grid Asia
14. Acken JM, Nelson LE (2008) Statistical basics for testing and security of digital systems for identity authentication. In: Presented at the 6th international conference on computing, communications and control technologies: CCCT2008, Florida
15. Rivest RL, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public-key cryptosystems. ACM Commun 21:120–126
16. Saripalli P, Walters B (2010) QUIRC: a quantitative impact and risk assessment framework for cloud security. In: Cloud computing (CLOUD), 2010 IEEE 3rd international conference on, pp 280–288
17. Ristenpart T, Tromer E, Shacham H, Savage S, Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In: Proceedings of the 16th ACM conference on computer and communications security, Chicago, Illinois, USA, pp 199–212
18. Osvik D, Shamir A, Tromer E (2006) Cache attacks and countermeasures: the case of AES. In: Pointcheval D (ed) Topics in cryptology—CT-RSA 2006. vol 3860, Springer Berlin/ Heidelberg, pp 1–20
19. http://www.computerworld.com/article/2487452/cybercrime-hacking/target-attack-shows-danger-of-remotely-accessible-hvac-systems.html
20. http://www.informationweek.com/Cloud/infrastructure-as-a-service/5-critical-Cloud-security-practices/a/d-id/1318801
21. https://securityintelligence.com/23-best-practices-for-Cloud-security/

Chapter 8
Migrating to Cloud

8.1 Cloud Business Models

While some applications, such as social networking, were born in the Cloud and are natural fits, many others traditional products such as Customer Relationships Management (CRM) were adapted by Salesforce.com and now are flourishing in the Clouds. Main attraction for migrating or creating applications in the Cloud is the economy of scale with pay as you need model for usage of IT resources. For customers, it translates to competitive products and cheaper services. There are three types of online business models of interactions between businesses (represented as B), and their customers (represented as C):

(1) **B2C**: Business-to-Consumers is a classic conversion of traditional brick and mortar businesses, such as bookstores to be online, as demonstrated by Amazon's first foray by putting the physical books for sale online and later offering e-books via Kindle, etc. In this model, a typical business can reach out to its customers faster and economically using Web Services. Many other examples abound such as booking airlines tickets online instead of calling or standing in a long queue to buy them.
(2) **B2B**: Business-to-Business is an example of aggregating many businesses offering similar services, for the ultimate good of consumers. An example is how Walmart manages its supply chain from vendor and inventory among different stores. If an item is selling well at a place, say a winter jacket because it snowed locally, while another store has excess supply, then instead of second store holding a clearance sale, merchandise can be quickly shipped from one place to another. This requires real-time tracking of different goods and services, with automation in decision-making. Another example is Dell ordering parts for a laptop, to assemble and ship it after a customer places the buy order on its Web site. Dell's vendors hold inventory of processors, memories, and hard drives in their own warehouses minimizing Dell's costs and risks of price drops, willing to supply the needed parts at a moment's notice. Such an inventory chain is called

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6_8

Just In Time (JIT), and Cloud Computing makes it even more efficient by linking Private Clouds of Dell with its Vendors' Clouds to make it easy to place orders, track shipments, handle payments, and returns, etc.

(3) **C2C**: Consumer-to-Consumer is a new model, enabled by Cloud Computing. Before that Internet connected people for exchanging messages or emails, but there was no central public place to store or access data, such as Ebay. Another example is Facebook, the largest social network till date, connects over a billion people across all continents to share pictures, messages, and stories. It has helped to create online communities and movements that have driven social and political causes in a manner not possible before.

8.2 A Case Study: B2C

A large business with many outlets and scores of customers spanning across geographical boundaries is the backbone of capitalism. However, managing good inventories in such a scenario is a logistical nightmare, as customer preferences vary across the Regions. So, winter jackets in a cold place may be sold out while they may be put on a clearance sale in another area due to unexpected warm weather. This may result in non-uniform pricing, driving customers to do comparison shopping across the stores. Worst case this may prompt businesses to have overstock or under-stock situation resulting in a financial loss. First one to take advantage of this situation on a large scale was Amazon, which started world's biggest virtual bookstore. They did not need to rent space in every community to sell books, thus could store remotely and mail them at a cheaper price. However, it needed Internet for customers to browse and place orders over the Web, and then wait for a few days for the book to show up. Their instant success caused a lot of local bookstores to fail, as they could not compete with Amazon's cheaper prices.

Jeffrey P. Bezos started Amazon in 1994 [1]. When many.com companies failed during the year 2000 crash, Amazon managed to survive and is now world's largest online retailer. Amazon has now expanded beyond books to any conceivable thing a customer may want to buy legally, including clothes, food, and electronics merchandise. They often hold limited time deals and offer a Prime membership for $99 per year in US, which delivers things free to a customer's home within 2 days. Amazon has attracted other sellers to advertise goods on the Amazon site, sometimes in a used condition at cheaper price, which opened up new revenue stream of commissions for Amazon. An example of e-commerce transaction is shown in Fig. 8.1

Amazon later expanded to Kindle, an e-book reader, which enables users to download electronic copy of a book instantly and allows carrying many books in a slim form factor of a 7″ tablet. This also enables reading at night without a light, and a special feature allows users to download first few pages of a book for free

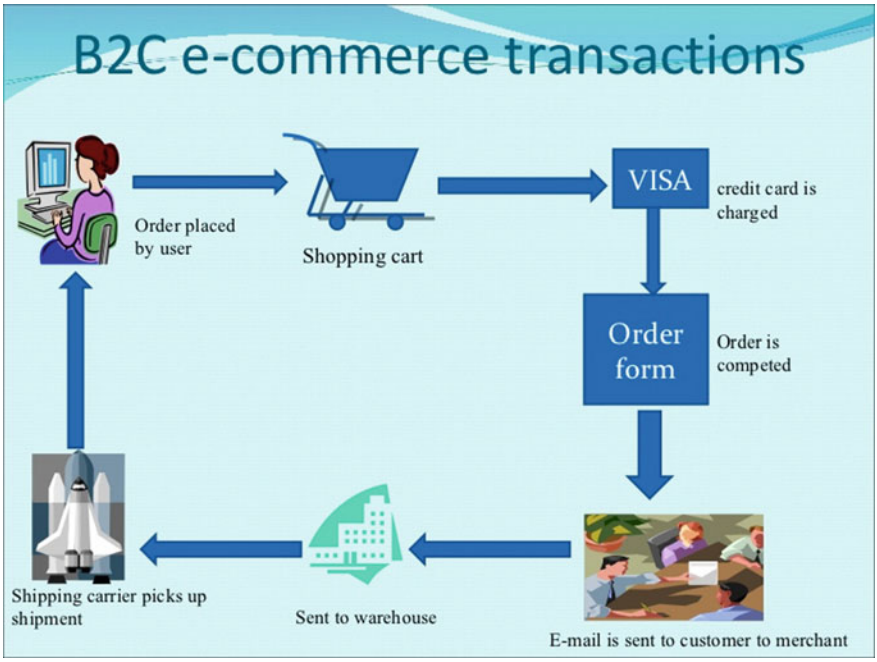


Fig. 8.1 Data and goods flow in a B2C transaction [2]

while they wait for the physical book to arrive by mail. Some other bookstores, namely Barnes and Nobles, tried to mimic this model but failed even after spending hundreds of millions of dollars.

As Amazon set up an enormous data-center to handle incoming customer connections, they entered into the Cloud Computing business by renting unused servers to other businesses and consumers. Their next attempt is to enter online entertainment business by offering movies and some original shows. Their stock price and high price-to-earning (PE) multiples are a testament of how investors feel about this Cloud success story.

8.3 A Case Study: B2B

According to one research firm [3], the total B2B digital commerce in US is expected to exceed $1 Trillion ($10^{12}$) by 2020, driven by a shift in buying behavior among B2B buyers. This is primarily driven by corporate customers having access to a similar intuitive shopping experience on B2B site as retail customers have on a B2C Web site.

There is a 90-year-old chocolate company started in Belgium, and now an international supplier with a large base in corporate gifting programs. Their business spans

Fig. 8.2 An example online gift basket [3]

multiple other industries such as automotive, legal, finance, retail, technology, and pharmaceutical. In order to reach and leverage its various corporate customers, suppliers, and partners, Godiva decided to use a Commerce Cloud platform offered by Salesforce.com. Incidentally, Salesforce itself runs out of Amazon's Elastic Compute Cloud (EC2) data centers. This Commerce Cloud offers support for critical B2B requirements, such as ordering and pricing lists, restricted login access and permission controls, sales, and discounting mechanisms during checkout.

Before adopting B2B Commerce Cloud, Godiva's corporate customers had to enter orders into a spreadsheet and submit it to Godiva's customer service teams, which manually keyed in data. This was time-consuming and error-prone process. Now business customers can directly enter orders and send chocolate gifts in large quantities to various recipients through a single Web order. An example is an auto dealer, who wants to send gifts to each individual car buyer in the previous month, as shown in Fig. 8.2. Their B2B interface is optimized for large split shipments, and

volume discounts are automatically applied to large orders. This previously required multiple complex transactions and unsatisfied customers due to human mistakes. Such a large-scale automation is only possible through the advent of Cloud Computing.

8.4 A Case Study: C2C

eBay is world's largest online marketplace [3] with more than 116 million active users that collectively drove more than $75B in sales volume in 2012. Each month, nearly one-third of all US Internet users visit eBay, to search for new and used goods. eBay started in America, but now has a footprint across all 5 continents, and with local sites in over 30 countries.

Pierre Omidya started an AuctionWeb from his apartment in 1995 to get rid of a few unused things, and concept caught on with other sellers and buyers joining in. He would get a small commission from every sale, and in 1997, his company got $6.7 million in funding from the venture firm benchmark Capital. eBay went public in September 1998 at $18 per share, but by early March 1999, the stock was trading at $282 per share. Within next 3 years, despite a major.com financial crash, eBay had 37 million customers and $129 million in new profits. Such is the power of a successful C2C business, as shown in Fig. 8.3 below.

Even though Amazon, Google, and Overstock attacked its business model, eBay continues to do well in the online auction market. Their reason for success is the

Fig. 8.3 C2C interaction on eBay [4]

initial focus on antiques and collectibles. Their second growing market segment is car and motor enthusiasts. Customers in the first segment are the antique lovers who are willing to bid very high for a rare item. Since eBay gets a certain percentage of gross sales, it contributes well to their profits, while bringing the antique lovers together. The motor lovers story followed a similar trajectory, using the credibility of a leading car collector Kruse, Inc., eBay expanded its categorical collections with sellers and new buyers making eBay Motors as one of the most successful target segments. Every three minutes during the fourth quarter of 2016, a car or truck was sold on eBay via a mobile device.

eBay's success is by ensuring that any commercial transaction between a buyer and seller can proceed smoothly and with full integrity [5]. This is assured by holding seller's payment until the buyer pays and receives the merchandise. Then each party ranks and rates the other. This rating is available to all other customers to see, so a bad remark can influence future sales of a seller. People care so much for a good review that they will follow-up and often refund or replace an item if the buyer is not satisfied. In cases where this does not happen, eBay is willing to step in like a policeman, and make things right for the aggrieved party in a dispute. Only credibility worth holding and displaying on an eBay site is one's public rating. Such is the power of Internet in offering a safe trading place, using a combination of Private and Public Cloud Computing services.

8.5 Summary

Several real-life examples were narrated of companies that have thrived only due to Internet and Cloud computing in their business models. Cloud Computing has ushered in a new era and proverbial gold rush of opportunities, expanding beyond the above list, such Uber for ride sharing, and BNB for housing, etc., which were not possible previously.

8.6 Points to Ponder

1. **Migrating to a Public Cloud may result in IT savings, but are there any potential pitfalls?**
2. **Business continuity is critical for many mission-critical operations in an enterprise. Can multiple Cloud providers be used to improve Business Continuity Practices (BCP)?**
3. **If local computing is needed, e.g., when Internet connections are not reliable, then a hybrid Cloud usage model is desirable with on-premise applications and data. In appendix, read about checkpointing, backup, disaster recovery, etc.**

4. **Under the following scenario, would an IT provider prefer a Private or Public Cloud?**
   a. Should a B2C IT service provider prefer a Private or Public Cloud?
   b. Should a B2B IT service provider prefer a Private or Public Cloud?
   c. Should a C2C IT service provider prefer a Private or Public Cloud?
5. **Can you think of a case, where a business tried a Cloud, but decided to not adopt it?**
6. **What factors may lead a business to consider a hybrid model?**

References

1. http://www.slideshare.net/annamalairamanath/case-study-on-amazon
2. http://www.slideshare.net/shynajain/b2c-business-models
3. https://www.demandware.com/uploads/resources/CaseStudy_Godiva_ENG.pdf
4. http://www.slideshare.net/SambathMetha/e-bay-presentation-27761391
5. http://teamcaffeine.wikidot.com/ebay

Chapter 9
Migrating a Complex Industry to Cloud

9.1 Background

As we have seen in previous chapters, there are three Cloud Services models (i.e., IaaS, PaaS, and Saas), which can benefit both businesses and consumers by migrating to Cloud Computing [1]. This has created a race for many businesses to adopt Cloud Services. However, that is easier said than done due to privacy and security concerns. While lower IT cost is the main driver, it is not the only consideration for Cloud Computing players and stakeholders. For example, some large financial houses and hospitals are concerned about the security of their data and prefer to maintain their own data centers, or create a Private Cloud infrastructure for their employees and customers. One such case is of Silicon Chip designers. It is interesting because most information in today's world flows through smartphones, laptops, and servers with backend data residing in the Clouds. Hospitals may be worried about exposing patient private medical records in a shared computing infrastructure of Public Clouds, which in turn will violate patient trust and local laws and have adverse effects such as societal or employer reaction. Similarly, large chip design houses are worried about the security of their product design data, lest it falls in the hands of a competitor. A Silicon Chip design company is comfortable putting its data on a Private Cloud, so internal teams located in different countries can collaborate on complex projects. However, it is reluctant to move the same data to a Public Cloud due to security concerns. Even if economic realities necessitate such a move, the design flows or tools used in a chip design are not available in the Cloud. Following study is an exercise to examine, on what would it take to move such a complex set of tools and design processes to Cloud. The steps used in this exercise can be applied to any set of complex tasks for an industry that is currently not in the Cloud, for determining the possibilities and challenges.

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6_9

9.2 Introduction to EDA

Electronics Design Automation (EDA) broadly refers to the software tools, methods, flows, and scripts that are used for very large-scale integration (VLSI) designs of silicon-based circuits, chips, and systems. The sheer number of transistors (now ranging into several billions) that reside on a single piece of silicon prohibits a handcrafted design and necessitates automation. The software technologies behind such automation have evolved over the last five decades, starting with university and large research laboratory tools such as Magic [2] (for basic layout placement), SPICE (for circuit simulation) [3], SUPREM [4] (for IC fabrication process simulation), PISCES [4, 5] (for semiconductor device simulation), SALOGS [6, 7] (for logic simulation), TEGAS [8] (for logic simulation), MOSSIM [9] (switch-level simulation), SCOAP [10] (for testability analysis), TMEAS [11] (testability analysis), ESPRESSO [12] (for logic minimization), SICLOPS [13] (for place and route), and Timberwolf [14] (for place and route) to several higher-level logic synthesis solutions in the 1980s. However, these technologies have since moved from academic institutions to commercial companies that specialize in providing EDA tools and necessary support for design teams. The growth rate for EDA companies has been cyclical during the past decade due to various factors and has opportunities for a greater growth [15].

So many industries are evaluating moving to the Cloud that researchers in Dublin have produced a systematic literature review (SLR) of 23 selected studies, published from 2010 to 2013 [16]. The authors produced a database of migrations of legacy software to Cloud, which provides useful information for the EDA companies considering such a move. Also, work has been done to identify performance bottlenecks in Cloud Computing [17]. The authors concentrate on network loading as affected by Cloud workload. Moreno et al. [18] present an analysis of the workload characteristics derived from a production Cloud data center and describe a method for analyzing and simulating Cloud workloads. However, they point out that: “Finally, the workload is always driven by the users; therefore, realistic workload models must include user behavior patterns linked to tasks.” We endeavor to meet this requirement by relating the general Cloud Computing benefits to the EDA industry-specific workload characteristics. Successful utilization of Cloud Computing by the EDA industry can create an opportunity for greater growth for both the EDA companies and the IC design community. Previous publications have discussed Cloud workload classifications [19] and Cloud security [20]. The purpose of this chapter is to apply that analysis to the EDA industry.

The EDA field is already several decades mature, and one wonders if there has been enough time to solve many key problems in the domain of silicon design automation (DA). The answer is no, as these problems are not static in nature. Many of these are NP-complete or NP-hard in nature, currently only with heuristic solutions. With Moore’s law of doubling the number of transistors available for integration every 18 months (current estimates are between 24 and 30 months), any heuristic solution tends to breakdown in a couple of product generations, as the

Table 9.1 Evolution of EDA over time

Years	EDA development	Characterization
1970–79	Handcrafted, small and simple designs	Nascent years
1980–89	Up to 1 million-transistor designs with schematic entry, layout editing tools	Roaring decade
1990–99	Layout automation, circuit simulation, logic synthesis, and analysis tools	Growing up
2000–09	1 billion-transistor designs using IP (intellectual property) blocks and SoC (System-on-Chip) products	Maturing
2010–present	On-premises hyper-scale distributed computing with some Private Cloud technologies	Predictable EDA

design solution space grows exponentially. This increases the cost of doing designs and is limiting new commercial design starts, further putting financial pressure on the EDA providers to recover their tool investments [21]. Thus, despite a growing electronics consumers market, the EDA industry enjoys a growth rate slower than for its design customers, at 9.5% over the past one year [15]. Paradoxically, a large number of small design efforts can readily use existing tools only if EDA costs come down allowing a broader adoption. This is especially true for new emerging applications, such as Internet of things (IoT), where per-unit silicon cost tends to be measured in cents instead of dollars. Broader access to large pools of inexpensive computing resources and availability of EDA tools to a broad number of current and potential users may result in reducing the cost of new designs. Then, more designs can be started by smaller teams and companies [21], especially in the emerging economies. Hence, a clear and present question for the EDA industry is, "How can Cloud Computing enable new growth opportunities?". To address this requires looking at the EDA workloads, and map them to Cloud workload categories. The Cloud workload mapping in this chapter applies to both Public and Private Clouds. However, EDA tools and flows have evolved over time and will continue to do so. Hence, a brief summary of the EDA history, shown in Table 9.1, is essential to understand the rationale behind our proposed workload mapping and to keep it updated with any future tools and flows.

9.3 A Brief History of EDA Tools and Flows

9.3.1 The Nascent Years of the 70s

Although computer-aided design (CAD) and DA activities started in early 70s, for the first decade almost all development was in large corporate design centers, using proprietary solutions. By the mid to late 70s, CAD and DA solutions branched into different engineering domains, such as automotive design, VLSI (very

large-scale-integrated) circuit design, and printed circuit board designs. The chips of that era were mostly handcrafted, and thus, the number of logic gates was limited to what a human mind could comprehend in terms of functionality. Circuits were of the simple adder and multiplier types, and point solutions included both analysis (e.g. equation solvers) and synthesis (e.g., simple logic minimization) type of stand-alone tools. During the 70s, digital electronic CAD and DA tools were mostly for board-level design, with some IC chip tools being developed in-house, such as at Intel.

9.3.2 The Roaring 80s

In the early 80s, many companies, of which the top three were Daisy, Mentor, and Valid, created specialized workstations for electronic design. Tools for interactive schematics and layout designs were becoming commonplace, even in the published papers of that era [22], including design synthesis and graphics with batch processing for large analysis problems.

Later, that decade saw specialized high-end workstations created to solve large design problems, such as Sun Workstations becoming a hardware Platform standard for using CAD/DA/EDA tools. The EDA industry began to focus on IC designs, while the term CAD became more common for mechanical engineering. The chips of that era were able to reach and cross the 1 million-transistor mark with Intel's N10 design in 1989. Very large analysis (such as design rule checking a whole chip or exhaustive digital simulation for design verification) was still performed on corporate mainframe computers, while incremental or small-scale analysis and simulation were performed on high-performance engineering workstations at each engineer's desk.

9.3.3 Growing up in the 90s

This decade witnessed a young EDA industry growing up fast, with many start-ups and subsequent consolidation phases, as EDA tools moved from specialized to general-purpose workstations. The solution vendors and their customers started to talk about flows and frameworks instead of individual point tools. Synthesis and analysis EDA programs were linked in automated script-driven flows with feedback loops. Heuristics-based solutions were used to address the size limitations for mostly NP-complete or NP-hard problems common in the EDA domain. There was an increasing interplay between automatic and human-interactive roles, especially in planning the layout and signal buses, to prune the search space for place and route tools.

The chips in this era routinely reached hundreds of millions of transistors, and tool limitations were overcome with the use of design hierarchies and cell libraries.

The latter also turned out to be a major productivity booster [23]. During the mid-90s, a trend included adding memory, and Cache to on-die microprocessors, to speed up the data processing times.

9.3.4 Maturing into the First Decade of Twenty-First Century

With the Internet boom came large data centers and massively parallel user tasks such as database searches. These clearly needed multi-core servers, instead of large single-core processor designs. The former became a convenient way to overcome the power and complexity limitations of large out-of-order superscalar single-core designs. This trend benefitted from the concept of hierarchical and cell-based modular designs developed in the previous decade. EDA evolved to increase inclusion of system-level concepts with the growth of System On a Chip (SoCs), comprising of Intellectual Properties (IPs), cores, and custom logic integration supported by the tools.

Using the new EDA solutions, large chip designs, such as Intel's Itanium microprocessors, crossed the 1 billion-transistor mark. This era saw an increased emphasis on low-power designs with the popularity of smart phones, tablet computing, and various new form factors. As individual processors became faster, a number of consumer products evolved to put more functionality in software on standard hardware cores, thereby decreasing the need for many new design starts in the industry. According to Prof. Michael Porter's competitive five forces theory, in any industry, it is desirable to have many customers instead of a few large customers to minimize the product pricing pressures [24]. This applies to the EDA industry as the uneven split of customers between high-end large designs done by a few companies and the smaller size designs by many companies presents a growth opportunity [21].

9.3.5 From 2010s till Now, EDA Stable

With the advent of the Internet, higher bandwidth of public and private networks, and consolidation of servers in large data-centers came the birth of Cloud Computing. Compute power required to complete a new chip design in acceptable time exceeded the available compute capacity in many design companies, thus creating a natural problem and solution match. However, the lack of security, or the perception of lack of it, prevented movement of proprietary chip designs into public Clouds [25]. Also, licensing models for EDA tools have not evolved much since the 90s, requiring user companies to buy yearly contracts for tool licenses, instead of hourly basis pay as you go software rental models popular in the Cloud.

Both of these factors are contributing to the limit on how many simultaneous designs a company can afford to do, thereby also limiting the revenue growth of the EDA industry. Solving this dilemma provides an opportunity for the EDA and chip design industries to grow.

Furthermore, a relentless pressure to reduce the project schedules and costs is causing most design teams to adopt SoC methodologies [26]. This has led many EDA companies to venture into new territory to provide large design components, often sold as IP (Intellectual Property) blocks, for specific functions such as audio or video processing, USB support, and even CPUs as building blocks for larger and more complex chips. This is one step away from an EDA company doing the complete designs, but does not want to directly compete with its customers. These points provide a basis for EDA companies to explore growth opportunities of their services and products in the Cloud. This situation has led to the growth of on-premises hyper-scale distributed computing with some Private Cloud technologies.

9.4 EDA Flow Steps Mapping to Cloud

A typical design flow has dozens of intermediate stages, with up to hundreds of tools generating thousands of design data and log files, requiring millions of CPU hours to complete a design. While actual design flows and tool versions may differ across individual projects, making it hard to share and maintain a consistent design environment, there is a desire to share EDA tool licenses between many design teams to optimize the software cost. An example design flow is shown in Fig. 9.1, and as an illustration, the mapping of some individual steps to the Cloud workload categories. Ghosh et al. [27] propose a structured approach for Cloud workload analysis and migration aimed at generic Web-based workloads such as ERP and CRM solutions, whereas our approach below goes into the details of silicon design steps to accomplish mapping to Cloud workloads.

Instruction set design involves high-level design decisions about the fundamental contract between hardware and software for the target design. Instruction set architecture (ISA) represents the machine-level instructions that a target architecture will natively support. This is an important step, as all optimizations and design decisions follow from this step. For example, if a specific architecture does not need to natively support the division operation, the architects do not need to design division units in the chip and can thus reduce the number of transistors on the chip. In this case, division function can still be supported with software doing repeated subtractions. If the target architecture needs to process large arrays of floating point elements, performing the same operation on all the elements of an array, the architects may choose to support this directly through the instruction set, by adding vector instructions to the ISA. An example of such an addition is the set of SSE instructions that Intel added to the Pentium 3 processor in 1999. Such decisions are made after careful examination of the performance of most popular

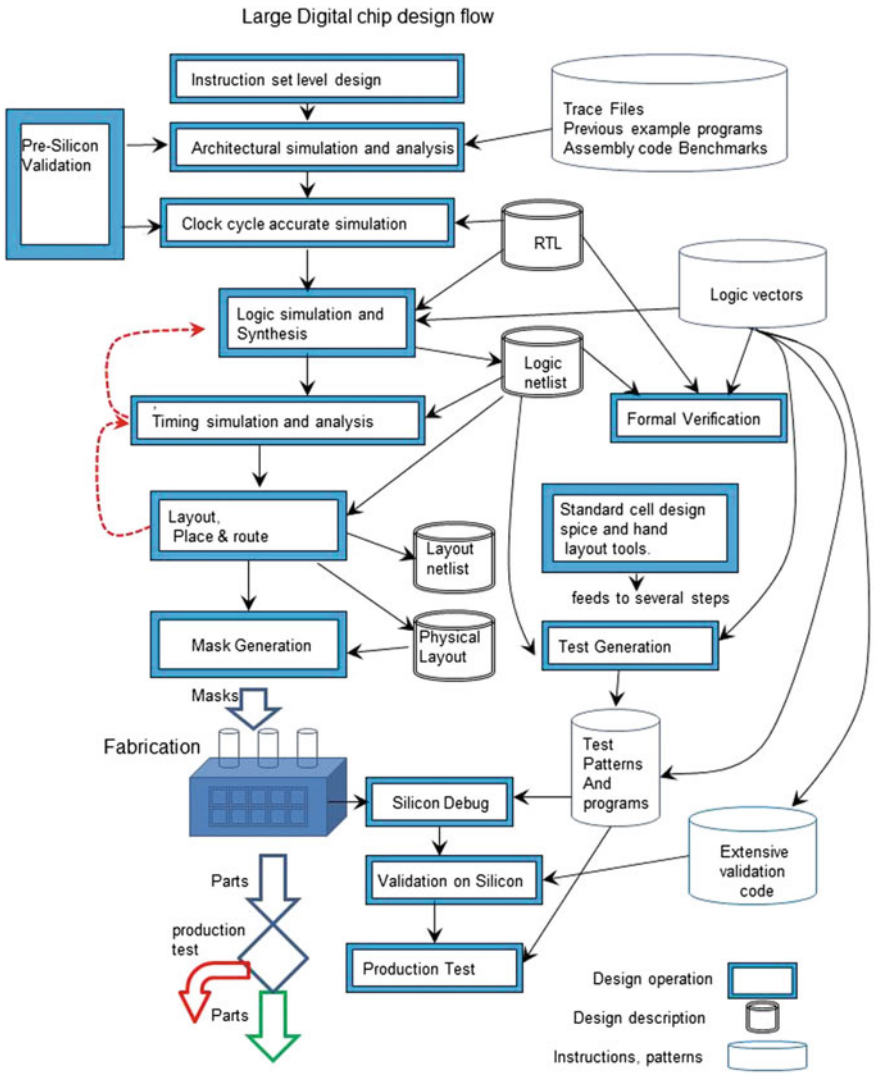


Fig. 9.1 An example chip design flow with several EDA tools

applications on the existing architectures and also by simulating the new architecture with proposed instructions. This task is performed during Architectural Simulation and Analysis explained below. The former usually involves scripting a number of runs on the existing machines, and automatically gathering the data in a presentable form. Depending on the amount of data being collected, it could involve a moderate number of disk accesses, and depending on the tasks being studied the overall categorization could be compute-heavy, memory-heavy or disk-heavy. The bulk of these tasks maps to the High-Performance Computing

(HPC) category. However, for mapping these to the Cloud, one must know the exact architecture and microarchitecture versions of the machines that run these scripts. In the current Cloud Computing model, this match-up might be typically difficult to achieve.

Architectural simulation and analysis is the exploration of microarchitectural design parameters and choices such as the size and organization of CPU Caches, or how many arithmetic logic units (ALUs) need to operate in parallel to achieve the desired performance, and involves setting area and timing goals for a chip. Design architects often work with an instruction set simulator, to see how sample applications will behave with the design configuration changes. Different simulators are used for exploring the design parameters at different levels. For example, full-system simulators can run unmodified operating systems and commercial applications while simulating the hardware structures in less detail, as compared to the component or sub-system simulators that simulate the hardware with a goal of understanding chip area and power consumption. Sub-system simulators usually take traces of events as inputs and do not have the ability to run high-level software directly. Full-system simulation involves the execution of a large, complex program running multiple independent simulations. What distinguishes this from the previous category (instruction set design) is that the simulators can run on machines with different underlying architectures. The capabilities or configurations of these machines do not affect the simulation results. It only affects the time it takes to finish the simulations. These tasks are thus well suited to run in the Cloud and map to the High-Performance Computing (HPC) category. Sub-system simulators, on the other hand, can be smaller in size and less complex and can often be run interactively. Thus, these would match the highly interactive single-person job category.

Clock cycle accurate simulation refers to the detailed functional and timing simulation of the proposed architectural design at the level of detail required to define the performance within each clock cycle. With both simulation and timing analyses, it checks and sets the boundary conditions for the time constraints on the lower-level implementations to perform these functions. This requires massive amounts of simulation to ensure coverage of the various architectural decisions. The appropriate workload category can be either Big Data calculation or High-Performance Computing. If extensive simulations are performed with the results being stored and then analysis is run, it belongs to Big Data calculation category. On the other hand, if the analysis calculations are run immediately after the simulations, thus skipping storage of large amounts OS simulation data, this falls into the High-Performance Computing category.

Logic simulation and synthesis refers to the step of detailed logic simulation of the system using the gate-level implementation including accurate timing models. The timing models include feedback from the later step of synthesis and place and route. This includes clock and power tree synthesis. This step requires extensive computer time for both the flattening and compilation of the circuit and the simulation of the circuit. This falls into the Big Data calculation category.

Timing simulation and analysis is the check for timing violations for the low-level implementation of the design. Depending upon how much of the circuit is analyzed, this has two possible categories. When all cases for the whole design are checked, this falls into the workload category of Big Data calculation. If a small sub-circuit of the design is being checked quickly, this is in the highly interactive single person category.

Pre-Silicon Validation refers to running software OS and associated test content on the model of a chip, before silicon is ready. The idea here is to find any bugs early and avoid the cost of a silicon re-spin. This can be done on an emulation model, derived from RTL (register-transfer level) description, or another HLM (high-level model) description or some combination thereof. This involves downloading the RTL model on a large FPGA (Field Programmable Gate Arrays)-based server boxes and running them, say at a speed of several hundreds of MHz. These server boxes are specialized and tend to be very expensive; hence, small design teams cannot often afford them. If these boxes are stored in a Private or Public Cloud, then these can be time-shared at an effective cost, thereby reducing the overall product design cycle time and cost. This step maps well to HPC in Cloud.

Formal Verification refers to the process of applying mathematical postulates and theorems to the high-level model or its representative code for ensuring that critical logic bugs are detected early, without having to run an exhaustive set of simulation tests. Current generation of chips has many states giving rise to complex combinations of interactions, some of which may result in a deadlock that can be only detected after a long series of events. Such a situation cannot be always anticipated or tested with all available test vectors, but may get exposed in the field. Thus, formal verification adds yet another capability to the arsenal of validation team by mathematically ensuring that the flow of events and interactions will be error free. However, it is also computationally challenged and methods for this are still evolving. Formal verification usually works at an IP level, so it maps well to the Single-computer intensive jobs category.

Standard cell designs are done on local machines using SPICE [3] and hand layout tools. These tasks require real-time editing of a layout figure by mask designers, or what-if simulations of circuit timing with interactive device size changes. These steps are performed on engineering workstations, or local terminals with graphics displays directly connected to nearby servers. If these servers are moved to a Public Cloud, it is likely to introduce higher latencies due to network delays, which may be intolerable to the human designers. This design task maps well to the highly interactive single-person category. This is not to imply that a single person can design a large cell library. Large complex standard cell libraries require large teams and a great amount of compute power. This translates to multiple highly interactive single-person tasks computer workload category.

Layout place and route refers to the process of planning, placing, and connecting design objects. Planning is a top-down process, whereas the placement and routing are done bottoms up, with only the rectangular shape of a lower-level block and its pins visible at the next level up. Transistors are at the lowest level of hierarchy in a standard cell and then cells in higher-level block such as an ALU.

Then, these higher-level blocks are placed and routed in yet another abstract-level function such as to implement a CPU or ISP (Image Signal Processing) unit on the silicon. Such a hierarchical arrangement limits the complexity and number of blocks to be placed and routed to a manageable size in a chip having billions of transistors. Planning is an interactive process, which is mapped to highly interactive single person, whereas place and route tools are batch mode, and can be mapped to the category of Single-computer intensive jobs. Some physical synthesis jobs can take a week or more to complete. A number of such jobs are launched in parallel to run on a server farm, and map well to HPC in Cloud.

Mask Generation requires some additional processing. With smaller geometries on current silicon designs, there is a need to do some adjustment to account for these dimensions being even smaller than the wavelength of light used for lithography process during chip manufacturing. The geometric steps are collectively called OPC (Optical Proximity Correction) and are very computationally intensive in nature. Hence, divide-and-conquer strategy is used to split the final layout in little pieces, which are then sent to a large number of servers for parallel processing, and then integrated back into a final layout database. Hence, Mask Generation qualifies as an HPC category with Big Database Creation and Calculations.

Silicon Debug is performed after a chip is manufactured. This chip is then tested as its power, performance, and even some functional bugs can only be found with actual silicon instead of with EDA models. Also, the chip after a successful power-on can run user applications at full speed, versus simulation models often running at greatly reduced speeds. Thus, the whole HW and SW stack can be tested on an actual chip, exposing some flaws that may require further silicon spins. If better models can be created, then there is a potential for Cloud Computing to add value without waiting for the Silicon arrival. However, currently there is no substitute for the post-Silicon validation. This is best mapped to a Single Person Interactive task.

The tasks of Pre-Silicon Validation and Mask Generation clearly fall in the realm of Cloud Computing, if the IP security can be assured. Clearly, security is a boundary condition before any design IP can migrate to a Public Cloud. EDA tasks tend to have elastic computing needs, thus pay as you go scheme works well. Sometimes bringing in more computing to EDA tasks will help to finish them sooner. This often results in faster run times and helps to reap economic benefits of Cloud Computing.

Production Test is the application of a fixed test program to each chip manufactured, in order to identify defective chips and discard them. The creation of the test program requires significant computation for fault simulation and test vector generation. The fault simulation and test generation can be split into several high computation parallel jobs. This falls in the category of Big Data calculation.

Big Data is a relative term. Large databases in the EDA industry are small compared to some in other industries. Of the three categories of Big Data (Big Data Storage, Big Data Transfer, and Big Data calculation) EDA projects are most

dependent upon Big Data transfer. Specific EDA challenges involve getting large design database from the customer into the Cloud and then getting the modified Big Data back to the customer.

Some workload categories are best performed on a local computing resource such as a desktop workstation. Other workload categories benefit from the availability of large amounts of computing capacity, whether calculation or storage. Because different EDA tools utilized at different steps of a design flow map to different workload categories, the decision on what resources should be allocated is different during the different steps of design process. Therefore, a decision to utilize Cloud Computing for the EDA design flow is more appropriately made at the individual steps rather than for the whole process. An interesting example is data mirroring for a multi-site design team. For some steps in the design flow, data mirroring maps to the big data storage category. For other steps in the design flow, data mirroring maps to the highly interactive multi-person jobs. The allocation of resources is exactly what Cloud Computing solves, and is needed for a multi-category process such as the EDA design flow. In summary, a one size fits all answer to the EDA design tools migration to the Cloud is replaced by a tool-by-tool, or step-by-step decision based on the needs and requirements. The mapping of the workload categories provides a basis for the broad adoption issues considerations described in the next sections.

9.5 Considerations for Cloud Computing Adoption

Beyond the workload category mapping as described in Sect. 4.6, there are some broad considerations that the EDA industry must address to enable a migration to the Cloud. The considerations for EDA moving to the Cloud include: tool licensing, information security, Cloud provider desire, lessons from past failed attempts to use the Cloud, tool delivery mechanism, and customer demand. HPC capabilities for meeting performance computing demand of EDA are a must.

First, consider the licensing of EDA tools, which often prohibits sharing a tool between different geographical sites of the same company. An EDA tool at run-time must renew its token from a central server at regular time intervals. A Cloud by definition has distributed computing, so a centralized token implementation is a hindrance to sharing the same EDA tool across different servers or data-centers within a logical Cloud. It is common for a design flow to include EDA tools from multiple EDA ISVs. Fixing the licensing scheme for Cloud is beyond individual EDA vendor's effort and will need the EDA industry to work together. The current EDA licensing model can be in harmony with the Cloud Computing paradigm of pay as you use when the EDA industry players address the appropriate licensing mechanism [28].

A second consideration is that information security in the Cloud must be solved to the satisfaction of all stakeholders in the EDA industry [29]. Information security can be viewed as including three functions: access control; secure communications, and protection of private data. Access control includes both the initial entrance by a

participant and the re-entry of that participant or the access of additional participants. The secure communication includes any transfer of information among any of the participants. The protection of private data includes storage devices, processing units, and even Cache memory. These and more information security issues especially related to the Cloud are described in [20]. An example security problem is denial-of-service attacks. Ficco and Rak [30] point out the need for Cloud providers to monitor, detect, and provide host-based countermeasures. From the user viewpoint, security is an important part of the problem. For example, Ghosh et al. propose a framework to allow a customer to evaluate Cloud providers using several factors, including different security issues [27]. The EDA industry does not in and of itself pose new information security issues [31]. However, the Cloud suppliers and the EDA vendors must work together with the EDA user community to find a solution that meets the needs of all stakeholders.

A third consideration for EDA vendors is whether the Cloud suppliers even want EDA tools in the Cloud [32]. Both Amazon and Rackspace have publicly stated a desire to attract HPC workloads in their data-center, to showcase their prowess and increase revenue. The list of Cloud Service Providers (CSPs) that may be potentially interested in EDA is fairly long [33] and spread across the world. Their services range from SaaS (Software as a Service), PaaS (Platform as a Service), and IaaS (Infrastructure as a Service), including hosting customers' servers in CSP-owned data-centers. Of this list, the most relevant is IaaS because a typical EDA company and its users will come with their own software and data and need physical or virtual servers to host them. Figure 9.2 shows how extra compute resources can reduce time-to-market.

A fourth consideration is that moving to Cloud has been attempted in recent years. Several large EDA companies have already experimented with offering Cloud Services using their own data-centers, with mixed responses from customers. Synopsys is an example with a verification-on-demand solution using VCS, which provides massively scalable verification via Amazon Web Services. According to Synopsys Web site, VCS [34] is the first EDA product to become available on a Public Cloud. While this does not demonstrate that EDA in the Cloud is a success or failure, it does show that the work has started.

A fifth consideration is how the EDA tools are offered to the end users. As it often happens in the software industry, some companies are aggregators of others' tools. This model may work well for EDA in the Cloud. Transim Technology Corporation hosts several Cloud-based engineering applications [35]. Productrinica is the company that launched the Cloud Testing Service (CTS) aimed at design verification engineers, fabless company, and any institutions with limited resources [36]. The service is a pay-per-use approach applied to IC prototype testing, debugging, and troubleshooting.

A sixth consideration is the EDA customers' demands. Cadence and a start-up Nimbic (purchased by Mentor Graphics) found limited customer demand for moving to the Cloud [37]. However, Synopsys has recognized the demand for EDA tools in the Cloud [34]. A model which may satisfy EDA customers' demand is a hybrid one, with some of the EDA tools being in the Cloud and others in the captive

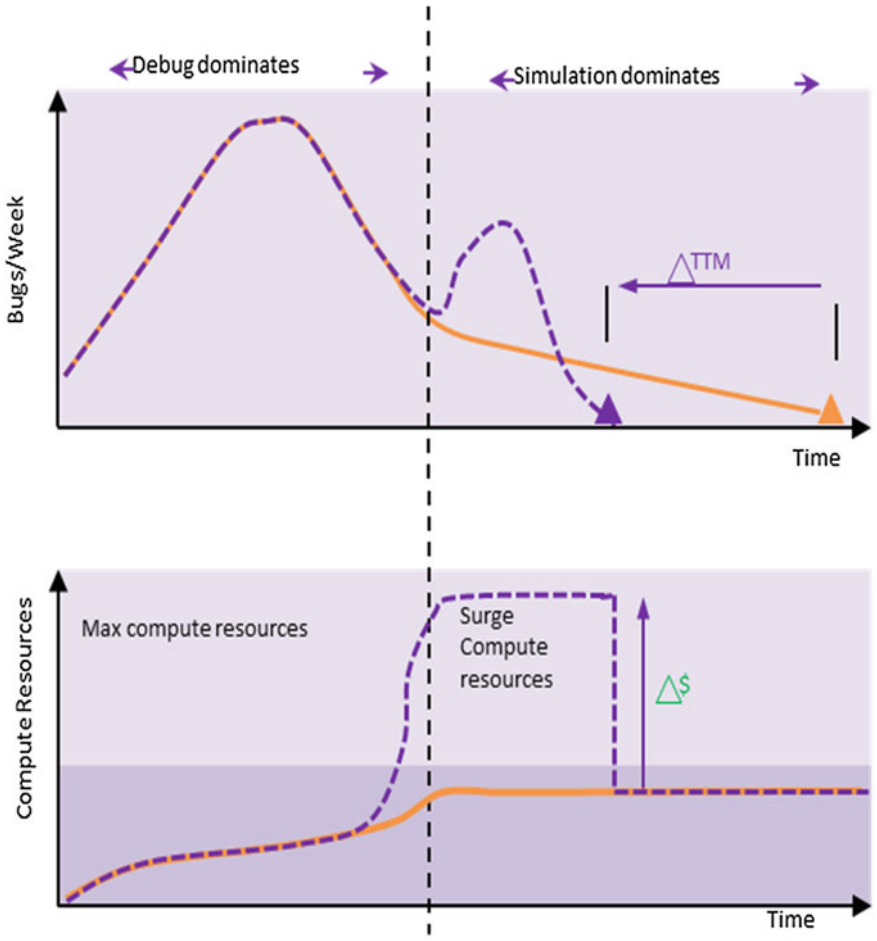


Fig. 9.2 Design teams can reduce time-to-market with "surge" computing resources [34]

data-centers. As Fig. 9.3 shows, 60% of participants in a recent DAC (Design Automation Conference) survey predicted that EDA licensing will move to the Cloud model in near future.

A seventh consideration is that the current EDA tools and flows were not designed with Cloud Computing in mind. Therefore, to fully utilized Cloud Computing, these may need to be redesigned. Potential benefits of redesigning the EDA tools for Cloud may include an order of magnitude improvement. This clearly requires further study and experiments on adopting EDA algorithms for Cloud Computing [38].

This chapter has concentrated on the case study of the EDA industry adoption of Cloud Computing. An excellent analysis of other issues for adoption of Cloud Computing by Small- and Medium-sized Enterprises (SME) described that the

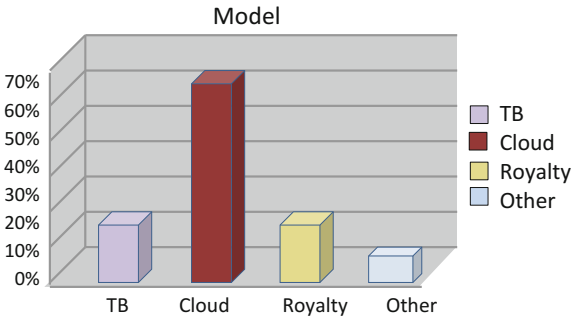


Fig. 9.3 EDA tool licensing model TB—time based, Cloud, royalty (on IC production), or other (mostly combination) [28]

larger barriers to adoption are non-technical [39]. First and foremost, it is worth noting that they found that most companies see the benefits of adopting Cloud Computing. The top seven reasons that companies gave for not adopting nor intending to adopt Cloud Computing solutions were as follows: security (including data loss), loss of control of data and infrastructure to manage data, lack of ability to measure cost–benefit analysis, availability and quality of service, data lock-in (inability to change providers), and data privacy and associated legal requirements. Their chapter emphasizes that in some cases, such as security, adoption of Cloud Computing has actually improved the situation. This is particularly important for security because it is listed as a top concern in the EDA industry. Cloud Computing adoption decisions should be made on a case-by-case basis for different categories of computing. We illustrated this with a case study of the EDA industry.

We have described broad considerations that must be addressed in order to take advantage of opportunities that Cloud Computing provides to the EDA industry. These considerations include the following: licensing, security, Cloud providers, past EDA attempts to move to the Cloud, delivery mechanism, and customer demand. The approach to meeting these considerations may vary with different steps (hence workload categories) in the design process or with different vendors, customers, and Cloud providers.

9.6 Summary

The EDA industry software, tool methodologies, flows, and scripts for VLSI design have evolved from individual programs on mainframe computers, through collections of tools on engineering workstations, to complete suites of tools with associated methodologies on networks of computers. Electronic design automation has had a significant impact on circuit and hardware design, which in turn has contributed to the vast progress in the field of computing. Design automation is one of the reasons why computer chips with upward of billions of transistors can be

designed. One can assert that the server farms that form the back end of the Cloud would not have been around without the EDA industry. Thus, it is interesting to see whether Cloud Computing can, in turn, facilitate future growth of the EDA industry.

While Cloud Computing is often thought of as a monolithic entity, it comes in many flavors and is comprised of several subsystems and services. Similarly, EDA tools are sometimes thought of as part of a system built with each tool having similar attributes. Based upon a previous categorization of Cloud Computing workloads, this chapter maps the sub-tasks of an example silicon design flow to the types of workloads. The mapping of workloads is applicable to both Private Cloud Computing and Public Cloud Computing. This mapping can serve as an example for EDA companies and hardware design firms as they look to explore the Cloud for hardware design tasks. Our method can potentially open new doors and customer bases for enabling EDA growth. This chapter also provides examples of some early adopters, the issues they faced, and new emerging challenges, whether real or perceived [40, 41]. Additionally, some considerations were mentioned, such as licensing and delivery mechanisms that go beyond the mapping of tasks to workloads. The major contribution of this chapter is a proposed method for mapping EDA tools to Cloud Computing categories to facilitate the decision of which EDA tools are candidates for moving to the Cloud. Such a step is needed to migrate any established industry to adopt Cloud Computing.

9.7 Points to Ponder

1. **For mission-critical application, such as EDA or health data, what precautions need to be taken? EDA companies want to recoup R&D investments faster by selling private licenses, while their customers are hesitant to put confidential design data in a Public Cloud. Given this situation, would it ever make sense to migrate EDA workloads to Public Clouds?**
2. **How do IP concerns influence the decision to move to Clouds? What precautionary measures need to be considered?**
3. **Are there any other enterprise or Private Cloud customers who faced similar dilemmas, but economics enabled their migration to Public Clouds?**
4. **What can Public Cloud Service Providers do to accelerate migration of EDA workloads?**
5. **What benefits can an EDA solution provider get by migrating to a Public Cloud?**
6. **What benefits can EDA customers get by migrating to a Public Cloud?**

References

1. NIST Special Publication 800-145
2. Ousterhout JK, Hamachi GT, Mayo RN, Scott WS, Taylor GS (1984) Magic: A VLSI layout system. In: 21st Design automation conference, pp 152–159
3. Nagel LW (1975) SPICE2: a computer program to simulate semiconductor circuits, ERL-M520. Electronics Research Laboratory, University of California, Berkeley
4. Beebe S, Rotella F, Sahul Z, Yergeau D, McKenna G, So L, Yu Z, Wu KC, Kan E, McVittie J, Dutton RW (1994) Next Generation Stanford TCAD—PISCES 2ET and SUPREM OO7. In: IEDM 1994 proceedings, December 1994, San Francisco, CA, pp 213–216
5. Pinto MR, Rafferty CS, Dutton RW (1984) PISCES-II—Poisson and continuity equation solver. Stanford Electronics Laboratory Technical Report, Stanford University
6. Acken JM, Stauffer JD (1979) Part 1: logic circuit simulation. IEEE Circuits and Systems Magazine, Mar 1979
7. Acken JM, Stauffer JD (1979) Part 2: logic Circuit Simulation. IEEE Circuits and Systems Magazine, June 1979
8. Szygenda SA (1972) TEGAS-anatomy of a general purpose test generation and simulation at the gate and functional level. In: Proceedings of 9th design automation Conference, pp 116–127
9. Bryant RE (1981) MOSSIM: a switch level simulator for MOS LSI. In: Proceedings of 18th design automation conference, pp 354–361
10. Goldstein LH (1979) Controllability/observability analysis of digital circuits. IEEE Trans Circ Syst 26(2)
11. Grason J (1979) TMEAS, a testability measurement program. In: Proceedings of 16th design automation conference, pp 156–161
12. Hachtel GD, Hemanchandra L, Newton R, Sangiovanni-Vincentelli A (1982) A comparison of logic minimization strategies using ESPRESSO: an APL program package for partitioned logic minimization. In Proceedings of the international symposium on circuits and systems, Rome, Italy, pp 42–48, Apr 1982
13. Preas B, Gwyn CW (1978) Methods for hierarchical automatic layout of custom LSI circuit masks. In: DAC 1978, pp 206–212
14. Sechen C, Sangiovanni A (1986) Timberwolf 3.2: a new standard cell placement and global routing package. In: 23rd DAC, pp 432–439
15. http://esd-alliance.org/wp-content/uploads/PDFs/2017/MSS_Q2_2017_Newsletter.pdf
16. Jamshidi P, Ahmad A, Pahl C (2013) Cloud migration research: a systematic review. IEEE Trans Cloud Comput 1(02):142–157
17. Guan H, Ma R, Li J (2014) Workload-aware credit scheduler for improving network I/O performance in virtualization environment. IEEE Trans Cloud Comput 2(2)
18. Moreno IS, Garraghan P, Townend P, Xu J (2014) Analysis, modeling and simulation of workload patterns in a large-scale utility cloud. IEEE Trans Cloud Comput 2(2)
19. Mulia WD, Sehgal N, Sohoni S, Acken JM, Lucas Stanberry C, Fritz DJ (2013) Cloud workload characterization. IETE Tech Rev 30(5)
20. Sehgal NK, Sohoni S, Xiong Y, Fritz D, Mulia W, Acken JM (2011) A cross section of the issues and research activities related to both information security and cloud computing. IETE Tech Rev 28(4):279–291
21. http://www.eetimes.com/document.asp?doc_id=1326801
22. Joyner W (2013) EDA: the first 25 years. Presented at the 50th design automation conference. Available: www.dac.com
23. Sehgal N, Chen CYR, Acken JM (1994) An object-oriented cell library manager. In: Proceedings of ICCAD, pp 750–753
24. Porter ME (2008) The five competitive forces that shape strategy. Harvard Business Review. http://hbr.org/2008/01/the-five-competitive-forces-that-shape-strategy

25. Kuehlmann A, Camposano R, Colgan J, Chilton J, George S, Griffith R, Leventis P, Singh D (2010) Does IC design have a future in the clouds? In: Design automation conference (DAC), 2010 47th ACM/IEEE, pp 412–414
26. Rutenbar R (2013) EDA: the Second 25 years. Presented at the 50th design automation conference. www.dac.com
27. Ghosh N, Ghosh SK, Das SK (2015) SelCSP: a framework to facilitate selection of cloud service providers. IEEE Trans Cloud Comput 3(1)
28. Ralph J. Challenges in cloud computing for EDA. Chip Design Magazine, http://chipdesignmag.com/display.php?articleId=4492
29. Stok L (213) EDA: the next 25 years. Presented at the 50th design automation conference. Available [Online]: www.dac.com
30. Ficco M, Rak M (2015) Stealthy denial of service strategy in cloud computing. IEEE Trans Cloud Comput 3(1)
31. Brayton R, Cong J (2010) NSF workshop on EDA: past, present and future. IEEE Des Test Comput, pp. 68–74
32. EDAC Forecast Meeting, San Jose, 14 Mar 2013. http://www.edac.org/events/2013-EDA-Consortium-Annual-CEO-Forecastand-Industry-Vision/video
33. http://talkinCloud.com/tc100
34. http://www.synopsys.com/Company/Publications/SynopsysInsight/Pages/Art6-Clouds-IssQ2-11.aspx
35. Designing in the cloud, Electronic specifier, 7 Nov 2012. http://www.electronicspecifier.com/design-automation/designing-in-the-Cloud-arrow-dr-uwe-knorr-transim-technology-es-design-magazine, June 2015
36. Elliott M (2013) Productonica: cloud clears way to IC testing. Electronic specifier, 13 Nov 2013. http://www.electronicspecifier.com/test-and-measurement/advantestcx1000d-productronica-Cloud-clears-way-to-ic-testing, June 2015
37. http://finance.yahoo.com/news/mentor-graphics-acquires-nimbic-inc-175400528.html
38. Schwaderer C (2014) EDA and the cloud. Embedded computing design, May 2014. http://embedded-computing.com/articles/eda-the-Cloud/
39. Trigueros-Preciado S, Perez-Gonzalez D, Solana-Gonzalez P (2013) Cloud computing in industrial SMEs: identification of the barriers to its adoption and effects of its application. Electr Markets 23(2):105–114
40. http://www.techradar.com/us/news/Internet/Cloud-services/security-is-a-red-herring-in-the-Cloud-1163167
41. B. Schneier, “The Hidden Battles to Collect Your Data and Control Your World,” March 2015, https://www.schneier.com/book-dg.html

Chapter 10
Costing and Billing Practices in Cloud

10.1 Cloud as a Service (CaaS): The Billing Imperatives

The primary mode of using Cloud Computing is through a service provider. Wherein, the two primary stakeholders here are the consumers of the service and provider of the service. However, unlike utilities such as electricity, water, postal or city services, the Cloud Computing services offer opportunities for many interesting and innovative provisions, which we shall discuss later in this chapter. In this section, we compare and contrast some facets of billing in traditional utility services versus Cloud Computing.

10.1.1 Billing and Best Practices

Consumer services are often competitive and market driven. Therefore, a service provider must operate in the customer first mode of operations to survive and succeed. In the traditional terms, a utility should be easy to access and provide customer delight. In Cloud Computing context, this translates to:

1. Easy to use interface;
2. Attract customers to repeat experience with greater vigor.

The second item above is important to make it attractive for higher volumes and greater capacity utilization of Cloud resources. Note this is not different from traditional utility services, which often offer cost-saving options for volume and frequency of business.

A Forrester study [1] observed that patron patience lasts no more than 8 s with not-so-friendly Web interface. In traditional utility services, there is not much transactional traffic. So, what this translates to in Cloud Computing context is that the service provider must provide a straightforward customer interface for

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6_10

multimodal transaction traffic. For example, multimodal transactional traffic should not result in password fatigue or delays with repeated exchanges for access requests. A single sign-on for a session and strong authentication is desirable.

A good traditional utility practice is to provide notifications to patrons. This is very easy in Cloud Computing context. It also is an opportunity for prompt cross-selling options. Depending upon the nature of usage and patterns of operations, the Cloud Computing service can offer most beneficial pricing. Later, in the chapter we will describe how Amazon adapts its billing to the services by employing auto-scaling and other practices.

An important consideration that goes with utility services companies is observance of some standards. Cloud Computing billing practices do have service-level agreements, as described in a later section of this chapter, but there are no known common standards as of now. In particular, there are undefined areas on customer assistance, default or minimal obligations and options, emergency or breakdown coverage, termination of services, transfer of accounts following mergers and breakups. This is still an open area and evolving.

10.2 Pay as You Go

The entire Cloud Computing usage model is based on a pay as you go, or pay for what you consume, concept. This enables Cloud customers to not buy IT hardware or software products, but rent them on a time and/or volume basis, e.g., how many CPU cores are needed for how long, with how much memory and persistence storage space, etc. The persistence storage also comes in various forms, such as disks directly attached to the computing instance for faster access, but at a higher cost, or stored across the network in a long-term storage repository at a cheaper cost, higher latency, etc.

At the end of a month, or agreed upon usage period, a Cloud Service Provider (CSP) will send the bill to its users. If up-front menu choices are not made wisely, then the user may be in for a surprise, but little can be done retroactively to reduce costs.

To create innovative and interesting billing propositions, major Cloud providers such as Amazon EC2 offer additional choices such as On-Demand versus Spot Instances, ability to do auto-scaling, Elastic Load Balancers, etc. Following sections will explain each of these concepts in detail. Although these details are pertinent to Amazon EC2, other major CSPs offer similar capabilities with different names. Let us consider a Cloud customer who is hosting an end-user facing Web site. This customer may experience variable user load during day and night hours. Rationale for offering a choice to customers is to ensure that their needs can be met and fill up the installed Cloud capacity.

10.3 Amazon EC2 Motivations and Setup

Amazon offers their computing capacity in availability zones (AZs), which are distinct locations within a Region that are engineered to be isolated from failures in other availability zones. As explained in Chap. 6, an availability zone is a distinct location within a Region that is insulated from failures in other availability zones. It also provides inexpensive, low-latency network connectivity to other availability zones in the same Region. A simple example can be a different Data-Center (DC), or a separate floor in the same DC, or an electrically isolated area on the same floor. Whereas a Region is a named set of AWS resources located in a particular geographical area, such as Northern California. Each Region must have at least two availability zones.

Motivation for multiple Regions is to provide worldwide customers with a server location closer to their usage points, so that signal transmission latencies are reduced. Otherwise, each click from a customer in India will potentially need to travel to a data-center in USA, and then sending the response back will require several hundreds of milliseconds, slowing down every web transaction. Furthermore, different customers have different usage patterns and financial abilities to pay for Cloud Services. Hence, someone who is an occasional user of Cloud may want to pay by the hour with no long-term commitments or up-front fees. Amazon's On-Demand Instances is the best choice for such a consumer. Another customer with a large fluctuating demand may want to use Amazon's Spot Instance bidding. An example of such a user is an accounting or tax preparer site. Yet another customer with a long-term horizon and predictable demand such as an engineering design firm with multiple contracts would be interested in Amazon's Reserved Instances to save money and get guaranteed capacity. Last but not the least is the category of customers who must operate on isolated and dedicated servers to ensure maximal security and performance. This includes healthcare providers, financial sector, certain government, police services, etc. Amazon's dedicated hosts and instances are the best way to go for such customers.

There are no minimum fees to use Amazon EC2 [2]. Currently, there are four ways to pay for what you consume in their Cloud Computing based on the above usage patterns:

(1) **On-Demand Instances,**
(2) **Spot Instances,**
(3) **Reserved Instances,**
(4) **Dedicated hosts and dedicated instances.**

Following section will present a detailed description of how each of these work and their usage patterns. However, before that let us take a look at Amazon's Partner Network (APN), which supports AWS customers with technical, business, sales, and marketing resources. We will only review the first category as relevant to the Cloud Computing context. APN Technology Partners provide software solutions that are either hosted on or integrated with the AWS Platform. Technology

Partners include Independent Software Vendors (ISVs), SaaS, PaaS, developer tools, management and security vendors. An example is AWS Proof of Concept (POC) Program, designed to facilitate AWS customer projects executed via eligible APN Partners. To accelerate a customer's AWS adoption, PoCs provide funding with free AWS usage or co-funded Professional Services. Goal is to accelerate customer onboarding of enterprise workloads to AWS and to develop customer references and/or case studies that APN Partners can use to demonstrate their AWS expertise.

10.3.1 Amazon's On-Demand Instances

With On-Demand instances, customers pay for computing capacity by the hour with no long-term commitments or up-front payments. They can increase or decrease the computing capacity depending on the demands of their applications and only pay the specified hourly rate for the instances used. On-Demand instances are recommended for:

- Users that prefer the low cost and flexibility of Amazon EC2 without any up-front payment or long-term commitment;
- Applications with short-term, spiky, or unpredictable workloads that cannot be interrupted;
- Applications being developed or tested on Amazon EC2 for the first time.

There are no guarantees that users will always be able to launch specific instance types in a timely manner, though AWS best case efforts to meet their needs.

10.3.2 Amazon Spot Instances

Amazon EC2 Spot Instances allow customers to bid on spare Amazon EC2 computing capacity for up to 90% off the On-Demand price. Spot Instances are recommended for:

- Applications that have flexible start and end times;
- Applications that are only feasible at very low computing prices;
- Users with urgent computing needs for large amounts of additional capacity.

Spot price fluctuates based on the supply and demand of the available computing capacity. It is the leftover capacity of AWS to be used on as available basis. There is no difference in the performance compared to On-Demand instances and is usually cheaper than the On-Demand instances as there is no guarantee of the availability. The user can choose a start time and end time for the instances or can make a persistent request (no end time specified) for this service. This service is preferable

for computing needs that are not tied to any deadlines, whose computing needs are large and the interruption of service is acceptable.

Users can bid for Spot Instances. Once your bid exceeds the current spot price (which fluctuates in real time based on the demand-and-supply), the instance is launched. The instance can go away anytime the spot price becomes greater than your bid price. You can decide your optimal bid price based on the spot price history of last 90 days available on AWS console or through the EC2 APIs.

10.3.3 Amazon Reserved Instances

Reserved Instances provide customers with a significant discount (up to 75%) compared to the On-Demand instance pricing. In addition, when Reserved Instances are assigned to a specific availability zone, they provide a capacity reservation, giving customers an ability to launch instances whenever needed.

For applications that have steady-state or predictable usage, Reserved Instances can provide significant savings compared to using On-Demand instances. Reserved Instances are recommended for:

- Applications with steady-state usage;
- Applications that may require reserved capacity;
- Customers that can commit to using EC2 over a 1- or 3-year term to reduce their total computing costs.

AWS introduced a single type of Reserved Instance with three payment options: one up-front payment for the entire Reserved Instance term (one or three years), pay for a portion of the Reserved Instance up-front with installments for the rest over the course of the one- or three-year term, or nothing up-front except the commitment to pay over the course of a year.

10.3.4 Amazon Dedicated Instances and Dedicated Hosts

A dedicated instance is a virtual machine assigned to a customer and hosted on a specific host. A dedicated host is a physical EC2 server dedicated for that customer's use. No other VM will share resources with it, and resources like CPU/memory will not be shared with anyone else. Dedicated hosts can help to reduce costs by allowing customers to use your existing server-bound software licenses, including Windows Server, SQL Server, and SUSE Linux Enterprise Server (subject to the licensing terms), and can help to meet compliance requirements.

- Can be purchased On-Demand (hourly);
- Can be purchased as a reservation for up to 70% off the On-Demand price;
- You can provision one or more EC2 instances on dedicated hosts.

10.4 Motivation and Methods for Right Sizing Customer VMs

Since each user application has a certain need of CPU, memory, and storage resources, it is important to provision these in advance to avoid adverse run-time performance effects. However, giving an application more resources may not necessarily make it run faster, but will certainly cost more in a Cloud. To start with, every time a new VM is launched in the Cloud, and it has a new IP address. This may not be desirable for a public facing Web site as it needs a Domain Name Server (DNS) entry mapping its name to a fixed IP address. This problem is addressed by Amazon's use of **Elastic IP**. Such a fixed IP can map to one or more servers with an **Elastic Load Balancer** (ELB) in between. It makes sure that the users' traffic is evenly distributed. Having two or more servers running in parallel also ensures that the Web site will be resilient, even when a server suffers an outage or slowdown. Another challenge is that usage of a public facing Web site may vary overtime. An example is a digital newspaper hosting site, which will have more users in the morning and evening, given that its readers may be working at their regular jobs during daytime, and probably a very few during late nights as most readers would be sleeping. In order to accommodate a variable workload, the newspaper site want to duplicate its content on multiple servers and bring them online as the users grow, such that each will get an acceptable respond time. Any Web site with a slow response during heavy traffic time stands to lose its customers and suffer a business loss. This problem can be easily solved with **auto-scaling**, which monitors the CPU usage of existing servers. When the utilization grows, it will start new servers to evenly distribute the load.

Thus, it is important to understand an application's requirement and plan for it in advance. In this context, following AWS concepts are introduced.

10.4.1 Elastic IP

When an instance is launched, AWS assigns it a public IP address. Every time the instance is stopped, this IP address is freed up. However, an application's users may expect a permanent IP address for their future sessions. AWS' solution is the use of Elastic IPs (EIPs).

An Elastic IP address (EIP) is a static IP address designed for dynamic Cloud Computing. With an EIP [3], one can mask the failure of an instance or software by rapidly remapping the IP address to another instance. An EIP is associated with customer's AWS account, not a particular instance, and it remains associated with the account until the customer chooses to release it.

10.4.2 Elastic Load Balancing

Elastic Load Balancing [4] automatically distributes incoming application traffic across multiple Amazon EC2 instances. It enables achieving fault tolerance in user applications, seamlessly providing required amount of load balancing capacity needed to route application traffic.

Elastic Load Balancing (ELB) ensures that only healthy Amazon EC2 instances will receive users' traffic by detecting unhealthy instances. ELB reroutes traffic to the remaining healthy instances, as shown in Fig. 10.1. If all of a customers' EC2 instance in one availability zone A are unhealthy, but additional EC2 instances were set up in multiple availability zones, E will route traffic to the healthy EC2 instances in those other zones, such as availability zone B.

Elastic Load Balancing offers two types of load balancers that feature high availability, automatic scaling, and robust security. These include the Classic Load Balancer that routes traffic based on either application or network level information, and an Application Load Balancer that routes traffic based on advanced application-level information, such as Web page load delays. The Classic Load Balancer is ideal for simple load balancing of traffic across multiple EC2 instances, while the Application Load Balancer is ideal for applications needing advanced routing capabilities, microservices, and container-based architectures. Application Load Balancer offers ability to route traffic to multiple services or load balance across multiple ports on the same EC2 instance. Amazon also offers EC2 Container Service (ECS) that supports Docker containers and allows running applications on a managed cluster of server instances. Amazon ECS eliminates the need to install, operate, and scale user's own cluster management infrastructure. With simple API calls, one can launch and stop Docker-enabled applications and query the complete

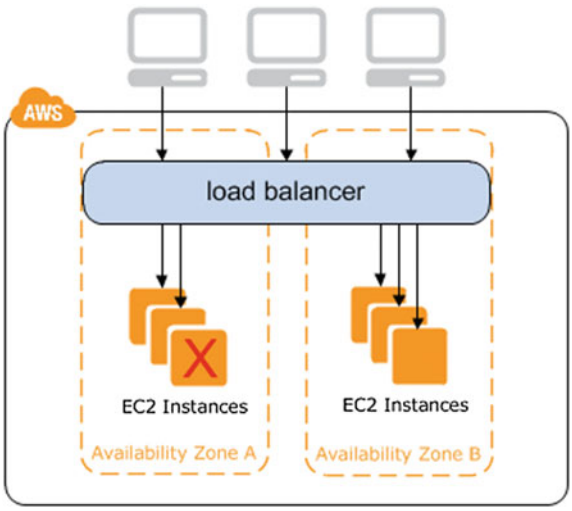


Fig. 10.1 An example of a server going bad in Zone A, and traffic getting routed to Zone B

state of a cluster. APIs also access features such as security groups, Elastic Load Balancing, EBS volumes, and IAM roles. Amazon enables scheduling the placement of containers across the clusters based on resource needs and availability. Users can integrate their own or third-party schedulers to meet business or application-specific needs.

10.4.3 Auto-Scaling

In this section, we will be using four concepts that are uniformly used in the IT community, and can be defined as follows:

(1) Scale up refers to increasing compute capacity of a server, by increasing either its density or performance, e.g., a CPU with more cores or more memory, resulting in higher throughput on a single threaded application. More memory can be of higher capacity or faster speed components.
(2) Scale out refers to adding multiple units of computing, e.g., more servers for a distributed application to share the load.
(3) Scale in is the process of reducing or compacting the computing capacity for efficient reasons, e.g., when load is light then moving virtual machines from many physical servers to run on a fewer servers. This results in reduced operating costs.
(4) Auto-scaling is the ability of a system to Scale In or Out depending on the workload.

Auto-scaling in EC2 [5] helps to maintain application availability, by scaling Amazon Server capacity to go up or down automatically, according to the conditions you define. You can use auto-scaling to help ensure that you are running a desired number of Amazon EC2 instances. Auto-scaling can also automatically increase the number of Amazon EC2 instances during demand spikes to maintain performance and decrease capacity during lull periods to reduce customer costs, as shown in Fig. 10.2.

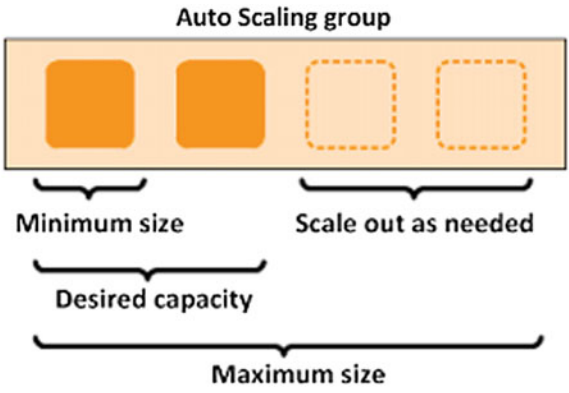


Fig. 10.2 An example of auto-scaling

When the load is light, only a few servers are needed. However, as more customers use a Web site or an application hosted in Cloud, their experience may deteriorate as existing server slows down. In this case, new servers are started and brought online to share the increased load. This can happen automatically, by checking the CPU utilization of current servers. If it increases beyond a threshold, then new servers are started. Similarly, when demand falls below a threshold, then additional servers are shut down to save money, and remaining customers are moved to fewer servers. It is called customer consolidation. Scale up or down often requires a stop and restart of application on a new server, unless a live migration of virtual machine is available. Scale In or Out is suitable for distributed applications that are typically multi-threaded and have many tasks that can benefit from additional machines at the run-time.

Auto-scaling is well suited to applications that have stable demand patterns or that experience hourly, daily, or weekly variability in usage. Auto-scaling enables following the demand curve for customer applications closely, reducing the need to manually provision Amazon EC2 capacity in advance. For example, one can set a condition to add new Amazon EC2 instances in increments to the auto-scaling group when the average utilization of Amazon EC2 is high; similarly, one can set a condition to remove instances in the same increments when CPU utilization is low.

10.5 Cost Minimization

If customer demand is fluctuating, and application is scalable, then instead of getting a large computer, it may be better to run multiple copies of an application. These can be hosted on identical EC2 instances, such that auto-scaling manages launch and termination of these EC2 instances. As shown in Fig. 10.3, one can use Amazon Cloud Watch (CW) to send alarms to trigger scaling activities and Elastic Load Balancing to help distribute traffic to customer instances within auto-scaling groups. Cloud Watch monitors CPU utilization of servers on which user applications are running. If and when the utilization goes above a certain threshold, say 85%, then an alarm is generated, which in turn can alert the application owner, and based on a predetermined scaling policy bring up new server instances to share the application load. When done correctly, CPU utilization of previous servers as well as latency experienced by new users will both go down. Thus, auto-scaling enables optimal utilization of Cloud resources.

If a customer has predictable load changes, then one can set a schedule through auto-scaling to plan scaling activities, as shown in the life cycle of Cloud application in Fig. 10.4. At the start, a new auto-scaling group and Scale out events are defined. Then, one or a few set of instances is launched, and whenever load increases new instances are added to the group. This can be done with a Cloud Watch trigger as defined previously, or based on a fixed schedule such as people waking up on Monday morning to check their emails, which will need more e-mail servers. Similarly, in Step 5, a **Scale-In** event is defined, which can also be based on

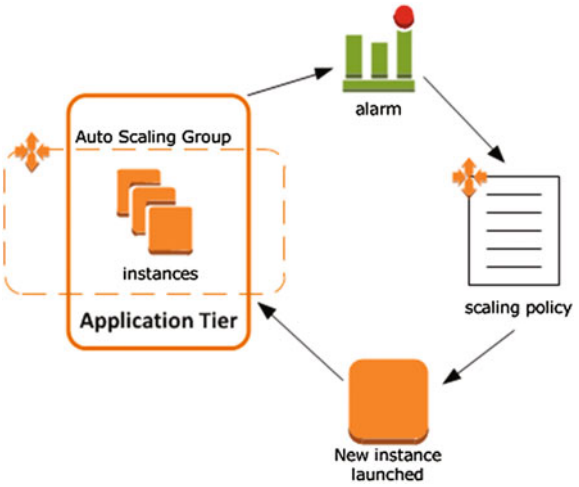


Fig. 10.3 Using Cloud utilization alarm to trigger auto-scaling

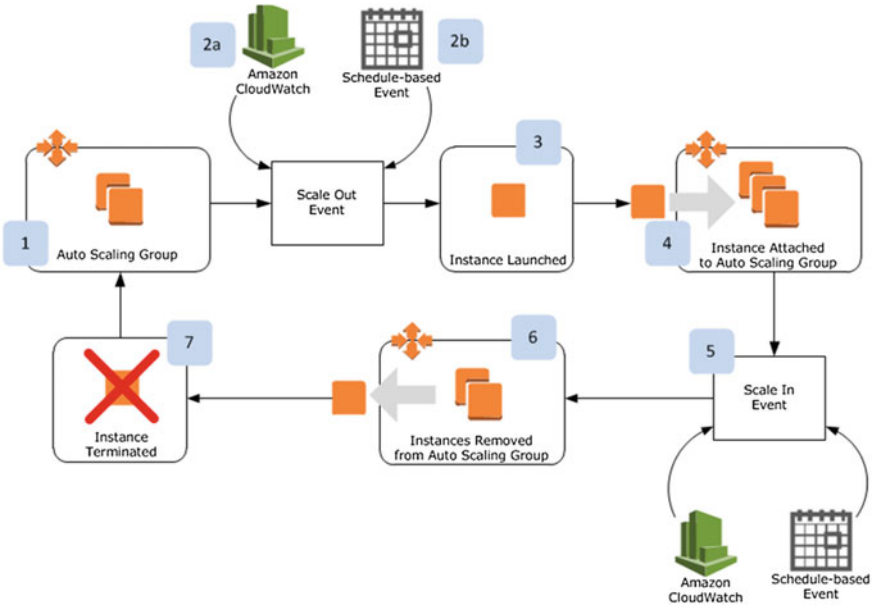


Fig. 10.4 Life cycle of a Cloud application

Cloud Watch monitoring CPU utilization going below a threshold, say less than 50%, or a scheduled event such as the start of a weekend when fewer people login and check their work emails.

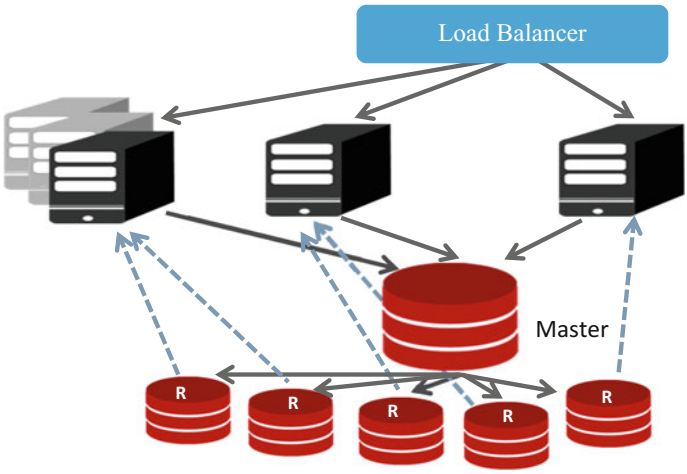


Fig. 10.5 A simple application's architecture

Any application needs both compute and storage capacity, as shown in Fig. 10.5, so Scale Out is done for both compute servers and storage disks. These disks either have identical copies of read-only data, such as an online newspaper read by many users or partitioned information such as different users looking at different segments of a large database, e.g., airlines flights or an online auction site, which can be independently partitioned.

Key is to understand an application's need of Cloud Computing resources, and then predict and provision them in advance, to avoid losing customers or paying for unused capacity, as shown in Fig. 10.6. The dotted inclined line shows predicted computing demand growing over time. Blue line shows a step function, as large capital expenditures occur to install more server racks, networking switches, storage devices, power supplies, air-conditioning, etc. Once the dotted line reaches the limit of blue step, then next step of capital expenditure must be incurred. However, actual compute demand as shown by the solid red line fluctuates over time. It should always stay within the installed capacity envelope as defined by the blue step function. Their difference, as shown by the green arrow, represents unused installed capacity, which is the opportunity cost as that money could have been used elsewhere in the business. In a case, where the red actual demand exceeds the blue envelope, then there is no computing capacity available to run the actual tasks, so it represents unhappy or lost customers. Of course, with Cloud Computing, the blue line can expand elastically up or down, to the following red curve, thus minimizing the green opportunity cost and no lost customers.

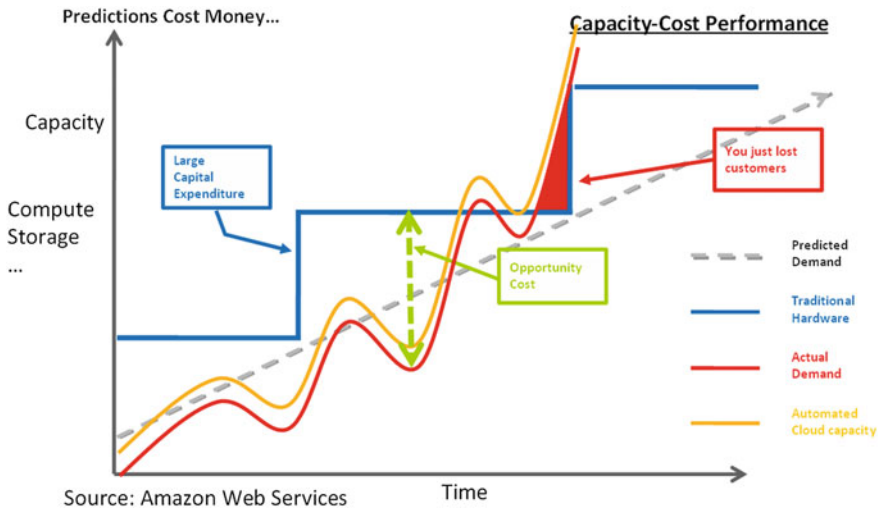


Fig. 10.6 Predicting Cloud Computing capacity needs

10.6 Capacity Forecasting

Every business organization wants to grow, but growth brings its own problems as quality of service may suffer causing slowdowns for the existing customers. In order to ensure that business operations run smoothly, IT department needs to understand the current bottlenecks, build future growth models, and predict users' demand in advance. This problem was exacerbated before Cloud, as it took weeks if not months to order, procure, and deploy new servers. In an elastic Cloud, new servers can be brought online in a matter of minutes.

An example is a movie-serving Web site, which could experience growth in a particular Region so the servers need to be located in the same or nearby Region. This will ensure minimal latency for the new and existing customers. Also, it needs to understand the bottleneck, as new customers may demand more of the existing movies or new films. Serving more customers with the same set of movies will need duplication of storage servers and network resources. Adding more movies to the repository will require new compute servers to encode the new movies, and then new storage devices to host them, while maintaining the older movies on the existing servers. Thus, it is important for a business to understand its users, their needs, and growth patterns. Furthermore, using spreadsheets, various scenarios need to be worked out on what to host and where, so the cost is understood and budgeted in advance. Following example will make this clear.

An application needs four EC2 instances and S3 storage, where the application running on EC2 requires 8 GB setup minimum. If the cost of each server instance is $0.10/h, and storage cost is $1.00/GB/month, what is the year budget for this organization?

$$\text{Server Instance cost/year} = 4 * \$0.10 * 24 * 365 = \$3504/\text{year}$$
$$\text{Storage cost/year} = 8 * \$1 * 12 = \$96/\text{year}$$
$$\text{Total outlay} = \text{Instance Cost} + \text{Storage Cost} = \$3600/\text{year}$$

Let us consider a business of online newspapers, magazines, or on-demand video services, where many users can access this read-only data through a Web application. If this business expects to grow by 50% per year due to enhanced readership, its storage cost shall remain constant due to *read-only* nature of operations. However, its server instances' cost/year will grow by 50% to support additional users, bringing the new total to $5352/year.

Now let us say, another set of 50% readers demand new magazines, which needs new computers and storage, by the same proportion, and requires 50% more server instances and storage. This will require new expenses to be $8082/year, which is more than 2× of the first year's expense. While more users is generally good news, but only if accompanied by a commensurate growth in the compute infrastructure. Else result will be similar to doubling the number of cars on existing roads, causing traffic jams slowing down all users.

10.7 Optimizations Across Clouds

Since multiple Cloud Service Providers (CSPs) exist, it makes sense to be able to compare their offerings on a common set of metrics and then choose what best meets a user's criteria. Furthermore, it may be desirable to have multiple accounts and make an application portable across CSPs, to avoid a lock-in. This is easier said than done, as each Cloud provider offers different set of tools and different flavors of solutions, with an aim to entice and lock in their customers. Currently, there are three large Public Cloud providers in USA.

Provider	Offering name	Description
Amazon	AWS	A comprehensive, evolving Cloud-Computing Platform provided by Amazon.com. Web Services are sometimes called Cloud Services or remote computing services. The first AWS offering was launched in 2006 to provide online services for Web sites and client-side applications
Microsoft	Azure	Microsoft Azure began as a Windows Server-based Cloud Platform, but has since grown to become a very flexible Cloud Platform that enables developers to use any language, framework, or tool to build, deploy, and manage applications. According to Microsoft Azure, features and services are exposed using open REST protocols. The Azure client libraries, which are available for multiple programming

(continued)

(continued)

Provider	Offering name	Description
		languages, are released under an open source license and hosted on GitHub
Google	Google Cloud	A Cloud-computing Platform by Google that offers hosting on the same supporting infrastructure that Google uses internally for end-user products like Google Search and YouTube. Cloud Platform provides developer products to build a range of programs from simple Web sites to complex applications

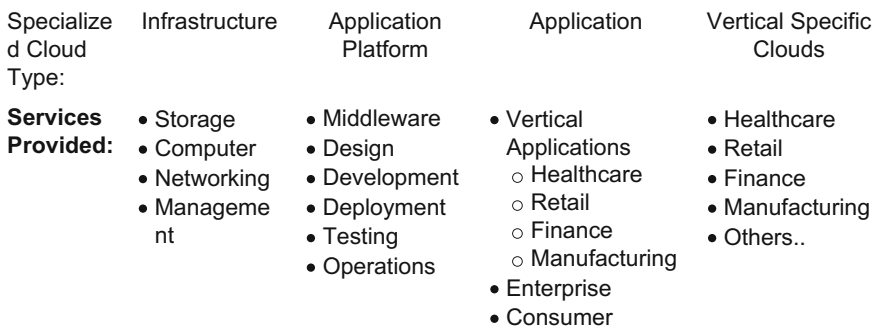


Fig. 10.7 Evolution of Cloud Service Providers

Providers' costs are changing constantly; e.g., AWS has changed its prices 50 times since it launched in 2006. Although other benefits are associated with Cloud infrastructure, the low up-front costs and inexpensive payments for servers attract a large segment of customers and are always mentioned as a major incentive for Cloud adoption. It is important to understand provider pricing to make informed decisions on IT spending optimization. When considering Cloud infrastructure's total cost, businesses need to be asking the following questions:

(1) What should the size of VMs?
(2) How many VMs of each size are needed?
(3) How long one wants to commit to a Cloud provider?
(4) How many peripheral services, such as storage, security, scaling, are needed?
(5) What discounts are available?

Pricing models vary across AWS and Azure and other providers, so above factors are a mere start to compare different CSP offerings. In the coming years, focus may shift from Cloud Computing to specific Cloud Services, such as for HPC and gaming. Specialized Clouds will begin to emerge providing services like those highlighted in Fig. 10.7.

10.8 Types of Cloud Service-Level Agreements

A service-level agreement (SLA) is a contract negotiated and agreed between a customer and a service provider. The service provider is required to execute incoming requests from a customer within the negotiated quality of service requirements for a given price. Due to variable load on a server in Cloud, dynamically provisioning of computing resources to meet an SLA, and allowing for optimum resource utilization, is non-trivial. An example is the guarantee of network bandwidth or data read–write speeds for an application. These tasks are generally conducted on shared resources, such as a Network Interface Card, or a memory controller on a server, with multiple virtual machines or applications accesses data over the same set of physical wires. Result is a conflict of demand for resources and their availability in a shared multi-tenanted server, causing unexpected wait times. The only way to guarantee a certain bandwidth is to reserve it in advance and leave it unused until an application requests it. However, that will cause unused capacity and higher cost for the Cloud operator. Hence, they often over-allocate the same resource to multiple customers and avoid specific SLAs with precise guarantee of quantified performance.

Most Cloud SLAs specify only uptimes, stating that their computing servers, storage devices, and networks will be available for a certain percentage of the time. Service is measured by "9's" of availability, such that 90% has one 9, 99% has two 9s, 99.9% has three 9's, etc. A good High Availability (HA) package with sub-standard hardware has 3 nines, whereas an enterprise class of hardware with a stable Linux Kernel offers 5 or more nines. Below is an example of typical availability levels in Table 10.1.

Microsoft Azure SLA [6] offers commitments for uptime and credits for any downtime. An example of virtual machines connectivity which encompasses server and network uptimes is given below, followed by credits offered for any downtime in Table 10.2.

- Monthly Uptime = (Minutes in the Month − Downtime)/Minutes in the Month.
- For all virtual machines that have two or more instances deployed in the same Availability Set, we guarantee you will have virtual machine Connectivity to at least one instance at least 99.95% of the time.

Table 10.1 Examples of availability by the nines

9's	Availability (%)	Downtime/year	Examples
1	90.0	36 days, 12 h	Personal computers
2	99.0	87 h, 36 min	Entry level business
3	99.9	8 h, 45.6 min	ISPs, mainstream business
4	99.99	52 min, 33.6 s	Data centers
5	99.999	5 min, 15.4 s	Banking, medical
6	99.9999	31.5 s	Military defense

Table 10.2 Example of MS Azure credits in event of downtimes

Monthly uptime percentage (%)	Service credits (%)
<99.95	10
<99	25
<95	100

Table 10.3 Example of Google Cloud credits in event of downtimes

Monthly uptime percentage (%)	Percentage of monthly credits (%)
99 to <99.95	10
95 to <99	25
<95	50

- For any Single Instance virtual machine using premium storage for all Operating System Disks and Data Disks, Azure guarantees virtual machine Connectivity of at least 99.9%.

Similarly, Google Cloud Platform offers a stringent SLA [7] and credits back for any downtime as shown in Table 10.3:

- **"Downtime Period"** means a period of 60 consecutive seconds of downtime. Intermittent downtime for a period of less than 60 consecutive seconds will not be counted toward any Downtime Periods.
- **"Monthly Uptime Percentage"** means total number of minutes in a month, minus the number of minutes of downtime suffered from all Downtime Periods in a month, divided by the total number of minutes in a month.
- **"Maximum Financial Credit"** refers to aggregate number of Financial Credits issued by Google to its Customers, for all Downtime Periods in a single billing month. It is limited to be less than 50% of the amount due from customer for the Covered Service, for the applicable month. Financial Credits will be in the form of a monetary credit applied to future use of the Covered Service and will be applied within 60 days after the Financial Credit was requested. Note that such policies are dynamic and subjected to change.

Besides availability, SLAs should specify downtime, mean time to repair, and mean time to respond. Scheduled maintenance of updating systems does not generally count as downtime. Cloud Service Providers offer multi-level SLAs; e.g., Rackspace offers Managed and Intensive Levels of support [8] as described below:

- 100% network uptime guarantee,
- 1-h hardware replacement,
- URL, port availability, and hardware monitoring,
- Managed firewall and VPN access,
- OS patching,
- Bandwidth and backup performance utilization.

Following services are offered only to Intensive Level customers for a higher cost:

- Server Virus Scanning,
- Guaranteed response times for support requests,
- Advanced system performance and device monitoring,
- Custom configuration trend performance.

In additional, following optional services are available for an additional cost:

- Distributed Denial of Service (DDOS) attack mitigation,
- Managed Security Services,
- Encrypted backups.

However, Rackspace customers may not get predicted, measurable, and quantified performance levels. Similarly, Amazon offers dedicated servers, but customers have to be content with whatever performance occurs even on those machines. This is one of the key limitations currently observed in the Public Clouds.

10.9 Summary

The entire Cloud Computing usage model is based on a pay as you go or pay for what users consume. This enables Cloud customers to not buy IT hardware or software products, but rent them on need basis, e.g., how many CPU cores are needed and for how long, with memory and persistence storage space, etc. The persistence storage also comes in various forms, such as disks directly attached to the computing instance for faster access, but at a higher cost, or stored across the network in a long-term storage repository at a cheaper cost, higher latency, etc. It is up to a user to understand their needs, pick a Cloud Service Provider, and wisely choose additional services or capabilities to do auto-scaling, load balancing, and plan for future growth.

10.10 Points to Ponder

1. **NIST has well-defined Cloud Standards; compare how Amazon, Google, and Microsoft Public Cloud offerings comply with NIST standards?**
2. **Amazon is innovating many additional services to attract and keep its EC2 customers. Give some examples?**
3. **Open Stack has been using open source code to build interoperable cloud solutions, but its adoption has been slow, why?**

4. **A start-up wants to save money, and their tasks require 4 GB memory. Should it rent a small size virtual machine (VM) that costs less, but supports only 2 GB, or a medium size that costs double?**
5. **What is the value of a constant IP address for a Web site hosted in the Cloud?**
6. **Under what circumstance will you prefer to use a load balancer versus an auto-scaler in a Cloud?**
7. **What criterion you would use to compare and contrast different Public Cloud Service Providers?**

References

1. Webinar "beyond the bill" by Forrester research (2011)
2. https://aws.amazon.com/ec2/pricing/
3. http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpc-ip-addressing.html
4. https://aws.amazon.com/elasticloadbalancing/
5. https://aws.amazon.com/autoscaling/
6. https://azure.microsoft.com/en-us/support/legal/sla/
7. https://cloud.google.com/pubsub/sla
8. https://www.rackspace.com/en-us/managed-hosting/service-levels

Chapter 11
Analytics in the Cloud

11.1 Background and Problem Statement

The sheer volume of information available on Cloud, and the rate at which new data is being generated, is overwhelming the capacity of enterprises to manage it and people to use it in a meaningful manner. This data deluge has surpassed the capacity of existing data centers to store and process it in a timely manner. This gave rise to a new class of algorithms, such as MapReduce, which we shall study in a latter section. First, consider the following statistics from 2017 about what people do on the Internet in a typical minute [1]:

1. Google does 3.6 million searches/min.
2. Spammers send 100 million e-mails/min.
3. Snapchatters send 527,000 photos/min.
4. Weather channel forecasts rain or shine 18 million times/min.
5. Netflix customers watch 69,000 h of video/min.
6. YouTube users watch 4.1 million videos/min.

These and a few more mind-blowing statistics are shown in Fig. 11.1, representing activities of 3.7 billion people on the Internet every single minute in a day.

Americans alone use over 2.6 million gigabytes (2.6×10^{15}) of data every single minute of every single day. This is not just limited to online behavior, as Uber customers take 45,787 trips each minute and Amazon customers purchase $250K+ of goods/min. We are still at the knee of a hockey-stick curve, as only half of world's population is online and most of them on slow Internet or wireless connections. With the advent of 5G technology, best is yet to come. Expect the data usage to grow by 10× over next few years.

Problem this poses for Cloud Computing infrastructure is not just the storage or transportation of this data, but processing it in a meaningful way to draw inferences. An example is of a retail company that needs to order a certain color and style of sweaters in summer. It has to be done months before winter comes, so merchandise

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6_11

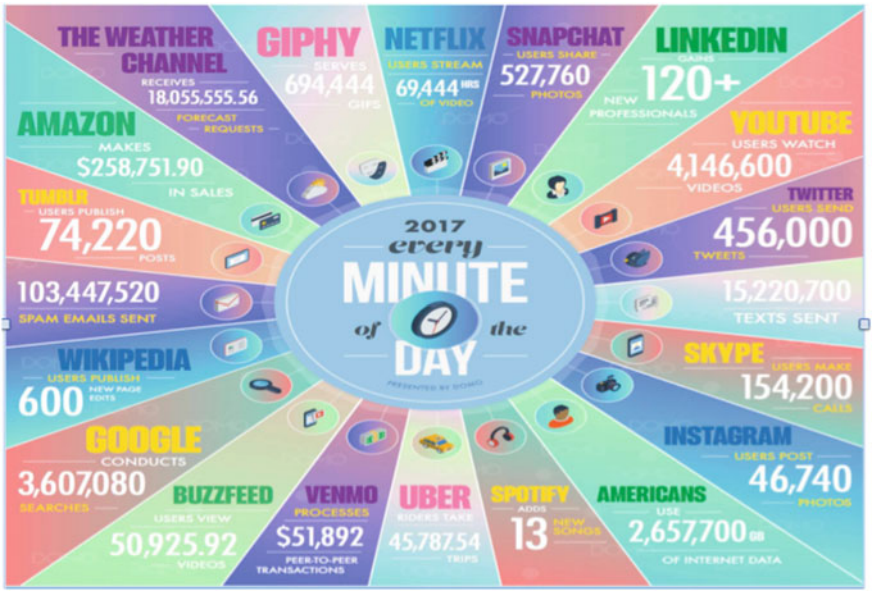


Fig. 11.1 Worldwide activity/minute of internet users [8] in 2017

will be on the store shelves for customers to buy. The assessment of customers' demographics and their preferences have to be guessed right, else most of these sweaters will have to be sold in loss making clearance sales. This problem and many like it are termed as Big Data [2]. It refers to extremely large datasets that need to be analyzed computationally to reveal patterns, trends, and associations. A dataset refers to a collection of related sets of information that is composed of separate elements but can be manipulated as a unit by a computer. An example is sales figures for a store, including the details of the items sold, when and preferably some information on the customer types. Then datasets for different stores can be compared and contrasted to see what type of items should be placed in which store and in which season etc. Predictability becomes harder if it relates to human behavior and interactions versus machines. Thus, a car manufacturing company can be automated with robots and its material ordering can be done in a predictable manner, but how those cars will sell in the marketplace is anybody's guess.

In a 2001 research report [2], Gartner defined data growth challenges and opportunities as increasing across three dimensions:

1. Volume (amount of data),
2. Velocity (speed of data in and out), and
3. Variety (range of data types and sources).

Much of the industry continues to use this "3Vs" model for describing big data. However, in this decade, 3Vs model has been expanded with two other characteristics of big data:

4. Variability: Inconsistency of datasets can hamper processes to handle and manage it. Big Data is often a by-product of digital interactions, e.g., Facebook posts or trends of Tweeter tweets.
5. Veracity: Data quality can vary greatly affecting the accurate analysis. Machine learning can be used to detect patterns and predict future behavior.

Above two is where the analytics in the Cloud become interesting and necessary, as no single server alone is able to process Big Datasets in a reasonable manner. Businesses are starting to use Cloud analytics on Big Data to attract new customers and better serve their existing customers, in cost-effective ways. An example is online booksellers suggesting additional titles when a book is purchased, based on what other readers are reading, or sending targeted customized advertisement to Internet users based on their search patterns or social media activities.

Some prevailing uses of Data Analytics in the Cloud [3] are as follows:

- **Social Media**: Billions of users are active on applications such as Facebook, Instagram, and Twitter sharing their stories, opinions, and preferences. This is heaven for marketers to identify potential customers as well as looking for what others are saying about their products or services online. By searching and linking activities across different online sites, it has become easier than ever before to construct a customer's profile, even without meeting that person.
- **Tracking Products**: An online business can track their inventory across warehouses and ship items to customers from the nearest location to minimize shipping time and costs. Similarly, new products can be ordered to replenish supplies and returns can be tracked in an automated way.
- **Tracking Preferences**: Online movie and song companies log what each user watches or listens to. Then, this information is used to recommend other movies or songs along similar themes, to keep the user interested. Internet has become a battleground for eyeballs and mindshare to keep user engaged, so more services and advertisements can be served in a relevant manner.
- **Keeping Records**: Cloud enables real-time recording and sharing of data regardless of location. An example is an online retailer notifying customers when goods are delivered at their home. Furthermore, a facility is offered to alert and buy next round of supplies at home just before the previous batch finishes. This data is stored in Cloud to track patterns across Regions and seasons, so business can stock their goods in an efficient manner.

An advantage of data analytics in Cloud is that entire datasets can be used instead of smaller statistical samples that may not represent the heterogeneous nature of a Big Dataset. This helps to eliminate guesswork and enables identification of data patterns to minimize uncertainty. In order to process a vast amount of data for meaningful results, new techniques such as MapReduce have emerged.

11.2 Introduction to MapReduce

Every problem has a solution; we just need to find it. Sometimes the solution is not scalable, e.g., with Non-Polynomial (NP) class of algorithms where the run-time grows exponentially with increase in the size of datasets.

MapReduce is a possible solution for large datasets that splits the inputs into independent chunks, which are mapped to different processors and processed in a parallel manner. Then output of each mapped tasks is combined (i.e., reduced) to get the final output that is equivalent to the value as if the original data was processed sequentially. An example is the task of finding how many times the word Cloud is used in this book, which admittedly will be many. One way to find this is by writing a single-threaded program that reads each word, compares it to string "Cloud", and increases a single counter by 1 if match is true. Given the number of large number of words, although nowhere close to Big Data, our sequential algorithm will process one page at a time. However, since each page is independent of others, imagine if these pages are mapped to different processors, each of which runs our sequential algorithm in parallel and updates its own counter, and at the end, all the counters are added (reduced) to produce a single integer value. That result should be identical to our first single algorithmic run through the entire book. However, the run-time of our little MapReduce algorithm will be much faster because each thread has to read through only one page. The only overhead is in the split or map, and final reduce tasks. Figure 11.2 illustrates both the serial and parallel nature of our search schemes.

11.3 Introduction to Hadoop

Hadoop is an open-source software project [4] for reliable and scalable computing. The Apache Hadoop software library is a framework that allows for the distributed processing of large datasets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than relying on hardware to deliver high availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly available service on top of a cluster of computers, each of which may be prone to failures.

The project includes these modules, with the full software stack shown in Fig. 11.3:

1. Hadoop Common: The common utilities that support the other Hadoop modules.
2. Hadoop-Distributed File System (HDFS™): A distributed file system that provides high-throughput access to application data.

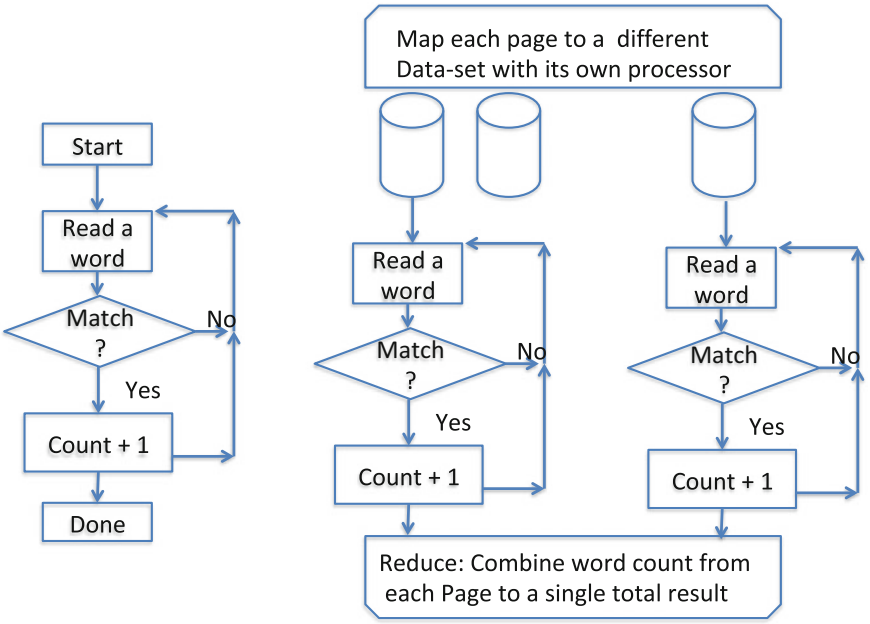


Fig. 11.2 A serial word count on left will be much slower than the MapReduce on the right-hand side, due to the parallel nature of the latter, both giving the same final result

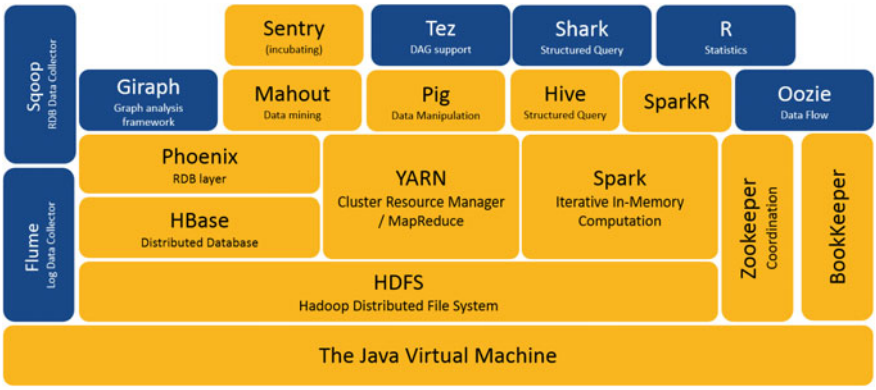


Fig. 11.3 A representation of Hadoop software stack [4]

3. Hadoop YARN: A framework for job scheduling and cluster resource management.
4. Hadoop MapReduce: A YARN-based system for parallel processing of large datasets.

Other Hadoop-related projects at Apache include:

5. Ambari™: A Wb-based tool for provisioning, managing, and monitoring Apache Hadoop clusters which includes support for Hadoop HDFS, Hadoop MapReduce, Hive, HCatalog, HBase, ZooKeeper, Oozie, Pig, and Sqoop. Ambari also provides a dashboard for viewing cluster health such as heat maps and ability to view MapReduce, Pig, and Hive applications visually along with features to diagnose their performance characteristics in a user-friendly manner.
6. Avro™: A data serialization system.
7. Cassandra™: A scalable multi-master database with no single points of failure.
8. Chukwa™: A data collection system for managing large distributed systems.
9. HBase™: A scalable, distributed database that supports structured data storage for large tables.
10. Hive™: A data warehouse infrastructure that provides data summarization and ad hoc querying.
11. Mahout™: A scalable machine learning and data mining library.
12. Pig™: A high-level data-flow language and execution framework for parallel computation.
13. Spark™: A fast and general compute engine for Hadoop data. Spark provides a simple and expressive programming model that supports a wide range of applications, including ETL, machine learning, stream processing, and graph computation.
14. Tez™: A generalized data-flow programming framework, built on Hadoop YARN, which provides a powerful and flexible engine to execute an arbitrary DAG of tasks to process data for both batch and interactive use-cases. Tez is being adopted by Hive™, Pig™, and other frameworks in the Hadoop ecosystem, and also by other commercial software (e.g., ETL tools), to replace Hadoop™ MapReduce as the underlying execution engine.
15. ZooKeeper™: A high-performance coordination service for distributed applications.

A wide variety of companies and organizations use Hadoop for research and production.

11.4 Usage of Amazon's MapReduce

Amazon's Elastic MapReduce (EMR) is a Web Service [5] that uses Hadoop for processing large amounts of data. Advantage of using EMR over generic Hadoop is that the system administration tasks, e.g., compiling, building, and maintaining the entire software framework, are handled by Amazon.

Public Cloud customers have been using EMR for log files analysis, Web indexing, data transformation, machine learning, financial analysis, scientific simulation, and bioinformatics projects. These can use data already stored in AWS databases such as Amazon Simple Storage (Amazon S3). Its security is handled by EC2 firewall, and users have the following level of control:

(1) Over all machines within a user's cluster
(2) Root access to every instance
(3) Ability to install additional applications
(4) Customization of every cluster
(5) Supporting multiple Hadoop open-source distributions and applications.

An example of an EMR cluster with master and slave nodes is shown in Fig. 11.4. Master node handles the Mapping tasks, whereas each slave node does the computation or data processing, results of which are then presented back to the master for final reduction.

Master node manages the cluster:

(1) By running software components
(2) Coordinate the distribution of data and tasks among other nodes for processing
(3) Tracks status of tasks and monitors cluster health.

Slave nodes do processing work as follows:

(1) Core node: a slave node that has software components, which run tasks and store data in the Hadoop-distributed file system (HDFS) on the user cluster.

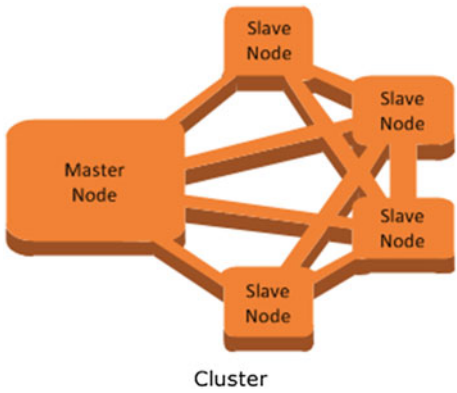


Fig. 11.4 An example of a master–slave cluster for MapReduce tasks

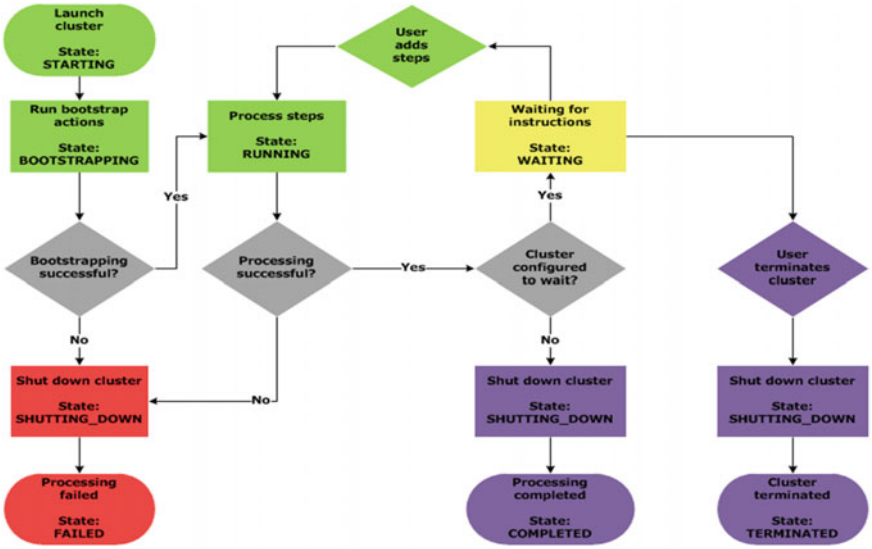


Fig. 11.5 An EMR cluster life cycle

(2) Task node: a slave node that has software components, which only run tasks. Task nodes are optional.

Life cycle of an EMR cluster, just like any other virtual server network in a Public Cloud, is ephemeral as shown in Fig. 11.5. Based on the dataset and desired parallelism, user can decide how many slave nodes are needed. Speed up will be proportional to it, barring any inherent data dependencies in the computational algorithms. User will only pay for the uptime of cluster, so it is better to not keep any nodes idle.

A Python implementation of our previous word count using MapReduce on EMR's Hadoop is shown below. It will count the number of times a word occurs within a text collection.

```
#!/usr/bin/python
import sys
import re
def main(argv):
line = sys.stdin.readline()
pattern = re.compile("[a-zA-Z][a-zA-Z0-9]*")
try:
while line:
for word in pattern.findall(line):
print "LongValueSum:" + word.lower() + "\t" + "1"
line = sys.stdin.readline()
except "end of file":
return None
```

```
if __name__ == "__main__":
main(sys.argv)
```

This can become the basis of sentimental analysis in Cloud, e.g., reaction of Twitter users to any latest breaking news story, if their reaction can be categorized in favorable (i.e., like, happy, prefer, good) or unfavorable (i.e., don't like, angry, bad, ugly) words. As one can see, any such prediction is subjective based on the extent of sentimental analysis words used and may not capture the whole sentences or meanings. As an approximation, this has been found to be useful [6] on social media trends.

11.5 Twitter Sentimental Analysis Using Cloud

An example of studying the popularity of US presidential candidates before 2016 elections will be presented. The sole purpose of this project was to study the positive and negative sentiments of Twitter users during a specific period of time, and results were obtained a year before the actual election took place. This project was done by two graduate students (Nora Susan Kurian and Ruchika Tayal) in a Cloud Computing class that one of the authors of this book taught at Santa Clara University during the Fall, 2015 quarter.

These students followed Amazon's guidelines [7], and used Amazon's EC2 and EMR services to setup their MapReduce cluster. Then on a live Twitter stream after presidential debates or any major news item, a key-value pair analysis was done to search for positive and negative sentimental words in users' tweets. The actual results much before presidential primaries were held are shown below in Fig. 11.6. Looking at these, there should be no surprise at how the actual elections turned out then.

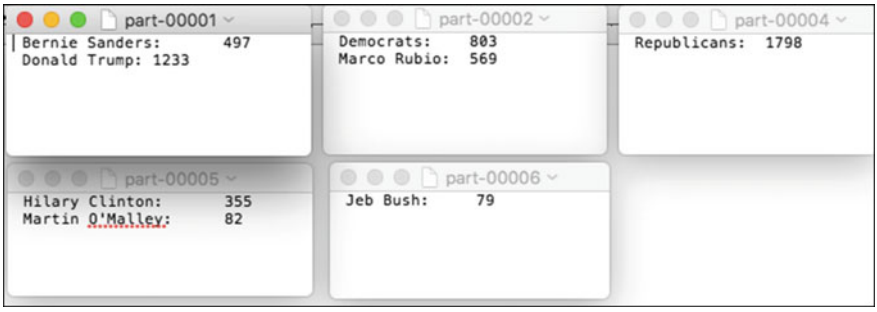


Fig. 11.6 A Twitter sentimental analysis study done in fall, 2015 by students

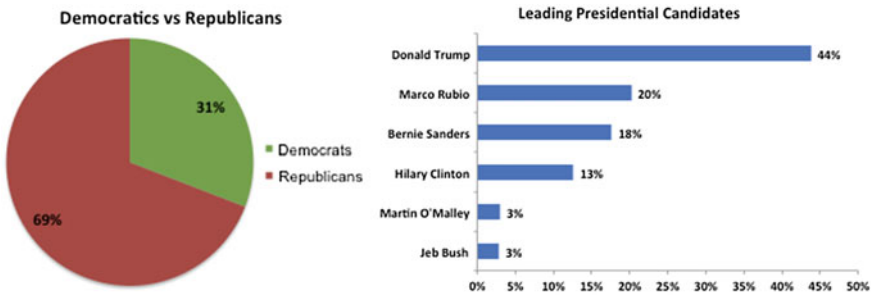


Fig. 11.7 Analysis of 10,000 users' tweets done during Fall, 2015

Further details of analyzing 10,000 users' tweets revealed that:

a. Republicans were leading as 69% of the Twitter users are talking about them.
b. Within Republicans, Donald Trump was the leading candidate with 66% of the users mentioning him in their tweets
c. Within Democrats, Bernie Sanders was leading candidate with 53% of the users mentioning him in their tweets
d. Out of the six candidates, Donald Trump was leading by a large margin with 44% twitter mentions (Fig. 11.7).

11.6 Future Possibilities

As the above students' project shows, possibilities of big data analysis are promising and the same can be applied to predicting financial markets. For example, before the 2008 housing crisis in USA, financial markets were unaware of the mortgage payments being missed by many new home owners, although some data was available at the individual levels and in local markets across the country. In theory, a Big Data analysis of many small patterns could have been used to predict the larger market direction. Some individual traders saw the big crash coming and profited from their insights, as immortalized in "The Big Short" Hollywood film. A tantalizing possibility is if future terrorist attacks can also be predicted by a combination of monitoring online chatter, financial transfers and assault material purchases, etc. Training such an algorithm will be challenging. It may be acceptable to have a few false positives, as long as no large attack is missed. This is the essence of machine learning, which needs large amount of data from multiple sources and combine it with previous occurrence to predict any future results.

11.7 Points to Ponder

1. **Who owns the datasets in Big Data?**
2. **What is desirable for external researchers to collaborate for data analytics in Cloud?**
3. **Are there any vendor lock-in concerns with big data?**
4. **AWS offers several attractive services, not required by NIST, such as Elastic IP and Auto Scaling. Can you think of a few more examples?**
5. **What equivalent services does other Cloud Service Providers offer, such Microsoft Azure and Google's GCP?**
6. **How do prices of different server types vary across different Cloud Service Providers?**

References

1. https://www.inc.com/john-koetsier/every-minute-on-the-internet-2017-new-numbers-to-b.html
2. https://en.wikipedia.org/wiki/Big_data
3. http://technologyadvice.com/wp-content/uploads/2013/05/Data-Analytics-in-Cloud-Computing_TechnologyAdvice.pdf
4. Hadoop: https://hadoop.apache.org
5. Amazon Elastic Map-Reduce: https://aws.amazon.com/emr/
6. Kouloumpis E et al (2011) Twitter sentiment analysis: the good, the bad and the OMG!. In: Processings of the fifth international AAAI conference on weblogs and social media, pp 538–541
7. https://aws.amazon.com/blogs/big-data/building-a-near-real-time-discovery-platform-with-aws/
8. Domo, marketing data company's 5th annual "data never sleeps" infographics (2017)

Chapter 12
Future Trends in Cloud Computing

12.1 Revisiting History of Computing

The journey of computing began with a single user running a single job. We have come a long way since. In the next phase of evolution, multiple users shared a computer system. It further evolved into a networked computer system, which was accessible to remote users. With the emergence of PCs (Personal Computers) in the 1990s, we witnessed PCs being used as a gateway to networked computers, as shown in Fig. 12.1.

12.2 Current Limitations of Cloud Computing

Thus far, we have studied the benefits of Cloud Computing including Economic, Elastic Infrastructure, On-Demand Resources, Pay for what you use. However, there are a few unseen costs before these benefits can be realized, and some are listed below:

(1) **Data Movement**: Since servers are located in a remote data-center, any input data needed for computation needs to be moved there, and results need to be moved out. Such I/O (input–output) transactions cost additional money in most Public Clouds and add to the latency as compared to computing on local servers. An example is the emerging area of self-driven cars, which have a multitude of sensors including multiple cameras. There may not be sufficient time to run the image processing algorithms in a remote Cloud due to the dynamic nature of traffic for real-time decision making while driving. Thus, a self-driven car needs to have server-like computing on board. By some accounts, a self-driven car in future may generate up to 5 TB of data per day, all of which needs to be stored and processed locally, representing a mini data-center on the Wheels.

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6_12

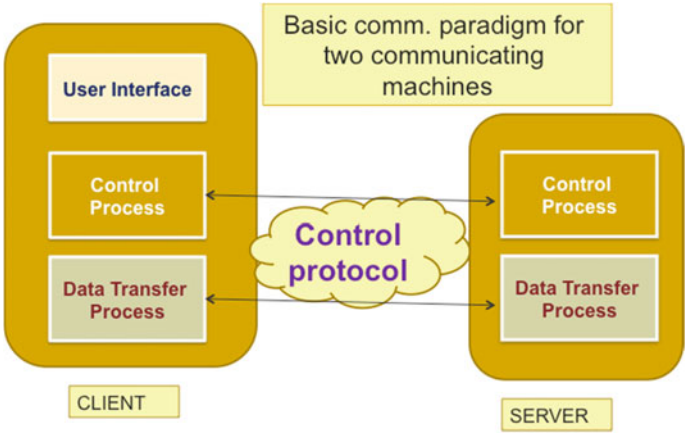


Fig. 12.1 An example of Client-Server architecture, with multiple users on left side interacting with a server

(2) **Loss of Control**: When a user's e-mails are hosted in the Cloud, these are often examined by bots, which then decide on relevant advertisements to display, to generate revenue for e-mail providers such as Google's Gmail. However, this raises a question on who owns the e-mail content and who can access it. For example, if there is a legal case and court subpoenas the e-mail provider to turn over the e-mails, it will be hard for the provider to say no. At the end, if a user wishes to own the content and keep it private, such as pictures or other business data, then it should be kept on a local computer.

(3) **Perception of Cloud Security**: While multiple people can access a data-center in Cloud, it may be no less safe than an enterprise data-center. Due to the loss of control as mentioned previously, there is a perception of Public Cloud being less secure. This in author's opinion is a red herring, and additional steps can be performed such as to encrypt one's data in the Cloud and also any virtual machine when running on a multi-tenanted server, with keys stored separately.

(4) **Uncertain Performance**: Cloud Computing operators make money by sharing same hardware infrastructure with many customers. While their virtual machines (VM) may be isolated in the memory and running on different server cores, there are other shared resources, such as a memory controller, and networking card that each VM's data must pass through. This creates bottlenecks similar to traffic jams in a data center at entry and exit points, as well as entry and exit to the shared servers. This causes the performance drop of a running VM without any notice. This problem has been described previously as a noisy neighbor and results in a delay in task completion.

All of the above, and a few similar issues, are causing some customers to rethink their Cloud Computing approaches, such as Hybrid Computing with critical tasks being performed using onsite infrastructure.

12.3 Emergence of Internet of Things (IoT)

Another emerging trend is a Cloud driven by things, versus current Cloud Computing mostly driven by people, as cameras and wireless sensors are becoming pervasive. Their applications include Retail Solutions, Transportation and Automotive, Industrial and Energy, etc. An example of retail industry is Amazon's user-facing portals where customers can visualize things and transact them. An example of Transportation and Automotive is a Software-Defined Cockpit in a commercial aircraft, or an autonomous vehicle. An example of manufacturing is a smart factor with robots, or energy savings in a building. Lastly, additional market segments such as health, print imaging, gaming, and education are being digitized at an unprecedented rate. The phrase "Internet of things" was first used by British technology visionary Kevin Aston in 1999. His perception was to think of "objects in physical world connected by sensors." Internet Architecture Board (IAB) RFC 7452 provides the definition of IoT, as follows:

- "Internet of Things" (IoT) denotes a trend where a large number of embedded devices employ communication services offered by Internet protocols. Many of these devices, often called "smart objects," are not directly operated by humans, but exist as components in buildings or vehicles, or are spread out in the environment.

Four basic communication models for IoT are:

1. Device to device,
2. Device to Cloud,
3. Device to gateway,
4. Backend data sharing model,

We are more interested in #2 and #4, as both involve Cloud services. An example is shown in Fig. 12.2, of home appliances such as a thermostat-controlled A/C connected to Cloud for better energy management.

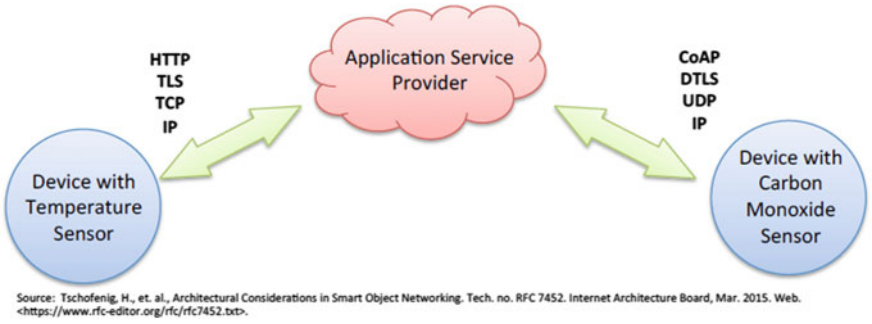


Fig. 12.2 Cloud-based energy management, monitoring, and optimization

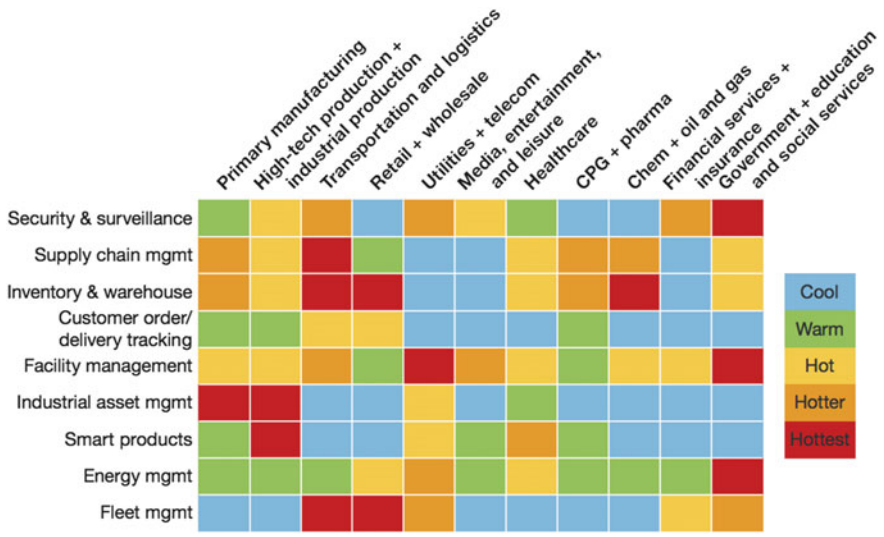


Fig. 12.3 Heat map of key IoT opportunities by industries and applications

Bains [1] predicts that by 2020 annual revenues for the IoT vendors could exceed $470B by selling hardware, software and comprehensive solutions. Forrester [2] published a heat map in 2016, showing how opportunities vary by the industry and applications, as shown in Fig. 12.3.

As seen above, the hottest (i.e., most financially attractive) applications are in transportation, government, and retail. Further discussion of IoT business opportunities is beyond the scope of this book, but can be found in [2]. These new opportunities also bring new security challenges. As an example, if these devices are connected to Internet then a hacker can potentially gain access to read the output data or alter device configurations to yield unexpected results. We will explain the security implications and potential solutions in a later section.

12.4 Emergence of Machine Learning

Due to the preponderance of IoT data being generated, it is nearly impossible for a human to draw any meaningful comparisons. This is reviving expert systems and artificial intelligence (AI), this time aided by unprecedented compute power and self-learning systems that improve with more incoming data. Some of the use cases for IoT-based machine learning are shown in Fig. 12.4, where a smart meter and building temperature control based on when its occupants are expected to arrive or leave. Furthermore, different parts of buildings where people are present or absent can be heated or cooled at different levels, instead of a single setting for the whole floor.

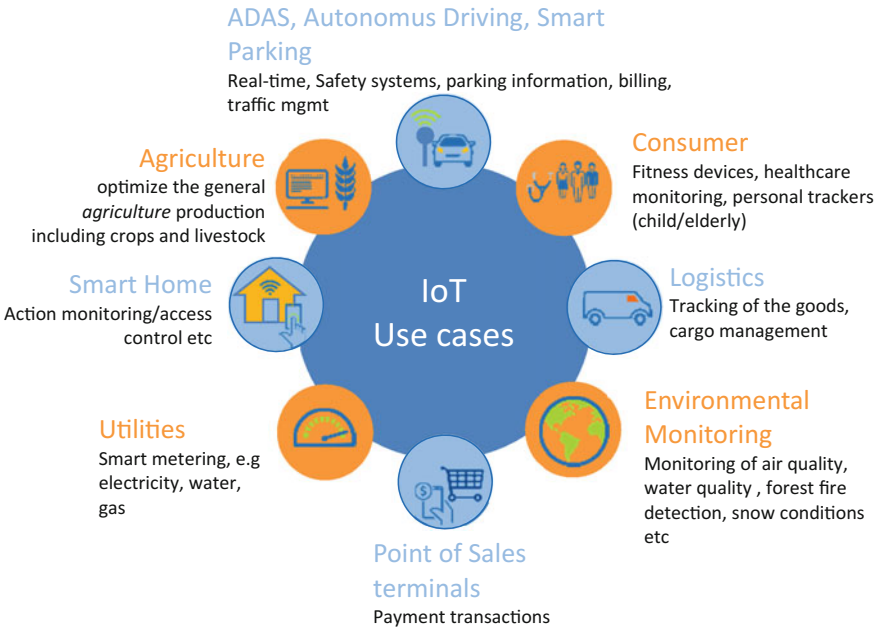


Fig. 12.4 IoT use case that needs both local and Cloud Computing

At a basic level, we can add intelligence to remote devices if processing elements and storage are used to locally collect data for rule-based control decisions. As an example, in a company cafeteria with multiple work-shifts and different number of employees served on different days, an intelligent refrigerator can check for the remaining packaged food items, including their expiration dates. If the goal is to have food ingredients for at least next 2 days in store, cafeteria manager can be notified to replenish them as needed. Machine learning part comes from not having a predetermined supply at hand, but making the solution self-learning based on consumption pattern of employees. If it is a Friday, and company is closed for the next 2 days, then system will look for supplies needed until the following Tuesday. Also, on different work days, menu and specific food items needed may vary, requiring a solution that can intelligently predict what needs to be ordered to minimize expenses and avoid food wastage, while ensuring that essential ingredients will never run. In general, such smart devices offer desired functionality and operate in an energy-efficient manner with minimal compute power and memory, while connected to a mobile App and Cloud on the backend. These connections are needed for human intervention and record keeping.

Current machine learning technology has its limitations. A case in point is an automobile accident [3] in which driver was on a self-driving system, but vehicle's camera failed to recognize a white truck reflecting a bright sky, and the car failed to brake. However, regulatory authorities absolved the automaker and blamed this

accident on dead driver. They ruled that driver should have paid attention and not be depended on the self-driving system. Future liability in accidents will be hotly contested.

Machine learning systems have proven useful in retail as the vendors can find the items that customers are buying, or not, and accordingly build next production order. In addition, they can build customer profiles and suggest additional items to customers who buy an item, based on what others bought after buying the same item. This has contributed to enormous success for online retailers such as Amazon.

12.5 Emergence of Edge Computing

With many IoT devices and use-cases, it is imperative to have localized compute power and data storage. An example is a car, as shown in Fig. 12.5, which can generate up to 5 TB of data/day. This comes from onboard cameras, IR sensors, and data collected from the engine, brakes, etc. However, an autonomous car cannot pause for a server in the Cloud to make a decision to accelerate or brake. Hence, it needs sufficient compute power in the car to drive safely, by some to dub it as a "Data center on the wheels." It can synch up with a remote data center in the Cloud overnight while parked, but on the road must focus on safe driving with real-time decision making. Hence, a part of the Cloud is migrated from remote data-center to field, termed as Edge Computing.

Similar examples can be found in other application domains, such as smart homes with security cameras, which can decide on the spot if an intruder is a family member or a stranger, and in the latter case sound an alarm.

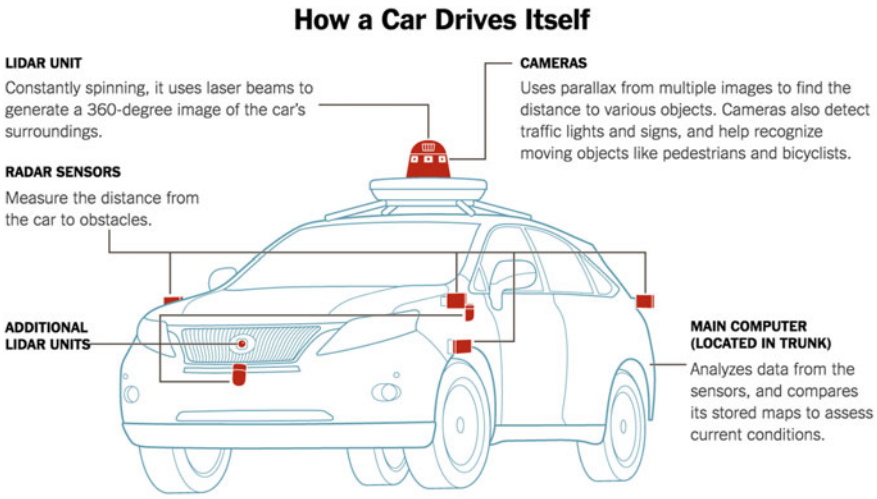


Fig. 12.5 A car's self-driving system with multiple sensors [3]

12.6 Security Issues in Edge Computing

Security concerns abound with the emergence of Edge Computing. In the car example, its computers are not behind a firewall but physically accessible to many people besides the owner. When a car is taken to a mechanic for an oil change or another repair, there is a risk of someone tempering with the hardware or software components setting up a future failure of the self-driven car. It is also possible for someone to access private data stored in the car, e.g., its travel points. Vulnerabilities in other unprotected devices, such as home appliances (TV, Fridge) on a network, can be used to launch a cyber attack. A recent DDOS (Distributed Denial of Service) attack was launched using hijacked home security cameras, while in another instance private video clips were stolen and posted on Internet.

Even for a simple home automation system, such as an intelligent door lock, it needs following security features for safety:

(1) A firewall to dissuade remote hackers with login authentication.
(2) Authentication requires identification of phone numbers, password, or biometrics such as face recognition, thumbprint, or retina scan.

Note that any single biometric can be easily defeated, e.g., a pictured mask to fool a face recognition, or copy of a thumb print image, presented to the door camera. It is desirable to have a multi-factored authentication system. Furthermore, a data-logging system is needed to record who opened or locked the door, and when. This data is immediately backed to a remote Cloud to avoid local tempering. Machine intelligence can be used to create a regular usage pattern and flag anomaly if door is opened at unexpected hours or with unusual frequency.

We need to remember that IoT devices are constantly collecting data about an environment or individuals, which can be potentially shared with third parties compromising privacy. It can range from personal preferences of Web-browsing habits, TV channels selection, or images from home security cameras. Some devices can be programmed to selectively transmit data to a Cloud Service for processing, e.g., a security camera which has a buffer of 15 s, but records and transmits a 30 s of clip only if any motion is detected, for 15 s before and 15 s after the motion is detected. This reduces storage requirements but increases chances of a mistake. Such devices are designed to render service with minimal intervention, and yet they need to be directed using voice activation or image recognition. On the flip side, if there is a continuous recording dashcam, which is a forward-looking recording device in a car. Purpose of this dashcam is to establish other party's guilt in case of an accident in a vehicle. It will also record voice conversations of passengers potentially violating their privacy rights. It is recommended for the vehicle driver to inform passengers and seek their consent in advance to make them aware.

For ensuring trust in Edge Computing, it has to start with a trusted environment, trusted protocols, and temper proof components. Vendors need to provide "anti-temper" solutions to start with. Software upgrades in field are needed for any

bug fixes during the lifetime of an Edge Computing device. A secure channel must exist to provide signed binary packets that are transmitted and installed in the field, e.g., on a car or TV at home. In our door example, vendor needs to provide an anti-temper solution, to prevent someone locally changing the firmware or settings in an unauthorized manner. Even remote software upgrades are authenticated, as unprotected home appliances can be used to launch cyber attacks, e.g., someone opening doors via remote Internet attacks. Besides security, there are privacy concerns, as home sensors are collecting data about individuals that can be shared with third parties for commercial purposes.

Undesirable consequence may emerge if a third party can remotely gain control of a self-driven car causing an accident on the road, or someone with malice can access the medicine drip-meters in a hospital with fatal consequences for the patients. This can be avoided with a balanced approach to interoperability and access control. This needs to be addressed at different layers of architecture and within the protocol stacks between the devices. Standardization and adoption of communication protocols should specify when it is optimal to have standards. Some vendors like to create a proprietary ecosystem of compatible IoT products. This creates user lock-in to their particular ecosystem, which from a vendor's point of view is desirable because a closed ecosystem approach can offer benefits of security and reduces costs. However, from a user's point of view, such practices can create interoperability problems with solutions from other vendors, thereby limiting user's choices in case of upgrades or future system expansion.

Solution-level cost considerations involve technical factors such as limited internal processing, memory resources, or power consumption demands. Vendors try to reduce the unit cost of devices by minimizing parts and product design costs. It may be more expensive to design interoperability features into a product and test for compliance with a standards specification. A non-interoperable device may lack in standards and the documented best practices. It may limit the potential use of IoT device, and the absence of these standards can result in deviant behavior by IoT devices.

12.7 Security Considerations for Edge Computing

Edge Computing is the most recent inflection point in the history of computing. With the advent of Edge Computing, evolutionary cycles between a concentration of powerful centralized computing and an emphasis on distributed powerful computing have changed to a network made of a combination of powerful centralized powerful computing and distribution of simple computers at the edges of the network. This network has vastly different security requirements. For example, a central system in the Cloud can send security breaches to the edge, or the edge computers can send security breaches to a server in the Cloud. A system-wide trust is difficult to achieve based upon the current start of art strategies, policies,

standards, or implementations. Edge Computing expands the potential threat plane and introduces the possibility of attacks from many directions.

Following classifications describe the types of security issues as related to the Edge Computing, with a few examples:

1. **Identity authentication**: By definition, the number of players in Edge Computing is large and these may not belong to the same organization. It is infeasible to verify their identity in a foolproof manner. Trust needs to be extended, as new customers buy their devices, such as security cameras, and bring these online with a remote registration. Central authority then must depend on the ability of these remote customers to protect their own devices.
2. **Unauthorized access**: Depending on the nature of sensors at the Edge, their access into data-center may be bidirectional in nature. If someone hacks into a remote device to impersonate a previously trusted remote device, it is nearly impossible to differentiate between genuine or fake users. Similarly, someone pretending to act as a central computer can access the remote devices and get critical user-data, such as on remote medical devices.
3. **Denial of service attacks**: An attack launched by hijacking multiple remote devices and simultaneously contacting the central server. This will cause the server to be overloaded, denying access to a genuine device in a timely manner.
4. **Data theft**: Depending on where data is stored and for how long opens the possibility of it being stolen. An example is a security camera at home with local storage. In event of a theft, it may be possible for an intruder to simply remove the local storage, thus circumventing the purpose of a security camera. However, if camera immediately uploads an image to Cloud upon detecting a motion, then any physical tempering will still protect the image of intruders.
5. **Data integrity and falsification**: A key difference between confidentiality and integrity is that in the latter case, an attacker does not need to read the protected data, but merely modify it, e.g., with a buffer overflow, rendering it useless. This system level attack can happen if multiple devices from different sources are writing back to a central server memory or database. This can be protected with assigning a virtual partition or container to the data coming from each distinct source, and checking the address range of each access to prevent data integrity of other users on the same server.
6. **Invasion of privacy**: Since multiple players may combine their data inputs from different sources to arrive at a desired conclusion, e.g., for real-time traffic updates, their identities need to be protected. This may include an individual's location, movements, and any other aspects of personal nature.
7. **Activity monitoring**: A simple example of cell phone which constantly pings the signal tower is sufficient for someone to monitor the location of phone's owners, their movements, etc. Furthermore, if a remote App can turn on the microphone or camera in a phone, then additional information and activities can be monitored in an illegal manner. Similar effects can be achieved with fixed cameras at commercial or public locations, e.g., in a shopping center.

It is recognized that traditional Trusted Compute Boundary (TCB) expands with Edge Computing to include domains that are physically outside the control of remote device or central data-center owners. The best they can do is to monitor/track a threat, identify an attacker, launch a recovery, and prevent false positives. These steps are outlined below:

1. **Monitor/track a threat**: This is possible by establishing a normal usage pattern and then looking for anomalies. Any deviation represents a potential threat.
2. **Identifying attackers**: Once a threat is detected, then attackers need to be identified. These could take the form of an IP address that is repeatedly pinging the central server, to launch a denial-of-service attack.
3. **Attack recovery**: This can take the form of blocking the offending IP address. However, situation is not always so simple as an attacker can corrupt the critical data before the attacker's presence is detected. In such a case, frequent checkpoints must be taken to do a rollback to the known good state.
4. **Accidental and unintentional failures confused with security attacks**: Any detection method suffers from the risks of false positives, e.g., mistaken flagging of a genuine access as a potential threat. An example of this is a stock market trading computer that detects unusual activity, which is genuine yet may flag a false alarm. Similar situation can happen with security alarms due to false sensor activity data, etc. This calls for a learning system that becomes smarter over time.

12.8 Future Work Needed

Internet Engineering Task Force (IETF) has identified the problem of interoperability, as many suppliers build "walled gardens" that limit users to interoperate with a curated subset of component providers, applications, and services.

Interoperability solutions between IoT devices and backend systems can exist at different layers of the architecture and at different levels within protocol stack between the devices. Key is the standardization and adoption of protocols, which should specify when and where it is optimal to use standards. More work is needed to ensure interoperability within the cost constraints for Edge Computing to become pervasive.

There are other regulatory and policy issues at play, such as device data being collected and stored in a Cloud may cross-jurisdictional boundaries, raising liability issues if the data leaks. This is especially important if data is of personal nature, e.g., related to shopping patterns or patient health records.

12.9 Example of an IoT-Based Cloud Service

A Cloud Service where intelligence extends beyond a data-center to the edge based sensors is also known as Fog Computing [4], which is a clever name for gathering and processing data at the local computing devices. In this model, sensors and other connected devices such as cameras send data to a nearby edge-computing device, which has processing power to analyze this data, make some local decisions, and then send the results to the Cloud. BI Intelligence forecasts that 5.8 Billion IoT devices owned by enterprises and governments will use Fog Computing in 2020, up from 570 million devices in 2015 (Fig. 12.6).

An example comes from mining industry [5], where drilling equipment is working in harsh conditions, with autonomous trucks and trains, tunneling and boring machines, moving at high speeds. In order to ensure worker safety and increase productivity, decisions need to be made locally. Even though mining equipment can generate terabytes of data/hour during normal operation, there may not be a reliable connection to backend Cloud given 100 s of feet of underground operation, say in a coal mine. This is where Fog Computing can help by processing the data locally, makes appropriate decisions, and, as shown in Fig. 12.7, sends only small uploads to the backend Cloud every few hours or at the end of each day.

However, this also increases risk of accountability and security as different legal entities may own the local sensors, edge gateway, and backend Cloud. If something goes wrong, e.g., in the event of a mining accident, then finger pointing will begin with hard to assign liabilities. This is where companies offering end-to-end services

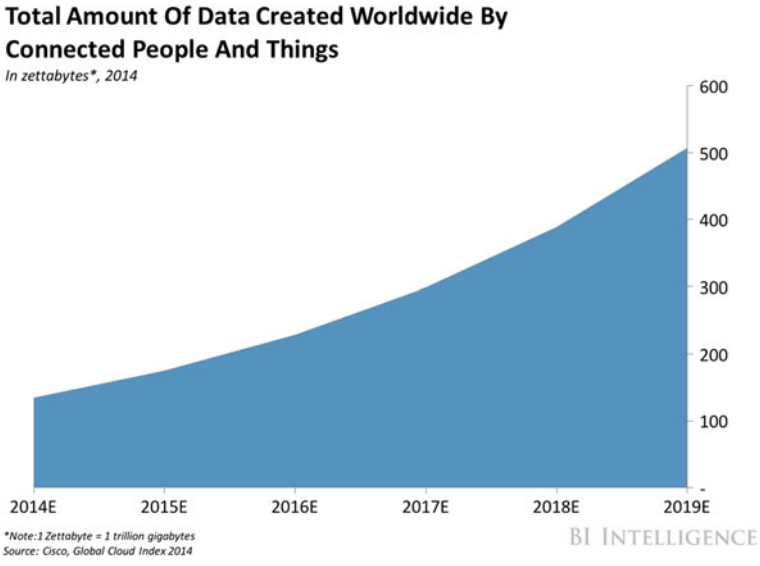


Fig. 12.6 Growth of data being generated by IoT and Cloud together [4]

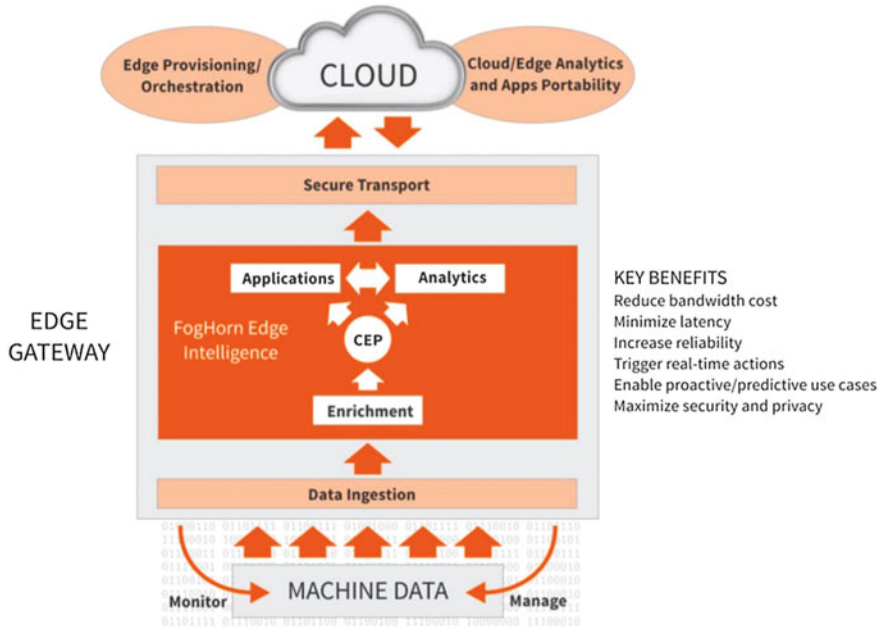


Fig. 12.7 An example of Fog Computing to support real-time decisions [5]

will have an advantage in Edge Computing deployment resulting in a business success. In the long run, as industry evolves and standards emerge, there will be a room for horizontal service and equipment providers to excel at competitive price points, but initially Fog Computing is likely to be a vertical play.

12.10 Summary

Combination of locally intelligent devices with backend Cloud-based processing is giving rise to a new class of Edge or Fog Cloud Computing, which offers new usage models, but also raises potential of new vulnerabilities with possibility of widespread cyber attacks. There are additional concerns of user lock-ins if vendors do not follow interoperability standards in their edge-based devices in proprietary Cloud solutions. Additional issues of user-data privacy and legal jurisdiction currently lag the fast evolution of Edge Computing domain with IoT-based solutions. This requires policy framework to be discussed by vendors and Cloud Service Providers with the users for avoiding any legal pitfalls.

As shown in the historic computing spiral of Fig. 12.1, industry has oscillated between large central computers, to localized computing and hybrid models leading

back to the spiral growth. This now requires large central computers to handle the distributed Edge Computing demand.

This trend is likely to continue as networks will become faster and machines will become more intelligent to recognize patterns of data to make decisions. In this evolution, it is important to develop standards for interoperability of computing devices on the edge and servers on the backend, to ensure a level-playing field for all players.

12.11 Points to Ponder

1. **There is potential to have more devices and machines in an increasingly automated world, and next wave of Cloud Computing growth is coming from IoT. Can you list additional areas to drive the growth of Cloud Computing?**
2. **How could one improve the Cloud's performance and support for IoT?**
3. **Why is Edge Computing needed for self-driven cars in future?**
4. **Can you think of another example of Edge Computing devices on a road?**
5. **What is the trust and security model for edge devices?**
6. **What kinds of attacks are possible using IoT and Edge devices?**
7. **What is the impact of vendor lock-in on Edge Computing devices?**

References

1. https://www.forbes.com/sites/louiscolumbus/2016/11/27/roundup-of-Internet-of-things-forecasts-and-market-estimates-2016/#634d80ab292d
2. https://www.cloudera.com/content/dam/www/static/documents/analyst-reports/forrester-the-iot-heat-map.pdf
3. https://www.nytimes.com/2017/01/19/business/tesla-model-s-autopilot-fatal-crash.html
4. http://www.businessinsider.com/Internet-of-things-cloud-computing-2016-10
5. http://www.nanalyze.com/2016/08/fog-computing-examples/

Chapter 13
A Quick Test of Your Cloud Fundamentals Grasp

1. _____ describes a Cloud Service that can only be accessed by a limited amount of people.

   A. Data center
   B. Private Cloud
   C. Virtualization
   D. Public Cloud

2. _____ describes a distribution model in which applications are hosted by a service provider and made available to users.

   A. Infrastructure as a Service (IaaS)
   B. Platform as a Service (PaaS)
   C. Software as a Service (SaaS)
   D. Cloud Service

3. True or False: Access to a Cloud environment always costs more money to compared to a traditional server environment.
4. _____ is the feature of Cloud Computing that allows the service to change in size or volume in order to meet a user's needs.

   A. Scalability
   B. Virtualization
   C. Security
   D. Cost savings

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6_13

5. True or False: A Cloud environment can be accessed from anywhere in the world as long as the user has access to Internet.
6. Which Cloud characteristic refers to the ability of a subscriber to increase or decrease its computing requirements as needed without having to contact a human representative of the Cloud provider?

   A. Rapid elasticity
   B. On-demand self-service
   C. Broad network access
   D. Resource pooling

7. Which customer scenario is best suited to maximize the benefits gained from using a virtual Private Cloud?

   A. A small start-up business focused primarily on short-term projects and has minimal security policies.
   B. An enterprise whose IT infrastructure is underutilized on average and the system load is fairly consistent.
   C. An enterprise that requires minimal security over their data and has a large existing infrastructure that is capable of handling future needs.
   D. An enterprise that does not want to sacrifice security or make changes to their management practices but needs additional resources for test and development of new solutions.

8. Which of the following statements are true about the Public Cloud model?

   A. It meets security and auditing requirements for highly regulated industries.
   B. Resources and infrastructure are managed and maintained by the enterprise IT operations staff.
   C. It shifts the bulk of the costs from capital expenditures and IT infrastructure investment to an utility operating expense model.
   D. It shifts the bulk of the costs from capital expenditures to create a virtualized and elastic infrastructure within the enterprise data center.
   E. Resources are dynamically provisioned on a self-service basis from an off-site third-party provider who shares resources in a multi-tenanted infrastructure.

9. A company would like to leverage Cloud Computing to provide advanced collaboration services (i.e., video, chat, and web conferences) for its employees but does not have the IT resources to deploy such an infrastructure. Which Cloud Computing model would best fit the company needs?

A. Hybrid Cloud
B. Public Cloud
C. Private Cloud
D. Virtual Private Cloud

10. A company has decided to leverage the Web conferencing services provided by a Cloud provider and to pay for those services as they are used. The Cloud provider manages the infrastructure and any application upgrades. This is an example of what type of Cloud delivery model?

A. Platform as a Service
B. Software as a Service
C. Application as a Service
D. Infrastructure as a Service

11. Which statement best describes the Software as a Service Cloud delivery model?

A. A virtual machine provisioned and provided from the Cloud which allows the customer to deploy custom applications.
B. A multi-tenant storage service provisioned from the Cloud which allows the customer to leverage the Cloud for storing software data.
C. A solution stack or set of middleware delivered to the client from the Cloud which provides services for the design, development, and testing of industry aligned applications.
D. An application delivered to the client from the Cloud which eliminates the need to install and run the application on the customer's own computers and simplifying maintenance and support.

12. Which statement is true about the platform as a Service Cloud Computing delivery model?

A. It provides a virtual machine and storage so that computing platforms can be created.
B. It provides a runtime environment for applications and includes a set of basic services such as storage and databases.
C. It provides the entire infrastructure along with a completed application that is accessible using a Web-based front end.
D. It is required by the infrastructure as a service delivery model so that end-user applications can be delivered on the Cloud.

13. What is one benefit of Cloud Computing?

A. Computer resources can be quickly provisioned.
B. A workload can quickly move to a Cloud Computing environment.
C. There is no operational cost for a Cloud Computing environment.
D. The resources can quickly move from one Cloud environment to another.

14. What is one benefit of a Cloud Computing environment?

    A. It improves server performance.
    B. It minimizes network traffic to the virtual machines.
    C. It automatically transforms physical servers into virtual machines.
    D. It maximizes server utilization by implementing automated provisioning.

15. Which statement is true about the maintenance of a Cloud Computing environment?

    A. In a Software as a Service (SaaS) environment, patches are automatically installed on the clients.
    B. In a SaaS environment, customers do not need to worry about installing patches in the virtual instances.
    C. In an Infrastructure as a Service (IaaS) environment, patches are automatically installed on the clients.
    D. In an IaaS environment, customers do not need to worry about installing patches in the virtual instances.

16. What are two common areas of concern often expressed by customers when proposing a multi-tenancy software solution compared to a single-tenancy solution? (Choose two.)

    A. Data privacy
    B. Higher costs
    C. Greater network latency
    D. Complexity to customize solution
    E. Less efficient use of computing resources

17. What is the role of virtualization in Cloud Computing?

    A. It removes operating system inefficiencies.
    B. It improves the performance of Web applications.
    C. It optimizes the utilization of computing resources.
    D. It adds extra load to the underlying physical infrastructure and has no role in Cloud Computing.

18. A company currently experiences 7–10% utilization of its development and test computing resources. The company would like to consolidate to reduce the number of total resources in their data center and decrease energy costs. Which feature of Cloud Computing allows resource consolidation?

    A. Automation
    B. Elasticity
    C. Provisioning
    D. Virtualization

19. A company operates data centers in two different Regions. Energy costs for one of the data centers increase during the warmer, summer months. The company already uses server virtualization techniques in order to consolidate the total number of required resources. How might the company further reduce operating costs at this data center?

    A. The company can shut down the data center in the summer season.
    B. The company does not need to do anything because they are already using server virtualization techniques.
    C. The company can leverage provisioning to optimize the availability of their environments in the summer season.
    D. The company can further leverage virtualization to easily and quickly move as many assets from the data center in the warmer Region to the data center in the cooler Region during the summer season.

20. A company has peak customer demand for its IT services in the month of April. It has enough IT resources to handle off-peak demand but not peak load. What is the best approach to handle this situation?

    A. Outsource all of IT services to a Cloud-based provider.
    B. Virtualize its servers and implement a Cloud-based system.
    C. Utilize an external Cloud Service Provider to handle the peak load.
    D. Acquire additional IT resources and design the system to handle the peak load.

Detailed Questions

1. An ELB is deployed in AWS, which has a public IP of 98.24.20.10. Two EC2 instances are behind this ELB, each with Public IP of 98.20.10.10 and 98.10.10.12. You have traffic from West Coast and East Coast coming to your ELB. Add comments for each of the following options if they will work, and if not then why not?

    A. You decide to move one EC2 in one of the East Coast Regions and another one in a West Coast Region and put both the Regions behind the ELB. (Circle one). Works does not work.
    B. You decide to create two availability zones and migrate an EC2 to each availability zone with the ELB front ending it. (Circle one). Works does not work.
    C. You decide to deploy an EC2 instance and install your own Load Balancer (like an open-source load balancer) and deploy EC2 to each availability zones. Thus eliminating ELB from AWS.
    D. What Service Model does ELB fall under and why?

2. What is the availability of a Cloud system (Web + DB + LB + Firewall + Data Center + ISP), where the availability of each component is 99.9%/year.

- How much downtime can you expect in a year with such a system?
- Describe the steps and a diagram of your new setup that gets you 99.95% availability a year.

3. Draw the block diagram of ELB, 4 EC2 instances and S3, where the application running on EC2 requires 8 GB setup minimum.
   The S3 instance needs a setup of 20 GB to handle all the external sources. Assume the cost of EC2 instance to be $0.10 an hour for 5 GB, $0.15 for 10 GB, S3 to be $0.10 an hour for 10 GB, and ELB to be $0.10 an hour.
   Cost the monthly price for ELB + 4 EC2 (8 GB) instances + S3 (4 points).
4. How will you make above setup more reliable and redundant in the most economical way and how much extra will it cost? AS group is not an option.
5. What is a noisy neighbor problem in Cloud Computing?

Answer Key

1. *B. Private Cloud.*
2. *C. Software as a Service (SaaS).*
3. *False.*
4. *A. Scalability.*
5. *True.*
6. *C. On-demand self-service.*
7. *D. An enterprise that does not want to sacrifice security or make changes to their management practices but needs additional resources for test and development of new solutions.*
8. *C. It shifts the bulk of the costs from capital expenditures and IT infrastructure investment to an utility operating expense model.*
   E. Resources are dynamically provisioned on a self-service basis from an off-site third-party provider who shares resources in a multi-tenanted infrastructure.
9. *B. Public Cloud.*
10. *B. Software as a Service.*
11. *D. An application delivered to the client from the Cloud which eliminates the need to install and run the application on the customer's own computers and simplifying maintenance and support.*
12. *B. It provides a runtime environment for applications and includes a set of basic services such as storage and databases.*
13. *A. Computer resources can be quickly provisioned.*
14. *D. It maximizes server utilization by implementing automated provisioning.*
15. *B. In a SaaS environment, customers do not need to worry about installing patches in the virtual instances.*
16. *A. data privacy*
    D. Complexity to customize solution.
17. *C. It optimizes the utilization of computing resources.*
18. *D. Virtualization.*

19. *D. The company can further leverage virtualization to easily and quickly move as many assets from the data center in the warmer Region to the data center in the cooler Region during the summer season.*
20. *C. Utilize an external Cloud Service Provider to handle the peak load.*

Detailed Questions

(1) *A. Does not Work.*
Key here is to know if an Elastic Load Balancer (ELB) can cover across Regions, and answer is NO. It can cover across Availability Zones (AZs) but not Regions as they may be thousands of miles away
B. Works
Yes, for the above reason, an ELB can manage instances within an AZ and across AZs that are located within a Region.
C. Works.
Yes, but you will have a lot of work to do, for doing health checks and making sure that your own load balancer knows how to distribute traffic across other instances.
D. IaaS: because it is still giving access to one instance or another, but though a layer of indirection.

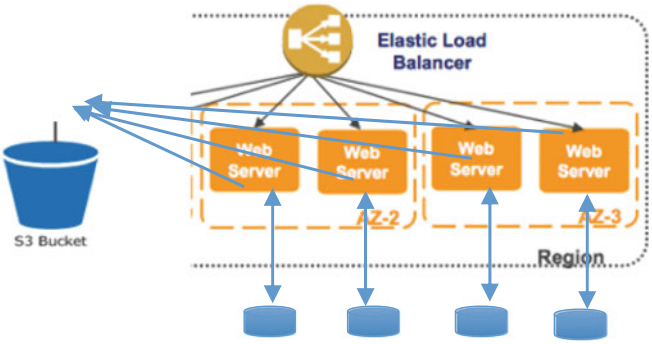
(2) $$\textit{Total Availability} = 0.999 * 0.999 * 0.999 * 0.999 * 0.999 * 0.999 = 0.994 = 99.4\%$$

$$(100 - 99.4) * 365.25 * 24 = 52.46469\ \textit{hrs}\ (2\ \textit{Days and}\ 4.56\ \textit{hours})$$

Double up on everything except the data-center, so the availability of each step becomes six 9's, then their multiplication will give you 99.95% total availability.

Web	0.999		0.999999
DB	0.999		0.999999
LB	0.999		0.999999
Firewall	0.999		0.999999
DC	0.999		0.999
ISP	0.999		0.999999
Total (%)	99.401%	52.46469	99.900%

(3)

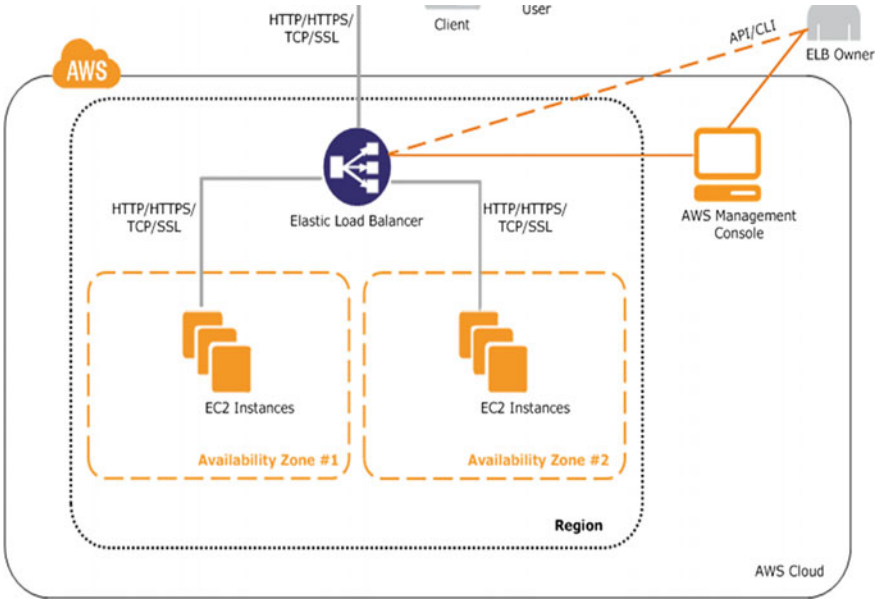


(4)

	5 GB	10 GB		1 h	Month
EC2	$0.10	$0.15		$0.60	
S3		$0.10		$0.20	
ELB		$0.10		$0.10	
total				$0.90	$648.00

If no auto-scaling, then extra instances need to be up and running all the time. S3 already has a lot of redundancy, so no need, but you do need four extra instances, preferably in a different AZ, doubling the cost of EC2 line item. Rest remains the same. Extra cost = $1080 − $684 = $396/month

	5 GB	10 GB		1 h	Month
EC2	$0.10	$0.15		$1.20	
S3		$0.10		$0.20	
ELB		$0.10		$0.10	
total				$1.50	$1080.00



(5) *In a multi-tenanted environment, several users are sharing the same hardware. Virtualization can isolate individual Virtual Machines (VMs), but if one of the users does excessive I/O on the disk, memory, Network Interface Card (NIC) or any other shared HW component then other users will see a drop in their available bandwidth. This will slow down their jobs, aka noisy neighbor problem.*

Additional Challenging questions:

(6) What are the PaaS or SaaS components that Cloud vendors provide and how do they compare with similar offerings from AWS? (For instance, Vmware's Cloud is big on Disaster Recovery (DR), but what does AWS provide?). Pick one of the PaaS or SaaS offering from an alternative vendor and compare it to AWS, in terms of features and price.

(7) What would be the cost of the following configuration for the Cloud Alternative vendor when run 24 * 7 for a whole year? Compare the cost to what AWS provides using Reserved Instances or Spot Prices, with upfront payments. For the following configuration, what would the lowest cost possible on AWS and on the Cloud Alternative vendor and how?

1 Load Balancer
4 EC2 instances
2 DB instances

(8) Compare the security offering of the Cloud Alternative vendor to AWS. Compare VPC and other ways that AWS provides security protection to what Azure, Google, or VMware provide. Which one is more secure and why? Give references, screen shots, etc., to support your answer.
(9) How is Docker supported in the Alternative Cloud vendor? If so compare to AWS support for Docker. Share some pros and cons between what AWS has and what the vendor provides?
(10) Research and compare the uptime (in 4 or 5 nines) and SLA offered by AWS with your Cloud Alternative vendor. How does the Cloud Alternative vendor report downtimes? If the Alternative Cloud Vendor has data centers around the globe, compare the global data center presence and the uptime for the various data centers around the globe.
(11) If you had to describe one really Cool feature of the Alternative Cloud Vendor what would that be and does AWS have anything like that or not?
(12) How does an on-premise computing stack help customers with multiple and distinct business divisions?

Chapter 14
Hands-On Project to Use Cloud Service Provider

Open an account on Amazon EC2, or Microsoft Azure or Google's GCP Clouds. For the first-time account holder, these Cloud Service Providers (CSP) will give you free access to use their facilities for a limited time.

(a) **Project 1**: Install LAMP stack (consisting of Linux, Apache Server, MySQL, and PHP) on your CSP account.
(b) **Project 2**: Setup Wordpress and PHP services for other users to access your blog.
(c) **Project 3**: Enhance security of your Cloud Server.
(d) **Project 4**: Setup load balancer to manage multiple users.
(e) **Project 5**: Use Elastic IP for your Cloud instances, so if multiple servers are spawned, users can access any of them with the same IP address.
(f) **Bonus**: Setup auto-scaling to increase or decrease the number of servers as number of users vary.

Steps below are shown for illustration purposes only:

14.1 Project 1: Install Lamp Stack on Amazon EC2

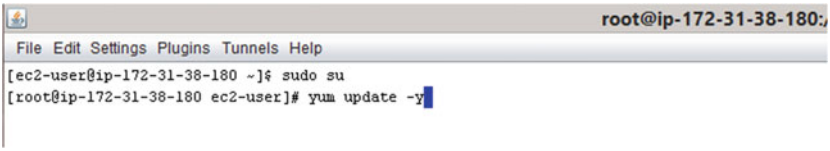
14.1.1 Installing Lamp Web Server on AWS via EC2

(Instance used for this: Instance 1—i-9adc516c)

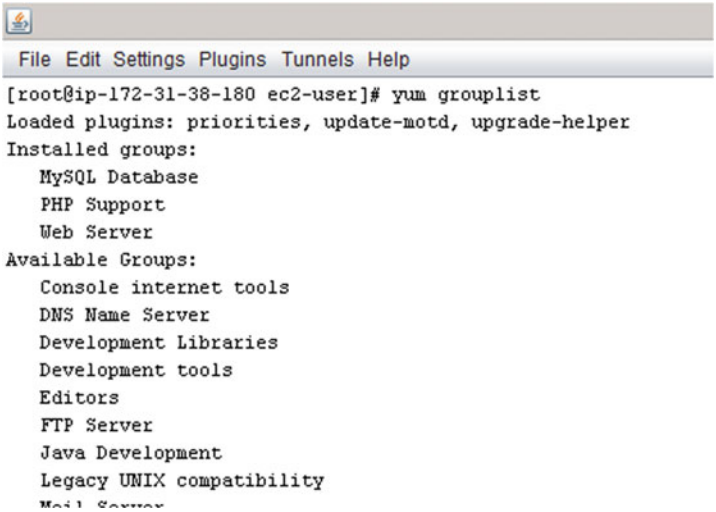
1. Launch the instance.

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6_14

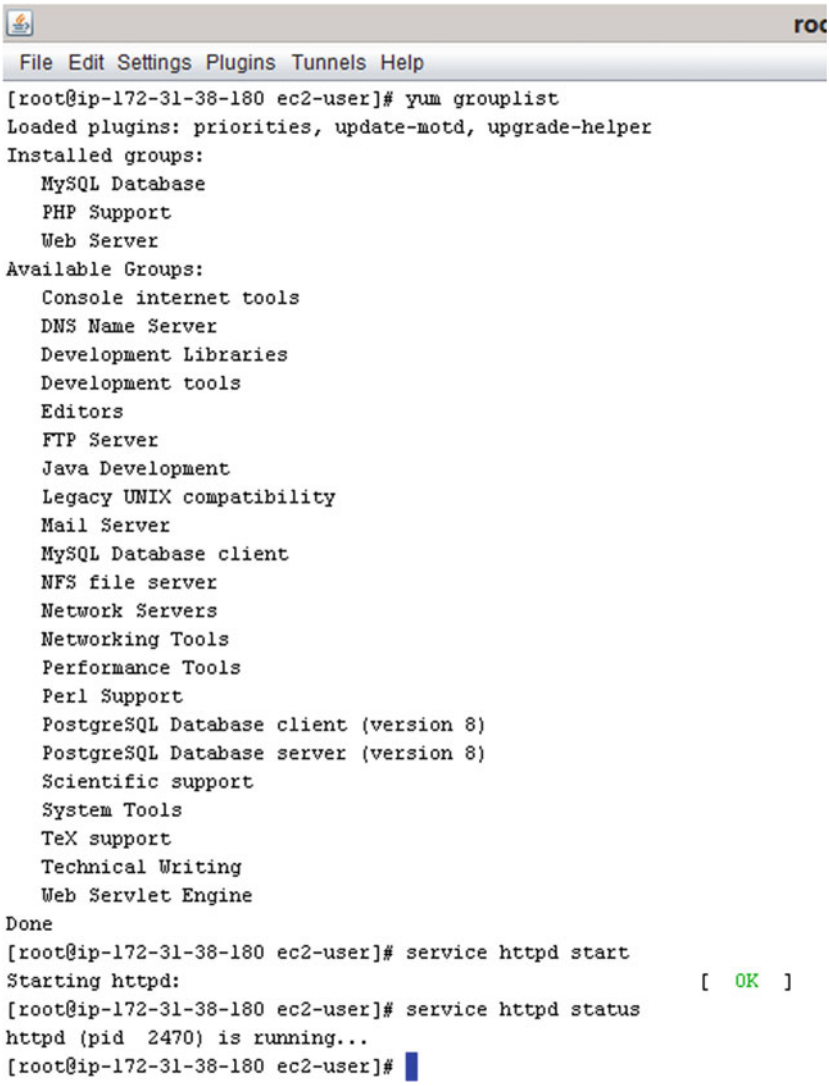
2. Perform a software update.



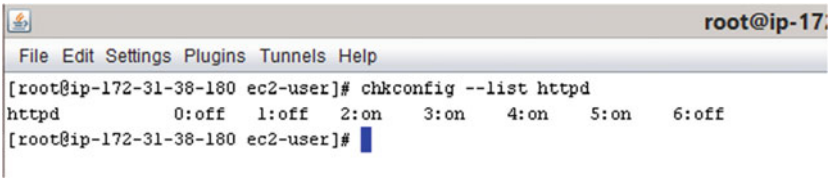
3. Group install the Web server, PHP support, and MySQL Database using the yum groupinstall command. Check to see if they have been properly installed.



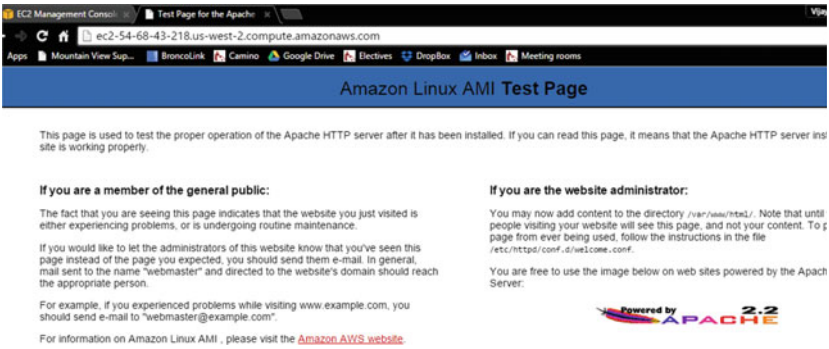
4. Start the Apache Web server.



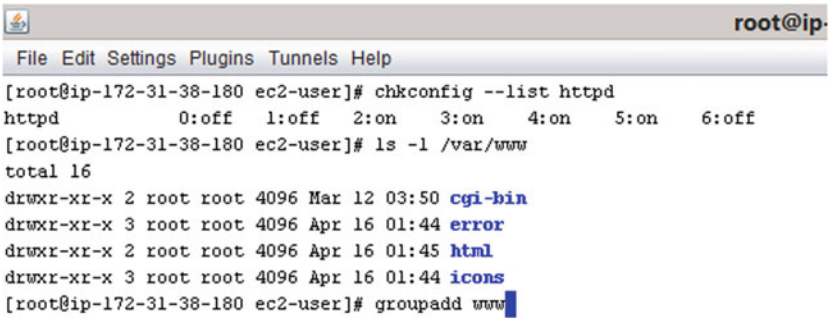
5. Ensure that the apache server remains switched on with every reboot.



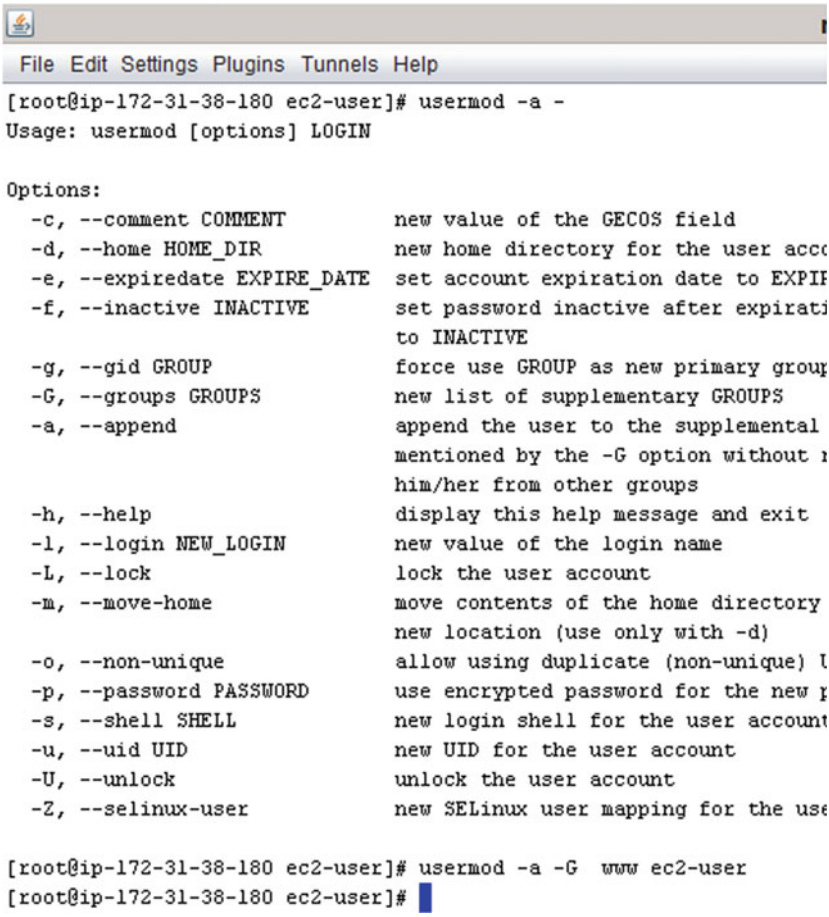
6. Copy the Public DNS link for this instance from the AWS console onto a browser tab, and click enter.
   We should be able to view the Amazon Linux AMI test page.



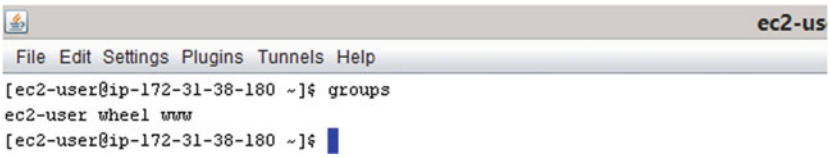
7. Add the www group to the current instance.



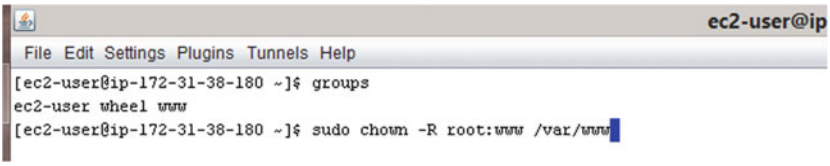
8. Add the ec2-user to this group.



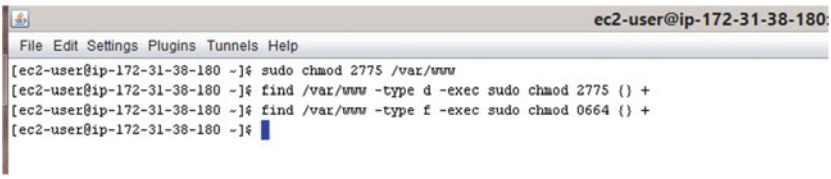
9. Log out and log in again to view membership.



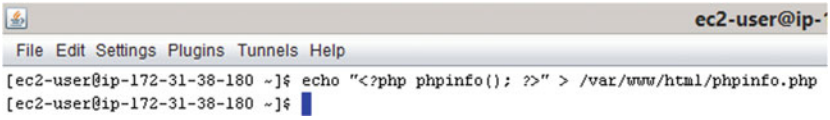
10. Changing group ownership of the /var/www folder to www group.



11. Change the directory and subdirectory permissions to add group write. Also, set a group id on future subdirectories. Also, change permissions recursively of /var/www and subdirectories to add group write permissions.

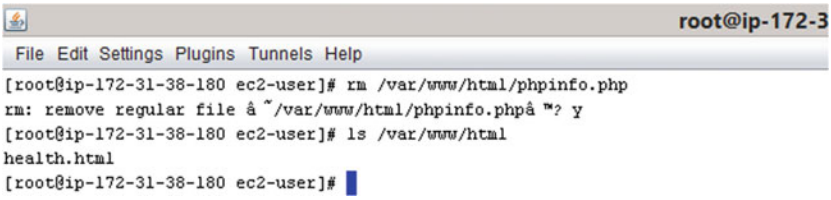


12. Create a php file in the Apache document root.

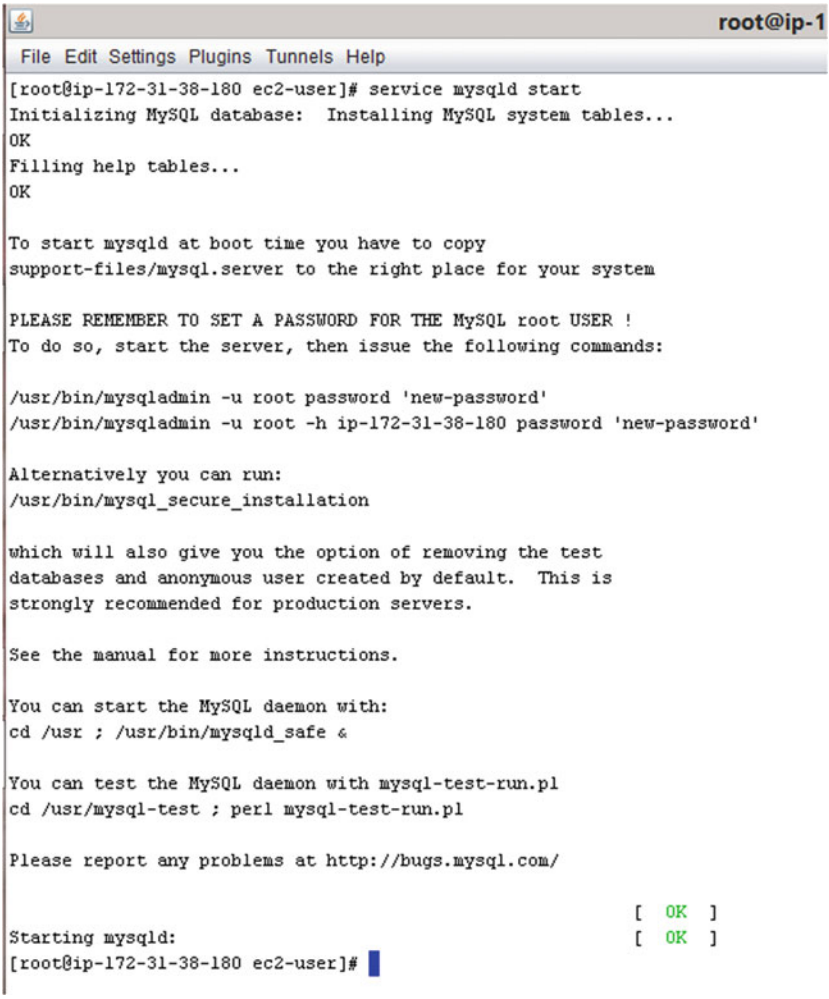


13. Type the Public DNS name followed by /phpinfo.php to view the following through a browser.

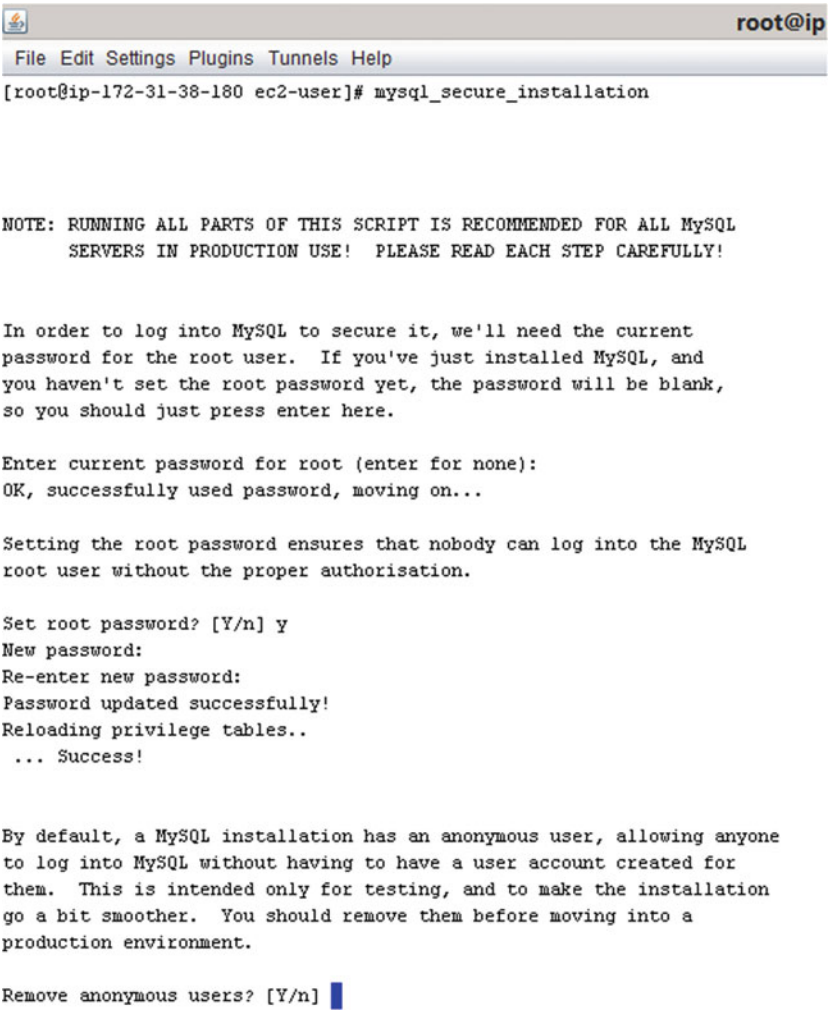
14. Delete the phpinfp.php file for security reasons.

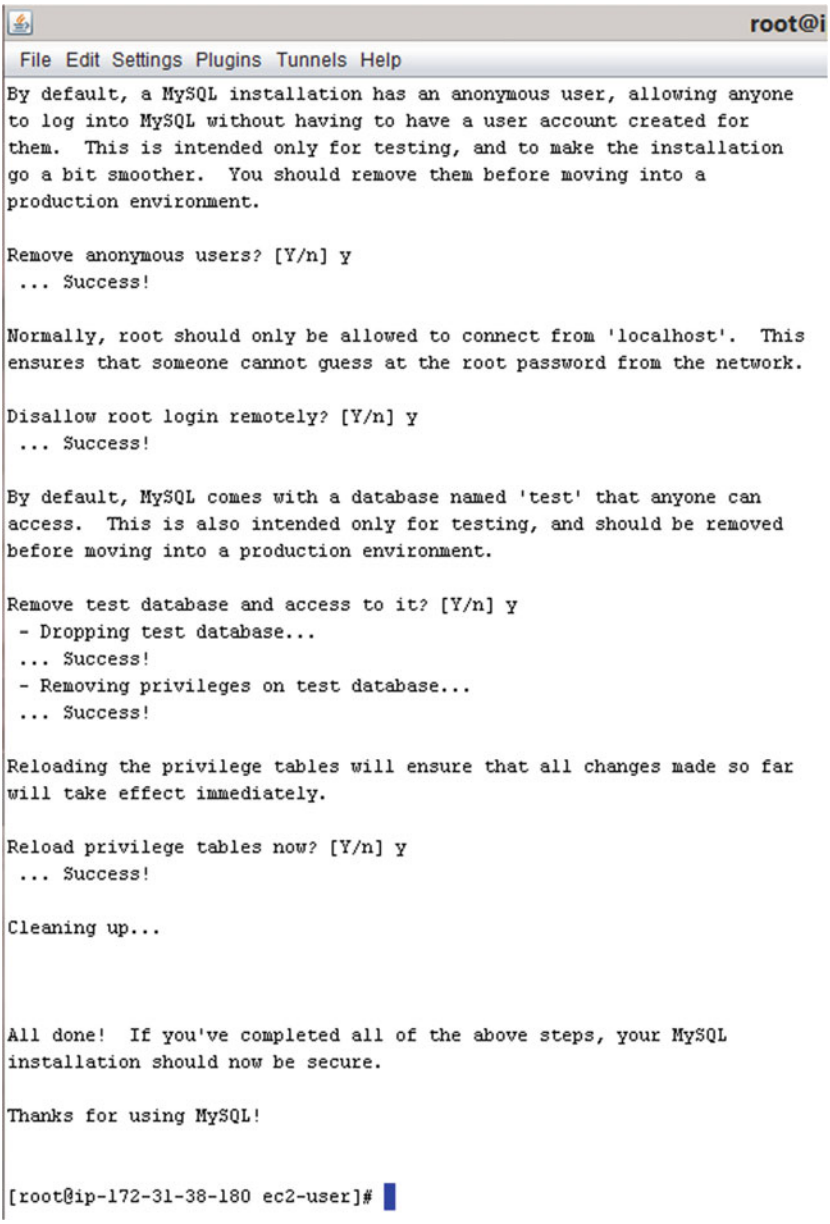


15. Start the mysql server to run mysql_secure_installation.

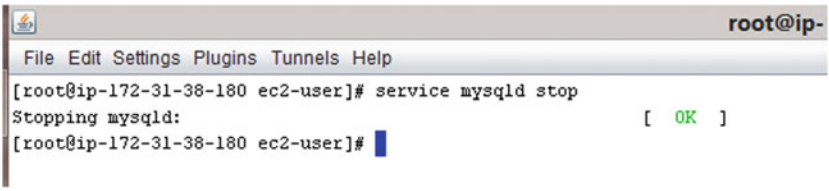


16. Set a strong password.




root@i
File Edit Settings Plugins Tunnels Help
By default, a MySQL installation has an anonymous user, allowing anyone
to log into MySQL without having to have a user account created for
them. This is intended only for testing, and to make the installation
go a bit smoother. You should remove them before moving into a
production environment.
Remove anonymous users? [Y/n] y
... Success!
Normally, root should only be allowed to connect from 'localhost'. This
ensures that someone cannot guess at the root password from the network.
Disallow root login remotely? [Y/n] y
... Success!
By default, MySQL comes with a database named 'test' that anyone can
access. This is also intended only for testing, and should be removed
before moving into a production environment.
Remove test database and access to it? [Y/n] y
- Dropping test database...
... Success!
- Removing privileges on test database...
... Success!
Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.
Reload privilege tables now? [Y/n] y
... Success!
Cleaning up...
All done! If you've completed all of the above steps, your MySQL
installation should now be secure.
Thanks for using MySQL!
[root@ip-172-31-38-180 ec2-user]#

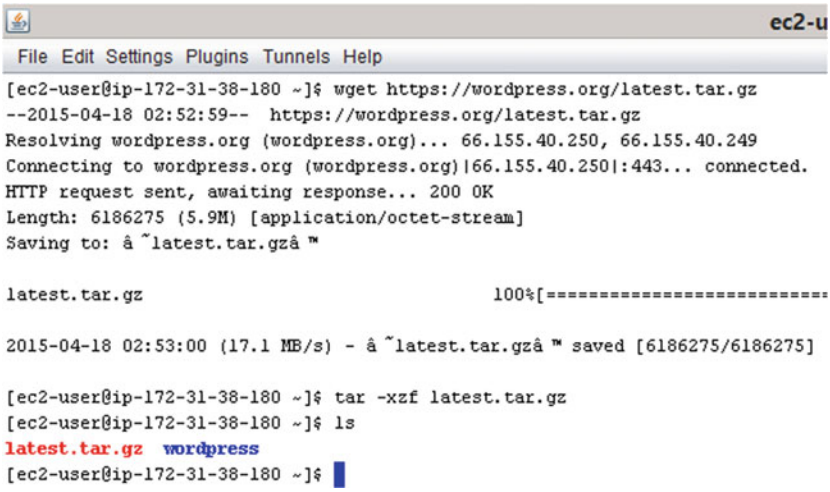
17. Stopping the mysql server as it will not be used for now.



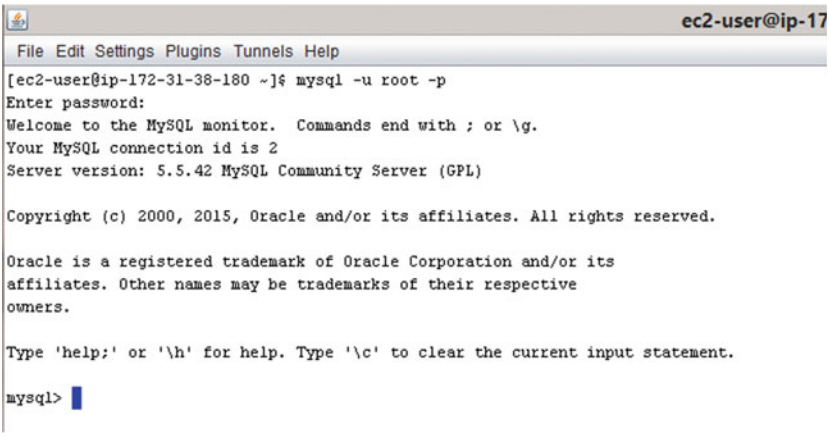
14.1.2 Installing Wordpress

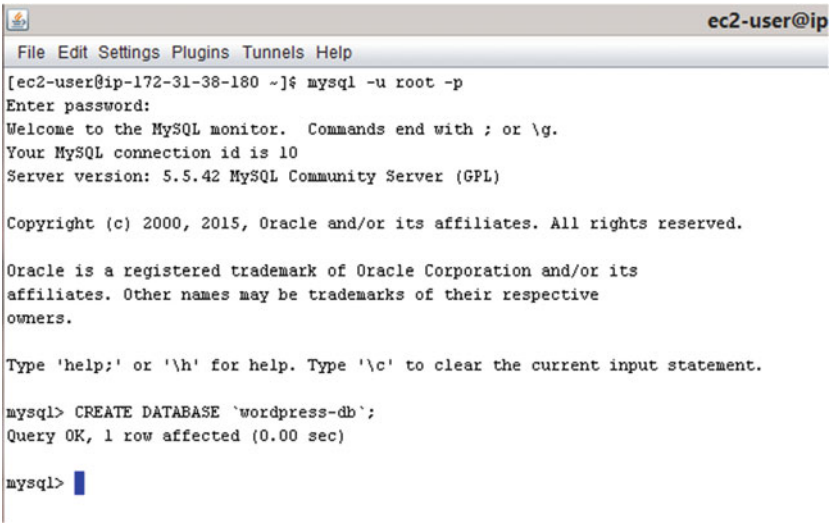
1. Download the Wordpress.tar file.

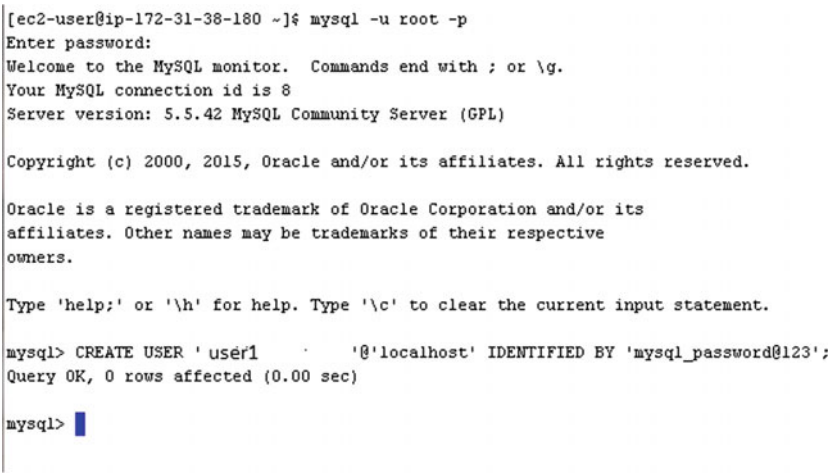
2. Unzip this tar file.



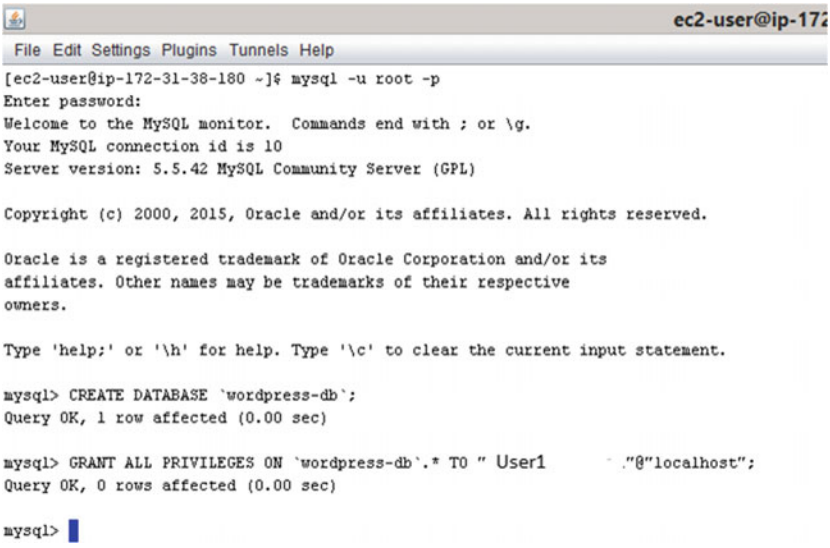
3. Create a mysql database and user.




ec2-user@ip
File Edit Settings Plugins Tunnels Help
[ec2-user@ip-172-31-38-180 ~]$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 5.5.42 MySQL Community Server (GPL)
Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> CREATE DATABASE `wordpress-db`;
Query OK, 1 row affected (0.00 sec)
mysql>

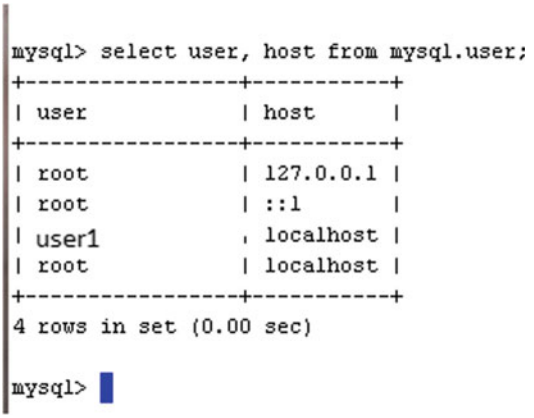

[ec2-user@ip-172-31-38-180 ~]$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 5.5.42 MySQL Community Server (GPL)
Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> CREATE USER 'user1 '@'localhost' IDENTIFIED BY 'mysql_password@123';
Query OK, 0 rows affected (0.00 sec)
mysql>

4. Grant privileges to the database for the wordpress user.

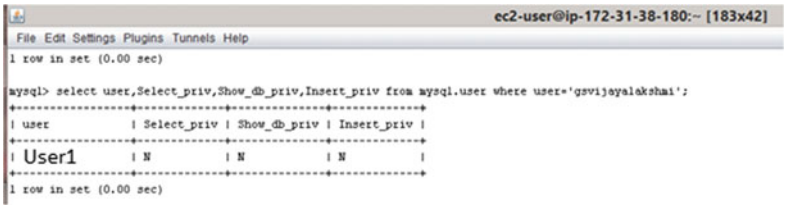


5. Screenshots of list of databases, user, and privileges on the database that would be used for Wordpress.

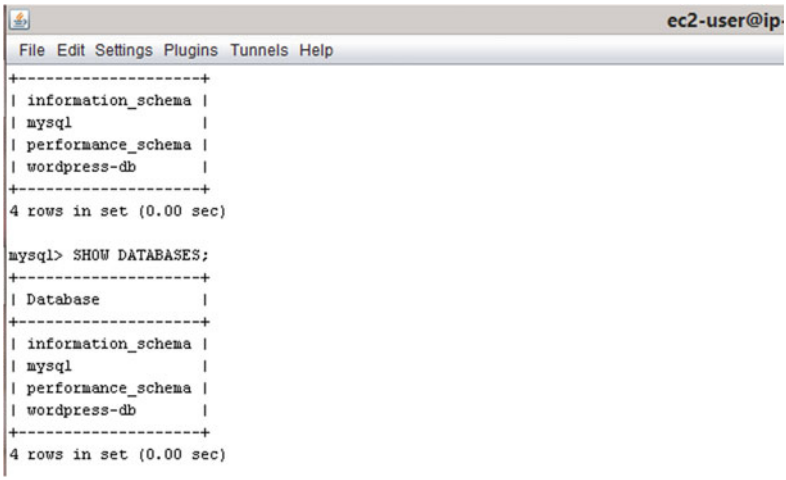
- View all users



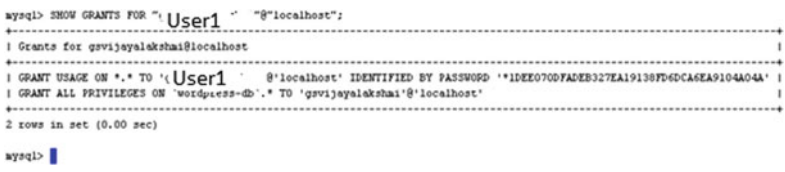
- View contents of the mysql.user table



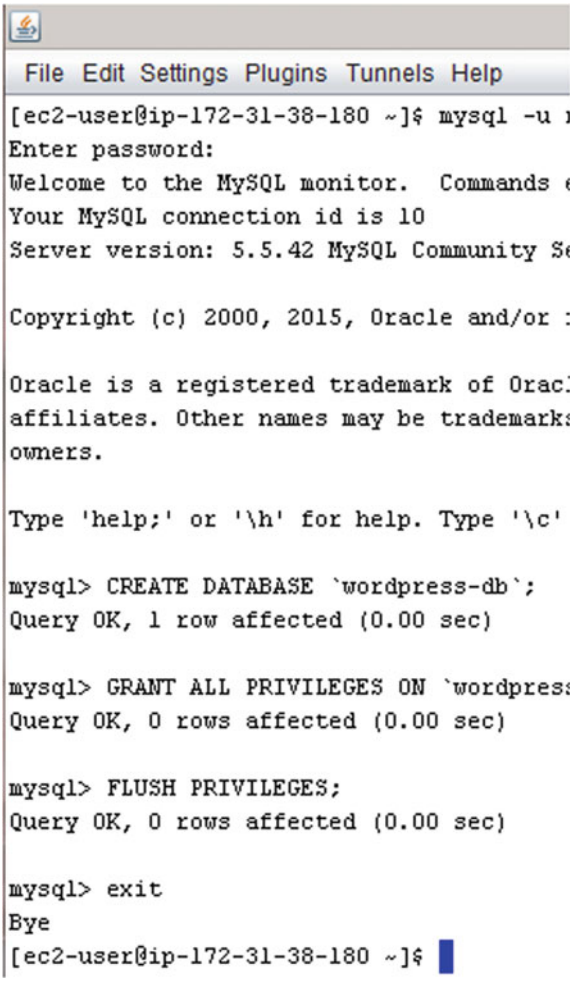
- View all the databases created



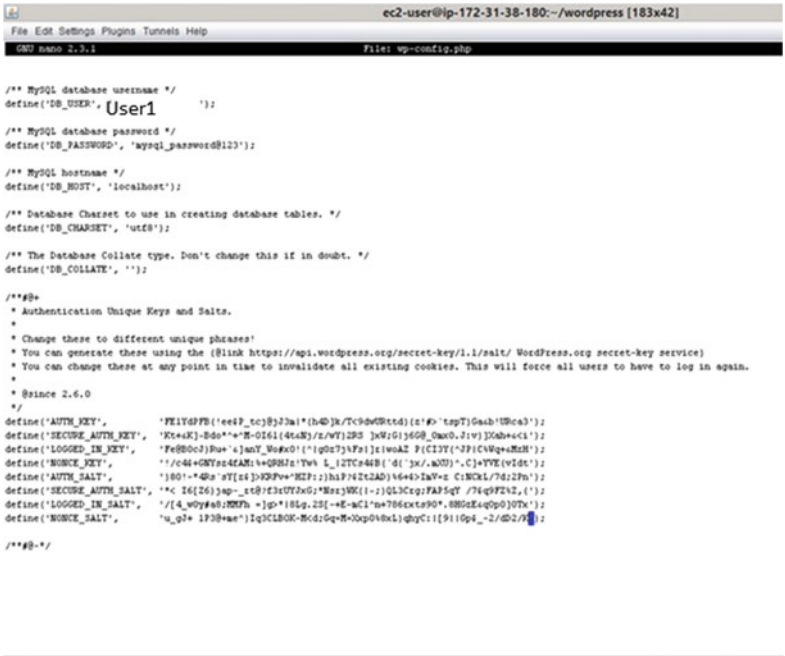
- View the privileges on the newly created database

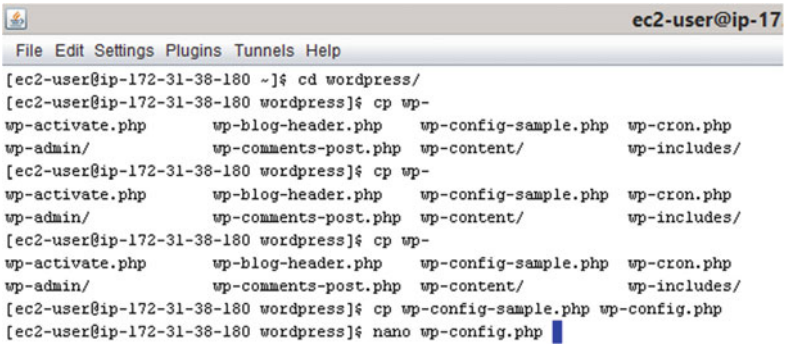


6. Exit mysql.

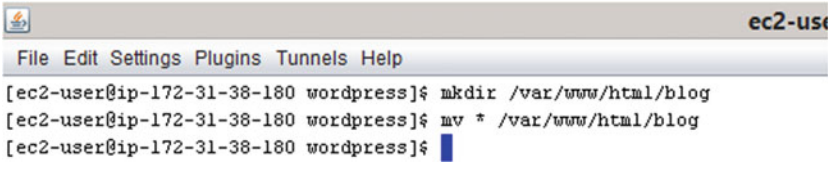

File Edit Settings Plugins Tunnels Help
[ec2-user@ip-172-31-38-180 ~]$ mysql -u
Enter password:
Welcome to the MySQL monitor. Commands
Your MySQL connection id is 10
Server version: 5.5.42 MySQL Community Se
Copyright (c) 2000, 2015, Oracle and/or
Oracle is a registered trademark of Orac
affiliates. Other names may be trademark
owners.
Type 'help;' or '\h' for help. Type '\c'
mysql> CREATE DATABASE `wordpress-db`;
Query OK, 1 row affected (0.00 sec)
mysql> GRANT ALL PRIVILEGES ON `wordpres
Query OK, 0 rows affected (0.00 sec)
mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.00 sec)
mysql> exit
Bye
[ec2-user@ip-172-31-38-180 ~]$

7. Create and edit the wp-config.php file.

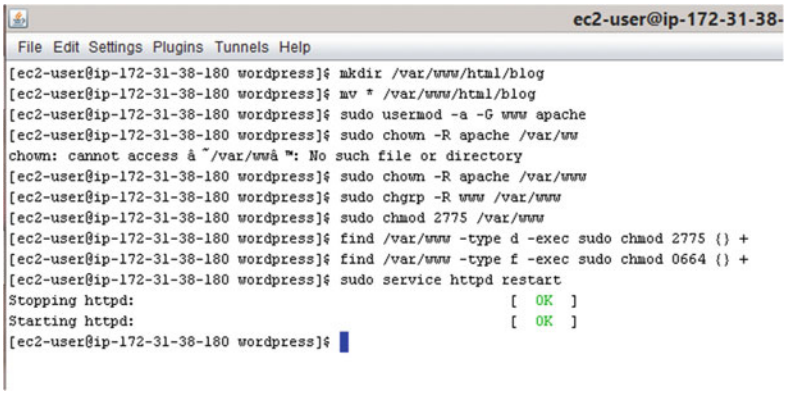




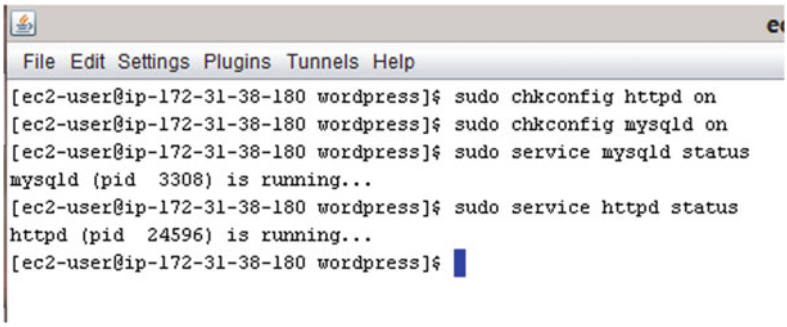
8. Create a directory called blog and move all Wordpress content into that directory.



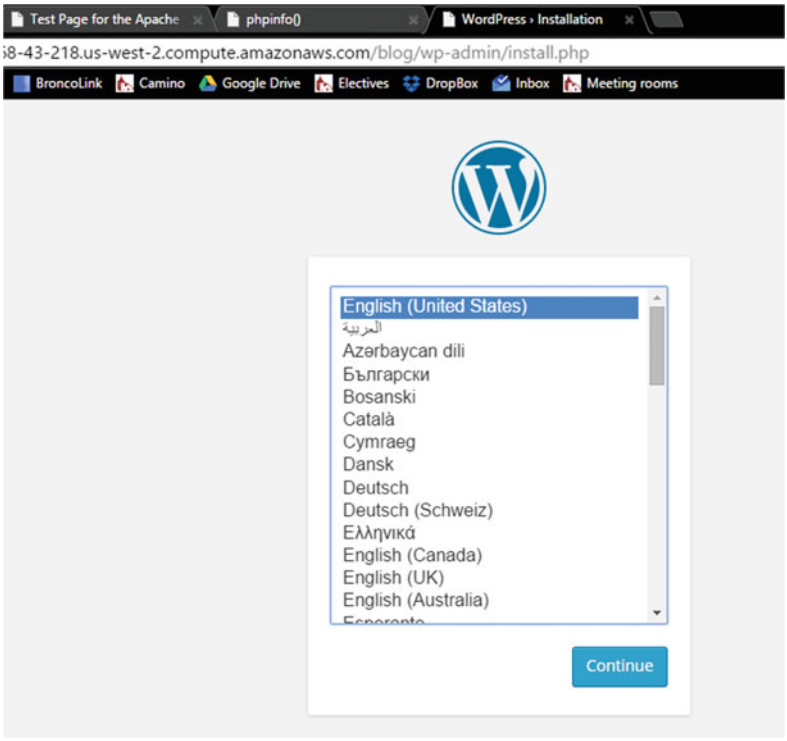
9. Fix the file permissions for the Apache Web server.



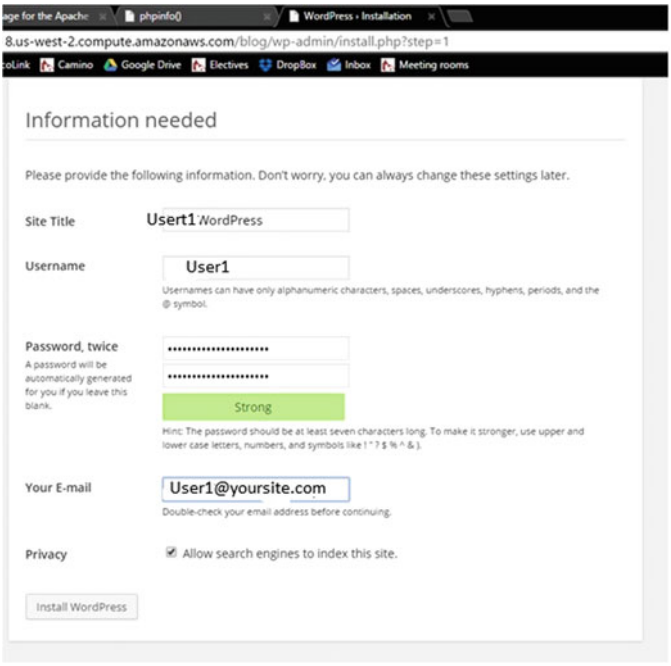
10. Check if all the required services are running.

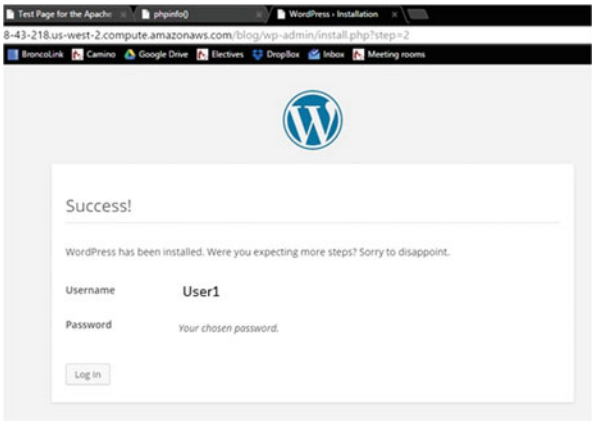


11. Copy paste the public DNS link followed by/blog to access the WP config page.

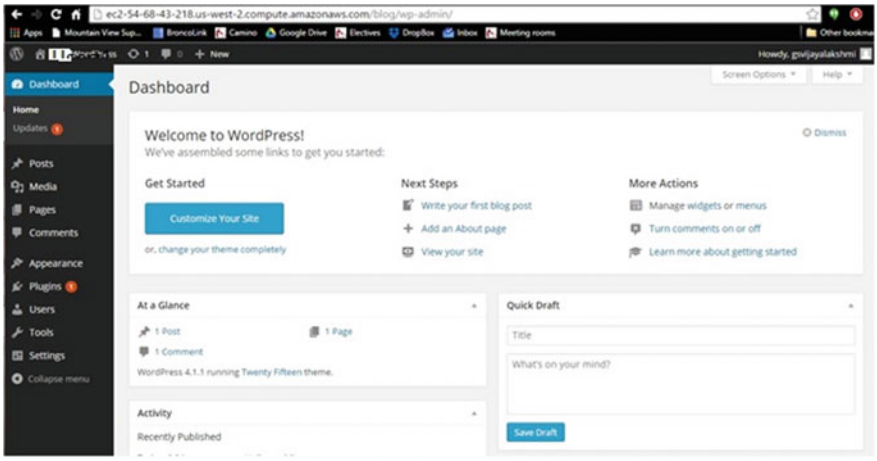


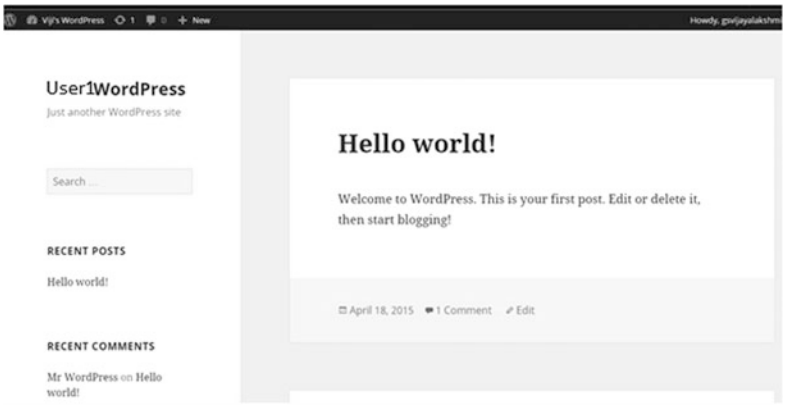
12. Install Wordpress.






Viji's WordPress › Log In
onaws.com/blog/wp-login.php
Electives
DropBox
Inbox
Meeting rooms
Username
User1
Password
Remember Me
Log In
Lost your password?
← Back to Viji's WordPress


ec2-54-68-43-218.us-west-2.compute.amazonaws.com/blog/wp-admin/
Howdy, gsvijayalakshmi
Dashboard
Home
Updates
Posts
Media
Pages
Comments
Appearance
Plugins
Users
Tools
Settings
Collapse menu
Screen Options
Help
Welcome to WordPress!
We've assembled some links to get you started:
Dismiss
Get Started
Customize Your Site
or, change your theme completely
Next Steps
Write your first blog post
Add an About page
View your site
More Actions
Manage widgets or menus
Turn comments on or off
Learn more about getting started
At a Glance
1 Post
1 Page
1 Comment
WordPress 4.1.1 running Twenty Fifteen theme.
Quick Draft
Title
What's on your mind?
Save Draft
Activity
Recently Published


Vij's WordPress
New
Howdy, gsvijayalakshmi
User1WordPress
Just another WordPress site
Search ...
RECENT POSTS
Hello world!
RECENT COMMENTS
Mr WordPress on Hello world!
Hello world!
Welcome to WordPress. This is your first post. Edit or delete it, then start blogging!
April 18, 2015
1 Comment
Edit

14.1.3 Wordpress URL

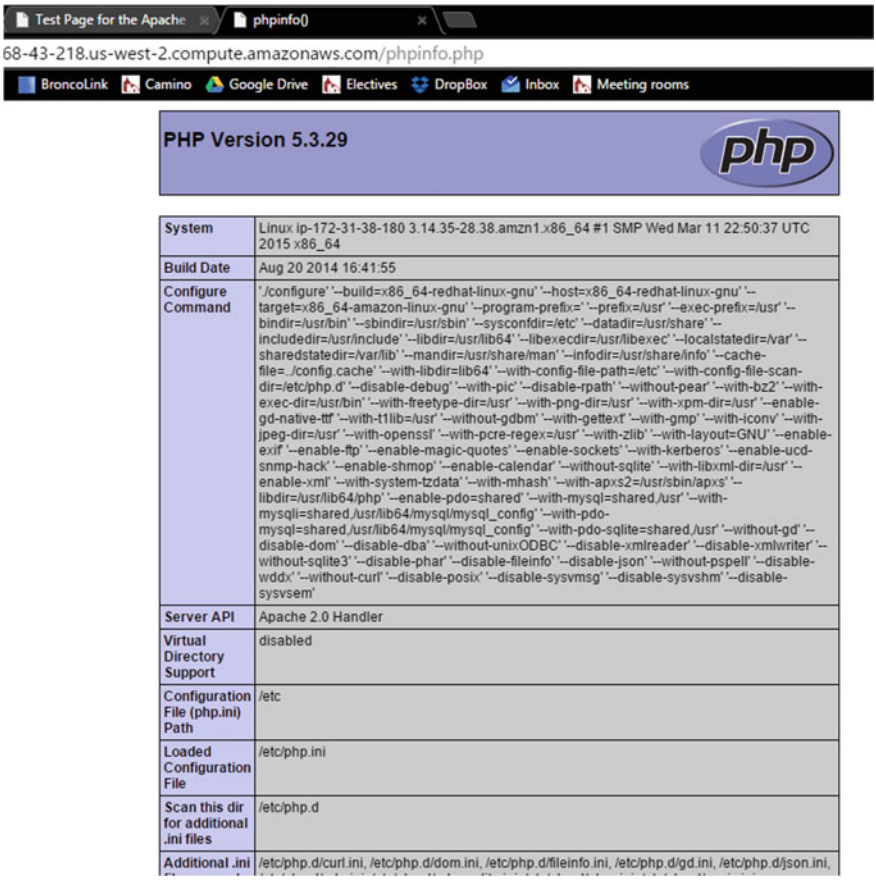
Wordpress Link:
http://ec2-54-186-40-103.us-west-2.compute.amazonaws.com/blog/
Uname:
xyz
Password:
abc

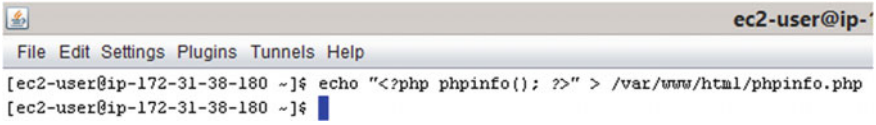
14.2 Project 2: Install PHP on Your AWS Instance

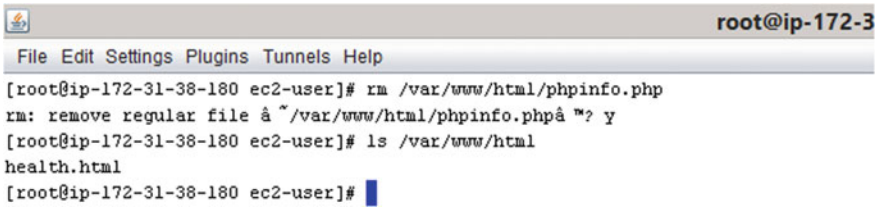
Generally, the "phpinfo.php" file is disabled because it consists of information about the server where our site is currently running on. This file is useful for debugging purposes, but its removal makes the server more secure. It would tell hackers what versions of software are running in the server, thereby enabling hack attacks.

The phpinfo() function is majorly used to check for configuration settings, available system variables, and to test if the PHP installation was successful or not. This function is contained within the phpinfo.php file. Sometimes, owners forget to remove this file and, thereby, expose information about physical paths in the system, environment variables, and the PHP settings.

The best work-around for ensuring our servers do not get hacked via the phpinfo.php file is to ensure that this file is not used on a production server. It is okay to use it on a development server that has a restricted member access. Also, as an additional step, we can disallow other IPs from accessing it. It is important that we give an obscure name to this file or maybe just delete it after use (e.g., delete this file after Web site development and testing are done).


Test Page for the Apache
phpinfo()
68-43-218.us-west-2.compute.amazonaws.com/phpinfo.php
BroncoLink
Camino
Google Drive
Electives
DropBox
Inbox
Meeting rooms
PHP Version 5.3.29
php
System
Linux ip-172-31-38-180 3.14.35-28.38.amzn1.x86_64 #1 SMP Wed Mar 11 22:50:37 UTC 2015 x86_64
Build Date
Aug 20 2014 16:41:55
Configure Command
'./configure' '--build=x86_64-redhat-linux-gnu' '--host=x86_64-redhat-linux-gnu' '--target=x86_64-amazon-linux-gnu' '--program-prefix=' '--prefix=/usr' '--exec-prefix=/usr' '--bindir=/usr/bin' '--sbindir=/usr/sbin' '--sysconfdir=/etc' '--datadir=/usr/share' '--includedir=/usr/include' '--libdir=/usr/lib64' '--libexecdir=/usr/libexec' '--localstatedir=/var' '--sharedstatedir=/var/lib' '--mandir=/usr/share/man' '--infodir=/usr/share/info' '--cache-file=../config.cache' '--with-libdir=lib64' '--with-config-file-path=/etc' '--with-config-file-scan-dir=/etc/php.d' '--disable-debug' '--with-pic' '--disable-rpath' '--without-pear' '--with-bz2' '--with-exec-dir=/usr/bin' '--with-freetype-dir=/usr' '--with-png-dir=/usr' '--with-xpm-dir=/usr' '--enable-gd-native-ttf' '--with-t1lib=/usr' '--without-gdbm' '--with-gettext' '--with-gmp' '--with-iconv' '--with-jpeg-dir=/usr' '--with-openssl' '--with-pcre-regex=/usr' '--with-zlib' '--with-layout=GNU' '--enable-exif' '--enable-ftp' '--enable-magic-quotes' '--enable-sockets' '--with-kerberos' '--enable-ucd-snmp-hack' '--enable-shmop' '--enable-calendar' '--without-sqlite' '--with-libxml-dir=/usr' '--enable-xml' '--with-system-tzdata' '--with-mhash' '--with-apxs2=/usr/sbin/apxs' '--libdir=/usr/lib64/php' '--enable-pdo=shared' '--with-mysql=shared,/usr' '--with-mysqli=shared,/usr/lib64/mysql/mysql_config' '--with-pdo-mysql=shared,/usr/lib64/mysql/mysql_config' '--with-pdo-sqlite=shared,/usr' '--without-gd' '--disable-dom' '--disable-dba' '--without-unixODBC' '--disable-xmlreader' '--disable-xmlwriter' '--without-sqlite3' '--disable-phar' '--disable-fileinfo' '--disable-json' '--without-pspell' '--disable-wddx' '--without-curl' '--disable-posix' '--disable-sysvmsg' '--disable-sysvshm' '--disable-sysvsem'
Server API
Apache 2.0 Handler
Virtual Directory Support
disabled
Configuration File (php.ini) Path
/etc
Loaded Configuration File
/etc/php.ini
Scan this dir for additional .ini files
/etc/php.d
Additional .ini
/etc/php.d/curl.ini, /etc/php.d/dom.ini, /etc/php.d/fileinfo.ini, /etc/php.d/gd.ini, /etc/php.d/json.ini,


ec2-user@ip-
File Edit Settings Plugins Tunnels Help
[ec2-user@ip-172-31-38-180 ~]$ echo "<?php phpinfo(); ?>" > /var/www/html/phpinfo.php
[ec2-user@ip-172-31-38-180 ~]$



References

http://www.rapid7.com/db/vulnerabilities/http-php-phpinfo-leak
http://securitymecca.com/article/helpful-phpinfo-are-you-putting-welcome-mat-out-for-hackers/

14.3 Project 3: Enhance Security of Your AWS Instance

Amazon Security groups are used to provide functionality similar to a "firewall" to safeguard the EC2 servers. These security groups can be used to filter the incoming traffic, also called ingress, on the basis of the chosen protocol (TCP, ICMP, UDP), an IP address or a range of IP addresses, and even ports. It is important that every instance or server configured through AWS is linked to at least one security group.

By default, all the incoming traffic to a server is rejected till the time that server is attached to a security group and that security group should allow traffic requests of that kind. EC2 security group only handles incoming traffic and does not have control over a server's outgoing data (requests/responses).

Security groups are not specific to availability zones but sync with AWS Regions. Any user can have more than one security group in their account to a maximum to 500. In the event that a user fails/forgets to attach a security group to their instance, AWS attaches a default security group to that instance.

The default security group opens ports and protocols only to servers that are in the default group.

In the case of an EC2 instance acting as a Web server, the security group must allow the HTTP and the HTTPS protocols.

Every server must be mandatorily attached to a security group.

Once the server is launched, security groups cannot be added or removed from it.

Changes made to the ingress rules are applied immediately on the running servers.

It is a best practice to select different security groups for different levels or sections of a project, unless they have similar functionalities.

It is also a good practice to remove unused security groups or to keep them clean to avoid any untoward incidents.

To allow incoming traffic from the same Region (EC2 instance), we need to create a rule with source port = 11.0.0.0/8.

Servers that are connected to different security groups can communicate with one other either via private or public IPs, as long as they are in the same AWS Region. Servers belonging to other AWS Regions communicate through public IPs.

A security group can be added to itself. In this feature, servers use their private IPs for connecting with other servers through any port and protocol. However, in this case, the rules of the security group will apply only to the servers that have a private IP.

When we add one security group (A) to another (B), servers of A can connect with servers of B through their private IPs, through a specific set of ports and protocols.

When a user makes use of EIP (Elastic IP), security groups can be used to allow requests only from this IP to be sent to another server.

VPC: Virtual Private Clouds are dedicated to the user's AWS account. They cater to a single Region but span over multiple availability zones. However, the scope of the subnet in use is just one availability zone.

The user has control over the methods used by instances to access information that are outside of the VPC.

The VPC security group can be used for controlling both incoming (ingress) network traffic and outgoing (egress) network traffic.

We can have separate rules set for the ingress and egress traffic.

It is always a best practice to reserve IP addresses for both instances and subnets for further expansion across availability Regions.

It is always better to place subnets alongside the tiers (e.g., ELB, Web/App).

To have all the data in private subnets by default is another good practice, keeping only the ELB or other filters in public subnets.

IAM should be used for assigning users and also for setting their access privileges.

It is also important to define subnet routes and tables, and VPC security groups.

To sum up, VPC provides extra security and isolates your network.

Differences between security groups and firewall:

- Firewalls control network traffic between subnets of networks or between different networks.
- They are provided by either vendors or open sourced.
- AWS security groups belong to Amazon. They are easier to manage than firewalls.
- Firewalls require manual updates whenever the IP address of a server changes or when a new set of servers are added as a cluster so that the users will be able to receive traffic from all these servers.
- AWS security groups make use of policies wherein more than one server can reference the same set of policies. This makes managing updates easier as it lowers the configuration error rates.
- For most part, firewalls and AWS security groups provide the same functionalities.

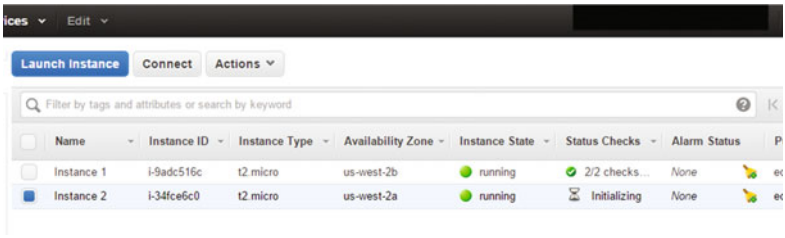
References

https://support.rightscale.com/12-Guides/Dashboard_Users_Guide/Clouds/AWS_Regions/EC2_Security_Groups/Concepts/About_EC2_Security_Groups
http://www.slideshare.net/wilsonrm/aws-meetup-march-6th-2013-vpc-architecture?related=1
http://searchcloudsecurity.techtarget.com/answer/AWS-security-groups-vs-traditional-firewalls-Whats-the-difference

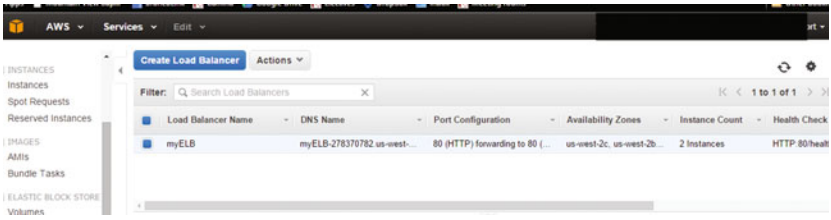
14.4 Project 4: Setup a Load Balancer for Your AWS Instance

14.4.1 Elastic Load Balancer Setup

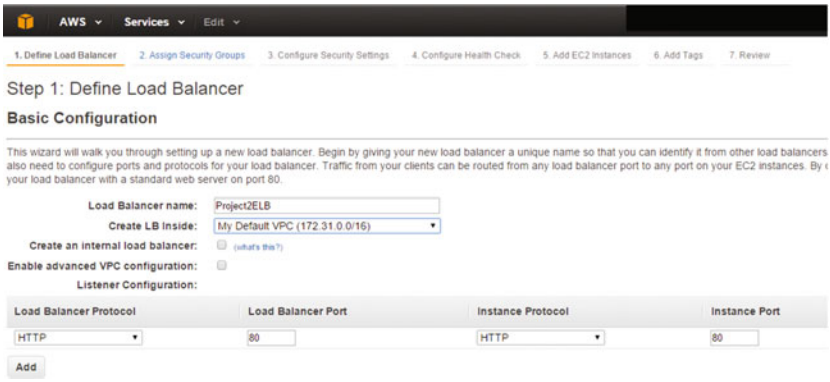
1. Enable 2 instances for the load balancer.
2. Ensure that the security groups of the instances must be covered by the load balancer.



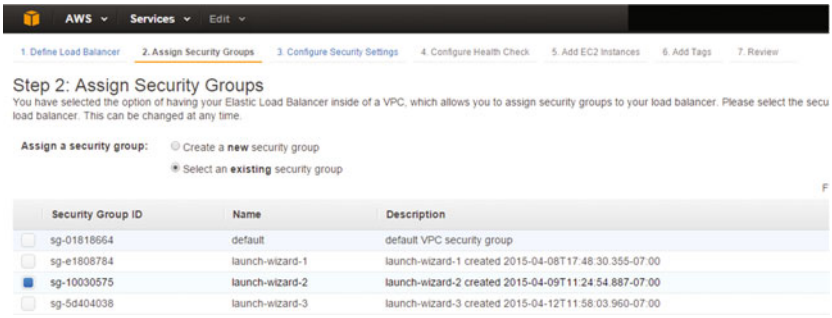
3. Click on load balancer on the left panel.



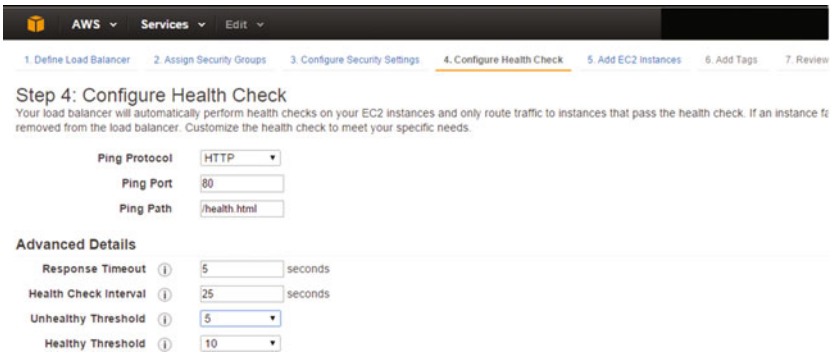
4. Define a name for the load balancer. Also define the http port.



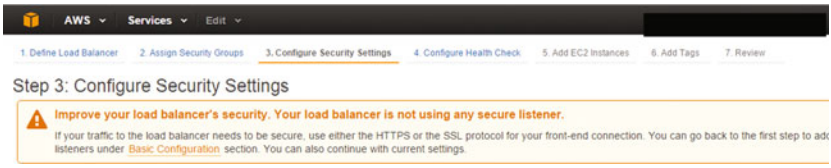
5. Choose all those security groups that are associated with the instances that would be serviced by this load balancer.



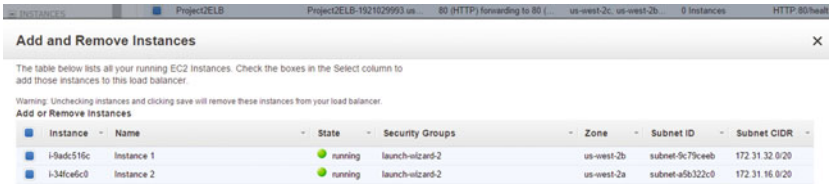
6. Configure the health check parameters. Make sure that the ping path (health. html) is actually present inside the /var/www/html folder. If not, the instances would be deemed out of service.



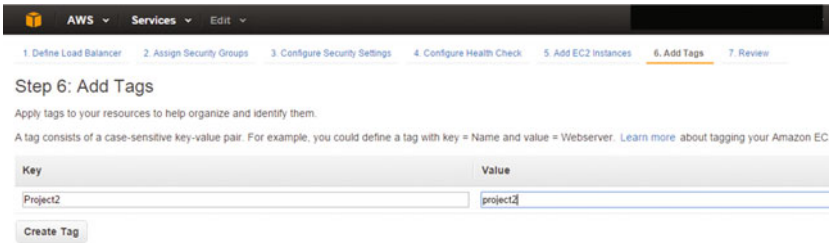
7.



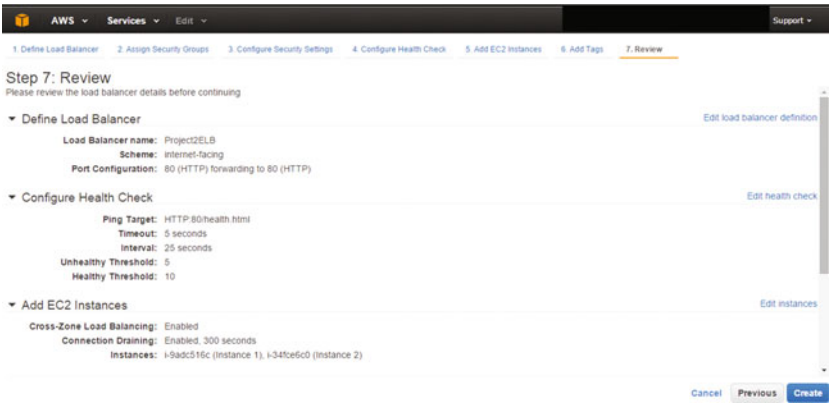
8. Add the required instances to this LB.



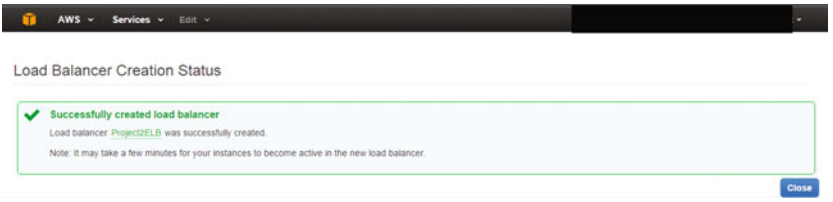
9. Add tags.



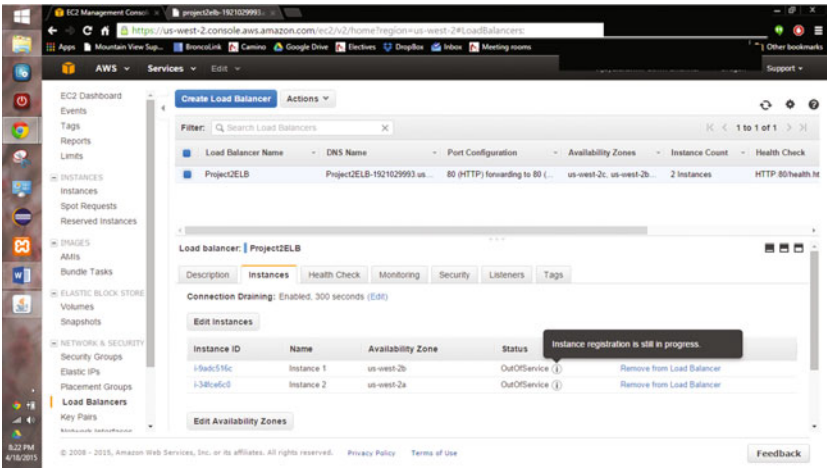
10. Review the specifications.



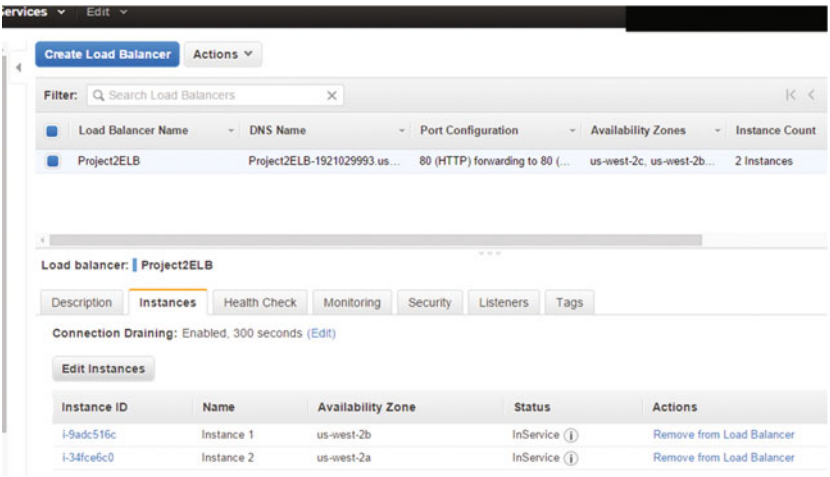
11. View LB creation message.



12. The 2 instances are initially listed as out of service.



13. Wait for some time. Once the health.html file is accessed across both the instances, we get the 2 instances to be in service.



14. URL of the project with the load balancer:
http://project2elb-1921029993.us-west-2.elb.amazonaws.com/

Hi

Hey. I am from Instance 1. My health is fine. Welcome!

Hi

Hey. I am from Instance 2. My health is fine. Welcome!

15. URL of the project without ELB:
    Link of Instance 1 (without ELB):
    http://ec2-54-186-40-103.us-west-2.compute.amazonaws.com/
    Link of Instance 2 (without ELB):
    http://ec2-54-186-51-77.us-west-2.compute.amazonaws.com/

14.4.2 Unique Features of AWS Load Balancer

- ELBs distribute traffic from the Web across all the tagged EC2 instances that reside on one or more than one availability Region.
- ELB makes use of the DNS link to distribute traffic.
- **Cross-zone load balancing** prevents some instances from receiving a higher than average amount of inbound request traffic. This feature ensures that the ELB distributes requests to all instances equally irrespective of the zones they are located in.
- **Connection Draining** serves existing connections on a deregistered instance for the configured time-out period. If an instance is removed from the ELB network due to faulty behavior or maintenance, the end user will not experience any glitch on the services rendered by the ELB.
- **ELB access logs** was made available recently and can go a long way in debugging. Users can therefore identify and rectify any issues they may face at all levels: instance, ELB, zones, etc.
- **Security** employs SSL termination in addition to the security groups provided by Amazon.

- **Sticky session or session affinity** helps connect the application to a user's session. As a result, all of the user's requests are sent to the same application.

Reference

http://cloudacademy.com/blog/top-5-new-features-on-aws-elastic-load-balancer/

14.5 Project 5: Use Elastic IP for Your AWS Instance

14.5.1 How to Make an Instance Elastic

Auto-scaling in generally used to make our instances in the Cloud structure elastic. Typically, elasticity of an instance or a structure means its ability to expand or contract (in terms of resources) depending on the current network demand. And, this process should be automatic without the user having to intervene to either scale up or scale down.

Auto-scaling has two major parts. One is launch configuration: This tells us how to create new instances. Here, we get to choose the name and type of AMI, storage, etc. Second is the scaling groups: This tells us what the rules regulations for the scaling procedure would be. AWS provides many functions which can be used to scale our infrastructure on the basis of network traffic or other parameters.

Auto-scaling is also used to determine the health of the instances created and define checkpoints based on which we can have more/less number of instances (notifications or alarms). Once an instance is deemed out of service (not working), auto-scaling function will take that instance out of its group and will create a new instance in its place.

If we use the elastic load balancer to distribute traffic among the various instances tied to it, ELB also functions as an elastic device. But this does not work in conditions that have huge traffic spikes.

Auto-scaling works in that under this, no instance can be classified as indispensable. This will help us to avoid storing information on any of the instances. Also, the launch or the termination of any new or existing instance is automated, thereby saving a lot of time for owners of the infrastructure.

14.5.2 Extra: Elastic IP

An EC2 instance currently has a public DNS that is used to access or view the Web site content. However, the DNS setting changes with every reboot. These public IPs are that not available in abundance, thereby making it difficult to allocate a dedicated IP address for every instance. This is where Elastic IPs figure in. The running instance can be easily accessed by these Elastic IPs.

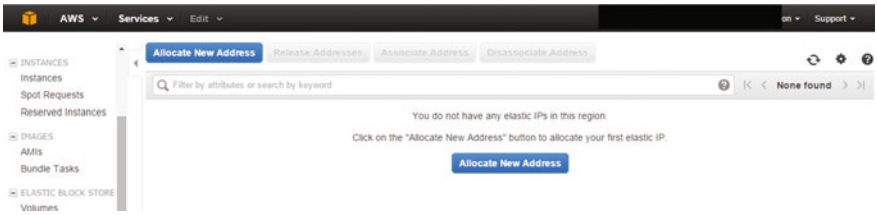
1. Ensure that the instance that is to be made elastic is running.
2. Go to the Amazon EC2 console. Click on "Elastic IPs" under Network & Security.
3. Click on "Allocate new address" that can be used in a VPC.
4. Select EC2-VPC under the network platform list. Click on 'Yes, Allocate'.
5. Then click on 'Associate Address' to link your running instance with this IP address.
6. To change the association of an Elastic IP to another instance, we have to first 'Disassociate Address' from the current instance and then tag this IP to the new instance.
7. When switching between AMIs, we have to stop the current instance, which would automatically disassociate the Elastic IP from this instance. We then start a new instance and then associate the Elastic IP to the latest instance.
8. The new Elastic IP address becomes the public DNS for that instance. To access the Web page, we need to copy this link onto URL tab.

The command/syntax to associate an EC2 instance to an Elastic IP is:

ec2-associate-address [-i *instance_id* | -n *interface_id*] [*ip_address* | -a *allocation_id*] [--private-ip-address *private_ip_address*] [--allow-reassociation]

This command outputs a table that has:

(a) An address identifier;
(b) Elastic IP (EIP);
(c) Instance to which the EIP has been associated;
(d) An allocation ID;
(e) A private IP linked to the EIP (if specified).



Reference

http://docs.aws.amazon.com/AWSEC2/latest/CommandLineReference/ApiReference-cmd-AssociateAddress.html

14.6 Bonus

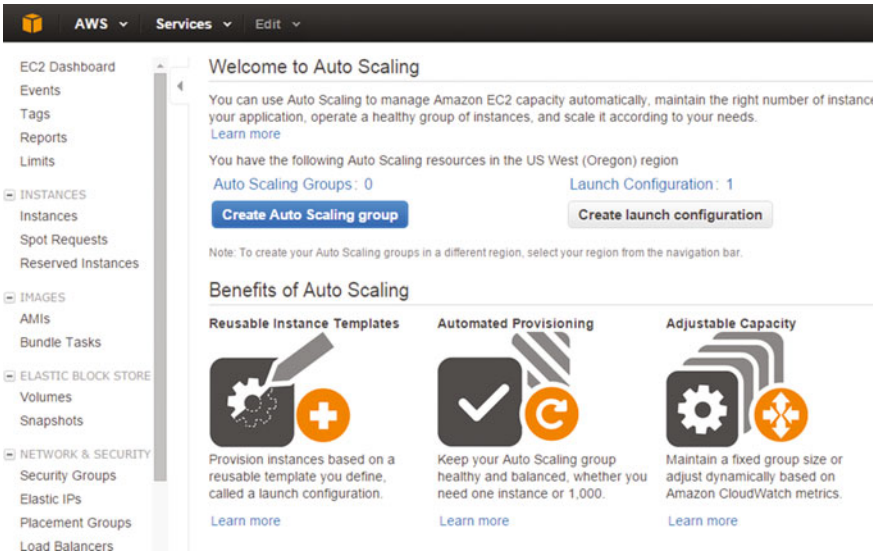
Auto-scaling is generally done when the current load on the existing instances/servers goes above their thresholds. This may lead to delays in processing. To avoid this, we can deploy an auto-scaling mechanism, wherein we can scale out (add instances) or scale in (remove instances) depending on our current needs. This process of scaling is automated.

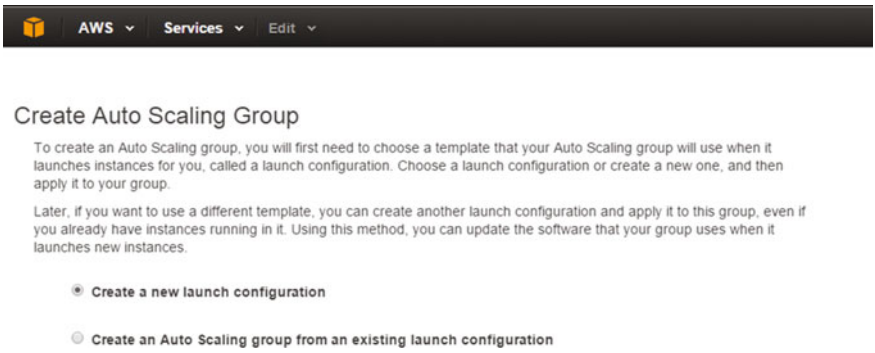
We can set the conditions as to when to scale in or scale out, the configurations of the new instances that will be added, get notifications to your e-mail id whenever a new instance is created, or if an existing instance gets terminated.

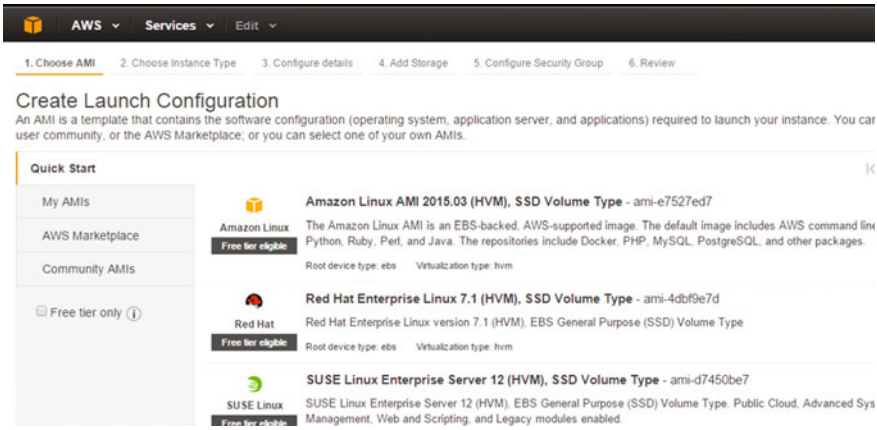
We can also specify the minimum, maximum, and the desired number of instances that are required in a group. Also, we have the option of tagging a load balancer to this auto-scaling group.

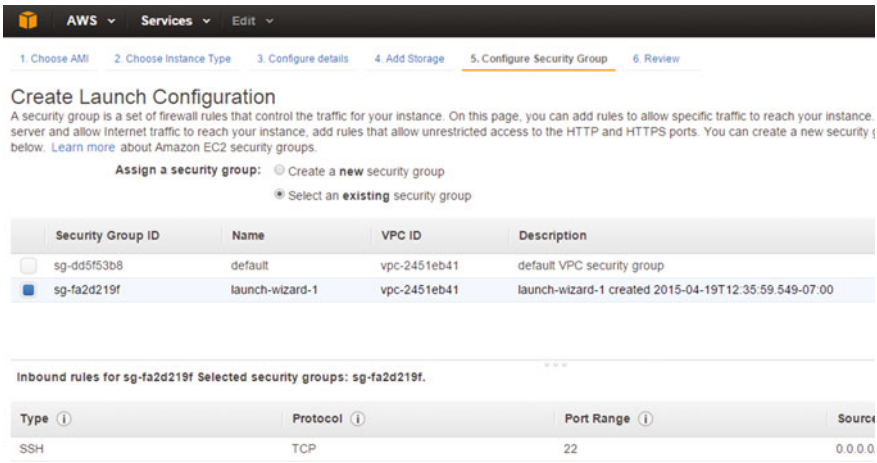
Some screenshots are included to depict the steps followed to set up a functioning auto-scaling group.

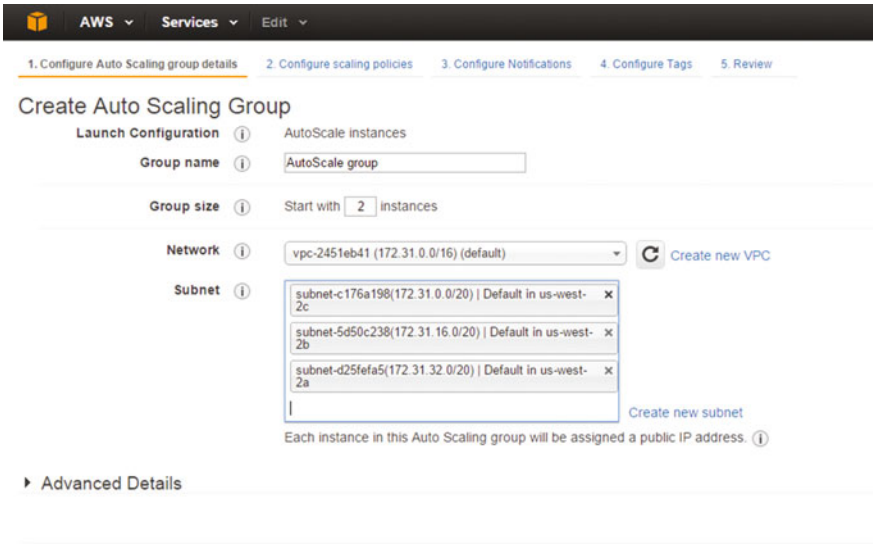
We can either have a load balancer initially sets up, or we can create a load balancer once auto-scaling instances have been created and then add these instances to the LB (but this requires manual addition).



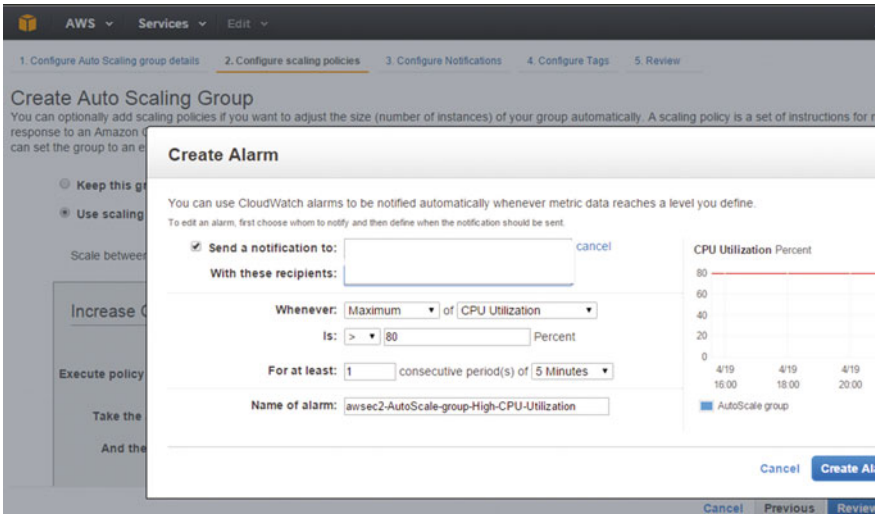

AWS
Services
Edit
Create Auto Scaling Group
To create an Auto Scaling group, you will first need to choose a template that your Auto Scaling group will use when it launches instances for you, called a launch configuration. Choose a launch configuration or create a new one, and then apply it to your group.
Later, if you want to use a different template, you can create another launch configuration and apply it to this group, even if you already have instances running in it. Using this method, you can update the software that your group uses when it launches new instances.
Create a new launch configuration
Create an Auto Scaling group from an existing launch configuration


AWS
Services
Edit
1. Choose AMI
2. Choose Instance Type
3. Configure details
4. Add Storage
5. Configure Security Group
6. Review
Create Launch Configuration
An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You car
user community, or the AWS Marketplace; or you can select one of your own AMIs.
Quick Start
My AMIs
AWS Marketplace
Community AMIs
Free tier only
Amazon Linux AMI 2015.03 (HVM), SSD Volume Type - ami-e7527ed7
Amazon Linux
Free tier eligible
The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line
Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.
Root device type: ebs
Virtualization type: hvm
Red Hat Enterprise Linux 7.1 (HVM), SSD Volume Type - ami-4dbf9e7d
Red Hat
Free tier eligible
Red Hat Enterprise Linux version 7.1 (HVM), EBS General Purpose (SSD) Volume Type
Root device type: ebs
Virtualization type: hvm
SUSE Linux Enterprise Server 12 (HVM), SSD Volume Type - ami-d7450be7
SUSE Linux
SUSE Linux Enterprise Server 12 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Sys
Management, Web and Scripting, and Legacy modules enabled

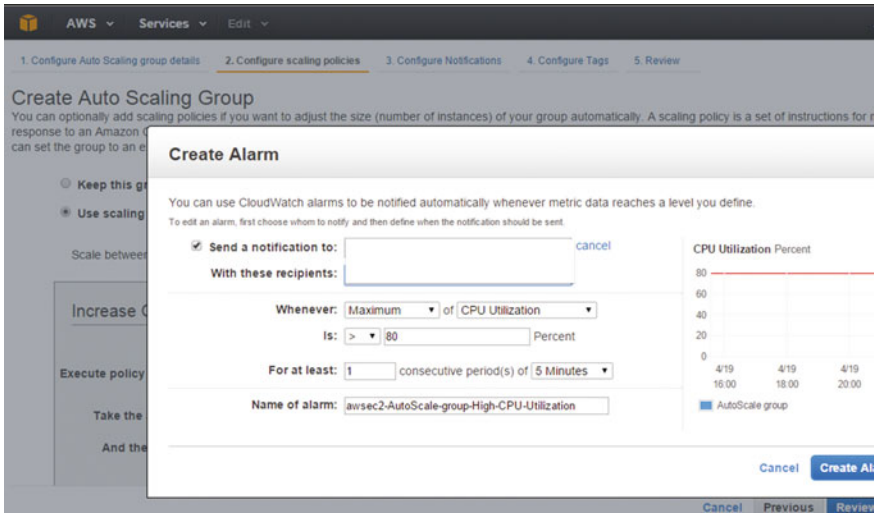

AWS
Services
Edit
1. Choose AMI
2. Choose Instance Type
3. Configure details
4. Add Storage
5. Configure Security Group
6. Review
Create Launch Configuration
A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance.
server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security
below. Learn more about Amazon EC2 security groups.
Assign a security group:
Create a new security group
Select an existing security group
Security Group ID
Name
VPC ID
Description
sg-dd5f53b8
default
vpc-2451eb41
default VPC security group
sg-fa2d219f
launch-wizard-1
vpc-2451eb41
launch-wizard-1 created 2015-04-19T12:35:59.549-07:00
Inbound rules for sg-fa2d219f Selected security groups: sg-fa2d219f.
Type
Protocol
Port Range
Source
SSH
TCP
22
0.0.0.0

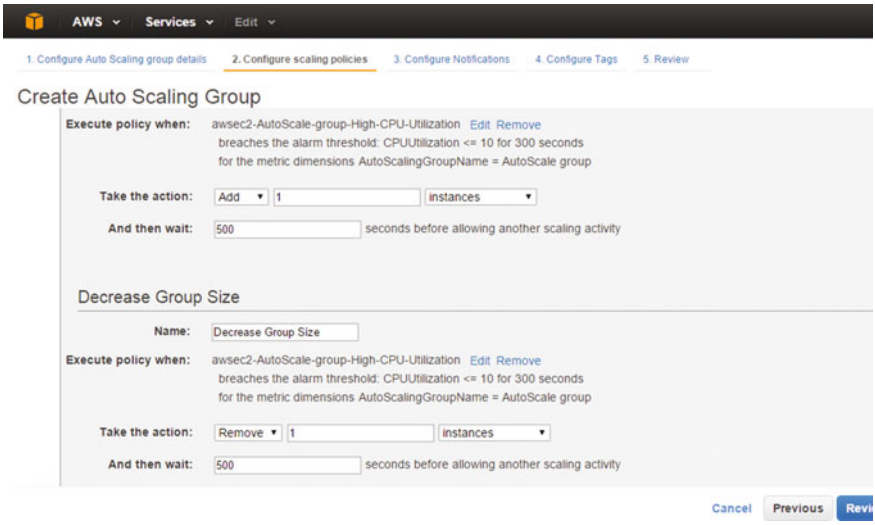

AWS
Services
Edit
1. Configure Auto Scaling group details
2. Configure scaling policies
3. Configure Notifications
4. Configure Tags
5. Review
Create Auto Scaling Group
Launch Configuration
AutoScale instances
Group name
AutoScale group
Group size
Start with 2 instances
Network
vpc-2451eb41 (172.31.0.0/16) (default)
Create new VPC
Subnet
subnet-c176a198(172.31.0.0/20) | Default in us-west-2c
subnet-5d50c238(172.31.16.0/20) | Default in us-west-2b
subnet-d25fefa5(172.31.32.0/20) | Default in us-west-2a
Create new subnet
Each instance in this Auto Scaling group will be assigned a public IP address.
Advanced Details

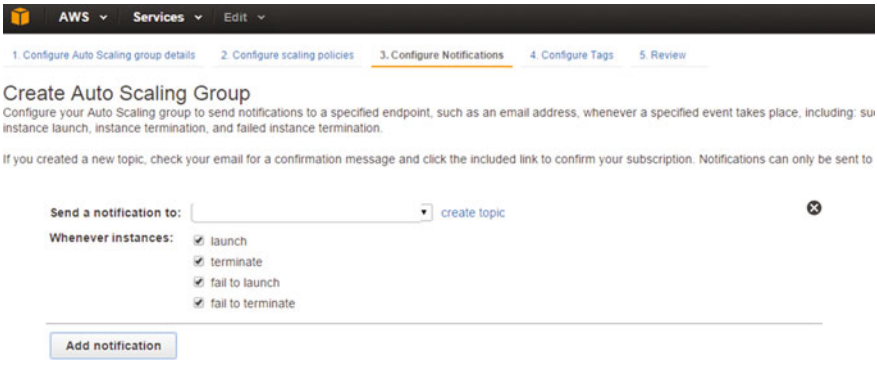
Creating alarm for increasing group size:

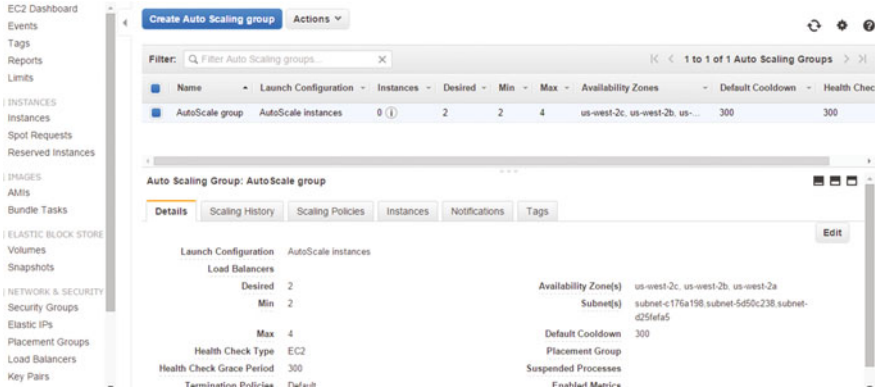


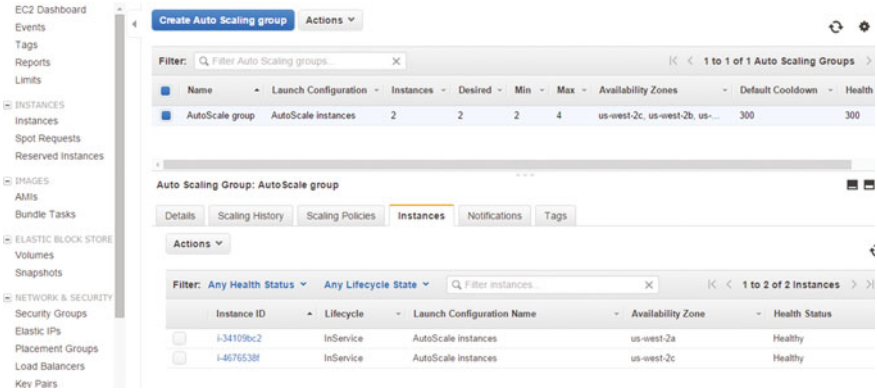
Creating alarm for decreasing group size:



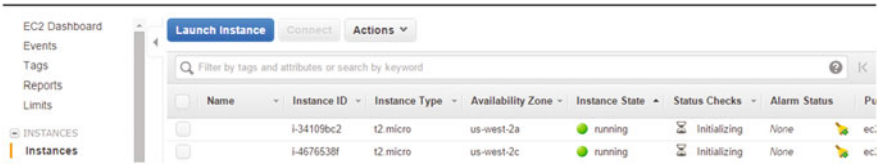

AWS
Services
Edit
1. Configure Auto Scaling group details
2. Configure scaling policies
3. Configure Notifications
4. Configure Tags
5. Review
Create Auto Scaling Group
Execute policy when:
awsec2-AutoScale-group-High-CPU-Utilization Edit Remove
breaches the alarm threshold: CPUUtilization <= 10 for 300 seconds
for the metric dimensions AutoScalingGroupName = AutoScale group
Take the action:
Add
1
instances
And then wait:
500
seconds before allowing another scaling activity
Decrease Group Size
Name:
Decrease Group Size
Execute policy when:
awsec2-AutoScale-group-High-CPU-Utilization Edit Remove
breaches the alarm threshold: CPUUtilization <= 10 for 300 seconds
for the metric dimensions AutoScalingGroupName = AutoScale group
Take the action:
Remove
1
instances
And then wait:
500
seconds before allowing another scaling activity
Cancel
Previous
Revi


AWS
Services
Edit
1. Configure Auto Scaling group details
2. Configure scaling policies
3. Configure Notifications
4. Configure Tags
5. Review
Create Auto Scaling Group
Configure your Auto Scaling group to send notifications to a specified endpoint, such as an email address, whenever a specified event takes place, including: su
instance launch, instance termination, and failed instance termination.
If you created a new topic, check your email for a confirmation message and click the included link to confirm your subscription. Notifications can only be sent to
Send a notification to:
create topic
Whenever instances:
launch
terminate
fail to launch
fail to terminate
Add notification

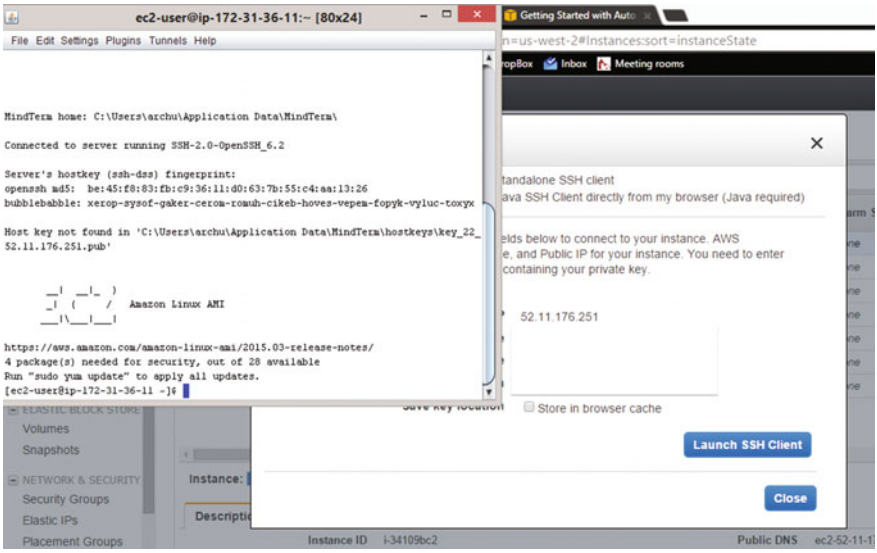




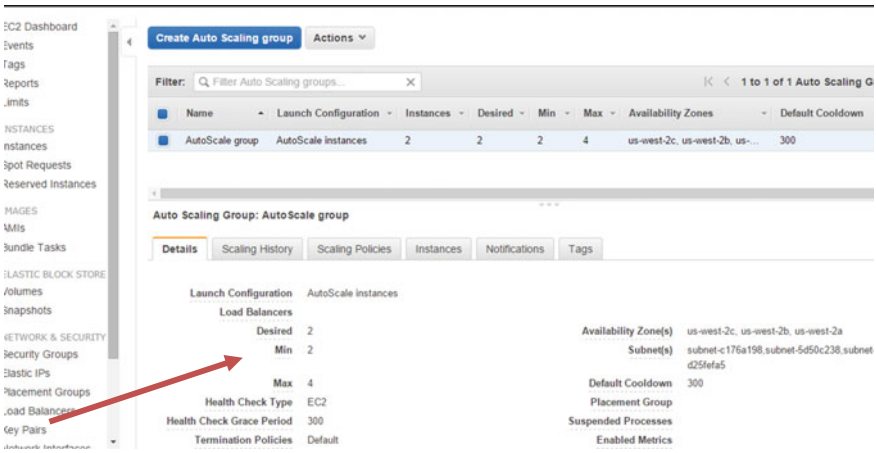
New instances getting automatically created due to auto-scaling:

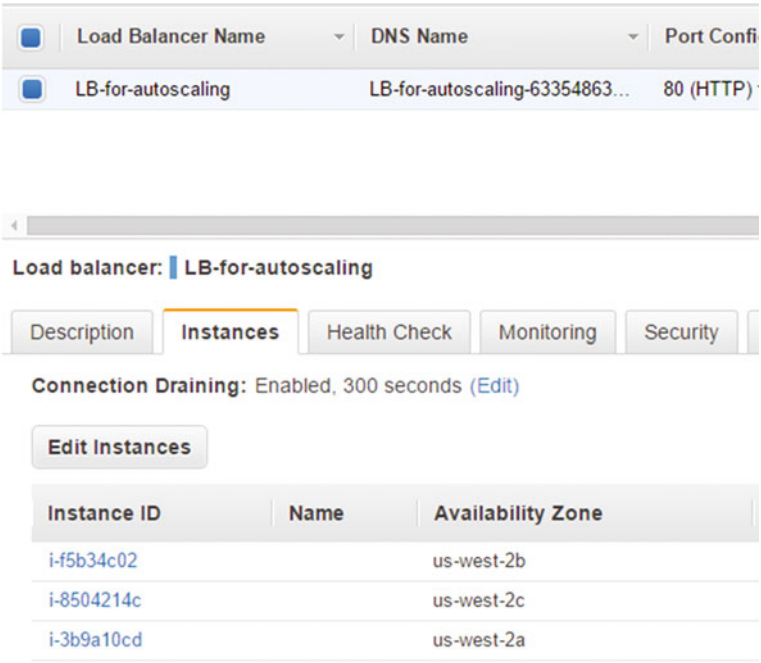


We can also connect these newly created instances:

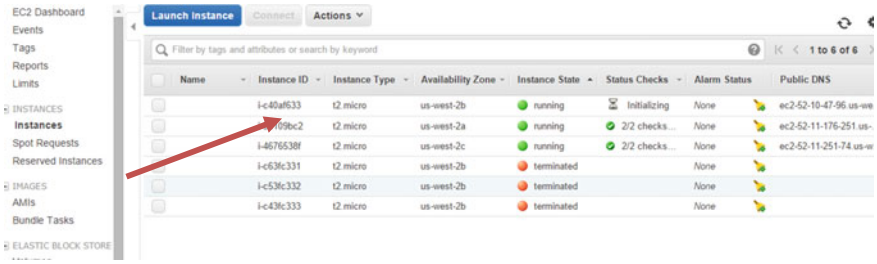


We can also add a load balancer to this group:

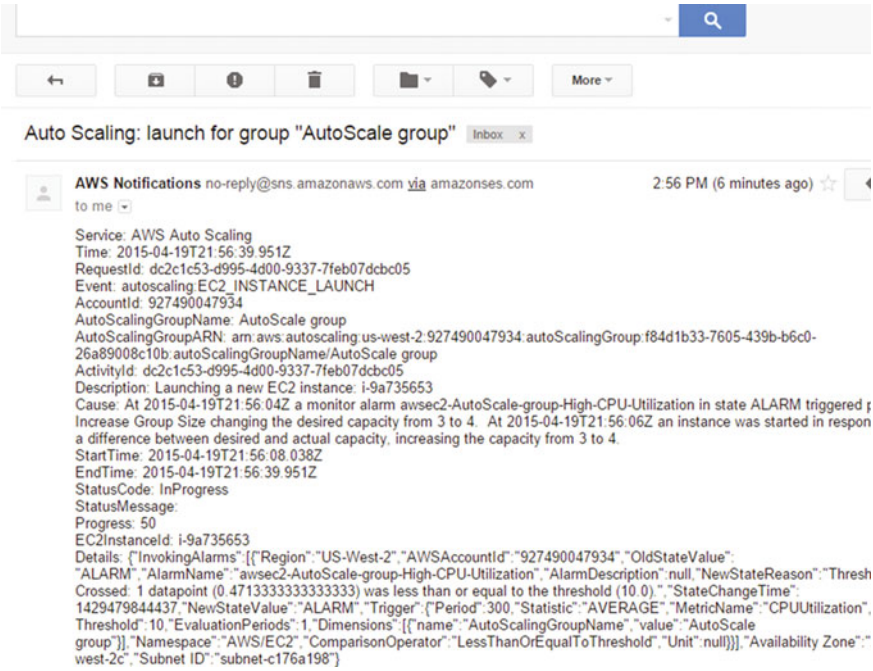




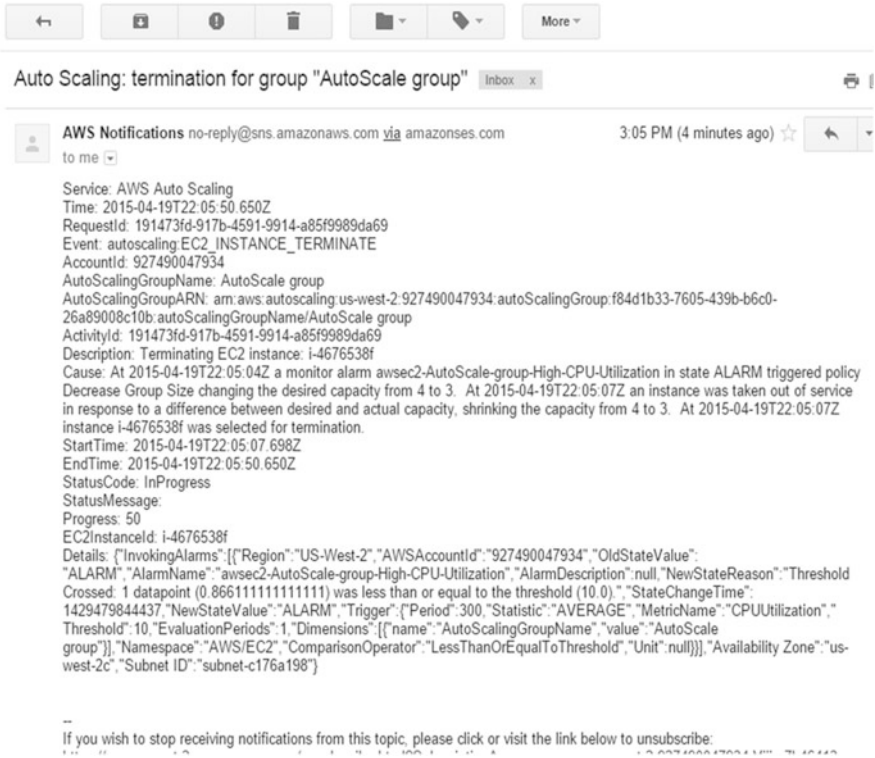
We can also see that when the load on the CPU increases, a new instance gets created automatically. The maximum limit of this group has been set to four instances.



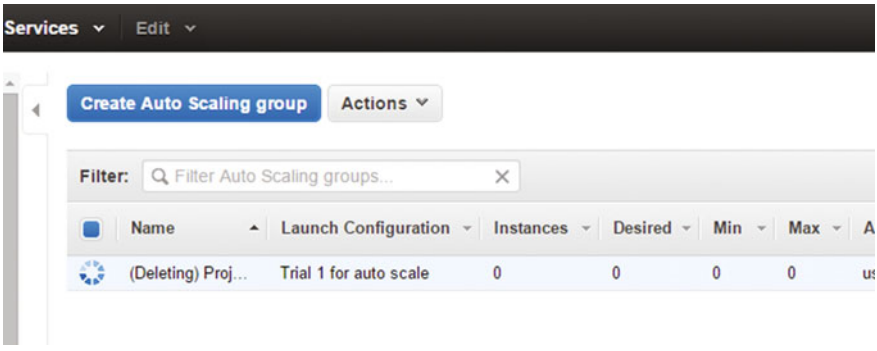
I also got e-mail notifications whenever a new instance was added to this group (scale out):

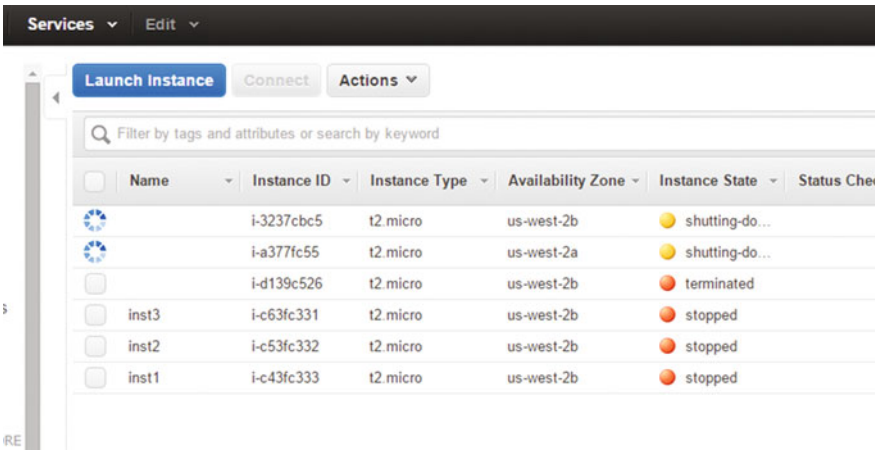


E-mail notification when an instance is terminated from this group (scale in):

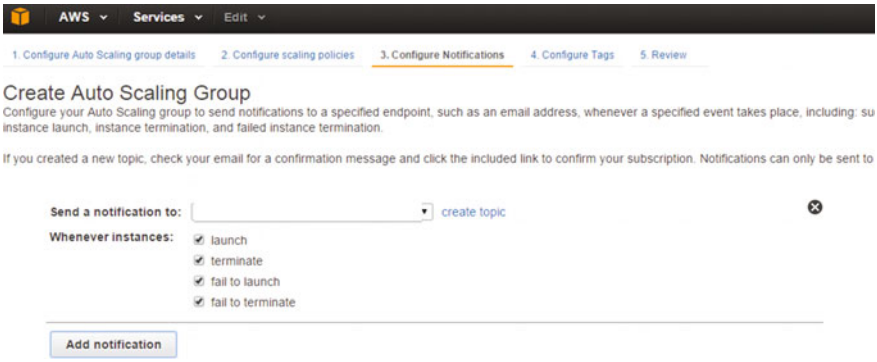


If we delete the auto-scale group, all the instances that were created as a result of this group also get terminated (not just stopped):





We can also view the scaling history to see the list of instances that were either launched or terminated:



14.7 Points to Ponder

1. **AWS offers several attractive services, not required by NIST, such as Elastic IP and auto-scaling. Can you think of a few more examples?**
2. **What equivalent services are offered by other Cloud Service Providers, such Microsoft Azure and Google's GCP?**
3. **How do prices of different server types vary across different Cloud Service Providers?**

Appendix A

Chapter #1 Points to Ponder

1. **Cloud Computing has enjoyed double digit yearly growth over the last decade, driven mainly by economics of Public Clouds versus enterprise IT operations. Can you think of three economic reasons why Public Cloud Computing is cheaper than an enterprise DC?**

   a. Three reasons are Capital expenditure reduction, operational expenditure reduction, and compute elasticity.
   b. Cloud Computing growth has been driven mainly due to reduced CapEx (Capital expenditure, e.g., cost of servers and other data-center equipment). Servers are shared among different customers and hardware costs are amortized over multiple tenants.
   c. Cloud also reduces OpEx (operational expenditure, e.g., power to run the servers and cool them, salaries for IT resources), due to multi-tenancy.
   d. Lastly, compute elasticity refers to the reaction time to provision additional capacity is far less in Clouds (of the order of minutes or hours). This is due to higher available capacity, than placing orders for buying new servers (hours), waiting for their arrival (days) and then deploying them in a private IT data-center.

2. **Concentration of data from various sources in a few data-centers increases the risk of data corruption or security breaches. How can this possibility be minimized?**

   a. Having lots of data at one place makes it for an attractive target of malicious hackers. So additional care should be taken to ensure that the confidentiality of data with encryption.

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6

b. Integrity of data needs bit-level error detection with frequent checkpoints. This will help to identify data corruption and restore it to a previously known good state.
c. Above steps are needed in addition to restricting physical and electronics access to the location of this shared data. It should be accessible to authorized personnel only.

3. **Under what circumstances an enterprise should opt to use a Public Cloud versus its own Private Cloud facilities?**

a. Cost is clearly a major consideration to use any Public Cloud. It imposes little to no Capital expenditure burden on the users, and only charges them what they use.
b. Second consideration is if the data being placed in a Public Cloud is not mission critical, e.g., its access time or performance has some latitude then ok to rely on shared facilities.
c. Lastly, resource availability within an enterprise, e.g., if the data and compute requirements vary greatly over time then elastic nature of Public Cloud is attractive.

4. **Why does a Public Cloud provider want to support a hybrid Cloud, instead of asking enterprises to use its public facilities?**

a. Any Public Cloud provider, who wants to attract new business from an established enterprise, recognizes the hesitation of an IT manager due to loss of control.
b. Thus, offering an on-premise compute service owned and remotely managed by the Cloud provider forms an attractive way to try a new usage model.
c. An option to extend into a Public Cloud for backups is attractive due to its low cost and overheads, making hybrid model a success.

5. **Is there a scenario when Private Cloud may be cheaper than the Public Cloud?**

a. Yes, in a rare circumstance if a private data-center can be kept fully loaded 24×7 (i.e., 24 hours a day for all 7 days a week), then its total cost of ownership may be less than using a Public Cloud.

6. **Can you think of scenarios where a Cloud user favors (a) PaaS over IaaS PaaS, and (b) SaaS over PaaS?**

a. Depending on the nature of workloads and applications, one or more of Public Cloud usage models are useful.
b. Customers who want to work with a bare bone server, or closer to hardware prefer IaaS. They can specify number of CPU cores, size of memory etc.
c. Users that want to use operating system facilities prefer PaaS, such as different storage systems and databases.
d. Companies that want to focus on their end-user applications often go with SaaS, e.g., Netflix and Salesforce.

Chapter #2 Points to Ponder

1. **What led to the desire of people with PCs to connect with one another?**
   a. Professionals wanted to share data and programs with other users.
   b. People wanted to share music with their friends.
   c. Socially motivated sharing of content, e.g., pictures, stories, and news between family and friends gave rise to social networks. This in turn gave rise to sites such as Facebook. This phenomenon brought many new consumers and devices to the Cloud-connected networks and databases.
2. **What led to the growth of thin clients? How thick should be a thick client and how thin should be thin clients?**
   a. Previously mentioned trend of people wanting to stay connected at all times through various means, such as while traveling on vacation with a smartphone but not carrying a PC, meant that phones evolved to have a larger screen and mobile applications. However, phones have limited compute capability due to smaller form factors and users' desire to make the phone charge last a full day. This is the essence of a thin client. Other example of thin clients is an airport display terminal with little or no local processing or storage capability.
   b. A laptop PC is a thick client due to its larger form factor, as compared to a smartphone. However, a laptop should be comfortable to carry around, have a reasonable battery life and not get overly hot during its operation.
   c. Thickness of a thin client depends on the usage model for each category (e.g., an information panel at the airport needs a larger display vs. an enterprise handheld computer in the hands of an Amazon parcel delivery driver needs more storage and larger battery vs. a smartphone to fit in the vacation going tourist's pocket). Besides usage, software update frequency and security requirements are also a consideration. Enterprises typically need encryption on their mobile handheld devices to protect confidential data, which needs more compute and battery power.
3. **How SOA has helped with the evolution of Cloud Computing?**
   a. If a service is well defined then a series of services can be combined. Service-oriented architecture defines inputs, outputs, and transformations carried for each service, a series of which can be chained to offer a compounded service. An example is requesting a virtual machine to run on a certain type of server, then provisioning it with certain OS and Apps, and lastly deploying it to accomplish desired results. This is the essence of IaaS in a Public Cloud, upon which then PaaS and SaaS service models are constructed using SOA as a guiding principle.

4. **Even though cookies pose security risks, why do browsers allow them and what's the downside for a user to not accept cookies?**

   a. Cookies are like a dual-edged sword. They store valuable information to give the appearance of continuity for repeat visitors of a public Web site, by storing passwords, shopping carts, and other user preference data on a user's local machine. However, in case, someone else can access the cookies will be able to see this data. Not accepting any cookies will mean that browser will not have any memory, causing some inconvenience to the users. So they are a necessary evil, but cookies need to be cleaned up via a browser to save storage space.

5. **What's the minimal precaution a public Web site's user should take?**

   a. A user needs to ensure that any secure Web site starts with https://, wherein the last "s" means secure. Also, if user arrives at a Web site by clicking on a link through an e-mail or text message, then the spellings need to be checked to ensure that it is the correct service provider's site instead of a phishing attack to steal financial information.

Chapter #3 Points to Ponder

1. **Small and Medium Business (SMB) users lack financial muscle to negotiate individual SLAs with Cloud Service Providers. So what are their options?**

   a. SMB users should analyze their needs and look for a better fit before moving their entire datasets to a Public Cloud. Reason is that vendor lock-ins are expensive and users may lack in-house capability to juggle data between multiple Cloud Service Providers. SMB customers want to maintain a backup copy of their data in-house, to deal with immediate business needs, so Hybrid Computing is a good option to consider.

2. **Some potential Cloud users are on the fence due to Vendor lock-in concerns. How can these be addressed?**

   a. Some potential users may be on the fence, as they are apprehensive about making a choice. Also, they are concerned about not being able to migrate to another Cloud Service Provider in future. A Cloud Service Provider needs to gain the trust of new users to win their business. Perhaps this can be done by offering to co-locate customers' data in multiple data-centers, and giving them an ability to convert from proprietary Cloud data formats to other industry acceptable standards, etc. This will assure customers that their data is not locked in any specific Cloud Service Provider location or format.

3. **What are the new business opportunities that enable movement of users and data between different Cloud providers?**

   a. Open Stack was started with such an aim to use only open-source software components to build a complete Cloud solution, so users can move between different providers with similar offerings. However, the difficulty of implementing and maintaining such a stack, with no single software vendor to support has made its industry acceptance harder. This presents opportunities for new solution providers to help customers migrate their data from one Public Cloud vendor to another.

4. **What are some key difference between the concerns of a Private versus Public Cloud's IT managers?**

   a. An enterprise's IT manager is generally aware of the users' workloads and thus can place them optimally. In contrast, a Public Cloud's IT manager may not be able to look into users' workload content due to privacy concerns.
   b. Secondly, the amount of trust is higher within an enterprise, since all employees work for the same company. Whereas, in a Public Cloud, an incoming user may be a hacker so extra caution is warranted in terms of resource usage and workloads' behavior, etc.

5. **Why is virtualization important for Public Clouds? Would it also help in Private Clouds?**

   a. By definition, a Public Cloud's servers are shared between different users so virtualization provides a convenient to isolate these users.
   b. In a Private Cloud, all users may belong to the same enterprise so it is acceptable to mix their workloads on the same machine without paying any performance overhead due to virtualization. However, this requires the users to use the same operating system.
   c. An alternative to virtualization is Containers, which allow for process isolation between different programs and their datasets on the same operating system.

6. **Is workload migration a viable solution for a large Public Cloud Service Provider?**

   a. Workload migration is an excellent way to do load balancing between servers.
   b. However, in a Public Cloud, the number of servers is very large and their workloads have an uncertain lifetime, migration is of questionable value unless done at a local level, e.g., in a rack of servers.

Chapter #4 Points to Ponder

1. **Multi-tenancy drives load variations in a Public Cloud. Different users running different types of workloads on the same server can cause performance variations. How can you predict and minimize the undesirable effects of performance variability?**
   a. Analytics can be used to predict periodic patterns of workloads, e.g., for enterprise users, weekdays may be busy, while for recreational users may use more compute on the weekends. For example, Netflix customers watch more movies on the weekend using Amazon's data-centers. Their jobs can be scheduled to run at complementary times, to balance the overall server utilization in a Cloud.
2. **How can you counter the undesirable effects of performance variability?**
   a. A user can submit jobs when the systems are less loaded and more available. Also, user needs to monitor long-running jobs to ensure that they are getting the compute resources necessary, else may need to check point (i.e., save the state of a running program) and migrate the job to another server or Cloud.
3. **How is performance variability addressed in Enterprise Data-Centers?**
   a. Enterprises can earmark different server types for running different types of jobs, e.g., a pool of graphics engines for video rendering applications.
   b. Also, an enterprise can restrict when certain type of jobs can be run, e.g., Software QA only during the nights or weekends with fewer interactive users. This may not be always possible in a Public Cloud.
4. **What is a good way to assess if a noisy neighbor problem exists?**
   a. It is non-trivial for a Public Cloud user to know if her job is slowing down due to a noisy neighbor problem, because IT manager isolates each customer's tasks. However, when these tasks use any shared resource, such as a memory controller, then tasks may slow down due to resource contention. This can be detected by careful monitoring and logging the time it takes for individual tasks at different times. A comparison will reveal if the same tasks, such as a memory access time, are increasing or decreasing in a substantial manner. This change can be attributed to noisy neighbors.
5. **How can a noisy neighbor problem be prevented?**
   a. In a Public Cloud, once a persistent noisy neighbor presence is detected, it is beyond the scope of an individual user to avoid it. Reason is that one user has no control over another user's tasks. However, it can be avoided by stopping the task and requesting a different machine. This may interrupt the service, so a better way is to start another server in parallel and migrate the task in a seamless manner, if possible.

6. **How can understanding a Public Cloud's usage patterns help?**
    a. Observing the performance of a server over time, via a monitoring agent, can often reveal its loading patterns that may be cyclic. An example is to find out when other users on the same server in a Public Cloud are running their jobs, and if possible to avoid delays by scheduling one's own tasks at a different time. Time is money, and in a pay-per-use model, one can save both time and money by scheduling tasks at a time when Cloud servers are lightly loaded.
7. **What operating systems related strategies can a user consider to optimize Cloud usage?**
    a. A Cloud customer needs to know the isolation technique used by the service provider. If HyperV is used as a VMM, then it is better suited for running Windows-based virtual machines. If Xen or KVM is used, then Linux-based VMs will run faster. Else, putting a Linux VM on top of Windows-based VMM will result in a multi-layered call stack that may cause unexpected latencies. In any case, user should do their own benchmarking in a Cloud environment before committing production workloads and making any long-term migration plans.

Chapter #5 Points to Ponder

1. **Many applications combine different workloads, give an example?**
    a. Yes, sometimes an application exhibits different behaviors during its different stages. For example, an online geographically mapping service used by mobile GPS users needs to read one or more large files with map data. This involves accessing storage servers. Then an in-memory graph is built on which different users can search for their traffic routes. This involves compute servers. Lastly, users can query GPS, which needs I/O and networking capabilities to support thousands of users in the Cloud.
    b. Similar scenario can play for other social networking sites too, as users upload or watch pictures or video, post comments, or hold a chat, etc.
2. **Different applications in a Public Cloud can offer opportunities to load balance and optimize resource usage, can you give some examples?**
    a. Compare workload posed by a social networking application versus a video playing site. Each requires different kind of computational resources.
    b. A video movie playback involves transfer of a large file over a few hours, possibly doing real-time decompression, needing some compute but a large networking bandwidth.
    c. A social networking site has a large number of customers, engage in small data interactions via text messages or photos being shared.

3. **What is the best way to improve fault tolerance with multi-tier architectures?**

   a. By having multiple sources of data and compute at each tier in a solution stack, the probability of a system-level failure due to any component-level failure is reduced. An example is of database servers, where the same information can be duplicated making any single server redundant.

4. **What is the advantage of characterizing your workload in a Cloud?**

   a. From a Public Cloud user's point of view, if you understand the nature of your workload, it will help you to pick the right configuration of server to rent. For example, if is CPU intensive then pick a stronger compute machine versus a server with larger memory for a memory-intensive workload such as for big data.

5. **From a Cloud Service Provider's perspective, how the knowledge of a workload's characterization may help?**

   a. As an IT manager for a Cloud data-center, one can use the knowledge of customers' workloads to properly mix them on shared hardware, such that a balanced resource loading can be achieved. An example is to put CPU-intensive job from a customer with memory-intensive task from another. This will ensure that both the CPU and memory controller can be kept busy, instead of having multiple tasks wait for CPU to free up.

6. **Can machine learning play a role to improve the efficiency of a Cloud data-center?**

   a. A Public Cloud Service Provider may agree to not look into customers' workloads for preserving confidentiality. However, one may look at the resource usage of individual customers' tasks to profile them, and predict in advance the future needs. This will help to optimally place them on the right type of servers and other hardware accelerators, such as graphic engines or faster numerical machines.

Chapter #6 Points to Ponder

1. **Distance causes network latency, so AWS has set up various regional data-centers to be close to its customers and reduce delays. This is one way to mitigate network latency. Are there other methods you can think of?**

   a. Last mile bandwidth is often the cause of latency, where traffic within a neighborhood causes the congestion over short distances.
   b. Depending on the nature of data, a service provider can use mirror sites or edge based servers at locations closer to the consumers.

c. Netflix uses such a technology, known as Content Delivery Networks (CDNs) to serve its movies. This works well if the data being delivered is read-only in nature, otherwise synchronization problems will arise.

2. **In an AWS Region, there are different availability zones (AZs) to isolate any electrical fault or unintentional downtime. These could be different buildings or different floors within a data-center building. What are other ways to build a failure safe strategy?**

   a. Another method is to replicate the content across different servers located in different geographies, and then use network switches to route the incoming traffic for both load balancing and fault tolerance to a lightly loaded site.

3. **What kind of metrics need be monitored to ensure the health of a data-center?**

   a. Metrics are of two types, a DOA (dead or alive) type of test, which involves regular pings to the compute and storage servers, and secondly performance tests are needed to ensure that all critical components are optimally loaded.

4. **What are the trade-offs of frequent versus in-frequent monitoring of a server's performance?**

   a. There is no free lunch in life, so any monitoring agent will need some CPU time and memory to run. This cause an additional loading of a server, which if done too often, may result in a perceptible slow-down.

5. **How long the monitoring data should be kept and for what purposes?**

   a. Assuming that Cloud server monitoring has a value beyond instantaneous decision, such as to immediate load balancing, then the result of monitoring should be saved in a log file for later offline analysis. That will help to reveal patterns of usage and enable cost optimizations with future scheduling decisions.

6. **What's the role of monitoring if a SLA is in place?**

   a. Even if a performance-based service-level agreement (SLA) is in place, its compliance needs to be ensured by the Cloud Server Provider's (CSP) IT managers and Public Cloud users. This can only happen with regular monitoring, and comparing actuals with the promised performance, such as the CPU cycles and memory bandwidth. These may translate to an application's observed performance, such as database Read–Write transactions per second. Any gaps will result in renegotiation of a SLA.

Chapter #7 Points to Ponder

1. **Networking gear, such as Ethernet cards, represents the first line of defense in case of a cyber-attack, how can these be strengthened?**
   a. When a denial-of-service (DOS) type of attack happens, a lot of packets of the same type often from a same or smaller group of IP addresses arrive at a Data-center. Such a pattern can be detected via packet sniffing algorithm, for which a watchdog agent can be running as an observer at the network entry point. If a pattern is detected, then the offending IP addresses can be blocked and system administrator is notified.
2. **Recent storage crash in an AWS Public Cloud was caused by the scripting error of internal admins, how could it have been prevented?**
   a. Human errors are generally the cause of most crashes, including those in a data-center. These can be minimized with an input validator. A recent AWS crash was caused by a scripting error. Such an issue can be detected in advance, if the script is run through a simulator or filter first, and if a wild card is found, then ask the IT person if they intend to launch a broader operation? If the answer is affirmative then they are allowed to proceed. This will not totally eliminate the errors but will reduce them.
3. **Does having multiple customers in a Public Cloud increase its risk profile? What operating system strategies you would recommend to prevent business impact?**
   a. Yes, with multiple customers in a data-center, impact of any downtime increases, as any failure will hit multiple businesses simultaneously.
   b. By having fault tolerance in operating systems, or by use of isolation technologies such as virtual machines, failure from any customer can be contained and not be allowed to spread to other users' virtual machines.
4. **Explain the role of attack surface management to minimize security risks? How would you reduce an attack surface?**
   a. An attack surface represents domains in the Cloud that hackers can gain access to, for making unauthorized changes. This may lead to a denial-of-service (DOS) attack for other users. An IT manager must minimize the attack surface available to hackers. An example is the recent Equifax breach, where hackers gained access to valuable personal information of 143 million customers using vulnerability in the Apache server software. Interesting fact is that the vulnerability was publicly known and Equifax was informed months before, but failed to apply the patch.
   b. Reducing the number of access ports, adding authentication and encryption on the remaining few access points can shrink an attack surface. An example is servers at a bank with no USB ports, and verifying any new software

upgrades to coming from legitimate sources before uploading and installing new drivers, firmware, etc. Authentication keys should be stored in a secure place.

5. **How the regular monitoring of a server's usage activity may help to detect a security attack?**

   a. An IT manager should know the regular usage patterns of servers in a data-center and ought to monitor for any irregular activities, e.g., unusual amount of traffic coming on certain network ports, or a high amount of Read–Write activities on certain storage servers. Then an alarm can raised for manual checks to verify if such an activity is legitimate and coming from authorized sources. In a Public Cloud, this means checking the IP address of incoming traffic, and if necessary blocking it to prevent an attack in progress.

6. **What are the security concerns due to a residual footprint?**

   a. Whenever a virtual machine, a Container, or a program ceases to run, it may leave behind some memory and disk footprint. This is in the form of data or code being used by that task. The operating system does not zero out the address space, but puts it on the available list for the next task to use. That next task may start reading the binary content of its memory or disk allocation, thereby getting information about the previous task, which is equivalent to unauthorized access. The previous program can prevent it by zeroing out its memory before exiting.

Chapter #8 Points to Ponder

1. **Migrating to a Public Cloud may result in IT savings, but are there any potential pitfalls?**

   a. Yes, as an old saying goes, there is no free lunch in life. Public Cloud savings are as a result of many customers' tasks consolidation on a fewer physical servers. One impact is that loading by other customers will slow down the machine such that a customer may experience higher or lower performance independent of its own loading of that machine. This in turn will determine how well that one customer's jobs can run, and the response time for its customers, impacting their business.

2. **Business continuity is critical for many mission-critical operations in an enterprise. Can multiple Cloud providers be used to improve Business Continuity Practices (BCP)?**

   a. If business critical is important than any business can't afford any single point of failure, including a Public Cloud. Such businesses can mitigate their

risks by spreading their data and jobs, sometimes with redundant servers in different clouds. Alternatively, they can keep a minimal set of critical data in-house with a local server, which will keep the business running even if the external Cloud goes down.

3. **If local computing is needed, e.g., when Internet connections are not reliable, then what are the other options?**

   a. A hybrid Cloud usage model is desirable for on-premise applications and data.
   b. Other options are check pointing to save the state of databases and virtual machines. Take their backups often and rehearse disaster recovery before it is needed.

4. **Under the following scenario, would an IT provider prefer a Private or Public Cloud?**

   a. Should a B2C IT service provider prefer a Private or Public Cloud?

      A B2C (business to customers) generally maintains an internal data-center in a Private Cloud, e.g., Ebay, but may benefit from a Public Cloud. A part of the data is already open to many customers, such as items being auctioned in the case of Ebay, but other data such as customers' credit card information is strictly private. Later may be the motivation for Ebay to maintain a private data-center. However, a fluctuation in the compute load during weekends and holidays may call for a Public Cloud with a hybrid usage model.

   b. Should a B2B IT service provider prefer a Private or Public Cloud?

      A B2B (business to business) supports interactions between partners, which are not open to Public. Due to its private nature of these transactions, generally a Private Cloud would be preferred. However, there may be a special case to use the Public Cloud hosting based on economic factors, but security considerations should be enumerated and agreed upon in advanced via a service-level agreement (SLA).

   c. Should a C2C IT service provider prefer a Private or Public Cloud?

      C2C (consumer to consumer) is the best case for using a Public Cloud, but the most prevalent site such as Facebook maintain its own data-center to keep a control on the data and performance aspects. However, other C2C users may prefer a Public Cloud to maintain a focus on their application and let someone else take care of their IT needs.

5. **Can you think of a case, where a business tried a Cloud, but decided to not adopt it?**
    a. Yes, several solution providers in Electronics Design Automation (EDA) industry considered using a Cloud, but as we will explain in the next chapter, decided to not adopt it.
6. **What factors may lead a business to consider a hybrid model?**
    a. If a business is located in a location where Internet connections are not reliable, it may lead to a loss of access to the data and programs in a Public Cloud. In such a situation, the business owners will prefer on-premise computing, but may still wish to use the Cloud for backing up their data or using on-demand compute. Besides reliability, other factors such as latency of long-distance networks, and desire to maintain control with some local compute may lead a business to consider hybrid Cloud models.

Chapter #9 Points to Ponder

1. **For mission-critical application, such as EDA or health data, what precautions need to be taken? EDA companies want to recoup R&D investments faster by selling private licenses, while their customers are hesitant to put confidential design data in a Public Cloud. Given this situation, would it ever make sense to migrate EDA workloads to Public Clouds?**
    a. Mission-critical applications need timely access to the data, and must protect its confidentiality and integrity.
    b. A similar business situation existed when most semiconductor companies owned their own fabrication plants (aka fabs), and manufactured their own Silicon Chips. As the cost of fabs kept rising, only a few design companies could afford to have their own fabs. This drove most semiconductor companies to go fabless and send their designs to third-party fabrication plants. Their concerns of confidentiality were superseded by economic needs. Similarly, the cost of EDA tools and need to maintain one's own data-centers may cause design companies to adopt third-party data-centers. Of course, their business confidentiality concerns will need to be addressed first.
2. **How do IP concerns influence the decision to move to Clouds? What precautionary measures need to be considered?**
    a. Intellectual property protection is often cited as the top reason for EDA companies to avoid Public Clouds. This can be addressed with strict regulations providing confidentiality and privacy safeguards, similar to what health-care industry is adopting before using Public Clouds.

3. **Are there any other enterprise or Private Cloud customers, who faced similar dilemmas but economics enabled their migration to Public Clouds?**
    a. Consider Salesforce customers, or medical records management, or health-care providers,
    b. Most companies treat their customer's business relationships as a top secret, but high cost of in-house Customer Relationship Management (CRM) tools drove many companies to adopt an online, Public Cloud-based solution by Salesforce.com. Currently, this Cloud company's business is growing and sustainable, causing in-house commercial tool providers to adopt Cloud too.
4. **What can Public Cloud Service Providers do to accelerate migration of EDA workloads?**
    a. Public Cloud vendors can offer economic incentives, well-defined SLAs, and above all some sort of insurance against data leakage accidents.
5. **What benefits can an EDA solution provider get by migrating to a Public Cloud?**
    a. An EDA solution provider can focus more on their domain specific application and leave the IT management to the Cloud Service Provider (CSP). Furthermore, they can pass on or share any cost savings with their customers, lowering the overall cost of doing business similar to many other Cloud-based application providers.
6. **What benefits can EDA customers get by migrating to a Public Cloud?**
    a. EDA customers are looking for potential cost savings and faster software updates, both of which are possible in the Cloud. However, EDA customers and software will need to migrate their design data and code to a Cloud. Considerations for which were discussed in the Chap. 9.

Chapter #10 Points to Ponder

1. **NIST has well-defined Cloud Standards; compare how Amazon, Google, and Microsoft Public Cloud offerings comply with NIST standards?**
    a. NIST refers to Cloud Computing elasticity, cost, and automation provided by the service providers. Since Public Cloud solutions are evolving rapidly, readers can compare various vendors along these lines.
2. **Amazon is innovating many additional services to attract and keep its EC2 customers. Give some examples?**
    a. Two recent examples are AMR (Amazon's MapReduce) solution and Analytics services. More details can be found on AWS Web site.

3. **Open Stack has been using open-source code to build interoperable Cloud solutions, but its adoption has been slow, why?**

   a. Open Stack is a combination of several mix-n-match open-source based tools, with no single supplier or entity to support the whole stack. Thus installation, deployment and maintenance tasks are left to the users, who may instead want to focus more on running their own business. They may find Open Stack hard to use, and use turnkey Public Cloud solutions.

4. **A start-up wants to save money, and their tasks require 4 GB memory. Should it rent a small-size virtual machine (VM) that costs less, but supports only 2 GB, or a medium size that costs double?**

   a. Even though a small-size VM is cheaper, using it to run tasks that need extra memory will slow it down. This is due to page faults in an operating system, which causes existing memory to be swapped out, written to a disk, and new data to be brought in. This is the only way a VM with 2 GB size can run tasks that require 4 GB. Constant swapping in and out will take extra time, and if the rental cost of a VM is time based, then overtime it will be more expensive to rent a smaller VM than the one that can run the tasks without page faults. Note that swapping will also use system resources, which will impact other VMs too causing a noisy neighbor behavior. An IT manager can detect this and evict the culprit VM.

5. **What's the value of a constant IP address for a Web site hosted in the Cloud?**

   a. A Cloud user's VM may be moved to different machines, each having a different IP address. This may be confusing for the customers of that VM as Domain Naming Server (DNS) mapping for the public Web site will be to the old IP address, Amazon Web Service offers Elastic IP facility to maintain a constant IP address, which can be used to reach a VM independent of the physical server it is hosted on.

6. **Under what circumstance will you prefer to use a load balancer versus an auto-scaler in a Cloud?**

   a. A load balancer acts as a reverse proxy and distributes network or application traffic across a number of servers, all of which must be up and running. While, an auto-scaler is a service to start a new server when the existing ones are getting overloaded. It typically works by monitoring the compute usage, and then it takes a few minutes for new server to get ready. While this saves money, because one pays for a new server only when needed, it can't handle rapid load fluctuations. That will require additional servers to be up and ready, to pick the incoming load almost instantaneously. Load balancer is generally more expensive than using an auto-scaler.

7. **What criterion you would use to compare and contrast different Public Cloud Service Providers?**

   a. A comparison between different Public Cloud Service Providers (CSPs) may start with a cost-benefit analysis, e.g., what kind of service they offer and how much they charge for it. However, making a decision solely on the basis of Return on Investment (ROI) will lead the users to later surprises, so be sure to look at a CSP's latency, reliability, customer support, and above all reputation before making a decision. It is easy to get into a Public Cloud, but hard to migrate between CSPs due to their different tools sets and data formats. The migration should also be considered for the sake of Business Continuity Practices (BCP).

Chapter #11 Points to Ponder

1. **Who owns the datasets in Big Data?**

   a. This is a tricky question, with no clear answer. In case of retail, a consumer owns the choice and financial transactions, but a retailer can combine data from different consumers and remove any personalization information to use it as an aggregated trend. Thus, Amazon is able to suggest what other books customers view or buy after purchasing a certain book. Similarly, a patient owns the medical records but a hospital or doctor can use them for deriving generalized trends. Then in turn, these can be sold to a pharmaceutical company to decide which drugs to develop. In all these case if data is collected and computed in a Cloud, then clearly the Cloud Service Provider (CSP) does not own this data.

2. **What is desirable for external researchers to collaborate for data analytics in Cloud?**

   a. Different parties can get together and collaborate in a Cloud, each bringing a unique value to the partnership. As an extension of medical example in the previous question, a hospital can provide medical records devoid of patient's personal identification information, a pharmaceutical company can bring the results of latest drug trials, and multiple medical researchers can bring their ideas and algorithms for future drug or medical device inventions. Any such collaboration needs to assure that no party can copy or take out the data belonging to another party, only use and learn from it. This is called privacy-preserving analytics and is still an evolving field in the Cloud Computing.

3. **Are there any vendor lock-in concerns with big data?**
    a. Yes, since each Public Cloud Service Provider may use a proprietary format and tools to manage the big data, it will be hard, if not impossible, for data owners to quickly migrate their compute to another Cloud. One way to manager this is to start with a multi-Cloud strategy or maintain copies locally.
4. **AWS offers several attractive services, not required by NIST, such as Elastic IP andAuto-scaling. Can you think of a few more examples?**
    a. Two recent examples are Amazon's MapReduce (AMR) solution and Analytics services. Details can be found on AWS Web site.
5. **What equivalent services does other Cloud Service Providers offer, such Microsoft Azure and Google's GCP?**
    a. Left for reader to discover as the Cloud industry is rapidly evolving.
6. **How do prices of different server types vary across different Cloud Service Providers?**
    a. Left for reader to discover as the Cloud industry is rapidly evolving.

Chapter #12 Points to Ponder

1. **There is potential to have more devices and machines in an increasingly automated world, next wave of Cloud Computing growth is coming from IoT. Can you list additional areas to drive the growth of Cloud Computing?**
    a. Yes, self-driving cars represent a large growth area for Cloud Computing to share data, analytics, and update software or instructions in real-time.
    b. Home and industrial surveillance using security cameras offer growth potential for Cloud, with any intrusion data stored in a remote Cloud.
    c. Pollution monitoring and environmental sensors located in field with central data collection will help to fuel the growth of Cloud.
2. **How could one improve the Cloud's performance and support for IoT?**
    a. By having distributed and redundant systems for a fail-safe solution.
    b. Avoid having a single point of failure.
    c. Backend Cloud Services are needed to log data and results for audit and machine learning inferences.
    d. Sensors can generate enormous data requiring Cloud storage and compute power. However, moving data in and out of Cloud is slow and expensive. So input–output considerations will require local compute and storage power.

3. **Why is Edge Computing needed for self-driven cars in future?**

    a. Sensors in a moving car can generate enormous data requiring Cloud storage and compute power. Examples of this are forward-looking and side-view cameras. However, moving data in and out of Cloud is slow and expensive. A car may need to react quickly due to changing road conditions. So input–output considerations for the sensor data will require local compute and storage power. However, any learning and performance data can be reconciled with backend servers during night or when the car is safely parked.

4. **Can you think of another example of Edge Computing devices on a road?**

    a. A network of traffic lights can communicate and coordinate between them, for adjusting to an accident that may be sending scores of cars to other roads. Normally, such a scenario can cause bottlenecks while other roads may be empty. Using a combination of artificial intelligence and machine learning methods, in future a network of traffic lights may be able to adapt their sequence of red–yellow–green sequence to ease the wait times on critical junctions.

5. **What is the trust and security model for edge devices?**

    a. Edge devices in a Cloud need backend Cloud services are needed to log data and results for audit and machine learning inferences. However, devices need to trust the Cloud servers, and Cloud needs to trust the incoming device data.

6. **What kinds of attacks are possible using IoT and Edge devices?**

    a. It has been shown that an army of botnets (a term used for devices on Internet) can be hijacked by hackers and used for launching distributed denial-of-service (DDOS) attacks on unsuspecting Cloud servers. An example is of home surveillance cameras that had unsecured IP addresses used for bringing down a security journalist's blog site.

7. **What is the impact of vendor lock-in on Edge Computing devices?**

    a. Vendor lock-in represents difficulty of migrating from one device maker or service provider to another. This can lead to higher costs or inefficiencies. An example is security cameras. If a business has initially installed surveillance cameras from a vendor, then as the business needs expand, it often must buy additional equipment from the same vendor. However, later on there may be another provider with better or cheaper cameras, but these may not work with the existing system. So the business must spend more money, or settle for older models just to ensure that the system works in a homogenous manner. This can be avoided if industry standards exist and only equipment complying with these standards is deployed to ensure a plug-n-play approach. An example of this is in home entertainment systems, where TVs, DVDs, and set-top boxes can come from different manufacturers yet can work together well.

Appendix B
Additional Considerations for Cloud Computing

(1) **Backups**: Backup refers to the copying of physical or virtual files or databases to a secondary site for preservation in case of equipment failure or other catastrophe [1]. Process of backing up data is pivotal to a successful disaster recovery (DR).
(2) **Check Pointing**: Check pointing is a technique [2] to add fault tolerance into computing systems. It basically consists of saving a snapshot of the application's state, so that it can restart from that point in case of failure. This is particularly important for long-running applications that are executed in failure-prone computing systems. Check pointing enables the system to be restored to a known good configuration.
(3) **Cloud Accelerators**: As Cloud applications become prevalent and specialized such as face recognition, or real-time traffic management, run-time efficiency becomes even more important. When using a VM or Container, a stack of drivers and operating system services may add to the workload latency. Some Public Cloud Service Providers have started to use Field Programmable Gate Arrays (FPGAs) to accelerate specialized applications. Both of these technologies allow mapping an algorithm to a hardware implementation, which enables a higher execution speed as compared to running the same application in software. Amazon's AWS is offering F1 as compute instances with FPGAs to create custom hardware acceleration [3]. Similarly, Microsoft has been deploying FPGAs in Azure servers [4], to create a Cloud that can be reconfigured to optimize a set of applications and functions. Unlike a CPU, that can be rapidly programmed and run-time binaries can be switched instantly, an FPGA takes time to be programmed for a specific application. Thus it needs to be running for a longer time to amortize the time and cost of setting up FPGAs, but in return the FPGA will run faster than the software only implementation of an application. In contrast, Google has been using Tensor Processing Units (TPUs) with Applications Specific Integrated Circuits (ASICs) for Machine Learning (ML) workloads [5]. This field is still evolving.
(4) **DevOps**: Deploying applications in an enterprise or Cloud is non-trivial. It may involve configuration changes in server and network, setting up environment variables, monitoring and collecting performance logs, and finally a method to

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6

roll-out new releases. When development engineers (in short Dev) work closely with IT operation engineers (in short Ops), then the automation and repeatability of following steps is called DevOps:

1. Planning
2. Coding
3. Building
4. Testing
5. Releasing
6. Deployment
7. Operating
8. Monitoring
9. And finally a loop back to step back for planning next release

A successful DevOps cycle requires following steps, for which several tools, such as Puppet [6] and Chef [7] are available.

1. Infrastructure management
2. Configuration management
3. Deployment automation
4. Log management
5. Performance management
6. Monitoring and corrective actions

Finally, DevOps is a set of principles and practices, both technical and cultural, which can help deploy better software, faster.

(5) **Disaster recovery**: Disaster recovery involves a set of policies, procedures, and tools to enable the recovery or continuation of vital technology infrastructure and systems following a natural or human-induced disaster. Disaster recovery focuses on the IT or technology systems supporting critical business functions, and is a subset of business continuity [8].

(6) **Function as a Service (FaaS)**: A new category of Cloud Computing service is growing fast that allows customers to develop and run applications as functions [9], without worrying about the underlying server architecture. Examples of this are AWS Lambda, Google Cloud Functions, and Microsoft Azure Functions. In a FaaS model, functions are event-driven and expected to start in milliseconds, while customer only pays for the execution time of a function, An example of a function is image loading in response to a Web site click, or an event triggered by a sensor reading, such as storing a video footage in Cloud upon motion detection by a security camera.

(7) **Fault Tolerance**: Fault tolerance is the property that enables a system to continue operating properly in the event of the failure of (or one or more faults with in) some of its components. If its operating quality decreases at all, the decrease is proportional to the severity of the failure, as compared to a naively designed system in which even a small failure can cause total breakdown. Fault tolerance is particularly sought after in high availability or life-critical systems.

The ability of maintaining functionality when portions of a system break down is referred to as graceful degradation [10].

(8) **Linux Containers**: Linux Containers enable packaging of applications, including their entire run-time files necessary, in an isolated environment. This enables it easy to move the contained application from development, to test, and finally to production while retaining full functionality. Containers are especially helpful with distribution of self-contained application packages, as described above in DevOps. Each Container has code, a run-time environment, system tools and libraries, all of which are packaged for a complete distribution and installation on a server. A Container has short lifetime, if used as Elastic Compute resources, as a disposable unit to be spun up or shut down as the demand grows or wanes. Cloud providers are increasingly exploring Containers, instead of or in addition to VMs, for deploying individual customers' applications on a shared server. One clear advantage is that Containers are considered lightweight in terms of memory or run-time requirements as compared to a VM. This allows more Containers, as compared to fewer VMs, to run on a given server platform. Putting an application in a Container enables it to be secure from other malware, as it has its own address space that an outside application cannot access. A possible downside is that the same operating system is shared between multiple Containers, increasing the possibility of noisy neighbors. In contrast, as shown in Fig. B.1, virtualization allows a mix of different operating systems on the same server, and better control of resource allocation between the VMs.

Linux Containers are based on open-source technology [11], such as CRI-O, Kubernetes, and Docker, making it easier for teams to develop and deploy applications. These technologies are also referred to as an orchestrator, which manages multiple application Containers across a cluster of servers and keeps them running. Docker Container is a widely used orchestrator to ensure that the code written by programmers will be portable and has the same working independent of the computing environment. A Docker enables multiple Containers to run on the same machine and share the operating system kernel,

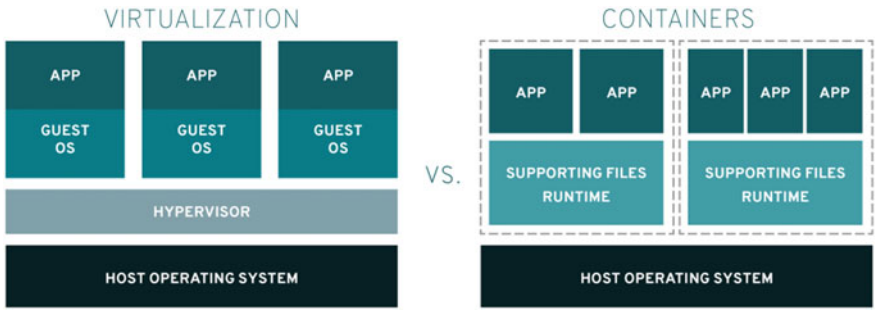


Fig. B.1 A comparison of VMs versus Containers [11]

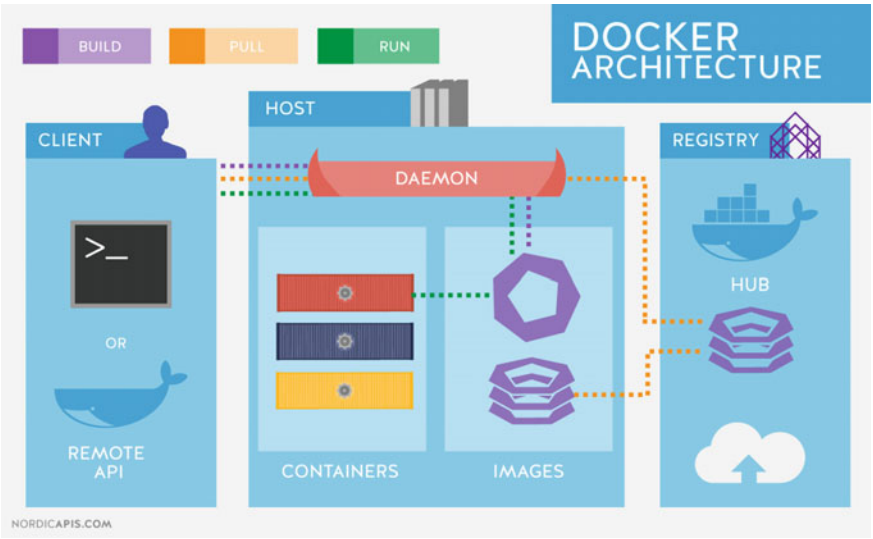


Fig. B.2 Docker Orchestration architecture [12]

each one of which could run as isolated processes in user space, thereby providing additional security and isolation as shown in Fig. B.2.

(9) **Watchdog**: A watchdog [12] is a device used to protect a system from specific software or hardware failures that may cause the system to stop responding. The application is first registered with the watchdog device. Once the watchdog is installed and running on your system, the application must periodically send information to the watchdog device. If the device does not receive this signal within a set period of time, it would reboot the machine or restart the application. Watchdog services are also available for Internet Web sites. In these instances, the watchdog may be set to monitor your own Web site by attempting to access it from several different city or country locations at regular intervals. You could then view an online report of the connectivity or have the watchdog system e-mail you should it become inaccessible. Some Web sites also offer readers a watchdog service where they receive instant e-mail notification when the Web site offering the service has been updated.

References

1. http://searchdatabackup.techtarget.com/definition/backup
2. https://en.wikipedia.org/wiki/Application_checkpointing
3. https://aws.amazon.com/ec2/instance-types/f1/
4. https://azure.microsoft.com/en-us/resources/videos/build-2017-inside-the-microsoft-fpga-based-configurable-cloud/
5. https://Cloud.google.com/tpu/
6. https://puppet.com/solutions/devops
7. https://www.chef.io/devops-tools/
8. https://www.redhat.com/en/topics/containers/whats-a-linux-container
9. https://en.wikipedia.org/wiki/Disaster_recovery
10. https://en.wikipedia.org/wiki/Function_as_a_service
11. http://scholarworks.umass.edu/cgi/viewcontent.cgi?article=1186&context=cs_faculty_pubs
12. http://apachebooster.com/kb/what-is-a-docker-container-for-beginners/
13. http://www.webopedia.com/TERM/W/watchdog.html

Appendix C
Suggested List of Additional Cloud Projects

1. AWS has made Genomes Project data publicly available to the community free of charge.
   a. https://aws.amazon.com/public-datasets/
   b. AWS provides a centralized repository of public data hosted on Amazon Simple Storage Service (Amazon S3).
   c. This data can be seamlessly accessed from AWS services such Amazon Elastic Compute Cloud (Amazon EC2) and Amazon Elastic MapReduce (Amazon EMR).
   d. AWS is storing the public data sets at no charge to the community.

   Project: Researchers can use the Amazon EC2 utility computing service to dive into this data without the usual capital investment required to work with data at this scale. Show how you can search and assemble a particular gene sequence, such as for e-coli? More details are given here: http://ged.msu.edu/angus/tutorials-2013/files/lecture3-assembly.pptx.pdf.
2. **Analyze a live Twitter stream** to indicate emotions of crowd for a particular event, using MapReduce to search for keywords to determine the sentiments of each tweet.
3. **Use a popular real-estate service**, such as Zillow's APIs, to search for households that are more likely to donate money for a symphony. Heuristics to look for owners who have a certain property value, such as greater than $1 million USD, or have already have paid off their houses with no mortgage payments due.
4. **Compare AWS to Google Cloud**. Technical and Business case study of how the two compare. Sign up for the Google Cloud free trial. Cost-benefit analysis of the two. Identify Iaas, Paas offerings, their cost and benefits and which one you would use. Are there any unique offerings on the Google Cloud if so go into the details and how are they different and better than AWS.
5. **Compare AWS to Azure**, http://azure.microsoft.com/en-us/services/virtual-machines/(Links to an external site.) (Links to an external site.). Technical and Business case study of how the two compare. Sign up for the Azure free trial. Cost-benefit analysis of the two. Identify Iaas, Paas offerings, their cost and benefits and which one you would use. Are there any unique offerings on

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6

the Microsoft Cloud if so go into the details and how are they different and better than AWS.

6. **Projects related to Online Libraries**: Imagine a community of users, such as in an apartment complex, which wishes to share books within the community or neighboring communities with like-minded readers. Create an online database of such users, with a list of books that each user is interested in, and another list of books that the user has read, to share with others. Using an online blog, such as Wordpress, each user can comment on a specific book.
7. **Projects related to Online Communities**: Extend the previous project to include movies, music, or any other items that a community of users would like to discuss using a Cloud-based database and a virtual network of users.

Index

A
Access, 70
Access control, 38, 93
Adoption, 136
Aggregation, 22
Algorithms, 159
Amazon Machine Image (AMI), 86
Amazon's Map Reduce (AMR), 254
Amazon Web Service (AWS), 53
Analytics, 161
Architecture, 65
Auto-scaling, 146, 148, 149, 169, 239, 257
Availability zones, 85, 143

B
B2B, 115
B2C, 115
Backups, 259
Bandwidth, 71
Benchmark, 62
Billing, 141

C
C2C, 116
Cache, 64, 67, 72, 73, 76, 77, 97, 102
Capacity forecasting, 152
Central Processing Unit (CPU), 56
Check pointing, 259
Clickjacking, 36, 40
Client–Server, 1, 13, 14, 172
Cloud bursting, 53
Cloud Computing, 2
Cloud Watch, 89, 90, 149
Cluster, 48, 165, 166
Coefficient of Variation (CV), 57
Communication, 93
Community Clouds, 43
Computer-Aided Design (CAD), 125
Controllability, 51
Cookies, 36
Cost, 149
Cross-Site Request Forgery (CSRF), 38
Customer Relationship Management, 254

D
Database, 68
Data-center, 46
DBMS, 20
Denial Of Service (DOS), 250
Defense Advanced Research Projects Agency (DARPA), 15
DevOps, 260
Disaster recovery, 260
Docker, 261

E
EC2, 86
Edge Computing, 171, 176–180, 182
Elasticity, 44, 188, 241
Elastic Load Balancer (ELB), 146
Elastic IP, 146
Elastic Map-Reduce, 165
Electronics Design Automation (EDA), 124
Energy, 23

N. K. Sehgal and P. C. P. Bhatt, *Cloud Computing*,
https://doi.org/10.1007/978-3-319-77839-6

Enterprise Service Bus (ESB), 31
Exposure, 38

F
Fabs, 253
Fault tolerance, 260
Frame busting, 36

H
Hadoop, 162
Health Insurance Portability and Accountability Act (HIPPA), 65
High-Performance Computing (HPC), 68
Hypertext Markup Language (HTML), 18, 36
Hypertext Transfer Protocol (HTTP), 18, 25
HTTP Secure, 17, 25, 219
Hybrid Cloud, 3, 35, 43, 52

I
IaaS, 2, 42, 95
In-Band (IB), 87
Inferences, 159
Infrastructure, 45
Injection, 38
Interactive, 69
Internet of Things (IoT), 173, 176
Interoperability, 14, 53, 171, 178, 180, 182, 183
Intellectual Property (IP), 128

J
JavaScript Object Notation (JSON), 20

L
Latency, 8
Last Level Cache (LLC), 76
Life cycle, 166
Local Area Network (LAN), 14

M
Machine learning, 161, 174–176
Map-reduce, 162
Memory, 56
Migration, 8, 48
Misconfiguration, 38
Monitoring, 88

N
Nagios, 90
National Institute of Standards and Technology (NIST), 2, 42, 44
Networking, 13
Noisy neighbor, 86

O
Observability, 51
Open stack, 255
Open web, 37
Optimizations, 153
Original Equipment Manufacturers (OEM), 62
Out-Of-Band (OOB), 87
Open Web Application Security Project (OWASP), 37

P
PaaS, 2, 42, 95
Performance, 48, 55, 86
Personal computers, 171
Private Clouds, 3, 43
Protection, 94
Protocols, 24
Public Clouds, 43

Q
Quality of Service (QoS), 41, 45, 47

R
Region, 86
Reliability, 53
Remote scripting, 36
Representational State Transfer (REST), 20

S
Saas, 2, 42, 95
Security, 22, 37
Security Issues, 177
Self-service, 44
Service-Level Agreements (SLA), 7, 47, 65
Service-Oriented Architecture (SOA), 3, 26, 28
Silicon, 132

Simple Object Access Protocol (SOAP), 18
Simulation, 130
Small and Medium Business (SMB), 44
Storage, 56
Storage Networking Industry Association (SNIA), 22
Streaming, 22, 67, 73, 80
Synthesis, 130

T
Transmission Control Protocol/Internet Protocol (TCP/IP), 14, 17
Thin client, 24
Twitter, 167

U
Universal Description, Discovery, and Integration (UDDI), 18

V
Variability, 161
Variety, 160
Velocity, 160
Veracity, 161
Verification, 131
Virtual machines (VMs), 47
Virtual machine monitor (VMM), 87
Virtual Private Cloud (VPC), 43, 51
Virtual Private Network (VPN), 105
Very-Large-Scale Integration (VLSI), 124
Volume, 160
Vulnerabilities, 35, 39

W
Watchdog, 262
Web App, 25
Web Service, 17, 25
Web Services Description Language (WSDL), 18
Web Threat Models (WTM), 35
Wide Area Network (WAN), 14
Workload, 46, 47, 51, 55, 61–63, 65–70, 72, 77–79, 81, 87, 124, 125, 128, 130, 131, 133, 136, 146, 148, 187, 247, 259
World Wide Web (WWW), 22

X
XSS, 38
XML, 18, 25