Contents

Software as a Service(SaaS) using Secure Shell(SSH) in Cloud Computing

Abstract :

Today memory of a computer is a major concern. Installing and running a program requires huge amount of memory, and also it slows down the computer. Installing more software requires more memory. We rarely require some software but they occupy lot of memory in computer preventing us from holding useful data for use. This project aims to overcome these issues using Cloud Computing technology. Software as a Service(SaaS) provides a platform for people to use more software at lesser cost of memory. It also assures the reliability of service. The high end server provides a fast way to access the data in a minimal amount of time. You need not install the software in your computer. You can hire the services of server to provide what software you need through X11 forwarding. The X Window System (X11**,** or shortened to simply X) is a windowing system for bitmap displays, common on UNIX-like computer operating systems. The server runs the software which forwards the X11 packets to the client using a secure shell connection (SSH).SSH, also known as Secure Socket Shell, is a network protocol that provides administrators with a secure way to access a remote computer. This way ensures security of the data that is being passed from the client to the server and vice versa. The user will be charged for the quantity of data he/she has consumed. A powerful server with many software could serve many clients at the same time. User starts a session and requests the software from server. Server forwards the X11 packets to the client. User can close the session at any time he/she wishes. Session details of user will be provided once session is closed. Total cost for usage of service will be provided to the user on monthly basis. So by this way we don't waste any memory in computer. We can use it effectively to store useful data. Also the speed of computer increases because processes are running in the server.

Introduction to Project :

The three important things that need to be defined before diving into the details of the project are as follows :

- **Secure Shell** is a cryptographic network protocol for operating network services securely over an unsecured network.

- **Software as a Service** (or SaaS) is a way of delivering applications over the Internet as a service. Instead of installing and maintaining software, you simply access it via the Internet, freeing yourself from complex software and hardware management.

- The **X Window System** (X11, or shortened to simply X) is a windowing system for bitmap displays, common on UNIX-like computer operating systems.

The core idea of the project lies in integrating above three core technologies to provide services that are secure and robust in nature. Secure Shell provides itself the best security way for accessing applications across networks. In X windows system each pixel on the display is transferred from the remote system to user so that it appears to user that application is running in his/her computer. Software as a Service is the ability to provide software to the users on demand whenever they need it. We can easily integrate Software as a Service with Secure Shell by transferring each pixel in a Secure Shell channel. Thus the pixels are saved from being handed to others. By this way user can safely access applications without worrying about the security aspects of the data they are using. Also applications can be handled to users at any moment they require it just through few mouse clicks. With information that is provided to the users, they can calculate the amount of time they are using the applications. This application is mainly targeted for UNIX and LINUX systems. With the advent of Microsoft's windows 10 Anniversary update it is now possible to communicate from a Windows system to Linux system, using Bash shell option that is being provided. So this application will make a huge impact on systems where the memory is still small. Although this application is targeted for computers with limited memory, it can also be used with any computers. Using this application the user will find that they can operate more than ten applications without consuming a lot of memory from RAM. Thus this application is useful for everyone who uses computer for day-to-day activities.

Literature Survey :

The following points discuss how SaaS is implemented now-a-days :

- ✓ Software as a Service (SaaS) is a software deployment model where applications are remotely hosted by the application or service provider and made available to customers on demand, over the Internet. Enterprises can take advantage of the SaaS model to reduce the IT costs associated with traditional on-premise applications like hardware, patch management, upgrades, etc. On demand licensing can help customers adopt the "pay-as-you-go/grow" model to reduce their up-front expenses for IT purchases.
- ✓ In a SaaS deployment model, sensitive data is obtained from the enterprises, processed by the SaaS application and stored at the SaaS vendor end. All data flow over the network needs to be secured in order to prevent leakage of sensitive information. This involves the use of strong network traffic encryption techniques such as Secure Socket Layer [SSL] and the Transport Layer Security [TLS] for security.
- ✓ SaaS lets software vendors control and limit use, prohibits copies and distribution, and facilitates the control of all derivative versions of their software. SaaS centralized control often allows the vendor to establish an ongoing revenue stream with multiple businesses [tenants] and users. The tenants are provided a protected sandbox view of the application that is isolated from other tenants. Each tenant can tune the metadata of the application to provide a customized look and feel for its users.

- ✓ A web-based software system delivered through a **browser,** just like web pages and web games do. This is, in fact, the best model in the opinions of most due to the compatibility that this guarantees as long as the browser is capable of sophisticated stuff.

The working of SSL is described as follows :

SSL (superseded by the more modern Transport Layer Security) is a general protocol that can be implemented on top of other transport-layer protocols such as HTTP and FTP. As

such, you can use it to transfer files or view web pages securely, and there are many other applications. The best-known application for SSL is encrypting a form submission so you can send your credit card details to a retailer without fear of an eavesdropper on your network viewing your credit card number. SSL communication is not necessarily authenticated (you can encrypt your communication with a website without giving any username/password).

Thus existing model of SaaS can be summarized as follows :

1) All applications are delivered through browsers like Google Docs, Google Drive etc.,
2) Sensitive information is transferred through SSL(Secure Socket Layer).
3) When demand grows user gets allocated extra servers and storage units.
4) User is provided a complete abstraction of how application is deployed.
5) The enterprise data is stored outside the enterprise boundary, at the SaaS vendor end.
6) Consequently, the SaaS vendor must adopt additional security checks to ensure data security and prevent breaches due to security vulnerabilities in the application or through malicious employees.
7) The use of strong encryption schemes to protect the backup data is recommended to prevent accidental leakage of sensitive information.

Example of existing applications :

In case of Amazon Web Services [AWS], the network layer provides significant protection against traditional network security issues, such as MITM attacks, IP spoofing, port scanning, packet sniffing, etc. For maximum security, Amazon S3 is accessible via SSL encrypted endpoints. The encrypted endpoints are accessible from both the Internet and from within Amazon EC2, ensuring that data is transferred securely both within AWS and to and from sources outside of AWS.

Drawbacks of Above Model :

- ☒ The user will not be provided application as such that is all the applications are provided through browser.
- ☒ The real application is not provided. Examples include Microsoft- word cannot be provided by the above model.
- ☒ All information cannot be passed through SSL because it will require huge amount of time to transfer all information.
- ☒ User cannot select the application which he/she wants to work. There wll be a predefined set of applications.
- ☒ User will be allowed to select an application from the available and that will also be provided through browser.
- ☒ Needs a well supported browser to support all applications.
- ☒ Need to update browser frequently if a change is made in applications.

To overcome the above draws back, we discuss a model that is probably best suited for SaaS applications.

Proposed Work :

We can use Secure Shell in order to provide SSL level security features and continuously use that channel to send and receive commands. Not only we can send and receive commands we can use it for file uploads, file downloads and using it as medium to send any form of digital data. We have to understand the key differences between SSL and SSH. The following picture gives us a better understanding of how SSH is different from SSL.

	SSL	SSH
Abbreviation	Secure Socket Layer	Secure Shell
Port	443	22
Application	For Encrypting Communication between Browser and Server	For Encrypting Communication between Two Computer
Adopted by Industry	Widely used by E-commerce, Banking, Social Media, Government, Healthcare, etc. industries.	Widely Adopted by Networking Industry.
Authentication	Via Public-Key/Private-Key Pair	Via 1) Public-Key/Private-Key Pair Or 2) User-Id/Password Pair.

Figure 1: Difference Between SSH and SSL

We can very well understand how SSH is different from SSL. Thus SSH can be used to send and retrieve all formats of digital data with high security and with varying options.

The task of delivering applications over the internet using SSH is explained as follows.

- ✓ Client will be provided username and password once he registers for the service.

- ✓ Once user starts the application SSH session is established with the server.
- ✓ Client verifies the host address. After successful validation of the host address, the user will be asked for username and password.
- ✓ Once verification is done user will be displayed a list of applications that is available on the host system.
- ✓ If it is first time a file name total will be created in order to monitor the amount of time client has used the applications.
- ✓ User will be asked to open an application from the list of applications.
- ✓ A new SSH channel is set up to transfer applications between the client and server.
- ✓ The application that is requested will be started as a process in server.
- ✓ The display pixels are converted and transferred over the network.
- ✓ Client side receives the pixel one by one and forms the application screen in the client computer.
- ✓ Once user closes the application the SSH(Secure Shell) channel will be closed.
- ✓ The amount of time from starting time to end time of application is calculated using a time snapshot.
- ✓ The time will be reflected in total file.
- ✓ This process continues till client requires application from the server.
- ✓ All the details about how much time he/she has used the applications can be viewed.
- ✓ The application can be closed when user finishes his work.

Thus the above proposed work can be done if we try to integrate Secure Shell(SSH) with X windows system.

Usecase Diagram :

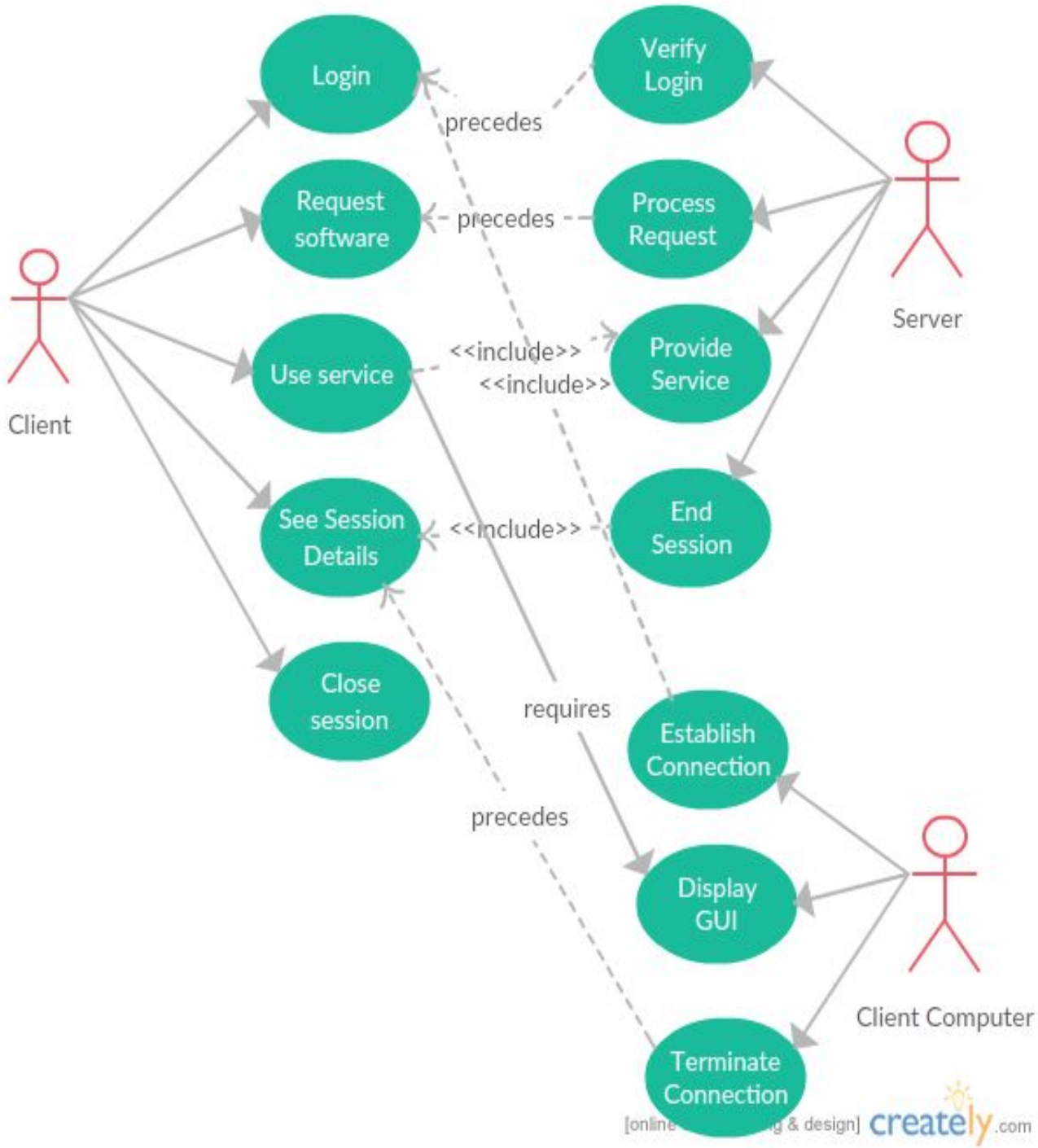

Fig 3: Usecase Diagram

Class Diagram :

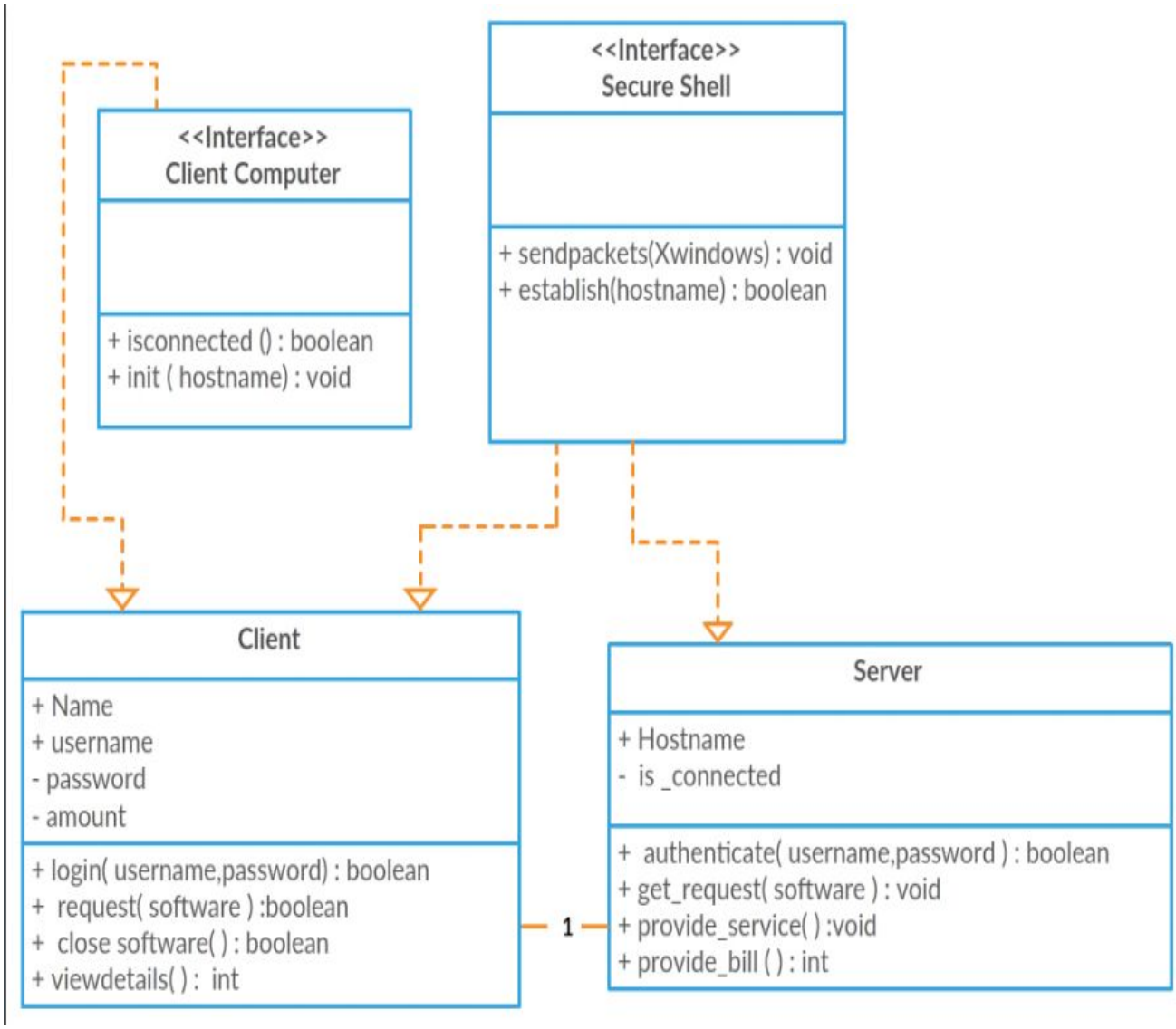

Fig 4: Class Diagram

Sequence diagram :

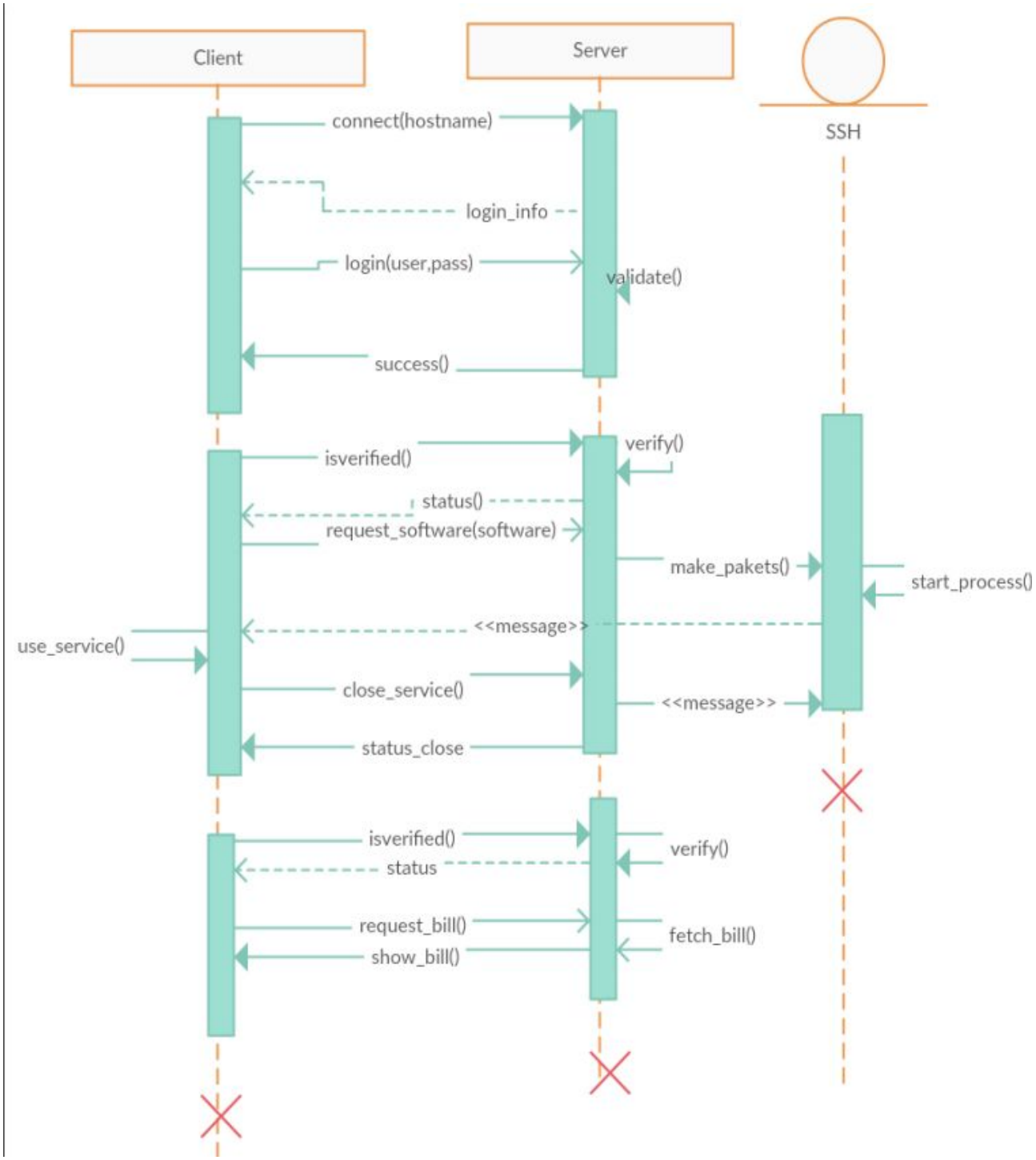

Fig 5 : Sequence Diagram

Implementation Details :

Algorithm :

Step 1 : Ask for Host address from user.

Step 2 : Verify Host address. If successful ask for username and password.

Step 3 : Validate username and password. If not correct go to step 2.

Step 3 : Provide users a list of applications by using the following command

```
'apt list –installed'
```

Step 4 : Using Jsch object create a new session and give port number as 22. The connection is established using createSession() function.

Step 5 : Request X11 forwarding through the object.

Step 6 : Now request the application in server.

Step 7 : Simply receive the application in client side and display it.

Step 8 : Once user closes the application in client side, close the Jsch session using closeSession() function.

Step 9 : Update the time used in total.txt file in server.

Step 10 : Go to step 3 if user needs to open another application.

Step 11 : Stop.

Tools used :

Two important tools used in project are

1. Secure Shell
2. X windows system
3.

Secure Shell :

SSH provides a secure channel over an unsecured network in a client-server architecture, connecting an SSH client application with an SSH server. Common applications include remote command-line login and remote command execution, but any network service can be secured with SSH. The protocol specification distinguishes between two major versions, referred to as SSH-1 and SSH-2.

The most visible application of the protocol is for access to shell accounts on Unix-like operating systems, but it sees some limited use on Windows as well. In 2015, Microsoft announced that they would include native support for SSH in a future release.

SSH was designed as a replacement for Telnet and for unsecured remote shell protocols such as the Berkeley rlogin, rsh, and rexec protocols. Those protocols send information, notably passwords, in plaintext, rendering them susceptible to interception and disclosure using packet analysis.

The encryption used by SSH is intended to provide confidentiality and integrity of data over an unsecured network, such as the Internet, although files leaked by Edward Snowden indicate that the National Security Agency can sometimes decrypt SSH, allowing them to read the contents of SSH sessions.

SSH uses public-key cryptography to authenticate the remote computer and allow it to authenticate the user, if necessary. There are several ways to use SSH; one is to use automatically generated public-private key pairs to simply encrypt a network connection, and then use password authentication to log on.

Another is to use a manually generated public-private key pair to perform the authentication, allowing users or programs to log in without having to specify a password. In this

scenario, anyone can produce a matching pair of different keys (public and private). The public key is placed on all computers that must allow access to the owner of the matching private key (the owner keeps the private key secret). SSH only verifies whether the same person offering the public key also owns the matching private key. In all versions of SSH it is important to verify unknown public keys, i.e. associate the public keys with identities, before accepting them as valid. Accepting an attacker's public key without validation will authorize an unauthorized attacker as a valid user.

SSH is typically used to log in to a remote machine and execute commands, but it also supports tunneling, forwarding TCP ports and X11 connections; An SSH client program is typically used for establishing connections to an SSH daemon accepting remote connections. Both are commonly present on most modern operating systems, including macOS, most distributions of Linux, OpenBSD, FreeBSD, NetBSD, Solaris and OpenVMS. Notably, Windows is one of the few modern desktop/server OSs that does not include SSH by default.

Proprietary, freeware and open source (e.g. PuTTY, and the version of OpenSSH which is part of Cygwin) versions of various levels of complexity and completeness exist. Native Linux file managers (e.g. Konqueror) can use the FISH protocol to provide a split-pane GUI with drag-and-drop.

The open source Windows program WinSCP provides similar file management (synchronization, copy, remote delete) capability using PuTTY as a back-end. Both WinSCP and PuTTY are available packaged to run directly off a USB drive, without requiring installation on the client machine. Setting up an SSH server in Windows typically involves installation (e.g. via installing Cygwin).

SSH is important in cloud computing to solve connectivity problems, avoiding the security issues of exposing a cloud-based virtual machine directly on the Internet. An SSH tunnel can provide a secure path over the Internet, through a firewall to a virtual machine.

X Window System :

The **X Window System** (**X11**, or shortened to simply **X**) is a windowing system for bitmap displays, common on UNIX-like computer operating systems.

X provides the basic framework for a GUI environment: drawing and moving windows on the display device and interacting with a mouse and keyboard. X does not mandate the user interface – this is handled by individual programs. As such, the visual styling of X-based environments varies greatly; different programs may present radically different interfaces.

X is an architecture-independent system for remote graphical user interfaces and input device capabilities. Each person using a networked terminal has the ability to interact with the display with any type of user input device.

In its standard distribution it is a complete, albeit simple, display and interface solution which delivers a standard toolkit and protocol stack for building graphical user interfaces on most Unix-like operating systems and OpenVMS, and has been ported to many other contemporary general purpose operating systems.

X uses a client–server model: an X server communicates with various *client* programs. The server accepts requests for graphical output (windows) and sends back user input (from keyboard, mouse, or touchscreen). The server may function as:

- an application displaying to a window of another display system
- a system program controlling the video output of a PC
- a dedicated piece of hardware

This client–server terminology – the user's terminal being the server and the applications being the clients – often confuses new X users, because the terms appear reversed. But X takes the perspective of the application, rather than that of the end-user: X provides display and I/O services to applications, so it is a server; applications use these services, thus they are clients.

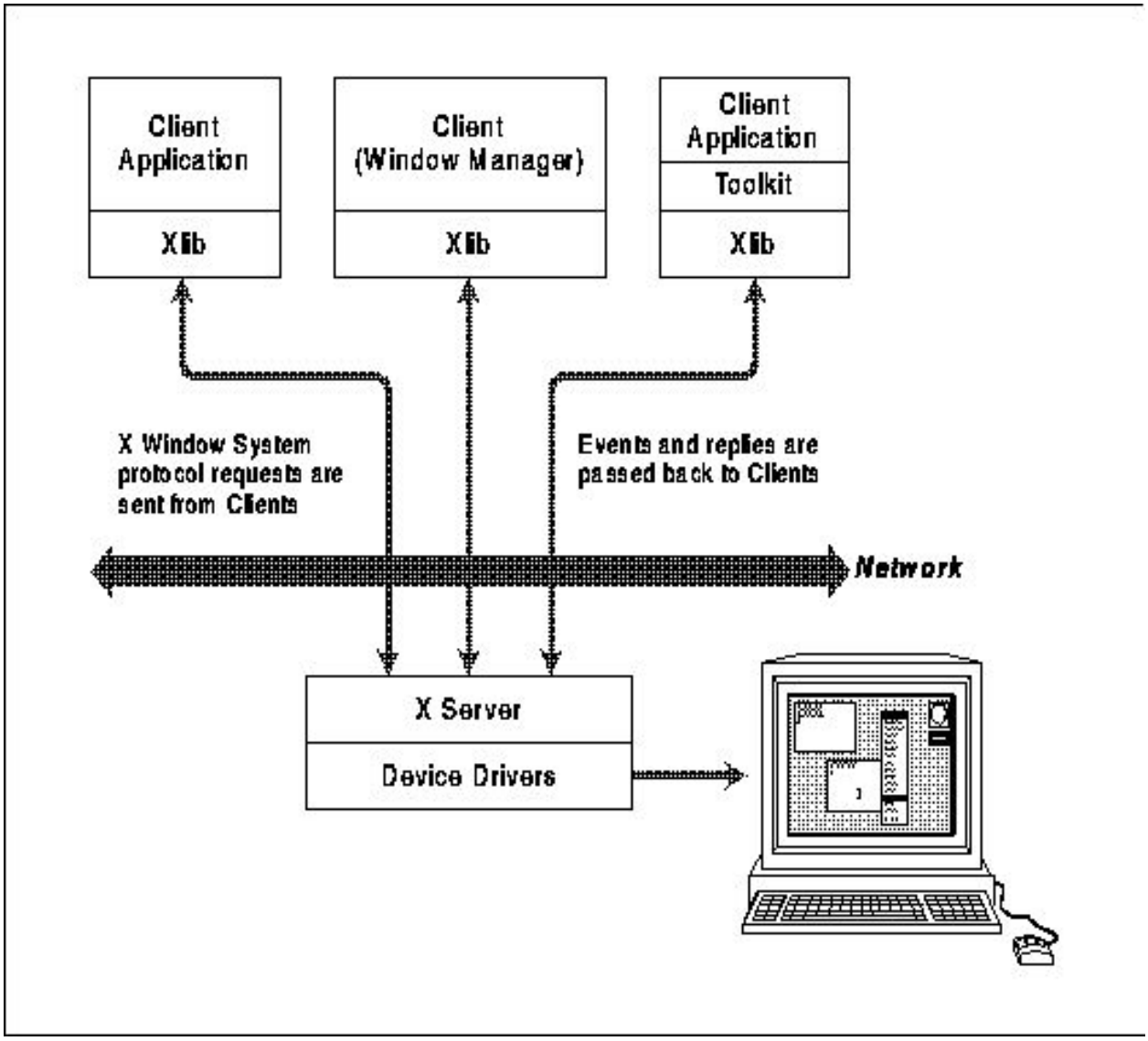

Figure 2 : X server architecture

Results and Inferences :

Working Screenshots :

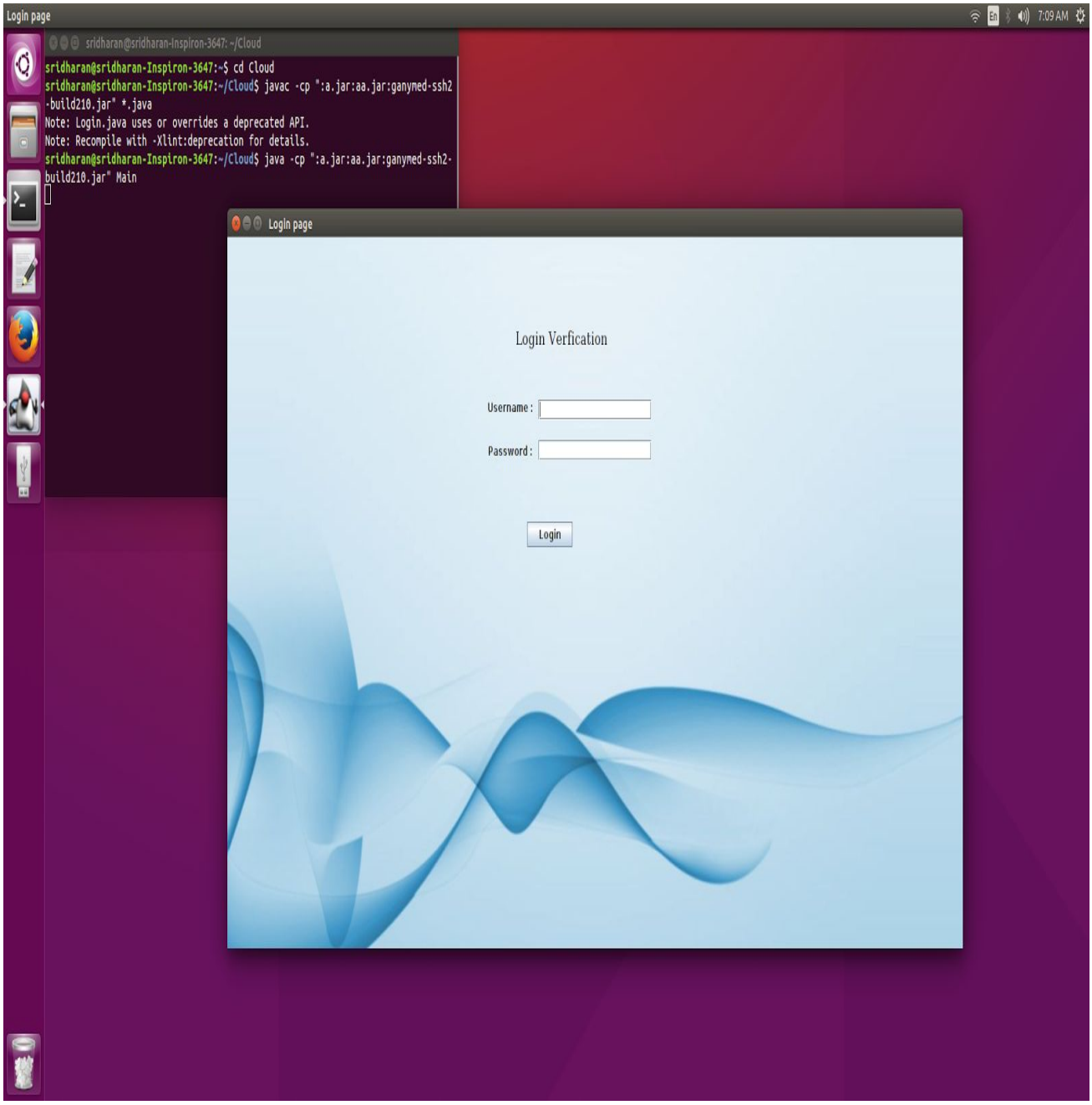

Fig 6 : Login Screen

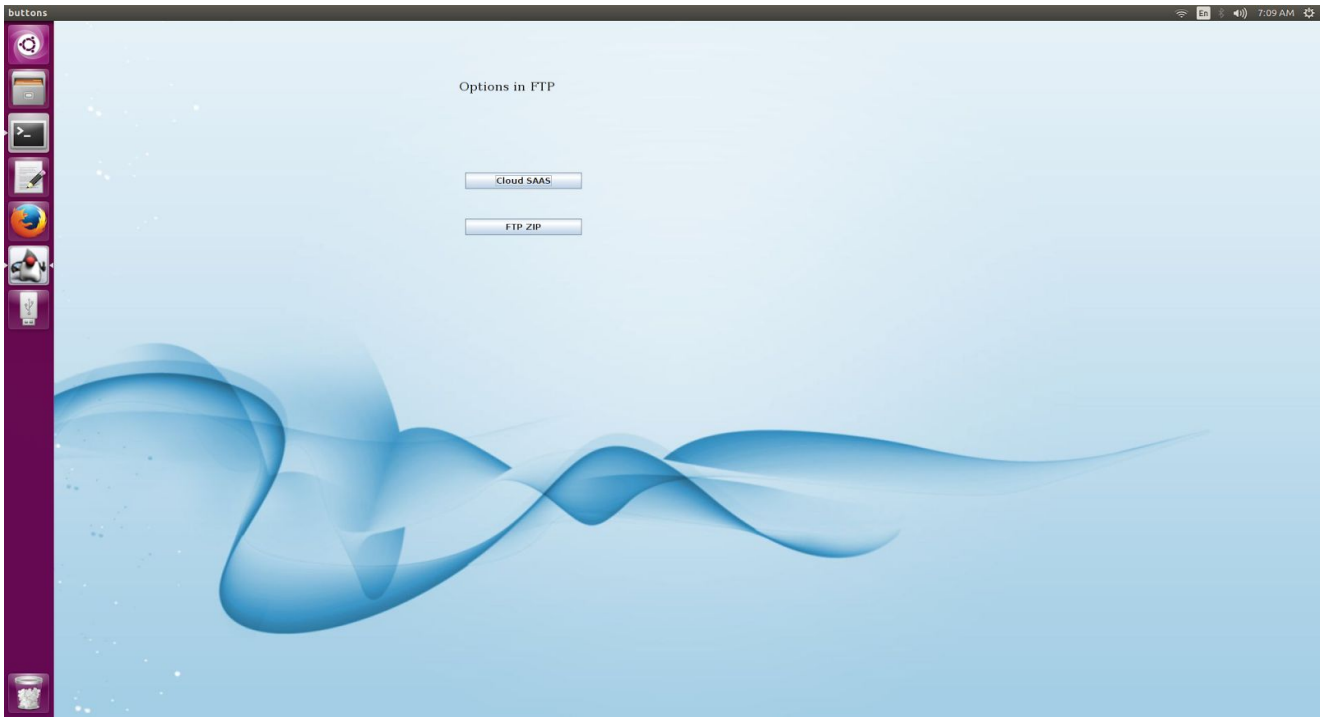

Fig 7 : Application Selection

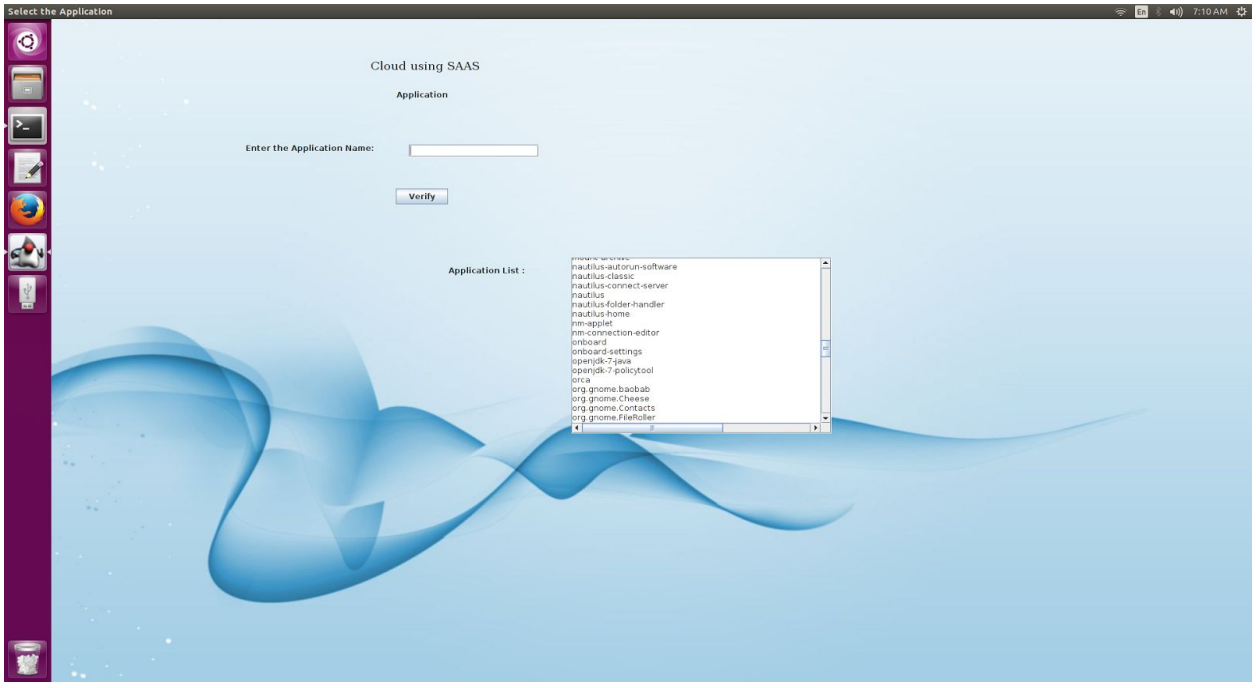

Fig 8 : List of installed applications

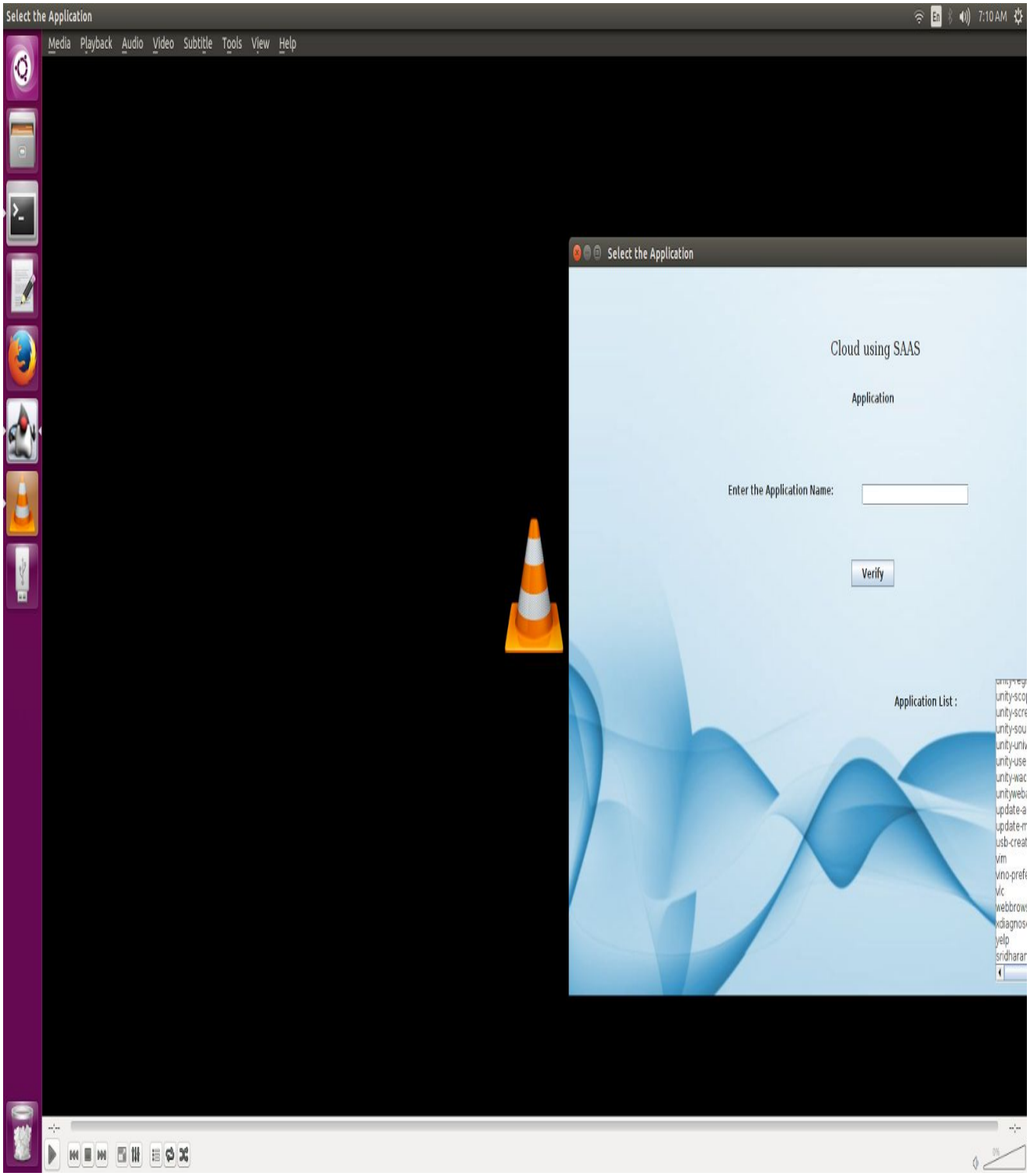

Fig 9 :Application live in client side

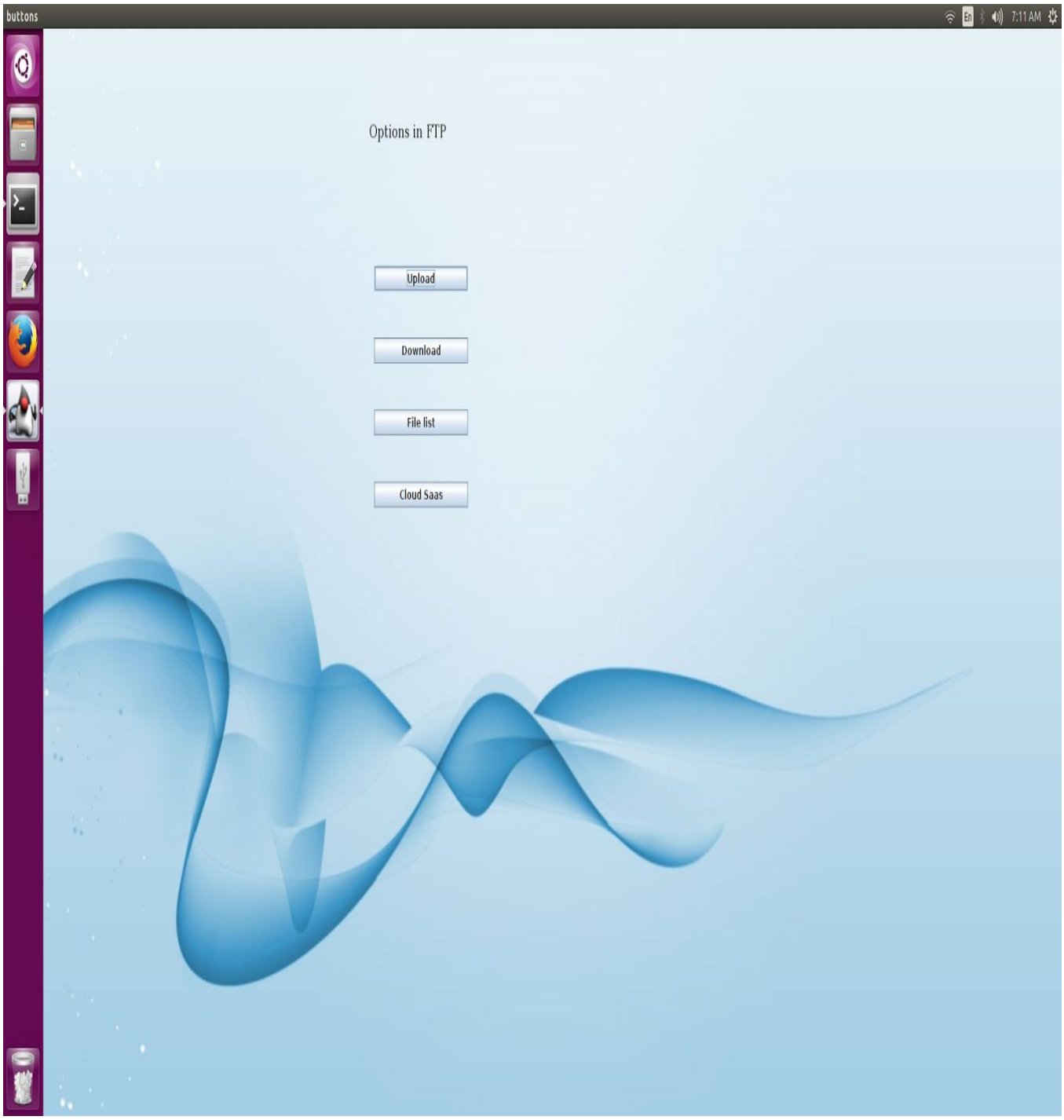

Fig 10 : List of options available

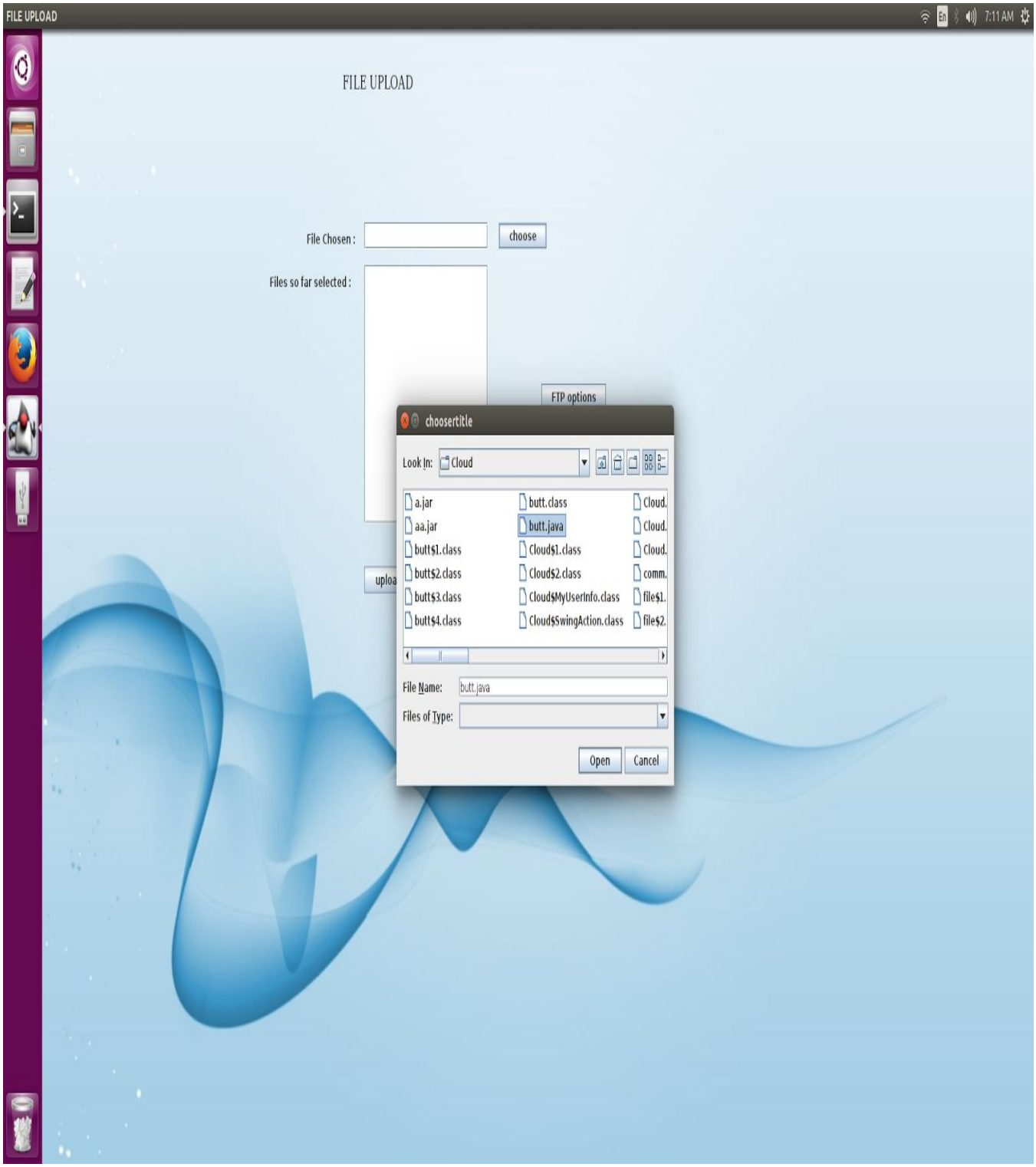

Fig 11 : File Upload window

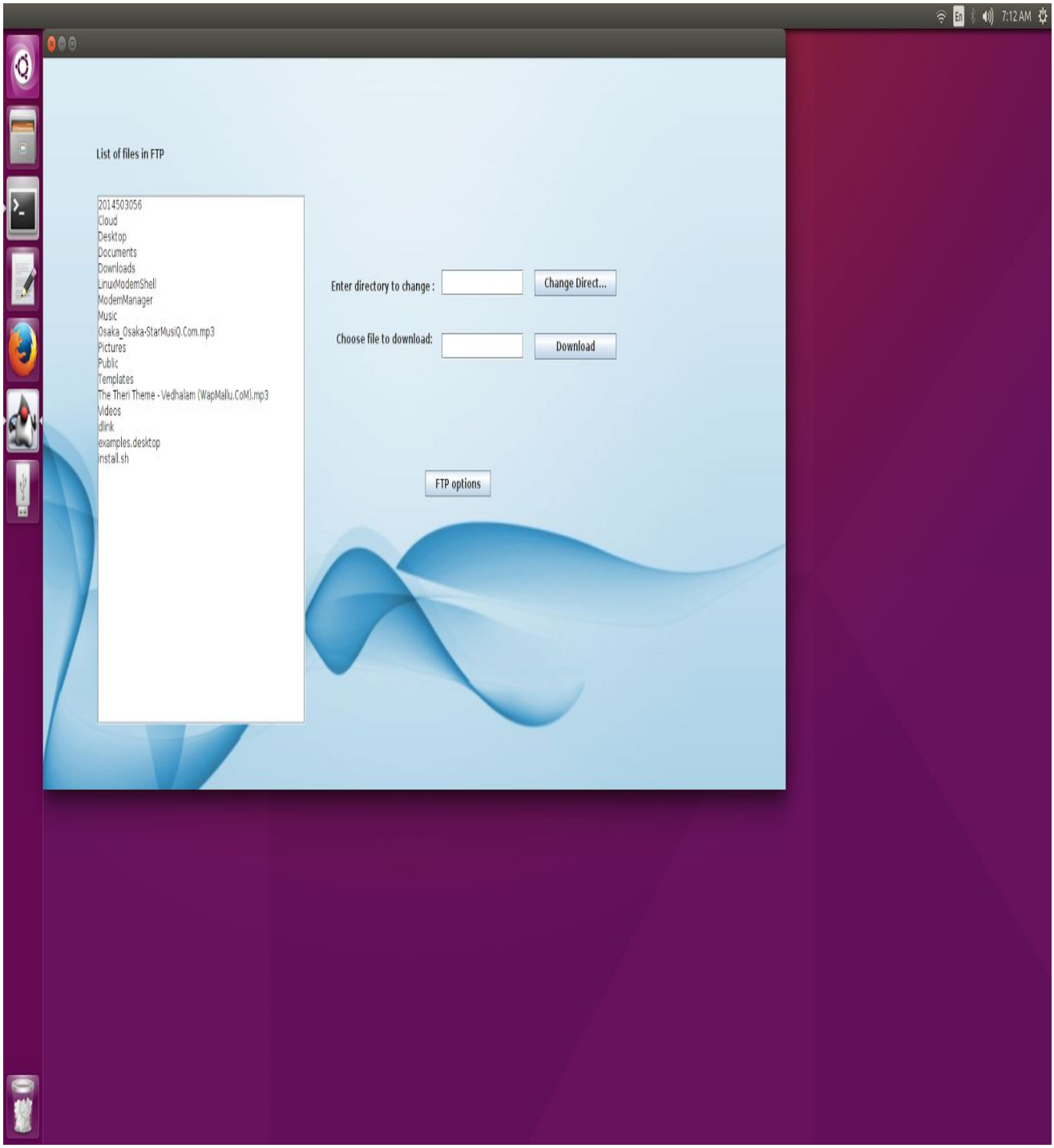

Fig 12 : File Download Window

Conclusion :

Thus the above project leads us to utilize the computer memory efficiently. We can store large number of useful data in our computer. We can use Secure Shell in order to provide SSL level security features and continuously use that channel to send and receive commands. Not only we can send and receive commands we can use it for file uploads, file downloads and using it as medium to send any form of digital data. We have to understand the key differences between SSL and SSH. We can also use more than one applications at the same time, without installing applications in our computer.

The above project supports both machines which are inside the LAN and outside the LAN as it is connected through internet. By using SSH we have ensured the security of the data that is being transmitted from server to client. The applications can be used as such without using any third party software like browser to get the application for use. Thus this project aims at providing quality service to users by providing entire application to user without any compromise in quality. Also with the advent of microsoft's windows 10 anniversary update we can access windows application in linux systems. The above project is implemented in java which makes it platform independent. Thus we can execute this software across all systems.

REFRENCES:

[1].Leena Rani; PreetiNarula&Neeti Panchal (2014).FTP - The File Transfer Protocol. International Journal of Research (IJR) ISSN : 2348-6848

[2].TanerArsan, FatihGünay and Elif Kaya IMPLEMENTATION OF APPLICATION FOR HUGE DATA FILE TRANSFER (August 2014). International Journal of Wireless & Mobile Networks(IJWMN)

[3].Parker, Don (September 2005). "Understanding the FTP Protocol". Windowsnetworking.com.

[4].Conrad Chung(2010). An introduction to File transfer protocol. http://www.2brightsparks.com/

[5].PETER F. LININGTON (September 1989). File Transfer Protocols.IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS. VOL. 7. NO. 7.

[6].S.AGGARWAL,K. SABNANI,andB.GOPINATH(December 1985). A new file transfer protocol (FTP). AT&TTechnicalJournalVol. 64, No. 10.

[7]. ANOOSHA GARIMELLA, D.RAKESH KUMAR (March2015). SECURE SHELL-ITS SIGNIFICANCE IN NETWORKING(SSH). International Journal of Application or Innovation in Engineering & Management (IJAIEM)

[8].NIDHI KANDHIL, Dr. ANIL KUMAR(August 2011). A Study on Secure Shell (SSH) protocol.IJCSMS International Journal of Computer Science & Management Studies, Vol. 11, Issue 02.

[9].Mrs. Monica Goyal and Rajinder Kaur (February 2013). A Survey on the different text data compression techniques.

[10]Thomas Beth and Dieter Gollmann, “Algorithm Engineering for Public Key Algorithms”, IEEE Journal on selected areas in communication, VOL. 7. NO 4. MAY 1989.

[11]M. Riedel, D. Mallmann and A. Streit, "Enhancing scientific workflows with secure shell functionality in UNICORE grids," *First International Conference on e-Science and Grid Computing (e-Science'05)*, Melbourne, Vic., 2005, pp. 8 pp.-139.

[12]P. Iyappan, K. S. Arvind, N. Geetha and S. Vanitha, "Pluggable Encryption Algorithm In Secure Shell(SSH) Protocol," *2009 Second International Conference on Emerging Trends in Engineering & Technology*, Nagpur, 2009, pp. 808-813.