ENTERPRISE ADOPTION ORIENTED CLOUD COMPUTING PERFORMANCE OPTIMIZATION

Moustafa Noureddine

A thesis submitted in partial fulfilment of the requirements of the University of East London for the degree of Doctor of Philosophy

April 2014

Abstract

Cloud computing in the Enterprise has emerged as a new paradigm that brings both business opportunities and software engineering challenges. In Cloud computing, business participants such as service providers, enterprise solutions, and marketplace applications are required to adopt a Cloud architecture engineered for security and performance. One of the major hurdles of formal adoption of Cloud solutions in the enterprise is performance. Enterprise applications (e.g., SAP, SharePoint, Yammer, Lync Server, and Exchange Server) require a mechanism to predict and manage performance expectations in a secure way. This research addresses two areas of performance challenges: Capacity planning to ensure resources are provisioned in a way that meets requirements while minimizing total cost of ownership; and optimization to authentication protocols that enable enterprise applications to authenticate among each other and meet the performance requirements for enterprise servers, including third party marketplace applications. For the first set of optimizations, the theory was formulated using a stochastic process where multiple experiments were monitored and data collected over time. The results were then validated using a real-life enterprise product called Lync Server. The second set of optimizations was achieved by introducing provisioning steps to pre-establish trust among enterprise applications servers, the associated authorisation server, and the clients interested in access to protected resources. In this architecture, trust is provisioned and synchronized as a pre-requisite step

to authentication among all communicating entities in the authentication protocol and referral tokens are used to establish trust federation for marketplace applications across organizations. Various case studies and validation on commercially available products were used throughout the research to illustrate the concepts. Such performance optimizations have proved to help enterprise organizations meet their scalability requirements. Some of the work produced has been adopted by Microsoft and made available as a downloadable tool that was used by customers around the globe assisting them with Cloud adoption.

Table of Contents

ABSTRACT **2**

LIST OF TABLES **6**

LIST OF FIGURES **7**

CHAPTER 1: INTRODUCTION **8**

1.1 MOTIVATION 8
1.2 RESEARCH METHODOLOGY 10
1.3 SUMMARY OF CONTRIBUTIONS 11
1.4 PLAN OF THE RESEARCH 13

CHAPTER 2: LITERATURE REVIEW **15**

2.1 CAPACITY PLANNING 15
2.1.1 Introduction *15*
2.1.2 Literature Review of Capacity planning in the Cloud *16*
2.1.3 Dynamic Capacity Allocation *19*
2.1.4 Static Capacity Allocation *21*
2.2 CLOUD AUTHENTICATION 24
2.2.1 Introduction *24*
2.2.2 Analysis of Single-Sign-On (SSO) and Federated Identity Management (FIM) *26*
2.2.3 OpendID *31*
2.2.4 SAML *35*
2.2.5 OAuth 2.0 ..... *39*
2.3 CONCLUSION OF LITERATURE REVIEW 45

CHAPTER 3: CAPACITY PLANNING USING MODALITY COST ANALYSIS **47**

3.1 INTRODUCTION AND MOTIVATION 47
3.2 MEDIA APPLICATIONS PERFORMANCE 48
3.3 MODALITY COST ANALYSIS (MCA) METHODOLOGY 49
3.4 EXPERIMENTS AND RESULTS 54
3.5 VALIDATION METHODOLOGY 59
3.6 HARDWARE BENCHMARKS 67
3.7 CONCLUSION 70

CHAPTER 4: LYNC SERVER CAPACITY PLANNING CALCULATOR **72**

4.1 OVERVIEW 72
4.2 USING THE CAPACITY CALCULATOR 74
4.2.1 Instant Messaging and Presence *76*
4.2.2 Enterprise Voice *76*
4.2.3 Conferencing *77*
4.2.4 Voice Applications *78*

4.3 Adjusting For Processors' Variations 80
4.4 Example Calculating Needed Resources 81

CHAPTER 5: PERFORMANCE OF CLOUD AUTHENTICATION 87

5.1 Introduction to Cloud Authentication 87
5.2 OAuth Optimization for Enterprise Adoption 89
5.3 Case Study 92
5.4 Conclusion 97

CHAPTER 6: FEDERATION IN THE ENTERPRISE 98

6.1 Introduction to Cloud Federation 98
6.2 Marketplace as a Service MaaS 100
6.3 Enterprise Federation 102
6.4 Federation Case Study 106
6.5 Conclusion 115

CHAPTER 7: CONCLUSION 116

7.1 Summary 116
7.2 Limitations & Future Work 117

REFERENCES 118

List of Tables

Table 1: Main Authentication Protocols 26
Table 2: Authentication Protocols 44
Table 3: Experiment I Results 56
Table 4: Experiment II Results 57
Table 5: Experiment III Results 57
Table 6: Experiment IV Results 58
Table 7: MCA-S Experiment Results 63
Table 8: MCA-M Experiment Results 64
Table 9: MCA-L Experiment Results 65
Table 10: Benchmarking example modalities 69
Table 11: Server CPU Capacity Calculations 82
Table 12: Audio/Video CPU Capacity Calculations 85
Table 13: Resource Server (SharePoint) Table 93
Table 14: Authorisation Server (AS1) Table 93
Table 15: Client Table (SAP) 94
Table 16: The Modified Protocol Access Token Parameters 96
Table 17: Authorisation Server (AS1) 107
Table 18: Access token parameters for the modified protocol 109
Table 19: Authorisation Server (AS1) Table Supporting Identity Federation 110
Table 20: Results of Case Study 112

List of Figures

Figure 1: OpenID Protocol Flow 33
Figure 2: SAML Protocol Flow 38
Figure 3: OAuth 2.0 Protocol Flow 42
Figure 4: CPU utilization 50
Figure 5: Memory utilization 51
Figure 6: Bandwidth utilization 52
Figure 7: Lync Server Stress Tool GUI 60
Figure 8: CPU average for each server 62
Figure 9: Lync Server Capacity Planning Calculator 75
Figure 10: OAuth 2.0 Protocol Flow 90
Figure 11: OAuth 2.0 Modified Flow 91
Figure 12: UML Sequence Diagram of a marketplace application 104
Figure 13: Overall view of the proposed authorisation model 106

Chapter 1: Introduction

1.1 Motivation

Adoption of Cloud computing model can only become a viable alternative for large enterprises if these infrastructures can provide proper levels of performance that assure quality and guarantee service level agreements are met. An enterprise that focuses on moving to the Cloud needs to know and apply the proper configuration to provide the right levels for individual services if they are deployed in the Cloud (Noureddine & Bashroush, 2011). Security, availability, performance, and privacy are examples of many metrics that are important for an enterprise to ensure a successful migration to the Cloud. This research covers an important metric necessary for Cloud adoption—performance optimization to guarantee service license agreements (SLAs). In order to guarantee performance SLAs, service providers in the Cloud tend to over-provision expensive resources. This is done mainly due to the lack of tools that guide such optimization of performance and cost, and SLA violations are costly for Cloud-hosted applications. This research addresses these issues by providing a methodology to help enterprises to make informed decisions with respect to the right level of resource provisioning and optimizing server-to-server authentication mechanisms for Cloud authentication and federation.

The first research area demonstrates a tested methodology to guide resource provisioning decisions for datacentre providers. In this research, a systematic methodology is presented

to approximate the performance predicted from each modality (e.g., instant messaging and voice calls). The methodology is based on the representation of resource cost per modality. Subsequently, the research presents how the estimate of the expected application performance could guide resource provisioning decisions. The methodology is illustrated through various case studies using a commercially available media application, the Microsoft Lync Server.

Secondly, in addition to capacity planning, in order to further encourage enterprise adoption of Cloud services, providers need to ensure adequate and secure architecture is implemented with minimal performance overhead. Today, most large Cloud providers use OAuth (Open Authentication) to enable authentication in the Cloud. OAuth protocol was published in December 2007 and quickly became the industry standard for web-based access delegation (Yang & Manoharan, 2013), (Pai, et al., 2011). However, OAuth faced many challenges to enter the enterprise domain, mainly due to the lack of performance capabilities currently offered by the protocol. In this research, optimization to OAuth is introduced where the Authorisation Server is provisioned with an explicit authorisation table so that servers can act on behalf of users and reduce per-client round trip to OAuth authorisation server. This reduces the amount of processing for some popular protected resources and alleviates the risk of potential threats such as Denial-of-Service (DoS) attacks (Symantec, 2010) and Distributed DoS (DDoS) attacks (Chao-yang, 2011)

With the combination of optimal hardware resource provisioning and reduced authentication overhead, Cloud providers can minimize the cost, guarantee SLAs, and secure access to data resources, all of which are key aspects for enabling better Cloud adoption in the Enterprise.

1.2 Research Methodology

In the early stages of this research, a hypothesis is developed to solve the issues related to performance for enterprise adoption of Cloud computing. The hypothesis calls for a methodology to estimate hardware resources cost for cloud applications to ensure service level agreements are met.

Two sets of experiments are performed. In the first set of experiments, the resource overhead for four modalities are measured in isolation, namely instant messaging, Voice over IP (VoIP), application sharing conference, and address book download. In the second set, the resource overhead for three scenarios that combine all of the four modalities together are measured simulating a real end user experiment. The second set of experiments is used to validate the hypothesis. Each set of experiments are run for several hours where data is collected every hour. The hardware for both set of experiments were fixed: A server with dual processors quad-core 2.0 GHz (2,000 megacycles per second), 16 gigabytes of memory, 30GB disk space, and 2-port 1 gigabit per second network adapter.

The results of the experiments showed that there is possible to calculate capacity using the developed methodology. To further validate the methodology, Office Lync Server is

deployed on the fixed experiment hardware server, and a simulation tool representing users accessing the server is developed for these experiments. The results show that the methodology is valid for production software.

1.3 Summary of Contributions

The key contributions of this research are in presenting methodologies that will allow the Enterprise to adopt Cloud Computing, the contributions are summarized in four different areas:

1. Develop a methodology for capacity planning to estimate the resources needed of the Cloud service providers, this will reduce total cost of ownership and guarantees service level agreements are met.
2. Develop the Lync Server Capacity Calculator as a tool to guide the resources needed to host Lync Server in the Cloud.
3. Optimization to OAuth authentication mechanisms to make it more suitable for Enterprise adoption, with focus on server-to-server authentication. Such optimization will enable large enterprise organizations to meet the performance and scalability expectations when hosting hundreds of millions of users.
4. Optimization to authentication mechanisms to enable cloud federation with focus on marketplace (third party) applications. Such optimization will enable large

enterprise applications to meet performance expectations when serving large numbers of third party applications across multiple federated organizations.

The following is a list of publications made on the findings of this research:

1. M. Noureddine, R. B. (2013). An Authentication Model towards Cloud Federation in the Enterprise. Journal of Systems and Software.
2. M. Noureddine, R. B. (2011). A Performance Optimization Model towards OAuth 2.0 Adoption in the Enterprise. Proceedings of the 7th International Conference on Global Security, Safety & Sustainability (ICGS3). Greece.
3. M. Noureddine, R. B. (2011). A provisioning model towards OAuth 2.0 performance optimization. IEEE 10th International on Cybernetic Intelligent Systems (CIS), (pp. 76-80).
4. M. Noureddine, R. B. (2011). Cost Effective Datacentre Capacity Planning Analysis Using Modality Cost Methodology. Ubiquitous Computing and Communication Journal (UBICC).
5. M. Noureddine, R. B. (2011). Modality cost analysis based methodology for cost effective datacentre capacity planning in the cloud. Special Issue on the International Conference on Information and Communication System, ICIC.
6. M. Noureddine, R. B. (2011). Modality Cost Analysis: A Methodology for Cost Effective Datacentre Capacity Planning in the Cloud. International Conference on Information and Communication systems (ICICS).

7. R. Bashroush, M. N. (2012). A Cost Effective Cloud Datacentre Capacity Planning Method Based on Modality Cost Analysis. International Journal of Communication Networks and Distributed Systems.
8. R. Bashroush, M. N. (in review). Predictive Cloud Service Management for Optimizing Energy Efficiency: A Modality Based Approach, IEEE Transactions on Network and Service Management (in review)
9. M. Noureddine, R. B. (in review). Capacity Planning Analysis and Workload Prediction for Lync Server, IEEE Transactions on Dependable and Secure Computing (progressed to second round of reviewing)

1.4 Plan of the Research

The thesis is structured in the following way:

Part I: Introduction

- Chapter 1: Problem statement and contribution
- Chapter 2: Literature Review

Part II: Capacity Planning

- Chapter 3: Capacity planning using modality cost analysis methodology
- Chapter 4: The Lync Server capacity planning calculator

Part III: Cloud Authentication Optimization

- Chapter 5: Performance of Cloud authentication
- Chapter 6: Federation in the Enterprise

Part IV: Conclusion

- Chapter 7: Conclusion and Future Work

Chapter 1 introduces the challenges with enterprise adoption of Cloud computing, summarizes the contributions and lists out the recent publications. Chapter 2 covers the literature review and organizes them into two main subsections, literature review for the capacity planning research area and literature review for Cloud authentication protocols and includes a survey of most popular authentication protocols including their strengths and weaknesses. Chapter 3 covers the methodology developed in this research to help Cloud providers ensures the optimal level of resources are used to serve the needs of their hosted customers while Chapter 4 illustrates the Capacity Planning Calculator tool developed out of this research to help guide the resource allocation for Lync Server. Chapter 5 discusses the performance of Cloud authentication and the optimization developed in this research to address the enterprise challenges with adopting authentication protocols. Chapter 6 builds on chapter 5 and extends the optimization to cover cloud federation for marketplace applications. At last, chapter 7 provides conclusion remarks with a summary and future work.

Chapter 2: Literature Review

2.1 Capacity Planning

A series of data centre capacity planning methods have been proposed to minimize resource utilization while guaranteeing service level agreements (SLAs). The goal of this research area is to help Cloud service providers to provision hardware resources while maintaining performance expectations. This literature review considers current approaches to Cloud capacity planning schemes. This consideration is beneficial to Cloud providers interested in reducing total cost while guaranteeing performance expectations.

2.1.1 Introduction

Cloud computing provides an advantage for companies and institutions that rely on a grand scale IT operations in a cost effective way. The early Cloud-based models have enabled the enterprises to access more computing resources and more services at an attractive price model to reduce total cost of ownership (TCO) (Sukumar, 2011). The Cloud architectures are designed in a way that multiple services as IaaS (Infrastructure as a Service), PaaS (Platform as a Service), and SaaS (Software as a Service) are provided to a large set of consumers in a shared set of hardware resources. In Cloud models, the jobs initiated by the customers are

allocated with a set of physical or virtual servers that run in the datacentres. These servers are available in different types with varying capabilities such as number of processors, different network bandwidth, different ranges of memory, and different storage capacity. The capacity planning process is one of the modern fields of interest for Cloud computing, and it is used for estimating the efficiency of the Cloud operations. Ad-hoc or intuitive resource allocation processes can drive the Cloud network towards an operational failure due to missed SLAs or wasted resources due to over-provisioning. Therefore, it is the responsibility of Cloud service providers to ensure appropriate resources are allocated to each customer to guarantee acceptable performance of their products. Many Cloud providers misunderstand the relationship between capacity planning and performance tuning (Allspaw, 2008). While they affect each other significantly, they have different goals. Performance tuning optimizes an existing system for better performance, while capacity planning determines what the system needs while maintaining the performance baseline. In this chapter, a summarized number of the well-known capacity planning models are presented, and an evaluation of the known strength and weaknesses in the context of capacity planning for Cloud providers are discussed.

2.1.2 Literature Review of Capacity planning in the Cloud

Cloud computing offers the facility to utilize shared hardware and software resources and common infrastructure, offering services on demand over the network to execute

operations that meet changing and evolving business requirements. The location of physical resources and devices being accessed are typically not known to the end user. Cloud providers such as Amazon EC2 (IaaS), Azure (PaaS), and Google (SaaS) are transforming labour intensive, hard-coded systems into shared, automated, and fully managed adaptive services which promise great opportunities for reducing energy and hardware costs (Lenk, et al., 2009). These providers need to provision hardware resources to meet required capacity. Required capacity is the minimum amount of capacity needed to satisfy resource demands for workloads on a server resource (Rolia, et al., 2005). An important issue is how various server resources may be allocated to an application such that the service level agreements (SLAs) are met while minimizing the cost. Using heuristic and intuitive methodology usually leads to more resources than are actually necessary. While such over-provisioning may guarantee performance, this guarantee may come at a very high cost. A capacity planning model may guide the Cloud service provider in making informed decisions about the right level of resources, so that acceptable service performance may be provided in a cost-effective manner. There are various research areas that provide guidance for Cloud providers to allocate the right capacity and minimize hardware resource utilization. They can be broken into two main areas, 1) dynamic capacity allocation (Simmons, et al., 2007), (Ye Hu, 2009), and 2) static capacity allocation (Zhu, et al., 2008), (Daniel Gmach, 2007), (Madhukar R. Korupolu, 2009), (Hasselmeyer & d'Heureuse, 2011).

In dynamic capacity allocation, the resources are allocated dynamically as needed and demand peaks, while, in statistic allocation, the needs are pre-estimated and expected resource requirements are preconfigured until the requirements or demand of the application is planned to change. For example, if an application is expected to have a response time of less than *t* seconds and if the workload is expected to increase then the required computing resources will increase accordingly to ensure response time remains < *t*. One approach for ensuring the satisfaction of the computing needs of a particular application is to provide enough resources for the anticipated peak demand (static allocation). Alternatively, in dynamic allocation, the resources are dynamically allocated as needed and de-allocated from a hardware resource when the demand for the applications is decreased.

Both dynamic and static allocation models research areas cover virtualization and virtualization overhead such as in (Chris Matthews, 2009), (Yexi Jiang, 2013), and shared resource pools vs. dedicated resource pools (Zhiliang Zhu, 2011) (Menasce & Bernnani, 2006). The next sections analyse various dynamic and static capacity planning research areas.

2.1.3 Dynamic Capacity Allocation

Every Cloud provider facility, composing of data centres, servers, tooling and fixtures, etc., has a limited amount of capacity. Effective consumption and management of production resources have significant implication to the profitability of the organization. Capacity planning helps in laying the base for all resource optimization activities. However, due to the fluctuation of' demand pattern, a Cloud provider may experience unexpected utilization caused by fluctuating demands. The ability to response to the dynamic demand and product mix changes has become a key competitive advantage, even at the short term and long term capacity planning arena. Much of the research in this field is at its early phases with respect to the understanding of effective allocation of the dynamic resources for optimal utilization in Cloud data centres. Yagiz Onat Yazir et.al (Yagiz Onat Yazir, 2010), have proposed a new approach for dynamic independent resource management in Cloud computing. It is a two-phase work. In the first phase, the resource management is allocated into independent tasks, which is done by independent node agents which are lightly coupled with the physical machines in the data centre. In the second phase, the autonomous node agents run the configuration in parallel through multiple threads that process decision analysis. Their approach claims to have a positive reflection on feasibility, scalability, and flexibility. T.R. Gopalakrishnan et.al (Gopalakrishnan & Vaidehi, 2011) developed a Rule Based Resource allocation model (RBRAM). It is a time-marching model that processes Supply-Demand analysis of the resources. These analyses have shown improved performance of the system,

achieved through allocating the right resources to the Cloud at the right time. Daniel Warneke and Odej Kao (Warneke & Odej, 2011) have designed a data processing framework and have compared it with other data processing framework. The research they proposed clearly takes advantage of the dynamic resource allocation offered by Infrastructure as a Service (IaaS) Cloud for task scheduling and execution. Jiayin et.al (Jiyin Li, 2010), have proposed a Cloud capacity planning system that uses adaptive resource allocation with preconfigured tasks. The algorithms in this research adjust the resource allocation dynamically based on the progress of the actual task completion. Therefore, they are able to improve utilization by predicting next in line task requirements. Graham Kirby et.al (Graham Kirby, 2010) designed an ad hoc Cloud model. Their research shows that the proposed model is performance resilient and self-managing where it can balance potentially conflicting objectives. The authors claim that their ad-hoc Cloud model allows complex Cloud style applications to utilize unused resources on hardware allocated in the datacentre. Another area of research on dynamic capacity allocation looks at specific application type. Peter Bodík et.al (Peter Bodík, 2009) research discusses methodologies for training and tuning data centre performance. It focuses on online training without violating the SLA in place. Mainly, it identifies the challenges in offline training, for example, not necessarily reflecting the capacity of application in production, and that the performance profile changes regularly because of changes of how applications are used. In this research, the authors propose to train the performance model using an exploration method that quickly collects data from various performance regimes of the application. The methodology

claimed in the research provides an exploration process to strike a balance between not violating the performance SLAs and the need to collect sufficient data to train an accurate performance model, which requires pushing the system close to its capacity. By using this exploration policy, they can train a performance model of a Web 2.0 application in less than an hour and then immediately use the model in a resource allocation controller. In this research, analysis is provided of how the resources in a Cloud computing infrastructure may be managed in a cost efficient manner to reduce the total cost of ownership, similar research focuses on e-Commerce applications exclusively (Gokhale, 2008), (Lu & Gokhale, 2006).

2.1.4 Static Capacity Allocation

Static allocation of hardware resources is a method for forecasting resources demands upfront. This method calls for a solution that ensure SLAs for applications in changing datacentre conditions while hiding the complexity from the owners or administrators of the datacentre and the application owners. Service providers, today, achieve this through virtual resources or physical resources. Virtualization technologies promise great opportunities for reducing energy and hardware costs through server consolidation (Song, et al., 2008) (J. Zhang, 2007). In addition, virtualization can optimize resource sharing among applications hosted in different virtual machines to better meet their resource needs. However, to safely transition and correctly estimate and predict the required capacity, service providers need to estimate the additional resource requirements incurred by virtualization overheads. In (Wood, et al., 2008), the authors provide a model for such estimation. In this research, they

design a methodology for estimating the additional requirements when transferring an application to virtual servers. The methodology has two key components: A set of micro-benchmark to assess the various types of overhead caused by virtualization on a given configuration, and another model to assess regression when virtualization is used. Their experiments show impressive results predicting requirements within 5% median error. The focus of their research is on CPU requirements due to the virtualization overhead. These findings provide a methodology for Cloud providers interested in virtualizing datacentres to assess additional capacity needed to make a move to the Cloud without excessively over-provisioning resources.

In (Zhu, et al., 2008), the authors look at analysing various workload requirements per application. In this research, the authors consider the issues of workload analysis, capacity planning, and performance modelling with a goal to automate the use of resource machines that are serving a large numbers of enterprise services. They provide a three tier approach that relies on the following:

1. The classification of workload demand configurations
2. The simulation of synthetic workloads that help predict future demands based on the configurations
3. A capacity placement recommendation service.

The accuracy of capacity planning predictions depends on the ability to depict capacity demand patterns, to recognize trends so that future demands are predicted, and to align business forecasts so that future demands are well understood. A capacity analysis

determines the peak and repetitive nature of enterprise expectations. Resources are automatically classified according to their expected behaviour based on the periodic model. The repeated occurrences are evaluated for similarities. Synthetic transactions and workloads are produced in a manner that mimics peak, periodic nature, and trending behaviour of the various resources. In addition, the authors illustrate these analysis in a case study that involves running 6 months of data for 139 enterprise applications used to apply and evaluate the enterprise capacity analysis and related capacity planning approaches. The results show that such approach is great for predicting the required capacity with little risk.

In (Bashroush & Noureddine, 2012), the research area provides a quantitative measurement of the resource cost (CPU, memory, storage, and network bandwidth) imposed by each of the modalities of the application, in isolation, allowing organizations to make informed decisions with respect to the right level of resource provisioning. The objective of the research is to illustrate a tested methodology to guide resource provisioning decisions. It presented a systematic methodology to estimate the performance expected from each modality based on the representation of resource cost per modality. Subsequently, the research discussed how the estimate of the expected application performance could guide resource provisioning decisions. The research illustrated the methodology using a case study of commercially available media application, the Microsoft Lync Server 2010 (Noureddine & Bashroush, 2011), (Noureddine & Bashroush, 2011), (Noureddine & Bashroush, 2011). Then it validated the performance estimation and resource provisioning methodology using a validation software tool to simulate a realistic workload against a production datacentre

with all the modalities working together. The results can guide providers into provisioning datacentres for optimizing performance and cost. By profiling the application into a set of modalities and measuring hardware resources cost in isolation, Cloud providers should be able to pinpoint their capacity to exact needs without wasting expensive resources. The experiments provided in this research represented various application profiles. The results showed that measuring modalities in isolation and using the results to provision datacentres is an effective methodology. The research also discussed the process for applying hardware benchmarks for scenarios where experimental hardware servers differ from deployment hardware or for upgrading hardware servers without invalidating experimental results. The challenge with shared CPU resources has been reviewed in (Sean Kenneth Barker, 2010) and (Jones, et al., 1997) extensively with profiling the CPU utilization in a shared resource pool. This research emphasized the opportunity to profile CPU utilization in a shared manner.

2.2 Cloud Authentication

A number of authentication protocols have been used over the years to enable Cloud authentication. These protocols have strengths and weaknesses that make them desirable for some organizations while an impediment for other organizations. This section analyses the most popular ones and discusses their strength and weaknesses.

2.2.1 Introduction

Single Sign On (SSO) or federated identity means linking the electronic identities across several authorisation servers (Yebin Chen, 2011), (Manshan Lin, 2001). Federated identity

management (FIM) systems are the systems in place to enable such linking (Ernest & Foo, 2009). Simply, applications and servers do not necessarily need to own and store users' credentials in order to authenticate them. Instead, they can use an identity management system that is already storing a user's electronic identity to authenticate the user, given, of course, that the user has the right to access the resources. For providers of protected resources, this is great as it relieves it from having to manage the identity of every user. It is also very convenient for users, since they don't have to keep a set of usernames and passwords for every single application that they use. There are many protocols for federated identity such as OpenID, SAML, and OAuth. These protocols are developed by three main body standards, namely OASIS, W3C, and the IETF. They all provide standards that cover current identity management protocols that combine multiple configurations. OASIS supplies SAML and the Web services (WS-* (IEFT, 2010), (IETF, 2010)) suite of standards. W3C provides HTTP architecture, URIs, and the service-related SOAP that are leveraged by federated and distributed identity solutions. The IETF body of standards provides several relevant standards, including the Transport Layer Security (TLS), Simple Authentication and Security Layer (SASL), and Public-Key Infra- structure (PKIX) along with OAuth. Below in Table 1: Main Authentication Protocols is a summary of these standards and their main protocols:

Table 1: Main Authentication Protocols

Standards Body	Protocols
OASIS	SAML, WS*, IMI
W3C	HTTP, URIs, SOAP
IEFT	SASL, TLS, PKIX, OAUTH

In this section, a survey of the three main ones, OpenId, SAML, and OAuth along with analysis of SSO (Single Sign On) and Federated Identity Management (FIM) services are discussed.

2.2.2 Analysis of Single-Sign-On (SSO) and Federated Identity Management (FIM)

SSO (Single Sign On) is a service in which the user authenticates once at home identity provider (IDP) and log in to access consecutive services within the federated service providers. This area is covered in much research over the years. In (Madsen, et al., 2005), Madsen et al. addressed multiple problems of federated identity. The research illustrated that Federated Identity Management (FIM), which is architected on numerous industry standards, streamlines the processes used by federated organizations in term of simplifying the authentication configuration experience, sharing user identity objects, and accessing various permissions using credentials requirements. However, Madsen illustrated the ongoing problems and challenges in a federated identity environment such as exploitation of user identity information through user's identity theft, single sign on capability, and

trustworthiness of the user. Problems and concerns of identity management are too many, among others, most worthy of mention is users having to provide their credentials to many service providers, leading to opportunities for the users' credentials to be compromised while executing the federated identity. In (Layouni & Pollet, 2009), Rodriguez et al. demonstrated Federated Identity Architecture as a technique for resolving exposures. The research identified the three methods for implementing security issue in Federated Identity Architecture including Shibboleth, Liberty Alliance, and WS- Federation (Layouni & Pollet, 2009). In (Archer, et al., 2011), Archer et al. argued that the most common attacks are the theft of users' identity. Identity theft happens on the least secure channels while it is very difficult to identify until the harm is done. Besides identity attacks, legal compliance is another security problem in federated identity; the research does not address the enterprise effect on such vulnerability. In (Eghbal Ghazizadeh, 2012), Eghbal et al. recognized five

security problems related to federated identity concerning relationships between vendors and consumers, these five security issues are:

1. Communication with a Human Resources system (HR) is tough where HR is the only master source for employees' identities.
2. Identities in partner organizations cannot be verified in an authoritative way.
3. Most organizations do not support federated identity.
4. The identity service provides the self-asserted identity for consumers yet it does not cover the rest of identity types.
5. Organizations do not have a way to communicate directly, but rather federate or proxy their identity services

These issues emphasize the need for good planning and to handling how identity attributes, accounts, and lifecycle management of all identity-types will operate in the Cloud (Archer, et al., 2011). Suriadi et al. in (Suriadi, et al., 2009) identified that one of the main problems with these identity models is privacy in an SSO environment, relying parties (RP) or service providers (SP). All of these three services hold important and secure user identities. These services can also be collecting information about users from various identity providers. Suriadi et al. also analysed sharing of user's information by malicious identity providers and service providers which can disclose a complete user's identity and activities. Related to these vulnerabilities, Zarandioon et al. in (Zarandioon, et al., 2009) shows that this issue has

caused web users to be careful of SSO implementations, the research claims that this is one of the main reasons for the lack of widespread adoption. In (Wang, et al., 2012), Wang discussed another problem of federated identity that is in the process of switching authentication mechanisms to a single sing on solution. Such switch requires education of the users and possible dissatisfaction if the transition to SSO is not smooth. What makes the situation worse is the lack of demand from users as studies have shown that users are already satisfied with their own password management.

Additional research on single sign on (SSO) studies the security implications of various authentication protocols. Somorovsky et al. in (J. Somorovsky, 2012) investigated fourteen implementations of SAML protocol and they founded many security issues that related to Extensible Mark-up Language (XML) signature wrapping. These implementations used SAML assertions for making security statements between subjects. Therefore, they developed an automated tool to penetrate XML testing of signature wrapping, their research did not address performance implications, rather focused exclusively on security penetration.

Wang (Wang, et al., 2012) considered alternative solutions to SSO when he analysed Privacy Aware Identity Management and Authentication for the Web (SAW) as viable alternatives. He highlighted vulnerabilities and security issues in the three common identity systems, namely, Microsoft Passport, OpenID and SAML (Wang, 2011). Rodriguez et al. (Layouni & Pollet, 2009) focused on Federated Identity Architecture (FIA) and analysed some of the problem related to it. Furthermore, they explored commercial Federal Identity Architecture solutions and investigated their security and privacy issues. Yan et al. (Yan, et al., 2009)

suggested a cryptography based federated identity with some important features to make it Cloud friendly and adaptable to the Cloud environment. They synchronized hierarchical identity-based cryptography with federated identity management in the Cloud. However, such research does not address specific values to increase adoption of SSO in the Cloud.

In (Li-chun, et al., 2008), the authors suggested a dynamic federated-domain authentication scheme based on credit worthiness. The trust relationship is dynamically formed based on the value of the credit. Based on this relationship, federated-domain authentication could be established. However, the calculations need to be processed by a trusted third party which can become bottlenecked during the authentication execution. In (Zhen-Guo, et al., 2009), the authors put forward an enhanced dynamic authentication process based on digital signature. The process can reduce how often the two parties communicate in order to authenticate, and thus improving the availability of the network resources. But the federated domains were not considered in this paper, so it was more applicable to single-domain, thus such protocol is not considered adequate for large-scale distributed computing environment.

Performance is yet another open challenge for organizations planning to adopt a security protocol to achieve SSO. These areas have been analysed in different research areas. The majority of the research have compared the performance of different tools that tune the performance. In (Ferraz, 2005) server SOAP based tools were profiled to assess SOAP inefficiency to pin point the features of SOAP that affect performance the most. Another

study (Gray, 2004) looked into the performance of other middleware such as Java RMI and CORBA. This research has illustrated that the performance of web services is a major performance bottleneck, especially when applying reliability and security assertions. In security related studies, Moralis et al. (Moralis, et al., 2007) have compared the performance of SSO requirements through X.509 Token Profile against Kerberos Token Profile. Liu et al. (Liu, et al., 2005) have performed several tests for different operations such as Encryption vs. Decryption and Signing vs. Verifying using multiple various algorithms to assess their performance capabilities for these operations. The study of Tang et al. (K. Tang, 2006) has compared the performance of WSS Encryption and WSS Signing. Shirasuna et al. (Shirasuna, et al., 2004) have assessed security methods for grid services. Their assessment has shown that message level security is slower than transport level security, and should be used if there is no additional requirement to use message level security. In (Sun Microsystems, 2010), WSIT (Web Service Interoperability Technology) is evaluated for the opportunity to improve performance for WS* protocols, and its tight integration with WCF, however, it is relatively new, and there is no published research that conducted performance assessment of applying WSIT Security Mechanisms at this time.

2.2.3 OpendID

OpenID is an open standard adopted by Microsoft, Facebook, Google, Symantec, PayPal, Yahoo, and Ping Identity, among many others. OpenID allows clients to be authenticated using third-party services called IDPs or identity providers. Clients can pick their choice of

OpenID providers to log in to protected services that accept the OpenID authentication model.

The OpenID specification defines three roles:

1. The client or the user that is looking to verify its credentials
2. The entity looking to verify the identity of the client, known as the relying party (RP)
3. The entity that registers the OpenID URL and can verify the end client's identity, known as the OpenID provider (OP)

For OpenID to be configured correctly, every single client or user in a client needs to have its OpenID configured or pre-registered. SAML and OAuth are able to sign in users based on their email address or username, which makes it much easier to administer. Consider, for example, an application that serves 10k users. To enable SSO via OpenID, it would be required to preconfigure the correct OpenID settings for each of the users of the application. To enable SSO via SAML or OAuth, all that's required is to configure a couple of URLs that are applicable to all users.

One of the important advantages of OpenID is the ability to auto-discover the identity provider. For example, the OpenID https://identityprovider1.com/user1/ contains the discovery information, in other words, that identity provider can be found at https://identityprovider1.com. This is a great feature for consumers because it makes

configuring an app for SSO very simple, but it becomes a drawback in a large enterprise context. There are also many Cloud-based OpenID identity providers to choose from which can create a wild-wild-west environment with no scrutiny of who can become an identity provider.

The following diagram Figure 1: OpenID Protocol Flow explains a use case for an OpenID scenario:

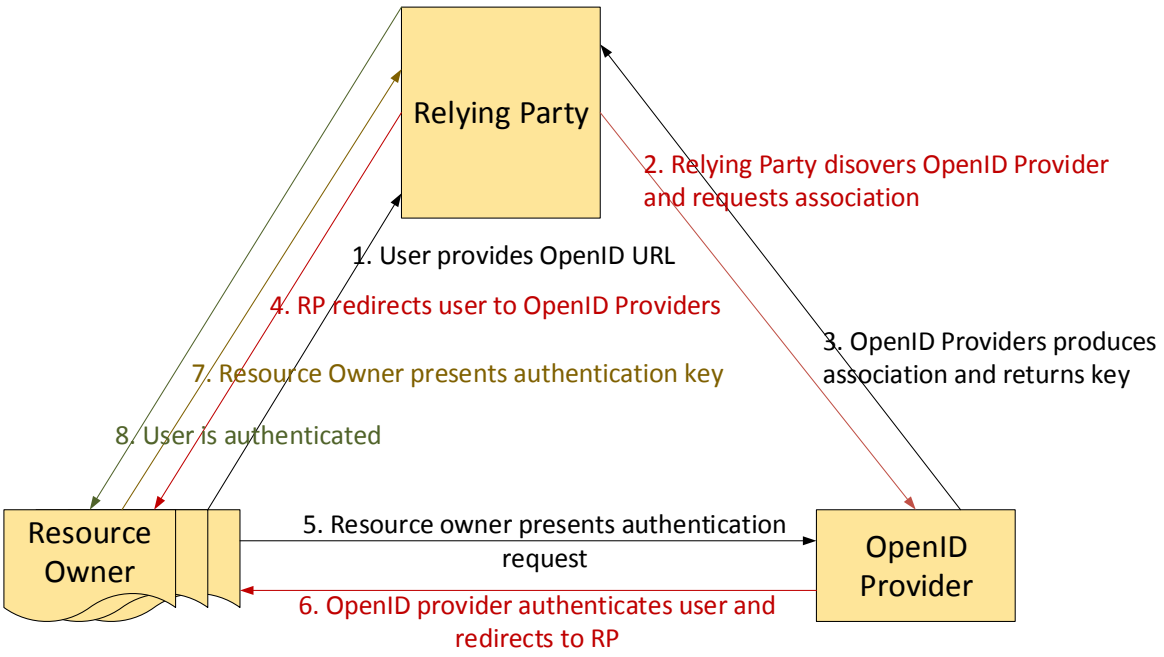


Figure 1: OpenID Protocol Flow

Though OpenID started as a very promising protocol, it failed to make a large impact. Many enterprises rushed into adopting OpenID by becoming an OpenID provider but declined to become a relying party. For example, Microsoft soon became an OpenID provider allowing its users (for example, a Hotmail user can use Hotmail credentials to login to other websites) to obtain an access token for logging into a relying party. However, most enterprises did not become relying parties to other OpenID providers (in the example of Hotmail, users were not allowed to login to Hotmail using third party OpenID tokens). Mostly due to security and performance concerns with OpenID.

OpenID can be described as being a user-centric rather than a site centric approach to identity management (Jacob Bellamy-McIntyre, 2011). Unlike OAuth, as also illustrated in Figure 1: OpenID Protocol Flow above, each user needs to obtain an access token. While in OAuth with some optimization, one token can be presented on behalf of other users (server level trust). What is more favourable, however, about OpenID compared to other identity protocols such as OAuth, is that the identity provider does not require any pre-established relationship with the web service or website for which it is providing authentication. Users choose their own OpenID identity providers who provide them with a unique URL that represents their identity. They then supply this URL to the relying party site that supports authentication with OpenID. This means that enterprises that are acting as relying parties

can leverage the authentication methodology of other identity providers and therefore reduce the time required for users to access their services.

You and Jun in (Jun & You, 2010) proposed I-PIN (Internet Personal Identification Number) method to strengthen the authentication of users with the Cloud OpenID environment to restrict phishing attacks. They compared and evaluated their method with the existing OpenID method, and came up with recommendations to secure OpenID ecosystem (Jun & You, 2010). Their method proposes that a user has to choose only 1 identity provider out of 3 identity providers, which delivers OpenID authentication in order to receive the OpenID service. A user receives I-PIN information from main Confirmation Authority via OpenID Identity Provider. In addition, they compared and evaluated their proposed method with an existing OpenID method, and confirmed that authentication was safe and secure against private information exposition and hence against phishing attacks.

2.2.4 SAML

Security Assertion Markup Language (SAML) is a product of the OASIS Security Services Technical Committee. Dating from 2001, SAML is a XML-based open standard for exchanging authentication and authorisation data between secure domains, that is, between a service provider and identity provider (Fang, et al., 2005) (Lynch, 2011) (Zhenxiang Tu, 2012) (Wang Xiuyi, 2007)

SAML assumes the principal, mostly a user but could be an application, has enrolled with at least one identity provider. The identity provider is anticipated to provide local authentication services to the principal. On the other hand, SAML is agnostic to how local authentication services are implemented, though individual service providers most likely are interested in how authentication is implemented. SAML is an extensible protocol by design since it is XML-based, which makes it a standard that is very flexible. For example, federation partners can decide to share whatever identity objects they want in a SAML assertion message payload as long as those objects can be represented in XML. Interoperability gives SAML a major advantage over other systems. Using SAML, organizations do not need to build relying party for every incoming authentication type. SAML can support single sing on with many different federated partners. Organizations who have gone through the pain of supporting single sign on for multiple providers value the benefits of SAML the most. They are now requiring the use of SAML for single sign on with Cloud services, applications and other external service providers. SAML defines XML-based assertions and protocols, profiles and bindings. The term SAML Core refers to the general syntax and semantics of SAML assertions as well as the protocol used to request and transmit those assertions from one system entity to another. SAML includes three types of assertions:

A. Authentication assertion
B. Authorisation assertion
C. Attribute assertion

Each assertion contains all the important information such as assertion id, version, issuer, and assertion creation time. The elements of assertion are used to provide the recipients of the assertion and the start time of the assertion. The element contains the important data that describes the certification object. Records such as IP address, authentication time, and authentication method, permissions, and authorisation basis are all part of an SAML element. It is important to note that SAML does not refer to how an object is transmitted but rather to what is transmitted. The 'how' part is obtained by the SAML binding. A SAML binding determines how SAML requests and responses map onto standard messaging or communications protocols. An important and synchronous binding is the SAML SOAP binding. A SAML profile is a combination of assertions, bindings and protocols. SAML is not concerned with confidentiality, integrity, or non-reputability of assertions in transit.

The SAML specifications define three roles:

1. The principal, which is usually the user looking to access a protected resource
2. The identity provider (IDP), which is the entity that is capable of authenticating and verifying the identity of the principal
3. The service provider (SP), which is the entity hosting the protected resource and interested in verifying the identity of the end user

The following Figure 2: SAML Protocol Flow shows a flow of SAML protocol:

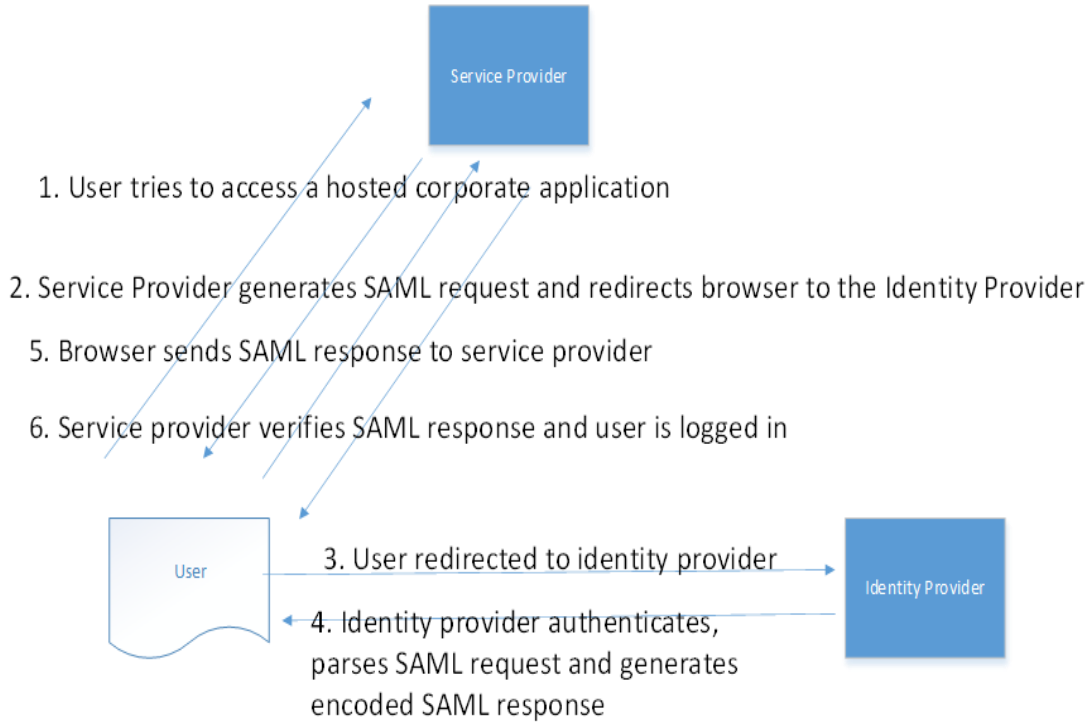


Figure 2: SAML Protocol Flow

SAML is viewed in the industry as a complex protocol and has been out favoured mainly due to the complexity in parsing the XML and the performance implications of such requirements (Menasce, 2002). The new generation of innovators viewed the Internet from inside the Web and brought a new set of languages and tools to bear on the development. This has shifted focus from XML and SOAP into lighter weight JavaScript Object Notation (JSON) and REST using HTTP architecture. Although XML can also be used in the REST model, the trend has been for a more stripped-down approach. OAuth has been implemented from the grounds upon HTTP architecture and natively supports JSON tokens.

As the interaction between service providers and requesters occurs via XML-based SOAP messages, securing web services tends to make these messages longer than they would be otherwise. Such interaction will require interpretation by XML parsers on both sides that reduces the performance of web services **(Menasce, 2002)**.

In order to minimize the drawback of performance, there are some research that were reviewed. One of them is discussed in **(Ragouzis, et al., 2008)**, an approach named as 'the Enhanced Client or Proxy (ECP)'. However, this profile refers to WAP-Gateways (Wireless Application Protocol), and it brings additional complexity and limitation to mobile application, yet there exists no performance evaluation for such approach. Another alternative approach is the virtual authentication proxy which enables communication with multiple IDPs during the authentication process **(Takeda, et al., 2006)**, but has not resulted in a complete implementation yet that can validate the performance gain.

In addition, in (K. Daniel, 2008) the performance analysis of SAML showed that the authorisation procedures need a lot of time in comparison to other processes due to the SAML specific communication sequences and resulting overhead. This makes SAML a challenging protocol for enterprises concerned with performance.

2.2.5 OAuth 2.0

OAuth **(OAuth, 2012)** (Open Authorisation) is an authentication standard for allowing users to share their private resources (e.g. photos, videos, contact lists) stored on a protected resource server without having to hand out their credentials. This allows the user to grant

a third-party site access to their information stored with another service provider, without sharing their access permissions or the full extent of their data. OAuth has become a popular choice for Cloud providers due to its simplicity. As large providers started using OAuth 1.0 (the first version of the protocol), the community of organizations interested in adopting OAuth came quickly to realize that the protocol does not scale well when it comes to performance. It required state management across different steps; temporary credentials management; and provided no isolation of the Authorisation server from the protected resource server itself. In addition, OAuth 1.0 required that the protected resource servers' endpoints have access to the client credentials in order to validate the request. This broke the typical architecture of largest providers in which a centralized authorisation server is used for issuing credentials, and a separate server is used for API calls. OAuth 1.0 required the use of both sets of credentials: the client credentials and the token credentials, which made the separation very hard (OAuth, 2012).

As the deployment of Cloud based enterprise software evolves (such as Exchange Online, SharePoint Online, and SAP to name a few), there is a growing trend for these applications to integrate with each other. In addition, many marketplace applications would highly benefit from integrating with these enterprise resources through an API over HTTP or other protocols. Often these resources require authorisation for access to such Protected Resources. The systems that are trusted to make authorisation decisions may be independent of the Protected Resources Servers for scalability and security reasons. The

OAuth Web Resource Authorisation Profiles (OAuth WRAP (IEFT, 2011)) enables a Protected Resource to delegate the authorisation task to one or more trusted authorities. Clients, which wish to access a Protected Resource, would have to first obtain authorisation from a trusted authority (Authorisation Server). Different credentials and profiles can be used to obtain this authorisation, but once authorised, the Client is provided with an Access Token and possibly a Refresh Token for obtaining more Access Tokens. The Authorisation Server typically includes authorisation information in the Access Token and digitally signs the Access Token. The Protected Resource server can verify that an Access Token received from a Client was issued by a trusted Authorisation Server and is valid using the digital signature. The Protected Resource Server can then examine the contents of the Access Token to determine the authorisation level that has been granted to the Client.

Figure 3: OAuth 2.0 Protocol Flow below shows the architecture for OAuth 2.0 with an independent Authorisation Server.

The abstract flow illustrated in Figure 3 describes the interaction between the four roles and includes the following steps:

1. The client requests authorisation from the resource owner
2. The resource owner redirects the request to authorisation server
3. The client requests authorisation grant from the authorisation server by presenting

the client credentials

4. The authorisation server validates the client credentials and the authorisation grant, and if valid issues an access token
5. The client requests the protected resource from the resource server and authenticates by presenting the access token
6. The resource server validates the access token, and if valid, serves the request.

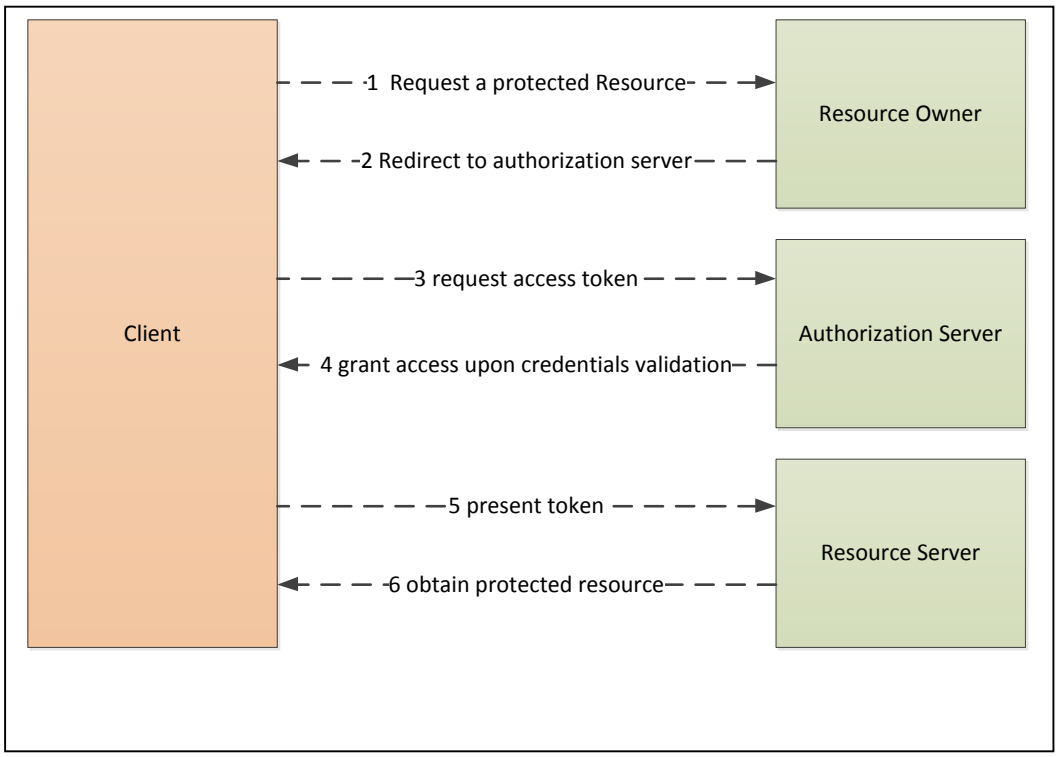


Figure 3: OAuth 2.0 Protocol Flow

OAuth specification aimed to complement OpenID and let users delegate access to a protected resource through an Authorisation Server that issues tokens that do not include users' credentials. OAuth implementation and deployment continues to grow with a lot of research currently emerging to address various limitations of the protocol. In (Marouf, 2012), the authors discuss an optimization to OAuth to enhance overall privacy. The research proposes a user-based, and category-based collaborative filtering mechanisms which are not provided by the protocol itself; however, the research does not address the performance challenges discussed in this research. In (Pai, et al., 2011), the authors formalize the OAuth protocol using a method called Knowledge Flow Analysis to discover known security vulnerability in OAuth, yet again, the research does not outline any performance issues such as the ones identified in this research paper.

Another related area of research adopts a similar optimization approach albeit using browser-based plug-ins and extensions to aid information privacy without necessarily specifying the protocol. In (Dymond, 1998), Jenkin and Dymond originally identified this approach to address the problem of "an information provider wanting to serve secrets embedded within regular webpages to authorised users."

So far, there has been no work focusing on Enterprise adoption of the OAuth protocol. This might indicate a gap between the state-of-the-art and the state-of-practice and highlights

an area where more research is needed. Some of this research are covered in upcoming chapters where the current state of existing protocol is extended to benefit adoption in the enterprise. (Noureddine & Bashroush, 2011), (Noureddine & Bashroush, 2011), (Noureddine & Bashroush, 2013)

This Table 2: Authentication Protocols explains the major differences between the three protocols:

Table 2: Authentication Protocols

	OpenID	OAuth	SAML
Dates from	2005	2006	2001
Current version	OpenID 2.0	OAuth 2.0	SAML 2.0
Main purpose	Single sign-on for consumers	API authorisation between applications	Single sign-on for enterprise users
Protocols used	XRDS, HTTP	JSON, HTTP	SAM, XML, HTTP, SOAP
Strengths	Open (no provisioning needed)	Performance	Extendable, simple to configure

Weaknesses	Performance, Security and Privacy, hard to configure	Pre-configuration requirements	Complex, time consuming to parse XML

2.3 Conclusion of Literature Review

Performance modelling in cloud computing is critical for any cloud provider to ensure appropriate provisioning of hardware resources and deployment of enterprise applications. This chapter reviewed various research areas around capacity planning and performance optimization for authentication protocols. The capacity planning research is broken into two areas, the dynamic capacity planning and the static capacity planning, both areas are reviewed with focus on current research. In addition to capacity planning, organizations need to ensure proper deployment of enterprise cloud solution in a secure and performant way. Selecting the right authentication protocol is critical to such success. This chapter reviewed the major authentication protocol with focus on current research while comparing pros and cons for each of the top three protocols, namely, OpenId, SAML, and OAuth. The chapter alluded to the lack of research on enterprise level performance evaluation. This research and the rest of the chapters cover the area of performance optimization for enterprise adoption of cloud computing.

Chapter 3: Capacity Planning Using Modality Cost Analysis

3.1 Introduction and Motivation

It is one of the responsibilities of service providers to ensure appropriate resources are allocated to each tenant to guarantee acceptable performance of their products. The relationship between capacity planning and performance tuning is often misunderstood (Allspaw, 2008). While they affect each other significantly, they have different goals. Performance tuning optimizes an existing system for better performance, while capacity planning determines what the system needs while maintaining the performance baseline. In order to guarantee performance SLAs (Service License Agreements), service providers in the Cloud tend to over provision mainly due to the lack of capacity planning tools that guide such optimization of performance and cost, and SLA violations are costly for Cloud hosted applications. A quantitative measurement of the resource cost (CPU, memory, storage, and network bandwidth) imposed by each of the modalities of the product, in isolation, may allow organizations to make informed decisions with respect to the right level of resource provisioning. In this research, tested methodology to guide resource provisioning decisions, is presented. The research presents a systematic methodology to estimate the performance expected from each modality based on the representation of resource cost per modality. Subsequently, the research discusses how the estimate of the expected application performance could guide resource provisioning decisions. A case study of the methodology

using a commercially available media application is presented using the Microsoft Lync Server 2010. Subsequently, the performance is estimated using a validation software tool to simulate a realistic workload against a production datacentre with all the modalities working together. The layout of this chapter is as follows: Section 3.2 provides an overview of media applications performance. Section 3.3 provides an overview of the Modality Cost Analysis, the research capacity planning methodology. Sections 3.4 and 3.5 present the results of performance analysis and validation tool. Section 3.6 discusses hardware benchmarks and Sections 3.7 offers concluding remarks and directions for future research.

3.2 Media Applications Performance

The performance of real-time media applications may be divided into two main categories, each categorized by the requirements of their intended applications. Conversational applications (also known as synchronous communication applications) are characterized by their stringent delay constraints, or latency, which makes it bound by the network bandwidth and processor speed. On the other hand, non-conversational applications (also known as asynchronous communication applications) are delay-insensitive as they operates in a similar way to email and bound by storage capacity. Performance analysis for media applications can be addressed from two perspectives: end-user's and service provider's perspective. A customer interacts with media applications through a series of consecutive but unrelated requests. This request sequence is termed as a session. Each session can include a combination of audio, video, instant messaging, or application and desktop

sharing. Metrics such as response time, session length, session availability, and quality of service are important from a user's perspective. On the other hand, metrics such as throughput, latency, and resource usage are important from a provider's perspective since they can guide the capacity planning and affect total cost and SLA guarantees. This research considers the performance from a provider's perspective since the focus is on capacity planning for Cloud providers (vs. consumer or end user perspective). The research addresses both synchronous and asynchronous method of communication for media application by assessing CPU, storage, memory and network bandwidth.

3.3 Modality Cost Analysis (MCA) Methodology

Modality Cost Analysis is a methodology for assessing resource cost for each of the modalities of an application (modality is a scenario in which an application is used, for example, instant messaging and voice calls are two different modalities of a media application). In this methodology, the application is broken into a set of modalities, and each is measured for resource cost (CPU, Network bandwidth, Storage, and Memory) in isolation. The first rationale behind using isolated cost analysis rather than the aggregated cost of the application in its entirety is that the workload for different modalities varies dramatically and aggregation may not capture these variations. The second rational is that Cloud providers may need to allocate resources based on their customers' user-profile. For example, when hosting communication software in the Cloud, one customer may be a heavy instant

messaging user, another may be a heavy video chat user, and a third one may be a very heavy voice customer such as a call centre. Instant messaging is CPU intensive while video and voice calls are network bandwidth intensive. Using this methodology, the service provider will be able to allocate resources appropriately and accurately for these different user profiles according to what they are going to be using.

When using modality cost analysis, resource cost is calculated separately, namely, the CPU cost, the Network cost, and the memory cost, and any other cost that might be relevant to the provider such as storage in scenarios where the application storage requirements are significant. The following graphs show the results of the experimentations after plotting the results. The next section, Experiments and Results, discusses the details of the experiments.

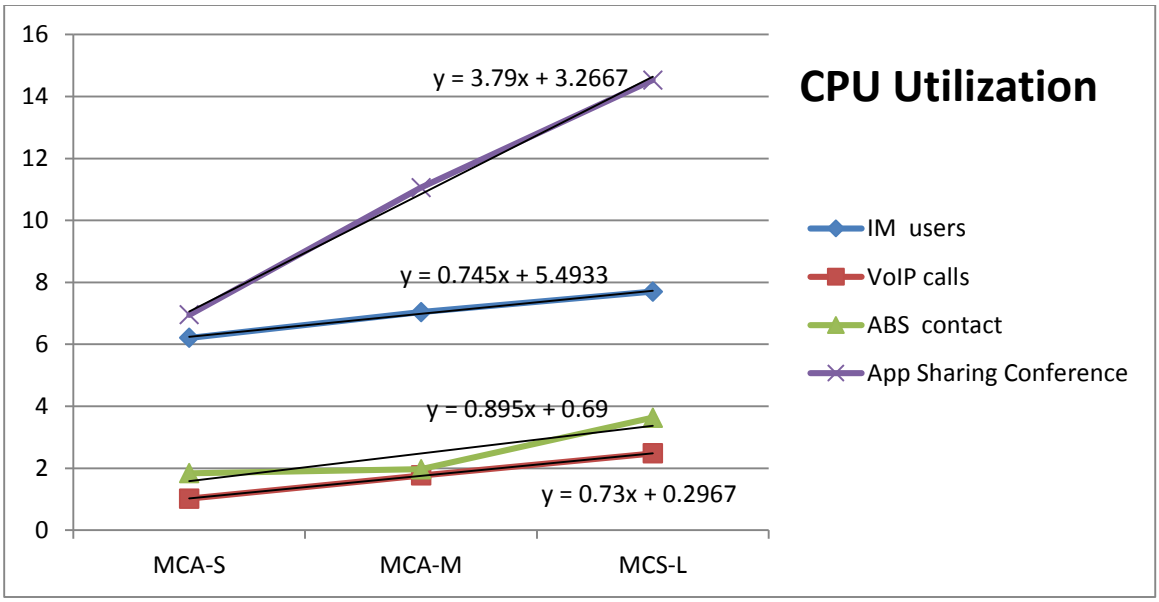


Figure 4: CPU utilization

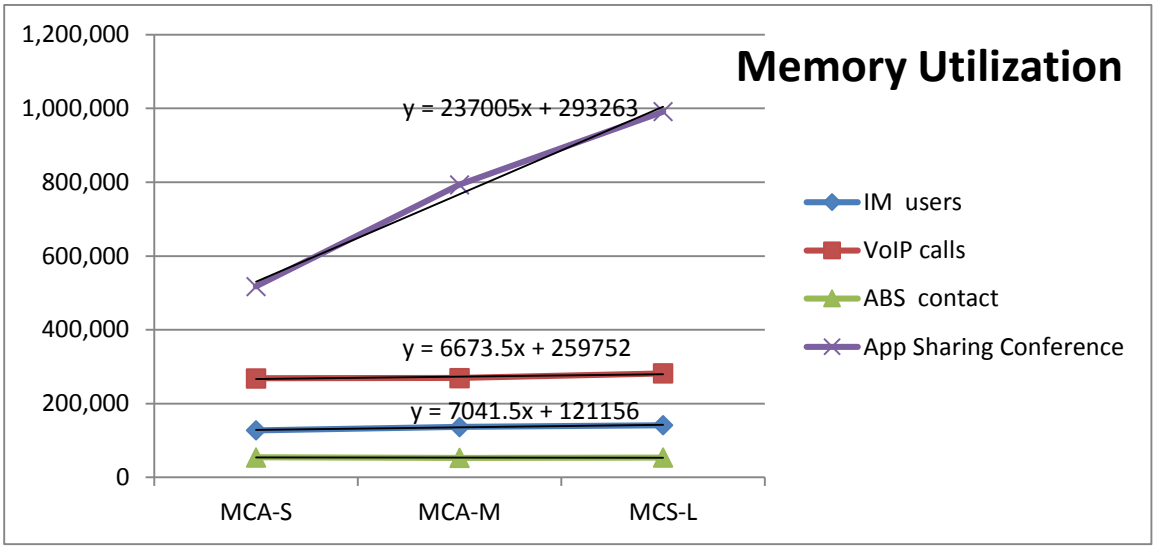


Figure 5: Memory utilization

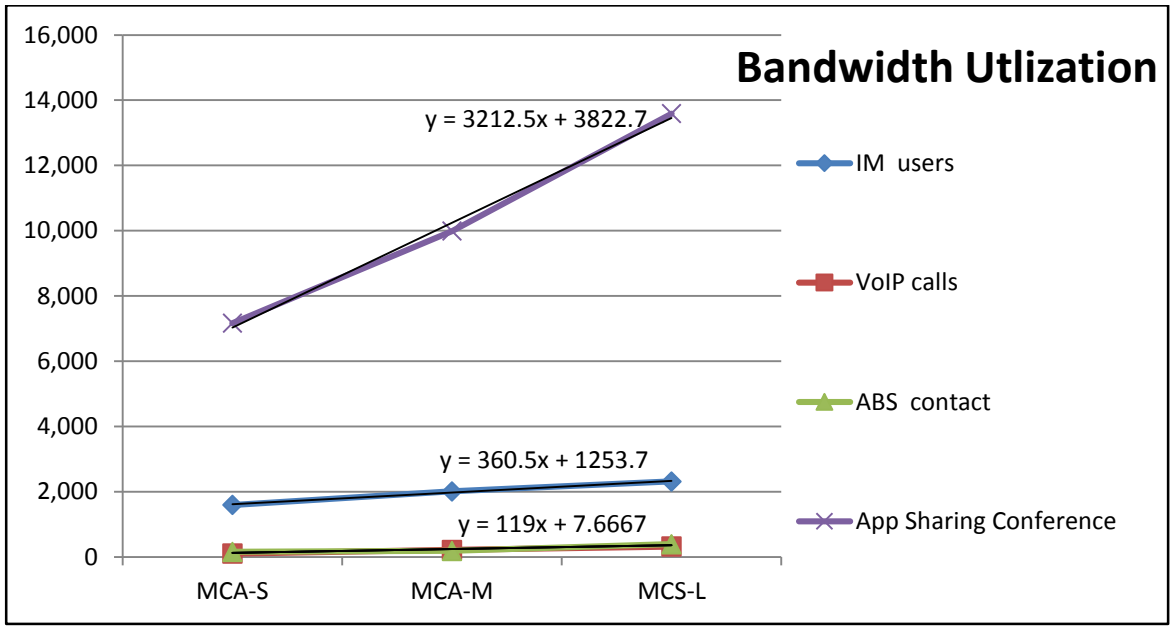


Figure 6: Bandwidth utilization

The figures, Figure 4: CPU utilization, Figure 5: Memory utilization, and Figure 6: Bandwidth utilization, summarize the results of the three experiments. By adding trend lines to the chart lines, it can be seen that the modalities grow linearly. The next section discusses the details of obtaining the results.

In order to simplify the methodology, consider *T* tenants (customers) with their distribution denoted by T1, T2. … Tn. Consider *m* modalities, and *r* resources. The provider can calculate the resources needed using the following equation:

$$\text{Cost of resource } r \text{ of tenant } T = \sum_{i=0}^{m} N \qquad (1)$$

Where *N* is the modality cost of resource *r*.

In the experiments in this research, the CPU cost for instant messaging modality using Office Lync Server 2010 (Microsoft, 2012) was found to be:

$$0.745x + 5.4933 \qquad (2)$$

Where *x* is the number of concurrent users being provisioned.

The CPU cost for application sharing was found to be:

$$3.79x + 3.2667 \qquad (3)$$

Where *x* is the number of concurrent or active users.

These equations were deduced by capturing CPU utilization while varying number of users (see Figure 4: CPU utilization below for CPU trend lines and subsequent sections for further information). Therefore, a provider wanting to calculate the CPU cost with these two modalities can obtain it simply by summing the resource cost of each modality being provisioned, that is by simply adding Eq. (2) and Eq. (3) above:

$$(5.4933 + 0.74x) + (3.2667 + 3.79x) \qquad (4)$$

3.4 Experiments and Results

In this section, early experiments with modality cost analysis are presented. Two sets of experiments are performed. In the first set of experiments, the resource overhead for four modalities are measured in isolation, namely instant messaging, Voice over IP (VoIP), application sharing conference, and address book download. In the second set, the resource overhead for three scenarios that combine all of the four modalities together are measured simulating a real end user experiment. The first scenario is named MCA-S for small load, the second one is named MCA-M for medium load, and the third one named MCA-L for large load, representing a small, medium, and large customers.

The performance estimation is based on the following hardware: A server with dual processors quad-core 2.0 GHz (2,000 megacycles per second), 16 gigabytes of memory,

30GB disk space, and 2-port 1 gigabit per second network adapter. The hardware topology remains fixed during the experimentation.

Office Lync Server 2010 (OLS) (Microsoft, 2012) which is an enterprise real-time communications server software, providing the infrastructure for enterprise instant messaging, data collaboration conferencing and multiparty Voice and Video calling. These features are enabled within an organization, between organizations, and with external users on the public internet. This product is also provided as a Cloud offering as part of Office 365.

Office Lync Server was deployed on the above described hardware server, and a simulation tool representing users accessing the server is developed for these experiments. In the first experiment, users are simulated using instant messaging modality only (in isolation where no other modality is running). In experiment 2, users are simulated making VoIP calls with no other modality running. In experiment 3, users are simulated joining a conference call and sharing a power point presentation. In experiment 4, users are simulated downloading an address book.

For the first experiment (Table 3: Experiment I Results), 5000 users were simulated sending IM messages to each other at the same time. The CPU utilization was measured over a period of 4 hours, and the averages were obtained for the CPU utilization of the server. In addition, the CPU utilization was measured using megacycles. The megacycles are obtained by multiplying the experiment server megacycles (2,000) by the number of cores (8) or a total of 16,000 megacycles per server. For example, if a modality is utilizing 10% of server

processors resources, it is calculated that it is consuming 1,600 megacycles. In addition, network bandwidth and memory utilization are measured. Then, the load is increased, and a medium size of 10,000 simultaneous users are run next. Finally, a large size experiment of 15,000 simultaneous IM users are run at last. The tables below show the result of the data collection.

Table 3: Experiment I Results

Instant Messaging Users	CPU %/Server	CPU Megacycles	Network/Bytes	Memory/Bytes
5000	6.21	998	1,596,403	117,435,418
10000	7.04	1,126	2,011,843	136,765,376
15000	7.70	1,232	2,317,056.51	141,518,365

Table 4: Experiment II Results

VoIP Users	CPU %/Server	CPU Megacycles	Network/Bytes	Memory/Bytes
200	1.02	163	104,508	268,334,836
400	1.7	272	216,545	269,283,186
600	2.48	396	320,444.62	281,681,544

Table 5: Experiment III Results

Application Sharing Conference	CPU %/Server	CPU Megacycles	Network/Bytes	Memory/Bytes
100 users	6.95	1,112	7,164,641	517,244,781
200 users	11.06	1,769	9,990,548.47	793,322,894
250 users	14.53	2,324	13,589,203.86	991,254,808.25

Table 6: Experiment IV Results

Address Book Download	CPU %/Server	CPU Megacycles	Network/Bytes	Memory/Bytes
5000 entries	1.84	294	157,286	53,965,229
10000 entries	1.97	315	185,179.73	52,671,103
15000 entries	3.63	580	395,116.23	53,686,217

In the second experiment, three research are simulated, namely, 200, 400, and 600 users making VoIP calls simultaneously. Table 4: Experiment II Results above shows the resource cost for each run.

In the third experiment, the research simulated a conference call with application sharing and 100, 200, and 250 users connecting simultaneously. Table 5: Experiment III Results above shows the resource cost for each run.

In the fourth experiment, 1000 simultaneous users downloading an address book with 5000, 10000, and 15000 contacts, respectively were simulated. Table 6: Experiment IV Results above shows the resource cost for each run.

Using the above results, for example, a provider that wants to provision 10,000 IM users, 6,000 VoIP users, and 250 application sharing conference, will need: 7.04 + 2.48 + 14.53 = ~24% of the CPU resource of one server (with 2.0 GHz and 8 cores or a total of 3,840 megacycles), and 136,765,376 + 281,681,544 + 991,254,808.25 = ~ 1.4GB of memory. Using such methodology, providers can plan their capacity to the exact needs without having to over-provision. However, it is well expected that Cloud providers do not necessarily provision the same hardware profile, it is also expected that hardware upgrades force a change to the provisioned resources, for example, upgrading memory or CPU. Providers that want to utilize this methodology and apply it to a different hardware profile can benchmark the processor used in this experiment against existing or planned hardware. Section 3.6 below discusses this method in more details to benchmark hardware variations using an industry accepted standard.

3.5 Validation Methodology

In the second set of experiments, the four modalities are mixed together to validate that measuring resources in isolation is an acceptable methodology for datacentre provisioning. In order to prove this hypothesis, the three experiments are run mixing IM, VoIP, Address Book download, and Application Sharing conference, using a tool called Office Lync Server Stress Tool (Microsoft, 2012) (LSS, Figure 4: Lync Server Stress Tool GUI). The stress tool generates a simulated load on Office Lync Server based on each experiment's load. For

example, when it is time to set up IM users, the tool will send instant messages between differently simulated users based on the load that is specified (in this case, 5000 users sending instant messages (at a rate of 4 instant messages per user per hour). This user profile remains constant across all the experiments.

Figure 4: Lync Server Stress Tool GUI shows a snapshot of the Lync Server Stress tool.

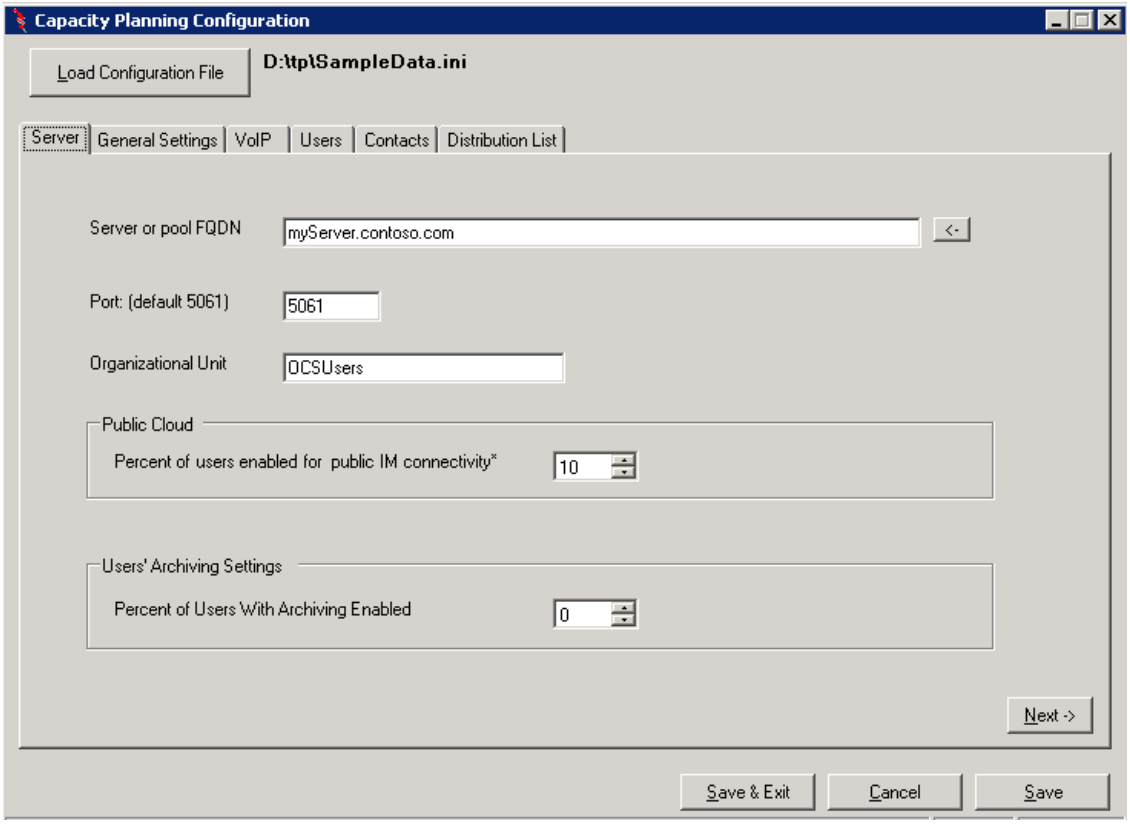


Figure 7: Lync Server Stress Tool GUI

The first experiment in the second set, named MCA-S, simulates users using all the four modalities in smaller quantities over time and putting the load against the Office Lync Server. To do this, the same hardware are set up to run the modalities in isolation and then using Lync Server Stress tool for simulating and experimenting the server with 5000 Instant Messaging users sending messages to each other where each user is sending 4 IMs/hour (the same load as when the modality in isolation was run). Then 200 VoIP calls were loaded, in parallel, 1000 users downloading 5000 contacts simultaneously, and 100 users sharing a power point presentation (size of 5 MB) at the same time.

Table 7: MCA-S Experiment Results below summarizes the findings for the first experiment in the second set of experiments.

In order to calculate the average, the experiment is run on 4 servers independently. The chart presented in Figure 5: CPU average for each server below shows how the CPU averages for each of the servers are measured and collected.

As shown in Figure 5: CPU average for each server, the averages for the four CPUs are 19.13%, 13.81%, 9.16%, and 15.82% or an aggregate average of 14.48%. Also it is important to note that the experiment is run for 2 hours and collected the data every ten minutes as shown in Figure 5: CPU average for each server.

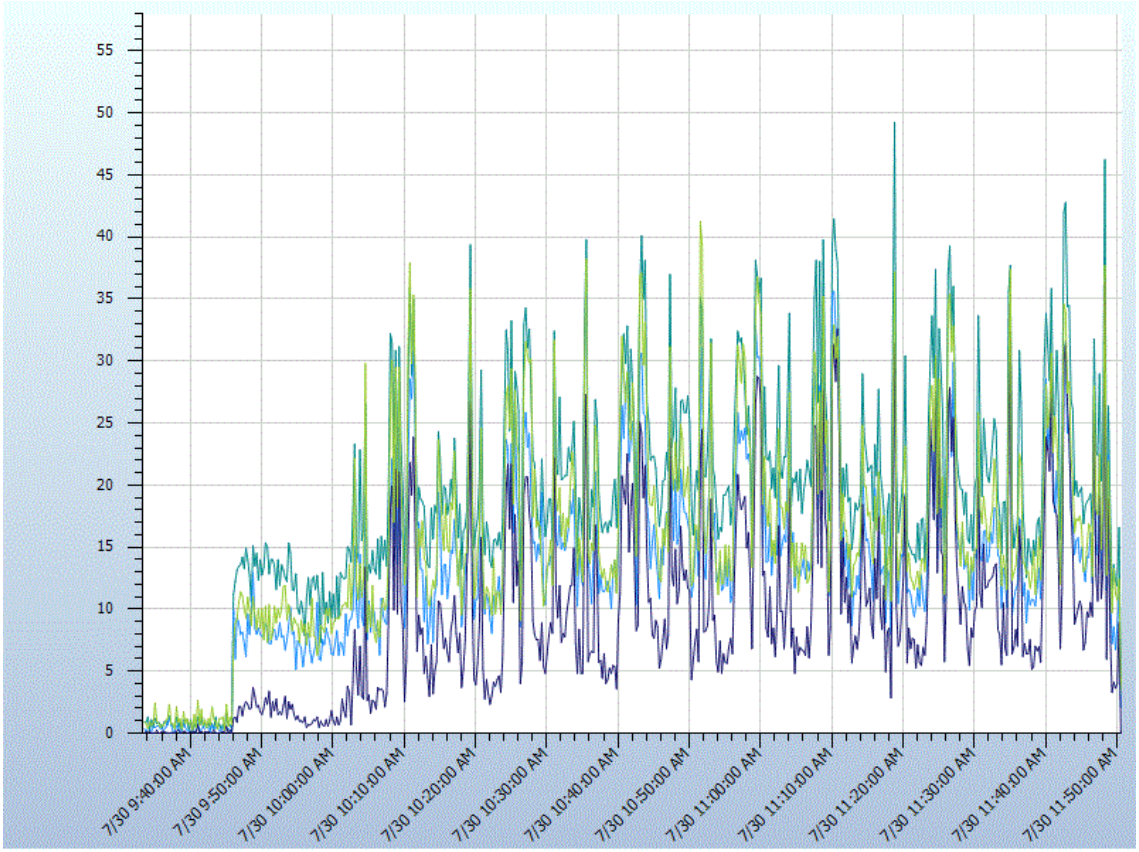


Figure 8: CPU average for each server

Table 7: MCA-S Experiment Results

		CPU % /Server	CPU Megacycles	Network /Kbytes	Memory /Kbytes
MCA-S	IM 5000 users	6.21	994	1,596	127,435
	VoIP 200 calls	1.02	163	104	268,334
	ABS 5000 contact	1.84	294	157	53,965
	App Sharing Conference 100 users	6.95	1,112	7,164	517,244
	Total of Isolated Measurements	16.02	2,563	9,022	956,980
	Measured Resource Cost	14.48	2,316	8,382	1,086,426
	Diff	**-10%**	**-10%**	**-7.10%**	**11.90 %**

Table 8: MCA-M Experiment Results

		CPU % /Server	CPU Megacycles	Network /Kbytes	Memory /Kbytes
MCA-M	IM 10000 users	7.04	1,126	2,011	136,765
	VoIP 400 calls	1.77	283	216	269,283
	ABS 10000 contact	1.97	315	185	52,671
	App Sharing Conference 200 users	11.06	1,770	9,990	793,322
	Total of Isolated Measurements	21.84	3,494	12404	1,252,042
	Measured Resource Cost	19.89	3,182	11,676	1,304,269
	Diff	-10%%	-10%	-6%	4%

Table 9: MCA-L Experiment Results

		CPU % /Server	CPU Megacycles	Network /Kbytes	Memory /Kbytes
MCA-L	IM 15000 users	7.70	1,232	2,317	141,518
	VoIP 600 calls	2.48	454	320	281,681
	ABS 15000 contact	3.63	580	395	53,686
	App Sharing Conference 250 users	14.53	2,324	13,589	991,254
	Total of Isolated Measurements	28.34	4,534	16,621	1,468,140
	Measured Resource Cost	33.98	5,436	16,953	1,492,496
	Diff	**16%**	**16%**	**2%**	**1.3%**

The second experiment, named MCA-M, simulates a user using all the four modalities in medium quantities. Table 8: MCA-M Experiment Results above summarizes the findings for the second experiment.

The third experiment, named MCA-L, simulates users using all the four modalities in large quantities. The outcome of that is summarized in Table 9: MCA-L Experiment Results above.

The results show that measuring modalities in isolation and using the results to provision datacentre is an effective capacity planning methodology. The variance between measuring in isolation and measuring the modalities running side by side is within ±16%. In order to better plan for such variance, it is recommended to add an adequate buffer for covering variation in side-by-side versus aggregated execution. 10% to 30% buffer is considered a minor buffer compared to current hardware over-provisioning estimates of 200-300% in best cases, and 5% to 10% of server resource utilizations in some of the worse cases (CA, 2010). Process isolation is typically known to affect how resources are utilized, however; it is not deterministically assessed (Fedorova, et al., 2008).

Using the equations discussed in the sections above, Cloud providers can predict the utilization at any point. It is expected that each modality will hit a ceiling level which is not captured in these experiments. Such work is needed to figure out the limits of where the system starts reaching a point of non- linear growth. It is important to note that this methodology is applicable for static allocation of resources, in other words, in cases where the Cloud provider understands the expectations and what the user model is. Example of such cases are companies with finite resources and known set of employees that access the system at a typical point in the day. To illustrate, consider the case of company contoso.com with 10,000 employees. All the employees will sign in between 9 AM and 10 AM and will send Instant Messages to each other at an average of 4 IMs per user per hour. Such company can easily predict the load on the system and provision the Cloud or data centre

to fit this need. This approach will not work for dynamic allocation of resources, in other words, if the company has no knowledge of how many users sign in per day, and what their user model is like. Such organizations need to apply a dynamic capacity model approach. An example is an organization that provides access to public users (for example "What's App"), where the usage is determined by events around the world (for example, if there is an earthquake or natural disaster in some country, the load will spike unexpectedly). Such model is best represented with dynamic capacity allocation. The model in this research is most applicable to enterprises model where the set of users are somewhat static and the user model is well understood (i.e. the time of the day employees are most active and the average usage per employee).

3.6 Hardware Benchmarks

Rapid change in hardware and the multitude of different hardware configurations available nowadays make it difficult for any provider wanting to adopt performance optimization or capacity planning methodologies. For example, a provider validating against existing hardware may find that the hardware is not available during procurement time. In order to ensure that this methodology is not hardware specific, benchmarking techniques can be used to adapt the methodology and equations identified in this work to different hardware settings. For example, processor benchmarking tools such as SPECint (SPECint Processor Benchmarking, 2012) can be used. The SPECint processor benchmark for the hardware used

in this research methodology is 186 for eight cores or 23.25 per core. So, providers interested in using this performance validation methodology against a different hardware can use the following steps:

- Visit the SPECint website (SPECint Processor Benchmarking, 2012)
- Select SPECint2006 Rates
- Find the server and processor they have deployed or intend to deploy, and look at the number in the Result column.
- Dividing this value by the number of cores in the server returns the per-core value. For example, if the Result number is 240 on an eight-core server, the per-core value is 30.
- The following equation can then be used to determine the per-core megacycles for the server: *(Per-core value) x 2,000/ 23.25*
- Finally, by multiplying the result above by the number of cores in the server, the total number of megacycles per server is obtained. This is then compared to the 16,000 megacycles for the baseline server used to produce the numbers in these experiments.

In order to clarify this further, consider the following example. Assume a provider to provision the following modalities as summarized in Table 10: Benchmarking example modalities below.

Table 10: Benchmarking example modalities

Modality	Test server CPU% cost	Megacycles needed
IM 15000 users	7.70	(7.7/100) * 2,000 * 8 = 1,232
VoIP 600 calls	2.48	(2.48/100) * 2,000 * 8 = 396
ABS 15000 contact	3.63	(3.63/100) * 2,000 * 8 = 580
App Sharing Conference 250 users	14.53	(14.53/100) * 2,000 * 8 = 2325
Total	~28% of total server CPUs	4,533 total megacycles needed

For this example, suppose the Cloud provider is deploying servers with a SPECInt result of 186 for 8 cores, which averages out to 23.25 per core. Using the calculations explained in the previous sections, once can compute the megacycles of the servers, which would be 16,000 megacycles each in this case.

To determine the number of such servers required to provision the above modalities, the number of needed megacycles (4,533) can be divided by the number of megacycles per

server (16,000 in this example). This can easily be replaced by the number of megacycles represented by the hardware being utilized.

Thus, in this example, it is needed a circa of 28% of total server CPU resources to run the modalities in the table above.

3.7 Conclusion

In this chapter, a quantitative methodology for capacity planning in Cloud datacentres was presented. The results are used to guide providers into provisioning datacentres for optimizing performance and cost. By profiling an application into a set of modalities and measuring hardware resources cost in isolation, Cloud providers should be able to pinpoint their capacity to exact needs without wasting expensive resources. In addition, a process for applying hardware benchmarks for scenarios where experimental hardware servers differ from deployment hardware or for upgrading hardware servers without invalidating experimental results was analysed. In addition, a discussion on how to validate the results by running three sets of experiments, MCA-S, MCA-M, and MCA-L that represent small, medium, and large user profiles was presented. The results showed that measuring modalities in isolation and using the results to provision datacentre is an effective methodology. As one of the future research directions, it is important to address virtualization using modality cost analysis methodology and address any effects or

limitations. This work can extensively benefit from virtualization to dynamically allocate resources based on usage profile. In order to achieve this, one can plan to look at Windows Azure as a virtualization platform where one can deploy MCA and provision dynamically in order to reduce the total cost of ownership while maintaining SLAs.

Chapter 4: Lync Server Capacity Planning Calculator

4.1 Overview

Microsoft Lync Server is an enterprise real-time communications server, providing the infrastructure for enterprise instant messaging, presence, file transfer, one or multiparty voice and video calling, conferencing (audio, video and web) among other features (Microsoft, 2012). These synchronous and real-time communication modalities are dependent on the availability of hardware resources. It is different from asynchronous communication modalities such as email where a user would not perceive an outage if the resources were not readily available. For example, sending an email that does not make it until few minutes later does not warrant an outage, while a voice or video call with few seconds of delay becomes useless. Software with such dependencies do require adequate capacity planning. Unfortunately, most cloud providers deploy excessive hardware resources to host Lync server at the cost of ensuring quality and meeting SLAs. This work is targeted to guide such cloud providers to better assess capacity needs in a methodological approach.

This chapter provides guidance for hardware allocation based on the analysis of organization needs. The organization would use their organization's number of users, user profiles, and workloads deployed to determine the necessary CPU clock speed, server memory

requirements, and network bandwidth required for the server. The results are applicable to both physical and virtual topologies.

The information in this chapter will be especially helpful if the user model or server hardware do not differ from what is described in Lync Server 2010 User Models (Microsoft, 2012). If the user model differs, then additional work is needed to benchmark the differences. For the sake of this these experiments, we expect a user model as defined in (Microsoft, 2012).

All the performance analysis provided assume a baseline that each server has dual quad-core processors with a clock speed per core of 2.33 GHz. This yields 2,333 megacycles per processor core, or 18,664 megacycles per server.

If the organization's servers have different processors, they can adjust the figures accordingly. For details, see "Adjusting for Processors' Variations" later in this chapter.

The Microsoft Lync Server 2010 capacity planning calculator is designed to assist the organization in determining server requirements based on numbers of users and communication modalities that are enabled at their organization. The organization administrators enters their organization's profile, and the calculator provides recommendations that help them plan their topology. The calculator is an Excel Tool that is based on the Modality Cost Analysis research.

The recommendations created by the calculator are for planning purposes only. Actual load simulation is required to ensure that Lync Server 2010 is adequately provisioned. To perform

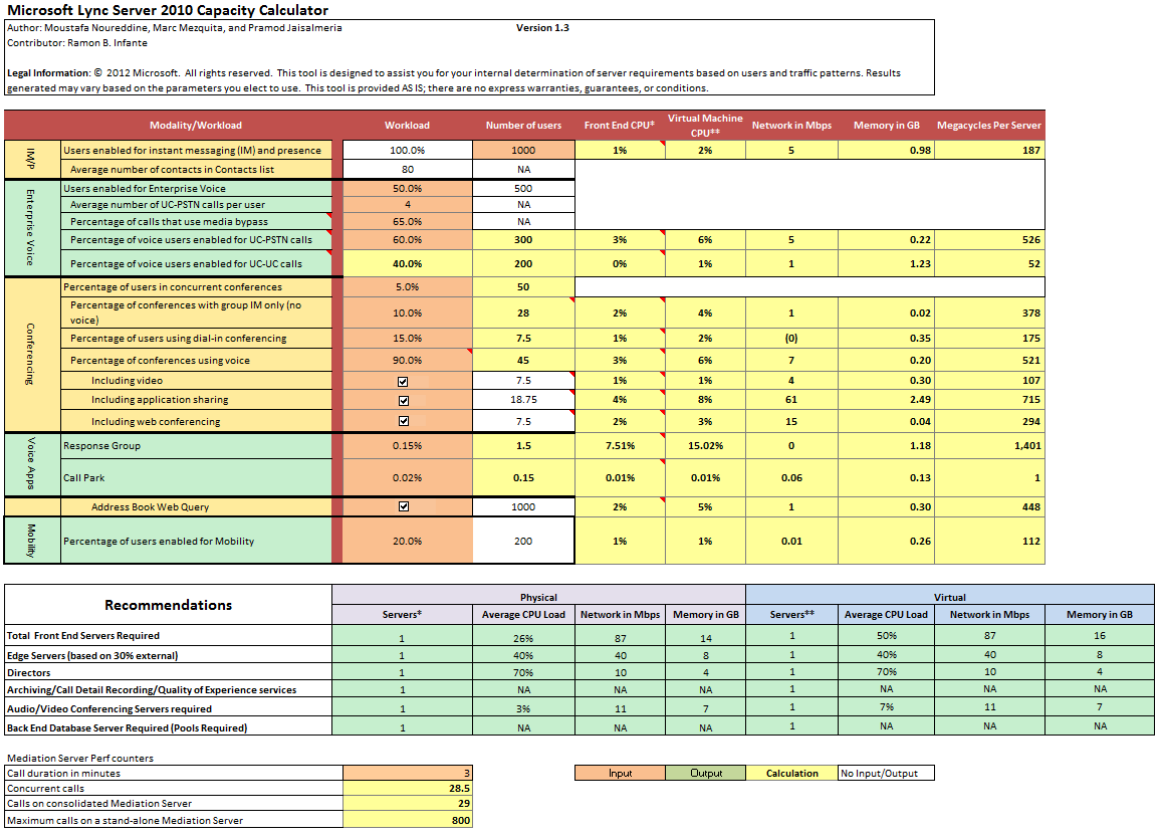
stress testing under a simulated load, the Lync Server 2010 Stress and Performance Tool can be used (Microsoft, 2012).

After the organization administrators have determined their user profile and the modalities that they want to enable for their users, it is time to use the calculator to plan the number of servers, memory, and bandwidth that are needed. This version of the calculator does not provide guidance for disk I/O requirements. For disk I/O requirements, refer to the Capacity Planning section of the Lync Server 2010 planning documentation (Microsoft, 2012). This calculator complements the Microsoft Lync Server 2010 Planning Tool and Microsoft Lync Server 2010 Planning Guide (Microsoft, 2012). The organization can benefit most from the calculator if they have accurate, detailed information about their specific user profile. For example, the percentage of voice-enabled users, average calls per user per hour, call duration, and the percentage of concurrent users in conferences can make a huge difference in server requirements. The accuracy of the recommendations created by the calculator depends on the accuracy of the information that the customer provides into the calculator.

4.2 Using the Capacity Calculator

The calculator is a Microsoft Excel spreadsheet and can be downloaded from http://www.microsoft.com/en-us/download/details.aspx?id=12295. Yellow-colour cells are for input from the customer. Default values are entered (80,000 users in one organization with eight Front End Servers), but the customer can change these values according to their organization's needs.

The following sections below explain what the organizations need to provide to calculate the capacity required. The following image Figure 9: Lync Server Capacity Planning Calculator shows a snap of the Excel calculator that is used in this guide. As mentioned above, the calculator is available for download with instruction manual on the Microsoft website. (Microsoft, 2011)



Microsoft Lync Server 2010 Capacity Calculator

Author: Moustafa Noureddine, Marc Mezquita, and Pramod Jaisalmeria — **Version 1.3**
Contributor: Ramon B. Infante

Legal Information: © 2012 Microsoft. All rights reserved. This tool is designed to assist you for your internal determination of server requirements based on users and traffic patterns. Results generated may vary based on the parameters you elect to use. This tool is provided AS IS; there are no express warranties, guarantees, or conditions.

	Modality/Workload	Workload	Number of users	Front End CPU*	Virtual Machine CPU**	Network in Mbps	Memory in GB	Megacycles Per Server
IM/P	Users enabled for instant messaging (IM) and presence	100.0%	1000	1%	2%	5	0.98	187
	Average number of contacts in Contacts list	80	NA					
Enterprise Voice	Users enabled for Enterprise Voice	50.0%	500					
	Average number of UC-PSTN calls per user	4	NA					
	Percentage of calls that use media bypass	65.0%	NA					
	Percentage of voice users enabled for UC-PSTN calls	60.0%	300	3%	6%	5	0.22	526
	Percentage of voice users enabled for UC-UC calls	40.0%	200	0%	1%	1	1.23	52
Conferencing	Percentage of users in concurrent conferences	5.0%	50					
	Percentage of conferences with group IM only (no voice)	10.0%	28	2%	4%	1	0.02	378
	Percentage of users using dial-in conferencing	15.0%	7.5	1%	2%	(0)	0.35	175
	Percentage of conferences using voice	90.0%	45	3%	6%	7	0.20	521
	Including video	☑	7.5	1%	1%	4	0.30	107
	Including application sharing	☑	18.75	4%	8%	61	2.49	715
	Including web conferencing	☑	7.5	2%	3%	15	0.04	294
Voice Apps	Response Group	0.15%	1.5	7.51%	15.02%	0	1.18	1,401
	Call Park	0.02%	0.15	0.01%	0.01%	0.06	0.13	1
	Address Book Web Query	☑	1000	2%	5%	1	0.30	448
Mobility	Percentage of users enabled for Mobility	20.0%	200	1%	1%	0.01	0.26	112

Recommendations	Physical				Virtual			
	Servers*	Average CPU Load	Network in Mbps	Memory in GB	Servers**	Average CPU Load	Network in Mbps	Memory in GB
Total Front End Servers Required	1	26%	87	14	1	50%	87	16
Edge Servers (based on 30% external)	1	40%	40	8	1	40%	40	8
Directors	1	70%	10	4	1	70%	10	4
Archiving/Call Detail Recording/Quality of Experience services	1	NA	NA	NA	1	NA	NA	NA
Audio/Video Conferencing Servers required	1	3%	11	7	1	7%	11	7
Back End Database Server Required (Pools Required)	1	NA	NA	NA	1	NA	NA	NA

Mediation Server Perf counters

Call duration in minutes	3
Concurrent calls	28.5
Calls on consolidated Mediation Server	29
Maximum calls on a stand-alone Mediation Server	800

Input | Output | Calculation | No Input/Output

Figure 9: Lync Server Capacity Planning Calculator

4.2.1 Instant Messaging and Presence

- Under Number of Users, type the number of users who will be concurrently signed in. This number is typically 80% of the total number of provisioned users. In some situations, 100% of the customer concurrent users will be enabled for IM and Presence. The default number of total users is 80,000. This number is based on the typical profile of a large enterprise hosting Lync Server in one Cloud datacentre.
- Average number of contacts in Contact list indicates the number of contacts that are being used to validate the system requirements. This number is not changeable. This number is not changeable because it severely affects the calculation in this research. Organizations that have significantly different number of contacts per user should customize the calculator to their needs using the Modality Cost Analysis methodology.

4.2.2 Enterprise Voice

- In Users enabled for Enterprise Voice, type the percentage of concurrent users who are enabled for Enterprise Voice. The default is 50%.
- In Average number of UC-PSTN calls per user, type the number of calls per hour that the customer expects the average user to participate in during times of peak load. The default is 4.
- In Percentage of calls that use media bypass, type the percentage of users who are enabled for Enterprise Voice who will place UC-PSTN phone calls that will bypass the Mediation Server.

- In Percentage of voice users enabled for UC-PSTN calls, type the percentage of users who are enabled for Enterprise Voice who will concurrently participate in UC-PSTN phone calls.
- In Percentage of voice users enabled for UC-UC calls, type the percentage of users who are enabled for Enterprise Voice who will be concurrently participating in UC-UC calls.

4.2.3 Conferencing

- In Percentage of users in concurrent conferences, type the percentage of concurrent users who will be concurrently participating in conferences. The default is 5%.
- In Percentage of conferences with group IM only (no voice), type the percentage of conferences whose conferences will involve instant messaging only, that is, that do not include an audio component.
- In Percentage of users using dial-in conferencing, type the percentage of concurrent participants in conferences who will be using dial-in conferencing.
- In Percentage of conferences using voice (web conferences), type the percentage of conferences that will include an audio component. If 20% of the voice conferences will also include video, select the 'Including video' check box. If 50% of your voice conferences will also include application sharing, select the 'Including application sharing' check box. If 20% of your voice conferences include data uploads, such as Microsoft PowerPoint® presentations, select the 'Including web conferencing' check box.

4.2.4 Voice Applications

- In Response Group Service, type the percentage of concurrent users who will use the Response Group service.
- In Call Park, type the percentage of concurrent users who will use the Call Park service.

If the customer will enable Address Book Web Query, select the corresponding check box.

When the customers have entered all the necessary information, the capacity calculator estimates the requirements. The pink cells show calculated values for CPU, memory, and bandwidth requirements based on tests performed in Lync Server 2010 performance labs. The numbers are provided as a guideline; not every single variation is tested and validated. The following values are calculated:

- Front End CPU: Percentage of CPU usage if the entire load was being handled by one Front End Server of the same specifications as the server that was used in Microsoft testing.
- Virtual Machine CPU: Percentage of CPU usage if the entire load was being handled by one virtual machine of the same specifications as those used in Microsoft testing.
- Memory Requirements: Memory required in gigabytes (GB) for the corresponding workload.
- Bandwidth Requirements: Bandwidth requirements in megabits per second (Mbps) for the corresponding workload.

The green cells show recommendations for the usage model that you entered.

Total Front End Servers calculation identifies the number of physical servers required. These calculations are based on dedicated servers running Lync Server 2010 with dual processor, quad-core, with 2,333 megacycles.

- Total Virtual Servers: The number of virtual servers required is based on dedicated servers running Lync Server 2010 with four cores, each with 2,333 megacycles.
- Edge Servers: The number of Edge Servers required, based on 30% of all concurrent users communicating through the Edge Servers. This percentage cannot be changed in the calculator.
- Directors: Number of Directors needed. Each Director is assumed to support 15,000 users.
- Audio/Video Conferencing Servers: Number of dedicated A/V Conferencing Servers needed to support the selected conferencing workload based on Lync Server 2010 with eight cores for physical servers or four cores with virtual servers. The usage model does not support collocating the A/V Conferencing service on Front End Servers. Hyper-threading is disabled on these servers. Note that enabling hyper-threading is recommended and has been proven to improve performance for servers that support audio/video.
- Back End Database Servers: The number of back-end database servers required to support the selected workload.

- Average CPU Load: The average CPU usage per server.
- Network in Mbps: The required bandwidth allocation to support the usage model that you entered.
- Memory in GB: Memory, in gigabytes, required for each server.

4.3 Adjusting For Processors' Variations

All the CPU usage figures in the spreadsheet assume that each server has a dual processor, quad-core, with 2.33 GHz. This yields 2,333 megacycles per second per processor core, or 18,664 megacycles per second per server.

If the servers have different processors, the customer can adjust the figures to match their hardware.

The SPECint processor benchmark for the processors used in these tests is 186 total for the eight cores, or 23.25 per core. To calculate the equivalent processor cycles for your servers, do the following:

1. In a web browser, go to http://www.spec.org.
2. In the navigation bar of the website, point to Results, point to CPU2006, and then click Search CPU2006 Results.
3. In the Available Configurations box, click SPECint2006 Rates, and then click Go!
4. Under Simple Request, select search criteria that will help you find your processor, and then click Execute Simple Fetch.

5. Find the server and processor that you have deployed, and look at the number in the Result column.
6. To obtain the per-core value, divide the value in the Result column by the number of cores in the server. For example, if the Result number is 240 on an eight-core server, the per-core value is 30.
7. Use the following formula to determine the per-core megacycles for the server:

   (Your processor's per-core value) × 2,333 / 23.25
8. Multiply the result by the number of cores in the server, and you have the total number of megacycles per server. This compares to the 18,664 megacycles for the baseline server used to produce the numbers in the previous sections of this topic.

For examples and for details about adjusting for your processors, see next section for an example of calculating needed resources.

4.4 Example Calculating Needed Resources

The following example shows how the customer can calculate their resource needs if the organization's use of Lync Server 2010 differs from that in the Lync Server 2010 User Models. In this example, the organization's use is significantly higher than in the user model.

30,000 users, 100% use Enterprise Voice (instead of 50% of users being voice-enabled, as in the user model). Mediation Server is collocated with Front End Server. 75% of UC-PSTN calls use media bypass.

An average of 7.5% of users are concurrently in conferences (instead of the 5% in the user model), giving us 2250 concurrent users in conferences.

Other conferencing uses follow the Lync Server 2010 user model.

Enterprise Voice usage is heavier than in the user model, with a busy hour average of five calls per hour lasting an average of 3 minutes (the user model is four calls per hour at busy hour). Following the user model, three of those five calls will be UC-PSTN, and two will be UC-UC.

The CPU needs for the Front End Server are calculated in Table 11: Server CPU Capacity Calculations as follows:

Table 11: Server CPU Capacity Calculations

Front End Server workload	Test server CPU% cost	Megacycles needed
Base IM and Presence	30,000 users * 0.001 = 30	(30/100) * 2,333 * 8 = 5,599
Address Book Web Query	(30,000 users * 0.0004) + 2 = 14	(14/100) * 2,333 * 8 = 2,613
Group IM (50% of conferences use group IM)	(1125 users * 0.001) + 2 = 3.125	(3.125/100) * 2,333 * 8 = 583

Web conferencing (75% of all conferences include web conferencing, and 20% of those conferences include group IM)	(337 users * 0.01) + 1.5 = 4.87	(4.87/100) * 2,333 * 8 = 909
PSTN conferencing (15% of conference attendees dial in from PSTN phones)	(338 users * 0.033) + (338 users * 0.0918) = 42.18	(42.18/100) * 2,333 * 8 = 7,872
Application sharing (75% of all conferences include web conferencing, and 50% of those conferences use application sharing)	(843 users * 0.071) + 2.5 = 62.353	(62.353/100) * 2,333 * 8 = 11,638
Enterprise Voice, UC-UC calls	30,000 users * 2 calls * 3 minutes / 60 = 5000 concurrent calls 5000 calls * .007 = 35	(35/100) * 2,333 * 8 = 6,532
Enterprise Voice, UC-PSTN calls	30,000 users * 3 calls * 3 minutes / 60 = 4500 concurrent calls	(114.12/100) * 2,333 * 8 = 21,299

	(4500 calls * .007) + (4500 calls * 0.0918 * (1 - .8)) = 114.12	
		57,045 total megacycles needed on Front End Servers.

On the Front End Servers, the total CPU requirement for the heavily-used deployment is 57,045 megacycles. For this example, suppose a customer is deploying servers with a SPECInt result of 258 for 8 cores, which averages out to 32.25 per core. Using the calculations in the previous section, we see that these servers have 25,888 megacycles each.

To determine the number of these servers that are needed, divide the number of needed megacycles (57,045) by the number of megacycles per server (25,888 in this example). Then divide this result by .7, to ensure that each server runs at no more than 70% of CPU capacity. Take this final result and round it up to the next whole number. In this example,

(57,045/25,888)/0.7) = 3.15.

The customer will need four of these servers (round up of 3.15). The four servers provide the customer with a total of 103,552 megacycles, and 57,045 is about 55% of that, so our four servers should be running at 55% of CPU capacity at peak times.

The following Table 12: Audio/Video CPU Capacity Calculations determine the A/V Conferencing Server needs in the example scenario.

Table 12: Audio/Video CPU Capacity Calculations

A/V Conferencing Server workload	CPU cost	Megacycles needed
Audio conferencing (75% of conferences include Enterprise Voice)	1688 users * 0.062 = 104.625	(104.625/100) * 2,333 * 8 = 19,527
Video Conferencing (75% of all conferences include web conferencing, and 20% of these conferences include video)	338 users * 0.07625 = 25.77	(25.77/100) * 2,333 * 8 = 4,810
		24,337 total megacycles needed on A/V Conferencing Servers.

The customer can deploy two of these servers, at 25,888 cycles each, and run A/V Conferencing Server at about 47% CPU on each.

One can perform similar calculations for the memory and network bandwidth needed for the projected workload, as well. For workloads or scenarios in which the customers think their organization has typical usage patterns, refer to Lync Server 2010 User Models (Microsoft, 2012) to see the user models tested by Microsoft

Chapter 5: Performance of Cloud Authentication

5.1 Introduction to Cloud Authentication

A major hurdle of formal adoption of OAuth protocol for enterprise applications is performance. Enterprise applications (e.g. SAP, SharePoint, Exchange Server, etc.) require a mechanism to predict and manage performance expectations. As these applications become more and more ubiquitous in the Cloud, the scale and performance expectations become an important factor impacting architectural decisions for security protocol adoption. This chapter proposes an optimization to OAuth 2.0 for enterprise adoption. This optimization is achieved by introducing provisioning steps to pre-establish trust amongst enterprise applications' Resource Servers, its associated Authorisation Server and the clients interested in access to protected resources. In this model, trust is provisioned and synchronized as a pre-requisite step to authentication and authorisation amongst all communicating entities in OAuth protocol, namely, the client requesting a protected resource, the resource server, and the authorisation server. For a case study, SAP authenticating with SharePoint is simulated using the proposed optimization versus existing OAuth protocol. Such optimization will further facilitate the adoption of OAuth in the enterprise where scale and performance are critical factors. (Noureddine & Bashroush, 2011)

Chapter II above introduces OAuth and discusses in details the strengths and weaknesses and compares it against other protocols. To summarize, OAuth is a claim-based security

protocol that enables users to grant third-party access to their protected resources without sharing their passwords (Microsoft, 2012). OAuth implements this by using a data structure, called token, that decouples the access right from the client login credentials. Clients request tokens from an authorisation server and present the token to the service provider. OAuth 1.0 was published in December 2007 and quickly became the industry standard for web-based access delegation. However, OAuth 1.0 faced lots of challenges to make it into the enterprise domain mainly due to the lack of performance optimization capabilities currently on offer by the protocol. Microsoft, Google, and other large organizations proposed OAuth WRAP (Web Resource Authorisation Profiles) to solve the performance challenges and facilitate adoption by the enterprise. One of the main optimizations is the introduction of an independent Authorisation Server. OAuth adopted the WRAP recommendation into OAuth 2.0. However, adoption has not yet been proven in enterprise deployments (e.g. Microsoft Exchange Server, Lync Server, Oracle, SharePoint, SAP, etc.). In this work, an optimization is introduced to OAuth 2.0 where the Authorisation Server is provisioned with explicit authorisation table so that access grants are rejected at the Authorisation Server before getting to the protected resource. This reduces the amount of processing some popular protected resources would have to do and alleviates the risk of potential threats such as Denial-of-Service (DoS) attacks and Distributed DoS (DDoS) attacks. In addition, by extending the parameters of OAuth authorisation request, the calling client can reduce the number of calls it makes to the authorisation server. In the model developed in this research, a client makes a single trip to the authorisation server to serve all its users.

In the case study presented, it is shown how a SAP server would only need to make a single acquisition of a token to serve all it's logged in users with documents available in SharePoint.

In the next chapter, detailed discussion of the drivers behind the introduction of OAuth2.0 and its architecture are presented. The upcoming sections argue the modifications suggested to OAuth 2.0 in order to facilitate enterprise adoption of the protocol through a case study.

5.2 OAuth Optimization for Enterprise Adoption

It is often required for servers to integrate with each other and exchange protected data. An example of this is SharePoint Online integration with SAP. A third party may want to develop an application to login into SAP and save completed financial reports in SharePoint for sharing with colleagues or managers, for example. Since these financial reports are protected resource with high business impact, it may not want to hand its protected data to any application with a valid token. Also since the example protected resource servers SharePoint and SAP can host millions of users in the Cloud in a Shared Tenancy model, request for access with valid tokens can easily burden the server.

In this research proposal, shown in Figure 10: OAuth 2.0 Modified Architecture below, a new provisioning step is added in which a pre-established trust between the Client and the Resource Server is configured. This step can reduce many of the unwanted requests to the Resource Server. Also, during this step the client is given information on where to go to

acquire a token, in other words, the address of the Authorisation Server. Figure 9: OAuth 2.0 Protocol Flow and Figure 10: OAuth 2.0 Modified Flow show the before optimization step and the after optimization is applied call flow, respectively.

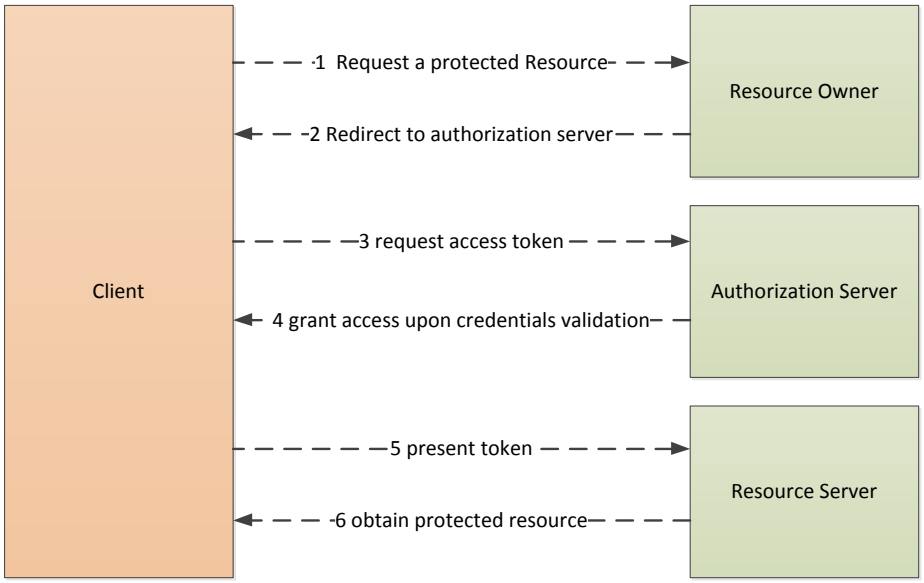


Figure 10: OAuth 2.0 Protocol Flow

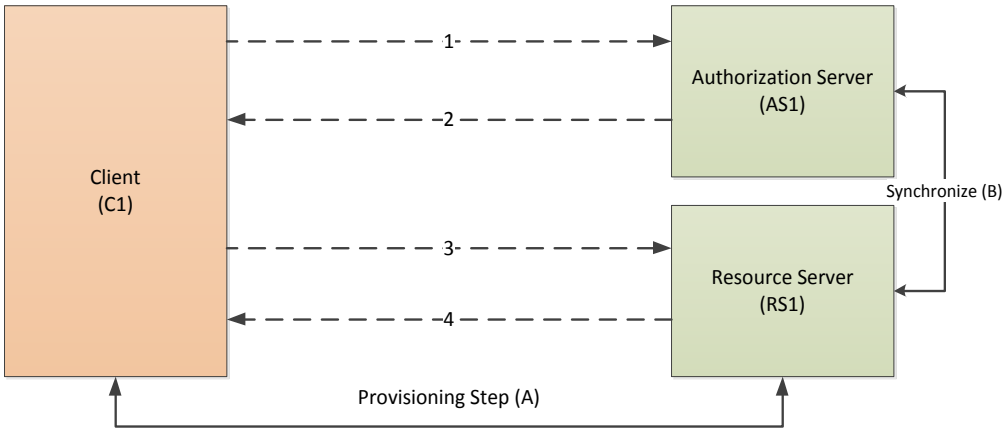


Figure 11: OAuth 2.0 Modified Flow

The abstract flow illustrated in Figure 10: OAuth 2.0 Modified Flow above describes the OAuth 2.0 modified interaction between the different roles and includes the following steps:

A. Provisioning step: clients interested in acquiring protected resources from resource server are provisioned (in a delegation table, for example). The address of the authorisation server is provided, for example: https://authserver.com
B. Resource Server synchronizes trusted client with Authorisation Server so that it only issues tokens to provisioned clients

When a client wants to access a resource from Resource Server:

A. The client requests authorisation grant from the authorisation server by presenting the client credentials

B. The authorisation server validates the credentials of the client and the authorisation grant. It also validates that the client is a trusted entity by Resource Server and issues an access token

C. The client requests the protected resource from the resource server and authenticates by presenting the access token

D. The resource server validates the access token, and if valid, serves the request.

It is important to note that, during provisioning step, the client may be given the rights to act on behalf of any user. In this case, the client makes a request to acquire a token (step 1 above) only once during the lifetime of the token. In this experiment, when a token is setup, the lifetime of the token is 24 hours, the client was making a round trip to the authorisation server once a day. This is a significant optimization over what the current OAuth 2.0 model offers.

5.3 Case Study

In this case study, the use of two enterprise applications are simulated, namely SharePoint Online and SAP that are interested in sharing protected resources on behalf of their users. For the purpose of this case study, SAP and SharePoint are also pre-provisioned to trust each other and can act on behalf of any user they trust. Therefore, SharePoint is provisioned to trust SAP and SAP is provisioned to trust SharePoint. The Authorisation Server is synchronized so it only issues tokens to trusted clients such as SAP and SharePoint. If any

client requests a token with no pre-established trust, the Authorisation server will deny access.

In order to simulate this, Resource Server (RS1) is built, and an Authorisation Table is setup as shown in below Table 13: Resource Server (SharePoint) Table. This table is synchronized on the Authorisation Server (Table 14: Authorisation Server (AS1) Table) as well as the client (Table 15: Client Table (SAP)). Resource Server can only issue tokens to SharePoint and for SAP. The Resource Server can only accept tokens issued by Authorisation Server (AS1) scoped to SharePoint.

Table 13: Resource Server (SharePoint) Table

Authorised Client	Credentials
SAP	SAP Credentials /public key
Client2	Client2 credentials/public key

Table 14: Authorisation Server (AS1) Table

Client	OAuth_Scope	Credentials
SAP	All users	SAP Credentials/public
SharePoint	All users	SharePoint Credentials/

Table 15: Client Table (SAP)

Resource Server	Authorisation Server
SharePoint	https://AS1
Resource Server 2	https://AS2

During trust establishment (Step A in Figure 11: OAuth 2.0 Modified Architecture above), the Resource Server (SharePoint) sets the credentials for the clients it allows access to its resources as shown in Table 13: Resource Server (SharePoint) Table. It synchronizes that data with its Authorisation Server. In return, the Authorisation Server will only issue tokens to clients in the table after validating their credentials. If, for example, a client C1 comes with a request, it will not be granted a token since it does not have an entry in the table. Such optimization reduces the number of unwanted calls to resource servers such as SharePoint or SAP, since they will be rejected at the authorisation server. During the provisioning process with the client, the Resource Server provides the address of its Authorisation Server so that the client goes there to acquire a token. These clients do not need to ping the resource server and be redirected to the authorisation server as in the original OAuth 2.0 protocol. This also helps in reducing potential DoS or DDoS threats since the resource server does not need to compute every request and redirect it. Instead, it will be rejecting the unwanted requests due to lack of an access token within the request itself. In this case study, both SAP and SharePoint simulated servers shared the same Authorisation

Server and exchanged the location “https://AS1”, in this example. https://AS1 simulates the location of the authorisation server so that the client knows where to go to obtain the authorisation token. In the original OAuth protocol, the client needs to hit the resource server first and gets redirected to the authorisation server. With this proposed optimization, due to the provisioning step, the client can go directly to the authorisation server.

A second optimization that is introduced in this research is the introduction of additional parameter to the Token itself. This parameter is called OAuth_Scope parameter. When SAP requests a Token from AS1, it will be receiving a token with OAuth_Scope parameter = ‘All users’ (if during trust provisioning such client is allowed to act on behalf of all users using a unique key such as user SMTP (Van Staden & Venter, 2011)). SAP (client in this case), therefore, does not have to request a token for every additional request against SharePoint. SAP in this case, can cache the token and make a request against SharePoint for additional claims throughout the lifetime of the token. In this case study, SAP was required to acquire a single token once every 24 hours since the lifetime of the token has been setup to 24 hours. It is important to note that OAuth protocol allows the authorisation server to set the token lifetime. For this case study, a 24 hours lifetime was used. This can be configured based on the security needs of the resource server, in another example, it can use a lifetime of 10 minutes or 1 hour if it is mission critical where security is more important than performance.

Table 16: The Modified Protocol Access Token Parameters below shows the modified protocol access token parameter list.

Table 16: The Modified Protocol Access Token Parameters

	Field	Value	Description
Claims in the token	Response_type	Code	Request parameter is used to identity which grant type the client is requesting
	Client_id	SAP	The name of the principal that issued the token
	Scope	SharePoint	The scope of the access request expressed
	Signature	Public key stamp	Client credential
Response parameters	ExpiresIn	…	The lifetime of the token

	Authorisation_type	Code	The authorisation code that was used to generate the access token
	OAuth_Scope	All users	A parameter to indicate that client is interested in acting on behalf of all users

5.4 Conclusion

As a consumer centric authentication protocol, OAuth is light-weight, secure, and simple identity management protocol. In this chapter, an optimization has been presented that can significantly reduce the number of authentication requests without jeopardizing security requirements. This optimization also reduces unwanted authentication claims and can potentially reduce potential DoS and DDoS threats. To better leverage OAuth 2.0 in the enterprise, two optimizations were proposed. The first optimization is requiring a pre-established trust provisioning step. In this step, an authorisation table is synchronized between the client, the Resource Server, and the Authorisation Server. The second

optimization is introducing OAuth_Scope parameter so that highly trusted clients can authenticate on behalf of users.

In the next chapter, additional optimizations are introduced to target federated identities in the Cloud, including m.

Chapter 6: Federation in the Enterprise

6.1 Introduction to Cloud Federation

Internet identity management is an umbrella that covers several related concerns, all of which stem from the use of multiple services based on various protocols available in the industry, all of which come from different providers and reside in different domains. Each service has a separate identity model and use separate authentication protocol. Developing usable tools that provide fine-grained control over user private data is an emerging problem in the Cloud (M. Hart, 2003), (Gates, 2007). As web users looked for ways to collaborate with others across multiple sites and services, the need for a simple, persistent way to identify oneself became a compelling issue. SAML, OpenID, OAuth are examples of the widely adopted protocols as discussed in previous chapters. In the OpenID scenario, a user creates an account with the Identity Provider of his or her choice and can then use an agent to negotiate authentication. This became a problem as the number of identity providers

became too many, and websites ran out of space displaying various identity providers' logos that a user may have chosen. SAML is a SOAP based protocol that saw wide adoption in its first iterations, however, the world around it changed as developers turned to REST and JSON to write their APIs. OAuth specification aimed to complement OpenID and let users delegate access to a protected resource through an Authorisation Server that issues tokens that do not include users' credentials. OAuth implementation and deployment continues to grow and recently seeing research coverage to provide optimizations in various areas. OAuth provides a worthy promise to organizations planning to leverage the Cloud yet need a solution for marketplace integration in a secure way that guarantees performance requirements. As enterprise organizations leverage Cloud computing, one of the important topics in the Enterprise is the marketplace integration. Whether the enterprise application is hosted in a dedicated tenancy or shared tenancy, the challenges are similar; that is to adopt the right security protocol engineered for security and performance. OAuth protocol is becoming a popular choice for solving such challenges. In the previous chapter, an optimization to OAuth 2.0 was introduced where the Authorisation Server is provisioned with explicit authorisation table so that access grants are rejected at the Authorisation Server before getting to the protected resource. This reduces the amount of processing popular protected resources would have to do and alleviates the risk of potential threats such as Denial-of-Service (DoS) attacks and Distributed DoS (DDoS) attacks (Chao-yang, 2011). In this chapter, the enhancements to the protocol are extended to cover marketplace applications. The notion of referral tokens is introduced to solve the identity federation

challenge for marketplace applications. In this architecture, trust is provisioned and synchronized as a pre-requisite step to authentication amongst all communicating entities using the OAuth protocol. And in the same way, multiple authorisation servers pre-establish trust amongst the federated organizations allowing clients to receive referral tokens that can be validated across organizations.

6.2 Marketplace as a Service MaaS

Marketplace as a Service has emerged as a model of software deployment with its own economics (Vivek Nallur, 2010) (Nallur & Bahsoon, 2010) whereby a provider licenses an application to customer for use as a service on demand. Marketplace software vendors typically host the application on their own web servers or download the application to the consumer device. The concept of marketplace as a Service started to circulate in December 2000 as shown in Bennett et al. "Beginning to Gain Acceptance in the marketplace" (Bennett, et al., 2000), while the "Software as a Service" (SaaS) concept started emerging prior to 1999 (Gilroy, 2009).

Using marketplace can reduce the up-front investment in Software through less costly and on demand pricing from the hosting service providers. The key characteristics of software marketplace include (Konary, 2005):

- Network access and management infrastructure
- Availability and ubiquitous web access

- Centralized feature updating, which avoids the need for downloadable patches and upgrades
- Centralized application delivery model, including architecture, pricing, partnering, and management characteristics;

When Apple opened the first marketplace service on July 10th, 2008, it transformed the scenario and behaviour traditionally presented in software publishing and purchasing to a new stage, that is, the application marketplace. The Apple store was a grand slam with over 10 million applications downloaded in just three days (Shih-Fang, 2010) . Proved successful by Apple, more and more manufacturers and service providers embarked on building and operating their own marketplace services. Examples included Android Market of Google, Ovi Store of Nokia, Windows marketplace of Microsoft, Office marketplace for Office 365, SAP marketplace, and so on.

While Google and Apple made strong headways in application marketplace, enterprise applications are still not up to speed. Authentication is proving to be one of the major issues hampering progress due to performance complications. Unlike Facebook and Google, enterprise applications are not architected from the grounds up to manage the simultaneous access of a huge number of clients, and the need for marketplace is scoped within the organization's boundaries.

There are three authentication types when it comes to marketplace: one is the user authentication, the second type is the application authentication, and the third type is the application federation. An example of application authentication is a Farmville application running inside Facebook, while an example of user authentication is a user logging in to Farmville through Facebook application, and an example of an application federation is accessing Google email of a user in Farmville (because a user is authenticated to Farmville, which is allowed to run in Facebook that is federated with Google). All of these models have various research areas. Central to all is OAuth as a federation protocol, introduced in the next section, as a protocol that can solve the identity challenge in the Cloud.

6.3 Enterprise Federation

It is often required for servers to integrate with each other and exchange protected data. An example of this is SharePoint Online integration with SAP. In the previous chapter, the addition of a provisioning step during which a pre-established trust between the Client and the Resource Server is configured was discussed. This step can reduce many of the unwanted requests to the Resource Server. Also, during this step the client is given information on where to go to acquire a token, in other words, the discovery of the Authorisation Server. In addition to server to server authentication, a third party may want to develop a marketplace application to login into SAP and provide a simplified set of completed financial reports in SharePoint for sharing with colleagues or managers, as an example. Such integration with third party marketplace can be costly in terms of

performance for the enterprise. To illustrate such model, the UML sequence diagram below (Figure 12: UML Sequence Diagram of a marketplace application) is drawn to illustrate each step:

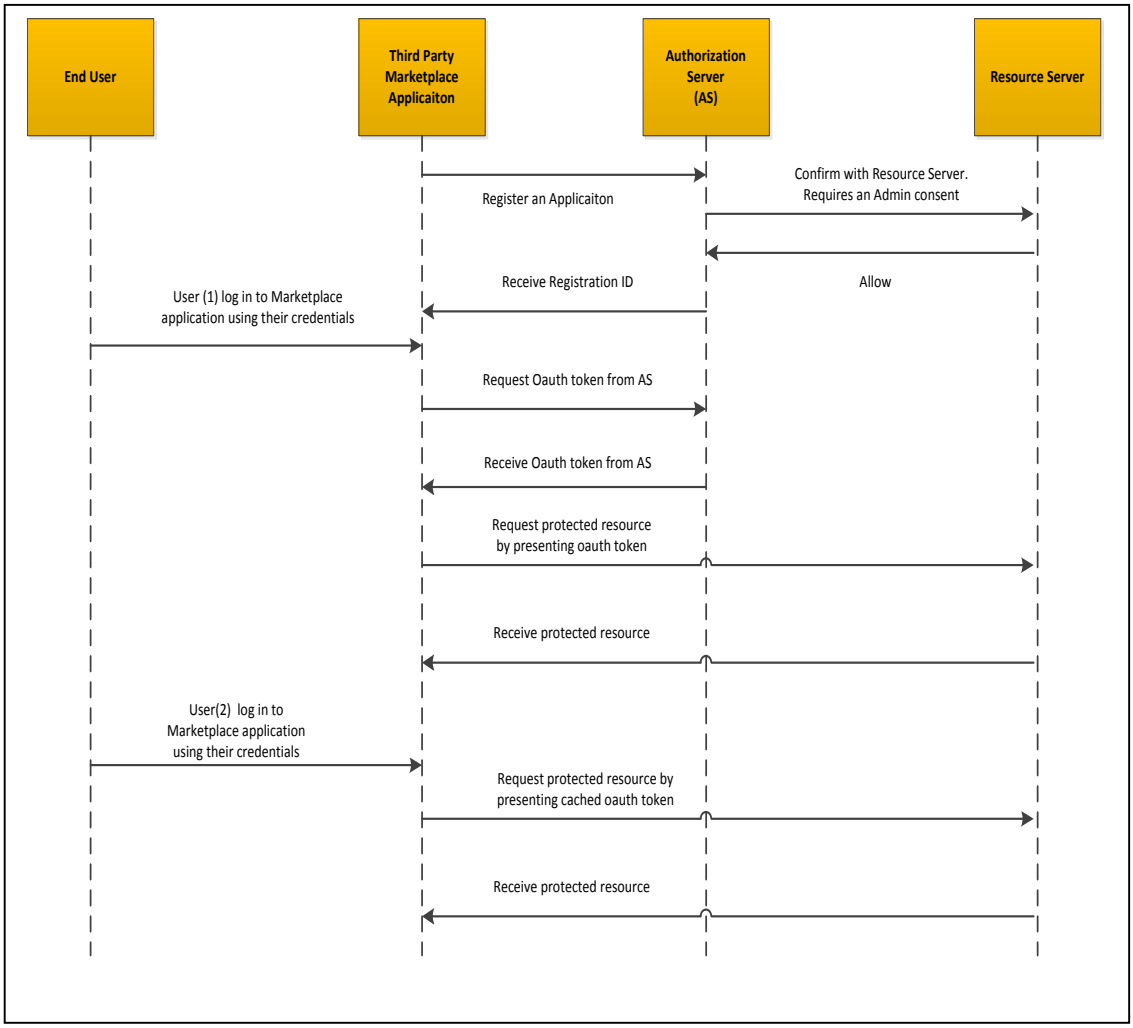


Figure 12: UML Sequence Diagram of a marketplace application

Note that when the second user (user 2) tries to access the protected resource server, the marketplace application does not request a token, instead it reuses the existing token because the protocol allows it to act on behalf of all its users. Such optimization is critical to enterprise application to reduce round-trips to the authorisation server and optimize performance. To make it more complex, the marketplace application may want to pull data from another organization (such as a subsidiary) that contain relevant data. This model is referred to as identity federation. In this research, another set of optimizations is added where the Authorisation server is able to provide a referral token. The referral token is served to facilitate federation across organizations. In this model, a pre-established trust is provisioned between these two organizations Authorisation Servers. This model is a rather simple solution to the identity federation challenge as compared to others far more complex approaches that are difficult and more expensive to implement such as the backend dedicated identity broker (Wang Bin, 2009) or the life-cycle model in (Ji Hu, 2010).

In the proposed model, if a marketplace application is authorised to access protected resources in a Protected Resource Server A (and thus trusted by Authorisation Server A), when the application wants to access a Protected Resource Server B (in organization B), it will be given a referral to Authorisation Server B as the authorisation servers in both

organizations are federated, and a pre-established trust has been setup. This is illustrated in Figure 13: Overall view of the proposed authorisation model below.

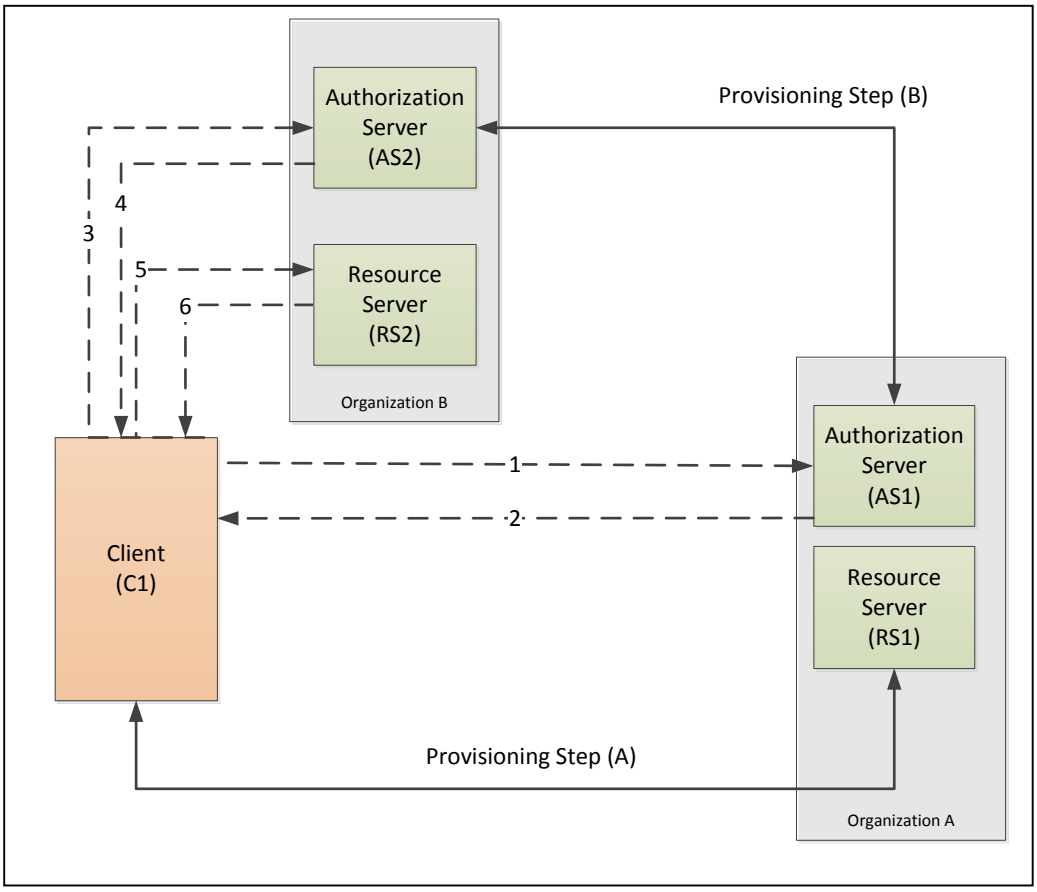


Figure 13: Overall view of the proposed authorisation model

- In step 1, the client C1 is requesting access to Resources in Organization B. However, since the client is pre-provisioned with the Authorisation Server in Organization A, it makes the request there.
- In step 2, the Authorisation Server (AS1) returns back a referral (with an address) to Authorisation Server (AS2) where the client needs to go to get the access token
- In step 3 the client C1 presents the referral token and requests access to Resource Server (RS2)
- In step 4, after the Authorisation Server (AS2) validates the referral token, it grants access token to Resource Server RS2
- In step 5, the client C1 presents the access token to Resource Server RS2 with request to the protected resource. Note that Resource Server only able to validate tokens issued by its Authorisation Server (AS2). The fact that the client is pre-configured with Authorisation Server (AS1) is not relevant to the Resource Server
- In step 6, the client C1 receives the requested resource.

6.4 Federation Case Study

In this case study, Enterprise application SharePoint Online in Organization A (OrgA as a subscriber to Office 365) and marketplace application (EasyAudit) are interested in sharing

Client	Credentials

protected resources on behalf of their users. EasyAudit is a third party marketplace application developed by an independent Enterprise called EasyAudit.com. SharePoint online is an Enterprise application available in the Cloud and scoped for OrgA. SharePoint for OrgA is provisioned to trust an Independent Authorisation Server known as AS1. The Authorisation Server is developed so it only issues tokens to trusted marketplace applications that present a valid license purchased through Office 365 licensing model for marketplace.

In order to simulate this, a Resource Server (RS1) is developed and configured with Authorisation Table as shown in Table 17: Authorisation Server (AS1) below. This table is synchronized on the Authorisation Server (AS1). Resource Server (RS1) can only allow access to applications presenting OAuth tokens issued by AS1.

Table 17: Authorisation Server (AS1)

EasyAudit	EasyAudit Credentials (Public Key)

The following steps show the functional flow for the authentication protocol:

1. The user types a URL in a web browser to go to a SharePoint page where EasyAudit is installed.

2. The page is detected by SharePoint processes indicating that there is a component from the EasyAudit.com app on the page. SharePoint must get a token that it can send to the EasyAudit.com app. SharePoint asks AS1 to create a token or provide an synchronize it to SharePoint for future use

3. AS1 requests licensing information from EasyAudit.com and creates an entry upon valid presentation of the licensing information and then synchronizes the data to SharePoint.

4. EasyAudit can cache the token and present it to AS1 in future service calls to SharePoint. It can reuse the token for all of its users where a single call to AS1 can serve the authentication needs of all users of EasyAudit.com

5. SharePoint returns the information that EasyAudit.com requested in its browser frame.

6. The EasyAudit.com app renders the browser frame contents as requested by the user. The user now sees the SharePoint page fully rendered completing the OAuth transaction process.

The following table (Table 18: Access token parameters for the modified protocol) shows the access token parameters for the modified protocol.

Table 18: Access token parameters for the modified protocol

	Field	Value	Description
Claims in the token	Response_type	Code	Request parameter is used to identity which grant type the client is requesting
	Client_id	EasyAudit.com	The name of the principal that issued the token
	Audience	SharePoint, OrgA	The scope of the access request expressed
	Signature	Public key stamp	Client credential
Response Parameters	ExpiresIn	24hrs	The lifetime of the token
	Authorisation_type	Code	The authorisation code that was used to generate the access token

	OAuth_Scope	All users	A parameter to indicate that client is interested in acting on behalf of all users
	Referral	Referral Address	The address where the token need to be issued from

In the next step, it is assumed that SharePoint Online for OrgA wants to federate trust with Organization B (OrgB). OrgB is a subsidiary of OrgA, and it is important for EasyAudit to access resources from both organizations to present the accurate reports. In this case, AS1 is configured to issue access tokens that can be authenticated by AS2. When EasyAudit token parameter 'Audience' is addressed to OrgB, AS1 response parameter token is returned with a referral to AS2 along with the referral address https://AS2. In this step, the client (EasyAudit) presents the referral token to AS2 at the address provided and obtains a token that can be presented to protected SharePoint resources in OrgB. No exchange of credentials is required between AS2 and the client (EasyAudit). AS1 is configured as below in Table 19: Authorisation Server (AS1) Table Supporting Identity Federation.

Table 19: Authorisation Server (AS1) Table Supporting Identity Federation

Client	Credentials	Federation
EasyAudit	EasyAudit Credentials (Public Key)	AS2, Https://AS2

In the next step, the case study analysed in this chapter is compared with a pure OAuth protocol use case. To do the study, Office 365 is simulated with 1000 users all provisioned with a license to run SharePoint online. 500 of these users are in OrgA and 500 are in OrgB where a federation is established per the optimization addressed in this chapter. Each of these users is simulated to have EasyAudit service installed and ready for use. When the first user in OrgA requests access to EasyAudit report, SharePoint server loads EasyAudit in its frame, EasyAudit then obtains a token from Authorisation Server (AS1) and presents it to SharePoint, SharePoint validates the token, caches it to 24 hours and allows access to the user. When user 2 requests access to the EasyAudit service, SharePoint uses the same cached token, it is able to do that because the trust is between the servers and the servers have the right to act on behalf users, no additional trips to Authorisation server is required for either of the servers. When user 1 in OrgB requests access to EasyAudit, SharePoint in OrgB loads EasyAudit in its frame and requests a token. EasyAudit makes a round trip to AS1 to obtain a token; it is then referred to AS2 to obtain the token. EasyAudit server obtains the token and presents it SharePoint in OrgB. The user is authenticated; the token is cached and reused for the rest of users 2...500 in OrgB. If using pure OAuth protocol without the proposed optimization, such use case will require 1000 trips to the Authorisation server, 500 calls in each of the organizations, while with proposed optimization, the Market Place client only needed to make 3 trips. The first call is made to the first Authorisation server (AS1) for

OrgA users, the second trip to AS1 for OrgB, and the third trip to the federated Authorisation Server (AS2) via a referral token. The following table (Table 20: Results of Case Study) summarizes these results.

Table 20: Results of Case Study

Number of users in the Study	Number of calls required to AS per OAuth Protocol	Number of calls required to AS per proposed OAuth protocol optimization
1000	1000	3

It is important to note that this ratio of 1000:3 is arbitrary and for demonstration of the optimization. The optimization will apply for any number of clients, in other words, this ratio could be 100000:3, etc. The optimization basically eliminates the dependency on number of clients trying to access the protected resource, using this optimization, the number of round trips to the authorization server is always the same and equal to 3 trips.

In more practical terms, and to illustrate how this is implemented in real world application for the authentication amongst SharePoint and marketplace as illustrated in (Microsoft, 2012), the client builds a HTTP "POST" request to the AS (Authorisation Server) endpoint and

includes the following mandatory parameters as defined in OAuth-WG Device Profile (OAuth-WG, 2010)

- oauth _identifier, the identifier of the client.

The client can also include the following optional parameters, as well as any additional parameters as defined by the authorisation server:

- OAuth_Server_Scope: This parameter should be used if the authorisation server has defined a method for the client to request certain capabilities of the access token.

Since the requests are sent via plain text, the server may require the user of TLS or SSL.

For example, the following HTTPS request is made by the client:

- POST /request_token HTTPS/1.1
- Host: server.as2.com
- oauth _identifier=easyaudit.com

When the client sends the authorisation request, a user verification code and a device verification code is generated by the authorisation server. These are included in the HTTP response body using the "application/x-www-form-url-encoded" content type as defined by [W3C.REC-html40-19980424] with a 200 status code (OK). The following PREREQUISITE parameters are included in the response:

- oauth_device_id, the device code that does the verification.
- oauth_user_id, the user code that does the verification.

- oauth_verification_url, the user URL that does the verification on the authorisation server.

The following parameters may also be included, as well as any additional parameters as defined by the authorisation server:

- oauth_verification_token_expires_in, the lifetime of the two verification codes in seconds.
- oauth_verification_rate_limit, the minimum amount of time in seconds that the client should wait between polling requests to the device authorisation URL.

For example:

- HTTPS/1.1 200 OK
- Content-Type: application/x-ww -url-encoded
- oauth_device_code=7u8TKKcKB&oauth_user_code=5696&oauth_verification_url=http%2A%2M%2Fwuw%2Zexample%2Gcom%2Fdevice

The verification code and the user verification URL must be displayed by the client to the end user. The client must instruct the user to visit the user verification URL in a web browser, and to enter the user verification token upon doing so.

This research does not cover the way in which the authorisation server handles the authorisation request, including whether it uses a secure channel such as TLS/SSL (OAuth-WG, 2010).

The example illustrates the authorisation case and the denial case for an authorisation server. This case study shall better explain the proposed solutions and how they work in

practice. Overall, the simplification is illustrated, and the optimization is clear where the calling application needs to authenticate with the Authorisation Server once to obtain the token, the token then can be reused for all its users. Such optimization is necessary to ensure the scale needed for Enterprise applications to practically leverage OAuth protocol.

6.5 Conclusion

As a consumer centric authentication protocol, OAuth is light-weight, secure, and simple identity management protocol. In this chapter, it was shown an optimization that can significantly reduce the number of authentication requests without jeopardizing security requirements. The same model is enhanced to solve the federation challenge. To better leverage OAuth 2.0 in the enterprise, this chapter proposed two modifications. The first modification requires a pre-established trust provisioning step. In this step, the proposal calls for synchronized authorisation table between the client, the Resource Server, and the Authorisation Server. The second modification is the introduction of the referral parameter to the protocol so that various Authorisation Servers can refer requests to each other and thus federate trust. Organizations that want to leverage this model will need to pre-establish trust with this global Authorisation Server. This would also potentially provide the platform that would allow for the creation of a unified Cloud identity. Some of these modifications are currently being considered for adoption as part of the next version of the OAuth protocol.

Chapter 7: Conclusion

7.1 Summary

This research addressed some of the challenges that large enterprises face when hosting Cloud solutions. In the beginning of the research, Cloud computing was researched with a focus on performance optimization. In the area of performance optimization, the area of capacity planning was researched since it is a major challenge for Cloud providers opting to reduce the cost and meet SLAs for Cloud providers. In addition to capacity planning, the area of security protocol adoption was researched due to the challenge of adopting an authentication protocol that meets the performance needs of a large enterprise. It is usually an issue in the enterprise due to the cost of performance overhead. Finding a secure solution without a performance hit is a major challenge for adopting Cloud computing. The research on capacity planning has led to the development of the Capacity Planning Calculator for Lync Server. In addition, the research has covered Cloud performance optimization for the marketplace (third-party developers that want to authenticate with enterprise applications), as this is an open challenge for large enterprises adopting OAuth protocol. With such optimizations, Cloud providers can minimize the cost, guarantee SLAs, and secure access to data resources, all of which are key aspects for enabling better Cloud adoption in the Enterprise.

7.2 Limitations & Future Work

It can be argued that one of the limitations of this work is that capacity-planning validation was only done via one real-life case study, the MS Lync platform, with wide industrial adoption of the results by providers using the MS Lync platform. Although this met our research objective and demonstrated clearly the level of efficiencies that can be achieved by better and more reliable resource requirement prediction at the planning stage, the next target would be to expand and generalize the applicability of our methodology to other Cloud application domains. This will involve studying the effect of how the actual inter-arrival of incoming requests of the different application types are distributed and how the service time durations are temporally distributed. For example, there might be a considerable difference between application types where the incoming requests are distributed in accordance to the Poisson distribution as compared to other types of distribution (e.g. the response time of M/M/1 queues versus that of G/G/1 queues).

Finally, future work will further investigate the reason behind the differences between the results produced by our methodology (section 3.3) and the real values measured (section 3.5). For example, in the first experiment (Table 3) there was a -10% difference between the CPU performance computed with the methodology and the simulator; in the third experiment (Table 5) it was +16%. This should help us better enhance our methodology by identifying a more precise overprovisioning upper bound margin which is currently set based on statistical data.

References

Allspaw, J., 2008. *The Art of Capacity Planning: Scaling Web Resources.* 1 edition ed. San Francisco, CA: O'Reilly Media.

Archer, D., Pushlmann, N., Boehme, A. & Kurtz, P., 2011. *Security guidance for critical areas of focus in cloud computing v3.0.* Miami, FL, Cloud Security Alliance.

Bashroush, R. & Noureddine, M., 2012. A Cost Effective Cloud Datacenter Capacity Planning Method Based on Modality Cost Analysis. *International Journal of Communication Networks and Distributed Systems.*

Bennett, k. et al., 2000. *The Future for Flexible Software.* Singapor, APSEC.

CA, 2010. *Utilization and Datacenter productivity.* [Online]
Available at: http://www.ca.com/files/technologybriefs/dca-manager-tech-brief-us.pdf
[Accessed 11 July 2012].

Chao-yang, Z., 2011. *DOS attack analysis and study of new measures to prevent.* Wuhan, IEEE.

Chris Matthews, Y. C., 2009. *Virtualized recomposition: Cloudy or Clear?.* Vancouver, BC, Software Engineering Challenges of Cloud Computing, 2009. CLOUD '09. ICSE Workshop, pp. 38-43.

Daniel Gmach, J. R. L. C. A. K., 2007. *Workload Analysis and Demand Prediction of Enterprise Data Center Applications.* Boston, MA, IEEE.

Dymond, M. J. a. P., 1998. *A Plugin-Based Privacy Scheme for World-Wide Web File Distribution.* Kohala Coast, HI , IEEE, pp. 621-627.

Eghbal Ghazizadeh, M. Z. a. J.-l. A. M., 2012. *A Survey on Security Issues of Federated Identity in the Cloud Computing.* Taipei, IEEE 4th International Conference on Cloud Computing Technology and Science.

Ernest, S. & Foo, A., 2009. A user-centric federated single signon system. *Journal of Network and Computer Applications,* 32(2), pp. 288-401.

Fang, Z., Ying, C. & Lin-Ping, W., 2005. "o,". *Computer&Digital Engineering,* Volume 33, no. 1, pp. 81-84.

Fedorova, A., Seltzer, M. & Smith, M. D., 2008. *Improving Performance Isolation on Chip Multiprocessors via an Operating System Scheduler.* Brasov, IEEE.

Ferraz, A. C. C. M. a. C. A. G., 2005. *Guidelines for performance evaluation of web services.* New York, NY, ACM, pp. 1-10.

Gates, D. C. a. E., 2007. *Access Control Requirements for Web 2.0 Security and Privacy.* New York, NY, W2SP.

Gilroy, A., 2009. Microsoft, Nokia Add Phone App Stores. *This Week in Consumer Electronics (TWICE),* 24(5), pp. 1-8.

Gokhale, J. L. a. S. S., 2008. *Resource Provisioning in an E-commerce Application.* Washington, DC , IEEE.

Gopalakrishnan, T. & Vaidehi, M., 2011. *Efficient resource arbitration and allocation strategies in cloud computing through virtualization.* Beijing, Proceedings of IEEE CCIS.

Graham Kirby, A. D. A. M. a. F., 2010. *An Approach to Ad hoc Cloud Computing.* Fife, Scottland, Arxiv, pp. 1-6.

Gray, N. A. B., 2004. *Comparison of web services, java-RMI, and CORBA service implementations.* Sydney, ASWEC.

Hasselmeyer, P. & d'Heureuse, N., 2011. *A System for Data Center Performance Estimation.* Minneapolis, Minnesota USA, ICDCS.

IEFT, 2010. *WS-Federation.* [Online]
Available at: http://msdn.microsoft.com/enus/library/bb498017.aspx
[Accessed 4 2011].

IEFT, 2011. *OAuth WRAP.* [Online]
Available at: http://tools.ietf.org/html/draft-hardt-oauth-01
[Accessed 17 4 2011].

IETF, 2010. *WS-Trust.* [Online]
Available at: http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.html
[Accessed 4 2012].

Intel, 2013. *HyperThreading.* [Online]
Available at: http://www.intel.com/content/www/us/en/architecture-and-technology/hyper-threading/hyper-threading-technology.html
[Accessed 2014].

J. Somorovsky, A. M. J. S. M. K. a. M. J., 2012. *On breaking saml: Be whoever you want to be.* Berkeley, CA, IEEE.

J. Zhang, M. Y. a. R. C., 2007. *Application resource demand phase analysis and prediction in support of dynamic resource provisioning.* New York, NY, IEEE.

Jacob Bellamy-McIntyre, C. L. G. W., 2011. *OpenID and the Enterprise: A Model-based Analysis of Single Sign-On Authentication.* Helsinki, 15th IEEE International Enterprise Distributed Object Computing Conference.

Ji Hu, X. F. R. F., 2010. *Business Driven Trust Federation Management fo Service Marketplaces.* Miami, FL, IEEE.

Jiyin Li, M. Q. J.-W. N. Y. Z. M., 2010. *Adaptive Resource Allocation for Preempt able Jobs in Cloud Systems.* Philadelphia, PA , IEEE International Conference on Intelligent Systems Design and Applications, pp. 31-36.

Jones, M., Rosu, D. & Rosu, M., 1997. *CPU reservations and time constraints: Efficient, predictable scheduling of independent activities..* Saint-Malo, France, ACM.

Jun, J. & You, H., 2010. *A Mechanism to Prevent RP Phishing in OpenID System.* Yamagata, IEEE, pp. 876-880.

K. Daniel, T. T. C. W., 2008. *Interoperable Role-Based Single Sign-On-Access to Distributed Public Auhority In- formation Systems.* Boston, MA, IEEE.

K. Tang, S. C. D. L. J. Z. a. B. Y., 2006. *A performance evaluation of web services security.* Hong Kong, IEEE, pp. 67-74.

Konary, E. T. a. A., 2005. 2005 Software as a Service Taxonomy and Research Guide. *IDC,* p. 7.

Layouni, F. & Pollet, Y., 2009. *An Ontology-Based Architecture for Federated Identity Management.* Bradford, IEEE.

Lenk, A., Llems, M., Nimis, J. & Tai, S., 2009. *What's inside the cloud? An architectural map of the cloud landscape.* Palo Alto, CA, ACM.

Li-chun, P., Xing-yuan, C., Ting, W. & Zhang, B., 2008. Credit-based dynamic authentication mechanism crossing heterogeneous domains. *Computer Applications,* Volume vol. 28, no. 6, pp. 1382-1384.

Liu, H., Pallickara, S. & Fox, G., 2005. *Performance of web services security.* Proceedings of the 13th Annual Mardi Gras Conference, IEEE.

Lu, S. & Gokhale, S., 2006. *Performance and availability analysis of e-commerce sites.* Chicago, IL , IEEE.

Lynch, L., 2011. Inside the identity management game. *Internet computing,* IEEE Volume 15(Issue 5), pp. Pages 78-82.

M. Hart, R. J. a. A. S., 2003. *More Content - Less Control: Access Control in the Web 2.0.* Miami, FL, IEEE.

Madhukar R. Korupolu, A. S. B. B., 2009. *Coupled placement in modern data centers.* Rome, IEEE.

Madsen, P., Koga, Y. & Takahashi, K., 2005. *Federated identity management for protecting users from ID theft.* New York, ACM, pp. 77-83.

Manshan Lin, H. G., 2001. Present Situation and Development of single sign-on technology. *Journal of Computer Applications,* 24(6), pp. 248-250.

Marouf, M. S. a. S., 2012. Recommendation Models for Open Authorization. *IEEE transactions on dependable and secure computing,* July/August.Volume Vol. 9, NO. 4.

Menasce, D. A., 2002. QoS issues in Web services. *IEEE Internet Computing,* Volume vol. 6, pp. 72-75.

Menasce, D. & Bernnani, M., 2006. *Autonomic virtualized environments.* Silicon Valley, CA, IEEE.

Microsoft, 2011. *Lync Server 2010 Capacity Calculator.* [Online]
Available at: http://www.microsoft.com/en-us/download/details.aspx?id=12295

Microsoft, 2012. *Implement OAuth in your Marketplace App.* [Online]
Available at: http://msdn.microsoft.com/en-us/library/windowsazure/gg193416.aspx
[Accessed 27 1 2014].

Microsoft, 2012. *Lync Server 2010 Planning Guide.* [Online]
Available at: http://www.microsoft.com/downloads/en/details.aspx?FamilyID=2da9fa26-e032-4dcf-b487-da916ddc508f

Microsoft, 2012. *Lync Server 2010 Stress and Performance Tool.* [Online]
Available at: http://go.microsoft.com/fwlink/?LinkId=212599

Microsoft, 2012. *Lync Server 2010 User Models.* [Online]
Available at: http://technet.microsoft.com/en-us/library/gg398811.aspx

Microsoft, 2012. *Microsoft Lync Server.* [Online]
Available at: http://office.microsoft.com/en-us/lync/
[Accessed 7 2013].

Microsoft, 2012. *Microsoft Lync Server 2010 Planning Tool.* [Online]
Available at: http://go.microsoft.com/fwlink/?LinkId=206134

Microsoft, 2012. *What's new in authentication for SharePoint 2013.* [Online]
Available at: http://labellesharesonsharepoint.wordpress.com/2013/01/10/whats-new-in-authentication-for-sharepoint-2013/
[Accessed 2 2013].

Moralis, A. et al., 2007. *Performance comparison of web services security: Kerberos token profile against X.509 token profile.* Athens, IEEE.

Nallur, V. & Bahsoon, R., 2010. *Design of a market-based mechanism for quality attribute tradeoff of services in the cloud.* New York, NY, ACM.

Noureddine, M. & Bashroush, R., 2011. *A Performance Optimization Model towards OAuth 2.0 Adoption in the Enterprise.* Greece, IEEE.

Noureddine, M. & Bashroush, R., 2011. *A provisioning model towards OAuth 2.0 performance optimization.* London, U.K., IEEE, pp. 76-80.

Noureddine, M. & Bashroush, R., 2011. Cost Effective Datacenter Capacity Planning Analysis Using Modality Cost Methodology.. *Ubiquitous Computing and Communication Journal (UBICC).*

Noureddine, M. & Bashroush, R., 2011. Modality cost analysis based methodology for cost effective datacenter capacity planning in the cloud. *nternational Journal of Communication Networks and Distributed Systems,* Volume 11(N 3).

Noureddine, M. & Bashroush, R., 2011. *Modality Cost Analysis: A Methodology for Cost Effective Datacenter Capacity Planning in the Cloud.* Jordan, ICICS.

Noureddine, M. & Bashroush, R., 2013. An Authentication Model towards Cloud Federation in the Enterprise. *Journal of Systems and Software.*

OAuth, 2012. [Online]
Available at: http://tools.ietf.org/html/draft-hardt-oauth-01

OAuth-WG, 2010. *OAuth-WG Device Profile.* [Online]
Available at: http://www.ietf.org/mail-archive/web/oauth/current/msg01346.html
[Accessed 3 2013].

Pai, S., Sharma, Y., Kumar, S. & Pai, R., 2011. *Formal Verification of OAuth 2.0 using Alloy Framework.* Katra, Jammu , IEEE.

Pai, S., Sharma, Y., Pai, R. & Singh, S., 2011. *Formal verification of oauth 2.0 using alloy framework.* Katra, Jammu , IEEE.

Peter Bodík, R. G. C. S. A. F. M. I. J. a. D. A. P., 2009. *Automatic Exploration of Datacenter Performance Regimes.* Orlando, FL , Automated Control for Datacenters and Clouds.

Ragouzis, N. et al., 2008. Security assertion markup language (saml) v2.0 technical overview. *Security Services Technical Committee of OASIS.*

Rolia, J., Cherkasova, L., Arlitt, M. & Andrzejak, A., 2005. *A Capacity Management Service for Resource Pools.* Spain, IEEE.

Sean Kenneth Barker, P. S., 2010. *Empirical evaluation of latency-sensitive application performance in the cloud.* New York city, NY, ACM.

Shih-Fang, C., 2010. *Application Marketplace as a Service, A Reference Architecture for Application Marketplace Service.* Orlando, FL, IEEE.

Shirasuna, S., Slominski, A., Fang, L. & Gannon, D., 2004. *Performance comparison of security mechanisms for grid services.* Miami, FL, IEEE, pp. 360- 364.

Simmons, B., McCloskey, A. & Lytfiyya, H., 2007. *Dynamic Provisioning of Resources in Data Centers.* Orlando, FL, IEEE.

Song, Y., Li, Y. & Wang, H., 2008. *Service-oriented priority-based resource scheduling scheme for virtualized utility computing.* Miami, FL, IEEE.

SPECint Processor Benchmarking, 2012. [Online]
Available at: URL: http://www.spec.org
[Accessed 4 2012].

Sukumar, R., 2011. *Platforms for Building and Deploying Applications for Cloud Computing.* Miami, FL, IEEE, pp. 6-11.

Sun Microsystems, 2010. *Metro Web Services Interoperability Technology (WSIT), https://wsit.dev.java.net/..* [Online]
Available at: https://wsit.dev.java.net
[Accessed 13 12 2012].

Suriadi, S., Foo, E. & Josang, A., 2009. A user-centric federated single sign-on system. *Journal of Network and Computer Applications,* pp. 288-401, vol 32.

Symantec, 2010. *DoS.* [Online]
Available at: http://www.symantec.com/security_response/glossary/define.jsp?letter=d&word=dos-denial-of-service-attack
[Accessed 10 6 2014].

Takeda, Y. et al., 2006. *Avoidance of performance bottlenecks caused by http redirect in identity management protocols.* New York, ACM, pp. 25-32.

Van Staden, F. & Venter, H., 2011. *Adding digital forensic readiness to electronic communication using a security monitoring tool.* Johannesburg, IEEE.

Vivek Nallur, R. B., 2010. Self-adapting Applications Based on QA Requirements in the Cloud Using Market-Based Heuristics. *ServiceWave ,* pp. 51-62.

Wang Bin, H. H. Y. L. X. X. X. J. M., 2009. *Open Identity Management Framework for SaaS Ecosystem.* Macau, IEEE.

Wang Xiuyi, W. L. J. C. Q., 2007. Security Research on a SAML-based Single Sign-on implement mode. *Microcomputer Information,* 23(8-3), pp. 81-83.

Wang, 2011. An Analysis of Web Single Sign-On.

Wang, R., Chen, S. & Wang, X., 2012. *Signing Me onto Your Accounts through Facebook and Google: a Traffic-Guided Security Study of Commercially Deployed Single-Sign-On Web Services.* Oakland, CA, IEEE.

Warneke, D. & Odej, K., 2011. Exploiting Dynamic Resource Allocation for Efficient Parallel Data Processing in the Cloud. *Parallel and Distributed Systems,* 22(6), pp. 985 - 997.

Wood, T., Cherkasova, L., Ozonat, K. & Shenoy, P., 2008. *Profiling and Modeling Resource Usage of Virtualized Applications.* New York, NY, USA, ACM.

Yagiz Onat Yazir, C. M. R. F., 2010. *Dynamic Resource Allocation in Computing Clouds using Distributed Multiple Criteria Decision Analysis.* Miami, FL , IEEE Third International Conference on Cloud Computing, pp. 91-98.

Yang, F. & Manoharan, S., 2013. *A security analysis of the OAuth protocol.* Victoria, BC , IEEE.

Yan, L., Rong, C. & Zhao, G., 2009. *Strengthen cloud computing security with federal identity management using hierarchical identity- based cryptography.* s.l., 1st International Conference on Cloud Computing, CloudCom 2009.

Ye Hu, J. W. G. I. a. M. L., 2009. *Resource Provisioning for Cloud Computing.* Riverton, NJ, USA , ACM.

Yebin Chen, B. W. B. X. L. S., 2011. *Design of web service single sign-on based on ticket and assertion.* Deng Leng , IEEE.

Yexi Jiang, C.-S. P. T. L. C. R., 2013. Cloud Analytics for Capacity Planning and Instant VM Provisioning. *IEEE Transactions on Network and Service Management,* 10(3).

Zarandioon, D., Yao, D. & Ganapathy, V., 2009. *Privacy-aware identity management for client-side mashup applications.* Chicago, IL, ACM, pp. 21-30.

Zhen-Guo, D., Duo-Zheng, L. & Mei-Ling, L., 2009. Optimistic dynamic authentication based on digital signature. *Computer Engineering and Design,* Volume vol. 30, no. 15, pp. 3511-3513.

Zhenxiang Tu, Q. L., 2012. *Design and implementation of unified identity management system based on SAML.* Yichang, IEEE, pp. Page(s): 3178 - 3181.

Zhiliang Zhu, J. B. H. Y. Y. C., 2011. *SLA Based Dynamic Virtualized Resources Provisioning for Shared Cloud Data Centers.* Washington, DC, IEEE.

Zhu, X., Young, D. & Watson, B., 2008. *1000 Islands: Integrated Capacity and Workload Management for the Next Generation Data Center.* Chicago, IL , IEEE, pp. 172-181.