Distributed Key Management for IT Infrastructure using Block Chain and Hash Graph

Sathya Priya S, Revathy S, Kamnag R, Yogeswar L, Sajal M, Suparna S

ABSTRACT--- The Cloud Computing Paradigm simplified the IT infrastructure management thereby making the organiza-tions and enterprises to rapidly migrate from conventional model to service driven model. The service driven model allows the users to access, maintain and backup the files from remote location over the internet through the central-ized cloud service providers. Inspite of major advantages, there are some concerns for the safety and privacy risks of organizational and users sensitive information in cloud stor-age. The existing service oriented model uses distributed key management which encrypts the data and stores it in cloud servers to provide the authentication to data, but it addresses challenges in identifying ownership of data, boot-strapping problem and securing the keys. The proposed system uses block chain, a distributed ledger for user authentication in cloud server and Hash graph with asynchronous byzantine fault tolerance algorithm is proposed for replicat-ed state machines with guaranteed Byzantine fault-tolerance and it can be utilized in the IT Infrastructure envi-ronment which solves the hassles of maintaining security keys.

Key words — Block Chain, Cloud Environment, Distributed key management, Hash-Graph, Security.

I. INTRODUCTION

The global spending on providing security to underlying IT Infrastructure is growing in an exponential rate. More enterprises and organizations are rapidly migrating towards service driven model for computations. But still, many Information officers hesitate to fully embrace the power of cloud. Their uncertainty stems mainly from the anxiety over the security of their systems and data. Data can be stored in a remote location that can be typically in an encrypted form and backed by strong firewall. So basic encryption solves the problem of stolen data to be made unusable. But the dark cyber world uses algorithms that are more advanced which makes the existing security techniques look outdated[14]. So security is compromised easily in this existing setup. Enterprise expects a scalable and stable hybrid solution while taking the necessary steps to secure their data across their enterprise. Developers when working on a particular project must show a keen concentration only in that project alone and don't need to concentrate on network packet security. Taking security concerns dynamically into consideration is good. But, if the whole work is dedicated to security and if efficiency and algorithmic quality of the end product goes down then threshold of QOS will be compromised which in turn downgrades the enterprise.

Independent service vendors (ISV) and Cloud platform service providers rapidly build and deploy rich applications, or extend their existing service with great risk of leaking or loss of data in spite of having good backup and tight security systems. In current scenario, breach in the network happens mostly by attacking the intrusion that sits in the core switch of the enterprise and monitors the network traffic for miscellaneous activities. Most of the times the attack will be successful because if core switch is under attack by dark cyber net, all the nodes in the network is also under attack indirectly in spite of having individual firewalls and gateways for each node. The entire hierarchy of the network goes into control of the dark cyber that too without the knowledge of network administrator because nodes in lower hierarchy inherit credentials from root node. This is the most common way of attacking a traditional network. To tackle the above mentioned problem, logical virtual images were introduced for network security where the switch configuration will be reverted using the created logically equivalent image in a timely basis. But if the configuration is replaced by attackers' credentials at the time of restoration, the entire network along with the infrastructure goes into the control of the attacker[12,13].

A lot of machine learning algorithms are being used for network security purpose, the machine learning patterns are currently being impinged with hard damaging patches that act more of like training biases which in turn damage the entire neural net, damaging the entire neural net. But distributed key management using hash graph consensus based model can solve the above listed problems to the core. When firewall protection for the underlying infrastructure is applied, dynamically if the configuration and all the security keys associated with the system is distributed and replicated among the virtual infrastructures and other nodes, damage caused by the attack can be reduced to extremely greater extent. So if same logic as blockchain is applied to secure a network in a distributed manner then it becomes

Revised Manuscript Received on July 10, 2019.

Sathya Priya, Associate Professor, Department of Computer Science and Engineering, Hindustan Institute of technology and science, Chennai, T.N, India. (E-mail: sathyap@hindustanuniv.ac.in)

S, Revathy S, Assistant Professor, Department of Computer Science and Engineering, Hindustan Institute of technology and science, Chennai, T.N, India. (E-mail: srevathy@hindustanuniv.ac.in)

Kamnag R, Department of Computer Science and Engineering, Hindustan Institute of technology and science, Chennai, T.N, India. (E-mail: rkamnag@gmail.com)

Yogeswar L, Department of Computer Science and Engineering, Hindustan Institute of technology and science, Chennai, T.N, India. (E-mail: yogi2198@gmail.com)

Sajal M, Department of Computer Science and Engineering, Hindustan Institute of technology and science, Chennai, T.N, India. (E-mail: mizzsam9@gmail.com)

Suparna S, Department of Computer Science and Engineering, Hindustan Institute of technology and science, Chennai, T.N, India. (E-mail: ssuparna@gmail.com)

inconceivable for cyberpunks to take the enterprise network into their hand. This way of securing the infrastructure indirectly also provides an efficient way of data or computation retrieval and works well for a low bandwidth networks which acts as the biggest bottleneck in the industry.

Securing the network infrastructure alone (considering other security measures are already up) will not solve the security threats as a whole. Modern tools, systems and appliance should also be secured in a way where it is possible to avoid threats and vulnerability in order to avoid the risk of failure towards the organization. This is where distributed security comes into the picture. Threat assessment, an important part of risk analysis plays a significant role in security management efforts. There is an increase in demand for a model and tool to support the assessment of risk management in security. The main ideology of this study is to answer a basic question: "Can distributed ledger system or hash graph can solve the age old problem of key and other network asset management?" The research will focus on distributed security system using hash graph Consensus based approach for IT infrastructure security appliances and systems assessment in Information technology organizations.

Using Hash Graph Consensus for access key management when compared with the solution for the existing public or private key management system which is purely dependent on the encryption methodologies where way it gets stored is forgotten about. Distributed ledger technology can be taken into consideration so that security can be provided from ground level i.e., who access the key and where it is being used and stored. This particular platform's model of governance and the way of implementation so far have been polarizing in the crypto currency world, which is obvious. On a generic note, the hash graph will consist of all the hashes of keys stored in various file formats and encryption methodologies backed by consensus methodology currently available in many of the blockchain networks such as Ethereum.

This paper is organized as follows. Section II covers the Existing System architecture Section III describes the Proposed system architecture. Section IV covers the Implementation and finally conclusion in Section V.

II. EXISTING SYSTEM HASH GRAPH CONSENSUS FOR KEY MANAGEMENT

In the existing data-center setup, firewall and encryption acts as the core security for the entire setup. But in current scenario where there are a lot of computation being involved, traditional firewalls can be breached easily. In traditional IT setup, even the end clients are supposed to take a lot of measures so that the system admins would not be in a position to access the end client data. The existing firewall system provides security solution in form of zones. The firewall may be a physical hardware or a virtualized hardware where the performance is evaluated on the basis of how much data that particular firewall can handle in a given amount of time. The firewalls are setup in a hierarchical manner where the core switch will be connected to a firewall with high throughput and the end nodes will usually be virtualized. These Virtualized firewall infrastructures are used when dynamic Scalability is to be incorporated. When architectural implementation is taken into consideration, It is so rigid such that alteration of one particular node disrupts almost the whole architecture. Performance drastically gets reduced when lot of virtualized hardware comes into existence in the hierarchy, eventually reducing the throughput. Performance also gets reduced because individual hardware performance must be taken into consideration. The network traffic management also becomes a hectic job when plug and play firewalls are being used. A mesh topology can be taken into consideration to solve this issue, but the cost of the setup will be very high. If a new application is to be made a part of the setup, the admin must have a prior knowledge of the middle-ware and deployment configuration.

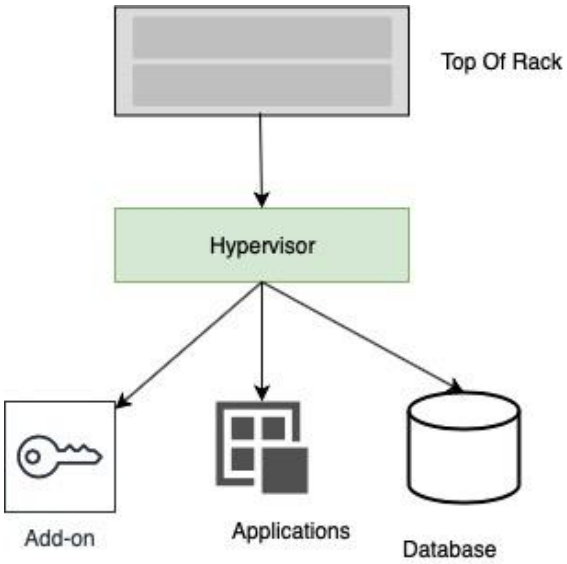

Fig.1 Traditional IT Hierarchy

Fig.1 depicts the traditional configuration. All the above mentioned short comings can be overcome using distributed security system with a little bit of hash graph configured hypervisor that can be flashed into the core switch of the enterprise. Now the hypervisor will be configured in such a way that an enforcement point that sits just above the VM which runs on hypervisor. Then all the applications can be tagged along to the enforcement point where key management stack acts as middleware. Since all the applications are indirectly linked to the enforcement point, there is no difficulty in steering the request response traffic that are intended for the applications.

Enforcement point is fully automated centralized or decentralized access-point where manual routing, traffic control, etc can be avoided. All the workloads (i.e applications) that are attached to enforcement point are micro-segmented and are isolated from other applications which provide high security and each workload will be dynamically assigned with policy loads only through which packets can be sent. The biggest question that prevails among many people is whether they will be able to integrate this type of setup with the traditional database or other

applications. Since all the micro-segments are integrated with technical policy agreements, upper and lower compatibility will be very easy taken care of. Here basically all the configurations are made in the form of policies. The enforcement point will contain the gateway which are in a distributed manner and the policy will be dynamically mirrored to the gateway. This way of dynamic mirroring of the policy also lets the Enforcement Point to manage the traffic distribution. By doing so, the choke point that frequently tampers firewall security is totally ignored which acts as a biggest bottleneck in the data-center outside world communication. Fig.2 depicts the analyzed, re-modified setup with enforcement point tagged along with key management stack.

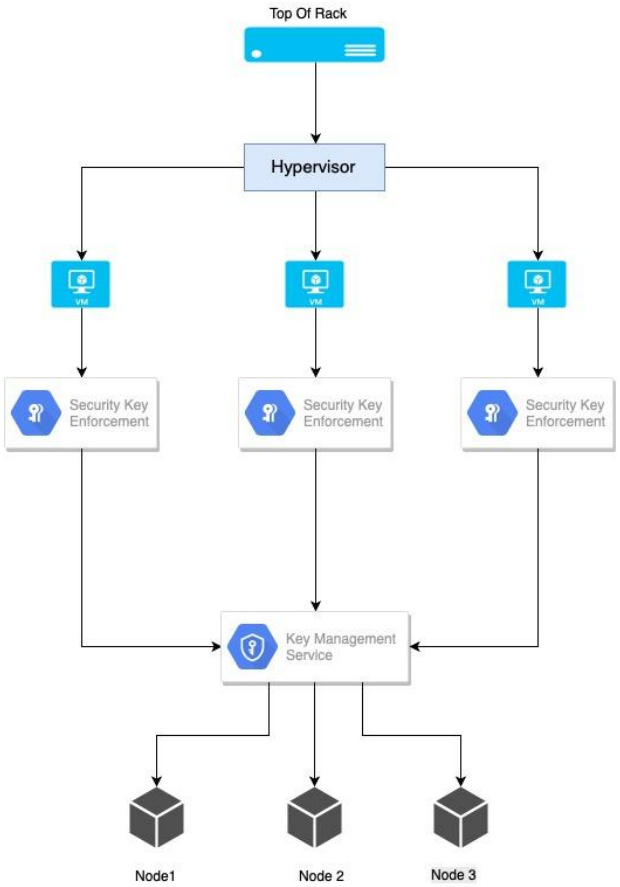

Fig.2 Enforcement Point mapped implementation

Key Management

The security of any crypto system depends upon how securely its keys are managed. The current key management goes by the below mentioned hard rule.

- Key generation
- Key registration
- Key storage
- Key distribution and installation
- Key use
- Key rotation
- Key backup
- Key recovery
- Key revocation
- Key suspension
- Key destruction

III. PROPOSED SYSTEM

The proposed system will tap into the key storage part where many loopholes can easily be found. The storage paths will be hashed and be available in hash graph and distributed along nodes.

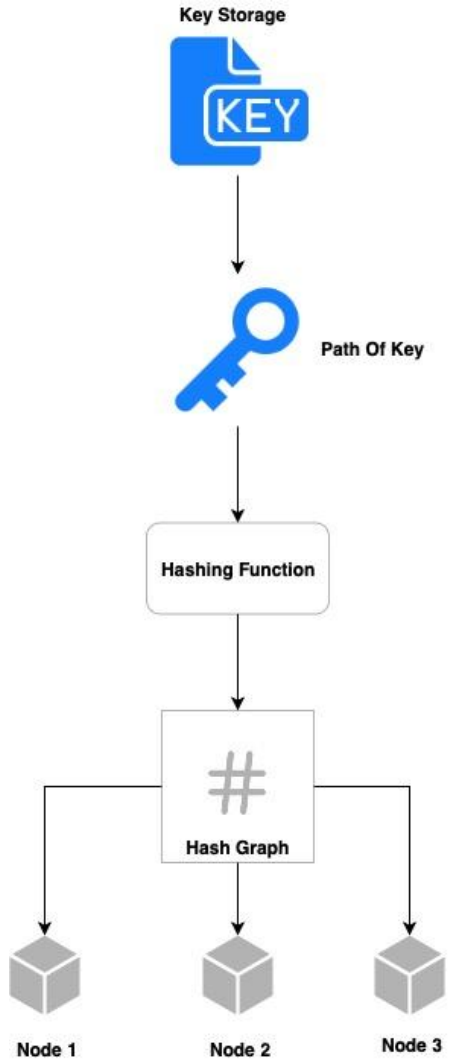

Fig. 3 Hashing workflow

Fig.3 depicts how proposed key management system works.

The Key management system is a set of operations that are to create, maintain, protect and control the use of cryptographic keys. These Keys follow a life cycle. The key Management Interoperability Protocol (KMIP) enables communication between key management systems and cryptographically - enabled application. These keys are stored in Hardware security modules (HSM) which are physical computing device that protects the digitals keys by using strong authentication and provides crypto processing.

Hardware security modules protects the DNS from “Cache Poisoning” and “Man-in-the-middle” attacks.HSMs provide auditable security advantages, enabling proper generation and storage for signing digital keys to assure integrity.

IV. IMPLEMENTATION KEY PATH HASHING ALGORITHM & RESULTS

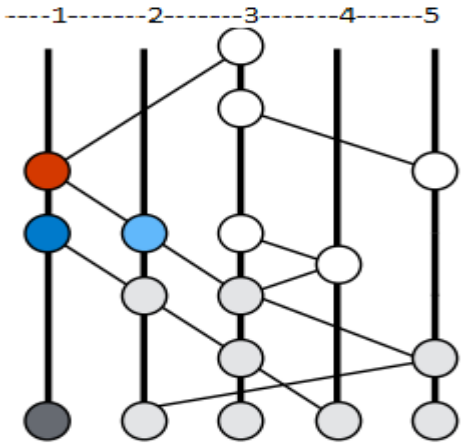

Fig.4 Path Hashing

'1' makes a key way occasion (red) recording the event of '2' completing a talk adjust to '1' and adding everything to '1' that '2' has. The occasion contains a hash of two parent key way occasions (blue): oneself parent (dull blue) by the equivalent creator '1', and the other-parent (light blue) by '2'. It likewise contains a payload of any new exchanges that '1' makes right then and there, and an advanced mark by '1'. The other predecessor occasions (dim) are not put away in the red occasion, but rather they are dictated by every one of the hashes. The other self-predecessors (dim dark) are those reachable by groupings of self-parent joins, and the others (light dim) are most certainly not.

Each part has a duplicate of the hash graph. In the event that '1' and '2' both have a similar hash graph, they can figure a complete request on the occasions as indicated by any deterministic capacity of that hash chart, and they will both find a similar solution. Along these lines, agreement is accomplished, even without sending vote messages. Obviously, '1' and '2' might not have the very same hash graph at some random minute. They will commonly coordinate in the more seasoned occasions. In any case, for the simple late occasions, each may have occasions that alternate has not yet observed. Besides, there may intermittently be another occasion discharged to the network that ought to be put in a lower (before) area in the hash chart. The hash chart accord calculation manages these issue utilizing a framework that is best idea of as virtual casting a ballot. Assume '1' has hash chart An and '2' hash graph B. These hash graphs might be somewhat unique at some random moment, however they will dependably be reliable. Steady implies that on the off chance that An and B both contain occasion x, they will both contain the very same arrangement of progenitors for x, and will both contain the very same arrangement of edges between those predecessors. On the off chance that '1' is aware of x and '2' does not, and the two are straightforward and effectively taking part, at that point we would anticipate '2' to learn of x decently fast, through the talk convention. The accord calculation accept that will hap-pen in the long run, yet does not make any suspicions about how quick it will occur. The convention is totally asynchronous, and does not make presumptions about timeout periods, or the speed of prattle, or the rate at which advance is made. '1' will compute an all-out request on the occasions in A by figuring a progression of decisions. In every race, a portion of the occasions in A will be considered to make a choice, and a portion of the occasions in A will be considered to get that vote. '1' figure numerous races, and a given occasion may take an interest in a few decisions however not others, and might cast diverse votes in various races. On the off chance that the occasion was made by '2', we will discuss '2' casting a ballot a specific path in a given race. Be that as it may, the genuine part '2' isn't involved. This is absolutely an estimation that '1' is performing locally, where she is computing what vote '2' would have sent her, if the genuine '2' were really sending votes over the web to her. This virtual casting a ballot has a few advantages. Notwithstanding sparing transfer speed, it guarantees that individuals al-ways compute their votes as indicated by the tenets. In the event that '1' is straightforward, she will ascertain virtual votes in favor of the virtual '2' that are straightforward. Regardless of whether the genuine '2' is a con artist, he can't assault '1' by making the virtual '2' vote mistakenly. '2' can attempt to cheat in an unexpected way. Assume '2' makes an occasion x with a specific self-parent hash indicating his previous occasion z. At that point '2' makes another occasion y, however gives it a self-parent hash of z, rather than giving it a self-parent hash of x as he should. This implies the occasions by '2' in the hash chart will never again be a chain, as they ought to be. They will currently be a tree, since he has made a fork. On the off chance that '2' tattles x to '1' and y to Carol, for some time, '1' and Carol may not know about the fork. Also, '1' may figure a virtual vote in favor of x that is not the same as Carol's virtual vote in favor of y. So it is feasible for a fork to be spread crosswise over steady hash graphs. For this situation, there might be a moment when '1' has a hash graph containing x however not y, and Carol has a hash chart with y and not x, thus a fork exists, but rather neither one of the members is yet mindful of the way that it is a fork. The hash graph accord calculation keeps this assault by utilizing the idea of one state seeing another, and the idea of one state unequivocally observing another. These depend on meanings of precursor and self-progenitor to such an extent that each occasion is viewed as both a predecessor and self-progenitor of itself. On the off chance that '2' makes two occasions x and y, neither of which is a self-precursor of the other, at that point '2' has bamboozled by forking. In the event that some occasion w has x as a progenitor however doesn't have y as a predecessor, at that point the occasion w can see occasion x. In any case, in the event that both x and y are predecessors of w, w is characterized to not see both of them, nor some other occasion by a similar maker. At the end of the day, w can see x if x is known to it, and no forks by that maker are known to it. On the off chance that there are n individuals (n > 1), an occasion w can unequivocally observe an occasion x, if w can see more than 2n/3 occasions by different individuals, every one of which can see x. This idea is delineated in Figure 3. Four duplicates of a similar hash graph are appeared, with an alternate occasion

on the base line shaded orange. In (d), the yellow occasion at the best can see 4 red occasions by various individuals, every one of which can see the orange occasion at the base. This is additionally valid in (a), (b), and (c), with (a) really having 5 red occasions. Be that as it may, just 4 are required for emphatically observing, in light of the fact that this model has n = 5 individuals, and the least number more prominent than 2n/3 is 4. This idea enables an understanding convention to accomplish Byzan-tine adaptation to non-critical failure with no genuine casting a ballot, simply through neighborhood virtual casting a ballot. In virtual casting a ballot, when occasion x cast a ballot on some YES/NO inquiry (e.g., regardless of whether some other occasion is popular), the vote is determined absolutely as a component of the precursors of x. That vote is just viewed as sent from x to its relative occasion w if w can firmly observe x. It is demonstrated in area 5 that on the off chance that x and y are on various parts of an unlawful fork, w can firmly observe at most one of x and y, however not both. Moreover, in the event that hash charts An and B are predictable, it isn't workable for one occasion to emphatically observe x in An and another occasion firmly observe y in B.

V. CONCLUSION

The hash graph and distributed system in security systems taking over all traditional databases is an overwhelming sight of improvement in the cyber industry. This paper is an attempt to review the capabilities of hash graph and other related methods for implementing security systems. As there are a wide variety of threads and keywords, it is important to cluster the security threads and nodes for faster and secure access of data and infrastructure. The proposed system can be implemented for any infrastructure configuration and traditional databases for harnessing the complete power of the proposed implementation for security system for IT Infrastructure.

REFERENCES

1. Licheng Wanga, Xiaoying Shen a, Jing Li b, Jun Shaoc, Yixian Yang a"Cryptographic primitives in blockchains" Journal of Network and Computer Applications 127 (2019) 43–58.
2. K. Huang *et al*., "Building Redactable Consortium Blockchain for Industrial Internet-of-Things," in *IEEE Transactions on Industrial Informatics*. doi: 10.1109/TII.2019.2901011
3. K. Fan, Y. Ren, Y. Wang, H. Li and Y. Yang, "Blockchain-based efficient privacy preserving and data sharing scheme of content-centric network in 5G," in *IET Communications*, vol. 12, no. 5, pp. 527-532, 20 3 2018. doi: 10.1049/iet-com.2017.0619
4. Ana Reyna *, Cristian Martín, Jaime Chen, Enrique Soler, Manuel Díaz" On blockchain and its integration with IoT. Challenges and opportunities, Future Generation Computer Systems 88 (2018) 173–190.
5. Ting Yang1, Feng Zhai1,2, Jialin Liu1, MengWang1 and Haibo Pen1 Self-organized cyber physical power system blockchain architecture and protocol, International Journal of Distributed Sensor Networks,2018, Vol. 14(10)
6. M. Díaz, C. Martín, B. Rubio, State-of-the-art, challenges, and open issues in the integration of internet of things and cloud computing, J. Netw. Comput.Appl. 67 (2016) 99–117.
7. Understanding Autonomous Organizations on the Block chainhttps://www.linkedin.com/pulse/understanding-autonomous organizations-blockchain-paul-kohlhaas
8. F. Brezo and P. Bringas, ``Issues and risks associated with cryptocurren-cies such as bitcoin," in *Proc. 2nd Int. Conf. Soc. Eco-Informat.*, 2012,pp. 20_26
9. Bonneau, J., Miller, A., Clark, J., Narayanan, A., Kroll, J.A., Felten, E.W. Sok research perspectives and challenges for Bitcoin and cryptocurrencies. In 2015 IEEE Symposium on Security and Privacy, SP 2015, pp. 104–121 (2015)
10. Salim Sarimurat ; Albert HaG: Hash graph based key predistribution scheme for multiphase wireless sensor networks IEEE(2013).
11. M. Castro and B. Liskov, ``Practical byzantine fault tolerance and proac-tive recovery," *ACM Trans. Comput. Syst.*, vol. 20, no. 4, pp. 398_461,Nov. 2002.
12. Karthik, A., Joseph, A., Sethuraman, R. "Creation of a mapping system to generate objects similar to bit-coins" 2016 ARPN Journal of Engineering and Applied Sciences.
13. Kulkarni R V, Patil S H, Subhashini R "An overview of learning in data streams with label scarcity, " Proceedings of the International Conference on Inventive Computation Technologies, ICICT 2016, 2017.
14. Vinothan, D.,Saravanan, M.," Institution System analysis by using similarity based clustering on social network access ",Pakistan Journal of Biotechnology. Vol 13,2016,pp 1-4.