

An Approach for Cloud Linux Server Security and Management

Madhumitha Ramamurthy¹, S.Manoj Kumar², Shanthi Palaniappan³

¹Associate Professor, Department of IT, Karpagam College of Engineering, Coimbatore, TamilNadu, India, madhuperu@gmail.com

²Professor, Department of IT, Karpagam College of Engineering, Coimbatore, TamilNadu, India. callsmk@gmail.com

³Assistant Professor, Department of MCA, Sri Krishna College of Engineering and Technology, Coimbatore, TamilNadu, India, shanthi.slm@gmail.com

Abstract. Cloud computing domain is a well developed technology that everyone uses now-a-days. Most of the users use cloud for computing servers, as virtual hard disk and for analytical purposes. Cloud servers make the users on same data simultaneously and from multiple places. This feature increases productivity in all sectors. IT sector need to manage many servers because most IT organizations have their website where they host the cloud instance for the employees to work. This paper proposes an approach to centrally manage the cloud server instances and also improves the security of cloud server instances. The security maintenance of a cloud server instance is necessary issue to be considered since there is a possibility of leaving vulnerability in the cloud server instance that leads to loss of critical information of an organization. Hence this paper proposes an approach for the maintenance of cloud server instances.

Keywords: Server, Agent, Cloud, Security.

1 Introduction

Cloud Computing [1] offers many services like storage of data, network, and compute which benefits the industries to reducing cost, increase production, performance and efficiency. Hence the evolution of Cloud Computing [2] made many organizations to make use of cloud servers and make their employees to work on the data stored on the cloud. This made possible to increase the productivity since the users can work simultaneously on the stored data and makes the users to be interconnected and access the data from any location. This is made possible since cloud computing uses virtualization technology. The various industries that makes use of cloud computing are automotive, banking, retail, education, healthcare, entertainment, IT industries, etc.

Security [3] is an important aspect to be considered with respect to cloud computing. This security issues are in both side the cloud providers and the cloud users. The cloud providers should ensure that they provide a secured infrastructure for the cloud users. The cloud users can be the organizations who host their applications and data on the cloud. The organizations which store their data on the cloud servers should ensure that the hosted data is managed in a secured manner.

Many IT organizations [4] [5] host their data in the cloud for the operations to be performed by their employees. Hence in these types of IT organizations, there will be a lot of servers to be managed and taken care of, because most IT organizations have their website and they host that in a cloud instance and their software in the cloud for their employees/users to access. Each IT organization maintains a team to manage the servers. These organizations store multiple users' data on the same server. As a result, there is a chance that one user's private data can be viewed by other users. If there is an issue in the server, the team must fix that issue and provide proper data isolation and logical storage segregation. The other teams should wait until the fixing those issues which will slow down the development process and it may lead to the loss of revenue for the organization. Also, it breaks the continuous development and continuous integration of the devops life cycle. The maintenance of a cloud server instance must be taken care or else there is a possibility of leaving vulnerability in the cloud server instance that leads to loss of critical information of an organization. Hence we propose a model for cloud server security and

management that will minimize these difficulties.

The rest of the paper is organized as follows. Section 2 outlines the related work. Section 3 describes the proposed architecture. Section 4 gives the experimental results and discussion. Section 6 gives the conclusion.

2 Related Work

Chef Automate by Chef Corporation is an automation tool which has the features like operational visibility, detecting the changes in real time and responding to them, audit configuration is made quickly along with compliance history, servers scanning and integrating with Chef's open source projects [6][15].

IUEM by Ivanti can be used to manage on-premise machines that are available in a network and it supports all operating system. It provides a real-time view of all changes either user initiated changes or machine-initiated changes which are happening in the environment. It also notifies any failures which happen by chat or webhook. The lack of support for cloud based instances is a great disadvantage. So that this tool cannot be used by enterprises and by individuals who has the work environment in cloud [7][14].

The security of data in cloud computing is discussed in [8][12][13]. This paper proposes methods for data protection. This paper also provides an insight on data security. The security aspects considered in this paper are data-in-transit and data-at-rest.

Cloud environment which provides a powerful computing platform is discussed in [9] and also this discusses the security vulnerabilities in Linux Server. Also, this paper compares the cloud service models with respect to cloud security risks and countermeasures for breaches were given in the paper.

Enhanced techniques for cloud security are discussed in [10][16], where the security issues are addressed. The security issues include data protection, network security, virtualization security, application integrity, and identity management. This paper analyzes and evaluates the most important security techniques for data protection in cloud computing. Also it recommends the data data protection techniques to security in cloud computing.

The era of cloud computing technology used in IT Industries was discussed in [11] and provides the growth of the IT industries before and after the cloud computing.

3 Proposed Work

The proposed work consists of two daemons that run continuously in the cloud instances, one will be running in a central cloud server instance from which we want to manage all the cloud server instance and another one will running in all cloud server instance that we want to manage. The main cloud server instance from which we manage is called as a server module and the cloud server instances that we manage are called as agent modules. The server has a console that shows all the agents under it. From the console, requests can be sent to all the agents like installing security fixes, changing ip rules etc. The agents in the rest of the servers wait for any request from the server by listening to a port.

If the agent receives any request, it does the job and sends a status to the server. The server then takes the response status and updates the console. The proposed approach saves a lot of time since since all the servers can be managed from a single console and that console can be accessed from any machine.

The console can be used to set firewall rules to control internet activity, enable or disable antivirus to protect from malicious program. The console also provides a view to see the missing application security fixes and option to select and apply those security fixes, a view to see all open ports in servers, since open ports can be accessed by hackers to steal your organizations data. The console also allows creating, editing and managing all the cron jobs in a server.

An automated patch management solution should meet a few key requirements, including: Continuous endpoint monitoring, A network-neutral architecture, Robust testing, Reporting capabilities. If the agent receives any request, it does the job and sends a status to the server. The

server then takes the response status and updates the console. The server has a console that shows all the agents under our server. From the console, you can send requests to all the agents like installing security fixes, changing ip rules etc. The agents in the rest of the servers wait for any request from the server by listening to a port. One of the greatest strengths of Patch Manager Plus Cloud is that it ensures that there are no security vulnerabilities in the network. It has Privacy Settings which you can use to get hold of the Enterprise's data privacy and security.

2.1 Product Architecture

The product architecture of the proposed approach is given in the Fig1. The server module in the Figure.1 is the main cloud instance from which the agent module cloud instances are managed. The server module has a console to manage the rest of the agents in the cloud. The server module can be installed in a cloud instance where system administrator is comfortable in managing the entire cloud instances. The arrows in the architecture diagram represent the communication between the server and agent. Here communication is both the ways, since server need to send commands to the agent instance and the agent cloud instance need to send the status for the corresponding commands to the server cloud instance.

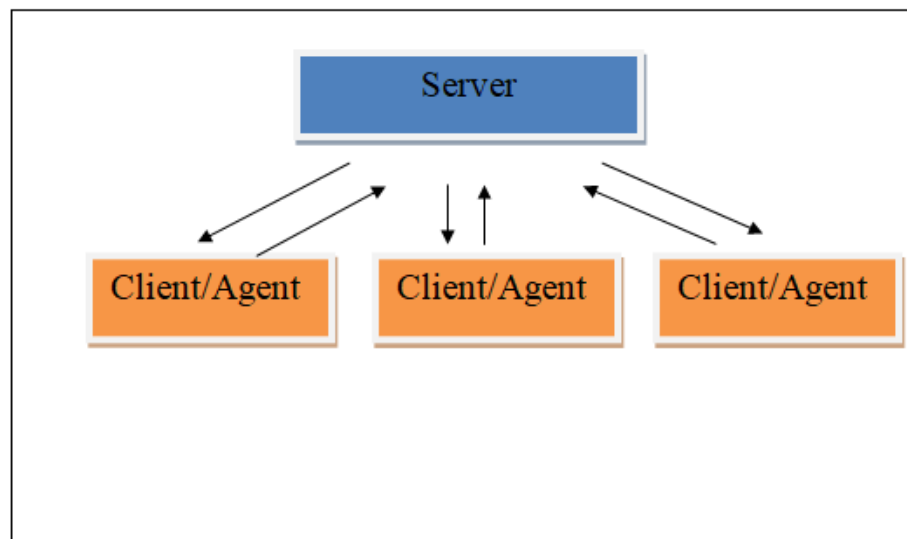


Fig. 1. Product Architecture

The server module and agent module is installed on the cloud instances manually, then both the server and agent module will take care of managing activities. The server module creates a configuration file which tells the agent what to do. The agent module gets the server and performs the configuration and sends the status to the server module. The main job of the agent module is to send response for the server module commands. In agent module, the service will be started automatically as soon as the cloud instance starts and it listens to a port to get the command from the server.

The server module uses this result of the config and process the config. The result of the config is stored in a file if it is a scan config. The agent module has a golang written ever-running process (daemon) which will be started automatically as soon as the agent cloud instance started and it listens to a port to get the task or config from the server.

The service that runs in the agent module sends a notification to the server for every 10 minutes to notify that the agent module is active or not, so that the server module knows that the agent is ready to process the server's task or config. After getting any config from the server cloud instance, it calls a corresponding executable to do the requested task or config and send a status of the task or config to the server. After getting any request from the server, it calls a corresponding executable to do the requested task and send a status of what is the result of the task to the server. The executable is a golang written code compiled and converted into an executable. The agent

module has every function that needs to be done from the console. The console manages the rest of the agents in the cloud.

The server sends the task and gets the status update from the agent. This also gives a console as a user interface for the user to manage the agents under that server module. The server module contacts the agent using the ip address and port combination, then it sends whatever the request to that agent. The server module always listens on a port for the incoming status from the agents.

The agent sends the task and gets the status update from the agent. The Apache web server also gives a console as a user interface for the user to manage the agents under that server module. A server manages all the processes and also stores all data. The client makes request to the server for a specified data or processes. The server in turn relays the output to the client. The processing is performed in the client side, but requires server data resources for completion.

The cloud instances in the cloud are periodically scanned for latest scan data and they are stored in a file. A comprehensive scanning mechanism checks for the existence of the cloud instance and also checks for the state of the cloud instance by performing file version checks and checksum. The vulnerability data of the cloud instance is periodically updated with the latest information from the cloud instance. The agent module has a golang written ever-running process (daemon) which will be started automatically as soon as the agent cloud instance started and it listens to a port to get the task or config from the server. The service that runs in the agent module sends a notification to the server for every 10 minutes to notify that the agent module is active or not, so that the server module knows that the agent is ready to process the server's task or config.

4 Results and Discussion

The server module has a golang written ever-running server. It gets started after the cloud instance is started. It has no GUI to control the agents, since sysadmins mostly use Command Line Interface to operate a cloud instance, so need to use CLI to control the server module and it's also not complicated to use the server module. Now there are options to list, scan agent cloud instances and apply firewall rules to agent cloud instances. The server module saves a config file for each of the agent module. The agent module gets this config and applies this config and sends the result of the config to the server module. The agent module is installed on the servers manually, then the server will take care of managing the rest of the security activities. In the agent module installed servers, there will be no user interface for the user to use.

The server module uses this result of the config and process the config. The result of the config is stored in a file if it is a scan config. The agent module has a golang written ever-running process (daemon) which will be started automatically as soon as the agent cloud instance started and it listens to a port to get the task or config from the server. The service that runs in the agent module sends a notification to the server for every 10 minutes to notify that the agent module is active or not, so that the server module knows that the agent is ready to process the server's task or config. After getting any config from the server cloud instance, it calls a corresponding executable to do the requested task or config and send a status of the task or config to the server.

The cloud instances in the cloud are periodically scanned for latest scan data and they are stored in a file. A comprehensive scanning mechanism checks for the existence of the cloud instance and state of the cloud instance by performing file version checks and checksums. The vulnerability data of the cloud instance is periodically updated with the latest information from the cloud instance. The scanning logic automatically determines which updates are needed on each client system. This is determined by considering the operating system, application, and update dependencies. After scanning is completed successfully, the results of each scanning are returned and stored in the server module file. The results of the scanning results can be viewed from the console.

The client-server model is a core network computing concept. This can also be used for email exchange and Web/database access. A server manages most processes and stores all data. The agent module is installed on the servers manually and then the server will take care of managing the rest of the security activities. In the agent module installed servers, there will be no user

interface for the user to use. Based on the client requests, server relays requested data or processes to the client.

In short the modules in our product works like master and slave model. The agent (slave) installed in the servers will do whatever the server (master) says. The server module has all the view to control the agents. The server module has a front end as the web console. The server module has a Apache web server from which the console will be displayed. The client-server model is a distributed communication framework of network processes among service requests, clients and service providers. The client-server connection is established through a network or the Internet.

Firewall rules are present in config and the config file is given by the server to the agent module. The agent module after downloading the config file, the firewall rules are applied to the machine using the command `nft -f file_path` to apply the firewall rules in the machine. The machine then applies this firewall rules in kernel level. In nftables the rules are flushed after a reboot so every time the instance booted the firewall rules are applied again by the agent.

```

    ▶ GOPATH=/usr/local/go #gosetup
    █ GOPATH=/home/thebatvic/MyWorks/CloudServerManagement/CSM_Utils:/home/thebatvic/go #gosetup
    █ /usr/local/go/bin/go build -i -o /tmp/___go_build_CsmServerConfigMain_go /home/thebatvic/MyWorks/CL
    █ /tmp/___go_build_CsmServerConfigMain_go scan 232 #gosetup
    █ Started Scanning....
    █
    █ Applied Config Status : Agent ID : 232 Status : Scan completed and result written to /home/thebatvic/
    █
    █ Process finished with exit code 0
    █
    
```

Fig. 2.Server Scanning

The Fig.2 shows server scanning process and the result is stored in a file to be viewed by the system administrator.

```

    ~~~~~ Start of Scan Result ~~~~~

    Last Successful Scan Time : 2020-03-09 23:00:17.451518211 +0530 IST m++107.884571637

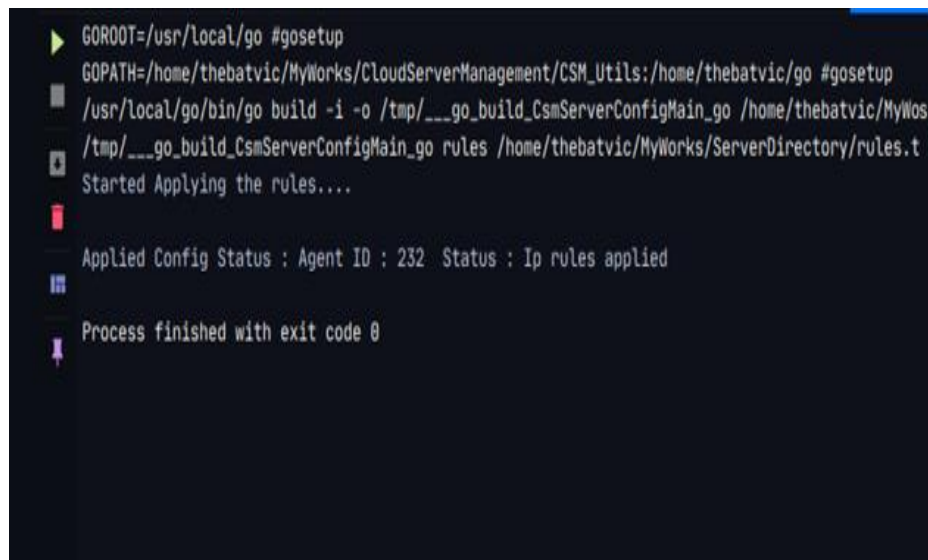
    AGENT DETAILS
    -----
    AGENT ID      AGENT HOST NAME  AGENT IP ADDRESS  AGENT MAC ADDRESS  SELINUX/APPARMOR ST
    232           BatMachine       192.168.43.96     b4:6b:fc:32:87:8c  SELINUX package is

    OPEN PORT DETAILS
    -----
    PORT    TYPE    PROCESS_NAME    PROCESS_ID    PROCESS_PATH
    53      tcp     systemd-resolve  788           /lib/systemd/systemd-resolved
    631     tcp     cupsd           886           /usr/sbin/cupsd
    63342   tcp     java            8768          /home/thebatvic/.local/share/JetBrains/Tool
    6942    tcp     java            8768          /home/thebatvic/.local/share/JetBrains/Tool
    8022    tcp     ___go_build_Csm  9076          /tmp/___go_build_CsmServerServiceMain_go

    RUNNING SERVICE DETAILS
    
```

Fig.3 Console showing agent details

The Fig.3 shows the details of the agent service, open port details, security packages to be installed.



```

GOROOT=/usr/local/go #gosetup
GOPATH=/home/thebatvic/MyWorks/CloudServerManagement/CSM_Utils:/home/thebatvic/go #gosetup
/usr/local/go/bin/go build -i -o /tmp/___go_build_CsmServerConfigMain_go /home/thebatvic/MyWorks/CloudServerManagement/CSM_Utils/___go_build_CsmServerConfigMain_go/rules /home/thebatvic/MyWorks/ServerDirectory/rules.t
Started Applying the rules....

Applied Config Status : Agent ID : 232 Status : Ip rules applied

Process finished with exit code 0
    
```

Fig.4 Firewall Config rules

The Fig.4 shows the firewall config rules from the central server instance to the agent module. Thus the problem of managing a lot of servers is reduced by the proposed approach.

5 Conclusion

The use of cloud computing in many sectors imposes security risks to the data stored in the cloud. Especially, IT sector need to manage many servers because most IT organizations have their website where they host the cloud instance for the employees to work. The proposed approach reduces the difficulty in managing a lot of servers. The efficiency in continuous integration and continuous deployment is increased due to the managing all the servers from a console and even that console can be accessed from anywhere. This project can be improved more like managing all the things of a server like installing any application from the console. Thus the proposed approach helps in managing the security of the servers, since security of a server is important for an organization.

References

1. Mohsen, Attaran. Cloud Computing Technology: Leveraging the Power of the Internet to Improve Business Performance. Int. J. of Information Technology and Management (2017).
2. Mohsin, Nazir. Cloud Computing: Overview & Current Research Challenges. IOSR. J. of Computer Engineering (2012).
3. Keiko, Hashizume., David, G, Rosado., Eduardo, Fernández-Medina., Eduardo, B, Fernandez. An analysis of security issues for cloud computing”, J.of.Internet Services and Applications (2013).
4. Indu, I., Rubesh, Anand, P,M., Vidhyacharan, Bhaskar. Identity and access management in cloud environment: Mechanisms and challenges. Engineering Science and Technology, an International Journal (2018) 574–588.
5. Qiang, Duan. Cloud service performance evaluation: status, challenges, and opportunities – a survey from the system modeling perspective. Digital Communications and Networks (2017) 101–111.
6. Chef Automate Architecture: <https://automate.chef.io/>.
7. Ivanti Unified Endpoint Management : <https://www.ivanti.com/products/unified-endpoint-manager>
8. Ahmed, Albugmi., Madini, O, Alassafi., Robert, John, Walters., Gary, Wills.:Data Security in Cloud Computing. Fifth International Conference on Future Generation Communication Technology (2016).

9. Nadiah, M, Almutairy., Khalil, H, A, Al-Shqeerat. A Survey on Security Challenges of Virtualization Technology in Cloud Computing. *Int.J.of Computer Science & Information Technology* (2019).
10. Eman, Meslhy., Hatem, Ahmed, Abd, Elkader., Sherif, Eletriby. : Enhanced data security model for cloud computing. *8th International Conference on INFormatics and Systems* (2012).
11. Priyanshu, Srivastava., Rizwan, Khan. A Review Paper on Cloud Computing. *Int. J.of Advanced Research in Computer Science and Software Engineering* (2018).
12. Ponmagal, R.S., Karthick, S., Dhiyanesh, B. et al. Optimized virtual network function provisioning technique for mobile edge cloud computing. *J Ambient Intell Human Comput* (2020).
13. Ramamoorthy, S., Ravikumar, G., Saravana Balaji, B. et al. MCAMO: multi constraint aware multi-objective resource scheduling optimization technique for cloud infrastructure services. *J Ambient Intell Human Comput* (2020).
14. Basha, A.J., Balaji, B.S., Poornima, S. et al. Support vector machine and simple recurrent network based automatic sleep stage classification of fuzzy kernel. *J Ambient Intell Human Comput* (2020)
15. Balaji, B.S., Balakrishnan, S., Venkatachalam, K. et al. Automated query classification-based web service similarity technique using machine learning. *J Ambient Intell Human Comput* (2020)
16. Viji, C., Rajkumar, N., Suganthi, S.T. et al. An improved approach for automatic spine canal segmentation using probabilistic boosting tree (PBT) with fuzzy support vector machine. *J Ambient Intell Human Comput* (2020).