Journal Publication of International Research for Engineering and Management (JOIREM)
Volume: 04 Issue: 07 | July-2026
ISSN (O) 3107-6696

TRAC: AN INTELLIGENT DECOUPLED IAM FRAMEWORK FOR SECURE CLOUD INFRASTRUCTURE

Vikas Dubey[1*], Anupam Chouksey[2]

[1]Computer Science Engineering, LNCT Univeristy Bhopal, Madhya Pradesh
[2]Computer Science Engineering, LNCT Univeristy Bhopal, Madhya Pradesh

---

Abstract - The foundation of cloud infrastructure is made up of cloud access control protocols. Industries have been taught to utilize role-based access to provide people permissions for decades. Despite decades of development, role explosions that lead to excessive user privileges continue to plague cloud identity and access management systems. The entire cloud infrastructure may be at risk due to this issue. This study presents an ultra-modern framework based on the fundamental ideas of decoupling and dynamic job assignment to address this problem. Instead of considering roles, the suggested framework bases user access on the tasks that users must do.

Keywords: R*BAC FRAMEWORK, ROLE EXPLOSION, DYNAMIC MODEL, DECOUPLING, JIT ACCESS.*

1. Introduction

In the last few decades, several access protocols have been developed for identity and access management in cloud environments. Nevertheless, role explosion remains a critical and common problem. Based on the requirements, users can create multiple roles. These roles are at most identical most of the time leads to a serious situation. System experience a spagete of roles where who accesses what is not clear leading to providing extra privileges to many users. Cloud systems are used worldwide in several planet-size applications, leading to serious consequences.

Roles are created according to the project and are then abandoned once the project is over or the employee leaves the organization. There is no clear ownership of the lifecycle of a role. Admins are not looking into the big picture; their quickest solution to any problem is to create a new role.

Although it immediately fixes the issue, it incurs a long-term technical debt. There are a number of circumstances in which it is necessary to grant an existing role additional privileges; in these circumstances, the administrator copies the position and grants additional privileges. You will encounter role explosion if you repeat the same action hundreds of times.

This architecture offers a cutting-edge solution to this fundamental problem. The roles are created during runtime and are only necessary for the task to be completed; there is no permanent role. In this case, roles are merely the logical reusable containers for the collection of tasks; access is determined by the work being completed rather than by the role. The dynamic context is provided by attribute tags on resources and users. Just-in-time access is provided for a particular task and time frame.

2. Literature Survey

The role explosion that endangers the entire IT infrastructure is the deepest issue still present in RBAC-based cloud systems, according to this research. Additionally, it offers a sophisticated framework solution to the role explosion problem by utilizing several computer science concepts and techniques to regulate cloud environment behavior.

Research Inquiries:

Q1. To what extent may the RBAC be used in a hybrid mode?

Q2. Is it possible to eliminate the role explosion issue from RBAC?

Q3. Can the goal be achieved with a new framework?

1.1 Structure of the Paper:

The issues surrounding RBAC are brought up in the First Section. The role explosion issue is explained in detail in the second section, along with potential solutions. The paper's third section discusses a number of strategies and ideas that can effectively combat role explosion. Decoupling, Just-In-Time Access, Dynamic Model, Attribute Tag, and Smart Policies are some of the ideas. This section aids in comprehending how these ideas work together to resolve IAM problems. The Ultrmordern TRAC Framework is suggested in the Fourth Section as a solution to the role explosion issue.

Although role-based access is a popular approach due to its ease of use, it highlights the issue of role explosion. This paper also proposes that using RBAC requires an appropriate design. Selamat, Siti Rahayu [2] discusses the critical issue of role mining and proposes a thorough review. It is also important since it talks about changing and reassigning user roles in accordance with permissions.

After experimenting with several models, Uddin [3] found that using them is still difficult. Additionally, it highlights how crucial dynamic access control is to enhancing current models. RBAC requires more granular control in healthcare systems because a single user performs numerous jobs, according to De Carvalho and Marcelo Antonio [4]. In this instance, Hlushchenko [5] highlights the context-based access paradigm in light of the complexity of contemporary cloud computing.

RBAC problems with scalability and rigidity in the dynamic context are noted by Ema, Oye [6]. According to this paper [7], role explosion can be controlled and the RBAC can be applied in medium-to large-scale systems. A zero trust solution that highlights the usefulness of hybrid RBAC models for dynamic and scalable environments was introduced by Adetayo [8]. According to these references, if the RBAC was combined with the new technology, its potential was virtually limitless.

A dynamic and network-adaptable framework was developed by Asim [9]. Additionally, Habibi [10] recommended upgrading outdated cloud models for increased performance and scalability. To improve cloud authentication, Zhang [11] used an RBAC Trust Based Model. A hyperledger-enabled RBAC architecture was presented by Usman [12] to improve scalability and efficiency.

In order to offer security and scalability in a dynamic context, this paradigm [3] improves RBAC with Dynamic Access Control. When necessary, Wang [13] found a lack of dynamic access authority. In order to improve cloud performance, this paper [14] uses the container as a service strategy. Zhang [11] came to the conclusion that none of the frameworks successfully combined access control, security, and trust.

To improve flexibility and scalability in cloud systems, Ding [15] introduces a decoupled layer with fault-tolerant features. This study also challenges the existing system's strong connection between design and execution code, which can be overcome by using a decoupled layer.

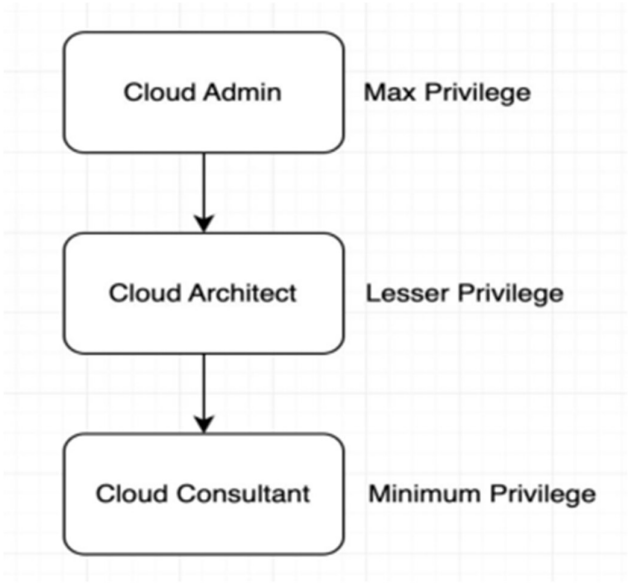

Fig 1. Smart Policies

Yang [16] proposes an event-driven cloud-centric system that dynamically creates policies based on occurrences after realizing that fixed rules are inappropriate for the dynamic, changing cloud environment. In order to accomplish global route optimization, Chungang [17] created a novel network structure using decoupling concepts.

According to Peng [18], traditional systems are unsuitable for scientific computations; hence, cloud environments combined with JIT are best suited to support dynamic computing requirements only when necessary. Sen Ma [19] discovered a novel approach that enables programmers to build hardware accelerators without requiring a thorough understanding of hardware. After identifying a problem with the Just In Time compiler, Siri [20] improved its design to boost efficiency.

Aniket [21] pointed out that JIT optimization is necessary in other architectural designs to enhance performance, and that modern frameworks, particularly.NET, are not thoroughly explored. Therefore, it was determined that a Just-In-Time compiler is required to meet the demands of cloud systems in a dynamic environment.

3. Method

The primary issue with RBAC is role explosion [7]. Applying cloud ideas such as Decoupling, Just-in-Time Access, Task Based Access, Smart Policies, and Dynamic Model can enhance RBAC [10], [22]. These modifications to the RBAC

Journal Publication of International Research for Engineering and Management (JOIREM)
Volume: 04 Issue: 07 | July-2026
ISSN (O) 3107-6696

Model can enhance its fundamental characteristics by transforming it from a static to a dynamic model that can adapt to changes and function well in a scalable context.

Smart policies that operate in a hierarchy from higher to lower privileges are required. The system will always give the child node, which has control over the generation of new roles, less authority.

By transferring complex reasoning from roles into intelligent policy rules, smart policies [23] aid in reducing role explosion. Smart rules assess contextual factors like time, device, user identity, project tag, or job function at runtime rather than generating several special-purpose roles (such as "Temporary Reviewer," "Weekend Analyst," or "Project-X Developer") [19], [21]. This eliminates the requirement for dozens of precisely defined jobs because one position can act differently depending on policy conditions.

By granting users temporary access only when necessary, Just-In-Time (JIT) Access further reduces role expansion. For numerous uncommon jobs that employees may occasionally undertake, permanent positions are created in traditional RBAC.

These permissions are only given for the necessary amount of time with JIT access [3], [23], after which they immediately expire. This helps businesses retain fewer long-lived roles by preventing the establishment of additional roles like "DBAdmin_Temporary" or "Emergency_Access_DevOps."

By modifying permissions based on current circumstances rather than predetermined static roles, a dynamic access model also significantly contributes to the prevention of role explosion. Users may be part of several teams, projects, or workloads that are always changing in cloud-native systems. A small core set of roles can adapt to new scenarios without constantly establishing new ones since dynamic models use metadata (projects, workload labels, time, risk score, etc.) to give rights at runtime [23, 24]. This prevents a scenario in which each project or setting necessitates a different set of roles.

Lastly, by dividing identities, roles, policies, and resources into separate layers, decoupling [25] in cloud computing lessens role explosion. Permissions and identities are too strongly linked in tightly coupled on-premises RBAC systems, which causes the role structure to grow quickly. Decoupling on cloud platforms enables the reuse of a single role across several services, and resource-specific access is managed by distinct policies [27][26] instead of distinct roles. Every new microservice, region, or environment is prevented from requiring its own role variant by this modular architecture [28][27].

In order to create a solid and dependable Decoupled Architecture, the suggested approach found the strengths in many IT concepts [25]. It separates the system into three main components:

Roles, privileges, and resources are defined by the system. The system's brain, the rule engine, aids in enforcing rules at a fine level in a dynamic setting [26]. It assigns a task for a set amount of time and creates rules at runtime. In addition to its tagged qualities and resources, access is the condition that verifies the smart policies [29], [30], and [28]. Access to the system is approved if the policy is met; otherwise, it is denied.

The whole system is decoupled in a way to achieve performance, efficiency and scalability. This cloud framework is simple yet robust, helps in accessing the cloud resources to the end users as per role and smart policy applied at runtime. The rule engine will apply the rules at runtime. This system design will reduce the role explosion, over privilege issues and mismatch during access.

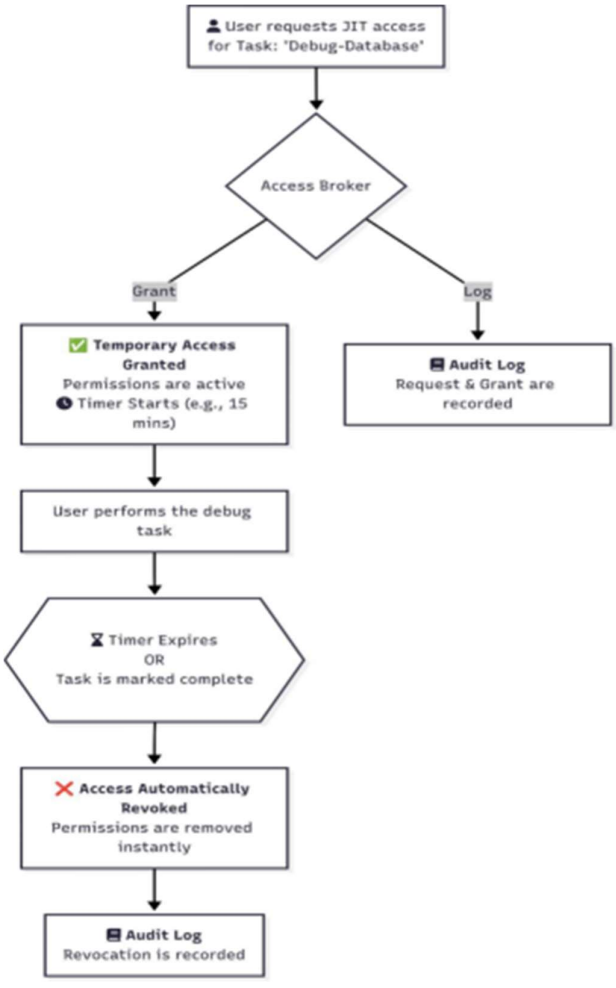

Fig 2. Just-In-Time Compiler

Journal Publication of International Research for Engineering and Management (JOIREM)
Volume: 04 Issue: 07 | July-2026
ISSN (O) 3107-6696

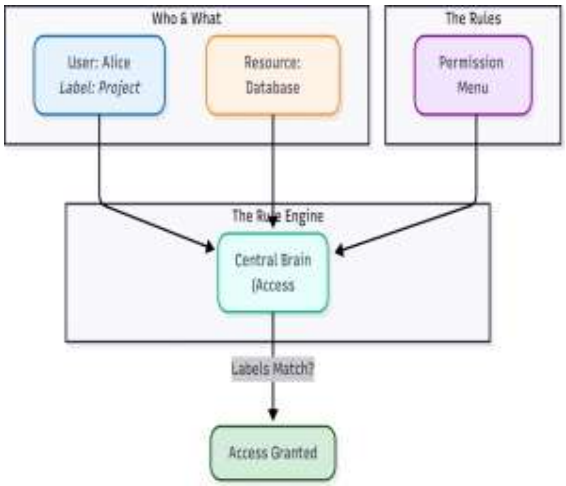

Fig 3. Decoupled Architecture

The solution to Role explosion is the TRAC Framework which firstly identifies the user, resource, action and environment. Secondly, it create a policy rule at runtime. Thirdly, allow or deny the access as per the context[12], [25], [31][29].

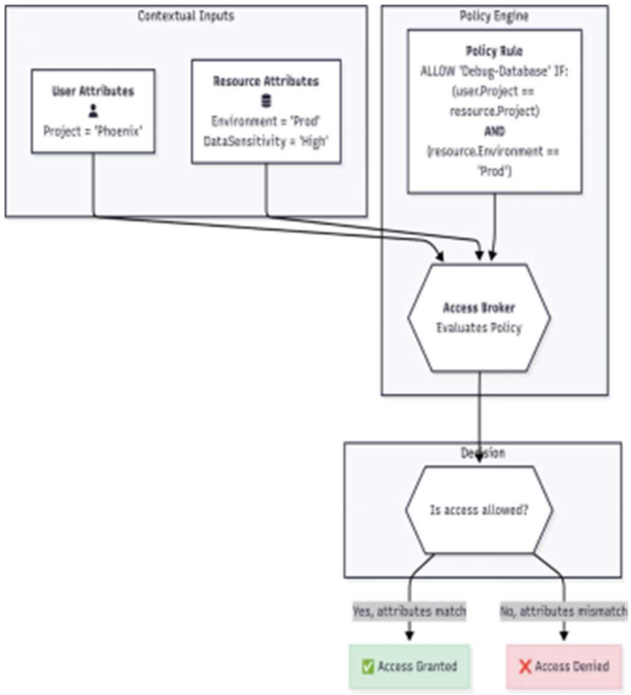

Fig 4. TRAC Framework

This section of the paper, explains how multiple technologies helps in building the appropriate model. The model is validated using the best practices used in software engineering like Design Mapping Validation, Scenario Based Validation, Comparative Validation Against Existing Models[30], [31].

4. Design Validation

1. This method gives the proof-of-work, how the design architecture validating the requirements.

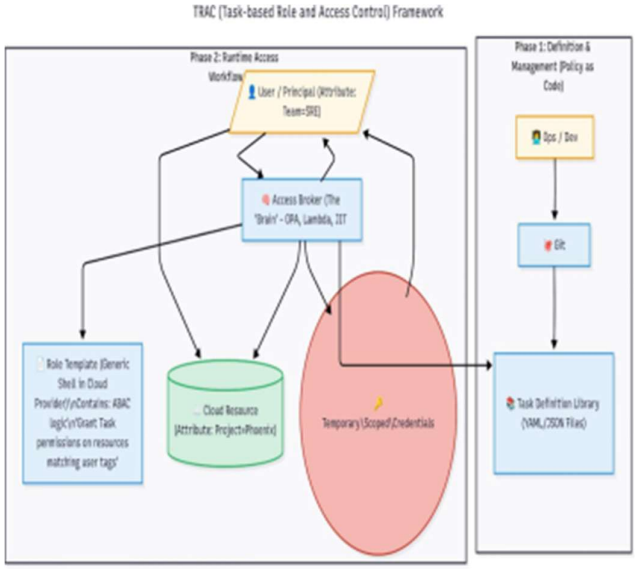

Fig 5. TRAC Framework Design Model

2. Explore the realistic A scenario is created to illustrate the usefulness of this model, as shown in Fig. 5. It aids in describing the TRAC Framework's operational viability. Situation: The SRE engineer needs short-term access to project resources.

Step 1: For a particular operational task, the user (Team=SRE) asks access.

Step 2: The Access Broker intercepts the request and collects the following: User attributes, Task definition from the Task Definition Library, and Resource attributes (Project=Phoenix). Step 3: OPA uses the Role Template's smart policy logic to evaluate policies.

Step 4: Temporary scoped credentials are issued following approval.

Step 5: When the operation is finished or the timeout occurs, access automatically ends.

Journal Publication of International Research for Engineering and Management (JOIREM)
Volume: 04 Issue: 07 | July-2026
ISSN (O) 3107-6696

Design Objective	Architectural Element	Validation Logic
Reduce role explosion	Role Template (Generic Shell)	Roles are no longer static; permissions are assigned via task definitions and attributes
Support dynamic access	Access Broker (OPA, Lambda, JIT)	Access decisions are made at runtime using context
Enforce least privilege	Temporary Scoped Credentials	Credentials are time-bound and task-scoped
Enable DevOps governance	Git + Task Definition Library	Policies are versioned and auditable
Cloud-native compatibility	ABAC logic in role template	Attributes align with cloud IAM primitives

Table 1. Requirement to Design Mapping Validation Validation Outcomes:

a. No static role assignment
b. No standing privileges
c. No manual intervention

2. Comparative Validation:

Model	Static Roles	Context-Aware	JIT Access	Role Explosion Risk
RBAC	Yes	No	No	High
ABAC	No	Yes	No	Medium
PBAC	Partial	Limited	Yes	Medium
TRAC (Proposed)	No	Yes	Yes	Low

Table 2. Comparative Validation: TRAC, RBAC, ABAC & PBAC

4. This design is proven secure due to the below reasons:

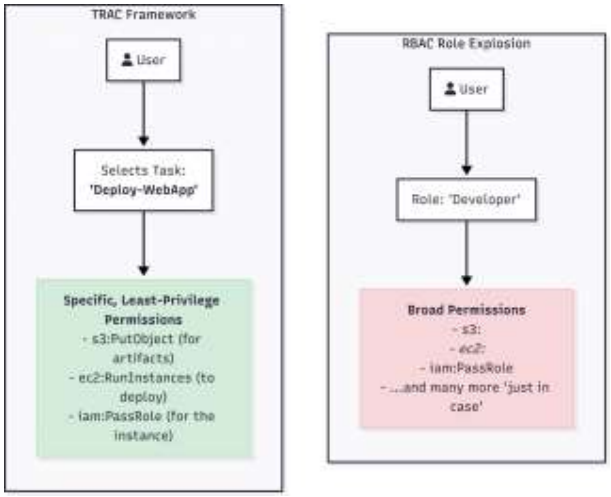

Fig 8. Control Flow & Trust-Boundary Validation

Access Broker serves as a single point of policy decision-making. Lack of direct access to cloud resources

a. Credentials include:
b. Temporary
c. Task-oriented
d. Validated attributes

The policy generating process with the Access Broker is isolated by the decoupled design concept. Before granting access, it also assesses and compares the attributes. Permit temporary access and revoke it in accordance with policy. These characteristics all contribute to the design's authenticity, durability, and dependability.

5. Discussion

The TRAC Framework's robustness, security, dependability, and scalability are all made abundantly evident in the validation part. Decoupling, Just-In-Time Access (JIT), Smart Policies, Attribute Tags, Dynamic Model, and Task Based Access are just a few of the many computer science engineering ideas that are cleverly used in this ultra-modern architecture. Coupling, JIT, and task-driven access are all successfully applied by the TRAC. TRAC are dynamically assessed at runtime, in contrast to RBAC. When the RBAC system has role explosion, TRAC eliminates it by giving derived classes the bare minimum of privileges.

The suggested framework is compatible with DevOps, where rules and policies are implemented as a CAAS. GIT oversees

the task definition library in order to provide version control. By using an access broker that puts security ahead of latency, TRAC also enforces flexibility and security.

6. Conclusion

The TRAC Framework, which aims to provide security, scalability, and performance, was emphasized in this study. It discusses how to solve RBAC's role explosion issue. The system must overcome the role explosion issue by applying ideas like decoupling, task-driven authorization, just-in-time access, dynamic access, and smart policies.

Architectural, scenario-based, and comparative analysis were used to validate the framework, and the outcome showed that the TRAC framework can be readily integrated with any contemporary cloud/devops environment. The system achieves flexibility, auditability, and security at runtime thanks to the decoupled components, which are revolutionary. There is no need to reconfigure the ultra-modern system. Because of this, the system is appropriate for a contemporary multi-tenant cloud environment.

7. Acknowledgements

The author gratefully acknowledges the support provided by the Department of Computer Science, PIET, Parul University, for providing necessary facilities and research infrastructure.

References:

[1] Z. Asaf, M. Asad, S. Ahmed, W. Rasheed, and T. Bashir, "Role based access control architectural design issues in large organizations," in Proc. Int. Conf. Open-Source Syst. Technol. (ICOSST), Lahore, Pakistan, Dec. 2014, pp. 197–205, doi: 10.1109/ICOSST.2014.7029344.

[2] S. R. Selamat, "Publications and citations profile," 2017.

[3] M. Uddin, S. Islam, and A. Al-Nemrat, "A dynamic access control model using authorising workflow and task-role-based access control," IEEE Access, vol. 7, pp. 166676–166689, 2019, doi: 10.1109/ACCESS.2019.2947377.

[4] M. A. De Carvalho and P. Bandiera-Paiva, "Evaluating ISO 14441 privacy requirements on RBAC restrict mode via colored Petri nets modeling," in Proc. Int. Carnahan Conf. Security Technol. (ICCST), Madrid, Spain, Oct. 2017, pp. 1–8, doi: 10.1109/CCST.2017.8167833.

[5] P. Hlushchenko and V. Dudykevych, "Exploratory survey of access control paradigms and policy management engines," (no publication details available).

[6] O. Emma and P. Peace, "Role-based access control (RBAC) enhancements for big data," (no publication details available).

[7] A. Elliott and S. Knight, "Towards managed role explosion," in Proc. New Security Paradigms Workshop, Twente, Netherlands, Sept. 2015, pp. 100–111, doi: 10.1145/2841113.2841121.

[8] A. Adeyinka, "Zero trust architectures in multi-tenant cloud platforms: A role-based access control reinforcement framework," Int. J. Innov. Res. Comput. Commun. Eng., vol. 12, no. 5, May 2024, doi: 10.15680/IJIRCCE.2024.1205372.

[9] M. Asim et al., "SecT: A zero-trust framework for secure remote access in next-generation industrial networks," IEEE J. Sel. Areas Commun., vol. 43, no. 6, pp. 2293–2311, June 2025, doi: 10.1109/JSAC.2025.3560015.

[10] P. Habibi and A. Leon-Garcia, "SliceSphere: Agile service orchestration and management framework for cloud-native application slices," IEEE Access, vol. 12, pp. 169024–169049, 2024, doi: 10.1109/ACCESS.2024.3492138.

[11] J. Zhang, T. Li, Z. Ying, and J. Ma, "Trust-based secure multi-cloud collaboration framework in cloud-fog-assisted IoT," IEEE Trans. Cloud Comput., vol. 11, no. 2, pp. 1546–1561, Apr. 2023, doi: 10.1109/TCC.2022.3147226.

[12] M. Usman et al., "A blockchain-based scalable domain access control framework for industrial Internet of Things," IEEE Access, vol. 12, pp. 56554–56570, 2024, doi: 10.1109/ACCESS.2024.3390842.

[13] H. Wang et al., "PAC-MC: An efficient password-based access control framework for time sequence aware media cloud," IEEE Trans. Mobile Comput., vol. 24, no. 7, pp. 5632–5648, July 2025, doi: 10.1109/TMC.2025.3534861.

[14] B. C. Şenel et al., "Multitenant containers as a service (CaaS) for clouds and edge clouds," IEEE Access, vol. 11, pp. 144574–144601, 2023, doi: 10.1109/ACCESS.2023.3344486.

[15] Z. Ding et al., "SCAFE: A service-centered cloud-native workflow engine architecture," IEEE Trans. Serv. Comput.,

vol. 16, no. 5, pp. 3682–3695, Sept. 2023, doi: 10.1109/TSC.2023.3259989.

[16] Q. Yang et al., "A service-based cloud-edge fusion approach for abnormality detection of power generation equipment," IEEE Access, vol. 12, pp. 51556–51569, 2024, doi: 10.1109/ACCESS.2024.3386189.

[17] C. Yan and S. Sheng, "SDN+K8s routing optimization strategy in 5G cloud-edge collaboration scenario," IEEE Access, vol. 11, pp. 8397–8406, 2023, doi: 10.1109/ACCESS.2023.3237201.

[18] P. Zhang, Y. Liu, and M. Qiu, "SNC: A cloud service platform for symbolic-numeric computation using just-in-time compilation," IEEE Trans. Cloud Comput., vol. 7, no. 2, pp. 580–592, Apr. 2019, doi: 10.1109/TCC.2017.2656088.

[19] S. Ma et al., "Breeze computing: A just-in-time approach for virtualizing FPGAs in the cloud," in Proc. Int. Conf. Reconfigurable Computing and FPGAs, Cancun, Mexico, Nov. 2016, pp. 1–6, doi: 10.1109/ReConFig.2016.7857159.

[20] S. S. Ponangi et al., "Java runtime optimization for copying arrays on AArch64," in Proc. Mediterranean Conf. Embedded Computing (MECO), Budva, Montenegro, June 2023, pp. 1–6, doi: 10.1109/MECO58584.2023.10155064.

[21] A. Deshmukh et al., "Performance characterization of .NET benchmarks," in Proc. IEEE Int. Symp. Performance Analysis of Systems and Software (ISPASS), Stony Brook, NY, USA, Mar. 2021, pp. 107–117, doi: 10.1109/ISPASS51385.2021.00028.

[22] V. Tsakanikas and T. Dagiuklas, "A generic framework for deploying video analytic services on the edge," IEEE Trans. Cloud Comput., vol. 11, no. 3, pp. 2614–2630, July 2023, doi: 10.1109/TCC.2022.3218813.

[23] G. Fragkos et al., "Dynamic role-based access control policy for smart grid applications: An offline deep reinforcement learning approach," IEEE Trans. Human-Machine Systems, vol. 52, no. 4, pp. 761–773, Aug. 2022, doi: 10.1109/THMS.2022.3163185.

[24] J. Gupta, K. Kant, and A. Abouelwafa, "FussyCache: A caching mechanism for emerging storage hierarchies," in Proc. IEEE Int. Conf. Cloud Computing Technology and Science (CloudCom), Bangkok, Thailand, Dec. 2020, pp. 74–81, doi: 10.1109/CloudCom49646.2020.00010.

[25] H. Deng et al., "EP-Net: Improving point cloud learning efficiency through feature decoupling," IEEE Trans. Instrum. Meas., vol. 73, pp. 1–14, 2024, doi: 10.1109/TIM.2024.3451587.

[26] F. Luo et al., "Key-policy attribute-based encryption with switchable attributes for fine-grained access control of encrypted data," IEEE Trans. Inf. Forensics Security, vol. 19, pp. 7245–7258, 2024, doi: 10.1109/TIFS.2024.3432279.

[27] S. T. Alshammari et al., "Trust management systems in cloud services environment: Taxonomy of reputation attacks and defense mechanisms," IEEE Access, vol. 9, pp. 161488–161506, 2021, doi: 10.1109/ACCESS.2021.3132580.

[28] Z. Fan et al., "Enhancing security in cloud computing: A comprehensive analysis of a zero-trust dynamic access control architecture," in Proc. Int. Conf. Networking Systems of AI (INSAI), Xi'an, China, Nov. 2023, pp. 275–285, doi: 10.1109/INSAI60116.2023.00057.

[29] L. Fen and L. Quan, "Digital right management based on cloud computing and dynamic secure permission," in Proc. Int. Conf. Consumer Electronics, Communications and Networks (CECNet), Xianning, China, Apr. 2011, pp. 3091–3094, doi: 10.1109/CECNET.2011.5768311.

[30] E. Chen et al., "Secure remote cloud file sharing with attribute-based access control and performance optimization," IEEE Trans. Cloud Comput., vol. 11, no. 1, pp. 579–594, Jan. 2023, doi: 10.1109/TCC.2021.3104323.

[31] R. Andreoli et al., "A multi-domain survey on time-criticality in cloud computing," IEEE Trans. Serv. Comput., vol. 18, no. 2, pp. 1152–1170, Mar. 2025, doi: 10.1109/TSC.2025.35.