International Journal of Engineering & Technology, 7 (2.8) (2018) 243-247

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research Paper

Cloud security using self-acting spontaneous honeypots

Nooreen Fatima Khan[1*], M Mohan[2]

[1]M.tech Student, Computer Science Department, SRM University, Delhi NCR Campus
[2]Professor, Computer Science Department, SRM University Delhi NCR Campus
**Corresponding author E-mail: khannooreenfatima@gmail.com*

Abstract

Cloud Computing is growing in terms users, infrastructure, services, also security issues like: Cyber attacks are increasing day by day security community need some better mechanism to learn about attacks and which can provide an improved response against these security issues in cloud effectively. Current defences, security solutions, security equipments doesn't cover two or all three security concepts which are prevention, detection and response. Honeypot security resource can be used to add value to the cloud security community it can cover all three security concepts if implemented intelligently. In this project a high-interaction based self-acting spontaneous honey pot, abbreviated as SAS HP, which can dynamically change its behavior after learning from an attacker, is proposed and its architecture is given which can be deployed in the cloud environment for the analysis of attack patterns and to secure cloud systems. Also, the concept that how the instances of this honey pot can be made available as a service to the customer and how this SAS HP can be deployed with in cloud is given in this report. The aim is to develop the working prototype of the proposed system in cloud environment.

Keywords: *Cloud, Honey Pot , Prototype, Attack Pattern, -Acting Spontaneous Honey Pot, (SAS HP), Instances.*

1. Introduction

Nowadays, the Information Technology (IT) world is witnessing the cloud computing era. Majority of the people are shifting to the cloud environment. With the growing size and complexity security issues in cloud are also growing. These security issue encompasses hacking, attacking, intrusions, trojan horse launching , logic bombs launching, denial of service attacks, spreading worms & viruses and more. But remote hacking of an instance running in the cloud is one of the major security issue. However, intrusion to a network or system cannot be entirely rejected but can be minimized. Most of the research work is being done mainly on three concepts of security and honeypots fulfills two main concepts of security that is: Detection and Response it gathers as much information as possible on the attack. This information could be utilized by the security community to develop better services and soultions in the direction of security, hence we can envisage that honeypot (if implemented intelligently) can help to cover prevention concept of security by the security community as well because honeypot can be used by them to save or protect the real instances in the cloud. The honeypot should operate very carefully so that the attacker can not be aware of it or cannot detect its presence and the attacks on the real instances of production or services in cloud can be prevented through filtering packets from vicious users from legal users and redirecting this malicious traffic to the self adjusting spontaneous honey pot instances deployed in cloud. But developing, deploying and securing such honeypots in a sneaky manner is not an easy task there can be different system level and technical level challenges in doing so.

Although many previous research contributions have addressed these underlying challenges of developing, deploying and securing such honeypots in a sneaky manner but very few people have demonstrated the ways to make honeypots intelligent, more spontaneous and more adaptive and have deploy such honeypot in cloud.Nowadays, attackers have become so smart to identify or detect existing honeypots easily and take over it due to high availability of smart advanced tools and bulk of knowledge on different attacking skills and moreover because of the static nature and limited dynamic nature of existing honeypots this take over has become easy for them and also such honeypots provide very little important information about attacker. So, idea is to develop a reinforcement learning based self adaptive honeypot that can optimally communicate with an attacker without being exposed using its adaptive nature during communication and can get much valuable detailed information about attacker like: method and software or tool for breaching into the system , data in which attacker show interest, attacking behaviour, techinal skills , what he tries to download or install, native language, IP address, platform details , geo-location etc.The main objective of this project is to develop a new prototype of self capable spontaneous honeypot which is inspired by the heliza honey pot prototype and RASSH and provide its architecture which can be used in cloud environment.

2. Cloud Security Brief Summary

Generally Cloud security involves set of rules, technologies, and security checks and controls which are used to protect applications, information, and the associated substructure of cloud computing. It comes under information security (or computer or network security) [1]. With the extensive use of virtualization technologies combined with self-service potentialities cloud service providers are providing services to their customers through the cyberspace. In cloud, same physical server is used to

Copyright © 2018 Authors. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

hold and run virtual instances from distinguished more than one organizations to optimally increase the efficiencies and effectiveness of virtualization. Vendors must make sure that their customer's critical data and applications are safe. Now a days , organizations are using features and services of cloud computing to extend their business but still security of data and application is considered to be at higher priority [2]. These services includes SAAS (Software as a service) , PAAS (platform as a service), IAAS (infrastructure as a service) which have helped enterprises to make huge business profits and growth [3]. Other services that are currently being seen as an emerging style of computing includes IT-as-a-Service that provide support to an enterprise IT functions [5] , Storage as a service in which storage or database like services are provided to the enterprises [6] , Anything-as-a-Service (XaaS) its examples include network as a service (NaaS) , storage as a service (SaaS), monitoring as a service (MaaS) and communications as a service (CaaS). In short XaaS technology has the potential to remold IT [7]. The computing style of pay-as-you-go applications plat-forms for development , storage capability , software utility , processing capability, other cloud based services makes cloud computing more attractive to use [8]. But still there are fears , concerning these emerging computing styles like SaaS, IaaS, PaaS, DaaS, ItaaS,XaaS etc, among cloud users. They fear about their data and application security , about effective management of resources in cloud as they think their resources are now in hands of service providers so they fear to put their critical resources to the cloud. To relieve above mentioned security concerns, a cloud service provider must assure security and must provide answer to these security concerns of their cusomers. This paper define how SAS (Self-Adaptive Spontaneous) honeypots can provide useful data to the cloud security community so that they can come up with better security measures and solutions in future. Honeypots are the security tools which are used to set up a trap to lure attackers, Honeypots are nothing but a fully operable computer or a computer service systems which are allowed to on-line attack but have no productive use or function for everyday work within the organization. These are set up to interact with the attackers and collect information about them. This information includes attacking patterns, platforms used by them for attacking , attacker's intention and needs, date & time of attacking, method of attacking , commonly used programs launched by them and much more can be obtained depending upon the capability of honey pot. Honeypot has to be self adaptive to intelligently interact more and more in order to obtain more and more information about the attacker and to hide their presence from sophisticated attacker. This information about attacker provided by deployed honey pot can help the security community to know more in detail about the attacker's ability particularly details about their technical skills. According to their level of interaction honeypot systems can be classified as : *Low interaction:* there is no interaction between attacker and underlying operating system and these honey pot system displays only particular services such as FTP, HTTP , SSH.. *High Interaction:* nothing is emulated or restricted in these systems, provide direct interaction between attacker and underlying operating system which helps in getting large amount of information about attackers. But underlying operating system can be possibly compromised. According to the level of adaptability honey pot can be categorized as: *Static:* These honey pot systems are with fixed configuration settings and provides same appearance to the attacker. Sophisticated attackers can identify such honey pots and can take over it. *Dynamic:*These honeypot systems change their behavior based on the network configuration around them and the interaction they are facing every time they interact with an attacker and are hard to be detected by the sophisticated attackers.

3. Literature Review

Nithin Chandra S.R, Madhuri T.M in their work entitled , "Cloud Security using Honeypot Systems ", give a broad overview of presents cloud security issues , the potential of honey pot for the security community and cloud security specialist , emphasises theoretical concept of honey pot systems, their types based on interaction level and implementation levels and their use for providing cloud security. Also, their work give brief overview of Advantages and Disadvantages of honey pots. But the main emphasis is on theoritical concepts rather than giving the practical implementation of honey pot to ensure security in cloud environment [10]. Gerard Wagner et al in , " Adaptive and Self Configurable Honey pot", made initiatory effort to build a honeypot which is adaptive in nature . They have tried exploring game theory and have used it in the implementation of high interaction based honeypots which can either allow or block the execution of a program. They have computed Nash Equilibrium resulting in a mixed equilibrium discovering the optimum blocking probabilities for the honeypot. The problem with their work is that no additional capabilities of honey pot have been demonstrated in their work except allowing and blocking the command of an attacker. Also state representation is needed to be improved. Moreover , experienced and sophisticated attackers could replace states and poison the learning process [11]. Gérard Wagener et al in , "Heliza: talking dirty to the attackers" , proposed Heliza, a self-adaptive honey pot system which uses reinforcement learning technique to interact with the attackers, self adaptability makes it to increase interaction with the attacker and gather much detailed information by using game theory & machine learning techniques. Authors uses Weakly configuered SSH server to implement and deploy heliza to estimate its performance which they further compared with other existing low interaction & high interaction based honey pot system's performances. Corresponding to attain two goals heliza is designed to be based on two reward functions. These goals includes:

- holding attackers to interact with the honeypot as much as possible.
- luring attackers to download or install custom software;

Heliza uses Markovian decision process based on which it attains quick learning capability with little human controlled interferences. But Heliza needs improvements in its scalability (because it uses linux system based UML implementation) , needs improvement in terms of learning capability (because its implementation uses exclusively the SARSA algorithm) , its insults in English language only which causes some (who don't know English) attackers to leave soon [12]. Adrian Pauna and Ion Bica , in their paper "RASSH - Reinforced Adaptive SSH Honey pot" they presents RASSH system which is an evolution of kippo and also emulates SSH server. They uses reinforcement learning methodology to present reinforced honeypot model. They have given comparison study of RASSH with Heliza and demonstrate their additional work of eliminating the need of second reward function in heliza by adding delay function in RASSH. They have used python to implement their working prototype. But author concluded RASSH as not fully operational and expects improvements in terms of scalabilty, coding efficiency and further optimizations of action from the honey pot in response to commands executed by attacker during the interaction [13]. Harald Gjermundrod and Ioanna Dionysiou wrote in their paper named as , "CloudHoneyCY - An Integrated Honeypot Framework for Cloud Infrastructures" , about how collection of low interaction and high interaction honey pots can be deployed in cloud environment using honey pi to provide security controlled environment. They have given an open source framework as " CloudHoneyCY " which utilizes the components of HoneyCY, an existing unified honeypot management system that supports a variety of open source honeypots monitored and managed from single interface (either web based or application based interface) , in cloud environment. CloudHoneyCY has 3 tiers architecture consisting of honey Pi (runs various honey pots on inexpensive Raspberry Pi and generic x86 servers), Honey Srv (manages the collection of honey pi devices and the generated information) and Honey Vm (Performs analysis of collected

malwares) as main components deployed in cloud environment to hide actual production system and distract attackers away from real system services. But Improvements to the security console of the honey pot unified management is required to add additional features , costly architecture , need more self adaptability features (like : better reverse turing test capabilities) [14]. In the research paper issued by European Research Area Specific Programme "European Network of Affined Honey Pots" , NOAH (Network of Affined honey pots) architecture have been proposed which glues existing components (i.e network components like funnels, tunnels , services like automated signature generation for zero day attacks, different existing low interaction (traffic filter) and high interaction honey pots (for handling traffic that cannot be handled by the LI honey pots and providing optimal level realism)) and mechanisms inside its core part to provide stable infrastruture to achieve security from hackers, attackers and malwares. NOAH is demonstrated as a distributed set of honey farms that can collaborate . Collaborating parties (like : home users , organizations , campuses , institutes etc) deploy a plug (or uses a tunnel) and share their black addresses (or port space) to NOAH core. These parties performs such sharing via Honey@home tool installed on their systems which uses DHCP that automatically takes unused IP addresses at the collaborating party site. But these Honey@home tool can also be installed by an attacker to detect honey pot in NOAH core and he may attack by flooding, DOS attacks, poisoning or black listing or even may manipulate other client black space. So, these honey pots (in NOAH core) and trusted clients must be hidden from attacker and automatic installation of this Honey@home tool must be prevented [15]. B Sri Chitra Poornima and M Balamurugan in "Honeypot as a Service in Cloud", gave a decision and a redirection based architecture that mechanically filters out attacks and keeps the particulars about the attackers. Based on these particulars about the attackers legitimate action can be taken or even these attackes can be blocked. The proposed architecture emphasizes the necessity of some components to implement a honey pot system in cloud envrionment these components include Cloud controller (centralized controlling unit , used to manage and expose the virtualized resources via user-facing APIs) , Cluster Controller (used to manage Multiple clusters present in a cloud environment and act like Virtual Machine Manager (VMM) controlling the execution of various VMs running on the nodes and their interaction with the users) , Honey Controller (Cluster Controller under which the Honeypots virtualized instances are running is called as Honey Controller.), Filter and Redirection Engine (engine running on the central Cloud Controller analyse the incoming packets based on some flagged signatures and the fake ports running in the Honeypot and separate the traffic of malicious users from legitimate users and to redirect them to the Honey Controller.) , Log Storage System (used to store logs which contain information regarding IP address of attacker, type, time, date and frequency of attack) [16].

4. Problem Description

EXISTING PROBLEM– Security issues in cloud computing are increasing. Fresh threats, attacks and vulnerabilities are discovered everyday. Attackers are becoming more sophisticated and coordinated and coming up with new methods of attacking exploiting cloud resources so there is a need for the vendors to be trained to prevent, detect, and recover from them. So, strictly monitoring cloud activity has become mandatory step for cloud providers and cloud security analysts to understand these threats and attacks and to build better protections. Use of intelligent Honey pot can help in fulfilling these needs and requirements of cloud providers and security community to a great extent.

PROPOSED SOLUTION - This paper work presents a SELF-ACTING SPONTANEOUS HONEYPOTS SYSTEM that we are developing to contribute to CLOUD SECURITY which will help Cloud Security community in all three concepts of security that is Detection, Prevention and Response. This proposed honeypot is based on reinforcement learning technique and it aims to build attacker profiles which includes much detailed information about attackers which can be further analysed by the security community to develop better security plans.

5. Main Component Block Diagram for SAS

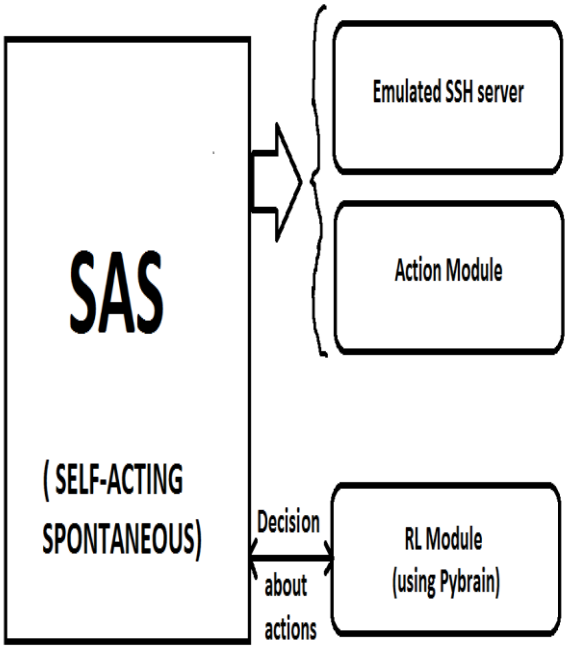

Fig. 1: Component Block Diagram

EMULATED SSH SERVER: For this existing solution kippo or cowrie with some modifications (to add additional capabilities) can be used. Kippo is a medium-interaction SSH honeypot written in Python. Kippo is used to log brute force attacks and the entire shell interaction performed by an attacker. It is inspired by Kojoney [17].

ACTION MODULE: This module is used to implement the actions (decided by the RL module in PyBrain) in the shell offered to attacker by self adaptive honey pot. These actions includes allow the execution of the command generated by the attacker , blocking the command , giving fake outputs for the command executed by the attacker, delaying in output, susbtitute command, simply execute the command , or just returning some errors for the issued command.

1. Allow Action: Allows attacker's generated command to get executed.
2. Block Action: Blocks the execution of the attacker's generated command. While blocking the execution it generates an appropriate error message from the database.
3. Imposter Output action: This action prints fake output message from the database for the command generated by the attacker and the command itself is not executed.
4. Delay action: this allows attacker's command to get executed after some delay.
5. Insult action: This action geo-localizes the IP address of the interacting attacker and generates an insulting message in the language of an attacker so that attacker can understand it and continue with its action.
6. Substitute action: if this action is triggered, the command will not get executed but the substited command will get executed and this will result in the continuity of interaction.

REINFORCEMENT MODULE: This module generates decisions based on which actions are implemented. SAS interacts with RL module to decide upon the following:

- ➔ When to take the action.
- ➔ Which action to take.

6. Deployment of SAS at Cloud Level

SAS within cloud environment can be deployed either on real (host) systems or on virtual machines. Virtual SAS will be cost effective and easy to deploy than real SAS honey pot but setting and management efforts for these instances of SAS will be increased and we would be requiring the Honey Controller to control these SAS virtual instances for better optimized control. Virtual honeypot can only collect confined information than the real honeypot. In real SAS Honeypots each honeypot will be implemented in a disjoined host system which will definitely cause an increase in expenses and difficulties for maintenance but will gather maximum amount of information to maintain attacker or malware profile. According to Studies another drawback of real Honeypots is that it can be easily exposed and can be used as weapon against other systems whereas in virtual honeypots instances are distinguished from one another so this problem is not mostly seen or otherwise (if seen) can be easily handled.
All the network traffic or requests (entering the cloud environment) arriving at any cloud router is passed through the legal request filter which checks for the legitimacy of the packet if request/ packet is legal then it is directed to the real service instances of cloud otherwise (being malicious) it is redirected to the SAS instances deployed with in cloud.

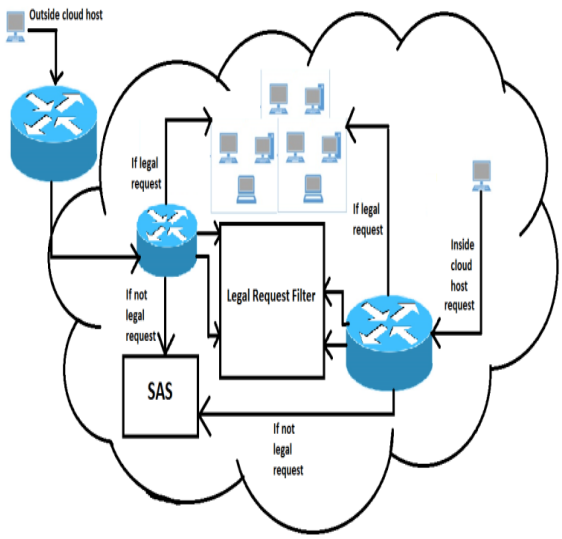

Fig. 2: Deployment of SAS in cloud environment

ARCHITECTURE OF SAS: Main components of SAS honey pot architecture consists of Honey Pot module which is a SSH server developed using existing Kippo code. Its architecture is supposed to consists of modified form of KIPPO which will be modified by developing & integrating with it some additional modules using Python platform. These modules will instill self learning , spontaneous and self acting capabilities into SAS Honey Pot by leveraging reinforcement learning. PYBRAIN which is a modular machine learning library for python can be used to implement reinforcement learning module and action module because it provides many Machine Learning Tasks supporting algorithm which are easy to use and quite flexible.
SAS Honey Pot module = KIPPO (with emulated commands) + additional commands.

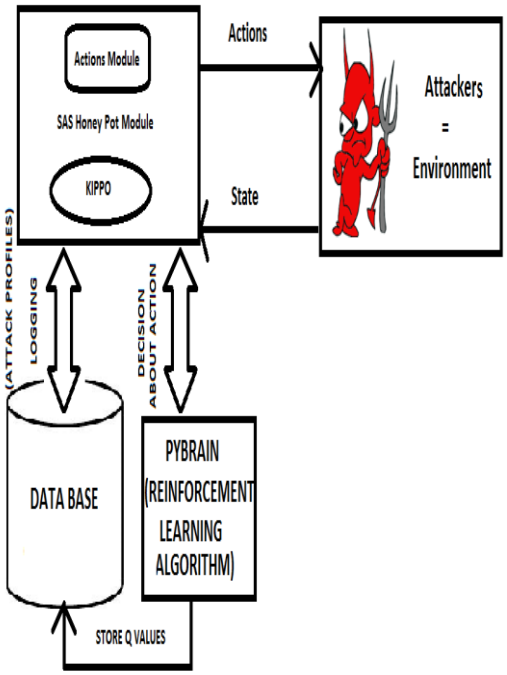

Fig. 3: Architecture of SAS Honey Pot

SAS : Self Acting Spontaneous Honey Pot
Additional commands as specified in VIII section and the number of additional commands may change during implementation. SAS architecture also consists of Database (which can MY SQL based database). SAS agent (honey pot module) can access this database to store/retrieve different messages (erroneous, fake , insult) corresponding to different actions,fake files, fake data, attack logs , attacker's profiles, malware logs, malware profiles, RL module access this database to store Q-values.
Attacker when issue any command to SAS agent then SAS has to implement some action towards attacker this action is specified by the reinforcement learning module (which stores Q-value on to database and define action and specify the timining of action module). The action module of the SAS honey pot implements that specified action. For this action a specific reward is received by tha SAS HP because this action may cause the attacker to change its behaviour and issue some other command. SAS HP module may use either kippo approach and use fake file system or may use proxy mode to operate in proxy mode it will act as a proxy between the imitated OS which is being accessed by the attacker and the real OS shell. So, Honey pot module waits for the action determined by RL module and executes it instead of directly executing attacker's command at real operating system.
SAS HONEY POT AS A SERVICE IN CLOUD : SAS honeypot can be used to make further business profits by providing it as a service to the clients. Customers should be allowed to purchase instances of SAS as a service (in order to increase security level) for their systems or instance in cloud.The idea is to keep honey pot instances deactivated and should be activated only upon its purchase. List of activated honey pots and its corresponding owner customer's instance (or system for which it is purchased) should be maintained and its entry should be done at legal request filter so that all the vicious traffic to the customer system can be directed to the corresponding hone pot instance by the filter. Hence, this facility can help in securing the real instances or systems in cloud from malicious activities and also customers can be provided with the logs of attacker and malicious code details so that customer can also know details about the attacker and can take necessary actions against him and also can take actions to protect their resources in future.

7. Conclusion

In this paper a high-interaction based self-acting spontaneous honey pot, (SAS HP) which can dynamically change its behavior after learning from an attacker, is proposed and its architecture is given which can be used to implement and deploy it in the cloud environment, different techniques of its deployment in cloud are suggested and details about how SAS can be helpful to the security community , cloud users (both service providers and their customers) are given. Also, the concept that how the instances of this honey pot can be made available as a service to the customer is presented.

References

[1] https://en.wikipedia.org/wiki/Cloud_computing_security.

[2] Krešimir Popović, Željko Hocenski ,“ Cloud computing security issues and challenges “, Institute of Automation and Process Computing Faculty of Electrical Engineering Osijek.

[3] S. Srinivasan, Cloud Computing Basics, SpringerBriefs in Electrical and Computer Engineering, DOI 10.1007/978-1-4614-7699-3_2, © Springer Science+Business Media New York 2014

[4] https://en.wikipedia.org/wiki/IT_as_a_service

[5] White paper on IT as a service Handbook : Ten key steps on the journey to ItaaS

[6] https://www.techopedia.com/definition/24900/storage-as-a-service-saas

[7] http://searchcloudcomputing.techtarget.com/definition/XaaS-anything-as-a-service

[8] http://www.computerweekly.com/feature/Pay-as-you-go-computing

[9] Wira Zanoramy Ansiry Zakaria, Miss Laiha Mat Kiah "A review of dynamic and intelligent honeypots." ScienceAsia 39S (2013): 1–5.

[10] Nithin Chandra S.R, Madhuri T.M, “Cloud Security using Honeypot Systems”, International Journal of Scientific & Engineering Research Volume 3, Issue 3, March -2012 1 ISSN 2229-5518.

[11] Gerard Wagener et al , “Adaptive and Self-Configurable Honeypots” , 12th IFIP/IEEE International Symposium on Integrated Network Management 2011

[12] GérardWagener et al , “Heliza: talking dirty to the attackers”, Received: 18 June 2010 / Accepted: 23 November 2010 © Springer-Verlag France 2010

[13] Adrian Pauna and Ion Bica , “RASSH - Reinforced Adaptive SSH Honeypot” , 978-1-4799-2385-4/14 ©2014 IEEE

[14] Harald Gjermundrød & Ioanna Dionysiou , “CloudHoneyCY - An Integrated Honeypot Framework for Cloud Infrastructures”, 2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing.

[15] European Research Area Specific Programme “European Network of Affined Honey Pots” , NOAH Research Paper.

[16] M Balamurugan and B Sri Chitra Poornima in “Honeypot as a Service in Cloud”, International Conference on Web Services Computing (ICWSC) 2011 Proceedings published by International Journal of Computer Applications® (IJCA).

[17] https://en.wikipedia.org/wiki/Kippo

[18] Mahesh Mudavath, K Hari Kishore, D Venkat Reddy "Design of CMOS RF Front-End of Low Noise Amplifier for LTE System Applications Integrating FPGAs” Asian Journal of Information Technology, ISSN No: 1682-3915, Vol No.15, Issue No.20, page: 4040-4047, December 2016.

[19] T. Padmapriya and V. Saminadan, “Improving Throughput for Downlink Multi user MIMO-LTE Advanced Networks using SINR approximation and Hierarchical CSI feedback”, International Journal of Mobile Design Network and Innovation- Inderscience Publisher, ISSN : 1744-2850 vol. 6, no.1, pp. 14-23, May 2015.

[20] S.V.Manikanthan and K.srividhya "An Android based secure access control using ARM and cloud computing", Published in: Electronics and Communication Systems (ICECS), 2015 2nd International Conference on 26-27 Feb. 2015,Publisher: IEEE,DOI: 10.1109/ECS.2015.7124833.

[21] K.Srikar ,M.Akhil ,V.Krishna reddy “Execution of Cloud Scheduling Algorithms” International Innovative Research Journal of Engineering and Technology ISSN NO: 2456-1983.Volume 2, Issue 4 June 2017. 108 -111.