Grenze International Journal of Engineering and Technology, June Issue

Innovative Approach for De-Centralization of Cloud Servers

Deepika Gautam[1] and Vipin Saxena[2]
[1-2]Department of Computer Science, Babasaheb Bhimrao Ambedkar University Lucknow, Uttar Pradesh, India.
Email: deepika.gautam.lko@gmail.com, profvipinsaxena@gmail.com

***Abstract*— A novel secure architecture for decentralized cloud servers is proposed that integrates robust cryptographic mechanisms directly into its design to ensure confidentiality, integrity and authentication throughout the network. The proposed system employs a hierarchical Reverse Cone Topology (RCT) that incorporates asymmetric key exchanges and symmetric encryption with digital signatures at every communication link. The RCT is a single trusted control node at the base and progressively increasing numbers of servers in the upper layers. Through extensive Python-based simulation, RCT demonstrates superior performance with ultra-low latency, high throughput, minimal packet loss, and effective attack detection, outperforming traditional linear, tree and hybrid topologies. The integrated security and scalable topology present a future cloud, edge and IoT environment, with promising applications in real-time and industrial cloud systems.**

***Index Terms*— Topology optimization, hierarchical server architecture, decentralized cloud, cryptography.**

I. INTRODUCTION

Cloud computing has become an essential backbone for modern digital services, offering scalability, flexibility and cost efficiency for a wide range of applications, from business analytics to critical infrastructure systems. The performance of cloud services is the design of server interconnection topologies, which directly affect parameters such as latency, throughput, resource utilization and resilience to failures. Traditional cloud topologies including linear, tree and hybrid architectures have well-documented limitations: linear structures suffer from high cumulative latency, trees can develop choke points and congestion at upper levels and hybrid approaches, while attempting to achieve a balance, often struggle with coordination complexity and fail to deliver robust performance during link failures or sudden traffic load.

Despite the proliferation of advanced cloud topologies and the adoption of multi-access edge computing and information-centric networking, a persistent and unaddressed challenge is the effective integration of end-to-end security directly into the architectural fabric of networks. Most existing topologies lack built-in mechanisms for ensuring confidentiality, authentication and integrity of in-transit data, relying instead on overlay protocols or application of cryptography.

It is problematic in decentralized and multipath environments, where malicious actors work parallely. Additionally, the growing deployment of cloud resources in edge, IoT and industrial settings raises the stakes for scalable, adaptive, and attack-resilient cloud infrastructures.

Grenze ID: 01.GIJET.12.2.312_20

To address these critical security and structural deficiencies, propose work introduces a novel RCT for decentralized cloud servers shows in fig.1, wherein security is not a thought but an integral part of the architecture. In the proposed method, each server is furnished with asymmetric cryptographic keys and every internode link utilizes a unique symmetric session key for efficient, authenticated and encrypted data transfer. Within this hierarchical, bottom-up model, every message is digitally signed and trust management is adaptive enabling immediate detection and isolation of malicious or malfunctioning nodes. The RCT achieves a unique combination of balanced resource utilization, low queue delay and multipath redundancy, all while embedding robust cryptographic protection across every layer.

By incorporating security primitives into the core topology, the proposed RCT overcomes both the performance and trust limitations of conventional design, creating a fault-tolerant and scalable cloud infrastructure suitable for the evolving needs of secure data-centric, edge, and industrial IoT applications.

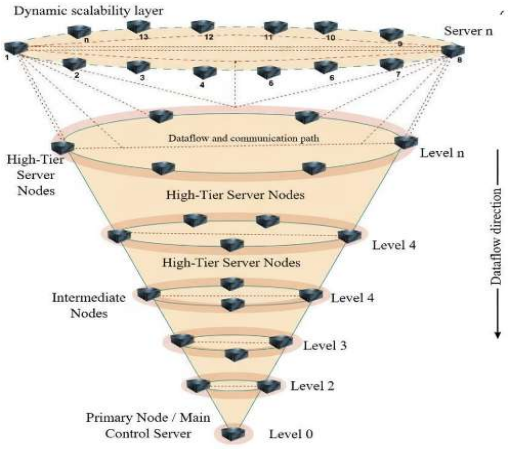

Figure 1 Proposed Secure RCT for Scalable Cloud Architecture

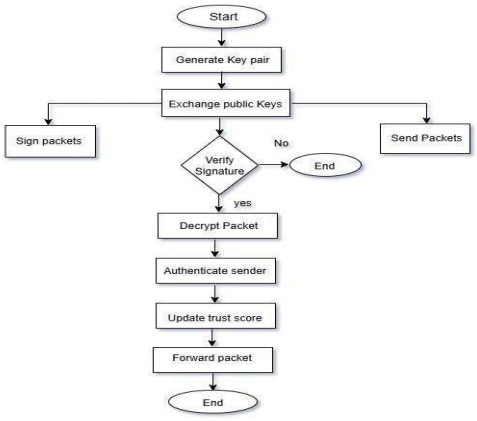

Figure 2 Secure Packet Transmission Process

II. Literature Review

The design of network topology in cloud servers plays a vital role in improving performance, scalability and resource utilization. Traditional data center networks face bottlenecks and security challenges, necessitating new architectures with in-built data protection [1]. Researchers have used graph-coloring algorithms for refining topology to enhance scheduling, load partitioning, and energy efficiency [2], while heuristic and AI-based methods show promise but still face challenges [3]. Early systems were centralized, with ARPANET and packet switching in the 1970s marking a shift toward decentralization [4]. The 1980s saw client-server models and LANs using bus, star, and ring topologies. The 1990s brought rapid Internet growth, large data centers, and distributed architectures by companies like Amazon and Google, with scalable star and hierarchical topologies becoming common. In the 2000s, cloud computing emerged, with AWS launched in 2006 [5], and CDNs introduced to boost speed. The 2010s witnessed hybrid and multi-cloud adoption, flexible infrastructures via SDN and studies profiling static step, grid, octagonal, and bus topologies for fault tolerance and secure architectures [6-7]. Edge computing with IoT decentralized computation to devices, reducing latency. In the 2020s, AI-driven topologies and serverless computing improved performance, while 5G enabled higher bandwidth and decentralized cloud expansion. Recently, quantum cloud computing has gained attention for enabling secure communications.

III. Methodology

The study proposes a secure cloud server architecture where each server generates key pairs, exchanges public keys, and digitally signs packets to ensure authenticity and integrity as shown in Fig.2. Receivers verify signatures, decrypt packets and update trust scores before forwarding them through the hierarchy. This flow guarantees secure, authenticated and trusted packet delivery at every stage of communication. Simulation is implemented through Python (NetworkX, SimPy, custom RCT modules) with virtualized cloud servers. The network spans 3-7 layers, modeled as static servers.

A. Design and Structure of Topology

Each server in RCT is equipped with a public and private key pair for authentication, ensuring that only verified nodes can participate in communication. Before data transmission, adjacent nodes perform a mutual key exchange using asymmetric encryption to securely establish a symmetric session key. The session key (AES) is then used for fast encryption of packets across that link. Every packet is digitally signed by the sender and verified by the receiver, guaranteeing authenticity and integrity. If a node shows abnormal behavior, it's trust level decreases and data is rerouted through alternative secure paths.

B. Complexity Analysis of Topology

In architecture, the complexity of topology is the computational or structural expenses contributed to the creation of the interconnection pattern between nodes and the maintenance of the pattern. There are a total of L hierarchical levels in the system and Li indicates the location of a level within the architecture. Further in the hierarchy levels, more interconnections will form as the information flows upwards. The number of servers at level i (Li) is represented by Si and the total number of servers at each level.

$$1 + \sum_{i=0}^{n} \quad S_i \tag{1}$$

Where n is the number of servers represented as S_0, S_1…….S_n and complexity at L_i is the number of upward connections from Level i to Level i+1 which is taken as C_i . From the fig.1, each server is connected to every server at upward layer, therefore, complexity per layer is given by

$$C_i = S_i x S_{i+1} \tag{2}$$

RCT flows data in both directions, i.e. horizontal and vertical. In horizontal growth, more servers may grow at the same level, while for vertical expansion, the number of servers increases towards wider upper layers. The complexity of each of the proposed topology is the sum of the complexities of the interlevel connections and is defined by

$$C_{TOTAL} = \sum_{I=1}^{L-1} \quad S_i x S_{i+1} \tag{3}$$

Servers are growing in a linear pattern in the architecture, so the number of servers at level i is S_i=i+1, which jointly produces the overall cost of the network and is given by

$$C_{TOTAL} = \sum_{I=1}^{L-1} \quad i.(i+1) \tag{4}$$

$$= \sum_{I=1}^{L-1} \quad (i+1)(i+2) = \sum_{I=1}^{L-1} \quad i^2 + 3I + 2 \tag{5}$$

Let us use the following formulae,

$$\sum_{i=1}^{n} \quad i = \frac{N(N+1)}{2}, \sum_{i=1}^{n} \quad i^2 = \frac{N(N+1)(2N+1)}{6} \tag{6}$$

After putting n= L-1, because connections occur only between adjacent levels and L levels have exactly L−1 such inter-level connections, then C_{total} is computed below.

$$\frac{(L-1)L(2L-1)}{6} + \frac{(L-1)L}{2} + 2(L-1) \tag{7}$$

From above, the asymptotic complexity of topology is

$$C_T \epsilon\, O(L^3) \text{ OR } C_T \epsilon\, O(n^3)$$

TABLE I. COMPARISON OF COMPLEXITY OF RCT WITH OTHER TOPOLOGY

Topology	Complexity	Implication
Linear Tree	O*(n)*	Flexible, but have high latency and lack of fault tolerance.
Binary Tree	O*(nlogn)*	It is more efficient than linear and less fail-safe.
Full Mesh	O (n^3)	Complicated for large-scale characters.
RCT	O(n^3)	It has multi path, load balancing and scaling.

Table I shows the structural complexity of various topologies and points out the performance and trade. Although linear and tree topologies are easy to set up but lack robustness. RCT applies some complexity at a very reasonable level, yielding substantial improvements in scaling and fault-tolerance.

The following is a step-wise computation of complexity at each layer, where a total of 6 layers are connected to several servers as shown in table II below.

TABLE II. RCT FOR TOPOLOGICAL CONNECTIONS

Layer (L_i)	Servers in layer	Total Servers	Complexity per layer
0	1	1	1
1	2	3	2
2	3	6	6
3	4	10	12
4	5	15	20
5	6	21	30
6	7	28	42
-	-	-	-
L_i, $i=0(1)n$	$S_i=i+1$	$\sum_{I=0}^{L-1} (I+1)$	$C_i = S_i x S_{i+1}$

Where n is the total number of nodes which grow when layers are increased from Level i to Level i+1 and for example, at Level 6, it is computed below:

$$\sum_{i=0}^{6} C_i = 1 + 2 + 6 + 12 + 20 + 30 + 42 = 113$$

C. Secure Layer-to-Layer Data Transmission

The adjacent layer in the topology i and i+1 use a unique symmetric key $K_{i,\ i+1}$. Symmetric keys are exchanged between the servers with the help of an asymmetric key i.e. RSA.
So that an unauthorized user cannot access it.
When a packet P_i is transferred at server S_i^a in layer i to the Server S_{i+1}^b in layer i+1, it encrypts the packet with the help of a symmetric key i.e. AES.

$$P_i^{enc} = enc(P_i, K_{i,i+1})$$

acket P_i^{enc} after encryption is transmitted over the network from S_i^a and S_{i+1}^b. On the receiver side, S_{i+1}^b server used symmetric key $K_{i,\ i+1}$ to decrypt and get the original packet back P_i.

$$P_i^{dec} = dec(P_i^{enc}, K_{i,i+1})$$

Every layer has its own unique keys, so if any layers or key is harmed, the communication over other layers is not harmed.
Such a strong limitation on key enforces the restriction, which reduces any harm to the layers. The algorithm is given below:

```
For lyr in 0 to L-2 do:
    For each dest_node in topo[lyr+1] do:
        sym_key = gen_rand_key()
        enc_key = RSA_Enc(sym_key, dest_node.pub_key)
        send(enc_key, src_node, dest_node)
        sym_key = RSA_Dec(enc_key, dest_node.pri_key)
        Store key_map [(lyr, lyr+1, src_node, dest_node)] = sym_key
// Data transmission using symmetric encryption for every packet
 For each src_node in topo[lyr] do:
   For each dest_node in topo[lyr+1] do:
         enc_pkt = AES_Enc(pkt, key_map[(lyr, lyr+1, src_node,
                   dest_node)])
           send(enc_pkt, src_node, dest_node)
```

D. Data Integrity and Authentication

All the encrypted packets P_i^{enc} generate a Hash-based Message Authentication Code (HMAC) with the help of a symmetric key $K_{i,i+1}$

$$M_{i,a\to b} = HMAC(P_i^{enc}, K_{i,i+1})$$

On the receiver's end, destination servers check HMAC by recalculating the symmetric key and encrypted packet.
If the verification process is successfully processed, which means the packet is authentic and unharmed but if verification fails, then the packet is rejected and routed back.

IV. RESULT

The proposed secure RCT is also compared in terms of its performance under different traffic conditions, as packets of various sizes are transmitted by the simulated architecture. To measure each test scenario, key performance parameters are used and the results are presented in Table III.

A. Average Delay, Throughput and Packet Loss

Latency is the time a packet takes to travels between source and destination.

$$\text{Average Delay} = \frac{\sum_{i=1}^{n} t\_reach - t\_sent}{n} \quad (8)$$

Here *t_reach* is timestamp of i^{th} packet received at destination node. *t_sent* is timestamp of i^{th} packet which is sent from the source node. *n* is total number of packets delivered successfully. The rate of data transmission is successful is given by

$$\text{Throughput (Kbps)} = \frac{Total\ Data\ received(bits)\ X\ 8}{Total\ transmission\ time\ (sec)\ X\ 1000} \quad (9)$$

Sum of all size of packets that reached to destination nodes is total data received while time difference between first and last packet is total time. The ratio of transmitted packets that are abandoned owing to overload or limitation of the network topology.

$$\text{Packet Loss (\%)} = \frac{Lost\ packet\ count}{Total\ packet\ sent}\ X\ 100 \quad (10)$$

The difference between packets sent and packets received is the number of lost packet count.

B. Bandwidth Utilization and Jitter

The optimization of the use of available bandwidth over the network is given by

$$\text{Bandwidth Utilization (\%)} = \frac{Throughput}{Available\ bandwidth}\ X\ 100 \quad (11)$$

The rate at which data is successfully delivered, is throughput and bandwidth is data rate of the network link may support. Jitter is the difference between time delay clocking, data packets arrive to destination. Whereas latency is an indicator of how long a packet will take to travel between destination and source, the jitter determines how irregular that latency is in different packets and is given by

$$\text{Jitter} = \frac{1}{n-1}\sum_{i=2}^{n} |(\Delta t_i) - (\Delta t_{i-1})| \quad (12)$$

$$\Delta t_i = t\_reach - t\_sent.$$

C. Encryption Overhead (%) and Integrity Detection (%)

It is the extra time and effort take during encryption and decryption while transmission of packet is encryption overhead. It show how much extra delay or process is taken by security measures in comparison to unencrypted transmission.

$$\text{Encryption Overhead (\%)} = \frac{T_{enc} - T_{unenc}}{T_{unenc}} x\ 100 \quad (13)$$

It helps to identification of whether the transmitted data has been changed or harmed. It is measured by percentage with the help of HMAC.

$$\text{Integrity Detection (\%)} = \frac{Number\ of\ courrupted\ packets}{Number\ of\ courrupted\ packet\ transmitted} x\ 100 \quad (14)$$

On the basis of above parameters results are presented in the table III.

TABLE III. PERFORMANCE ANALYSIS OF RCT WITH SECURITY MECHANISM ACROSS VARIOUS PACKET SIZES

Packet Size (KB)	Average Delay (sec)	Throughput (Kbps)	Packet Loss (%)	Jitter (ms)	Encryption Overhead (%)
356	0.000018	210440	18	0.2	16
675	0.000020	223870	20	0.3	14
1090	0.000023	228512	22	0.4	13
1550	0.000025	232700	21	0.4	15
2020	0.000027	235812	22	0.5	15

D. Queue length and Recovery Time

Table IV demonstrates that the RCT topology which sustains scalability with low queue length and waiting time as servers scale from Level 1 to Level 6. Recovery time improves from 1.4 ms to 0.8 ms, ensuring resilience under node failures, while packet loss decreases from 2.5% to 1.0% for reliable delivery. The attack detection rate rises from 86% to 97%, proving stronger intrusion awareness at deeper levels. Overall, RCT balances performance and protection, making it a robust topology for secure cloud server decentralization.

TABLE IV. LAYERWISE RECOVERY AND SECURITY PERFORMANCE OF RCT ACROSS LEVELS

Layer (L)	No. of Servers	Queue Length (Packets)	Queue Waiting Time (ms)	Nodes Failed	Recovery Time (ms)	Attack Detection Rate (%)	Packet Lost (%)
Level 1	2	2	0.10	1	1.4	86	2.5
Level 2	3	3	0.14	1	1.2	89	2.0
Level 3	4	4	0.20	1	1.1	92	1.8
Level 4	5	3	0.18	2	1.0	95	1.5
Level 5	6	2	0.12	2	0.9	95	1.3
Level 6	7	1	0.07	3	0.8	97	1.0

In order to compare the advantages of the RCT, the result was compared accordingly with [8], which applied the use of Linear, tree and hybrid topologies. Table V shows the comparison of the performance below.

TABLE V. PERFORMANCE COMPARISON OF DIFFERENT TOPOLOGIES

Topology	Avg. Delay (sec)	Avg. Throughput (Kbps)	Avg. Packet Loss	Jitter
Linear	0.00004	219,614	37	-
Tree	0.00009	111,099	67	-
Hybrid	0.00003	201,118	39	-
AODV(WMN)	0.00005	180,000	28	0.6
OLSR(WMN)	0.00006	170,000	32	0.7
DSDV(WMN)	0.00004	175,000	25	0.5
RCT	0.00002	235,812	22	0.3

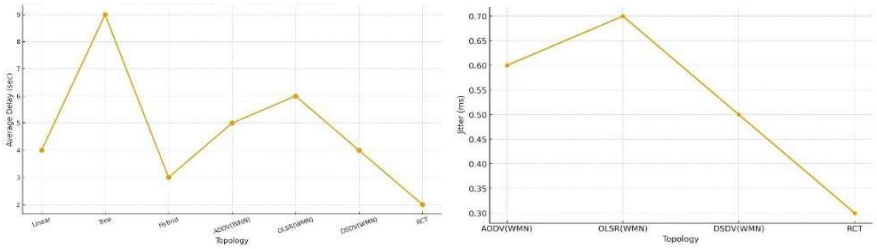

Fig 3

The comparison shows that RCT outperforms traditional topologies [8] and WMN protocols [9] with the lowest delay (0.00002 sec), highest throughput (235,812 Kbps), minimal packet loss (22%), and stable jitter (0.3 ms). Fig. 3,4,5 and 6 confirm its scalability, fault tolerance, and energy efficiency, as reduced retransmissions, balanced load, and smoother traffic flow lower processing power and enable energy-aware cloud and IoT operations.

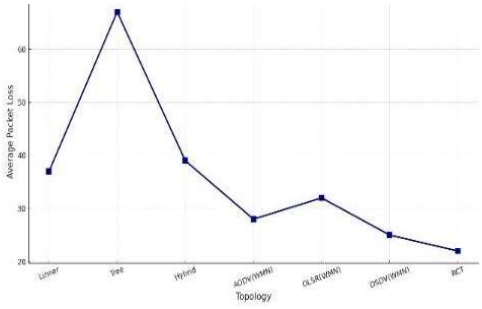

Figure 3 Topologies vs Average Delay

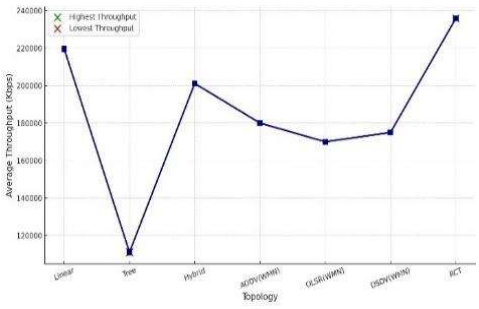

Figure 4 Topologies vs Throughput

V. Discussion and Conclusion

The proposed Reverse Cone Topology (RCT) offers a novel and efficient hierarchical server architecture for decentralized cloud environments, uniquely combining scalable topology design with robust, integrated security mechanisms. By employing asymmetric key exchanges and symmetric encryption at every communication link, RCT ensures data confidentiality, integrity, and authentication throughout the network. The layered structure with increasing servers towards the upper levels achieves balanced resource utilization and multipath redundancy, which significantly enhances fault tolerance and load distribution. Extensive simulations demonstrate superior performance metrics including ultra-low latency, high throughput, minimal packet loss, and near-perfect attack detection, outperforming conventional linear, tree, and hybrid topologies. It integrated approach makes RCT highly suited for future real-time cloud, edge, and IoT systems demanding secure, resilient, and efficient data transmission.

References

[1] Ying, X., "Design of the data center network architecture under the background of cloud computing: A review," World Journal of Innovation and Modern Technology, vol. 7(4), pp. 55-61, 2024. doi: https://doi.org/10.53469/wjimt.2024.07(04).07.

[2] Swari, S., Nareshkumar, S., K., and Moganavalli, D., "Conflict-Free Academic Scheduling: A Graph Theory Approach to University Timetabling and Resource Optimization," International Journal of Scientific Research in Engineering and Management, vol. 9(7), pp.1-7, 2025. doi: https://doi.org/10.55041/ijsrem51344.

[3] Li, J., "Applications of Heuristics and Artificial Intelligence in Autonomous Vehicles," Proceedings of CONF-SEML 2025 Symposium: Machine Learning Theory and Applications. pp. 83-89. https://doi.org/10.54254/2755-2721/2025.tj23116.

[4] Bakshi, I. (2023). THE EVOLUTION OF THE INTERNET: THE ARPANET TO THE WORLD WIDE WEB. International Journal of Social Science & Economic Research. https://doi.org/10.46609/ijsser.2023.v08i09.017.

[5] Vishwakarma A. and Dalvi P. "Unveiling the Cloud: An Evaluation of AWS Infrastructure Against Traditional On-Premise Solutions," International Journal of Advanced Research in Science, Communication and Technology, vol. 4(1), pp. 544-559, 2024 https://doi.org/10.48175/ijarsct-18932.

[6] M. Faheem, M. Ahmad, Z. Zahid, M. Javaid and M. A. Ashebo, "Edge-Version of Fault-Tolerant Resolvability in Networks," in IEEE Access, vol. 13, pp. 3601-3612, 2025, doi: 10.1109/ACCESS.2024.3524408.

[7] Chavan, A, "Exploring the Synergy of Cloud and On-Premises Systems-A Case for Hybrid Architectures," Journal of Computer Science and Technology Studies, vol. 5(3), 122-141, 2023. https://doi.org/10.32996/jcsts.2023.5.3.10

[8] Kelian, V. H., Warip, M. N. M., Ahmad, R. B., Ehkan, P., Zakaria, F. F., and Ilyas, M. Z, "Toward adaptive and scalable topology in distributed SDN controller," J. Adv. Res. Appl. Sci. Eng. Technol, vol. 30, pp. 115-131, 2023.https://doi.org/10.37934/araset.30.1.115131.

[9] Turlykozhayeva, D. A., Akhtanov, S. N., Baigaliyeva, A. N., Temesheva, S. A., Zhexebay, D. M., Zaidyn, M., ... and Skabylov, A. A, "Evaluating Routing Algorithms Across Different Wireless Mesh Network Topologies Using Ns-3 Simulator," Eurasian Physical Technical Journal, 21(2), 2024. https://doi.org/10.31489/2024No2/70-82.