Applications hosting over cloud-assisted IOT: a productivity model and method defining accessibility of data security

Prashant Vaish, Indian Institute of Information Technology, Lucknow, India.

Niharika Anand, Indian Institute of Information Technology, Lucknow, India.

Vishal Krishna Singh, Indian Institute of Information Technology, Lucknow, India and University of Essex, UK

Gaurav Sharma, University of Sheffield, UK.

Accepted for publication in The Journal of Supercomputing.

Applications Hosting Over Cloud-assisted IoT- A Productivity Model and Method Defining Accessibility of Data Security

Prashant Vaish[1]. Niharika Anand[1]. Vishal Krishna Singh[2]. Gaurav Sharma[3]

Abstract

The Internet of Things (IoT) has increased the demand for data, which has been met through the use of IoT-assisted cloud computing. However, this paradigm introduces new security complexities regarding the exchange of data between entities. In this paper, we investigate a secure strategy for managing IoT data in a cloud-assisted environment, protecting data privacy during data collection, storage, and access. Our method for mitigating the impact of IoT scalability is meticulously devised and supported by empirical evidence. Our productivity model is based on key operations, configurations, and efficiency factors. Our proposed method makes a substantial contribution to both system scalability and user data privacy, surpassing previous scale levels by a significant margin. In particular, our research investigates a secure strategy for managing IoT data in a cloud-assisted environment. We believe that our research will assist in bridging the divide between infrastructure, development, and testing teams, resulting in robust and stable productivity software. Our findings demonstrate the viability and efficacy of our proposed method, which outperforms previous models and previous research.

Keywords— Internet-of-things (IoT); Productivity model in cloud (PMC); Platform as a service (PaaS); Workflow's execution model (WEM); Cloud hosting model (CHM).

1 INTRODUCTION

Any application or software built over the cloud exists virtually, making it accessible from any device anywhere, be it at home or away, depending on how we set the permissions and access fronts. Recent deployments of Internet of Things (IoT) applications process massive gigabytes of data in health and banking industries, requiring high-performing and massive storage infrastructures. Leveraging cloud in IoT benefits these high storage and computation abilities and provides IoT users great convenience to collect, store, and access data from anywhere(Gubbi et al. 2013). However, this convenience over the cloud brings data security challenges over any application, especially with the more versatile nature of user requests. Whether establishing a trust-based system, this data security risk is hardly negotiated over data obfuscation or operations(Wei Wang, Xu, and Yang 2018). Once an application gets hosted over the cloud and steadily different teams and users get trained on that application, the business growth and resources get higher welfare over the long run. Fig 2 shows below other types of application hosting over the cloud (Gao, Zhang, and Zhou 2019). Although the hybrid type allows organizations to remove on-premises dependencies and scale up quickly, data centers are not mandatory for all cloud types. That is why public cloud hosting holds a significant percentage for

Cloud Integration.

It shows a large-scale dispersed computing example driven by the scale of economies. A pool of inattentive, virtualized, dynamically elastic, accomplished computing power, storage, podium, and amenities are transported on request to external customers over the internet (Hilman, Rodriguez, and Buyya 2020). At the same time, many organizations are still trying to cope with their way to the cloud and currently using the heritage culture of no-cloud and on-premises hosting of applications. The current age has increased companies' and individuals' day-to-day operation of smart devices and computers(Naha et al. 2018). Subsequently, many organizations face the requirements and responsibilities of storing substantial data volumes. However, standards for cloud hosting of applications are not uniform, requiring different cloud providers' services over troublesome computing, synchronization, competencies, and technical skills. Additional cloud services like Infrastructure as a Service (IaaS), platform as a service (PaaS), and software as a service (SaaS) are elegant and elastic; still, there have to be factors to consider and monitor them to increase productivity on the overall system level. Infrastructure as a service (IaaS)

Prashant Vaish is a Research Scholar at Indian Institute of Information Technology, Lucknow, India (e-mail: rwc202002@iiitl.ac.in).
Niharika Anand (e-mail: niharika@iiitl.ac.in) is working as Assistant Professor at Indian Institute of Information Technology, Lucknow, India .

Vishal Krishna Singh (e-mail: v.k.singh@essex.ac.uk) is a Lecturer in the School of Computer Science and Electronics Engineering, University of Essex,Colchester, U.K.
Gaurav Sharma is working as Visiting Lecturer and perusing a PhD at University of Hertfordshire Hatfield, UK (e-mail: g.gaurav@herts.ac.uk).

clouds proposals execution of large-scale work as a new utility-based platform(Wanyuan Wang, Jiang, and Wu 2016) (Sousa et al. 2014) where virtual machines (VMs) are the leading resource of computing for consumers wherein they can lease as per demand. Various computing models assumed an unrealistic study that execution time on all VMs of all tasks is known in advance(Z.-G. Chen et al. 2015).

In contrast, a more realistic rehearsal study defines the VMs model based on server capacity, cost, or execution. However, most central processing units (CPU) have been proposed and governed to determine the speed and thought homogeneous as VMs(Nieuwenhuis, Ehrenhard, and Prause 2018), the influence and involvement of other infrastructures or characteristics are usually ignored (Rodriguez and Buyya 2014) (Bittencourt and Madeira 2011). Thus workflows scheduled on different VMs perpetually run on cost sizing and cannot be practiced (Jung and Kettimuthu 2014) (C. L. P. Chen and Zhang 2014). While over the years, additional features have (Heilig and Voß 2014) been revisited on design and architecture like scheduling of jobs, secure hosting, and monitoring, still, they did not weigh upon the continuous productivity increase with an increase of resources and requirements. Nowadays, the needs of the customer over Cloud Computing (Wan et al. 2018) show individualization and a variety of trends, and the customization of production has eventually become mainstream and tends to develop a regular want of high-quality production and productivity.

Below, Fig 1 shows data transfer classifications and IoT collection classifications over the cloud. IoT users can navigate their tasks over the cloud and collect or transfer expected data from the cloud based on requirements. As the number of IoT users increases, the requests and responses for these data are always high, free-flowing, and not traced in terms of data security.

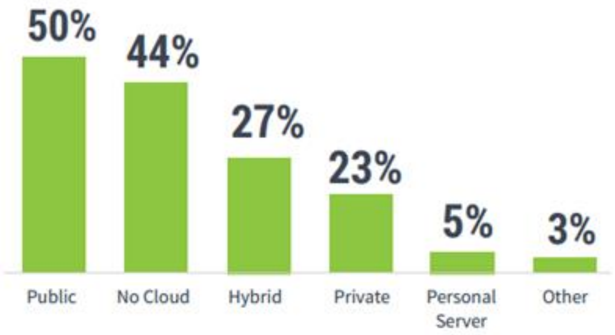

Fig. 1 Types of application Hosting on cloud

This paper also proposed a Productivity Model on Cloud (PMC) to increase productivity for any application hosting overcloud. It contributed in enhancing Clous data security and data encryption for IoT components so as to ensure secure communication in addressing data access limitations. We discuss existing traditional models on data security and cloud implementation, which mainly focus on reducing cost and increasing execution time, and some of the essential factors that are lacking to suppress the productivity of any system. Section 2 talks about the three-stage method of IoT's data security and some important usage characteristics in the cloud for high-performing teams (Jiang and Wan 2021)(Jia et al. 2018). Section 3 highlights the critical problem statements faced by traditional cryptographic methods. Our current research focused on nullifying and about problems for any team or application to consider hosting any Cloud application. Section 4 discusses this three-stage method of data security for cloud-assisted IoT. Section 5 brainstorms some crucial operations, configurations, and efficiency factors on which our Productivity model has been laid upon. Section 6 displays the major results in time and communication between the data security method and also in operations and proficiency as a part of productivity increase. Section 7 defines the conclusion about these methods and models while Section 8 has the declaration index.

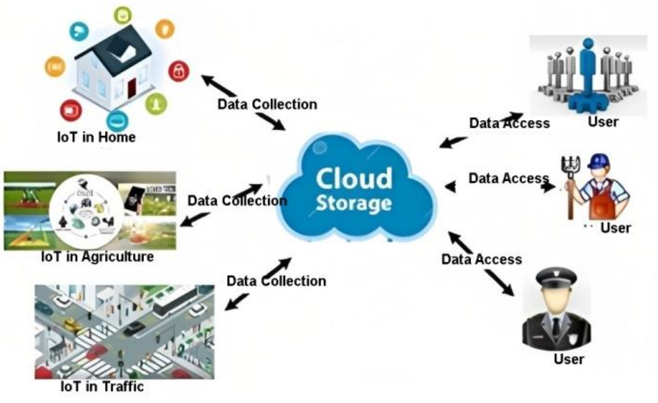

Fig. 2 Data Transfer in Cloud-assisted IoT

2 Related Works and Key Findings

The significant issues in the literature for IoT over the cloud are integration and authentication via the cryptography method. However, none thought around data security over cloud-assisted IoT. Singh et al. in (Singh et al. 2015) were the initial ones to think about any confidential data and cryptography with the private key however did not produce effective and appropriate results to justify any security. In addition to that, Bhuse et al. (BHUSE 2014) also emphasized encryption via public key but did not present effects on any data access security and storage. If we break the IoT components into two categories: IoT internal components and IoT external components, these previous works mainly revolve around IoT external components over without any data security. IoT internal components can be stated as artifacts that only interact within themselves and not with the outside world whereas IoT external components interact with outside technology and cloud integration.

Cloud computing technology is generally employed in most existing IT organizations in partnership with cloud vendors, such as Google, Amazon, and Microsoft. With increased customer demands and requirements, all organizations shifted towards cloud-based models for their overall system

Existing Work	Cloud Hosting	Cost	Execution Time	Number of Users	Workflow execution	Technical Practices	Automation	Code Maintenance	Monitoring	Data Access & Security
(Singh et al. 2015)	✓	×	×	×	×	×	×	×	×	×
(BHUSE 2014)	✓	×	×	×	×	×	×	×	×	×
(Durillo and Prodan 2014)	✓	✓	×	×	✓	×	×	×	×	×
(Pandey, Wang, and Calyam 2019)	✓	✓	×	×	✓	×	×	×	×	×
(Fraj, Hlaoui, and BenAyed 2020)	✓	✓	×	×	✓	×	×	×	×	×
(Rimal and Maier 2016)	✓	×	✓	✓	×	×	×	×	×	×
(Jagaty et al. 2020)	✓	×	✓	✓	×	×	×	×	×	×
(Shi et al. 2019)	✓	×	✓	✓	×	×	×	×	×	×
(Lakhan et al. 2022)	✓	×	✓	✓	✓	✓	✓	✓	×	×
Our Work	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 1 Survey Comparisons

deliverables (Nieuwenhuis, Ehrenhard, and Prause 2018)(Ghahramani, Zhou, and Hon 2017). In previous years, hosting applications over the cloud was inspired by Workflow models where we had several workflows to push chunks of software. However, with the continuously growing complexity of systems and tedious client requirements, handling data and communication with computation is becoming difficult.

These workflow models are significant as they involve multiple dependents and independent applications on equally massive Infrastructure and storage. Integrating these pieces in their best form is always a challenge regarding costs, practices, technical competencies, and disaster recovery planning.

Some of the previously suggested or worked models on these workflows are Durillo and Prodan (Durillo and Prodan 2014), Wang(Pandey, Wang, and Calyam 2019), Ben(Fraj, Hlaoui, and BenAyed 2020) to reduce the cost of the workflows execution model (WEM) Fig 3 and Rimal and Maier(Rimal and Maier 2016), Sahoo(Jagaty et al. 2020), Tianbing(Shi et al. 2019) proposed cloud hosting model (CHM) to increase execution time Fig 4. This CHM model has tenants (users or organizations) interacting with the cloud provider to access cloud services. In the virtual infrastructure layer, the cloud provider allocates virtual resources to the tenant based on their subscription or requirements. The tenant's applications and data are hosted within their allocated virtual machines in the virtual infrastructure layer. When a user accesses a service hosted by the tenant, the request is sent to the middleware layer. The dispatcher in the middleware layer routes the request to the appropriate virtual machine based on predefined rules. The virtual machine processes the request and interacts with the application and data hosted within it. If necessary, the virtual machine can communicate with other virtual machines or services within the same tenant's environment or even across different tenants. The workflow scheduler in the middleware layer ensures that resources are allocated efficiently to handle incoming requests. The service queue manages the queue of incoming requests, preventing overload and ensuring fair processing. The virtual machines interact with the virtual infrastructure layer to access resources like storage or networking services.The virtual infrastructure layer maps the virtual resources to the physical resources in the physical infrastructure layer, ensuring efficient utilization. The physical infrastructure layer manages the actual hardware resources, providing the computing power and storage required by the virtual infrastructure.This multi-layered cloud hosting model providers to offer scalable, flexible, and efficient services to various tenants while maintaining isolation and security between them. One of the recent works (Lakhan et al. 2022) also depicted an algorithm model on dynamic service composition at phases of sequencing and scheduling on healthcare platforms, however did end up with a lack of scalability and heavy and insecure infrastructure. While WEM operates on chunks of workflow execution in scheduling over virtual machines over limited resources, CHM, on the other side, tries to reduce the execution time by adding more users and tenants on numerous resources and comprehensive Infrastructure. However, both these models lack technical practices, automation pipelines, testing of DR, and team synchronization. Moreover, the code maintenance is so high that it burdens the overall architecture.

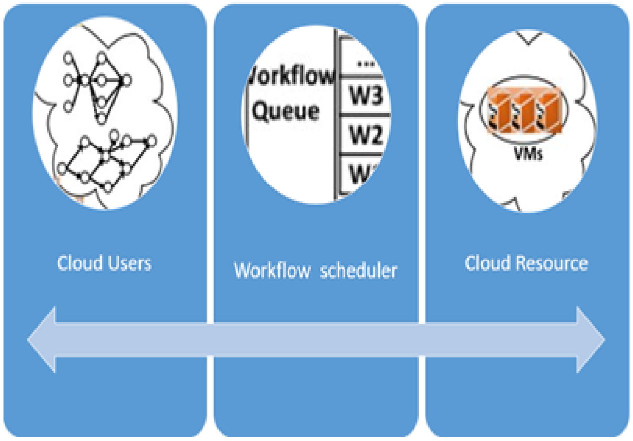

Fig. 3 Workflow execution model

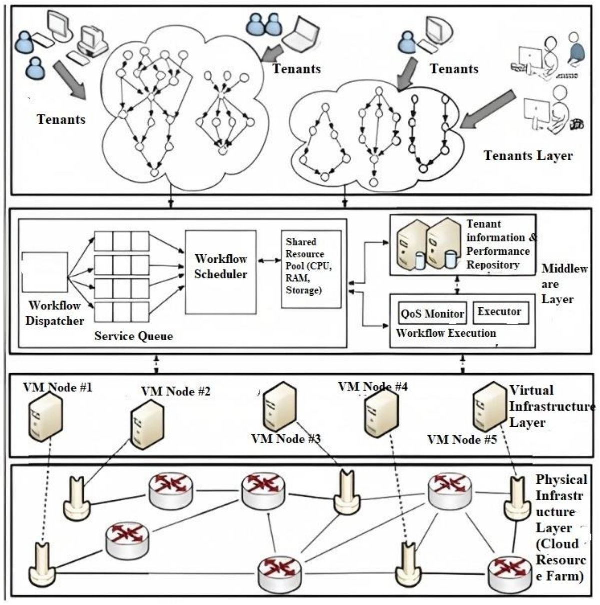

Fig. 4 Cloud Hosting Model

Table 1 above shows a literature review or comparison between those parameters in previous models and our research where none of the earlier models has code maintainability and data security factors. These are well governed by continuous monitoring via automation and technical practices using easy-to-go tools. All the previous models were hosted over the cloud; however, only a few recent ones focused on workflow execution and cost. The most crucial factor always lies in data security as to how data is collected, accessed, and stored. Irrespective of multiple and several requests by many users of heterogeneous nature, our research model has provided a fixed cost of communication with constant obfuscated text. Our proposed model is more crucial and proves superior as it is structured into three distinct phases, each addressing specific security aspects: ensuring security during data collection, data storage, and data access. These phases take into account varying types of potential attackers. During the data collection phase, the primary threats typically stem from adversarial IoT-edge objects and potential eavesdroppers. In the data storage phase, the focal point shifts to users who access IoT data from the cloud. Lastly, the data access phase is susceptible to attacks from adversarial users and eavesdroppers. Given the diverse characteristics of potential attackers, our proposed model has effectively addressed the following critical challenges:

- To counter eavesdroppers, it is imperative to ensure that all communications occur through secure channels and are encrypted. Additionally, the decryption keys must remain undisclosed to any potential eavesdropper.
- To thwart adversarial IoT-edge objects, a key requirement is that each IoT-edge object possesses distinct encryption keys for encrypting its data during the data collection phase. This enforces the principle that no object can decrypt the ciphertext of another object.
- To mitigate risks originating from the cloud, our model mandates that all IoT data be stored in the form of ciphertexts within the cloud. Importantly, the cloud is rendered incapable of decrypting any ciphertext.
- Conventional access control mechanisms are inadequate to combat adversarial users. Traditional access control permits a server to fulfill a user's data request based on their specific rights. In many cases, data is stored in plaintext on the server. Consequently, the effectiveness of traditional access control hinges upon complete trust in the server. However, in scenarios where the server's trustworthiness is compromised, traditional access control becomes ineffectual, allowing the server to release sensitive data directly to adversarial users. Given that the cloud operates as an honest-but-curious entity, traditional access control proves unsuitable for achieving our objectives. In light of this, our proposed model introduces encryption-based access control as a promising solution.

We built a method considering three stages of collecting, storing, and accessing IoT data as part of this research. Each stage has different security threats like collecting data with IoT external objects and cloud or between cloud and users. Storing data has threats from users having the accessibility of IoT data in the cloud. Data access on similar grounds has the main threat from users having full access to IoT external components over the cloud.

The main contribution of this paper is to establish a secure cloud-assisted IoT and to establish that to choose perfect encryption methods between two categorized schemes, public-key encryption (PKE) and symmetric-key encryption

(SKE). The primary distinction between Public Key Encryption (PKE) and Symmetric Key Encryption (SKE) lies in their utilization of asymmetric and symmetric keys, respectively. In the context of cloud-assisted IoT implementation, PKE offers the advantage that users and IoT-edge objects do not necessitate simultaneous online presence. Conversely, SKE mandates that both users and IoT-edge objects are concurrently online.

In terms of time efficiency, PKE typically demands more time than SKE for generating ciphertext. However, the execution duration of PKE does not present a significant vulnerability, as the time cost is associated with generating the private key rather than directly with the file. Consequently, when encrypting slightly larger files, the time expenditure of PKE does not emerge as the principal determinant impacting the performance of the cloud-assisted IoT system. In our cloud-assisted IoT system implementation, utilizing Public Key Encryption (PKE), each IoT-edge object and user possesses distinct sets of public and private keys. These private keys are employed for decrypting the respective ciphertexts encrypted through PKE.

On the other hand, while building a plan to excel, the below characteristics of cloud are also being followed and executed.

• Cost of Cloud usage: Managing cloud cost over any implementation or set of any infrastructure has many changing minds. Whether it's a whole stack of servers or utilizing a complete data center, how it can be measured and paid is always a matter of discussion.

• Services that are self-oriented and on-demand: Providers and Consumers can interact and provide on-demand computing resources free from human interactions.

• Accessing network over broader platform: Abilities of various platforms like mobiles, laptops, and desktops to access cloud infrastructure or resources.

• Maintaining a pool of resources: Works on a rental model where on-demand resources can be assigned to physical or virtual servers from the maintained pool by just providing inputs such as demographics.

• Elastic and scalable: It should be scalable to rapidly changing requirements, whether expansion or degradation over a short period.

• Measurable services: Depending on the service requests like bandwidth, utilization, and storage, it can measure, control, and report resources

3 Problem Statement(s) and research

For excelling IoT applications over the cloud, the below problem statements need to be addressed against the three stages of collecting, storing, and accessing data.

•All communication should happen via a secure key channel to limit IoT external components.

•Each IoT external component should have its data key to encrypt for collecting data.

•To restrict cloud access, all the data storage should be in the form of obfuscated text.

•Traditional data access methods allow the complete user control of data if he has access to the servers and all data is stored as plain texts over those servers. To restrict this, an encrypted-based model needs to be set up.

Also, the findings discussed above on Cloud characteristics will build a success path for various teams in any organization in terms of readiness and control. However, extra factors lay how productive any group, application, or organization can be.

Below are the factors that weigh extra pounds on the productivity factor of hosting applications on the cloud.

• Practices to adopt technically: Different companies nowadays always want to adopt automation and their respective technologies to enhance their cultural and process roadmaps. However, how soon they adapt or start these ventures is highly dependent on their current state. This is often measured by the results of minor or significant deliveries of software/applications and the nature of those models, whether robust or elastic.

• Team's synchronized effort at organization stages: Competences in any organization are developed at either team's level or organization level; however, both need to work concurrently many times to deliver and deploy in continuous delivery (CD). Still, the CD must be closely eyed with productivity and client satisfaction for any organization to succeed.

• Technical competencies at the team's stage: This revolves around testing automation in CICD. The aptitude for fixing issues quickly, getting continuous feedback from clients for their testing, and improving the quality of test runs' re-iteration governs it. However, even after testing automation keeps on fixing and integrating, a successful software release can only measure how productive that end product gets released.

• Technical competencies at the organization's stage: On a separate side from team-level competencies, we also have organization-level competencies that are impacted by design or decision. Designing architecture or making change management decisions impacts productivity or the successful release of any project.

• Testing of Disaster Recovery: Every organization nowadays that runs different software systems has service level agreements on the incidents that may be coming into a Production environment. Few incidents are legitimate to reach in productivity. Some testing is always out of scope for lower environments because of the infrastructure resource limitations of lower settings or the data complexity of productivity, which cannot be reproduced in testing environments. To counter these scenarios, disaster recovery testing is a must for increasing productivity and testing factors of restoring backups in case of system downtime. This is more substantial where we have complex and huge systems.

Table 2 shows how often organizations perform disaster recovery testing on the infrastructure of production in terms of their performance. These defendants also depict different disaster recovery testing performed in an organization.

- ✓ Availability of a system is always on the positive and higher side by organizations that conduct testing of this Disaster recovery. This will help interns benefit from robust and scalable systems and client expectations.
- ✓ This also improves communication and process effectively as it will touch many corners of cross-function and diverse teams.

• Change Approval and Process Management: Pushing a change in productivity is often a complex procedure due to the number of teams involved in the integration and the change approval process. While we can plan the integration of groups by segregating their duties, on the other side, the change management process is hard to control by practice leaders. While we know that a single person cannot control the entire process, still measures have to be laid out to break into trivial ways which increase productivity and launch numerous services with enhanced service level agreements (SLAs) so that customers can get more support on production environments and business-critical operations(Stamford 2019).

• Safety of Psychological trends: The environment in a working team often lays down trends of psychological safety in terms of brisk and clear communication, trust, and significant work. This allows and predates high performers to take calculated risks and think more about design and productivity.

Table 2: Disaster Recovery Test Types

	Low	Medium	High	Elite	Overall
Table-top exercise that are not carried out on real systems	35%	26%	27%	30%	28%
Infrastructure (including datacentre) failover	27%	43%	34%	38%	38%
Application failover	25%	46%	41%	49%	43%
Simulations that disrupt production-like test systems (including failure injection such as degrading network links, turning off routers, etc.)	18%	22%	23%	29%	23%
Simulations that disrupt production systems (including failure injection such as degrading network links, turning off routers, etc.)	18%	11%	12%	13%	12%
Creating automation and systems on a regular, ongoing basis	9%	8%	7%	9%	8%

4 Data Security Method for Cloud Assisted IoT

The first and most crucial step in a secure cloud-assisted IoT is generating and encrypting the public key. Public key encryption is not limited to IoT users or external components to be online constantly and is quite effective in handling large data volumes. Our new proposed model with three stages below defines this key encryption and mathematical conditions to lay out their respective encryption model which is quite differentiated as compared to any other previous models of cloud hosting.

- *COLLECTING DATA –*

In this stage, all IoT external components upload data to the cloud over a request done by an IoT user or any other external element. As mentioned earlier in this research, this stage will be based on the concept of encrypting/obfuscating data via public key once that has been sent to the cloud by an IoT external component or user, as shown in Fig 5.

- ✓ The user over the cloud made a data collection request.
- ✓ Cloud identifies that request as € and collects respective data as α and assigns a public key μ to it.
- ✓ IoT external object validates this request € and ignores any additional data that came along with this request.
- ✓ IoT external objects then apply a condition β on this data sharing and obfuscate this data by algorithm C = Obs (€, μ, α, β) and send this obfuscated text to the cloud.
- ✓ Cloud then stores this obfuscated text C and also € and sends (C, €) to the corresponding user.
- ✓ The IoT user then decrypts this data request using the same public key μ.

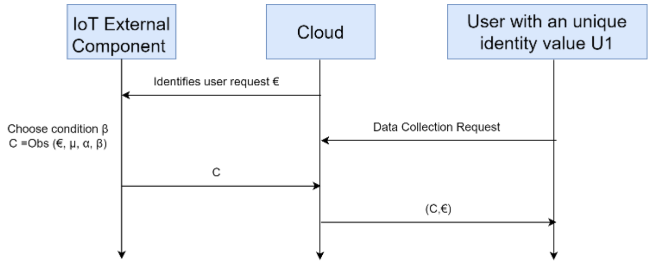

Fig. 5 Data Collection

Algorithm 1: Data Collection.

```
Algorithm: Data Collection
    Initialization:
    Initialize
    INPUT: Req €, Collects α, public key μ
    End Initialization
    Task Processing:
    For
        € is external object data
If €=β
        Then Obfuscate (€, μ, α, β) as C
And
        Send C and € to user
Else if
    OUTPUT: Decrypt € as U1
Else
        α not valid
End
    End
```

- *STORING DATA-*

Storing data is determined as confidential in terms of cloud and obfuscated text. This confidentiality ensures that only IoT users having requests as € can identify data α. Therefore, the only way for the cloud to view obfuscated text C is to merge with that corresponding user and seek his nod to recognize the same.

Algorithm 2: Storing Data

```
Algorithm: Storing Data
    Initialization:
    Initialize
    INPUT: Req €, Collects α
    End Initialization
    Task Processing:
        For
            € is external object data
    If €=β
            Then Obfuscate (€, α) as C
    And
    OUTPUT: Send C and € to user
    Else
            α not valid
    End
        End
```

- *ACCESSING/SHARING DATA –*

This stage defines sharing the received IoT data by a user to other users via cloud assistance. Consider a user with a unique identity value of U1 who wanted to share the data α, stored over the cloud as obfuscated text C with another user with a unique identity as U2; the below steps in Fig 6 depict this data access stage.

- ✓ A user with a unique identity value of U1 sends his value and requests to share data to the cloud with its unique public key µ.
- ✓ The cloud validates these requests over certain intervals and determines requests € having data as α with condition β and also a unique identity value of U1.
- ✓ Another user having a unique identity as U2 also chooses the same request € with condition β and initial obfuscated text C and generates a re-obfuscation key R=RObs(µ, U2, α,β) and sends to the cloud.
- ✓ The cloud again identifies this re-obfuscation key and stores obfuscated text C2.
- ✓ The IoT user then decrypts this data using the same unique identity as U2, re-obfuscation key R, and public key µ.

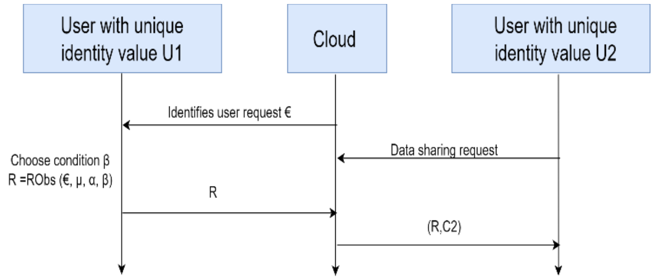
User with unique identity value U1
Cloud
User with unique identity value U2
Identifies user request €
Data sharing request
Choose condition β
R =RObs (€, µ, α, β)
R
(R,C2)

Fig. 6 Data Access/Share

Algorithm 3: Data Sharing/Accessing.

Algorithm: Data Sharing/Accessing

```
        Initialization:
        Initialize
        INPUT: Req €, Collects α, public key µ, user U1, user U2
            End Initialization
        Task Processing:
        For
        € is external object data
    If €=β
            Then Obfuscate (€, µ, α, β, U1) as C
    And
            Send C and € to user
    Else if
    OUTPUT: Decrypt € as U1
    Else
            α not valid
    End
    For
    € is external object data
    If €=β
    OUTPUT :Then Re-Obfuscate (µ, U2, α, β) as C2
    And
                Key as R
    Else If
    OUTPUT: Decrypt (µ, U2, R, C2)
    Else
        α not valid
    End
        End
```

5 Improving Productivity:

Productivity is the ability to complete complex, time-consuming tasks with minimal distractions and interruptions. This research proposes the below PMC model to increase productivity by identifying the capabilities that genuinely impact it. This proves to be an important goal in teams and organizations to get more value out of your transformation and your employees.

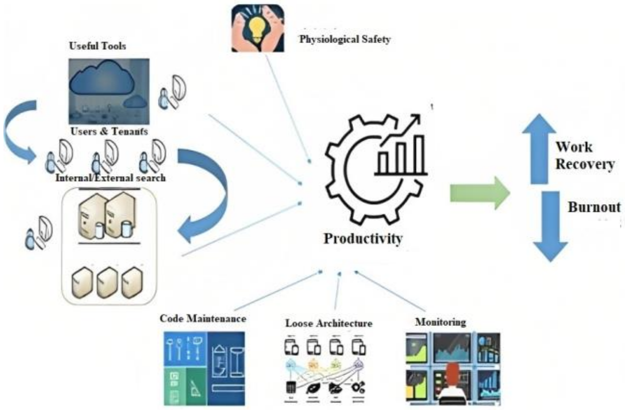

Fig. 7 A Pictorial view of Productivity Model in Cloud

Fig 7 shows a pictorial, and Fig 8 depicts a flowchart representation of our proposed Productivity Model on Cloud. The outlines of how it can be supported by any organization with the right choice of valuable tools that are easily used and searchable information and how it is hampered by debt on the technical side resulting in burnout and imbalanced work and life measures are shown in these two figures. Being Productive always yields good performers and gives them extra time to take up other assignments like stats, reports, documentation, or any other functional piece that is always beneficial in overall delivery or Infrastructure.

• ***Tools that are easy to use and useful:*** Choosing the best and proper tools is a must-have in any technology, especially with retail and banking clients, for effectively managing complex Infrastructure and critical systems. This factor is ignored most of the time. Professionals are under the impression that whatever tool they already have will work well and other factors like cost and vendor management are much more important to consider.

We engrossed automation toolchain which deploys software via Continuous Integration and Continuous Deployment (CI/CD) via DevOps and drives below important qualities of productivity:

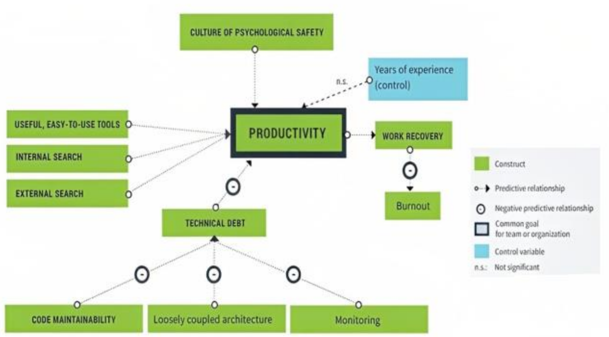
CULTURE OF PSYCHOLOGICAL SAFETY
Years of experience (control)
n.s.
USEFUL, EASY-TO-USE TOOLS
INTERNAL SEARCH
EXTERNAL SEARCH
PRODUCTIVITY
WORK RECOVERY
Burnout
TECHNICAL DEBT
CODE MAINTAINABILITY
Loosely coupled architecture
Monitoring
Construct
Predictive relationship
Negative predictive relationship
Common goal for team or organization
Control variable
n.s.: Not significant

Fig. 8 Productivity Model in Cloud (PMC)-Flowchart view

- ✓ It is easy to use and deploy in CI/CD Pipeline with user-friendly interfaces and operations.
- ✓ Useful in Integrating different projects and overall software as a whole.
- ✓ We explored more on the tools and software usage among teams and noticed below essential decorations:
- ✓ Wherever Proprietary software is used majorly, the performing stats depreciated with high support and maintenance costs. The elite performers have upgraded themselves from this software and always suggested scaling up to automation tools.
- ✓ Have also found that high-performing teams or organization uses software that is easy to develop Out of the box (OOTB) customization. Regular old model tools do not have this scalability.

Teams that generally tend to this proper selection of tools can yield better productivity results and thus software delivery.

• ***Searching internally and externally:*** Everybody is trying to build complex systems, which often generate many errors in today's world. Searching for the exact and correct details will be a prominent factor in debugging any exceptions or errors, thus increasing productivity and maintaining workflow. It can be categorized into two categories-

- ✓ Internal: Core functions and processes can often be searched internally within an organization's website where development or support teams have login permissions to search or compare their project-related work, such as ticketing, service level agreements, or similar reusable codes. This provides easy and quick searchable options and rapidly applies to their system for swift productivity.
- ✓ External: This defines searching over external web engines like technology forums. The frequent nature of technology helps us to indulge in learning communities and grow ourselves. This feeds to increase productivity as it allows various options to study and implement the best one.

• ***Debt on technology:*** These concerns are anything we owe to technology due to undeveloped or unripe coding, whether in the infra side, development, or configurations. Below are well-known debts that have an inverse impact on productivity and need improvement.

- ✓ We are trying to deploy new subsets of code, although the parent system has open bugs.
- ✓ Inadequate test planning and execution.
- ✓ Not proper designing and thus a poor class of code.
- ✓ Existing features that are obsolete or have not been used have to be cleaned and maintained.
- ✓ The team does not have the right technical skillset for software or application implementation.
- ✓ Implementation where only some % of the software has been deployed.
- ✓ Poor documentation of software code or components.

In today's complexity around the software being implemented, debt on technology is an essential factor in which productivity is weighed to a significant percentage.

• ***Psychologically safe culture:*** Building a culture of trust and contribution is deemed psychologically safe for increasing productivity in any project or team. Existing and new joiners must update and maintain a good gift of documentation that is easy to understand and helpful for anybody entering any complex project. Updating the document's code and configuration components allows examining and upgrading to any required versions in the coming years. Maintaining this open environment of trust and contribution also always inspires team members to flourish in best practices, whether in code or tools.

• ***Architecture that is loosely coupled:*** One of the main features of designing at the Organization level is the development of teamwork in a silo on any change to fix, test, and deploy any system feature without coordinating with any other subsequent teams of support or services. This involves holistic planning at the organizational level but results in quick delivery and less back-and-forth communication.

• ***Code maintenance:*** Huge organizations like Google and Facebook have millions of code written over their system by many stay-and-go developers and teams. Code maintainability has a direct positive impact and improves productivity. It lays out practices to understand and reuse different groups' or developers' code to enhance or upgrade to new systems, eventually decreasing technical depth. How well teams or developers maintain this code builds a significant and subtle performance landmark to increase an organization's performance.

• ***Monitoring:*** Productivity also regulates code review and commit process and enhances proper monitoring of the entire system as a whole. Code must be peer-reviewed and deployed in chunks after successfully testing the previous merge. This monitoring also examines code and sub-segments that increase system-predefined or finalized costs. The next level in monitoring the team's review is IT change and service management, which monitors the overall ready-to-deploy software as per industry procedures and management approvals.

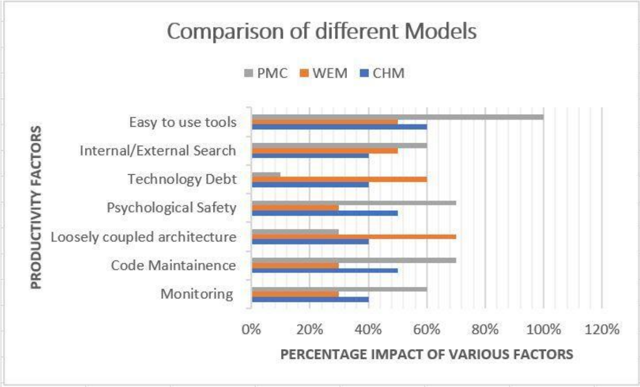

Fig. 9 PMC comparison with other models

Fig 9 shown above depicts the graphical comparison of PMC with the previous two models over the capabilities mentioned above factors. While Easy to use tools show an almost double increase in productivity in the PMC model compared to others, it also helps little with the searching mechanism used. That is why internal or external search has a positive percentage increase in PMC compared to other models. Technology debt diminishes to a significantly low percentage on the PMC model due to accessible and valuable tools compared to other models. Code maintainability, monitoring, and psychological safety factors are also on the positive and higher percentage in PMC due to advanced and accessible tools and technology selection. On the other hand, loosely coupled architecture characteristics diminish at a low rate since architecture is tightly coupled with proper planning and coordination of services and support teams.

6 Results

Let's compare traditional models with our data security method of three stages, as shown in Fig 10. We have used the RightScale Optima tool to compare these different models and values and provide quantitative performance results. The generated cryptographic text in traditional methods lauds a size of around 1000-1500 bits; however, the obfuscated text generated from the above data is secure over IoT research method size of around 2500 bits constant for every request. In other words, this provides a continuous and fixed cost of communication independent of several requests or heterogeneous users. To share any kind of collected data, traditional methods have the IoT objects doing the same work again and again while sharing the same data, which means time and cost of communication increase directly with the number of requests and users. Also, shown in the below Fig 11 Indifference to this, our PMC method on data security saves time and communication costs because of the regular size of obfuscated text irrespective of several requests and users. Assuming the number of users is the same across the different models, the cryptographic text still holds a fixed bit for every request.

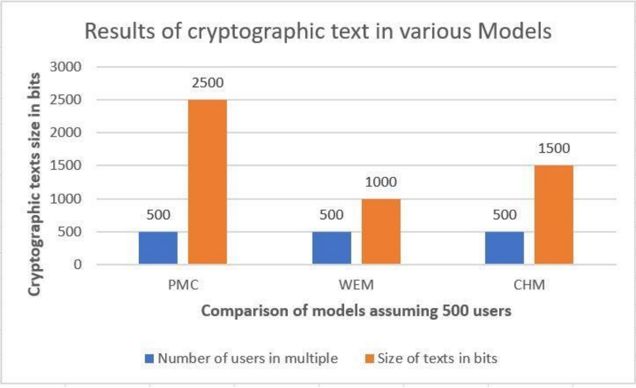

Fig. 10 Data security method results

Also, going with the cloud deployments with the help of the above-proposed PMC model, while the productivity is on the higher side, will welfare the organizations and different teams to get more work done and with higher value. The main factors which genuinely will benefit are: -

- Impact on Work Recovery
- Dealing with Burnout
- More flexible and agile
- Increasing consistency and collaboration
- Workflow improvements and downtime reduction:

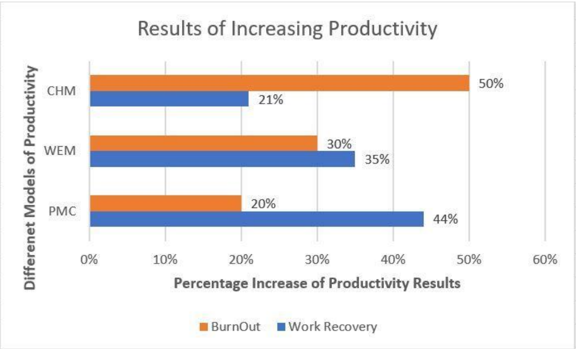

Fig. 11 Results of comparing different models

Productivity directly and positively impacts work recovery. Work recovery is the straight detachment of the individual from their work when they are out of the office or during log off-hours and giving cent percent while at work. The below factors are thus increased while work is performed in recovery mode.

- ✓ Profitability
- ✓ Market share
- ✓ Quality of products or services
- ✓ Operating efficiency
- ✓ Customer satisfaction
- ✓ Quality of products or services provided
- ✓ Achieving organizational or mission goals.

This also benefits in giving robust and elastic solutions and overall system-level outcomes as a whole rather than application-level planning. By planning and designing in little chunks and integrating them into a major one, a team can focus on developing and performing strategies at an organizational level, thus increasing individual and peers' performance over ongoing client demands.

Fig 11 shows a chart of how productivity increases results in almost 44% of work recovery in the PMC model; however, reducing cost on WEM tends to increase only 21%. On the other hand, CHM gets increased Work Recovery by around 35% over an increase in execution time. Similarly accessing the negative impact of burnout, the PMC model reduces burnout to almost 20% over productivity. However, CHM holds major burnout to 50% carrying out continuous work on execution time. On the other part, WEM has a 30% burnout ratio while utilizing limited resources and infra over cloud reduction. Organizations worldwide documented burnout as stress at a workplace that is unmanaged and long-lasting. This very well leads to inefficient performance at the workplace. The above model depreciates the burnout factor to increase productivity. It helps organizations make quick and reliable software features without touching the existing user list. Reducing burnout can improve the performances of individuals so that they can quickly and rapidly adapt to ongoing changes and shifts in technology and market trends and thus benefit the organization to achieve their desired results in multiple streams and competencies.

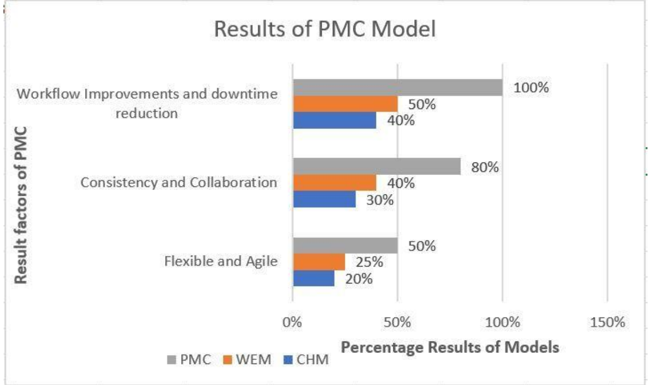

Fig. 12 Results of the PMC model

Fig. 12 also depicts workflow improvement, downtime reduction, and consistent and collaborative performance, increasing almost twice over PMC model results. It also increases agility and flexibility over work recovery and burnout to almost 1x factor. While burnout increases productivity, it also helps performers explore and shift their organization according to current market needs. They discover more options for more features or quick implementation over software pieces. Higher productivity results in cooperation between elite performers to maintain a cloud repository that can be shared and updated with other performers in different teams or organizations. This helps immensely in getting a notification on any updates or outdated software. Also, it will help me gain exposure to various client issues or business queries in less time. Productivity increase cuts down the actual time to complete a project workflow and focuses on any Disaster recovery downtime to plan on any backups. This eventually helps in business continuity planning or to deal with any kind of systems disaster.

On the other hand, Fig. 13 below shows the results of different stages of the Data Security method. For an average number of 50 users, the Data collection stage takes around 267ms to encrypt and 73ms to decrypt the ciphertext, whereas the Data access stage takes approx. 76ms to re-encrypt and 66ms to decrypt the text. Data Storage goes somewhat at the same rate as 100ms for storing the data. Public key encryption is not limited to IoT users or external components to be online constantly and is quite effective in handling large data volumes. The three stages define this key encryption and mathematical conditions to lay out their respective encryption model.

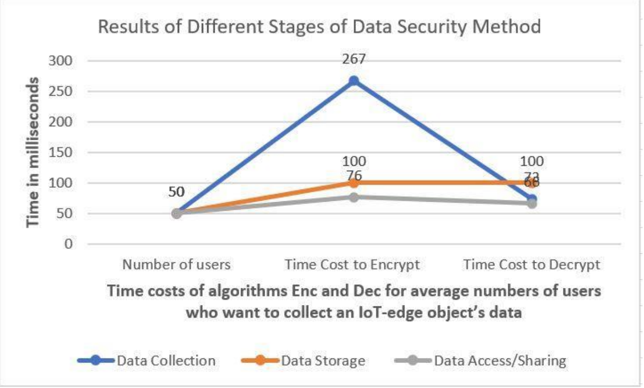

Fig. 13 Results of different stages of Data Security

7 CONCLUSION

This research focuses on three stages of Cloud assistance over IoT and produces a mathematical framework for data security over massive IoT data. This framework depicts data collection, storage, and access over the cloud and sharing over IoT users and external components.

This research also shows that productivity is always treated as a base console for getting work done in a drift. Whether the development team develops the code or other project teams test and review it, there is always a chance of improving it with the productivity mentioned above model. Whether it is automation or following best practices during the phases of a project, productivity always tends to improve the consistency and scalability of any application software.

Close coordination between deployment and development is essential to bridge gaps between infra, development, and testing teams, leading to robust and stable productivity software.

Working in today's complex and competing world, the above-said model will keep the system and employees away from any chaos and always motivate or help the organization keep the best business results and solutions.

Going to the next level and as part of future research activities, the three stages of Cloud-assisted IoT will be further integrated into a single algorithm with the elimination of some common parameters and to further eliminate redundant steps of encryption and decryption.

DECLARATIONS

Ethical Approval

There is no ethical approval required for this research.

Competing interests

There are no Competing interests involved in this research.

Authors' contributions

Mr. Prashant Vaish and Dr. Niharika Anand have done the experimentation work, Dr. Vishal along with Dr. Niharika and Mr. Prashant has written the manuscript and Mr. Gaurav has reviewed the manuscript, proofread, and prepared the figures and tables.

Funding

The authors have not received any funding for this research.

Availability of data and materials

This research doesn't include any data set which needs to be accessed.

8 References

BHUSE, VIJAY. 2014. "Security and Privacy Challenges for Healthcare Records and Wearable Sensors in Cloud." *Transaction on IoT and Cloud Computing* 2 (3): 11–17.

Bittencourt, Luiz Fernando, and Edmundo Roberto Mauro Madeira. 2011. "HCOC: A Cost Optimization Algorithm for Workflow Scheduling in Hybrid Clouds." *Journal of Internet Services and Applications* 2 (3): 207–27.

Chen, C L Philip, and Chun-Yang Zhang. 2014. "Data-Intensive Applications, Challenges, Techniques and Technologies: A Survey on Big Data." *Information Sciences* 275: 314–47.

Chen, Zong-Gan, Ke-Jing Du, Zhi-Hui Zhan, and Jun Zhang. 2015. "Deadline Constrained Cloud Computing Resources Scheduling for Cost Optimization Based on Dynamic Objective Genetic Algorithm." In *2015 IEEE Congress on Evolutionary Computation (CEC)*, 708–14. IEEE.

Durillo, Juan J, and Radu Prodan. 2014. "Multi-Objective Workflow Scheduling in Amazon EC2." *Cluster Computing* 17 (2): 169–89.

Fraj, Imen Ben, Yousra BenDaly Hlaoui, and Leila BenAyed. 2020. "A Control System for Managing the Flexibility in BPMN Models of Cloud Service Workflows." In *2020 IEEE 13th International Conference on Cloud Computing (CLOUD)*, 537–43. IEEE.

Gao, Yongqiang, Shuyun Zhang, and Jiantao Zhou. 2019. "A Hybrid Algorithm for Multi-Objective Scientific Workflow Scheduling in IaaS Cloud." *IEEE Access* 7: 125783–95.

Ghahramani, Mohammad Hossein, MengChu Zhou, and Chi Tin Hon. 2017. "Toward Cloud Computing QoS Architecture: Analysis of Cloud Systems and Cloud Services." *IEEE/CAA Journal of Automatica Sinica* 4 (1): 6–18.

Gubbi, Jayavardhana, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. 2013. "Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions." *Future Generation Computer Systems* 29 (7): 1645–60.

Heilig, Leonard, and Stefan Voß. 2014. "A Scientometric Analysis of Cloud Computing Literature." *IEEE Transactions on Cloud Computing* 2 (3): 266–78.

Hilman, Muhammad H, Maria A Rodriguez, and Rajkumar Buyya. 2020. "Multiple Workflows Scheduling in Multi-Tenant Distributed Systems: A Taxonomy and Future Directions." *ACM Computing Surveys (CSUR)* 53 (1): 1–39.

Jagaty, Pratyush, Sampa Sahoo, Dimple Patel, and Bibhudatta Sahoo. 2020. "Priority Queue Based Real-Time Task Scheduling in Virtualized Cloud Environment." In *2020 7th International Conference on Signal Processing and Integrated Networks (SPIN)*, 862–66. IEEE.

Jia, Ya-Hui, Wei-Neng Chen, Huaqiang Yuan, Tianlong Gu, Huaxiang Zhang, Ying Gao, and Jun Zhang. 2018. "An Intelligent Cloud Workflow Scheduling System with Time Estimation and Adaptive Ant Colony Optimization." *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.

Jiang, Chun, and Jiafu Wan. 2021. "A Thing-Edge-Cloud Collaborative Computing Decision-Making Method for Personalized Customization Production." *IEEE Access* 9: 10962–73.

Jung, Eun-Sung, and Rajkumar Kettimuthu. 2014. "Challenges and Opportunities for Data-Intensive Computing in the Cloud." *Computer* 47 (12): 82–85.

Lakhan, Abdullah, Mazin Abed Mohammed, Ahmed N Rashid, Seifedine Kadry, Karrar Hameed Abdulkareem, Jan Nedoma, Radek Martinek, and Imran Razzak. 2022. "Restricted Boltzmann Machine Assisted Secure Serverless Edge System for Internet of Medical Things." *IEEE Journal of Biomedical and Health Informatics* 27 (2): 673–83.

Naha, Ranesh Kumar, Saurabh Garg, Dimitrios Georgakopoulos, Prem Prakash Jayaraman, Longxiang Gao, Yong Xiang, and Rajiv Ranjan. 2018. "Fog Computing: Survey of Trends, Architectures, Requirements, and Research Directions." *IEEE Access* 6: 47980–9.

Nieuwenhuis, Lambert J M, Michel L Ehrenhard, and Lars Prause. 2018. "The Shift to Cloud Computing: The Impact of Disruptive Technology on the Enterprise Software Business Ecosystem." *Technological Forecasting and Social Change* 129: 308–13.

Pandey, Ashish, Songjie Wang, and Prasad Calyam. 2019. "Data-Intensive Workflow Execution Using Distributed Compute Resources." In *2019 IEEE 27th International Conference on Network Protocols (ICNP)*, 1–2. IEEE.

Rimal, Bhaskar Prasad, and Martin Maier. 2016. "Workflow Scheduling in Multi-Tenant Cloud Computing Environments." *IEEE Transactions on Parallel and Distributed Systems* 28 (1): 290–304.

Rodriguez, Maria Alejandra, and Rajkumar Buyya. 2014. "Deadline Based Resource Provisioningand Scheduling Algorithm for Scientific Workflows on Clouds." *IEEE Transactions on Cloud Computing* 2 (2): 222–35.

Shi, Jie, Tianbing Zhang, Songlin Wang, Boya Deng, Gangyong Jia, and Guangjie Han. 2019. "A New Task Scheduling for Minimizing Completion Time and Execution Cost in Smart Grid Cloud." In *2019 Computing, Communications and IoT Applications (ComComAp)*, 151–56. IEEE.

Singh, Jatinder, Thomas Pasquier, Jean Bacon, Hajoon Ko, and David Eyers. 2015. "Twenty Security Considerations for Cloud-Supported Internet of Things." *IEEE Internet of Things Journal* 3 (3): 269–84.

Sousa, Erica, Fernando Lins, Eduardo Tavares, Paulo Cunha, and Paulo Maciel. 2014. "A Modeling Approach for Cloud Infrastructure Planning Considering Dependability and Cost Requirements." *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45 (4): 549–58.

Stamford, C. 2019. "Gartner Forecasts Worldwide Public Cloud Revenue to Grow 17% in 2020."

Wan, Jiafu, Baotong Chen, Muhammad Imran, Fei Tao, Di Li, Chengliang Liu, and Shafiq Ahmad. 2018. "Toward Dynamic Resources Management for IoT-Based Manufacturing." *IEEE Communications Magazine* 56 (2): 52–59.

Wang, Wanyuan, Yichuan Jiang, and Weiwei Wu. 2016. "Multiagent-Based Resource Allocation for Energy Minimization in Cloud Computing Systems." *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 47 (2): 205–20.

Wang, Wei, Peng Xu, and Laurence Tianruo Yang. 2018. "Secure Data Collection, Storage and Access in Cloud-Assisted IoT." *IEEE Cloud Computing* 5 (4): 77–88.

Prashant Vaish is pursuing a Ph.D. degree from the Indian Institute of Information Technology, Lucknow, India. He received his M.Tech in Computer Science & Engg. from Glocal University, Saharanpur, India, and his Bachelor's degree from Uttar Pradesh Technical University, Lucknow, India. His areas of research include DevOps, Cloud Computing, and IoT over the Cloud.

Dr. Niharika Anand received a Ph.D. degree from the Indian Institute of Information Technology, Allahabad, India. She is currently working as an Assistant Professor with the Department of Information Technology Indian Institute of Information Technology, Lucknow, India. Her areas of research include Cloud Computing, the Internet of Things, Cyber Forensics 3-D Wireless Sensor Networks, Wireless Sensor Network Localization, and Wireless Sensor Network Topology Control and Maintenance.

Dr.Vishal Krishna Singh received a bachelor's degree in Information Technology, in 2010, a master's degree in Computer Technology and Application from the National Institute of Technical Teachers Training and Research, Bhopal, India in 2013, and a Ph.D. degree in Information Technology from Indian Institute of Information Technology, Allahabad, India in 2018. During his Ph.D., he published several papers in International Journals of repute and presented many papers at international conferences. He was Assistant Professor in the Department of Computer Science, Indian Institute of Information Technology, Lucknow, India, and now works as a Lecturer in the School of Computer Science and Electronics Engineering, University of Essex,Colchester, U.K . His research interests include compressed sensing, in-network inference, Markov random field, wireless sensor networks, matrix completion algorithms, the Internet of Things, and data analytics.

Gaurav Sharma received his bachelor's degree in computer science from Gautama Buddha University, Lucknow, India in 2010, and his master's degree in VLSI and CAD systems from Thapar University Punjab, India in 2012. He is currently working as a Visiting Lecturer and pursuing a Ph.D. in the cyber security research group at the University of Hertfordshire Hatfield, UK- School of Computer Science since 2016 focusing on Real-Time Semi-Automated Threat Assessments in Informational Environment.

Data Availability Statement: Data sharing does not apply to this article as no datasets were generated or analyzed during the current study.