Science and Information Conference 2015
July 28-30, 2015 | London, UK

Software as a Service: Understanding Security Issues

Pushpinder Kaur Chouhan
Centre for Secure Information Technologies
Queen's University Belfast
Northern Ireland, UK
Email: p.chouhan@qub.ac.uk

Feng Yao
Centre for Secure Information Technologies
Queen's University Belfast
Northern Ireland, UK
Email: fyao02@qub.ac.uk

Sakir Sezer
Centre for Secure Information Technologies
Queen's University Belfast
Northern Ireland, UK
Email: s.sezer@ecit.qub.ac.uk

***Abstract*—Our life has been significantly enhanced and improved by a wide range of multifarious software services. Technological advancements and availability of high speed Internet connection has enabled software providers to transform traditional on-premise software deployment model to off-premise centralized deployment model. Providers and users are attracted by the software as a service (SaaS) paradigm due to the appealing features such as pay-as-you-go and optimal hardware utilization it can offer. However, lots of enterprises are still reluctant to move their business into SaaS because of the security concerns. Security issues plaguing the SaaS environment cuts across various areas which could lead to increased attack surface for potential security threats. This article first introduces the architecture of SaaS and highlights its utility and applicability in different environments like cloud computing, mobile cloud computing, software defined network and Internet of things. Then elaborate the SaaS security challenges spanning across data security, application security to SaaS deployment security and how security issues might affect different layers in the SaaS architecture.**

***Keywords*—Software-as-a-Service; SaaS Architecture; SaaS security; Cloud security**

I. Introduction

Software is ubiquitous in every aspect of business today. Software applications manage large inventories, train employees, track shipments across multiple countries, and even help to form good working relationships with users.

Software makes services better and in return services help to make software better. By bringing the best of both the worlds in Software as a Service (SaaS), the choice of flexibility and capabilities is maximized. SaaS is a software application delivery model where providers deliver web-based applications to users over the internet. SaaS users can then pay a per usage subscription fee to use software instead of owning the entire suite. SaaS has revolutionized the way people build, sell, buy, and use software. In recent years, it has rapidly become the emerging paradigm for delivering application to development, production and management environments. Survey predictions [1] suggest that SaaS is set to grow and account for up to 85% of all new software, and SaaS worldwide revenue is projected to reach $22.1 billion by 2015.

Thus, as the requirement of SaaS market expands, a growing number of software service providers are seizing every opportunity to jump on the bandwagon and transform their conventional software service model into SaaS. The inherent business model of SaaS allows enterprises to monetize their businesses, saving costs and raising productivity and profits. However, SaaS security is one of the most important factors to hamper the wide deployment and utilization of SaaS.

The advantages of considering SaaS in various scenarios and across various application providing environments have been highlighted in [2], [3]. However, challenges still exist for a full-scale application delivery implementation of SaaS. A number of these challenges in terms of cloud computing have been presented in [4], [5]. It is highly critical to contend with and approach security issues to maintain a reliable platform in SaaS. This article focuses on the state-of-art security problems existing in contemporary SaaS framework encapsulating various areas and technologies which are indispensable elements comprising the SaaS architecture.

An analysis of the security challenges of SaaS is presented in this article. Furthermore, the effect of SaaS security issues (data security, application security and deployment security) are mapped unto the SaaS architecture layer to provide further insight into these issues as they relate to SaaS. The proposed and emerging solutions to these challenges are then discussed and categorized according to SaaS security challenges. In-depth analysis of loopholes and associated possible countermeasures discussed in this article are envisaged to be useful to the wider research community and industry as a reference to take preemptive action against the emerging security threats to SaaS.

The organization of article is described as follows. In section II the evolution of SaaS and its main features are introduced. Section III presents the architecture of SaaS and the brief functionalities of its constituent components. Section IV provides an all-round introduction of SaaS security challenges encompassing data security, application security and SaaS deployment security. Finally, the conclusion of this article is given in section V.

II. Software as a Service

Software as a Service (SaaS) [6] refers to the applications delivered as services over the Internet. SaaS service providers host the software and perform all the necessary maintenance operations (hardware purchase, software installation, manage-

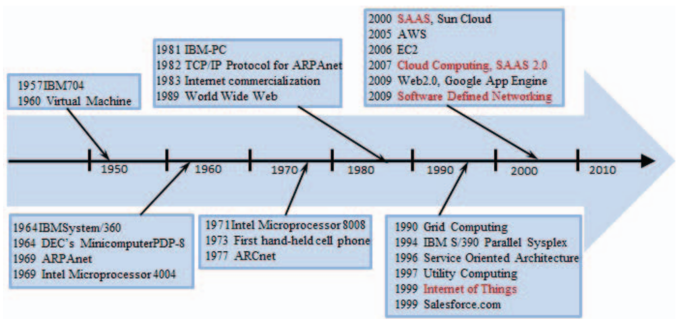
1957IBM704
1960 Virtual Machine
1981 IBM-PC
1982 TCP/IP Protocol for ARPAnet
1983 Internet commercialization
1989 World Wide Web
2000 SAAS, Sun Cloud
2005 AWS
2006 EC2
2007 Cloud Computing, SAAS 2.0
2009 Web2.0, Google App Engine
2009 Software Defined Networking
1950
1960
1970
1980
1990
2000
2010
1964IBMSystem/360
1964 DEC's MinicomputerPDP-8
1969 ARPAnet
1969 Intel Microprocessor4004
1971Intel Microprocessor8008
1973 First hand-held cell phone
1977 ARCnet
1990 Grid Computing
1994 IBM S/390 Parallel Sysplex
1996 Service Oriented Architecture
1997 Utility Computing
1999 Internet of Things
1999 Salesforce.com

Fig. 1: Roadmap to SaaS and beyond.

ment etc.) thus offering their clients easy access to the software for use.

SaaS has emerged as a new software delivery service in recent years mainly relying on the combination of the technologies (shown in the Figure 1). However, the concept behind SaaS has been evolving since 1950s. Implementation by research and industry groups includes large-scale mainframe computers (1950), Virtualization concept (1969), Remote Job Entry (1970), Grid Computing (1990), Service Oriented Architecture (1996) and Utility Computing (1997).

SaaS concept first emerged in 2000 by simply offering stand-alone applications, followed by the on-premise SaaS model and subsequently off-premise SaaS model encompassing Cloud Computing (2007), Mobile Cloud Computing (2008), and Software Defined Networking (2009).

The one-time licensing model is commonly used for on-premise software. In contrast, SaaS application access is frequently sold using a subscription model, with users paying an ongoing fee to use the application. While the on-premises SaaS model involves installing and running software application on computers on the users location, the off-premise SaaS model refers to the model in which software and associated data are hosted and stored in the cloud facilities. Off-premises SaaS model takes advantage of the benefits of centralization through a single-instance, multi-tenant architecture, and also provides a feature-rich experience competitive with comparable on-premise applications. From the viewpoint of SaaS providers, multi-tenancy and virtualization technologies significantly improves resource utilization and efficiency over previous technologies [7]. According to [1], SaaS is set to grow up to 5 times faster than on-premise software.

TABLE I: Advantage of SaaS for users and providers.

Benefits	SaaS User	SaaS Providers
Customizability	User specific	Group same user demand
Maintenance cost	Low	Low
Management cost	Low	Low
Mobile Extension	Feasible	Easy
Offsite deployment	Reduced hosting cost	Maximized efficiency
Organization	Decentralized use	Centralized management
Payment	Pay as you go	Flexible metering
Scalability	Easy	Varies
Service availability	24/7	High customizability
Stability	Improved	Maximized

SaaS can also take advantage of Service Oriented Architecture (SOA) to enable software applications to communicate with each other. Each software service can act as a service provider, exposing its functionality to other applications via public brokers, and can also act as a service requester, incorporating data and functionality from other services. SOA provides service-oriented services, which are deployed and provisioned by SaaS.

In a SaaS environment, users usually access the application through a web browser or thin client software installed on users devices. On the technical side, users are unhindered by application upgrade and patch deployment as the application maintenance is handled transparently by the SaaS provider. From the financial viewpoint, the pay-as-you-go payment model is a constant revenue stream for the SaaS providers incentivizing them to focus on continuous improvement of the application. Table I summarizes the benefits of SaaS for both SaaS providers and SaaS users.

TABLE II: Comparison of traditional software and SaaS.

Features	Traditional software	Software as a Service
Cash flow	Fixed	Pay-as-you-go
Compliance	Tedious	Simpler
Control	User	Service Provider
Fees	One time	Subscription
Installation	Required	Not required
Internet	Not necessary	Required
Mobile synchronization	Costly requires additional software	Free/affordable may requires additional software
Ownership	User	Service Provider
Package location	On premises	No restriction
Responsibility	Owner responsibility	Third Party
Security	Organization	Service Provider
Scalability	Limited	Elastic
Testing	Intensive	Backup option
Upgrade	Complicated, expensive	Easy, free/cheap, automatic

III. Software-as-a-Service Architecture

The traditional software application deployment is a tedious, expensive, resource intensive and time consuming process. The user end points contain the user interface and also the program in hich the application logic is prescribed and the application must comply with application logic when it is up and running. It is cumbersome to maintain and upgrade software on user side. Besides, it is hard to scale up when the server reaches its limit. Traditional software and SaaS comparison is presented in Table II.

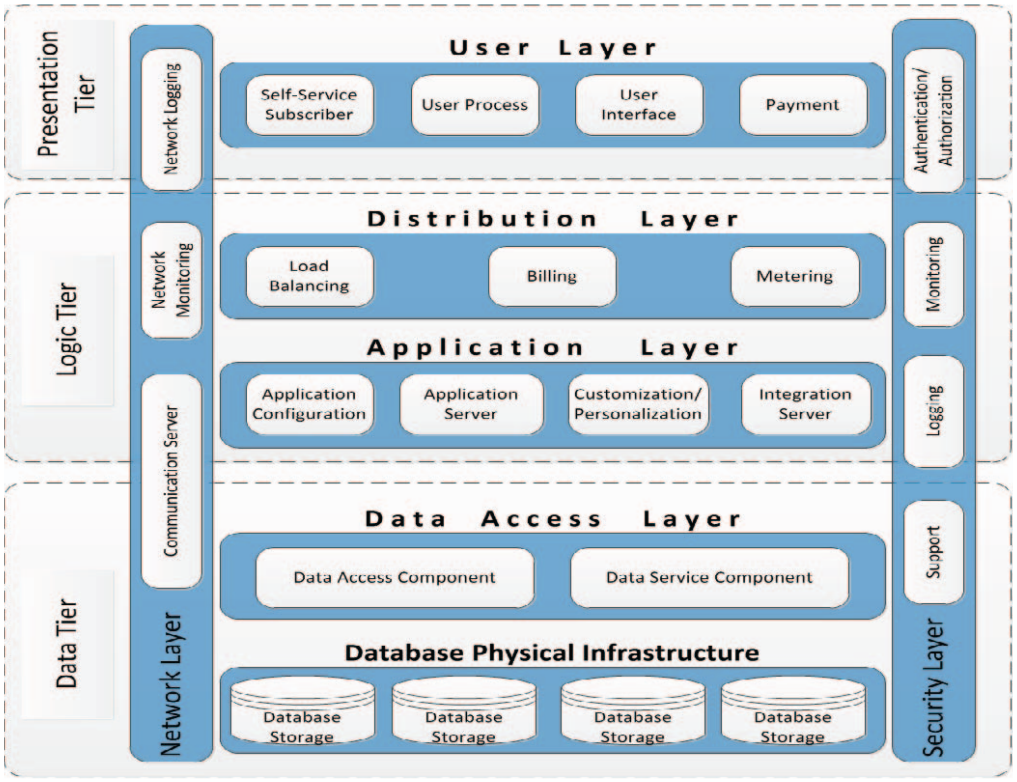

Fig. 2: Architecture of Software as a Service.

There are several shortcomings of traditional software that can be overcome by SaaS. Our vision of SaaS architecture is described in Figure 2. This architecture encompasses the complete SaaS platform.

User layer is composed of the web browser, user interface or thin client software which is easy and fast to download. Through these channels, along with the necessary SaaS multi-tenant API components, user can access the applications by using their devices. User process component helps to synchronize and orchestrate user interactions. This way the process flow and state management logic is not hard-coded in the user interface component, and the same basic user interaction patterns can be used by multiple user interfaces.

Besides, a webpage to guide the user on how the payment is processed should be included in this layer as well as a self-service subscription interface through which users can select additional services as needed at their will.

Distribution layer is between user layer and application layer. Once the user subscribes to the services, the application instances hosted on the application server should be easily distributed to the user in an effective way. However, it cannot be denied that one server group might be flooded with request in a certain period of time, or another server group still have large amount of resources available. In this situation the distribution layer handles load balancing between application servers. The distribution layer is also responsible for replicating and transferring application-associated data to another infrastructure to provide redundancy and high availability. Above all, one of the main responsibilities of this layer is to provide the user with the best level of service through optimizing resource utilization, throughput and response time.

Moreover, because SaaS is also a business model, proper financial and security solutions are needed in distribution layer as well. Thus, distribution layer includes metering and billing component. Depending on how the billing system is designed or what necessary parameters are needed to calculate the total fee, a metering solution should be in place to meter user usage and simultaneously pass the result to the billing system to generate a human-readable users invoice for payment.

Application layer is an application server farm which helps to handle http requests sent from the user and process these requests with business logic. These servers may have embedded components which are responsible for customization/personalization and application configuration. Customization/personalization component informs configuration options to users that can affect the functionalities and look-and-feel of the application. For example, application has users brand or logo or specific background settings. Application configuration component refers to configuration made by the service provider which cover configuration of security policies, output formatting and default authorization configurations.

Additionally, the server farm in some circumstances might consist of integration servers. Integration server is specially designed according to applications main functionality to make it much easier to integrate SaaS applications into an enterprise existing system. Apart from these, application layer also provides services such as clustering of application servers for redundancy.

Data Access layer abstracts the logic necessary to process

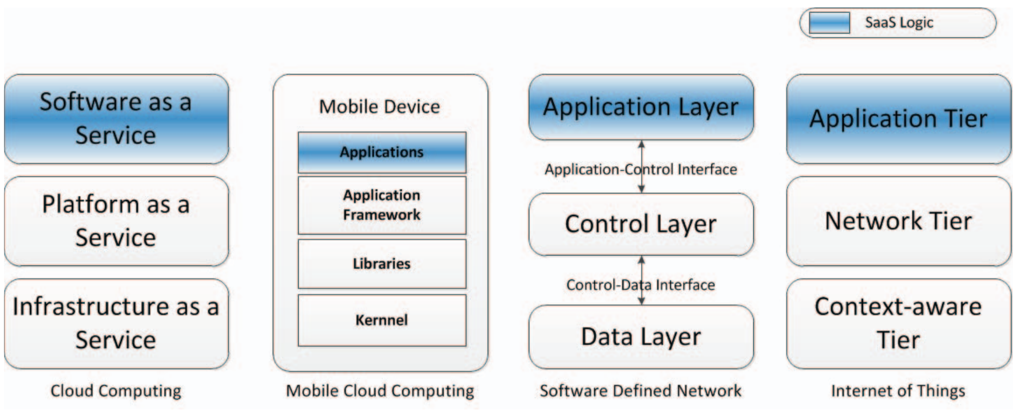

Fig. 3: Software as a Service in different environments.

of all requests for data, which is fulfilled by employing data access component. The data can be either stored in a well-structured and managed database model such as SQL database and object database or an unstructured file system. This layer should apply appropriate measures to enable the security and integrity of data accessed during the process of migration and transmission from database server to application server.

Data service component helps to call external methods by mapping the format of the data exposed by the method to the format of the application. Data service component also helps to manage the semantics of calling the methods.

One reason to separate data layer from application layer mainly is to separate business logic from data which provide a layer of abstraction and apportion the overhead of application servers. Another reason is to fulfill users demands of different hardware requirements according to negotiated SLA. For example, some might request more CPU and additional RAM, others might request for specific hard disk such as SSD.

Network layer is an intermediary channel which links all layers together. Data need to be transmitted over the SaaS providers internal network which is protected by relevant control and security policies, data need to be transmitted over the external network which is outside the boundary of the provider as well. The external network mostly refers to the Internet or to be more accurate, it refers to OSI model in which data flowthrough the network complying with the rules prescribed in TCP/IP and its associated protocols. But in some cases, the network data flowing through is a combination of internet and mobile network.

Network layer consists of three main components which are communication servers, network monitoring and network logging. The communication servers enable the communications under different protocols between mutual partners. Network monitoring is responsible for constantly monitoring network traffic to identify slow or failing components and to check the SaaS application server and associated servers status. By implementing a network monitoring system, SaaS provider can prevent the application server from being overloaded or be notified if one or several servers go down. Network logging is a component used for logging all incoming and outgoing network traffic that take place at the SaaS facilities. For example, it logs data about what port is opened by which remote host as well as the user context and protocol used by the communication. By implementing networking logging system, SaaS provider can either use them as the base of the network intrusion detection or track down the potentially malicious remote host and block their access to the server.

Security layer is indispensable through different layers of SaaS architecture. It provides authentication, authorization, log system and monitoring solutions to each layer of SaaS architecture. For example, in distribution layer, it is essential to employ appropriate effective security solutions such as monitoring scheme and intrusion detection and prevention system to handle the security-related issues. In application layer, authentication/authorization for protecting user privacy, monitoring and logging system for monitoring application performance and keeping record of user log history should be in place to enhance the security. Furthermore, authentication/authorization and monitoring mechanism are some of the main components which are used to provide security to data access layer as well.

Security layer also comprise user support mechanism, which contains various security enforcement policies or specialized requirement formulated by the user. For instance, transmitting data using https protocol instead of http; PKI should be in place for security concern; VPN channel should be established between communicating parties.

Security solutions could be carried out either in one specific layer or across several layers depending upon SaaS users and providers. They are important factors which must be taken into consideration to make SaaS architecture more mature, secure and reliable. The components in each layer are not a must-have fo r SaaS architectures in different contexts, but they are essential to run the SaaS business smoothly and effectively.

Aforementioned, software as a service is a way of delivering applications over the Internet as a service. SaaS delivers a single application through the browser to thousands of users using a multi-tenant architecture. A typical SaaS application is offered either directly by the provider or by an intermediary party called an aggregator, which bundles SaaS offerings from different providers and offers them as part of a unified application platform.

The emergence of SaaS as an effective software-delivery mechanism creates an opportunity for IT departments to change their focus from deploying and supporting applications to managing the services that those applications provide. To avail the benefits of SaaS features, cloud computing has considered SaaS as a fundamental component of its architecture. SaaS software delivery business model can be incorporated in Software Defined Networking (SDN) and Internet of T hings (IOT) in a similar manner as cloud computing is being utilized by mobile service provider through Mobile Cloud Computing (MCC), as shown in Figure 3.

IV. SOFTWARE-AS-A-SERVICE SECURITY CHALLENGES

The enormous growth of off-premise application services has changed the way the application services are delivered and brings presentable benefits and convenience to the software providers and users. However, as more and more individuals and enterprises deploy their applications in the Software-as-a-Service model, concerns begin to surface which doubt how reliable and safe SaaS is? There are issues that are same as forever and even more compelling and complicated due to the loophole of the SaaS deployment and inadequate security concern. In this section, SaaS security challenges are presented as 3 main groups which are data security, application security and deployment security.

A. *Data Security*

Database security deals with all various aspects of protecting the database content of its owners and users. Data security protection ranges from intentional unauthorized database uses to unintentional database accesses by unauthorized entities. Data in SaaS architecture reside in the database and transmit over Internet and mobile network which may be provided by third party and is outside the boundary of the enterprise. The client has to depend on the provider for proper security measures. The issue related to data security can be further classified into five subgroups which are data storage, data access control, data backup and recovery, data integrity and data transfer security.

1) Data Storage: Database storage is the container of the physical materialization of a database. The fundamental requirement is that SaaS provider keeps multiple users from seeing each others data. As multi-tenancy through virtualization is one of the major features of the SaaS platform, data of different users may reside in the same server, or in the different instances hosted in the same server. If one instance is compromised by the malicious attack such as SQL injection, there might be high risk that sensitive data in another instance hosted in the same server will be compromised as well. For example, in a virtualized world, hypervisor allows multiple operating system to run on a hardware at the same time, if a hacker is able to get control over the hypervisor, he can make changes to any instances and get access to all the sensitive data storage in that hardware.

If SaaS providers deploy their application on a public PaaS provider, the sensitive data might occasionally be stored together in the same physical infrastructure with the data of another unrelated SaaS application. To ensure data high availability, redundancy and backup, service provider might, additionally, duplicate the data and transfer it to another sites across countries. It is a high risk that sensitive information might be leaked to unexpected parties. Even worse, when the data in one data center encounters outage, the data in another location might travel through borders. However, in some countries the law prescribe that certain type of sensitive data is not allowed to cross the border [8] and in times of emergency the government has the right to invoke the log system to check incoming and outgoing traffic in past several weeks or months. This is regarded as a high potential threat to the confidentiality of users data.

Similarly, when users make up their minds to withdraw from the SaaS application, the data of users are supposed to be completely destroyed and cannot be recovered by any means. Nonetheless, SaaS environment complicates this issue because as mentioned above, the data of users might be replicated across world to maintain availability, redundancy or backup. There is no guarantee that all the files will be wiped out and no trace of recovery can be found since multiple copies of data are storage in multiple locations. It may happen that some backup disks or other media are left somewhere and forgotten which are not connected to the environment. Traditional methods such as physical destruction proves to be obsolete in the SaaS platform as data of different users may be occasionally stored in the same hardware.

To protect against the occasional leakage of data, SaaS provider adopts encryption to enhance the confidentiality of data in database. However, encryption techniques have their own set of challenges including key management, correct use of crypto algorithms and libraries. And new questions arise: which type of encryption is the most appropriate and at which layer should it be implemented? Where are these keys stored and who is responsible for these keys? How can encrypted data be retrieved if the key is lost? How is the situation handled when the key is compromised?

2) Data Access Control: In SaaS environment, the breach of security or privacy can come from the inside trusted partner where providers personnels deliberate or careless actions cause data loss or data leakage, even it can originates from the external malicious attackers who intend to exploit the weakness of the system and profit from it. In this case, provider must guarantee that access to data will only be limited to the authorized access by controlling who is allowed to access what type of information in the database. The information may refer to specific database objects such as record types, specific records or data structure; computations or operations over certain database objects such as query types; access path used to access the database objects such as specific indexes or other data structure. Usually, database access control is set by the authorized administrators through secure database management system console or interface. In some cases some companies may have their own access policies to adhere to the SaaS application provided by SaaS providers. Hence, SaaS providers must be capable of incorporating these specific policies into their own access control settings.

Furthermore, the employee or personnel information in a corporation cannot stay stationary all the time which might affect the data access control. Thus, when SaaS application is adopted by a corporation, how will SaaS provider confirm their staff directory is consistent with the directory in the corporations file? Is there any relatively secure notification mechanism implemented to keep the SaaS provider updated with the latest change of personnel information? These issues will pose threat on the companys sensitive information when they are overlooked by the administrators.

3) Data Backup and Recovery: Backup and recovery is one of the most important as pects of database administration. The occurrence of data corruption, hardware failures and data failures in a database is possible. SaaS users do not have any backup facilities at their level and are totally dependent on SaaS providers for data backup and recovery, it will be devastating for users to suffer data loss and in return revenue loss. It falls on the SaaS providers shoulders to make sure when any accident occurs, some countermeasures should be available to bring the database to the previous state and minimize the impact of the accident.

4) Data Integrity: Data Integrity [9] refers to maintaining and assuring the accuracy and consistency of data over its entire life-cycle. It is initially promoted to prevent the occurrence of data which does not accord to the formulated syntax in database and prevent the input/output of inappropriate message which could render invalid operation and error message in database. Data integrity can be compromised by human error when the data is entered into database or when the data is inappropriately migrated from one server to another. Software bugs and virus, hardware crash and natural disasters (earthquake and flood) can result in the compromise of data integrity as well.

Further, applications deployed in SaaS environment complicate the maintenance of data integrity. Multi-tenancy is one of the main features of SaaS which makes concurrency of transactions and consistency of database harder. In addition, the storage system of SaaS is mostly distributed storage system. If transactions occur in one database location, to ensure data integrity, data information in other databases need to be updated as well. The measures (checksums and error-correcting codes or algorithms) adopted to support integrity during transfer however bring along a considerable overhead to bandwidth resource. Therefore, in opinion of SaaS providers, enforcing integrity over a network require too much of the network's resources to be feasible. Even, methods such as two-phase commit protocol and two-phase locking protocol become inadequate in SaaS environment due to the congestion they might bring which potentially will slow down the whole system.

It is fundamental that SaaS providers should implement mechanisms to ensure data integrity and be able to tell what happened to a certain dataset and at what point.

5) Data Transfer Security: The transmission channel which established between SaaS providers site and users is not regarded always secure in SaaS environment. Data is transmitted in a large number of packets and flow through numerous third party infrastructure devices before reaching the destination. So, the data flow over the transmission process may be subject to the network threats such as DNS attack, MITM (Man-In-The-Middle) attack, IP spoofing, port scanning and sniffer attacks [10].

During the transmission procedure, data communicating parties always includes the sender part and the receiver part, so the identity verification between them becomes essential and must assure that transferred data should not be modified by the third party.

The security risks for the data in transit has increased due to eavesdropping [10]. It is usually done stealthily and hard to detect whether the networked has been eavesdropped by any hackers. Attackers can make use of eavesdropping to intercept the TCP session, then the attackers can search the sequence number of the ongoing communications to either forge a false segment with a malicious payload or impersonate as the sender IP address to do the further malicious act. Data security over the transitioning networks becomes quite important factor that is in pressing need to be addressed.

B. Application Security

Application security refers to the utilization of the software and hardware to guarantee that no harmful or unexpected action is executed, and the attacker is not able to get access to the administrator interface and make malicious changes. Application security issues can be induced at different levels of application design, developments, implementation and access. Therefore, it can be rendered by the design flaw inside the program itself or the insecure configuration of the user interface or web service-based APIs via which the user can access the application. In addition, rampant malware concealed in the SaaS application is another concern targeting at the user.

1) Software Design Flaws: Software design for SaaS is different from the conventional client-server environment in respect of architecture, user interface and various APIs. Multi-tenancy architecture of SaaS application is a barrier that an architect accustomed to designing isolated, single-tenant applications has to overcome. For example, when a user at one company accesses user information by using a CRM application service, the application instance that the user connects to may be accommodating users from dozens, or even hundreds, of other companies - all completely abstracted to any of the users. This requires an architecture that maximizes the sharing of resources across tenants, but that is still able to differentiate data belonging to different users.

APIs are the backbone of SaaS environment because of SaaS heterogeneity. For example, SaaS application running in the browser is supported by a lot of programming languages, such as the combination of HTML, CSS and JavaScript in the application presentation tier; or an engine using some dynamic Web content technology (Java, PHP, Python etc.) in the application middle tier (application logic tier). A lot of APIs must be employed to achieve interoperability, either APIs that already exist in the market or APIs need to be designed for special needs. However, poorly designed APIs might be subject to the malicious attack which software developer should take into account [11]. Customers follow the instructions presented by the providers on how to use such APIs, but also in some cases attackers can abuse these APIs and exploit their vulnerabilities.

2) User Interface and Web Technologies: SaaS providers usually present a set of interfaces and APIs that allow their customers to perform various operations. User can interact with application through user input fields. However, if these input fields do not validate user input, it could be exploited by attackers by launching injection attacks. Similarly, some

TABLE III: SaaS Security Affecting Different SaaS Layer.

Security Issues/Attacks	SaaS Layer Affected				
	User layer	Distribution Layer	Application layer	Database	Network Layer
Data Security					
Data Storage				✓	
Data Access Control		✓		✓	
Data Backup and Recovery				✓	
Data Integrity		✓		✓	
Data Transfer Security		✓		✓	✓
Application Security					
Software Design Flaws		✓	✓		
User Interface & Web Technologies	✓	✓		✓	✓
Web Services	✓			✓	✓
Malware	✓	✓	✓	✓	✓
SaaS Deployment Security					
Vulnerabilities of VM	✓	✓	✓	✓	
Vulnerabilites of Virtual Networks	✓	✓		✓	✓

SaaS applications enable customers to leave comments which are visible to all customers. Attackers can inject a malicious script in the comment field, which is then taken by application without validation and send to web browsers of other people. All the victims browsers viewing this site will execute this script which can hijack user sessions, deface websites or redirect user to another malicious site.

Another vulnerability in web application is the lack of unpredictable CSRF (cross-site request forgery) token for every sensitive operation (includes pages that link to sensitive operations and the code that actually carries out sensitive operations). Assume a user is just authenticated to a SaaS banking applications site, and then this user accidentally clicks into an website under attackers control, the attacker can forge malicious requests (such as transferring money to attackers account) and embed this attack in an image which appears to be interesting in the websites under his control. If this SaaS application does not have token against CSRF, customers become victims when they click this picture. There are some other threats such as, hidden field manipulation and backdoor and debug options.

HTTP is the protocol used for intercommunication between browser and application server. However, HTTP protocol is designed to be a stateless protocol. Thus, web application developers prefer to apply session management techniques or cookies to guarantee delivery. However, it is considered insecure because this type of mechanism is susceptible to session hijacking. If cookies implemented in the application website is not encrypted during the transmission or the transmission channel is not secured, it is most likely that the credentials and other sensitive information such as session ID stored in cookie can be read by a hacker to launch the further attack. For instance, attackers can masquerade as the trusted user and do anything the user is authorized to do on the network.

3) Web Services: Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other web-related standards. However, these web services and associated technologies also bring vulnerabilities to the application in SaaS environment. SOAP is a protocol specification for exchanging structured information in the implementation of web services. Since the vital sensitive information is contained in SOAP message, it should not be tampered with or changed unintentionally in the procedure of transfer. WSDL describes the web service such as how the service can be called?, what parameters it expects to receive? and what structure should returned data be? It is roughly equal to signature in programming language. But this information collected at the beginning can be used by the attacker to identify the entry and prepare for the attack of next stage.

Various attacks [12], [13] against web services includes over-size payload, coercive parsing, XML injection, XML wrapping attacks, WSDL scanning. Also, Denial of Service (DoS) attack or Distributed Denial of Service (DDoS) attack is representative in web services-related attacks. For example, sending numerous HTTP request to the server at a short period of time; sending a malicious query to force a backend NoSQL database into an infinite loop; sending a "billion laugh attack" exponential entity expansion to suspend an XML configuration file parser. The consequence of those can bring difculties to get assurance that the application will be available when needed.

4) Malware: Malware can be defined as a malicious software or code which is designed to use the device and information without its owners consent. If user download application software without the appropriate security check, it is likely that their personal and sensitive information will be at risk. The channels through which the device can be affected, the ways in which the malicious software is installed on the device and the incentive for doing this are multifarious. The classification of the infection channels are Bluetooth, SMS/MMS, Internet Connectivity, Portable Memory and Connection to other Devices [14]. Malware can be grouped into five categories according their features which are virus, worm, Trojan, rootkits and botnet.

C. Software-as-a-Service Deployment Security

Virtualization refers to the act of creating different instances on hardware and on each instances a guest OS is installed. Nowadays SaaS is largely built on virtualization technology. The provider runs a single instance that serves every user, with configurable meta data providing a unique user experience and feature set for each one. Certain policies are adopted to ensure that each user's data is kept separate from that of other users and it is transparent to end users that the application instance is being shared among multiple tenants. This approach eliminates the need to provide server space for as many instances as the provider has users. Moreover, it enables SaaS platform to dynamically allocate the virtual machine to the users and therefore resource utilization is optimized and server consolidation is fulfilled.

Virtualization technologies are used by SaaS to provide multi-tenancy. However, the vulnerability and weakness of virtualization [15] affects the SaaS security. Thus, it is vital to consider the virtualization-related security issues in SaaS environment. The SaaS deployment security can be mainly classified into two aspects: vulnerabilities of Virtual Machine (VM) and vulnerabilities of Virtual Network (VN).

1) Vulnerabilities of Virtual Machine: A large amount of VMs with different OS can be hosted on the hypervisor or virtual machine monitor (VMM). Attacks against VM or VMM [16], [17], [18] give the attacker channel to confidential data stored in many different virtual servers which hosted in the same hardware. The attacker can gain access to the hypervisor [17] through a running instance hosted on the hypervisor.

The attacker can monitor another VMs resources and CPU utilization and intercept the incoming and outgoing data flow, even worse, can shut down any instances. The well-known hypervisor VM escape [4] attacks are BluePill [19], SubVirt [20] and Direct Kernel Structure Manipulation (DKSM) [21]. Each VM is established by operation through management console which is usually a graphic user interface. The console is directly linked with hypervisor. In some cases, because the hypervisor to some extent is connected to the outside network, so it is likely that the functionalities of this console could be exploited by the hacker.

In virtualization environment, VMs share the hardware resources such as CPU, RAM, hard disk and network bandwidth. If one hosted server (instance) takes up the majority of the resources, the request from another client might be declined because of the shortage of the available resources. This can occur if one instance comes across the DoS or DDoS attack from the third party.

2) Vulnerabilities in Virtual Networks: Nowadays, a lot of hypervisors allow establishing the virtual network for VMs to connect to the outside physical network. Xen presents three ways to set up virtual networking which are bridged networking, route networking and NAT networking. However, virtualized network in SaaS environment might lead to the degraded security compared with traditional physical network. [22] briefly discusses about the vulnerabilities existing in bridged and route networking. In the bridged networking, the bridge acts as a virtual hub to link all VMs together (cluster), so if one VM is connected to this cluster by bridged adapter, it enables the attacker to sniff the packet to all the other VMs in this same network. In the route networking, the attacker can forge a fake IP address the same to other VM in this network, in this way attacker impersonate to redirect other VMs traffic.

Table III presents a connection between the type of SaaS security issues/attacks and the SaaS layer affected by the issues/attacks. The distribution and database layers are the clear layers of attack. This reflects the main characteristics of the SaaS that multi-tenancy, pay-as-you-go and distributed storage are attacker focus.

User layer is mainly affected by application security issues along with the vulnerabilities of VM and VN. The two components user process and user in terface (mentioned in section II) resides in user layer are targeted objects. Application layer could be affected by application security issues and the vulnerabilities of virtual machine. Database is vulnerable to almost all the SaaS security issues. This is because the breach of security mostly aims at the compromise of user sensitive or non-sensitive information directly or indirectly, and user associated information are stored in the database. Network layer can be attacked by malware, data transfer vulnerabilities, hacking web technologies and virtual network vulnerabilities. Distribution tier is what SaaS application deployment model distinct it from traditional application deployment model. The table shows this layer is susceptible to data security, application security and SaaS deployment security. This section analysis point out that even though SaaS provides better advantages than traditional software, it has to tackle more security challenges than traditional software.

V. Conclusion

Software service has turned around the way people work on business and other type of daily affairs to the extent which is regarded as more and more inseparable in our modern life. In this article, the architecture of Software as a service is presented in six different layers and the function of each is briefly discussed. SaaS promises scalability, integration lower costs, reduces time to market, easy upgrades and use to perform proof of concepts. In order to achieve this goal, a number of outstanding challenges must be resolved. We present a discussion of a number of challenges in the area of SaaS data security, application security and deployment security.

Our future research vision will focus on two directions to provide confidentiality, integrity and secure data management for SaaS. First, extending authentication techniques. Second, integrating authentication solution with secure resource management on virtual environment to get more controlled environment. Finally, a prototype will be implemented to demonstrate the system feasibility and performance.

References

[1] C. Pettey and R. v. d. Meulen, "Gartner says worldwide software-as-a-service revenue to reach 14.5 billion dollars in 2012," Gartner, Inc, Tech. Rep., March 2012.

[2] P. Mell and T. Grance, "The nist definition of cloud computing," 2011.

[3] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *Wireless communications and mobile computing*, vol. 13, no. 18, pp. 1587–1611, 2013.

[4] B. Grobauer, T. Walloschek, and E. Stocker, "Understanding cloud computing vulnerabilities," *Security & privacy, IEEE*, vol. 9, no. 2, pp. 50–57, 2011.

[5] R. Rai, G. Sahoo, and S. Mehfuz, "Securing software as a service model of cloud computing: Issues and solutions," *arXiv preprint arXiv:1309.2426*, 2013.

[6] S. Subashini and V. Kavitha, "A survey on security issues in service delivery models of cloud computing," *Journal of Network and Computer Applications*, vol. 34, no. 1, pp. 1–11, 2011.

[7] S. He, L. Guo, M. Ghanem, and Y. Guo, "Improving resource utilisation in the cloud environment using multivariate probabilistic models," in *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*. IEEE, 2012, pp. 574–581.

[8] S. Pearson and A. Benameur, "Privacy, security and trust issues arising from cloud computing," in *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*. IEEE, 2010, pp. 693–702.

[9] J. E. Boritz, "Is practitioners' views on core concepts of information integrity," *International Journal of Accounting Information Systems*, vol. 6, no. 4, pp. 260–279, 2005.

[10] R. Bhadauria, R. Chaki, N. Chaki, and S. Sanyal, "A survey on security issues in cloud computing," *arXiv preprint arXiv:1109.5388*, 2011.

[11] M. Jensen, J. Schwenk, N. Gruschka, and L. L. Iacono, "On technical security issues in cloud computing," in *Cloud Computing, 2009. CLOUD'09. IEEE International Conference on*. IEEE, 2009, pp. 109–116.

[12] M. Jensen, N. Gruschka, and R. Herkenhöner, "A survey of attacks on web services," *Computer Science-Research and Development*, vol. 24, no. 4, pp. 185–197, 2009.

[13] M. McIntosh and P. Austel, "Xml signature element wrapping attacks and countermeasures," in *Proceedings of the 2005 workshop on Secure web services*. ACM, 2005, pp. 20–27.

[14] D. Gligoroski, I. P. Stephanow, and S. Fraunnhofer, "Security as a service in cloud for smartphones," 2011.

[15] D. A. Fernandes, L. F. Soares, J. V. Gomes, M. M. Freire, and P. R. Inácio, "Security issues in cloud environments: a survey," *International Journal of Information Security*, vol. 13, no. 2, pp. 113–170, 2014.

[16] M. Schwartz, "New virtualization vulnerability allows escape to hypervisor attacks," *Information Week Security. Luettu*, vol. 4, p. 2013, 2012.

[17] J. Granneman, "Virtualization vulnerabilities and virtualization security threats," July 2012. [Online]. Available: http://searchcloudsecurity.techtarget.com/tip/Virtualization-vulnerabilities-and-virtualization-security-threats

[18] M. Pearce, S. Zeadally, and R. Hunt, "Virtualization: Issues, security threats, and solutions," *ACM Computing Surveys (CSUR)*, vol. 45, no. 2, p. 17, 2013.

[19] J. Rutkowska, "Subverting vistatm kernel for fun and profit," *Black Hat Briefings*, 2006.

[20] S. T. King and P. M. Chen, "Subvirt: Implementing malware with virtual machines," in *Security and Privacy, 2006 IEEE Symposium on*. IEEE, 2006, pp. 14–pp.

[21] S. Bahram, X. Jiang, Z. Wang, M. Grace, J. Li, D. Srinivasan, J. Rhee, and D. Xu, "Dksm: Subverting virtual machine introspection for fun and profit," in *Reliable Distributed Systems, 2010 29th IEEE Symposium on*. IEEE, 2010, pp. 82–91.

[22] H. Wu, Y. Ding, C. Winer, and L. Yao, "Network security for virtual machine in cloud computing," in *Computer Sciences and Convergence Information Technology (ICCIT), 2010 5th International Conference on*. IEEE, 2010, pp. 18–21.