* **Implementation**

Implementation of this project is to build the nexus between Bitcoin and cloud technologies.

(Implementation of this project is to develop a POC to showcase the nexus between Bitcoin and cloud technologies). The cloud setup for the project was chosen as Infrastructure as a Service (IaaS) model. The web application acts as a means of user interface for payments in Bitcoins is “InStaRS” or Instance Renting Service.

The following sections describe in detail the different software components, cloud setup and various APIs used in the process of implementation of this project.

* **Software Components**

The software components for this project can be divided broadly as Cloud Software Components and Bitcoin-Web application Software Components.

* + *Cloud Software Components*
    - **Eucalyptus**

Eucalyptus is open source software for building AWS-compatible private and hybrid clouds. The installation package chosen was Eucalyptus FastStart 3.2.2 which comes preconfigured for CentOS and KVM. This package facilitates installation of Eucalyptus components on CentOS 6.0 , an Enterprise class Linux Distribution, along with AWS compatible addons.

* + - **Euca2ools**

Euca2ools is a Eucalyptus management utility. It provides with command line based management of various Eucalyptus components like :

* + - * Instance Key Management.
      * Image Management.
      * Instance Management.
      * Network Management.

Apart from being management utility for cloud setup management, Euca2ools also provides with the necessary interfacing with AWS which is required by the web application to remotely control the cloud as per the client requests.

* + - **KVM - Qemu**

Kernel Virtual Machine or KVM is a Linux module which allows user programs to utilize hardware virtualization features of the modern processors. Qemu-KVM fork is used for x86 machines. It allows us to create multiple virtual machines which run as normal Linux processes.

* + *Bitcoins-Web Application Software Components*
    - **Bitcoin-Qt**

Bitcoin-Qt is a desktop Bitcoin client application. This comes bundled with bitcoind service. Bitcoin-Qt along with bitcoind allows users to maintain a wallet on local system and make transactions. It works by downloading the complete blockchain on the local system, current blockchain size stands at 6400MBs. Bitcoin-Qt also allows uses to generate Bitcoin addresses required for any type of Bitcoin transactions.

* + - **Bitcoin JSON-RPC APIs**

Bitcoin supports various APIs to enable merchants/businesses join and make use of the Bitcoin payment network for their daily business transactions. These APIs are available in a variety of programming languages like Python, Ruby, Java etc. These APIs are also available as lightweight and heavy, where heavy APIs make use of local resources for all Bitcoin related activities. For the implementation of payment module of InStaRS, we have made use of JSON APIs provided by blockchain.info network. This set of APIs is lightweight as it utilizes the blockchain.info’s infrastructure for Bitcoin transactions. Since local resources are not burdened, the transaction verification process is sped up remarkably as compared to heavy APIs like BitcoinJ.

* + - **Bouncy-Castle Open-PGP encryption**

Bouncy Castle is a collection of lightweight Java APIs used for cryptography. As part of InStaRS’s Bitcoin payment module, an extra layer of security based upon Open-PGP encryption is implemented using Bouncy Castle API for Open-PGP.

**System Design**

This project uses 4 machine configuration to create a private cloud using Eucalyptus integrated with a web server to host the InStaRS web application for payments and cloud control as shown in the below figure :

* A web server running the InStaRS web application on Apache Tomcat Server 7.0 using struts2 framework for client interfacing and cloud management.
* A front end server, running Eucalyptus Front End components : cloud controller, cluster controller Walrus storage service and storage controller.
* A Back end server also known as Node Controller to run the virtual machines.
* A client system to bundle, test and register the images for the cloud service.

To showcase the scalability of cloud setup, the split architecture is used. The complete project setup has also been readied and tested with the Back-end and Front-end systems clubbed onto a single machine. InStaRS is designed in such a manner that with minimum configuration changes, the web application can be made to work in tandem with the monolithic cloud architecture as well.