**Sciences du Nord Nature Science and Technology**
**Vol. 2, No. 02, December 2025: pp. 195-207**
**E-ISSN: 3046-6504**
**P-ISSN: 3047-4922**
**DOI: https://doi.org/10.71238/snnst.v1i02.248**

# Risk-Adaptive Zero-Trust Authorization for Dynamic Workloads Across Hybrid and Multi-Cloud Infrastructure

**Shubh Prabhat** [1]
[1] Solution Architect, GlobalLogic Inc., Richmond Hill, Ontario, Canada
Coresponding Author: sprabhat.research@gmail.com

**ARTICLE HISTORY**
**Received Dec, 2025** **Revised Dec, 2025** **Accepted Dec, 2025**

**ABSTRACT**
Traditional applications are short-lived, which run on containers across on-premises and multiple public cloud clusters. This movement cannot follow fixed rules and static positions of the network. Stolen workload credential could remain usable long and a breached container could access numerous of the neighbors without anyone noticing it. The paper introduces a dynamic zero trust authorization system of risk-adaptive workloads in the hybrid cloud and multi-cloud environment. The framework assigns marks on each request based on the four signals which include identity confidence, node posture, behavior anomaly and exposure context. It ranks the score on resource urgency, and time since a verification. A three-way decision is then used to permit or request step up verification or deny it and revoke the credential. Decision thresholds adjust to new traffic in such a way that the false denials during normal surges are not increased. We have scaled the framework on a testbed that comprised of one on-premises cluster and two public cloud clusters and have scaled 1,840 workload identities and 48.2 million authorization requests across 21 days. In five scenarios, there were 300 injected attacks that were in the test period. The framework scored an F1 score of 0.91 compared to 0.80 zero trust with fixed thresholds and 0.60 fixed role-based control. The mean time to revoke access dropped to 9.4 seconds and average blast radius dropped to 12 workloads. Staying low were the false denials of 0.8 percent, which remained low throughout the test period. The maximum delay added was 4.7 milliseconds per request. The outcomes indicate that the adaptive risk scoring is more effective in improving the containment with a small delay cost.

***Keywords:*** *Access Control; Dynamic Workloads; Microservices; Multi Cloud Security; Risk Adaptive Authorization; Zero Trust*

## INTRODUCTION

Enterprise applications now execute a large number of small services which start, stop and relocate all day long. This style was implemented by one of the large European online retailers to enable over 500 live deployments per week [1]. One team which migrated a monolith to microservices commented that the transformation facilitated continuous delivery, and short release cycles [2]. Micro services are tiny components and teams either construct and operate separately and each component communicates with others via the internet [3]. Each of the routes between units requires an authorization decision.

These applications are now located in various locations. Cloud computing provides access to shared resources on the order and also transfers control of the infrastructure to the owner of the data [4]. Cloud security lists the vulnerabilities and threats per service model, and the countermeasures to apply to each service model [5]. Hybrid and multi cloud configurations bring

other issues. Each of the providers possesses a system of identities, a policy language and a log. What is a rule in one location may not be in a different one.

Part of this issue is addressed by zero-trust. An overview of zero-trust architecture states that the main concept is not implicit trust, and all requests are verified based on existing evidence [6]. The majority of deployments continue to use fixed rules to perform the check. A strictly followed rule would take an identical request by a healthy workload at noon and that of a drifting workload at midnight differently when the workload roles are similar. With change of risk, there should be change in the response to a request.

There are three objectives in this paper. The first is to develop request level risk score that cuts across all the environments. The second is to transform the score into the allow, step up and deny decisions that behave according to the workload. The third will be to test the quality of detecting, containment speed and the added delay on a testbed that spans one on-premises cluster and two public clouds.

## LITERATURE REVIEW

### *2.1 Access Control Models*

Initial definitions of access control used included a set of rules that establish who can access what object. It divided discretionary, mandatory and role-based forms [7]. Role based control assigns roles to users and roles to permissions, and it became the popular model in enterprises [8]. Roles are permanent and thus more suitable in individuals compared to temporary boxes. Attribute based control determines based on the subject attributes, the object attributes and environment attributes [9]. It is possible to use context in this model although the rules that read the context are handwritten, and hardly change.

### *2.2 Risk Based Access Control*

The beginning of risk-based control is based on another concept. The ruling will be made depending on the extent of damage that the access may bring about. An experiment involved a fuzzy multi-level security, quantified risk, and restricted the overall risk that a user was allowed to take [10]. Another subsequent research introduced risk estimates to role-based control such that the use of roles is conditioned to an estimated risk [11]. In the case of clouds, risk is computed during run time of every access request [12]. Another study used attribute-based control model with event calculus and weighted risk indices [13]. The existence of a distributed access control cloud architecture involving multiple providers demonstrates cloud decisions need to cross provider boundaries [14].

### *2.3 Zero Trust and Learning*

Continuous checking is added to zero trust work. An overview of automation and orchestration in zero trust reports machine learning classifiers of security events of about 95 percent accuracy in the studies it discusses [15]. An arrangement of industrial Internet of Things indicates that the concept of a zero trust also extends beyond the data centres [16]. Policy control is also involved with learning. A single study employed offline deep learning which included deep reinforcement learning to modify role-based policy in smart grid applications [17].

Table 1 contrasts six methods and indicates the difference. None of them pool request level risk, work life, resource criticality and threshold flexibility in a single decision across a number of clouds. This gap is filled in this paper.

**Table 1. Six access control strategies evaluated based on decision basis, primary strength and gap by each to dynamic multi cloud workloads.left**

| Approach | Decision basis | Main strength | Gap in dynamic loads |
|---|---|---|---|
| Role based control | Static role membership | Simple to audit | Roles are not followed by short lived workloads |
| Attribute based control | Attributes typed up manually on attributes | Uses context | Rules are handwritten and do not move |
| Fuzzy multi-level security | Risk limits and risk quantities | Graded decisions | Constructed user-friendly and fixed labels |
| Role based controls that are risk conscious | Role as used with risk estimates | Contributes risk to an established model | Still tied to roles |
| Dynamic cloud risk models | Request run time risk | Fits cloud access | Single provider and fixed weights |
| Policy control approach to learning | Learned policy updates | Adapts over time | Offline/training and limited domain |

## FRAMEWORK

The framework is depicted in figure 1. It consists of a circuit of four layers with one feedback. The identity, posture and traffic evidence on the environments are collected in the collection layer. That evidence is converted to a risk value by the scoring layer. The decision layer provides a comparison with the thresholds. The outcome is enforced by the enforcement layer with the help of sidecar proxies beside every workload. Each deny result is reviewed by an operator and the decision is emitted back to the scoring layer in the form of labels.

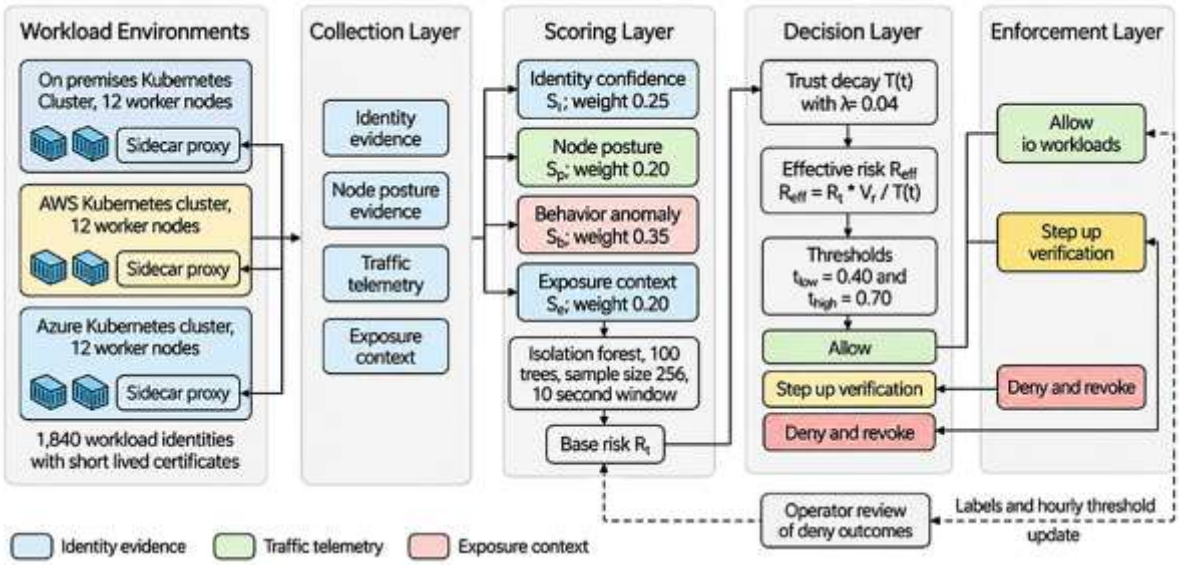



**Figure 1. Framework architecture indicating the telemetry collection, risk scoring, decision engine and sidecar enforcement and feedback path in three environments.**

This design is based on how the containers behave. Containers are lightweight and they start quickly and this is the reason why orchestrators make and take them throughout the day [18]. They all share the host kernel, exposing a weak isolation that exposes compromised container to its neighbors [19]. Microservice systems were also associated with numerous dependencies and numerous owners, and this contributes to the fact that it is difficult to have fixed rules that are easy to maintain [20]. The framework thus connects all the decisions to an interim identity and new evidence.

### *3.1 Risk Signals*

Workloads are allocated a temporary certificate every time they are started. There are four sets of evidence read by the collection layer and Table 2 provides them with weights. Base risk at time t can be referred to as Equation (1).

**Table 2. Symbol, collected evidence, value range and weight on the request risk score of four risk signals.**

| Signal | Symbol | Evidence collected | Range | Weight |
|---|---|---|---|---|
| Identity confidence | $S_i$ | Attestation result, certificate age, token scope | 0 to 1 | 0.25 |
| Node posture | $S_d$ | Patch level, image signature, runtime alerts | 0 to 1 | 0.20 |
| Behavior anomaly | $S_b$ | Calling rate, destinations, data, time | 0 to 1 | 0.35 |
| Exposure context | $S_e$ | Internet reach, cross cloud path, network zone. | 0 to 1 | 0.20 |
| Total | | | | 1.00 |

$$R_t = w_i S_i + w_d S_d + w_b S_b + w_e S_e, \quad where\ w_i + w_d + w_b + w_e = 1 \#(1)$$

In Equation (1) each S value is a risk value 0 to 1 each w value the weight in Table 2. Behavior has the greatest weight since it responds initially in case a workload gets compromised. Experts were used to determine the weights and they were verified during tuning.

## RISK MODEL AND DECISION LOGIC

Figure 2 depicts routes taken by a request within the engine. The request is passed on to the nearest decision engine by the sidecar. The engine will retrieve the attributes and calculate the score, scale it and compare with two thresholds. The product is reappeared in a few milliseconds in the sidecar.

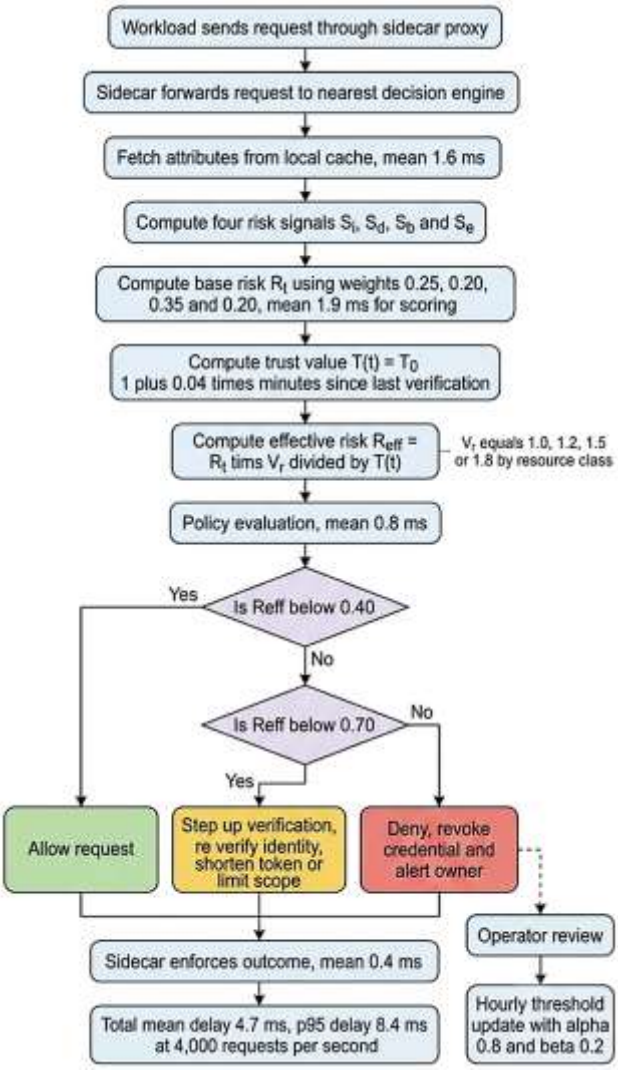



**Figure 2. Score based authorization decision based on comparison of the score against a threshold to an allow, step up or deny outcome.**

### *4.1 Behavior Anomaly Score*

An isolation forest is used as a source of the behavior signal. This technique separates abnormal points that have random splits and unusual points require fewer splits than ordinary ones [21]. Twelve traffic features help the model in learning per workload. The score of the anomaly is provided in equation (2).

$$a(x) = \frac{1}{2^{\frac{E[h(x)]}{c(n)}}} \#(2)$$

In Equation (2), h(x) is the path length of request x in a tree, E[h(x)] is the average path lengths of all the trees and c(n) is the average path length of a sample of n points level. When the average path is small, then a score of around 1 is the result. The sample size was 256 and we used 100 trees. Each 10-second window is recalculated and is computed again until the actual score is found which is based on the concept of continuous authentication whereby the system keeps verifying the actor since he/she has already logged in [22].

### *4.2 Trust Decay and Effective Risk*

Reliability to an authenticated identity ought to reduce with the lapse of time. This shrinkage is modelled by equation (3).

$$T(t) = \frac{T_0}{1 + \lambda \Delta t} \#(3)$$

In this case, $T_0 = 1$ at the point of verification, $\Delta t$ = the minutes since the last verification and $\lambda = 0.04$. The workload that is not verified after 25 minutes maintains the value of the trust at 0.50. Equation (4) ratios an equation of the base risk, resource criticality and the trust value.

$$R_{eff} = \frac{R_t \cdot V_r}{T(t)} \#(4)$$

For the resource class, $V_r$ is a criticality factor. Table 3 contains four classes, and the factors of each of them, as well as post-outcome actions.

**Table 3. Four resource classes in the form of criticality factor, share of requests and activity performed in the step up and deny bands.**

| Resource class | Vr | Share of requests (%) | Step up action | Deny action |
|---|---|---|---|---|
| Public telemetry | 1.0 | 41.2 | Re-verify certificate | Block call and log |
| Internal service data | 1.2 | 33.5 | Re-verify and further abbreviate token | Block call and revoke token |
| Customer records | 1.5 | 17.6 | Re-verify and put scope into read only | Block, revoke and alert |
| Secrets and keys | 1.8 | 7.7 | Re-verify and re-sign workload | Block, revoke and quarantine workload |

### *4.3 Decision Rule and Threshold Adaptation*

The equation (5) transforms the effective risk to either of three. The first thresholds are 0.40 in case of $\tau_1$ and 0.70 in case of $\tau_2$.

$$Allow\ if\ R_{eff} < \tau_1. \quad Step\ up\ if\ \tau_1 \leq R_{eff} < \tau_2. \quad Deny\ and\ revoke\ if\ R_{eff} \geq \tau_2. \#(5)$$

Traffic thresholds do not work when there is a normal change in the traffic. Call rates increase with a release or seasonal peak and bring about false denials. The values of the thresholds are updated every hour (Equation (6)).

$$\tau_{new} = \alpha \tau_{old} + \beta \tau_{obs}, \quad where\ \alpha + \beta = 1 \#(6)$$

In Equation (6), $\tau_{obs}$ represents the threshold level that would permit the false denial percentage rate to stand at 1 percent in the previous hour, according to operator judgements. We set $\alpha$ to 0.8 and $\beta$ to 0.2. The weight gradually used over new evidence prevents one noisy hour from taking the thresholds too far.

## EXPERIMENTAL SETUP

We created three clusters of Kubernetes testbed. One of them was on-premises, the other was on Amazon Web Services and the other was on Microsoft Azure. The entire configuration is provided in table 4. Cloud native applications are recommended to model-controlled load and fixed resource sizes, and the load generator adhered to that recommendation [23]. Workload mixes and the fault style were based on published microservice benchmark of 41 services with 22 fault

reproductions cases [24]. We added it with batch jobs, and style function tasks. Attributes were stored locally in their respective caches by the decision engines and a separate instance of a decision engine was present in each environment such that there was no request that crossed a cloud boundary to get a decision.

**Table 4. Experimental controls including cluster sizes, number of workloads, request sizes, number of attack episodes, baselines and metrics in every run.**

| Item | Setting |
|---|---|
| Worker nodes | 12 on premises, 12 on AWS, 12 on Azure, 36 in total |
| Workload identities | 1,840 in total |
| Workload types | Long running services = 240, Batch jobs = 410, Style of functions (Tasks) = 1,190. |
| Median workload lifetime | 14 minutes |
| Observation period | 21 days (14 days training and 7 days test) |
| Authorization requests | 48.2 million altogether, 16.1 million during test period |
| Injected attacks | There were 300 episodes in the test period, 60 of each in five scenarios |
| Baselines | Role-based control, attribute-based control (static), fixed thresholds zero-trust |
| Metrics | Precision, recall, F1, false denial rate, time to revoke, blast radius, delay |
| Repetitions | Three repeat executions using random seeds |

Figure 3 displays the number of requests per day. This was an average of 2.3 million requests per day. Day 18 had 3.9 million requests as we replayed a release wave. This day will inform whether the thresholds can deal with a normal surging.

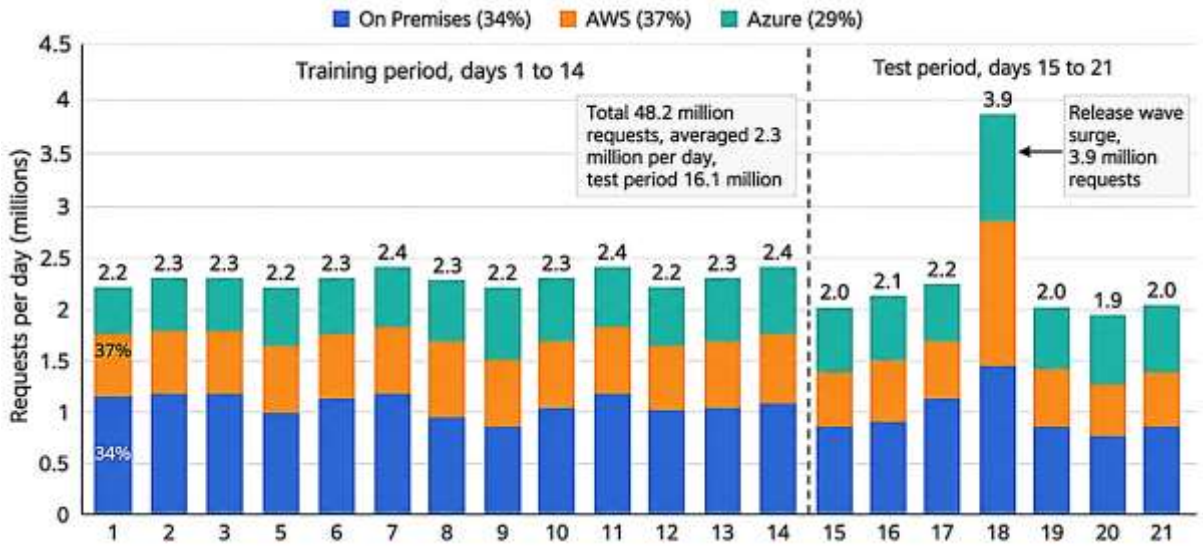



**Figure 3. The three environments have a workload peak on day 18 whereas the volume of the authorization requests each day is of millions of requests over 21 days.**

The five attack scenarios were stolen token reuse, lateral movement out of a compromised pod, privilege escalation by the use of a broad role, bulk data pull across clouds and misuse post policy drift. Each episode had its start time recorded in the injection tool, therefore, the correctness of all the labels is accurate. We endeavour to quantify delay using Equation (7).

$$L_{total} = L_{attr} + L_{score} + L_{eval} + L_{enf} \#(7)$$

Attributes fetch, risk scoring, policy evaluation and enforcement time are the four terms. The three baselines shared enforcement path resulting in their difference in delay depending on decision logic.

## RESULTS

The F1 score according to equation (8) is used to compare methods. P denotes precision and R denotes recall and they were calculated on an episode wise basis.

$$F1 = \frac{2 \cdot P \cdot R}{P + R} \#(8)$$

The 16.1 million test requests were reported as the main results as shown in Table 5. The framework had F1 score of 0.91. It is 0.11 over zero trust with fixed limit, 0.21 over static attribute-based control and 0.31 over static role-based control. A paired experiment through the 300 episodes had a result of p-value less than 0.01 against the most robust baseline. Security teams are most interested in the fact that the recall gain differs value based on the number of episodes missed since the attacker is left with live access.

**Table 5. False denials, detection quality, revocation time, p95 delay and blast radius on 16.1 million test requests.**

| Method | Precision | Recall | F1 | False denial (%) | Mean time to revoke (s) | Blast radius (workloads) | p95 delay (ms) |
|---|---|---|---|---|---|---|---|
| Static role-based control | 0.71 | 0.52 | 0.60 | 0.9 | 1,860 | 164 | 2.1 |
| Static attribute-based control | 0.78 | 0.63 | 0.70 | 1.4 | 540 | 97 | 4.8 |
| Zero trust with fixed thresholds | 0.84 | 0.76 | 0.80 | 2.6 | 74 | 41 | 6.3 |
| Proposed framework | 0.93 | 0.89 | 0.91 | 0.8 | 9.4 | 12 | 8.4 |

The framework reduced the revocation time average from 74 seconds to 9.4 seconds or 87 percent shorter. It reduced the blast radius by 71 percent over the same baseline and by 93 percent over the same baseline as compared with the role-based control that was fixed. False rejections decreased from 2.6 percent to 0.8 percent. Figure 4 presents the F1 score of each scenario of attack.

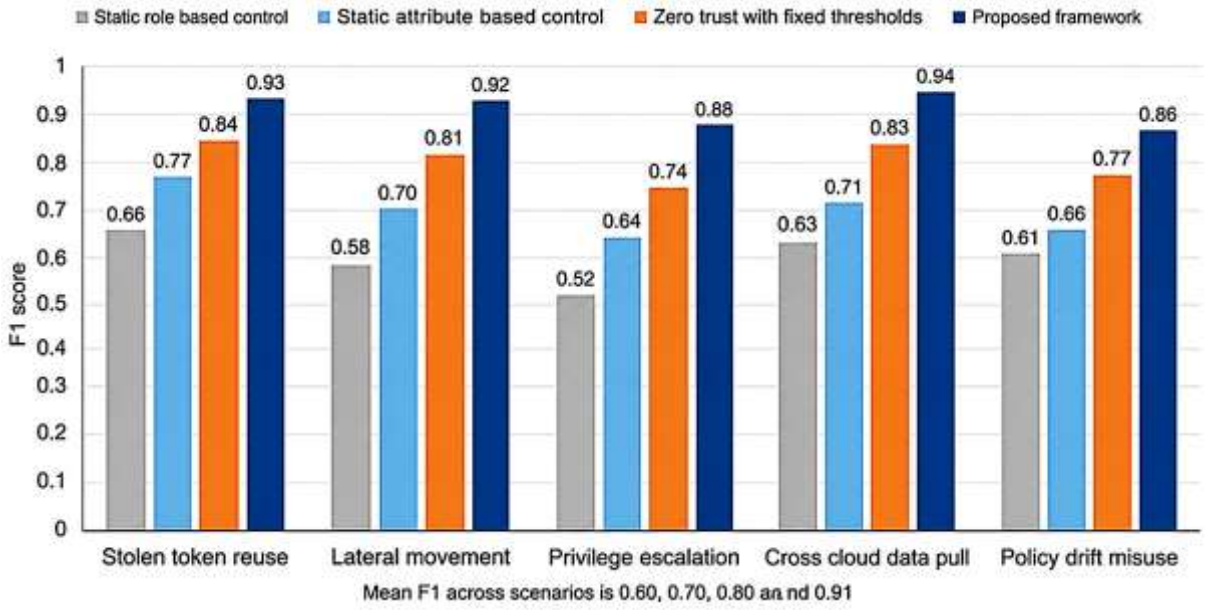



**Figure 4. The F1 scores of four algorithms in five scenario attacks, with the proposed framework scoring the highest in all the scenarios.**

The model had a score of between 0.86 and 0.94 in all the cases. The simplest problems were bulk data pull across the clouds and the reuse of stolen tokens, which were 0.94 and 0.93 respectively. The most difficult was the misuse of policy drift (0.86) since the actor made a valid use of permissions and the behavior signal had to make the decision independently. The framework experienced the highest increase in privilege over the fixed baseline in privilege escalation with a gain of 0.14.

These gains have a price of delay. Table 5 indicates that at 4,000 requests per second, a delay of 95 is 8.4 milliseconds with a fixed threshold baseline where 2.1 milliseconds above the threshold considered as a successful delay. The average delay was 4.7 milliseconds which included 1.6 to attribute fetch, 1.9 to score, 0.8 to evaluate and 0.4 to enforce. Comparing classical and deep learners to predict failures in clouds both of them report that simpler models achieve faster inferences [25]. Our cost of scoring concurs with this. Figure 5 indicates the increase of the p95 delay as a result of increasing load.

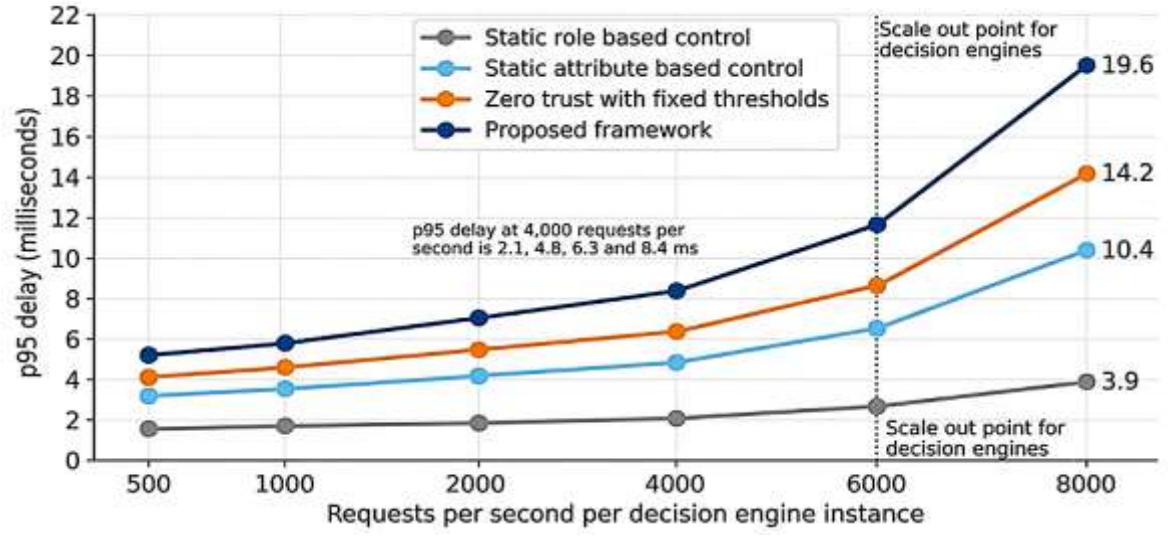


**Figure 5. Decision delay by mechanisms in milliseconds at 95 percentiles against request rate per instance of decision engine of the four methods.**

The framework was 19.6 milliseconds at 8,000 requests per second. Decision engines are then finished off by scaling when they are being called 6,000 times a second on a single instance. Containment in time will be seen in figure 6. Ultimately, the attacker achieved 8 workloads in 10 seconds in the framework and ceased at 12 following a compromise. When there were fixed thresholds, the count had increased to 41. It continued to expand in size and under static role-based control, it reached 164.

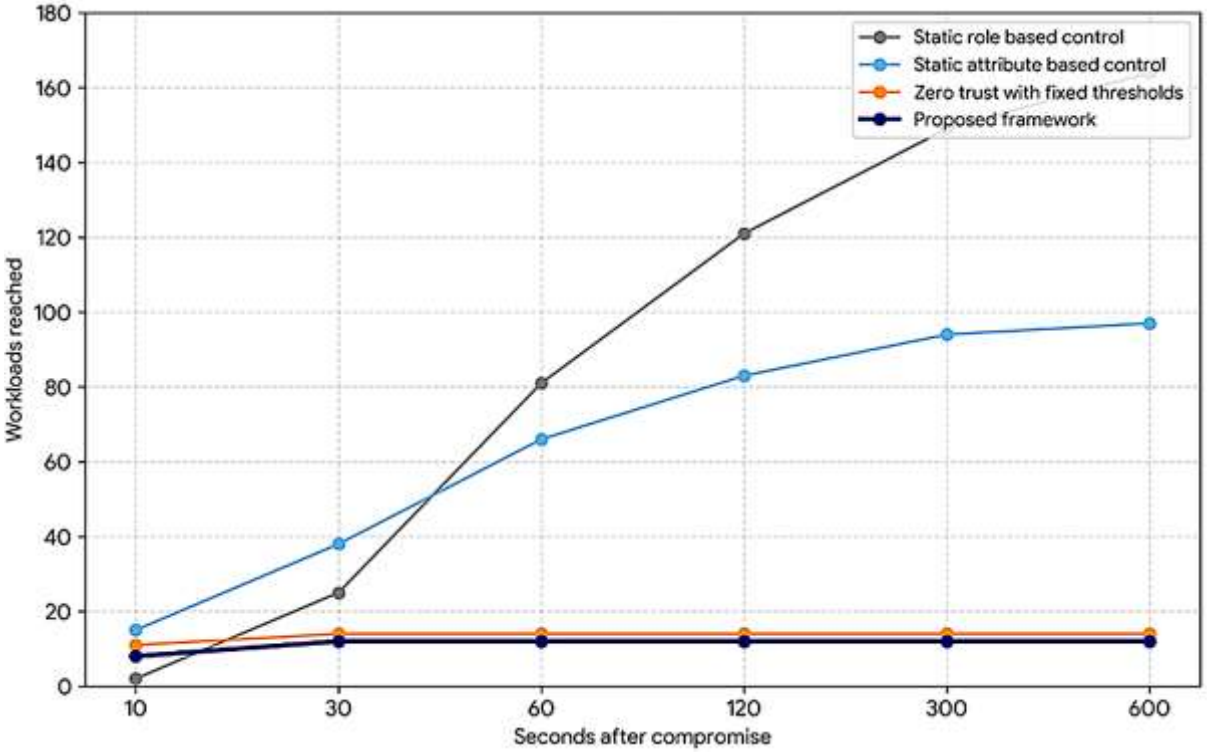



**Figure 6. Attackers were able to accumulate workload after being compromised within the first 600 seconds, by the various methods.**

## DISCUSSION

The ablation in Table 6 represents the contribution of each part. Behavior score removal yielded the highest results of being removed, reducing F1 from 0.91 to 0.83 and increasing time to revoke from 9.4 seconds to 41.2 seconds. This finding generalizes a diagnosis of failure study, in which ensemble of multiple types of evidence provided more consistent scores as compared to single source [26]. By eliminating trust decay, revocation was only 28.5 seconds, as long stale sessions remained trusted.

**Table 6. Ablation and false denial rate/mean time to revoke results using frameworks with a component removed.**

| Variant | F1 | False denial (%) | Mean time to revoke (s) |
|---|---|---|---|
| Full framework | 0.91 | 0.8 | 9.4 |

| | | | |
|---|---|---|---|
| Without behavior score | 0.83 | 0.9 | 41.2 |
| Without node posture | 0.87 | 0.8 | 14.7 |
| Without trust decay | 0.86 | 0.8 | 28.5 |
| Without adaptive thresholds | 0.88 | 1.9 | 10.1 |
| Without resource criticality | 0.85 | 0.7 | 12.3 |

Adaptive thresholds are primarily used to guard against false denials. In their absence, false denials increased by three times to 1.9 percent. The impact is indicated in figure 7 on a daily basis. The fixed thresholds on the surge day were 3.6 percent whereas the adaptive thresholds were 1.1 percent.

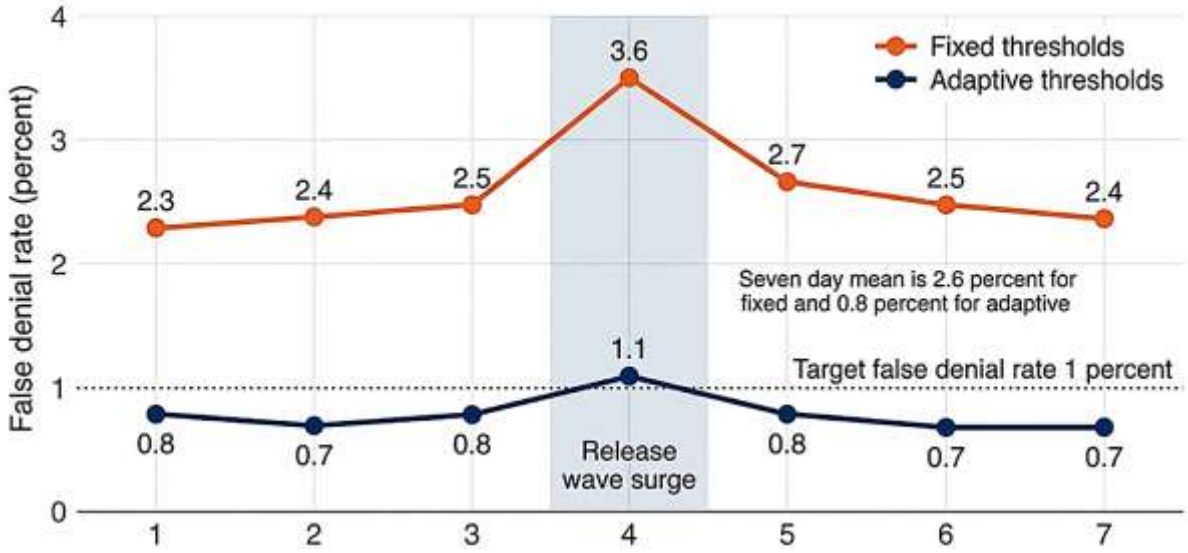



**Figure 7. False denial rate on a daily basis of the fixed and adaptive thresholds, over seven test days with a request surge on day four.**

A known cost is associated with anomaly-based security. A variety of papers about machine-learning based intrusion detection has identified that false alarms are relatively costly to operate and models based on previous traffic are not able to adapt to novel behavior [27]. This risk is reduced in our design under three aspects. Behavior represents just one of four signals. Middle band requests step up check and do not block the call. Labels are provided by operators and adjust the thresholds.

The outcomes are also in line with the perception of reliable and secure computing where an attack is an ill intent fault which needs to be contained before it can turn into a breakdown [28]. This is met directly by a short revocation time and a small blast radius.

There are limitations in the study. The injected attacks used in the testbed might be conducted in whatever manner, with real attackers acting in a different manner. The weights on table 2 were a result of judgment and need tuning of the field data. There was only a seven-day test and thus there was no coverage of seasonal patterns. Future will involve testing the framework using production traffic, and learning weights and comparing them with fixed weights. The operators may begin with a monitor only mode, preview the step up and deny results of two weeks and implement them.

## CONCLUSION

The paper has introduced the concept of risk adaptive dynamic workload zero trust authorization, in the context of hybrid and multi cloud infrastructure. It provides each request with a rating based on four signals, and multiplies the rating by resource criticality and trust decays, and adjusts the thresholds to recent traffic. It achieved F1 score of 0.91, mean revocation time of 9.4 seconds and blast radius of 12 workloads on a testbed of three clusters. It maintained false rejections at 0.8 percent and comprised 4.7 milliseconds on average to each request. The decision model can be executed on the premises, in either of the public clouds, without alteration of the rules. The framework provides the security teams a single decision model that fits every environment and a clear tuning of the decision model.

## REFERENCES

[1] W. Hasselbring and G. Steinacker, "Microservice architectures for scalability, agility and reliability in E-Commerce," 2017, pp. 243–246. doi: 10.1109/icsaw.2017.11.

[2] A. Balalaie, A. Heydarnoori, and P. Jamshidi, "Microservices architecture enables DevOps: migration to a Cloud-Native architecture," *IEEE Softw.*, vol. 33, no. 3, pp. 42–52, 2016, doi: 10.1109/ms.2016.64.

[3] N. Dragoni, "Microservices: yesterday, today, and tomorrow," in *Springer eBooks*, 2017, pp. 195–216. doi: 10.1007/978-3-319-67425-4_12.

[4] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *J. Internet Serv. Appl.*, vol. 1, no. 1, pp. 7–18, 2010, doi: 10.1007/s13174-010-0007-6.

[5] K. Hashizume, D. G. Rosado, E. Fernández-Medina, and E. B. Fernandez, "An analysis of security issues for cloud computing," *J. Internet Serv. Appl.*, vol. 4, no. 1, 2013, doi: 10.1186/1869-0238-4-5.

[6] 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.

[7] R. S. Sandhu and P. Samarati, "Access control: principle and practice," *IEEE Commun. Mag.*, vol. 32, no. 9, pp. 40–48, 1994, doi: 10.1109/35.312842.

[8] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," *Computer (Long. Beach. Calif).*, vol. 29, no. 2, pp. 38–47, 1996, doi: 10.1109/2.485845.

[9] V. C. Hu, D. R. Kuhn, D. F. Ferraiolo, and J. Voas, "Attribute-Based access control," *Computer (Long. Beach. Calif).*, vol. 48, no. 2, pp. 85–88, 2015, doi: 10.1109/mc.2015.33.

[10] P.-C. Cheng, P. Rohatgi, C. Keser, P. A. Karger, G. M. Wagner, and A. S. Reninger, "Fuzzy Multi-Level Security: An Experiment on Quantified Risk-Adaptive Access Control," in *IEEE Symposium on Security and Privacy (S&P)*, 2007, pp. 222–230. doi: 10.1109/sp.2007.21.

[11] L. Chen and J. Crampton, "Risk-Aware Role-Based access Control," 2012, pp. 140–156. doi: 10.1007/978-3-642-29963-6_11.

[12] D. R. D. Santos, C. M. Westphall, and C. B. Westphall, "A dynamic risk-based access control architecture for cloud computing," in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, 2014, pp. 1–9. doi: 10.1109/noms.2014.6838319.

[13] L. Xie, R. Wei, Y. Ning, and H. Yang, "Dynamic Risk Access Control Model for cloud platform," 2018, pp. 12–22. doi: 10.1007/978-3-030-00012-7_2.

[14] A. Almutairi, M. Sarfraz, S. Basalamah, W. Aref, and A. Ghafoor, "A distributed access control architecture for cloud computing," *IEEE Softw.*, vol. 29, no. 2, pp. 36–44, 2011, doi: 10.1109/ms.2011.153.

[15] Y. Cao, S. R. Pokhrel, Y. Zhu, R. Doss, and G. Li, "Automation and Orchestration of Zero Trust architecture: Potential solutions and challenges," *Mach. Intell. Res.*, vol. 21, no. 2, pp. 294–317, 2024, doi: 10.1007/s11633-023-1456-2.

[16] A. Atieh, P. Nanda, and M. Mohanty, "A Zero-Trust framework for industrial internet of things," 2023, pp. 331–335. doi: 10.1109/icnc57223.2023.10074295.

[17] G. Fragkos, J. Johnson, and E. E. Tsiropoulou, "Dynamic Role-Based Access Control Policy for Smart Grid Applications: An offline deep reinforcement learning approach," *IEEE Trans. Human-Machine Syst.*, vol. 52, no. 4, pp. 761–773, 2022, doi: 10.1109/thms.2022.3163185.

[18] C. Pahl, A. Brogi, J. Soldani, and P. Jamshidi, "Cloud Container Technologies: A State-of-the-Art Review," *IEEE*

*Trans. Cloud Comput.*, vol. 7, no. 3, pp. 677–692, 2017, doi: 10.1109/tcc.2017.2702586.

[19] S. Sultan, I. Ahmad, and T. Dimitriou, "Container security: issues, challenges, and the road ahead," *IEEE Access*, vol. 7, pp. 52976–52996, 2019, doi: 10.1109/access.2019.2911732.

[20] P. Jamshidi, C. Pahl, N. C. Mendonca, J. Lewis, and S. Tilkov, "Microservices: the journey so far and challenges ahead," *IEEE Softw.*, vol. 35, no. 3, pp. 24–35, 2018, doi: 10.1109/ms.2018.2141039.

[21] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," 2008, pp. 413–422. doi: 10.1109/icdm.2008.17.

[22] V. M. Patel, R. Chellappa, D. Chandra, and B. Barbello, "Continuous User Authentication on Mobile Devices: Recent progress and remaining challenges," *IEEE Signal Process. Mag.*, vol. 33, no. 4, pp. 49–61, 2016, doi: 10.1109/msp.2016.2555335.

[23] S. Henning and W. Hasselbring, "A configurable method for benchmarking scalability of cloud-native applications," *Empir. Softw. Eng.*, vol. 27, no. 6, 2022, doi: 10.1007/s10664-022-10162-1.

[24] X. Zhou, "Fault analysis and debugging of microservice systems: industrial survey, benchmark system, and empirical study," *IEEE Trans. Softw. Eng.*, vol. 47, no. 2, pp. 243–260, 2018, doi: 10.1109/tse.2018.2887384.

[25] T. N. T. Asmawi, A. Ismail, and J. Shen, "Cloud failure prediction based on traditional machine learning and deep learning," *J. Cloud Comput. Adv. Syst. Appl.*, vol. 11, no. 1, 2022, doi: 10.1186/s13677-022-00327-0.

[26] S. Zhang, "Robust failure diagnosis of microservice system through multimodal data," *IEEE Trans. Serv. Comput.*, vol. 16, no. 6, pp. 3851–3864, 2023, doi: 10.1109/tsc.2023.3290018.

[27] R. Sommer and V. Paxson, "Outside the closed world: on using machine learning for network intrusion detection," 2010, pp. 305–316. doi: 10.1109/sp.2010.25.

[28] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE Trans. Dependable Secur. Comput.*, vol. 1, no. 1, pp. 11–33, 2004, doi: 10.1109/TDSC.2004.2.

## BIOGRAPHIES OF AUTHORS



**Shubh Prabhat — Researcher Biography**

Shubh Prabhat is a Software Engineering Leader, Solution Architect, and Independent Researcher with over 18 years of experience in distributed systems, cloud computing, cybersecurity, and AI-driven enterprise solutions.

He earned his **Bachelor of Technology (B.Tech.) in Electronics and Communication Engineering in 2008**.

He is the Creator and Maintainer of **AeglysAI**, an open-source research initiative focused on AI-driven security automation, intelligent anomaly detection, Zero Trust architectures, and resilient distributed systems.

His research interests include Artificial Intelligence, Machine Learning, Cloud Security, Distributed Systems, Identity and Access Management, and Cybersecurity.

He is an **IEEE Senior Member** and actively contributes to scholarly research, peer reviewing, technical mentoring, and professional knowledge sharing.

**Research Profiles:**

- **ORCID:** https://orcid.org/0009-0008-1716-8789
- **Google Scholar:** https://scholar.google.com/citations?user=HrEzOfIAAAAJ
- **Research Initiative:** https://aeglysai.com/
- **Email:** sprabhat.research@gmail.com