The Eastasouth Journal of Information System and Computer Science
Vol. 2, No. 02, December 2024, pp. 364-376
ISSN: 3025-566X, DOI: 10.58812/esiscs.v2i02

Policy-Evolving IAM Architecture for Just-in-Time Privilege Control in Multi-Account Cloud Enterprises

Kedar Pradhan
Solutions Architect, Amazon Web Services, Austin, USA

Article Info

Article history:

Received Dec, 2024
Revised Dec, 2024
Accepted Dec, 2024

Keywords:

Attribute-Based Access Control;
Formal Verification;
Identity and Access Management;
Just-in-Time Privilege;
Least Privilege;
Multi-Account Cloud;
Policy Mining;
Privilege Escalation;
Zero Trust Architecture

ABSTRACT

Multi-account cloud enterprises spread workloads across multiple accounts, each of which is managed separately, but privileged access is often given to users within these accounts as default and is built up over time increasing the attack surface. This systematic review combines 23 studies and standards into a policy-evolving identity and access management (IAM) architecture for the following: risk-based access control, formal policy verification, identity federation, just-in-time (JIT) elevation, role and attribute mining, multi-tenancy authorization, and zero trust architecture. The proposed design is to perform risk-scoring on each elevation request, validate it with a trust-safety property using satisfiability modulo theories (SMT) reasoning and issue a per-session credential that only lasts long enough for a specific session; and record the credential in a log for log-driven policy mining, followed by regression tests to test the candidates before promotion. A formal check of the results shows that the tests are sufficiently quick to be used in an inline mode, with 99% of the production policy questions answered within 160 milliseconds; in addition, the tests are able to handle a task that the other algorithms were not able to handle within 24 hours, while the attribute based miners scored an area under the curve of 0.9973, compared to 0.9269 for the role based miners. The use of a modeled scenario further suggests that four two-hour JIT sessions a week provide 95.3% less privilege exposure than standing access, over the course of the year. Approval fatigue, proportionality monitoring, scalability of mining, and continuity of break glass are mentioned as the major challenges to adoption.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Name: Kedar Pradhan
Institution: Solutions Architect, Amazon Web Services, Austin, USA
Email: kedar_pradhan90@yahoo.com

1. INTRODUCTION

Multi-account cloud enterprises are organizations that share workloads, data and administration across many individually-managed cloud accounts and/or cloud subscriptions, with each cloud boundary limiting the geographical reach of compromised data. In these estates, it's common for privileges to be granted as an ongoing privilege, and retain that status long after the triggering task has completed. Dynamic access control has been proposed as a risk-based approach based on contextual and historical evidence before giving an access decision for each access request [1]. As

shown in Figure 1, risk history was the most common factor used (18 studies) while no explicit estimation procedure was included in 18 studies in the systematic screening procedure of 1,044 records that resulted in 44 primary studies. The scoring process is abstracted in Equation (1): L is the probability that the subject s can perform the action a in a specific context c and that the action is allowed to be performed if its score is below the threshold θ.

$$R(s, o, a, c) = L(s, a, c) \cdot I(o, a) \leq \theta \quad (1)$$

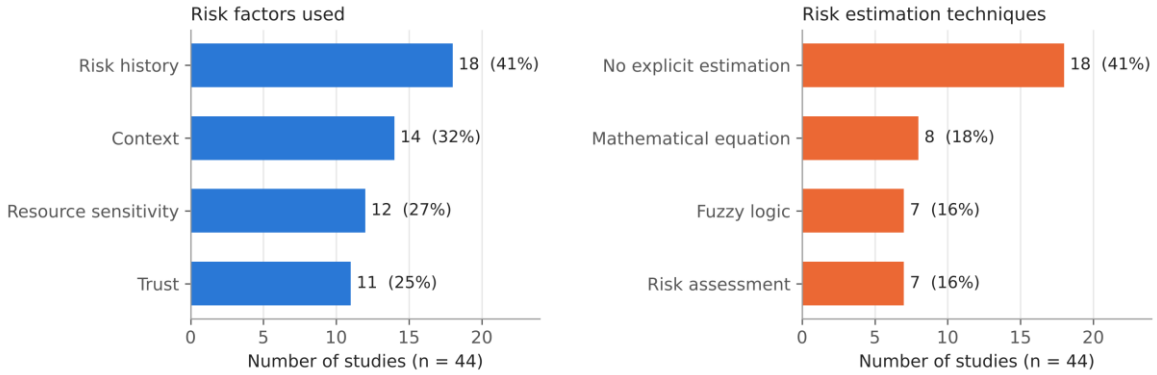


Figure 1. Distribution of Risk Factors and Risk Estimation Techniques Across 44 Primary Studies

With all the combinations of conditions, wildcards, and cross-account principals tossed in with policies, it can be hard to understand the semantics of it all in a declarative language that's used to record entitlements. An automated reasoning engine was therefore designed that transforms access policies into satisfiability modulo theories (SMT) formulas, so that answers to such questions as 'is one policy more permissive than another?', can be provided mathematically, rather than by inspection [2]. 99% of production policy questions were answered in less than 160ms, and production storage, configuration and threat-detection services used the engine millions to tens of millions of times per day. Then formal checks can be carried out in an elevation workflow without any perceptible delay.

The second challenge is with regard to how least-privilege roles are generated. If the number of permissions a role may include, the number of roles a user may include, or the number of users that can include a role is limited, then role mining is very difficult, and heuristic procedures are designed that do respect such limitations in real data where roles are discovered from existing user–permission assignments [3]. In this context, 23 studies are summarized as a policy forming evolving identity and access management (IAM) architecture in which Just In Time (JIT) elevation, formal verification and policy mining based on logs are in one closed loop. In section 2 literature review is performed, and in section 3, the method is described. The Architecture and comparative results are given in Section 4 and the conclusions are drawn in Section 5.

2. LITERATURE REVIEW

2.1 Formal Verification and Misconfiguration Risk

Formal verification from going from policy comparison to more powerful property of trust safety, where a policy is accepted only if all the requests it permits are made by principals in a specified trusted set [4]. The checking is exhaustive, because this is PSPACE-complete; each policy is used to make the check a very small residual policy (just checking for untrusted access), and in the common case, the property is established without a solver call. All control is programmed into the object-storage authorization, and millions of policies are checked every single day to prevent open access by mistake. The property is stated in

Equation (2) where R is the request space, π(r) is the requesting principal and T is the trusted set.

$$\forall r \in \mathcal{R}: \text{Allow}(\text{P}, \text{r}) \Rightarrow \pi(r) \in \mathcal{T} \qquad (2)$$

Empirical research on the topic of identity misconfiguration gives further empirical weight to its practical importance. The following two escalation paths were demonstrated in a controlled environment of the Azure Active Directory service in Microsoft Azure for accounts with only the Reader role [5]. The first one is set up horizontally, an external guest with an address matching a dynamic-group rule was automatically added to a group with virtual machine administrator user rights. In the second (vertical) path, an impact sub-score of 9.2 was lowered to 2.7 after scoping an Owner managed identity to specific resources on a virtual machine at subscription scope for a low-privilege user to retrieve storage keys.

On the other hand, a broader discussion of formal reasoning was presented on the productization of policy analysis and network reachability analysis to provide the customers with proof of their configuration rather than test results [6]. An interesting engineering decision was to run multiple SMT engines in parallel and take the first solution obtained – this is called a solver portfolio. As shown in Figure 2, out of 1 million unsatisfiable checks and 1 million satisfiable checks Z3 was the fastest solver in 965,092 and 959,543 checks, respectively, while CVC4 was the faster one in 34,908 and 39,932 checks, respectively.

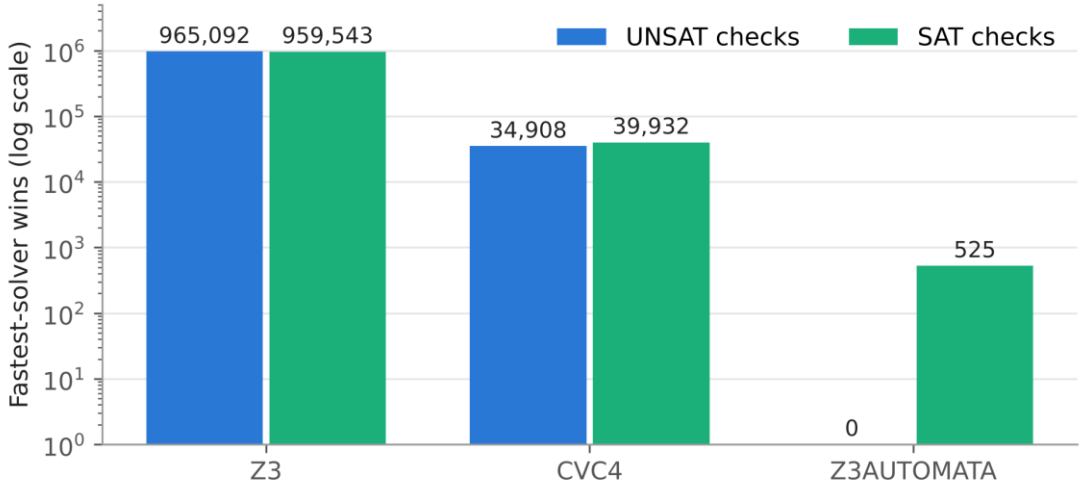


Figure 2. Solver Portfolio Win Counts Across One Million Unsatisfiable and One Million Satisfiable Policy Checks

2.2 *Identity Federation and Just-in-Time Elevation*

All dynamic decisions are enabled by the attributes that are supplied by identity federation. In a decentralized single sign-on (SSO) IAM scheme, multiple applications outsourced in the cloud and the authentication is federated between applications, while the authorization attributes are dynamically updated without re-registering users in the applications [7]. The significance in multi-account estate scenarios is that it will isolate the identity source of truth from multiple accounts relying on it and ensure that the removal of an attribute cascades across all accounts that rely on it. By separating, this will allow the ability to compare elevation requests against a set of attributes rather than against differing versions of those attributes.

JIT privilege management can also deny revoke privileges once they are established and only allow high privileges for a limited time, which can be activated either when authorized by a request, a ticket, or a contextual trigger, and automatically deactivate at the end of the time [8]. Examples of variants include creating an ephemeral account, temporary group membership, and the issuance of credentials that expire after a certain period of time. The security gain is expressed in terms of privilege hours as the number of privileges that are reduced. In the case of N entitlements held continuously for a period of T, the exposure can be calculated as given in Equation (3); in the case of JIT, the exposure can be calculated as the sum of the durations of all N sessions d, and the exposure reduction ratio ϱ can be calculated as shown in Equation (4).

$$E_{\text{std}} = N \cdot T,\ E_{\text{JIT}} = \sum_{i=1}^{n} d_i \quad (3)$$

$$\rho = 1 - \frac{E_{\text{JIT}}}{E_{\text{std}}} \quad (4)$$

2.3 Least-Privilege Role and Attribute Models

The problem of least-privilege in role-based access control (RBAC) has been formalized to find a set of roles that can be assigned to each user that gives exactly the permissions he needs, and not more [9] showed that it is an NP-hard problem in general, thus motivating heuristic solutions. The scale at which such heuristics have to operate is suggested by the benchmark datasets in Table 1 (from 35 users with 3046 permissions to 10021 users with 277 permissions). Densities that are less than 1% for the two datasets suggest that there is limited access to the datasets, and that they are likely to be accessed by manually curated roles that are especially susceptible to becoming over-privileged.

Table 1. Real-World Role-Mining Benchmark Datasets and Assignment Density

Dataset	Users	Permissions	User–permission assignments	Density (%)
healthcare	46	46	1,486	70.23
domino	79	231	730	4.00
emea	35	3,046	7,220	6.77
apj	2,044	1,164	6,841	0.29
firewall1	365	709	31,951	12.35
firewall2	325	590	36,428	19.00
americas_small	3,477	1,587	105,205	1.91
americas_large	3,485	10,127	185,294	0.53
customer	10,021	277	45,427	1.64

Source: Benchmark datasets widely used in heuristic role-mining evaluations, including those of Blundo et al.; density computed as $|UA| / (|U| \times |P|)$.

In a cloud-based context, the IAM survey classified the field into the following three categories: authentication, authorization and accountability mechanisms and identified the following three problems: federated identity, multi-tenancy, credential compromise and limited interoperability of standards [10]. In the case of multi-account enterprises, accountability is especially important as elevated decisions need to be traceable even after the temporary credential is lost. The audit records, then, serve two functions for the architecture: First, as a forensic trail to support governance processes; and second, as a source of training data that will be used to inform and inform more stringent policies at a later time.

Attribute-based access control (ABAC) is even more fine-grained than roles and relies on the attributes of the subject, object, action and environment

to make access decisions. Both positive and negative rules that allow or deny access were extracted by mining procedures, since exceptions are written as deny rules, it results in more compact, accurate rules than enumerating all possible exceptions that allow access [11]. Negative rules are very effective for JIT governance, for instance, if you don't want to ever allow an audit trail to be deleted from any session with elevated privileges, you only have to specify it once and everybody understands it.

It was shown that three classical models of access control (Mandatory, Discretionary and Role-based) are special cases of attribute evaluation in a unified ABAC model, which can be configured in the same attribute framework [12]. This outcome provides the theoretical footing for the development of policy in a progressive way. An organization can start off with role-equivalent attributes, and gradually add contextual attributes to them, such as ticket identifiers, risk of a request or the remaining session duration, without changing the enforcement layer or re-issuing identities previously federated between member accounts.

The question of user authorization to other clouds' resources comes up when using two or more clouds. The optimal rule mining under ABAC was formulated for such collaborating clouds, for a minimal rule set, the required cross-organization authorizations are reproduced and authorizations are changed dynamically [13]. For multi-account enterprises, the issue is the same within the enterprise, as each member account is a separate administrative domain that is semi-autonomous and has its own owners and resources. All the studied works have made conceptual contributions with respect to architectural functions, which are described in Table 2.

Table 2. Mapping of Reviewed Literature to Architectural Concepts

Architectural concept	Representative studies	Contribution to the architecture
Risk-adaptive decisions	[1], [14]	Context and history scoring; machine-learning detection of escalation
Formal policy verification	[2], [4], [6]	SMT encoding of policies; trust safety; solver portfolio
Misconfiguration threats	[5], [10]	Empirical escalation paths; catalogue of cloud IAM challenges
Federated identity and SSO	[7], [10]	Single attribute source with dynamic revocation
JIT elevation and zero trust	[8], [15], [16]	Removal of standing privilege; per-session, policy-driven access
Role mining and least privilege	[3], [9]	Constrained and least-privilege role derivation
ABAC models and mining	[11], [12], [14], [17], [18], [19]	Unified attribute model; positive, negative and log-driven rule mining
Cross-domain authorization	[13], [20], [21]	Collaborating-cloud rules; tenant trust; task-scoped cloud access
Least-privilege testing	[22]	Test-guided pruning of infrastructure-as-code permissions

Source: Compiled by the authors from the reviewed studies.

3. METHODS

A systematic review design was used. Coded architectural function, evaluation method, dataset and reported metrics for 23 peer reviewed studies and standards from 2010 to 2023 in the corpus. If policy-quality results were reported, they were normalised by normalising them with the policy-quality metric for the ABAC extraction method reported in [17] which is based on the combination of correctness and complexity

using a log-based method. The metric comes in Equation (5), where F is the F score of the mined policy on the access log and ΔWSC is the normalized complement of the weighted structural complexity (WSC) of the mined policy, defined in Equation (6).

$$Q_\pi = \frac{2 \cdot F \cdot \Delta WSC}{F + \Delta WSC} \quad (5)$$

$$\Delta WSC = \frac{WSC_{max} - WSC(\pi) + 1}{WSC_{max}} \quad (6)$$

The second coding axis was the difference between three types of combinations between attributes and roles: dynamic roles (attributes activate roles), attribute-centric designs (one attribute out of many is considered a role), and role-centric designs (attributes limit permissions of an assigned role) [23]. Studies were also categorized into five themes based on when they were published (Figure 3), with 11 of the 23 studies published after 2020. This was used to formally define the evolution cycle as the following equation (7), with unused permissions U removed and with the rules mined by M from the log window added to the system while the operator V only considers the candidates that pass the trust safety and regression tests.

$$P_{t+1} = \mathcal{V}\left((P_t \setminus U_t) \cup \mathcal{M}\left(L_{t-\Delta,t}\right)\right) \quad (7)$$

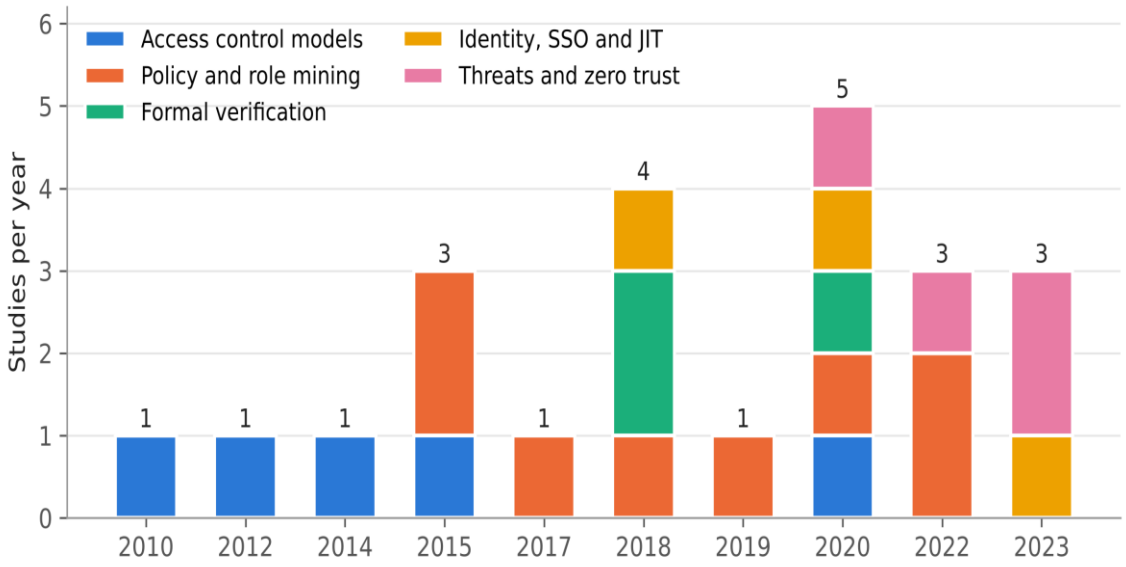


Figure 3. Temporal Distribution of the 23 Reviewed Studies by Research Theme

4. RESULTS AND DISCUSSION

4.1 *Reference Architecture for Policy-Evolving JIT Control*

The result is a layered reference architecture using machine-learning methods in detecting privilege escalation. The use of a supervised classification approach is an option for identifying escalation behavior of cloud activity, and for triggering mitigation, classifiers including random forest, adaptive boosting, extreme gradient boosting and light gradient boosting machines have been used [14]. These classifiers are used in the architecture also on the same audit stream as the policy mining which means that if an audit of a session is detected to be anomalous, the session will be cut short and the records will not be added to the window which means that the malicious activity can't be considered as normal.

The decision and enforcement of the security policy are linked to the zero trust architecture (ZTA) of the National Institute of Standards and Technology (NIST) [15] that comprises a policy engine and a policy administrator that form the point of decision and a policy enforcement point between each session. The seven tenets of ZTA dictate that access is given per session and the access

is decided by the dynamic policy. Figure 4 illustrates the flow of the components and table 3 provides a summary of the evidence for each component. Elevation requests contain the role(s) requested, the target account(s), the scope of the elevation, the duration of the elevation and the reason for the elevation and are approved after a successful Risk Scoring and formal verification.

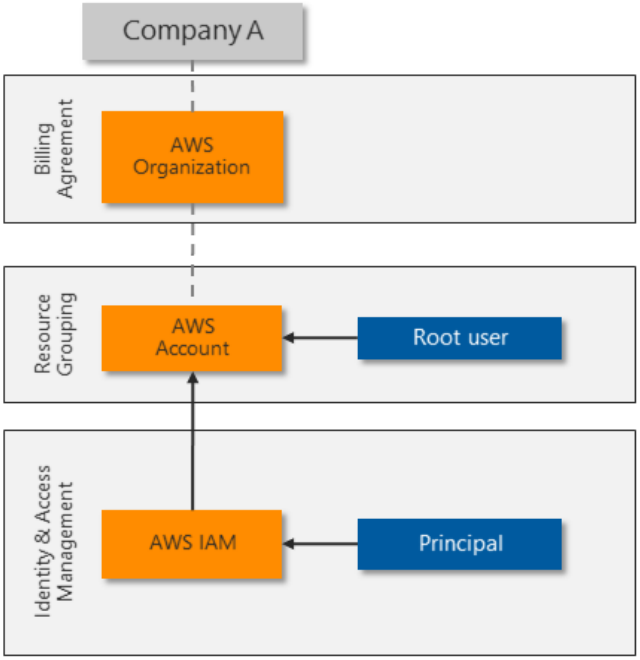


Figure 4. Policy-Evolving JIT IAM Reference Architecture for Multi-Account Cloud Enterprises (AWS)

Table 3. Architectural Layers, Enabling Techniques, and Supporting Quantitative Evidence

Layer	Function	Enabling technique	Reported or modeled evidence
Identity plane	Federate identities and attributes	Decentralized SSO-IAM	One attribute source for all consuming applications
Request and risk	Score each elevation request	Risk-based access control	Risk history used in 18 of 44 primary studies
Verification gate	Prove candidate grants safe	SMT reasoning, trust safety	99% of questions within 160 ms; millions of policies per day
Enforcement	Issue per-session credentials per account	ZTA enforcement point, JIT	Modeled exposure reduction of 95.3% (two-hour sessions)
Detection	Flag and terminate escalation	Supervised classifiers	Four ensemble and boosting families evaluated
Evolution	Mine least-privilege ABAC rules	Frequent itemsets, Cscore ranking	AUC of 0.9993 with a 30-day window
Validation	Confirm candidates before promotion	Test-based permission pruning	Permissions retained only if removal fails tests

Source: Synthesized from the reviewed studies; the exposure value is modeled with Equations (3) and (4).

4.2 *Log-Driven Policy Evolution*

The evolution layer is based on the study of 4.7 million CloudTrail audit events created by 38 users on 77 services during 16 months [18]. The true positive rate (TPR) was used to measure the under-privilege and the false positive rate (FPR) was used to measure over-

privilege (Equation 8), and candidate rules were ranked by the score in Equation (9) that trades coverage for over-privilege with weights ω. In Table 4, it is reported that mined ABAC policies achieved an area under the curve (AUC) of 0.9973 while RBAC achieved 0.9269. A 30-day window had the best AUC of 0.9993 with 7.2 minutes of mining, but longer windows increased cost, without increasing accuracy, as seen in Figure 5.

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP+FN}},\ \mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP+TN}} \quad (8)$$

$$C_{\mathrm{score}} = \mathrm{CR} + \omega(1 - \mathrm{OPR}) \quad (9)$$

Table 4. Least-Privilege ABAC Mining Results on 4.7 Million CloudTrail Events

Comparison	Setting	AUC	Mining time (min)
Access model	RBAC (3–120-day windows)	0.9269	–
Access model	Mined ABAC	0.9973	–
Rule-scoring metric	Cscore	0.9993	–
Rule-scoring metric	QrulImpl	0.9983	–
Rule-scoring metric	QrulFreqImpl	0.9978	–
Rule-scoring metric	λ-Distance	0.9972	–
Rule-scoring metric	QrulFreq / Qrul	0.9946 / 0.9922	–
Itemset threshold ε (ω = 1/10)	0.30	0.9640	1.2
Itemset threshold ε (ω = 1/10)	0.20	0.9940	2.8
Itemset threshold ε (ω = 1/10)	0.10	0.9993	15.3
Itemset threshold ε (ω = 1/10)	0.05	0.9996	29.8

Source: Adapted from the evaluation of Sanders and Yue (38 users, 77 services, 16 months of audit data).

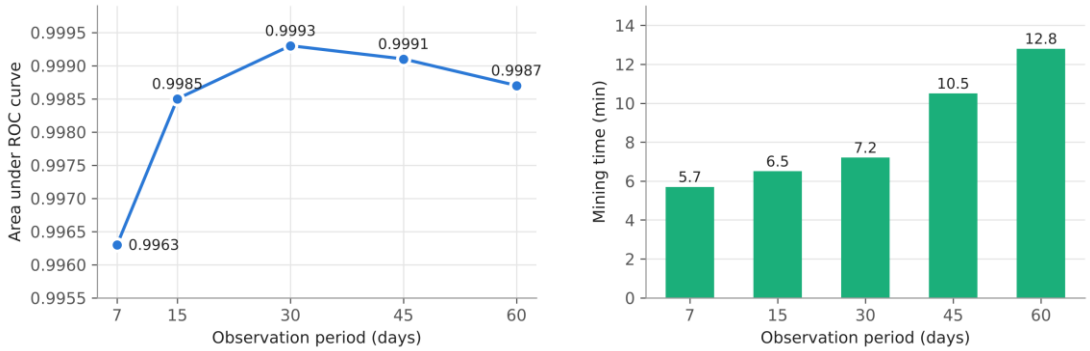


Figure 5. Effect of Observation-Period Length on Mined-Policy AUC and Mining Time

No direct promotions of mined candidates. A test-based discovery method for infrastructure as code (IaC) was presented that is able to detect least-privilege permission sets by deploying code and then gradually revoking a permission set until the deployment fails, which leaves only the revoked permission sets that caused the deployment to fail [22]. The technique enables to put the threshold trade-off in practice as a 'safeguard' in a practical sense: the lower threshold (ε = 0.1) means only a small loss of AUC (0.0003) and almost doubles the mining time (from 15.3 to 29.8 minutes), so tests for under-privilege of the lower threshold can be performed before it is deployed to the live network.

4.3 *Comparative Assessment of Access Control Paradigms*

The results of a comprehensive ZTA survey found that identity is the new security perimeter and that the continuous authentication, micro-segmentation and least-privilege authorization technologies must be integrated, not simply paired together as products [16]. These principles are compared to the reviewed paradigms in the following table, Table 5. Four 2-hour elevations per week reduces the annual exposure per entitlement from 8,760 to 416 privilege-hours as modeled in Figure 6, using Equation (4), and to 416 if four 4-hour elevations per week are used.

Table 5. Comparative Assessment of Access Control Paradigms for Multi-Account Enterprises

Paradigm	Decision basis	Temporal scope	Cross-account support	Formal verification	Adaptivity
RBAC	Assigned roles	Standing	Role duplication per account	Not inherent	Manual re-engineering
ABAC	Subject, object, environment attributes	Standing or conditional	Attribute federation	Possible via SMT	Rule mining
Risk-based	Estimated risk vs. threshold	Per request	Context-dependent	Rarely addressed	Risk feedback
Multi-tenant authorization	Declared tenant trust	Standing	Native trust relations	Not inherent	Trust updates
Zero trust	Dynamic policy per session	Per session	Identity-centric	Not mandated	Continuous evaluation
Policy-evolving JIT	Risk, attributes, verified policy	Bounded session	Trust relations plus ephemeral grants	Gate on every change	Log-driven mining with tests

Source: Synthesized from the reviewed studies.

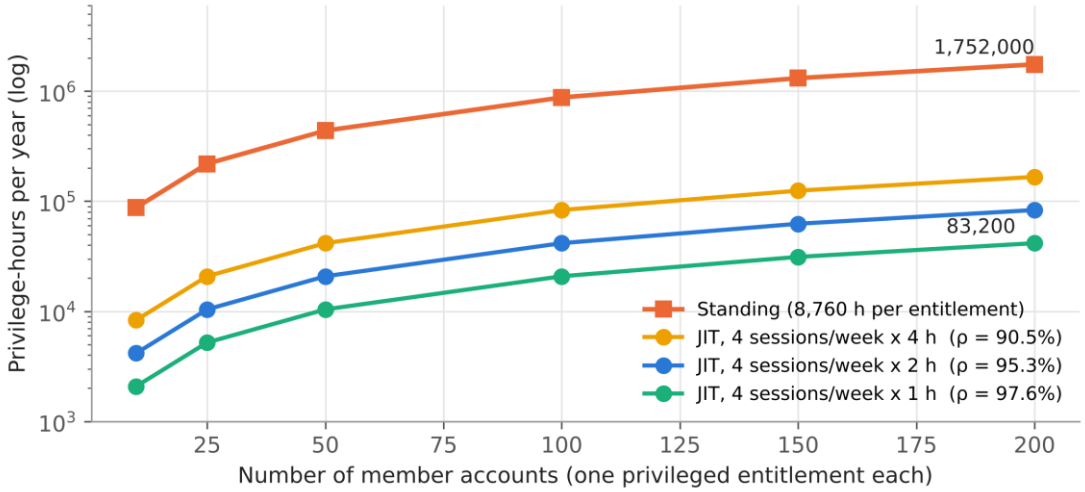


Figure 6. Modeled Annual Privilege Exposure Under Standing and JIT Access as Account Count Grows

Multi-tenancy authorization models were born to describe trust relationships between tenants, and the trustee tenant can only place users from a trustor tenant on roles of a trustor tenant if it is stated that it trusts a trustor tenant. There are two lessons passed on: the multi-account enterprise is one such lesson. Cross account role assumption via declared trust, not duplicate users and accounts should not create trust relations; these should be established by

organizational units. There are 200 accounts with 1,752,000 standing privilege/hours/year (1/200); versus 83,200 privilege-hours per estate (2-hour JIT profile).

4.4 Scalability of Policy Mining

The initial ABAC policy mining algorithm set WSC as the benchmark measure of the complexity of a policy and was tested on a set of sample university policies, healthcare policies and project management policies, each having WSC values of 33, 33 and 44, respectively [19]. A subsequent comparison (Table 6, Figure 7) showed that it was incapable of efficiently handling large logs: 32,645 s was used to process one log in the healthcare domain, and the unsupervised clustering approach took from 7.3 to 22.7 s to process all six synthetic logs with policy quality ranging from (0.66 to 0.89)).It was not efficient for big logs: compared to the unsupervised clustering approach, which used 7.3 s to 22.7 s (with a quality of policy ranging from 0.66 to 0.89) for the six synthetic logs, the QAUTO took 32,645 s for a single healthcare log. The differences in run time of a policy determine the frequency of policy refresh in continuous evolution.

Table 6. Policy-Mining Performance on Partial (0.1%) Access Logs

Dataset	Method	Time (s)	ACC (%)	F-score (%)	Rules	WSC	Q
University P	Xu–Stoller miner	227	94.74	65.87	10	34	0.79
University P	Cotrini et al.	126	80.74	45.30	132	508	0.58
University P	Unsupervised clustering	7.3	96.00	74.20	7	29	0.85
Healthcare P	Xu–Stoller miner	32,645	64.43	63.61	3	6	0.78
Healthcare P	Cotrini et al.	529	72.72	64.00	65	272	0.75
Healthcare P	Unsupervised clustering	7.9	79.78	68.23	13	49	0.81
University PN	Xu–Stoller miner	4,230	73.37	16.10	10	34	0.28
University PN	Cotrini et al.	204	93.55	88.50	385	1,389	0.86
University PN	Unsupervised clustering	15.0	89.30	80.00	10	40	0.89
Amazon UCI	Cotrini et al.	1,345	70.93	75.64	466	1,247	0.85
Amazon UCI	Unsupervised clustering	1,010.4	98.49	99.00	24	92	0.99

Source: Adapted from the comparative evaluation of Karimi et al.; the Xu–Stoller miner produced no output within 24 hours on the Amazon datasets.

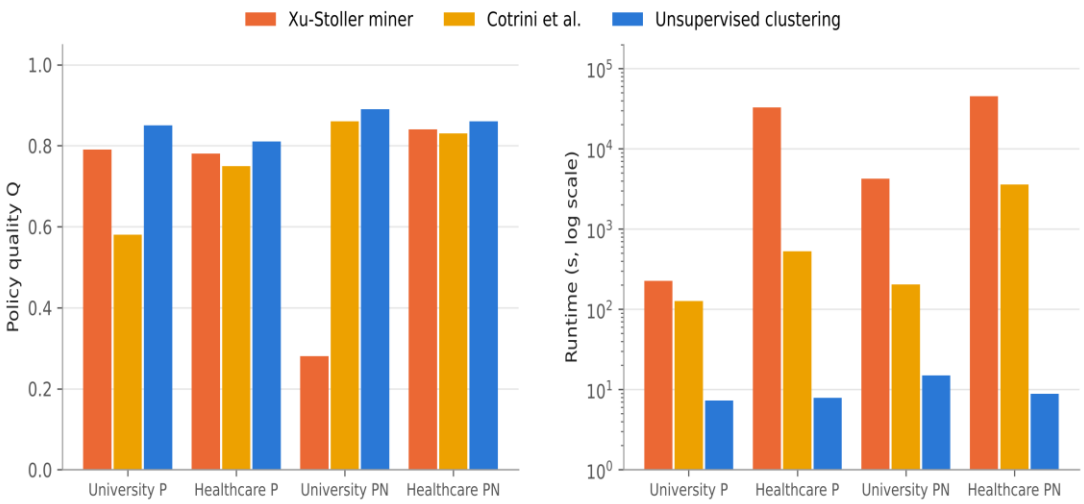


Figure 7. Policy Quality and Runtime of Three ABAC Mining Approaches on Partial Logs

4.5 *Challenges, Ethical Considerations, and Regulatory Impacts*

To accommodate for access control in cloud computing, roles, tasks and data security classifications were combined, and permissions would be granted only for access as needed at the time of access, and only for the data that has an appropriate sensitivity level [21]. The task orientation will then follow the JIT control, and common issues that occur in this adoption are listed in Table 7. Approval fatigue may result in JIT becoming a standing-access exercise, sessions must be monitored at all times, thus increasing the proportionality and privacy concerns for employees, and the policy requires least privilege and auditability, where all grants, verification-proofs and changes to policy must be recorded. When it is not possible to use the decision plane, break-glass procedures must be used.

Table 7. Challenges, Observed Mechanisms, and Architectural Mitigations

Challenge	Observed mechanism	Indicator	Mitigation in the architecture
Dynamic-group abuse	Guest invitation matches a group rule granting VM administrator login	Exploitability 6.9; base 4.9	Rules on non-user-controllable attributes; expiring JIT membership
Over-scoped managed identity	VM identity holds Owner at subscription scope	Impact 9.2, reducible to 2.7	Resource-scoped ephemeral credentials; trust-safety gate
Public exposure	Policy admits an untrusted principal	Millions of policies verified per day	Mandatory verification before any grant or change
Privilege drift	Unused permissions accumulate in roles	AUC 0.9269 (RBAC) vs. 0.9973 (ABAC)	Periodic mining and pruning
Mining scalability	Exhaustive miners on large logs	> 24 h without output vs. 7.3 s	Clustering miners on windowed logs
Approval fatigue and privacy	Routine approvals; intensive session monitoring	Qualitative	Risk-tiered auto-approval; data minimization
Decision-plane outage	Broker or verifier unavailable	Qualitative	Break-glass accounts with post-hoc review

Source: Synthesized from the reviewed studies, including the Azure Active Directory experiments of Bu Haimed et al.

5. CONCLUSION

A policy-evolving IAM architecture was synthesized from 23 studies on a variety of areas of IAM, including risk-based access control, formal verification, federated identity, policy mining, multi-tenancy authorization and zero trust, to serve as a solution for privilege control on the fly (JIT) in multi-account cloud enterprises. The synthesis suggests that the constituent techniques are sufficiently developed to be combined within a closed loop where an elevation request is scored as it is made, privileges are granted for bounded sessions only, and audit evidence is mined to tighter attribute-based policies which are admitted only after formal verification and regression testing. The quantitative results have validated this, with 99% of the questions being answered by inline SMT checks within 160ms, and the mined attribute-based policies clearly outperforming the role-based baseline, and bounded sessions eliminating more than 90% of the modeled privilege exposure with realistic usage profiles.

Here are some study tips that are suggested for the following study. The collection of benchmarks is important because most mining evaluations continue to be synthetic or based on single organization data, which should also comprise elevation requests and audit logs as well as ground-truth entitlements on a selection of accounts. All risk scoring should be rethought and used in conjunction with approval processes to ensure that JIT is not routinely approval and the proportionality of session monitoring should be considered against relevant privacy requirements. Finally, operators for reasoning about trust relationships at the entire account hierarchy level, instead of as individual policies, should be created to enable the ability to prove the safety of policy evolution at the organizational level.

REFERENCES

[1] H. F. Atlam, M. A. Azad, M. O. Alassafi, A. A. Alshdadi, and A. Alenezi, "Risk-based access control model: A systematic literature review," *Futur. Internet*, vol. 12, no. 6, p. 103, 2020, doi: https://doi.org/10.3390/fi12060103.

[2] J. Backes *et al.*, "Semantic-based Automated Reasoning for AWS Access Policies using SMT," in *2018 Formal Methods in Computer Aided Design (FMCAD)*, 2018, pp. 1–9. doi: 10.23919/FMCAD.2018.8602994.

[3] C. Blundo, S. Cimato, and L. Siniscalchi, "Role Mining Heuristics for Permission-Role-Usage Cardinality Constraints," *Comput. J.*, vol. 65, no. 6, pp. 1386–1411, Jun. 2022, doi: 10.1093/comjnl/bxaa186.

[4] M. Bouchet *et al.*, "Block public access: trust safety verification of access control policies," in *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, in ESEC/FSE 2020. New York, NY, USA: Association for Computing Machinery, 2020, pp. 281–291. doi: 10.1145/3368089.3409728.

[5] I. B. Haimed, M. Albahar, and A. Alzubaidi, "Exploiting misconfiguration vulnerabilities in microsoft's azure active directory for privilege escalation attacks," *Futur. Internet*, vol. 15, no. 7, p. 226, 2023, doi: https://doi.org/10.3390/fi15070226.

[6] B. Cook, "Formal Reasoning About the Security of Amazon Web Services," in *Computer Aided Verification*, H. Chockler and G. Weissenbacher, Eds., Cham: Springer International Publishing, 2018, pp. 38–47. doi: https://doi.org/10.1007/978-3-319-96145-3_3.

[7] S. Fugkeaw, "Achieving decentralized and dynamic sso-identity access management system for multi-application outsourced in cloud," *IEEE Access*, vol. 11, pp. 25480–25491, 2023, doi: https://doi.org/10.1109/ACCESS.2023.3255885.

[8] M. J. Haber, "Just in Time," in *Privileged Attack Vectors*, M. J. Haber, Ed., Berkeley, CA: Apress, 2020, pp. 285–294. doi: 10.1007/978-1-4842-5914-6_21.

[9] H. Huang, F. Shang, J. Liu, and H. Du, "Handling least privilege problem and role mining in RBAC," *J. Comb. Optim.*, vol. 30, no. 1, pp. 63–86, 2015, doi: 10.1007/s10878-013-9633-9.

[10] I. Indu, P. M. R. Anand, and V. Bhaskar, "Identity and access management in cloud environment: Mechanisms and challenges," *Eng. Sci. Technol. an Int. J.*, vol. 21, no. 4, pp. 574–588, 2018, doi: https://doi.org/10.1016/j.jestch.2018.05.010.

[11] P. Iyer and A. Masoumzadeh, "Mining Positive and Negative Attribute-Based Access Control Policy Rules," in *Proceedings of the 23nd ACM on Symposium on Access Control Models and Technologies*, in SACMAT '18. New York, NY, USA: Association for Computing Machinery, 2018, pp. 161–172. doi: 10.1145/3205977.3205988.

[12] X. Jin, R. Krishnan, and R. Sandhu, "A Unified Attribute-Based Access Control Model Covering DAC, MAC and RBAC," in *Data and Applications Security and Privacy XXVI*, N. Cuppens-Boulahia, F. Cuppens, and J. Garcia-Alfaro, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 41–55. doi: https://doi.org/10.1007/978-3-642-31540-4_4.

[13] J. C. John, S. Sural, and A. Gupta, "Optimal Rule Mining for Dynamic Authorization Management in Collaborating Clouds Using Attribute-Based Access Control," in *2017 IEEE 10th International Conference on Cloud Computing (CLOUD)*, 2017, pp. 739–742. doi: 10.1109/CLOUD.2017.104.

[14] M. Mehmood, R. Amin, M. M. A. Muslam, J. Xie, and H. Aldabbas, "Privilege Escalation Attack Detection and Mitigation in Cloud Using Machine Learning," *IEEE Access*, vol. 11, pp. 46561–46576, 2023, doi: 10.1109/ACCESS.2023.3273895.

[15] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero trust architecture," *Natl. Inst. Stand. Technol. Spec. Publ. 800-207*, 2020, doi: https://doi.org/10.6028/NIST.SP.800-207.

[16] N. F. Syed, S. W. Shah, A. Shaghaghi, A. Anwar, Z. Baig, and R. Doss, "Zero Trust Architecture (ZTA): A Comprehensive Survey," *IEEE Access*, vol. 10, pp. 57143–57179, 2022, doi: 10.1109/ACCESS.2022.3174679.

[17] L. Karimi, M. Aldairi, J. Joshi, and M. Abdelhakim, "An Automatic Attribute-Based Access Control Policy Extraction From Access Logs," *IEEE Trans. Dependable Secur. Comput.*, vol. 19, no. 4, pp. 2304–2317, 2022, doi: 10.1109/TDSC.2021.3054331.

[18] M. W. Sanders and C. Yue, "Mining least privilege attribute based access control policies," in *Proceedings of the 35th Annual Computer Security Applications Conference*, in ACSAC '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 404–416. doi: 10.1145/3359789.3359805.

[19] Z. Xu and S. D. Stoller, "Mining Attribute-Based Access Control Policies," *IEEE Trans. Dependable Secur. Comput.*, vol. 12, no. 5, pp. 533–545, 2015, doi: 10.1109/TDSC.2014.2369048.

[20] B. Tang, R. Sandhu, and Q. Li, "Multi-tenancy authorization models for collaborative cloud services," *Concurr. Comput. Pract. Exp.*, vol. 27, no. 11, pp. 2851–2868, 2015, doi: https://doi.org/10.1002/cpe.3446.

[21] Y. A. Younis, K. Kifayat, and M. Merabti, "An access control model for cloud computing," *J. Inf. Secur. Appl.*, vol. 19, no. 1, pp. 45–60, 2014, doi: https://doi.org/10.1016/j.jisa.2014.04.003.

[22] R. Shimizu and H. Kanuka, "Test-Based Least Privilege Discovery on Cloud Infrastructure as Code," in *2020 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, 2020, pp. 1–8. doi: 10.1109/CloudCom49646.2020.00007.

[23] D. R. Kuhn, E. J. Coyne, and T. R. Weil, "Adding Attributes to Role-Based Access Control," *Computer (Long. Beach. Calif).*, vol. 43, no. 6, pp. 79–81, 2010, doi: 10.1109/MC.2010.155.