Security Implementation through PCRE Signature over Cloud Network

Gaurav Raj[1], Munish Katoch[2]
[1]Asst. Prof. , Lovely Professional University,
[1]er.gaurav.raj@gmail.com,
[2]M. Tech. Student, Lovely Professional University, Phagwara.
[2]munishkatoch710@gmail.com

ABSTRACT

With invention of new tools and technologies, the attackers are designing new methods to evade present security models. One of such security models is Intrusion detections. Intrusion detection systems work on signature based analysis and anomaly based detection, which makes it vulnerable for new evasion techniques. This work describes the mitigation of evasion techniques by implementation of better PCRE based rules approach.IN this paper we are designing improved PCRE based rules to prevent evasion techniques on cloud systems.

KEYWORDS

Cloud Computing, PCRE Signature, Virtual Machine

1. INTRODUCTION

Intrusion detection is the process of monitoring and analysing the events occurring in a computer system or network for signs of possible incidents. Intrusion prevention is the process of performing intrusion detection and attempting to stop detected possible incidents. Intrusion detection Systems and Intrusion prevention systems (IDS/ IPS) are primarily focused on identifying possible incidents, logging information about incidents, attempting to stop them, and reporting them to security administrator. We set the platform to implement IDS and IPS system over Ubuntu (Linux version) with the help of Snort as an IDS and IPS engine. We are using BASE and Aanval as real-time network monitoring frontends. To store alerts and logs we use Mysql as a central database. Apache server is also required for running frontend server. As we are setting up these systems on Cloud, we need to implement them on virtual machines so we implement virtual environment for attack detection and traffic analysis using VMWARE. Bind server used for providing DNS services for client and setting up Ubuntu desktop as a gateway system. Backtrack 5 is used as an attacking windows host as victim host.

1.1 Snort as IDS/IPS

The Snort is open source intrusion detection and prevention technology. Snort uses a rule-driven language which combines the benefits of signature, protocol and anomaly-based inspection methods. As an IDS and IPS, It can perform real-time packet analysis and logging on network. Protocol analysis and content searching/matching are the most powerful features of snort which can be used to detect variety of attacks such as buffer overflow, stealth port scans, SMB probes, OS footprinting etc. Snort can be configured for three different modes:-

a) Sniffer
b) Packet logger
c) Network intrusion detector

The Sniffer mode gets real-time traffic from the network and simply display's it on the console whereas the Packet logger mode is used where we need to store the real time traffic into the

database for further analysis and research. The best part of Snort IDS is Network intrusion detection in which Snort can be configured to user –defined rules and signatures for different protocols, ports and attacks. Entire network traffic flows through the Snort IDS engine and if a signature triggers an alert, it is then saved to the central database in Mysql.

1.2 Detection Process

We implement two intrusion detection approaches in Snort are as follows:-

1. Signature-based: It compares threat signatures with the threat database.
   a. This is very effective at detecting known threats but largely ineffective at detecting unknown threats.
   b. Signature-based detection cannot track and understand the state of complex communications, so it cannot detect most attacks that comprise multiple events.

2. State full protocol analysis:
   a. Protocol analyzers can natively decode application-layer network protocols, like HTTP or FTP.
   b. Once the protocols are fully decoded, the IPS analysis engine can evaluate different parts of the protocol for anomalous behaviour or exploits against predetermined profiles of generally accepted definitions of benign protocol activity for each protocol state.

2. RELATED WORK

In the paper "A Cooperative Intrusion Detection System Framework for Cloud Computing Networks" the authors Chi-Chun Lo, Chun-Chieh Huang and Ku, J. has proposed a framework of cooperative intrusion detection system (IDS) in order to reduce the impact of denial-of-service (DoS) attack or distributed denial-of-service (DDoS) [1].

In the paper "Robust and Fast Pattern Matching for Intrusion Detection" the author Namjoshi, K. Narlikar, G explores how the rules are being written using the more powerful regex syntax, which includes non-regular features such as back-references.The author also explores the implementation of the algorithms in the context of the Snort IDS and experimental results on several packet traces which show substantial improvement over the backtracking algorithm [2].

In the paper " An Efficient Broker Cloud Management System" the author Mr. Gaurav Raj explored to manage Broker Cloud Communication, an efficient management infrastructure is required which also take care for cost and time to get application services. In this paper, an Efficient Broker Cloud Management (BCM), a system which works in Broker Cloud Communication Paradigm (BCCP) has been proposed to find a communication link with minimum cost of link uses between broker and cloud. An algorithm, Optimum Route Cost Finder (ORCF) has been proposed for finding optimum route between broker and cloud on the behalf of cost factor [3].

In the paper "Issues of cloud computing security in high speed railway" the author Xiang Tan Bo Ai has explore the status of the development of cloud computing security, Analyse the data privacy, Security Auditing, Data Monitoring and other challenges that the cloud computing security are facing. The author also describes the solutions for issues like Virtualization Security and Traffic Monitoring between virtual machines and the challenges arise due to growing of audio and video data such as Large scale distributed computing ,Data analysis and processing ,Data sharing and the integration of computing resources [4].

In the paper "Implementing Digital Signature with RSA Encryption Algorithm to Enhance the Data Security of Cloud in Cloud Computing" the authors Uma Somani, Kanika Lakhani, Manish Mundra focuses on how Data Security in the cloud implementing the digital signature

with RSA Algorithm. The paper also focus that as we are using the shared environment in the cloud computing so we are always lose control over our physical security and the data is always exposed with the shared environment of the other companies. To maintain integrity is also the major concern of this paper it means the changes should only be made in the case of authorised transactions. That's why the concept of digital signatures is included in this case along with RSA Algorithm [5].

3. PROPOSED SECURITY SYSTEM ARCHITECTURE

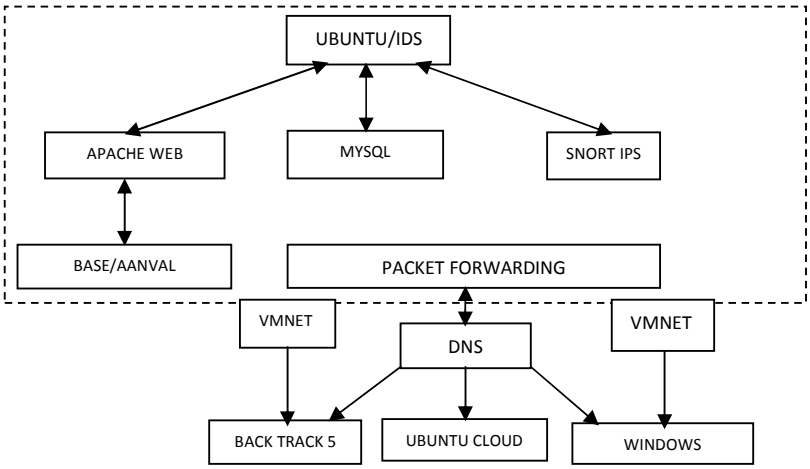

Figure 1. Experimental Design of Secure System Architectural

3.1 Applied Methodology

1. Firstly, we set up a gateway machine that runs with two virtually independent network adapters. Snort is configured in IPS mode (Intrusion prevention) on Ubuntu 11.04 desktop system (gateway).
2. The gateway machine (Ubuntu) requires two services to serve as router :-
   a. DNS (Domain Name Server):- Service requires a package **'dns-server'** on ubuntu machine. The service is running on **TCP/UDP port 53** on gateway machine (Ubuntu). The purpose of the service is to resolve DNS queries from different internal virtual networks (VMNET1 and VMNET2).
   b. Packet forwarding:- Packet forwarding is required to route packets from internal hosts and gateway machine and between vmnet1(network1) to vmnet2(network2). This is done by enabling packet routing in ubuntu machine:-
   Edit the file **- /proc/sys/net/ipv4/ip forward**
   Setting its value to 1 (to enable packet routing).
3. Now we require virtual environment with manually configured and virtually independent internal adapters.
4. Two machines are implemented on virtual environment as –
   a. Windows XP (as target machine) and backtrack5 (as attacker machine).The Windows XP machine is configured for VMNET1 (network 1) with DNS and gateway configured as VMNET1.
   b. Backtrack5 machine is configured with VMNET2 (network2) with DNS and gateway configured as of VMNET2 (network2).
5. The machine is also required a web server – we install apache server to run frontend and a central backend database server as Mysql.
6. Now setup of Snort requires packages :–

a. -Snort (detection engine)
b. -Barnyard (to log alerts to the backend database 'mysql'
c. -Adodb (to create connection between the backend database and the front end.
d. -BASE (Basic analysis and security engine) as the front end to view real-time alerts.

7. Now we have a fully functional virtual organisational network. Next thing we are doing is generation attack traffic from attacker machine targeted for victim machine and analysis the traffic with the network analyser (wireshark). Deal with the detection of more and more attack evasion techniques.
8. Next we require is network traffic analysis – for this purpose we require a network traffic analyser – in this case we are using wireshark to study different attack vectors.

4. CONFIGURATION AND RESULT ANALYSIS

1. Configurations to detection engine
   The entire detection engine (snort) is configured using one configuration file:
   /etc/snort/etc/snort.conf
2. Here we need to setup two variables –
   i. ipvar EXTERNAL_NET define external network
   ii. ipvar HOME_NET define internal network
3. In our case HOME_NET is set to 'any' and EXTERNAL_NET is set to 'any', it means we are considering any ip address for internal as well as external network.
4. Next we need to setup the signature directories we are going to use for detection. For this we need to setup variable 'var RULE_PATH' to /etc/snort/rules, which is the present rules directory in our setup.

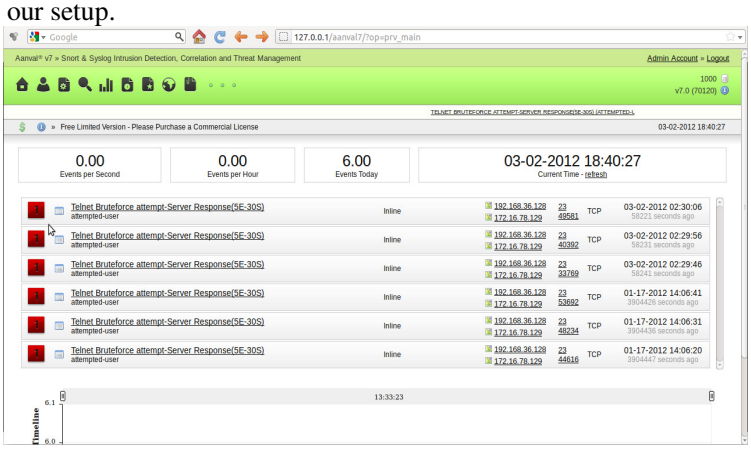

Figure 2. Screen Shot ofAanval7 as Front ends

5. Now we need to configure a tool 'Barnyard2' that is responsible for storing alerts to the central database mysql. Every snort rules requires four variables to generate an alert:
   a. reference file,
   b. classification file,
   c. gen_file,
   d. sid_file.

   /etc/snort/etc/barnyard.conf is a file where we define all variables. This file also responsible for storing alerts to the database.
6. Once the detection engine, backend database and barnyard are configured properly, we need front ends to monitor network traffic. Here we are using BASE and Aanval as front ends to minitor the realtime alerts. We need to create two databases 'snort' and 'aanval' to store alerts.

7. Both the databases 'snort' and 'aanvaldb7' are placed in limited access user accounts.
   Now we need to setup the front ends– in this setup we are using base and Aanval7 as front ends.
8. Now snort (detection engine) need to be configured:
   Snort runs in two modes – passive and inline.
   Passive mode runs in background and only monitors network traffic whereas inline mode is configured as gateway and it is capable of dropping network traffic.
   USAGE: ./snort [-options] <filter options>

4.1 Commands Required To Run Snort As Inline Mode

sudo ./snort –daq afpacket -Q -i vmnet1:vmnet2 -c /etc/snort/etc/snort.conf

Here -Q is setting up snort as a bridge.
-i used to set vmnet1 and vmnet2 bridge mode.
Snort runs on multiple packet processing libraries – pcap, afpacket, ipfw.

In order to run snort in inline mode, need to set the packet decoding library to **'afpacket'.**
The last main parameter is **'-c'**, used to set the configuration file used by snort.
Now setup Barnyard to log alerts to the mysql database. Barnyard requires multiple parameters to run:

sudo ./barnyard -c /etc/snort/etc/barnyard.conf -S /etc/snort/etc/sid-map.conf -G /etc/snort/etc/gen-msg.map -d /var/log/snort -f snort.u2 –w /var/log/snort/barnyard.waldo

-G maps snort to the gen-msg.map file– Snort needs a generator id file that is assigned to every rule(/etc/snort/etc/gen-msg.map).
-S maps snort to the sid-msg.map files – Snort also need a signature id file this is assigned to every rule (/etc/snort/etc/sid-msg.map).
-d directs snort to a directory where snort saves every alert log (generally **/var/log/snort/**). Barnyard reads all alerts from this directory and stores alerts to mysql database.

Finally Front end is configured to fetch all alerts from mysql database. Now Detection engine is waiting for the network traffic.

The major part is to perform realtime attacks on virtual enviornment.This requires a network traffic analysis tool- in this case we are using 'Wireshark' as a traffic analysis tool.

For this report, we are considering bruteforce attack and writing a 'pcre' based signature. For this we need a attacking machine (backtrack5 machine), and a victim machine (Windows machine).

For generating the attack traffic, we need a linux tool – hydra. Hydra is multiprotocol bruteforce tool.

hydra -u username -P password.list ip-address service

-u username	used for bruteforcing
-P	used for defining password list for bruteforcing.
-service	Here we are attacking is 'telnet' service.

Now we need to analyse the attack traffic on 'wireshark'. Here is a sample attack traffic of bruteforce attempt.

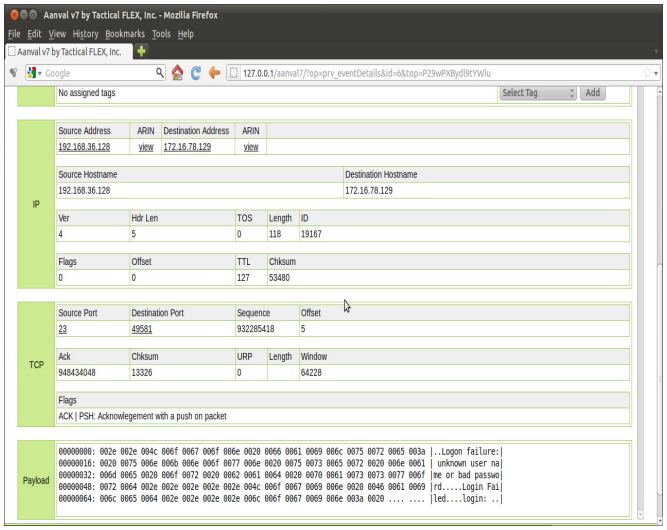

Figure 3. Analysis of sample attack traffic of bruteforce attempt

000 : 0D 0A 4C 6F 67 6F 6E 20 66 61 69 6C 75 72 65 3A ..Logon failure:
010 : 20 75 6E 6B 6E 6F 77 6E 20 75 73 65 72 20 6E 61 unknown user na
020 : 6D 65 20 6F 72 20 62 61 64 20 70 61 73 73 77 6F me or bad passwo
030 : 72 64 2E 0D 0A 0D 0A 4C 6F 67 69 6E 20 46 61 69 rd.....Login Fai
040 : 6C 65 64 0D 0A 0A 0D 6C 6F 67 69 6E 3A 20 led....login:

We need to write rule for this attack on the basis of attack parameters.
Here is rule defined for this attack traffic:

alert tcp $HOME_NET 23 -> $EXTERNAL_NET any (msg:"Telnet Login Bruteforce(5E-30S)"; fragbits:D; flags:AP,CE; pcre:"/login:/smi"; \detection_filter:track by_src , count 5, seconds 30; classtype:attempted-user; \sid:1000008; rev:1;)

This rule is generating an alert based on telnet bruteforce attempt on windows machine.
Tcp- Because telnet is a tcp service, the attack traffic generated is tcp traffic.
$HOME_NET – defines the internal network we are monitoring.
23 is the source port that is responsible for generating traffic.
-> defines the traffic flow, here it is from internal network to the external network.
$EXTERNAL_NET – defines the destination ip address.
Any – any means that the traffic is targeted for any destination port.
msg- defines the message that is displayed on front end once the alert is generated.

Next thing we require to analyse is – ip header parameters.

Here we have only 'D' flag (d'nt fragment bit set).

4.2 Sample Dump of IP Header

Internet Protocol, Src: 192.168.36.128 (192.168.36.128), Dst: 172.16.78.129 (172.16.78.129)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
....0 = ECN-CE: 0
Total Length: 87
Identification: 0xb1ce (45518)
Flags: 0x02 (Don't Fragment)
0... .... = Reserved bit: Not set

.1.. .... = *Don't fragment: Set*
..0. .... = More fragments: Not set
Fragment offset: 0
Time to live: 127
Protocol: TCP (6)
Header checksum: 0x6a18 [correct]
[Good: True]
[Bad: False]
Source: 192.168.36.128 (192.168.36.128)
Destination: 172.16.78.129 (172.16.78.129)

Next thing to look for is tcp header:

Transmission Control Protocol, Src Port: telnet (23), Dst Port: 60286 (60286), Seq: 25, Ack: 25, Len: 47
Source port: telnet (23)
Destination port: 60286 (60286)
[Stream index: 7012]
Sequence number: 25 (relative sequence number)
[Next sequence number: 72 (relative sequence number)]
Acknowledgement number: 25 (relative ack number)
Header length: 20 bytes
Flags: 0x18 (PSH, ACK)
000. .... = Reserved: Not set
...0 .... = Nonce: Not set
.... 0... .... = Congestion Window Reduced (CWR): Not set
.... .0.. .... = ECN-Echo: Not set
.... ..0. .... = Urgent: Not set
....1 .... = *Acknowledgement: Set*
.... 1... = *Push: Set*
.... .0.. = Reset: Not set
....0. = Syn: Not set
....0 = Fin: Not set
Window size: 256936 (scaled)
Checksum: 0xf7ae [validation disabled]
[Good Checksum: False]
[Bad Checksum: False]
[SEQ/ACK analysis]
[This is an ACK to the segment in frame: 67280]
[The RTT to ACK the segment was: 0.005075000 seconds]
[Number of bytes in flight: 47]

4.3 Look For ' TELNET Traffic'

TELNET

Data: Welcome to Microsoft Telnet Service \r\n
Data: \n
Data: \rlogin:

```
0000  00 50 56 3b 48 08 00 50 56 c0 00 02 08 00 45 00    .PV;H..PV.....E.
0010  00 57 b1 ce 40 00 7f 06 6a 18 c0 a8 24 80 ac 10    .W..@...j...$...
0020  4e 81 00 17 eb 7e 2a 0f 03 63 26 60 45 47 50 18    N....~*..c&`EGP.
0030  fa ea f7 ae 00 00 57 65 6c 63 6f 6d 65 20 74 6f    ......Welcome to
0040  20 4d 69 63 72 6f 73 6f 66 74 20 54 65 6c 6e 65     Microsoft Telne
```

```
0050  74 20 53 65 72 76 69 63 65 20 0d 0a 0a 0d 6c 6f      t Service ....lo
0060  67 69 6e 3a 20                                       gin:
```

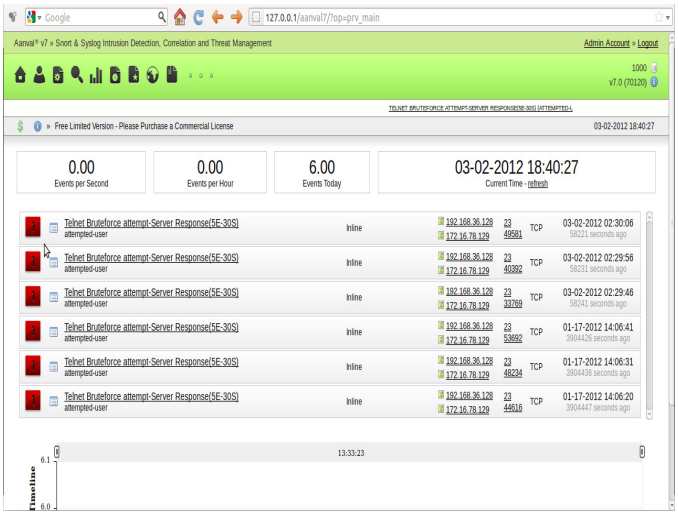

Figure 4. Screen Shot of Front end Report for Telnet bruteforce attempt.

pcre:"/login:/ -> This signature option looks for a regular expression '/login:/' in traffic pattern. Means snort is looking for string 'login:' in decoded packet.detection_filter:track by_src , count 5, seconds 30 ;

This rule option is used to track the connection state. Once the connection retry limit exceeds count 5 for 30 seconds, this option triggers the alert for once.

This option helps in the detection of brute force login attempts.

Class-type: This rule option describes the type of class to which an attack belongs to. It may be a 'unauthorized login attempt', ' attempted root access' or anything else.

Sid: Refers to the signature ID.

Rev: Stands for revision number, every time rule is modified- the rev number is to be incremented by 1.

Here is a front end report for Telnet bruteforce attempt.

5. CONCLUSION AND FUTURE WORK

Cloud system runs multiple services like computation, software applications, data access, data management and storage resources to the end user. Depending on these services we need enhanced security mechanisms. Here, we are implementing an IDS approach to deal with various possible attack vectors and minimizing the effect of these attack techniques using improved PCRE rule based approach. Also it helps effectively dealing with the attack evasion techniques. The current prevention mechanisms are still working to build the better devices to deal with new and complex attacks. With growing attacking techniques and evasion techniques, we need the better approach to deal with such threats. The IDS devices need to be more accurate in defending these threats. As with the implementation of complex pattern matching schemes and anomalous behaviour detection, we can achieve better detection and prevention against these threats. The approach behind this study is to detect and prevent evasion of rapidly evolving web 2.0 attacks and application level attacks with better deployment regular expressions. We have discussed basic regular expressions that may result in false positives on the other hand we also discussed rules for more accurate detection of the attacks with focus on reducing false positives. The current prevention mechanisms are still working to build the better

devices to deal with new and complex attacks. With growing attacking techniques and evasion techniques, we need the better approach to deal with such threats. The IDS devices need to be more accurate in defending these threats.

REFERENCES

[1] Chi-Chun Lo, Chun-Chieh Huang and Ku, J *"A Cooperative Intrusion Detection System Framework for Cloud Computing Networks"*.In 2010 39th International Conference on Parallel Processing Workshops

[2] Namjoshi,K.;Narlikar,G.; Bell Labs., Alcatel-Lucent, Paris, France,**"***Robust and Fast Pattern Matching for Intrusion Detection"*.In INFOCOM, 2010 Proceedings IEEE

[3] Raj Gaurav, Lovely Professional University, Phagwara, Punjab, India *"An Efficient Broker Cloud Management System"* in ACAI 2011, July 21–22, 2011, Rajpura, Punjab, India.

[4] Xiang Tan Bo Ai,"*Issues of cloud computing security in high speed railway"* State Key Laboratory of Rail Traffic Control and Safety Beijing Jiaotong University Beijing, P.R.China. In 2011 International Conference on Electronic & Mechanical Engineering and Information Technology

[5] Uma Somani, Kanika Lakhani, Manish Mundra, *Implementing Digital Signature with RSA Encryption Algorithm to Enhance the Data Security of Cloud in Cloud Computing* .In 2010 1st International Conference on Parallel, Distributed and Grid Computing (PDGC – 2010)

[6] Amarnath Jasti, Surya Mohapatra, Bhargav Potluri and Dr. Ravi Pendse Wichita State University, 1845 N Fairmount, Wichita, Kansas. *CLOUD COMPUTING IN AIRCRAFT DATANETWORK*. In 2011 Integrated Communications Navigation and Surveillance (ICNS) Conference May 10-12, 2011

[7] Farzad Sabahi Faculty of Computer Engineering Azad University Iran Cloud Computing *Security Threats and Responses*

[8] Xuan Zhang , Nattapong Wuwong, Hao Li and Xuejie Zhang, *"Information Security Risk Management Framework for the Cloud Computing Environments"*.In 2010 10th IEEE International Conference on Computer and Information Technology (CIT 2010)[

[9] Hassan Takabi, James B. D. Joshi and Gail-Joon Ahn, *"Secure,Cloud: Towards a Comprehensive Security Framework for Cloud Computing Environments"*.In 2010 34th Annual IEEE Computer Software and Applications Conference Workshops

[10] Vadym Mukhin, Artem Volokyta National Technical University of Ukraine "Kiev Polytechnic Institute", "*Security Risk Analysis for Cloud Computing Systems"*.In The 6th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications 15-17 September 2011, Prague, Czech Republic

[11] C*loud Security* by Ronald L. Krutz.pdf

Author

Gaurav Raj
M.Tech(SE) from MNNIT, Allahabad
e-mail ID- er.gaurav.raj@gmail.com
Affiliated with Lovely Professional University, Phagwara as Assistant Professor.
Research area: Cloud Computing & Wireless Network

Munish Katoch
M. Tech Student (CS)
e-mail ID- munishkatoch710@gmail.com
Lovely Professional University, Phagwara.
Research area: Cloud Computing & Wireless Network