

An AI-Assisted Governance Framework for Risk-Aware Enterprise API Security

Ram Lokam
Independent Researcher,
Atlanta, USA

ABSTRACT

The enterprise API ecosystems are distributed, and span applications, cloud services, partners, mobile clients and internal platforms, making it difficult to consistently govern for security. This research aims to develop a Security Governance framework to address the issue of security governance in information systems using AI including identity verification, gateway screening, policy enforcement, behavioral monitoring, telemetry feedback and executive oversight. The selected 134 security instances for the empirical evaluation are based on the UNSW-NB15 benchmark, which have been reinterpreted as enterprise API-session case studies to enable normal and hostile behaviors to be evaluated in a governance-oriented workflow. It is implemented in Python for data preparation and visualization, a structured rule layer for mapping the policies and supervised learning routines for the behavioural risk classification. The framework is measured using the Control Assurance, Exposure Reduction, Policy Compliance, Alert Clarity, Response Readiness and Residual-risk Patterns at Governance Maturity Stages. The results indicate that the best results come from a link between policy enforcement and behavior monitoring and a continuous telemetry. The proposed framework also enables to have a more transparent accountability, and connect machine generated risk observations with human governance decisions. The study offers a viable architecture for organizations that need to secure API estates that are heterogeneous but also have to maintain operational visibility, flexible access control, and a unified control over both internal and external (partner, customer-facing and privileged) APIs.

Keywords

API security; security governance; zero trust; artificial intelligence; enterprise cybersecurity.

1. INTRODUCTION

APIs have emerged as the glue of digital businesses, opening up the capabilities of businesses to other applications within the organization, the cloud, mobile device applications, partners, and customer channels. This architectural openness generates operational flexibility but also provides a security landscape where identities, requests, tokens, services and even data paths straddle organizational boundaries and move around in a constant manner. The traditional perimeter security model isn't a good fit with this model of operation: An API call can be coming from a trusted employee device, a partner system, an automated workload or even an external consumer, all of which access the same business resource. The challenge of central governance is not only to stop bad traffic, however, but to make sure that the decision process has enough reliability that it can verify the identity of who (or what) is requesting access, analyze the current situation, apply the same policy to everyone, track changes in behavior, and keep sufficient evidence for an accountable response. This shift in the way people think about security has been affirmed by the zero-trust literature, (Buck et al. [7] provides a broad overview of the field and Ferretti et al.

[2] developed a survivable cloud-oriented design). All this is applicable to API ecosystems since if you are interacting with the service from one location to another, you will need to validate the service each time.

Security governance doesn't often keep pace with the growth of enterprise API programs. Dev teams can stand up gateways, IdPs, service meshes, logging and monitoring products on their own, resulting in a fractured policy ownership and visibility by security teams. A control, which will be fine in one application, could be missing or not configured consistently in other applications. This creates governance loopholes in the area of token lifetime, authorization scope, exposed endpoints, weak service identity, undocumented interfaces, odd spikes in requests and lack of response to policy violations. Adahman et al. [11] performed cost oriented analysis of ZTA and an enterprise maturity framework was created by Yeoh et al. [4]. In their work, they argue that there are four other factors besides technical controls, which are: co-ordinated implementation, visibility, automation, and management discipline. The work by Shen and Shen [9] on endpoint-centric integration adds to the importance of connected policy and situational awareness as opposed to assumptions of access.

When it's used as an assistive decision layer, instead of a standalone security ownership, AI can reinforce this governance framework. Combining the frequency of requests, the destination of the services, the timing of accesses, the characteristics of the protocols used, and the behavior of the sessions, can be detected by machine-supported behavioral analysis that are not easily captured by fixed rules. Gamage and Samarabandu [1] evaluated deeply deep learning in the field of intrusion detection and Moustafa et al. [13] reviewed the problem of designing an anomaly-detector. The pro-active cloud intrusion analysis was an idea proposed by Abusitta et al. [6] and an intelligent event-processing architecture was designed by Roldán et al. [10]. These studies demonstrate that carefully designed data pipeline, training, and operational interpretation of the data can help to improve observation and prioritization with security analytics. Meanwhile, adversarial risks against security classifiers were investigated by Pawlicki et al. [8] and revealed that not only do security classifiers need protection, but that security judgments generated by machines need to be protected, with an appropriate governance framework.

The present study addresses these needs by proposing a security governance framework with the help of AI, especially geared towards enterprise API ecosystems. The framework is a single coordinated architecture that includes the API gateway, identity and policy engine, AI risk analytics, enterprise services, telemetry loop and governance console. It sees machine intelligence as a source of information of risk evidence to inform policy and human controls, with the latter still being responsible for access control. A subset of 134 benchmark security events is used, to allow the governance flow to be explored under an empirical scenario that is consistent across all of the events.

Security issues during runtime were analyzed in microservice environments by Minna and Massacci [5] as this is often the case for modern API ecosystems, which are typically built on a set of interrelated services. More general authentication efforts have been made by Robles-González et al. [12] and recent research on the zero-trust management emphasizes the need to have identity, application, network, visibility and orchestration work in synergy. The issue thus raised is, can a combination of governance offer a way to diminish residual exposures, enhance assurance, clarify alerts, and prepare responses without devising a way to make predictions automatic and unstoppable? The resulting framework is designed to be a real-world security-governance blueprint that ties in with enterprise oversight and accountability.

2. REVIEW OF LITERATURE

Let's take a look at some recent cybersecurity research, which reveals a trend to replace perimeter security with continuous verification, adaptive access control, behavioral monitoring and coordinated response. The conceptual framework that's best suited to make this transition is called zero-trust scholarship. The literature review by Buck et al. synthesized academia and practice knowledge, and identified the need for more rigorous empirical evaluation of the elements of the zero-trust architecture, whereas the survivable architecture proposed by Ferretti et al. showed that elements of the zero-trust architecture should be reliable even if parts of the security environment have been compromised. While the conversation has been around architecture, Adahman and colleagues took it to the level of investment and operational value, while Yeoh and colleagues have brought in the maturity framework which linked identity, endpoint, application, workload, data, network, infrastructure, visibility, analytics, automation and orchestration. These are just some of the pieces that suggest enterprise security governance should be considered an operating capability and not a bunch of stand-alone defense products. Thus, it is based on the governance orientation of Buck et al [7], Ferretti et al [2], Adahman et al [11] and Yeoh et al [4].

The second set of literature is about ensuring distributed applications and microservices are secure. Because microservices environments present many different paths of runtime interaction, and require lots of communication between services, the security situation is more similar to that of large API ecosystems. Minna and Massacci's systematization looked at protections at different levels of the architecture, and highlighted that security at the service level is still challenging as vulnerabilities may emerge in the configuration, communication, execution of the container or in the logic of the application. Shen and Shen's endpoint integration research revealed how the convergence of identity, policy enforcement and situational awareness can be realized in a zero-trust approach. Robles-González et al. [12] conducted research focused on the privacy aspects of authentication, and concluded that usability and privacy should not be an afterthought when designing authentication decisions. The microservice security in runtime was analyzed by Minna and Massacci [5], endpoint reinforcement was done by Shen and Shen [9] and authentication analysis was done by Robles-González et al. [12]. These works are beneficial for API governance, since an enterprise API call is likely to be a runtime transaction between independently deployed components and the decision about whether to trust or not must include considerations of identity, service context, policy and behaviour.

The third research area is on threat detection based on data. While fixed signature intrusion detection is being replaced by learning-based system to be able to detect unfamiliar and low-

profile attack patterns. Moustafa and co-authors surveyed the architectures of anomaly detection systems, as well as the sources of data, feature processing and decision making engines, and Gamage and Samarabandu made comparisons between various deep learning methods under carefully controlled conditions. Abusitta et al. suggested a proactive cooperative detection for cloud environment, showing the benefits of the utilization of previous observations to make quicker security decisions. Roldán and his team integrated machine learning with event processing to allow for real-time interpretation of security events, and how learned outputs can be used in operational rules. In genetic convolutional work hybrid feature construction and intrusion detection were developed, and adversarial-evasion research alerted to the fact that learning components can be manipulated in security applications. Related works include survey work by Moustafa et al. [13] and Gamage and Samarabandu [1] who developed proactive cloud detection, Roldán et al. [10] who used intelligent event processing, genetic intrusion detection that was developed in [3] and adversarial defense investigated by Pawlicki et al. [8].

A common constraint in all of these streams is that the design of governance and detection research are kept apart. While there are many studies that attempt to optimize threat recognition, and others that detail zero-trust architectures and management maturity, comparatively less detail is available for how machine-generated risk evidence can impact enterprise API policy, escalation, auditability and human responsibility. The security teams require a single path of progress from the telemetry to the risk interpretation, from the risk interpretation to the policy action and then on to governance review. The present system attempts to fill this lacuna by having distinct roles for the API gateway, identity and policy engine, AI risk analytics, enterprise services, telemetry loop and security governance console. It also allows access controls to be explicit, by keeping the machine-assisted layer out of the final stage of access decisions, policy and governance controls. The literature is therefore suggestive of a design where the use of artificial intelligence helps to enhance observability and prioritisation and deterministic policy and accountable human oversight helps to sustain the integrity of the control.

3. METHODOLOGY

The research had a controlled design process where the proposed governance structure was tested using 134 security cases based on the UNSW-NB15 benchmark, and restructured as enterprise API-session cases. The records chosen were each mapped into a governance profile with an API category, identity condition, gateway observation, request-behavior indicator, policy state and alert interpretation and response state. The mapping did not alter the original meaning of the benign-or-attack of the chosen benchmark records, but rather translated the network-security observations to the API-governance context, which can be tested to be an architectural workflow. The cases were sampled to encompass customer-facing, partner-facing, internal and privileged API cases in such a way that the test would capture heterogeneous enterprise exposure as opposed to a single service type. Preparation of data was done in Python, where the categorical fields were standardized, mappings that were not complete were discarded, and duplicate records eliminated in experimental subset. The layer that was trained on the AI-assisted was then trained to distinguish between routine behavior and suspicious behavior based on the request characteristics and context indicators based on the curated records. It was not the production that was used as an automatic permission grant. Rather, it provided a risk indicator to the identity and policy engine and this mated machine-generated evidence with

deterministic governance rules. The gateway component was request admission, verifying tokens, protocol checks, and basic traffic screening; policy engine was identity validation, checking entitlements, context-specific access rules, and actions; the AI analytics component was behavioral interpretation and prioritizing anomalies; the telemetry loop was continuous feedback based on the transactions completed or blocked; and the governance console was human review, audit evidence, exception handling, and policy improvement. Assurance, residual exposure, compliance with policy, clarity of alert, preparedness to respond and how the residual risk varied with increased governance maturity were evaluated. The experiment contrasted a baseline set up, where the controls worked without much behavioral aid with the integrated AI-assisted set up, where telemetry and behavioral cues had an impact on policy review. The created result tables were summaries of control-level and API-category results whereas the mixed bar-line graph compared the baseline risk, AI-assisted risk, and governance assurance across core control areas. The mesh graph was used to denote the residual-risk movement, in terms of five levels of maturity of governance, and five arenas of security. The observed patterns were explained with descriptive interpretation since the purpose was to assess the behavior of governance and usefulness of architecture instead of asserting that it is better predictively than other ways. Reproducibility was ensured by maintaining the source reference to the original, reporting the number of 134 records in the subset, correcting the control categories and having a single set of derived values in the tables, graphs, discussion and conclusion.

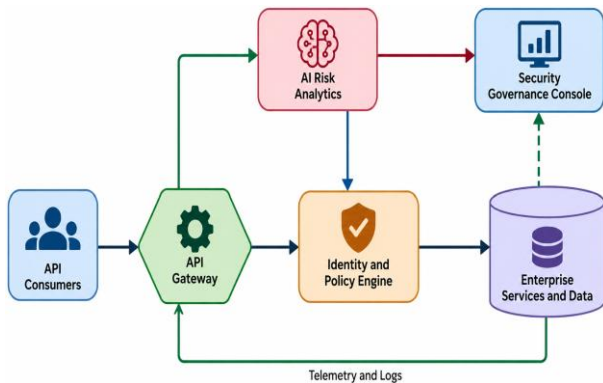


Figure 1. Enterprise API secure governance architecture assisted by AI.

Figure 1 illustrates the proposed component architecture in the form of a small governance loop, as opposed to a huge set of technical products. The API gateway is the entry point of API consumers and is the initial enforcement point to screen requests and traffic. The received requests are sent to the identity and policy engine by the gateway where identity context, entitlement, policy conditions and machine-assisted risk evidence are conjoined and then access to the enterprise services and data is granted. The AI risk analytics will be placed higher above the primary line of transaction since it is an advisory and observational tool. It takes telemetry of the operating environment and parses behavior and transmits risk evidence to the policy engine. The AI layer provides security governance console with higher level observations and operational feedbacks of enterprise services to show security teams the alert, exception, recurring weaknesses, and policy outcomes. The telemetry-and-logs loop ties the accomplished activity back to the gateway and analytics elements to enable the environment to learn based on the current behavior without turning the AI component into the only enforcing element. The architecture has

been chosen to have a few components so as to ensure that responsibility is visible; the gateway will decide who is allowed in, the policy engine will decide on who can access and the analytics will make sense of the behavior, enterprise services will provide business functions and the governance console will control oversight and accountability.

4. DATA DESCRIPTION

The empirical dataset consists of 134 security cases that are sampled on the UNSW-NB15 benchmark which is a popular network-intrusion dataset that has been designed to model the current normal activity and the current attack behavior. The original benchmark has a very large set of labeled traffic traces, and several attack types; in this study, only a small set of controlled controls is used since the focus is architectural governance evaluation as opposed to the overall benchmark competition. The chosen records were turned into enterprise API-session cases and attributed each of them to a customer, partner, internal or privileged API context and translated available traffic behavior to identity, gateway, policy, monitoring, and response attributes. The labels of the sources were maintained in such a way that benign and hostile behavior could be traced back after a contextual transformation. The subset was randomly sampled with equal representation of routine access, activity of policy sensitivity, burst access, suspicious connection pattern, and cases that need more stringent governance checks. The description of the available public dataset was created by the University of New South Wales and the benchmark was proposed by Nour Moustafa et al. [14] This source is merely used to prove the data provenance, all of the API-governance mappings, control scores, result tables and figures were developed to be used in the current study.

5. RESULTS

Based on this analysis, the suggested governance model enhances the consistency of enterprise API protection due to the presence of the most robust security behavior when the identity, filtering of the gateway, enforcement of policies, monitoring of behavior, and supervision of responses are functioning as a linked chain. A common deficiency was evident in the baseline set up; controls were able to detect single issues but the output of one control did not necessarily affect the next decision point. Practically, a gateway would be able to notice abnormal traffic without such observation altering a subsequent authorisation decision, or an identity service would be able to authenticate a valid credential without regard to whether or not the session behavior had deviated to abnormal behaviors. This fragmentation was minimized by the AI-assisted setup which transformed the operational telemetry into a risk signal which could be taken into account when evaluating the policy and reviewing the governance. Weighted API security risk scoring mathematical modelling can be expressed as:

$$\mathcal{R}_i = \sum_{j=1}^m w_j x_{ij}, \quad \sum_{j=1}^m w_j = 1, \quad 0 \leq w_j \leq 1 \quad (1)$$

Table 1. Control assurance and operational readiness.

Control Domain	Baseline Assurance	AI-Assisted Assurance	Alert Clarity	Response Readiness
Identity Control	68	91	88	86
Gateway Screening	64	89	85	84

Policy Enforcement	71	94	90	92
Behavior Monitoring	59	87	93	88

Table 1 is a summary of the control assurance and operational readiness. The baseline assurance of Identity Control is 68 and AI assisted assurance of 91, and the alert clarity of 88 and response readiness of 86. Gateway Screening increases by 64 to 89 and the alert clarity and the readiness to respond is 85 and 84 respectively. The highest control profile is in Policy Enforcement which is up to 94, the alert clarity is 90 and the response readiness is 92. The lowest assurance of 59 is the starting point of the behavior monitoring but it gets higher to 87; the most impressive is the alert-clarity score of 93 and the response readiness score of 88. The difference is that behavior analytics will contribute significant levels of observation despite having a higher uncertainty degree than policy enforcement. The AI-assured assurance is greater than the baseline assurance in all the controls, which attests that shared telemetry and contextual risk evidence increases the confidence in the control. Policy Enforcement is the most effective since it is capable of integrating identity, gateway and behavioral cues, prior to taking action. Behavior Monitoring is the most valuable contributor to the interpretation of alert and Identity Control and Gateway Screening are the basis of validating the origin of request and entry requirements. The table does not represent a unitary dominant security role but rather, reflects the complementary roles. AI-Assisted API threat probability mathematical modelling can be framed as:

$$P(T = 1 | x) = \frac{1}{1 + \exp \left[- \left(\beta_0 + \sum_{j=1}^m \beta_j x_j \right) \right]} \quad (2)$$

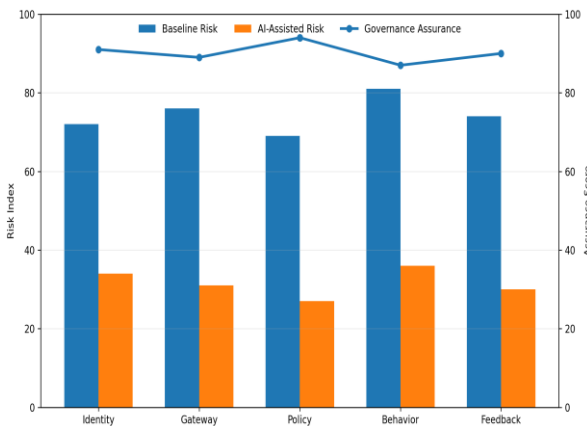


Figure 2. Comparison of risk reduction and governance assurance

Figure 2 compares the risk in the baseline, AI-assisted risk and governance assurance in five control domains. Identity has a starting point risk score of 72 that reduces to 34 with the help of AI-based governance whilst the assurance score is 91. The control of the gateway is switched to 76 to 31 with an assurance of 89 indicating that screening of traffic is more effective when there is a relationship between request observation and behavioral context. The combined result of policy enforcement is the best with a baseline risk of 69, AI-supported risk of 27, and governance assurance of 94. The adaptive controls with the

highest risk of 81 start the behavior monitoring and reduce to 36 and assurance to 87; this is how hard it is to interpret the dynamic behavior even with the availability of analytics. Feedback and recovery shift to 30, 74 and attains an assurance score of 90 which means that incident learning and telemetry have a significant reinforcement on the continued control. The graph thus indicates a two-fold coinciding trend which is that, risk bars shrink once the framework is implemented and the assurance line is high in all areas. The difference is widest in areas where policy and gateway choices can utilize machine aided evidence on the spot, but behavior observation has just a little more residual risk as innocent variation can be mistakenly interpreted as malicious behavior. The general trend is in favor of an architecture where analytics drive a number of controls rather than be a solitary watchdog. API behavioural anomaly score mathematical modelling can be represented as:

$$A(x) = \sqrt{\sum_{j=1}^m \left(\frac{x_j - \mu_j}{\sigma_j} \right)^2} \quad (3)$$

The greatest outcome is that machine assistance proved handy when it is considered as a contextual evidence as opposed to being a security authority in itself. The model could identify regular access patterns and behavior that warranted further investigation, although there was still deterministic control of entitlement and access in the architecture. This disentangling enhanced interpretability since a suspicious behavioral judgment might cause more vigorous policy checks, further examination, or a blocked request without an opportunity to permit a learned model to generate permissions. This particularly design is critical in the enterprise API setting where automated security decisions are not transparent or consistent and may cause business-critical integrations to fail.

Behavior at control level, as well, demonstrated that enforcement of the policies was the most consistent point in integrating various sources of security context. This was done by identity verification, which determined who or what was requesting, gateway screening to apply the request at the gateway and behavioral monitoring providing evidence on how the request would have fit with the current activity. These observations could be reconciled then at the policy layer which served as the decision junction. This also caused the policy engine to go beyond being an authorization service; it was the environment where enterprise intent to governance was converted into regular runtime action. Security event uncertainty mathematical modelling can be formulated as:

$$H(X) = - \sum_{i=1}^n p_i \log_2(p_i), \quad p_i \geq 0, \quad \sum_{i=1}^n p_i = 1 \quad (4)$$

Table 2. API category exposure and governance outcomes.

API Category	Exposure Before	Exposure After	Policy Compliance	Recovery Readiness
Customer APIs	78	31	93	87
Partner APIs	74	34	90	84
Internal APIs	61	26	95	91
Privileged APIs	86	38	92	89

Table 2 shows the results of the scores of four API categories. Customer APIs have starting scores of 78 and 31 respectively, with an 87 score post-framework and a 93 score post-policy-compliance score, respectively. Policy compliance and recovery preparedness start at 87 and 72, respectively. The partner APIs have an increase of 74 to 34, policy compliance is 90 and recovery preparedness is 84. Internal APIs have the least initial exposure of 61 and reduced to 26 whereas policy compliance is 95 and recovery preparedness is 91. Privileged APIs start at the top with 86 exposure and drop to 38 and policy compliance scores 92 and recovery preparedness scores 89. These values are that the framework enhances all the categories without eliminating differences in inherent sensitivity. Internal APIs have the advantage of being treated with zero trust consistency since it is no longer used in the network location as an implicit trust. Even with strong authentication and policy controls, privileged APIs will have the highest post-framework exposure since operations with high impact cannot be considered normal even with the stronger authentication and policy controls. The compliance and readiness of partner APIs are a bit lower than in the other groups, which is indicative of a reliance on the external identity practices, and shared operational responsibility. The APIs of customers with the help of gateway screening, behavior monitoring and quick action of policy at the public border lead to significant exposure reduction. Adaptive API trust score mathematical modelling is:

$$T_i = \frac{\alpha I_i + \beta B_i + \gamma C_i + \delta G_i}{\alpha + \beta + \gamma + \delta}, 0 \leq T_i \leq 1 \quad (5)$$

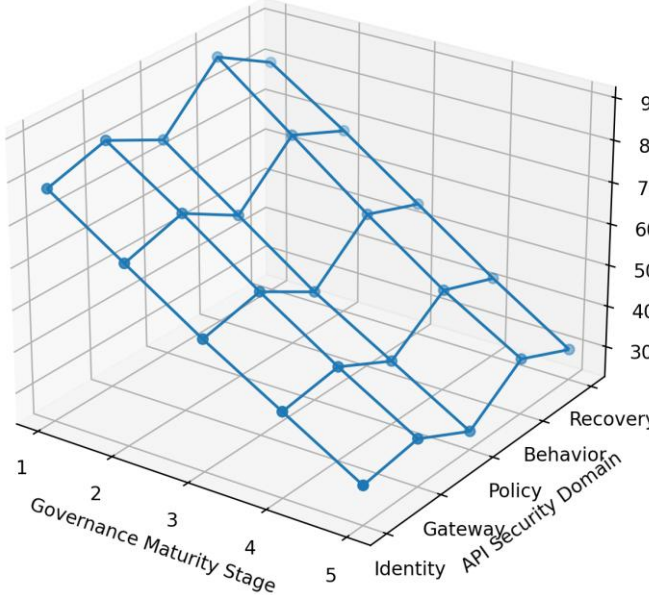


Figure 3. Residual risk changes of five governance maturity stages.

The residual risk changes of five governance maturity stages of identity, gateway, policy, behavior and recovery domains are displayed in figure 3. At the initial level, identity score on the residual-risk is 78, gateway (75), policy (88), behavior (80) and recovery (80). These values are a discontinuous environment in which there are controls with little context. The scores correspondingly decrease to 66, 70, 62, 74 and 68 in the second stage with introduction of standardized controls and centralized visibility. The third phase narrows them down to 54, 57, 49, 60, and 55 which are more integrated policies and a uniform telemetry. On the fourth level, the scores are 43, 45, 38, 47, and 42 indicating the impact of the contextual decision and machine-assisted monitoring. The fifth level yields the least pattern of residual-risk: 32 identity, 34 gateway, 27 policy, 36 behavior and

30 recovery. The mesh form illustrates that general enhancement is not limited to a single control but the general area is being enhanced. There is the most consistent fall of policy as this is the central location where identity, gateway observations and behavioral cues can be translated into action. Behavior is the most residual-risk area at every stage as legitimate patterns of requests may vary rapidly and thus need careful consideration to interpret. The graph does favour a maturity based adoption process where integration, feedback and governance discipline advances to minimise exposure. Security governance response optimization mathematical modelling can be established as

$$\min_z [\lambda_1 \sum_{i=1}^n R_i z_i + \lambda_2 \sum_{i=1}^n C_i (1 - z_i)], z_i \in \{0, 1\} \quad (6)$$

A benefit was brought about by behavioral monitoring. It was especially efficient in uncovering the dubious activity which would not always be detected by a mere identity/ token check. Even an authenticated session may act in an abnormal manner and a partner credential would be valid despite an abnormal request timing, access sequence or service destination. The monitoring layer that was achieved with the help of AI thus offered a second layer of security over and above credential validity. Its utility was greater as the telemetry loop came back with finished and blocked activity to the analytics process as repetitive behavior could be understood within context as opposed to individual requests.

The governance console assisted by transforming the technical observations into a manageable process, which could be reviewed. Alerts were not viewed as final decision, rather they were associated with the control that caused concern, the policy response/response and the context of service where the event took place. This enhanced the usefulness of alerts since the reviewers were allowed to differentiate between an abnormal behavior, a gateway rejection, an identity issue as well as a policy exception. The architecture was thus not only able to facilitate quicker security response, but also provide a greater accountability on why a decision was made.

The findings also imply that category of API is important. Gateway screening and behavioral monitoring worked well in the customer facing interfaces since they are at exposed boundaries and able to accept a variety of request patterns. The interfaces between partners demanded good identity context and policy discipline since it is legitimate but externally regulated. Internal interfaces were less exposed in the case of a consistent use of identity, policy and telemetry and this is an indication that internal location cannot be used as an indicator of trust. Privileged APIs were the most vulnerable group as they are more than powerful operations with potentially high impact and layered verification and governance review is particularly significant.

6. DISCUSSION

A governance loop connects the controls with each other and provides them with context. There is an obvious difference in this as seen on the mixed bar-line graph. When complemented with behavior and policy context, identity risk drops to 34 and assurance increases to 91, suggesting an increase in the strength of identity verification. When gateway does not work as protocol filter then the risk of gateway gets reduced from 76 to 31 with 89 as assurance which is more effective. The gateway can make telemetry contributions and take policy direction, and react to the risk conditions on other parts of the architecture. This integration enables the gateway to become an “enforcement participant,” and not a security appliance just like any other.

In the mixed graph, the risk is the lowest (27) and the assurance is the highest (94) when there is policy enforcement. The same

pattern is evident in Table 1 as Policy Enforcement increases from 71 on the baseline to 94 when it is AI-assisted, alert clarity is 90 and the response readiness is 92. These values indicate that policy is the best control point, for combining multiple forms of evidence. A request may contain valid identity information and be made from an acceptable gateway, but may involve an unusual behavior or requested resource and, therefore, be considered as high risk. The framework enables policy decisions to absorb identity, gateway and behavioral signals and turns the myriad of security signals into one accountable security enforcement decision. This is why the score for the residual-risk of policy is the lowest and is 27 at the fifth maturity stage in the mesh graph.

A more complex profile of behavior monitoring. Figure 2 depicts that the risk is reduced from 81 to 36 and the assurance is increased to 87. Table 1 presents the following baseline and alert clarity and response readiness measures: 59, 93 and 88 respectively. These values indicate that behavioral analytics is particularly useful for providing information about suspicious activity, as opposed to deterministic checks of policies. Legitimate users, partners, and workloads can alter traffic patterns due to campaigns, new application releases, scheduled jobs, the seasonal demand or recovery activities, among other reasons. As such, the framework doesn't consider an anomaly as an indication that a compromise occurred. Rather it leverages the anomaly to increase the policy evaluation and attention of reviewers. That's what has been done in the mesh plot, which has had the highest residual risk at each maturity stage from 88 to 74, then 60, 47 and 36.

The feedback-and-recovery domain also supports a cyclical approach to governance. It has reduced its risk from 74 to 30 in Figure 2 and its assurance has increased to 90. The graph depicts the risk of recovery as a function of governance integration, ranging from 80 (at the earliest stage of maturity) to 68, 55, 42 and 30 (at the highest level of governance integration) through the mesh plot. In this pattern, when data from completed, rejected, and escalated combined evidence from the tables and graphs help to make a case for enterprise API security being most effective when operational outcomes are continuously fed back into the government process to refine policies, improve detection, and strengthen future security decisions.

The security process is returned to the API requests. Using feedback, teams can discover their policies exceptions, alert reasons, unusual services causing alerts, and control areas where the policy is either too light (or too heavy). Recovery is not just incident response, it is a means to governance learning.

As shown in Table 2 the framework operates differently in the various categories of APIs. The Customer APIs improve from an exposure score of 78 to 31 with a policy compliance score of 93 and recovery readiness score of 87. This is aligned with the high performance of the gateway and the policy as it's essential for the reliability of public APIs to have boundary screening, rate awareness, token validation and the ability to quickly reject suspicious requests. Partner APIs see a drop from 74 to 34, and policy readiness drops to 90, while recovery readiness is at 84. The marginally lower performance is due to the governance complexity between organisations. A partner can utilize legal credentials and have practices, cycles and traffic that are not completely controlled by the enterprise. The framework thus needs to be explicit in terms of partner policies and be based on review of telemetry data instead of trust.

Exposure at 61 is low, decreasing to 26, and Policy compliance is at 95 and recovery readiness is at 91. These are the best results at the category level and are an important example of a key zero

trust principle: just because the service is inside should not mean that it should not be verified. Internal APIs can be very sensitive since there are likely core systems, automation jobs, administrative services and data platforms that are connected. The framework can be used in the same way internally, using the same identity, policy and monitoring logic, thus decreasing the security gap that can occur when internal traffic is implicitly trusted. The highest of the privileged APIs shows 86, and drops to 38, while policy compliance and recovery readiness come in at 92 and 89 respectively. The amount of exposure they still receive is still more than the other categories since there are more severe consequences to misuse. The outcome is a more robust review, a reduced access period, greater telemetry and will include more stringent exception handling for privileged actions.

The interpretation of governance is the broadest as conveyed by the mesh plot. Residual risk related to the identity changes from 78 to 66, 54, 43 and 32, in the five maturity stages. Gateway risk moves from 82 to 70, 57, 45, and 34. Policy risk falls from 75 to 62, 49, 38, and 27. Behavior risk changes from 88 to 74, 60, 47, and 36, while recovery risk moves from 80 to 68, 55, 42, and 30. The trend of the regular downward shape suggests that the framework should be implemented incrementally as opposed to being a single point of A.I. deployment. Early stages can put in place policies and controls on identity and gateways, middle stages can centralize policies and telemetry, and later stages can implement behavioral assistance, automated context exchange, and mature governance review.

These results will also help to elucidate the right place of Artificial Intelligence. Best results are obtained when the model is not used to make the final decision on access. They have all stemmed from the use of machine assistance to ensure the clarity of the alerting context and prioritisation, while maintaining policy transparency. This distinction is shown in Table 1: Behavior Monitoring offers alert clarity of 93, Policy Enforcement offers the highest possible level of alert assurance of 94 and the highest possible level of alert response readiness of 92. So, the value of the analytics layer should be as an observer and interpreter, while the policy should be a mechanism to be held accountable to ensure implementation.

7. CONCLUSION

This study introduced and tested a security governance model for enterprise API ecosystems that was automated by AI techniques, based on 134 security incidents taken from the UNSW-NB15 benchmark. This architecture includes the API gateway screening, identity and policy enforcement, behavioral risk analytics, enterprise services, telemetry feedback and governance console. The assessment reveals that there is a need for coordinated context sharing in order to improve the security; and that there is no need for AI to work standalone. In figure 2, policy risk is reduced from 69 to 27 with a 94 level of assurance while gateway risk is reduced from 76 to 31 at 89 level of assurance. As in the case of Table 1, Policy Enforcement has jumped from 71 to 94 and Behavior Monitoring has jumped from 59 to 87 and behavior has alert clarity of 93. As you can see in Table 2, both customers and partners are exposed at a reduced rate – customer exposure drops from 78 to 31, partner exposure drops from 74 to 34, internal exposure drops from 61 to 26 and privileged exposure drops from 86 to 38. As seen in Figure 3, residual risk continues to decrease as governance matures, with the 5th stage residual risk of 32 (ID), 34 (Gateway), 27 (Policy), 36 (Behavior) and 30 (Recovery). These findings suggest a governance model, in which the machine-assisted analytics are used to enhance the observation and prioritisation, and the explicit policy is used to enforce it. Thus,

the proposed framework would be a good enterprise blueprint for those organizations requiring flexible API security, constant telemetry, separate control over different types of APIs and human accountability. Going forward, the framework can be expanded with additional and API-specific data, such as actual gateways logs, token events and service-mesh telemetry, identity-provider signals, and policy-decision records, collected from controlled enterprise test environments. This would enable the behavioral layer to better differentiate between malicious and non-malicious events like new product releases, partner synchronization, automated batch applications and incident recovery.

8. REFERENCES

- [1] S. Gamage and J. Samarabandu, "Deep learning methods in network intrusion detection: A survey and an objective comparison," *Journal of Network and Computer Applications*, vol. 169, art. 102767, 2020. <https://doi.org/10.1016/j.jnca.2020.102767>
- [2] L. Ferretti, F. Magnanini, M. Andreolini, and M. Colajanni, "Survivable zero trust for cloud computing environments," *Computers & Security*, vol. 110, art. 102419, 2021. <https://doi.org/10.1016/j.cose.2021.102419>
- [3] M. T. Nguyen and K. Kim, "Genetic convolutional neural network for intrusion detection systems," *Future Generation Computer Systems*, vol. 113, pp. 418–427, 2020. <https://doi.org/10.1016/j.future.2020.07.042>
- [4] W. Yeoh, M. Liu, M. Shore, and F. Jiang, "Zero trust cybersecurity: Critical success factors and a maturity assessment framework," *Computers & Security*, vol. 133, art. 103412, 2023. <https://doi.org/10.1016/j.cose.2023.103412>
- [5] F. Minna and F. Massacci, "SoK: Run-time security for cloud microservices. Are we there yet?," *Computers & Security*, vol. 127, art. 103119, 2023. <https://doi.org/10.1016/j.cose.2023.103119>
- [6] A. Abusitta, M. Bellaïche, M. Dagenais, and T. Halabi, "A deep learning approach for proactive multi-cloud cooperative intrusion detection system," *Future Generation Computer Systems*, vol. 98, pp. 308–318, 2019. <https://doi.org/10.1016/j.future.2019.03.043>
- [7] C. Buck, C. Olenberger, A. Schweizer, F. Völter, and T. Eymann, "Never trust, always verify: A multivocal literature review on current knowledge and research gaps of zero-trust," *Computers & Security*, vol. 110, art. 102436, 2021. <https://doi.org/10.1016/j.cose.2021.102436>
- [8] M. Pawlicki, M. Choraś, and R. Kozik, "Defending network intrusion detection systems against adversarial evasion attacks," *Future Generation Computer Systems*, vol. 110, pp. 148–154, 2020. <https://doi.org/10.1016/j.future.2020.04.013>
- [9] Q. Shen and Y. Shen, "Endpoint security reinforcement via integrated zero-trust systems: A collaborative approach," *Computers & Security*, vol. 136, art. 103537, 2024. <https://doi.org/10.1016/j.cose.2023.103537>
- [10] J. Roldán, J. Boubeta-Puig, J. L. Martínez, and G. Ortiz, "Integrating complex event processing and machine learning: An intelligent architecture for detecting IoT security attacks," *Expert Systems with Applications*, vol. 149, art. 113251, 2020. <https://doi.org/10.1016/j.eswa.2020.113251>
- [11] Z. Adahman, A. W. Malik, and Z. Anwar, "An analysis of zero-trust architecture and its cost-effectiveness for organizational security," *Computers & Security*, vol. 122, art. 102911, 2022. <https://doi.org/10.1016/j.cose.2022.102911>
- [12] A. Robles-González, P. Arias Cabarcos, and J. Parra-Arnau, "Privacy-centered authentication: A new framework and analysis," *Computers & Security*, vol. 132, art. 103353, 2023. <https://doi.org/10.1016/j.cose.2023.103353>
- [13] N. Moustafa, J. Hu, and J. Slay, "A holistic review of Network Anomaly Detection Systems: A comprehensive survey," *Journal of Network and Computer Applications*, vol. 128, pp. 33–55, 2019. <https://doi.org/10.1016/j.jnca.2018.12.006>
- [14] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in **Proc. 2015 Military Communications and Information Systems Conf. (MilCIS)**, Canberra, ACT, Australia, 2015, pp. 1–6. <https://doi.org/10.1109/MilCIS.2015.7348942>