Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Feature Selection and Evidence Fusion for Reducing False Alarms in API Anomaly Detection

Collin Arnold Kabwama

ckabwama@miu.edu

Maharishi International University

Research Article

Keywords: anomaly detection, API security, feature selection, ablation, evidence fusion, behavioural baselines, zero trust

Posted Date: November 25th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11503719/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Feature Selection and Evidence Fusion for Reducing False Alarms in API Anomaly Detection

Collin Kabwama

Department of Computer Science, Maharishi International University, Fairfield, Iowa, USA

Corresponding author: ckabwama@miu.edu

Abstract

Per-identity behavioural detectors score each request on several features and usually fuse them by taking the largest per-feature deviation. We study what each feature contributes, how few are needed and how they should be combined in AI-Sentinel, an open-source API risk engine that fuses six capped z-scores by their maximum. In closed form, max fusion of d independent features flags $1-(1-0.0455)^d$ of normal requests at a z threshold of 2, 24.4% for six features, and it cannot add up evidence spread over several features, whereas a chi-square rule mapped to an equivalent z holds the false-alarm rate fixed and gains power when an attack's shift is spread. We generated two simulated suites of seven attacks and two benign look-alikes (1.46 million requests) and replayed them through the engine itself, then rescored all 63 feature subsets under both fusion rules and three engine configurations. Subsets were chosen on one suite and confirmed on the other and on a day of a public university access log. Drifting features drove false alarms: removing token age cut alarms on normal traffic from 42% to 3.0% with a 30-sample warm-up, and removing the endpoint-histogram pair cut them from 37% to 16% on real traffic as shipped. With the learning gate on, request rate and parameter count matched or beat all six features out of sample. Alarms were often credited to a feature other than the one attacked. Sum fusion cut false alarms but left weak multi-feature attacks undetectable at a usable threshold.

Keywords: anomaly detection; API security; feature selection; ablation; evidence fusion; behavioural baselines; zero trust

1. Introduction

Per-identity behavioural detectors describe each request by a handful of features, such as how often the caller has called this endpoint recently, how varied its endpoint use is, how old its

token is, how many parameters it sends and how large its body is, and compare each feature with what the caller usually does. Adding a feature is cheap, and each new feature seems to cover a new kind of abuse. The OWASP API Security Top 10 lists resource consumption, broken authentication, mass assignment of object properties and data exposure among the leading API risks [1], and a feature can be proposed for each. Whether a feature earns its place, however, depends on two things that are easy to overlook: what it costs in false alarms on normal traffic, and how its evidence is combined with the evidence of the others.

The combination rule matters because it decides whether many weak signals can add up. AI-Sentinel, an open-source per-identity API risk engine [2], computes a capped z-score for each of six features and takes the largest. In the language of multiple testing this is Tippett's minimum-p rule, or a Šidák union of per-feature tests [3, 4]. The rule is transparent and attributes each alarm to one feature, but it has two consequences that we make precise below. Every added feature adds its own false alarms, and a request that departs moderately from normal on several features at once can never score higher than its single largest departure. Recent research has investigated a range of AI-driven techniques for security and anomaly detection, spanning federated learning for decentralised identity [5], quantum machine learning [6], large language models for fraud detection [7], graph attention networks and generative adversarial networks [8], and the integration of graph neural networks with reinforcement learning [9]. Despite this progress in AI-based detection, further investigation is needed to determine how individual behavioural features contribute to detection performance and how effectively their combination improves per-identity API risk detection.

This study holds those choices fixed and asks the three questions that feature methodology raises: which features contribute to useful detection (ablation), how small the feature set can become without losing that usefulness (minimal sets), and whether combining several weak signals outperforms relying on one strong signal (fusion). Feature selection on the same data used to report its benefit is known to produce optimistic results [10, 11]. We therefore choose subsets on a development suite and report them on a held-out suite and on real traffic. We make the following contributions.

1. **Closed forms for max fusion.** With d independent features, max fusion at a z threshold of 2 flags $1 - (1 - 0.0455)^d$ of normal requests, 24.4% at d = 6, whereas a quadratic rule mapped to an equivalent z keeps 4.6% for every d. At a common false-alarm rate, max fusion detects a shift concentrated on one feature better, and quadratic fusion one spread over several. Caps of 2 standard deviations on parameter count and payload size

mean those features alone can reach the throttle band and the analysis threshold but never the block band.

2. **A feature-exercising scenario suite.** Seven attacks, each aimed at one feature family or at several weakly, and two benign look-alikes, generated twice with disjoint seeds and passed through AI-Sentinel's own feature extractor and decision engine (1.46 million requests).
3. **Exhaustive ablation.** All 63 non-empty feature subsets under two fusion rules and three engine configurations, with a per-request attribution of each alarm to the feature that decided it.
4. **Findings that held out of sample.** Removing a single non-stationary feature, or correlated pair, gave the largest cut in false alarms in eight of nine combinations of data set and configuration: token age on the simulated clients and the endpoint-histogram pair on real sessions. Request rate with parameter count matched or beat all six features on the held-out suite and the real log whenever the learning gate was on. Alarms were often attributed to a feature other than the one attacked. Chi-square fusion cut false alarms and, at a matched false-alarm rate, detected spread evidence better than max fusion, but did not make weak multi-feature attacks detectable at a usable threshold, did no better than the best single cue, and lost to max fusion on concentrated attacks.
5. **Open artefacts.** The suite generator, the masked and fused reference scorer, and all tables.

Section 2 reviews related work. Section 3 describes the features and the fusion rule. Section 4 presents the analysis, Section 5 the method and Section 6 the results. Section 7 discusses implications and threats to validity, and Section 8 concludes.

2. Background and related work

2.1 Feature selection and its pitfalls

Feature selection is usually framed as finding a subset that preserves predictive performance while reducing cost or noise. Filter methods rank features individually, and wrapper methods evaluate subsets with the model itself [12, 13]. For an unsupervised anomaly detector the model is fixed and has no training labels, so a wrapper evaluation needs labelled scenarios, and the same scenarios cannot then be used to report the result without bias. In gene-expression studies, selecting genes on the full data set before cross-validation produced error rates that were far too

optimistic [10] , and the general problem of selection bias in model evaluation is well documented [11]. Evaluations of security detectors face the same risk, and are additionally sensitive to how realistic their benign data are [14, 15] .

2.2 Combining evidence across features

Combining several test statistics into one decision is a classical problem. Tippett's rule rejects when the smallest p-value is small, which for z-scores is the same as rejecting when the largest |z| is large. Fisher's and Stouffer's rules sum transformed p-values, and Hotelling's T^2 and the Mahalanobis distance sum squared standardised deviations, accounting for correlation [4, 16]. No rule is uniformly best. Loughin's comparison showed that rules like Tippett's are most powerful when the evidence is concentrated in one test and sum rules when it is spread over many [4]. In multiple-comparison terms, a max rule over d features at a per-feature level α has a family-wise false-alarm rate of $1-(1-\alpha)^d$ [3]. Behavioural detectors in practice use both styles. Per-entity UEBA and fraud scores often take the largest deviation so that each alarm has a simple explanation [17], while multivariate control charts sum squared deviations [16]. The trade-off between the two has, to our knowledge, not been measured for per-identity API detection.

2.3 Explanations of alarms

An alarm that names the feature that caused it is easier to triage. Under max fusion each alarm has a single deciding feature, which an engine can report. That explanation is only useful if it is right, and we show that it often is not: alarms are frequently decided by a feature other than the one the attack targets, and sometimes by a feature unrelated to the attack, because that feature happens to be noisy or its baseline is new.

3. Features and fusion in AI-Sentinel

We study AI-Sentinel at commit `41abf74` (version 0.4.0) [2]. Table 1 lists the six features its statistical scorer uses. For each identity and endpoint the scorer keeps a running mean and standard deviation of each feature, divides each deviation by its learned standard deviation (floored at a fixed resolution), caps the result, and fuses the six capped z-scores by taking the largest:

$$S = \frac{1}{1+\exp\left(-3\left(\max_j z_j - 2\right)\right)}. \qquad (1)$$

The engine's default policy maps S to monitor from 0.2, throttle from 0.4, block from 0.6 and quarantine from 0.8, and learns a request into the baseline only if its action is allow or monitor, that is if $S < 0.4$ (the learning gate). It scores a new baseline's first requests with a fixed warm-up value of 0.4 and the action monitor. For analysis we call a request flagged when $S \geq 0.5$, that is when the largest capped z is at least 2. This is the classification threshold that AI-Sentinel's own detection evaluation uses, and it lies between the throttle and block bands. Because the gate uses the fused score, a feature that is noisy also stops the others from being learned.

Table 1. Features scored by AI-Sentinel (commit 41abf74).

Feature	Definition	Floor	Cap	Highest score from this feature alone
Request rate	requests by the identity to the endpoint in the rolling window	2	20	1.00
Endpoint entropy	Shannon entropy of the identity's 16-slot endpoint histogram	0.05	6	1.00
Endpoint concentration	largest share in that histogram	0.05	6	1.00
Token age	now minus `X-Token-Issued-At` (−1 if it or `Authorization` is missing or invalid; 0 if up to 300 s in the future)	1 s	6	1.00
Parameter count	number of query or form parameters	1	2	0.50
Payload size	`Content-Length`	1 byte	2	0.50

4. Analysis

4.1 False alarms accumulate under max fusion

Let the d features be independent and standard normal under normal behaviour, and let $p = P(|z| \geq c) = 2\Phi(-c)$ be one feature's false-alarm rate at the threshold $c = 2$, so $p = 4.55\%$. Max fusion flags a normal request when any feature exceeds c, so its false-alarm rate is

$$P_{\max}(d) = 1 - (1 - p)^d, \qquad (2)$$

which is 8.9% for two features, 13.0% for three and 24.4% for six (Fig. 1a). To hold the false-alarm rate at p, the threshold would have to rise with d. An alternative is a quadratic rule. With $T = \sum_j z_j^2$ over the k active features, T follows a chi-square distribution with k degrees of freedom under independent normal features, and mapping T to the two-sided normal z with the same tail probability,

$$z_{\text{eq}} = \Phi^{-1}\Big(1 - 1/2\ P(\chi_k^2 > T)\Big), \qquad (3)$$

and passing z_{eq} through Eq. (1) keeps the false-alarm rate at p for every k. With one feature, max and chi-square fusion coincide. We call this rule chi-square fusion; Hotelling's T^2 is its correlated generalization [16].

4.2 Which rule detects what

Suppose an attack shifts k of the d = 6 features by δ standard deviations each, with total size λ = kδ². At a common false-alarm rate α, max fusion detects the attack with probability $1 - \prod_j P\left(\left|N(\delta_j, 1)\right| \le c_d\right)$, where c_d solves $P_{\max}(d) = \alpha$, and chi-square fusion with probability $P(\chi^2_d(\lambda) > q)$, the tail of a noncentral chi-square beyond its central (1 − α) quantile q. The chi-square power depends only on λ, so it does not matter how the shift is spread. Max fusion is best when the shift sits on one feature and loses power as it spreads (Fig. 1b). At α = 1% and λ = 9, max fusion detected 44.8% of attacks concentrated on one feature against 35.1% for chi-square fusion, but only 18.9% of attacks spread as 1.5 standard deviations over four features against 35.1%. Monte Carlo simulation agreed with every closed-form value to within 0.15 percentage points. This is the weak-signal fusion question in its simplest form: a max rule cannot add up weak evidence, and a sum rule can.

4.3 Caps decide what a feature can do

Under Eq. (1), a feature capped at ζ can raise the score to at most $1/\left(1 + e^{-3(\zeta-2)}\right)$. AI-Sentinel caps parameter count and payload size at ζ = 2 to prevent unit changes from escalating a request. Each of them alone can therefore reach exactly the analysis threshold of 0.5, and so the throttle band, which also stops learning, but never the block band (Fig. 1c, Table 1). A second consequence matters for attribution. When any uncapped feature exceeds z = 2 on the same request, a capped feature can never be the largest, so it is never named as the cause of an alarm, however large its deviation.

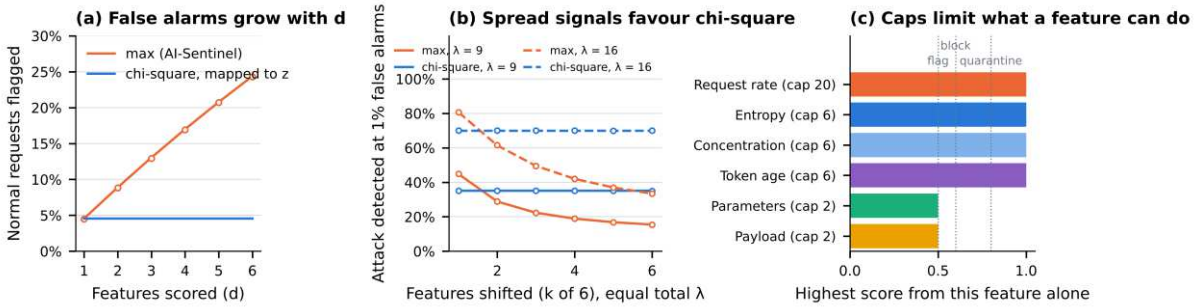

Fig. 1. Closed-form results (lines) and simulations (circles; in panel (a) for max fusion only). (a) False alarms of max fusion and of chi-square fusion as the number of scored features grows, Eqs. (2) and (3). (b) Detection at a 1% false-alarm rate of an attack of fixed total size λ spread over k of six features. (c) Highest score each AI-Sentinel feature can produce on its own, given its cap.

5. Method

5.1 Scenario suite

Simulated API clients send Poisson arrivals at 30 requests per minute to six endpoints with Zipf(1) usage shares. Two endpoints accept POST bodies with lognormal sizes (medians 800 and 20,000 bytes), each endpoint has a typical parameter count that changes by ±1 in two fifths of requests, and each client sends an `Authorization` header with a token re-issued every 15 minutes, so token age rises steadily from 0 to 900 seconds and resets. Each client sends 30 minutes of normal traffic, a 10-minute test phase and 10 more minutes of normal traffic. During the test phase, extra requests are added while normal traffic continues (Table 2). Seven attacks each aim at one feature family or, in the multi-weak attack, at several features weakly. Two benign look-alikes produce legitimate unusual behaviour and measure the cost of sensitivity. The control scenario adds nothing. Each scenario has 40 clients. The suite was generated twice with disjoint seeds: the development suite (731,167 requests) was used to choose subsets and the held-out suite (733,792 requests) only to report them.

Table 2. Scenarios. Extra requests are added during a 10-minute phase after 30 minutes of normal traffic; normal traffic continues.

Scenario	Label	Extra requests	Features it should move
Control	benign	none	none
Volume	attack	the client's own mix, so the total rate is 3×	rate
Focus	attack	3× the client's rate on its most-used endpoint	rate, concentration
Crawl	attack	the client's rate over 50 endpoints it has never used	entropy, concentration
Extra parameters	attack	the client's rate in its own mix, 4 extra parameters each	parameters, rate
Exfiltration	attack	the client's rate on the upload endpoint, bodies 10×	payload, rate
Stale token	attack	the client's rate in its own mix with a token issued 4 hours earlier	token age, rate
Multi-weak	attack	0.5× extra rate in the client's mix, 30% on 2 new endpoints, 1 extra parameter, bodies 1.5×	several, weakly
Bulk upload	benign look-alike	the client's rate on the upload endpoint, bodies 5×	payload, rate
New feature	benign look-alike	half the client's rate on 2 new endpoints	entropy, concentration

5.2 Real traffic

As a second held-out set we used one day of a public access log of a university information system, the Charles University SIS Access Log Dataset [18], replayed through the same components and keyed by the application's session key (844,099 requests on 23 May 2018), together with the volume, focus and crawl attacks that the second companion study injected into 150 real sessions each. The log contains no bodies and no token headers, so payload size and token age are constant on real traffic and only four features vary.

5.3 Engine, masks and fusion

Every request passes through AI-Sentinel's own `DefaultFeatureExtractor` and `SentinelDecisionEngine` under a simulated clock with one-second timestamps, using three operator configurations: as shipped (learning gate on, two-sample warm-up), the gate on with a 30-sample warm-up (W = 30), and the gate off. A reference implementation of the scorer then rescores the stored features with a feature mask, which removes features from the score and therefore from the learning gate, and with either fusion rule. With all six features and max fusion it reproduces the engine on every simulated request, with a largest score difference of 8.2×10^{-15} and no decision differences. A mask is a counterfactual: the engine has no setting that removes a feature from scoring.

5.4 Metrics and selection protocol

A request is flagged when its score is at least 0.5 (Section 3). The benign false-alarm rate is the share of the control scenario's requests and of every scenario's 30-minute history that is flagged; the look-alike rate is the share of look-alike requests flagged; and an attack's detection rate is the share of its injected requests flagged. We summarise a subset by J, the mean detection rate over the seven attacks minus the benign false-alarm rate, at this threshold. J rewards detection and penalises false alarms equally and needs no recalibration. Threshold-free separation is measured by the area under the ROC curve (AUC) of attack requests against the same clients' history, using scored requests only, because warm-up requests carry a fixed score. For each subset size, the best subset is the one with the highest J on the development suite. It is then evaluated, unchanged, on the held-out suite and on the real log, where J averages the three injected attacks. Each alarm is attributed to its deciding feature, the one with the largest capped z (the last on ties, as in the engine).

6. Results

6.1 More features, more false alarms

Table 3 and Fig. 2 report the ablation. With all six features, the shipped configuration flagged 97% of normal requests on the development suite, and the 30-sample warm-up 42%. Without the gate the rate was 8.5%. Removing token age gave the largest reduction of any removal in every configuration on both simulated suites. It cut the benign rate to 3.0% under W = 30, to 58% as shipped and to 1.9% without the gate, and raised J under W = 30 from 0.33 to 0.65. Token age rises steadily between token refreshes, so it is not stationary within a baseline's life,

and a baseline learned during one part of the cycle flags the rest. Removing it cost detection only where token age was the target: stale-token requests flagged under W = 30 fell from 100% to 72%, the remainder being caught by the extra request rate that the stale-token client also produced.

Request rate carried most of the useful detection. Without it, under W = 30, volume detection fell from 89% to 32% and focus detection from 99.4% to 56%. Parameter count supplied detection that no other feature did. The extra-parameter attack also added requests, so rate and token age caught part of it, but without parameter count its detection fell from 100% to 77% under W = 30, and from 86% to 42% without the gate. Removing entropy, concentration or both changed nothing on the simulated suites. Their Spearman correlation on normal traffic was −0.83 on the development suite and −0.94 on the real log, so each largely duplicates the other.

On real traffic the culprit was different. Token age and payload are constant there, but entropy and concentration behaved as if non-stationary within real sessions, as can be expected of a cumulative per-identity histogram; we infer this from the ablation and attribution rather than measuring it directly. As shipped, removing both cut the benign rate from 37% to 16% and raised J from 0.14 to 0.30, while volume detection fell only from 71% to 65%. Removing this pair gave the largest reduction in false alarms on real traffic as shipped and without the gate. Under W = 30 false alarms on real traffic were already low (4.2%), the pair contributed little (3.3% without it) and most remaining alarms were decided by request rate. Which feature does the damage therefore depends on the traffic and the configuration, but in eight of the nine combinations of data set and configuration the largest avoidable share came from a feature, or a correlated pair, whose normal values drift during a baseline's life.

Table 3. Ablation with max fusion. Development suite: normal and look-alike requests flagged, mean attack detection over seven attacks, and J; J on the held-out suite; normal requests flagged and J (three attacks) on the real log.

Configuration	Features	Normal flagged	Look-alike flagged	Mean attack flagged	J (dev)	J (held-out)	Real: normal flagged	J (real)
As shipped	All six	97%	99%	92%	−0.05	−0.04	37%	0.14
	− rate	96%	96%	91%	−0.04	−0.04	33%	0.01
	− entropy	97%	99%	92%	−0.05	−0.04	25%	0.23
	− concentration	97%	99%	92%	−0.05	−0.04	35%	0.16
	− token age	58%	66%	75%	0.17	0.16	37%	0.14
	− parameters	97%	99%	92%	−0.05	−0.04	34%	0.09
	− payload	97%	98%	92%	−0.04	−0.04	37%	0.14
	− entropy, concentration	97%	99%	92%	−0.04	−0.04	16%	0.30
	− parameters, payload	97%	98%	92%	−0.05	−0.04	34%	0.09

Configuration	Features	Normal flagged	Look-alike flagged	Mean attack flagged	J (dev)	J (held-out)	Real: normal flagged	J (real)
W = 30	All six	42%	76%	75%	0.33	0.31	4.2%	0.33
	− rate	37%	58%	57%	0.21	0.17	0.9%	0.01
	− entropy	42%	76%	75%	0.33	0.32	3.6%	0.34
	− concentration	42%	76%	75%	0.33	0.31	4.0%	0.33
	− token age	3.0%	65%	68%	0.65	0.65	4.2%	0.33
	− parameters	42%	76%	71%	0.29	0.28	4.0%	0.32
	− payload	42%	75%	74%	0.33	0.31	4.2%	0.33
	− entropy, concentration	42%	76%	75%	0.33	0.32	3.3%	0.33
	− parameters, payload	42%	75%	71%	0.29	0.28	4.0%	0.32
Gate off	All six	8.5%	34%	54%	0.45	0.45	6.8%	0.10
	− rate	8.0%	13%	33%	0.25	0.25	5.2%	−0.03
	− entropy	8.4%	34%	54%	0.45	0.45	4.5%	0.12
	− concentration	8.5%	34%	54%	0.45	0.45	6.5%	0.10
	− token age	1.9%	27%	40%	0.38	0.38	6.8%	0.10
	− parameters	8.5%	34%	47%	0.39	0.38	5.7%	0.09
	− payload	7.6%	31%	53%	0.45	0.45	6.8%	0.10
	− entropy, concentration	8.4%	34%	54%	0.45	0.45	3.8%	0.12
	− parameters, payload	7.6%	31%	46%	0.38	0.38	5.7%	0.09

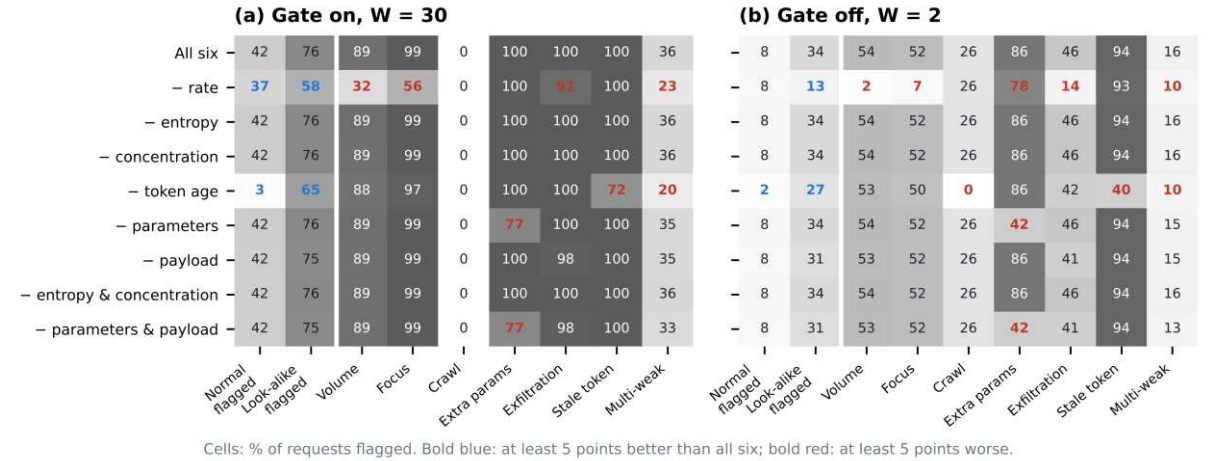

Fig. 2. Leave-one-out and leave-group-out ablation on the development suite under (a) W = 30 and (b) the gate off. Cells give the share of requests flagged (normal, look-alike and each attack). Bold blue marks a change of at least 5 points in the desirable direction relative to all six features, bold red one in the undesirable direction.

6.2 Two features match six

Table 4 and Fig. 3 show the best subset of each size, chosen on the development suite and then evaluated unchanged. As shipped, the best single feature was request rate (J 0.61 on the development suite, 0.61 held out, 0.34 on the real log), and the best pair was request rate with parameter count (0.66, 0.65 and 0.30). Both outperformed all six features (−0.05, −0.04 and 0.14). Under W = 30, rate with parameter count reached J 0.66 on the development suite and 0.66 held out, flagging 1.1% and 1.0% of normal requests, against 0.33 and 0.31 for all six. On

the real log it matched the full set (0.33 against 0.33). Under W = 30, no larger subset raised J by more than about 0.002 on any of the three data sets. The worst pair, entropy with concentration, had J −0.08 as shipped. Without the gate the ranking differed: the best subsets kept token age, because without the gate it is learned, and it was then needed for the stale-token attack. The development and held-out values agree to within 0.02 for every best subset under max fusion, so the selection did not overfit the development seeds. The real log, whose traffic and attacks were not used for selection, gives lower J values throughout, largely because the real crawl was almost never flagged (at most 2.2% of its requests in any configuration), which pulls the three-attack mean down by about a third. Its ranking of the 63 subsets agreed well with the development suite under W = 30 and without the gate (Spearman correlations of J of 0.84 and 0.76) but less well as shipped (0.38), where request rate alone was the best subset on real traffic (J 0.34), ahead of the development choice of rate with parameter count (0.30).

The subsets in Fig. 3 fall into two clouds. Every subset that contains token age sits at the right, with 36% to 44% of normal requests flagged under W = 30 on the two simulated suites, and every subset without it at the left, with at most 3%. Within the left cloud, detection rises by less than one point beyond rate with parameter count.

Table 4. Best and worst subset of each size under max fusion, chosen by J on the development suite; J on the development suite, the held-out suite and the real log.

Configuration	Size	Best subset (dev)	J (dev)	J (held-out)	J (real)	Worst subset (J, dev)
As shipped	1	rate	0.61	0.61	0.34	entropy (−0.08)
	2	rate + parameters	0.66	0.65	0.30	entropy + concentration (−0.08)
	3	rate + parameters + payload	0.58	0.58	0.30	entropy + concentration + token age (−0.07)
	4	rate + concentration + parameters + payload	0.34	0.34	0.23	entropy + concentration + token age + parameters (−0.06)
	5	rate + entropy + concentration + parameters + payload	0.17	0.16	0.14	rate + entropy + concentration + token age + payload (−0.05)
	6	all six	−0.05	−0.04	0.14	same
W = 30	1	rate	0.62	0.62	0.33	token age (−0.00)
	2	rate + parameters	0.66	0.66	0.33	entropy + token age (0.00)
	3	rate + parameters + payload	0.66	0.66	0.33	entropy + concentration + token age (0.00)
	4	rate + entropy + parameters + payload	0.66	0.65	0.33	entropy + concentration + token age + parameters (0.10)
	5	rate + entropy + concentration + parameters + payload	0.65	0.65	0.33	entropy + concentration + token age + parameters + payload (0.21)
	6	all six	0.33	0.31	0.33	same
Gate off	1	rate	0.32	0.32	0.13	concentration (−0.00)

Configuration	Size	Best subset (dev)	J (dev)	J (held-out)	J (real)	Worst subset (J, dev)
	2	rate + parameters	0.39	0.38	0.12	entropy + concentration (0.00)
	3	rate + token age + parameters	0.45	0.45	0.12	entropy + concentration + payload (0.01)
	4	rate + token age + parameters + payload	0.45	0.45	0.12	entropy + concentration + parameters + payload (0.13)
	5	rate + concentration + token age + parameters + payload	0.45	0.45	0.12	entropy + concentration + token age + parameters + payload (0.25)
	6	all six	0.45	0.45	0.10	same

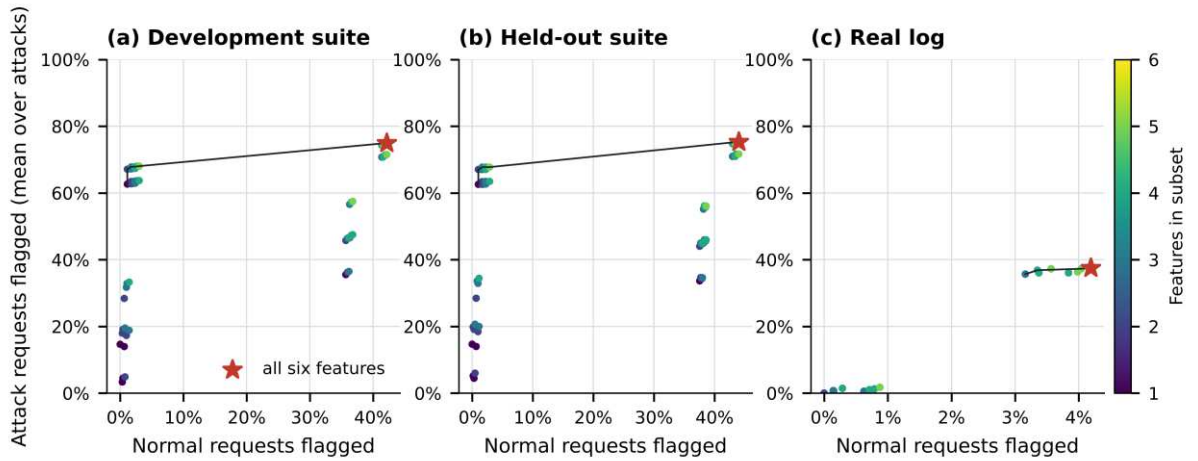

Fig. 3. All 63 feature subsets under W = 30 and max fusion: normal requests flagged against mean attack detection, coloured by subset size, on (a) the development suite, (b) the held-out suite and (c) the real log. The line joins the best subset of each size chosen on the development suite; the star marks all six features.

6.3 Alarms are often credited to a feature other than the target

Fig. 4 attributes every alarm to its deciding feature. Exfiltration requests, whose bodies were ten times the normal size, were flagged on 99.8% of requests under W = 30, but payload size decided only 2.0% of those alarms and request rate 98%. The exfiltrating client sent its whole extra rate to a rarely used endpoint, so that endpoint's request count rose far beyond 2 standard deviations, and payload, capped at 2, could not exceed it (Section 4.3). Rate is a genuine effect of this attack, but the attribution hid the feature that characterises it. The crawl, which sends requests to new endpoints, was flagged on 54% of requests as shipped and 26% without the gate, but token age decided 99.3% and 99.3% of those alarms. A crawl does not change token age. The new endpoints started new baselines, and a two-sample baseline of a steadily rising value flags later values systematically. Removing token age reduced crawl detection to 2.1% as shipped and 0.4% without the gate. These flags came from token age on new two-sample baselines, which any use of new endpoints triggers: the benign new-feature look-alike was flagged on 96% of its requests as shipped by the same mechanism. They do not specifically indicate crawling. On the real log, entropy and concentration decided 36% and 17% of the

shipped engine's alarms on normal traffic, but only 4.1% and 1.9% of its alarms on the volume attack, which request rate decided (92%).

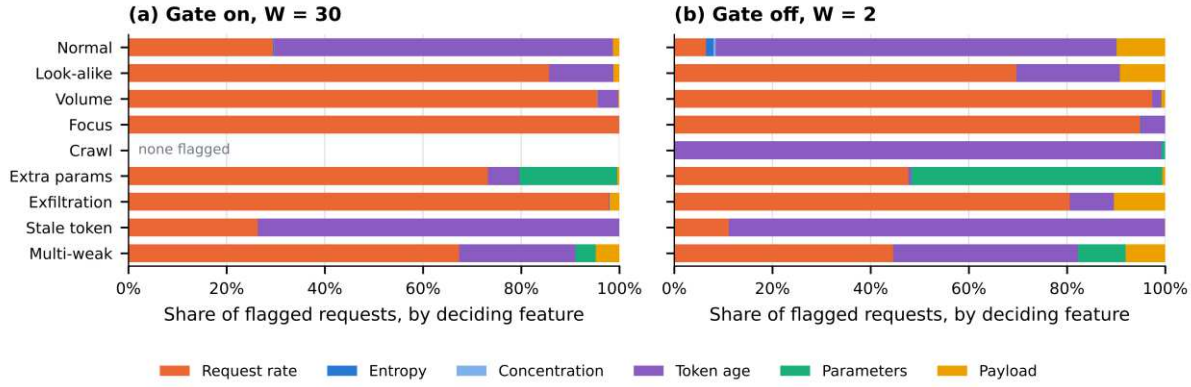

Fig. 4. Deciding feature of flagged requests on the development suite under (a) W = 30 and (b) the gate off, for normal traffic, look-alikes and each attack. Under W = 30 no crawl request was flagged.

6.4 Fusing weak signals

Table 5 and Fig. 5 compare max fusion with chi-square fusion. Chi-square fusion lowered false alarms sharply: on the development suite with all six features, from 42% to 10% under W = 30 and from 97% to 87% as shipped, in the direction predicted by Eq. (2), although not to the 4.6% it gives for independent normal features, because the features are correlated and not normal. It also lowered detection at the 0.5 threshold, from 75% to 40% on average under W = 30, so J did not improve (0.33 against 0.30), although the threshold-free separation did (mean AUC 0.82 against 0.84). The attacks in the suite are mostly concentrated on one feature, which is the case Section 4.2 predicts max fusion to favour, and the real attacks were concentrated on request rate. On the real log, chi-square fusion of rate and parameter count gave J 0.26 against 0.33 for max fusion under W = 30.

The multi-weak attack was built to favour fusion: each of its cues is small. No single feature detected it usefully. Under W = 30, request rate flagged 15% of its requests, parameter count 2.5% and payload 2.7%, and entropy, though it separated the attack best (AUC 0.84), flagged none at the 0.5 threshold. With five features (all but token age), max fusion separated the attack with AUC 0.80 and flagged 20% of its requests, and chi-square fusion separated it better than max fusion (AUC 0.84) but no better than entropy alone, and flagged only 0.5%. On the held-out suite entropy alone (AUC 0.85) again matched or exceeded both fused sets. Because the two rules flag different shares of normal requests at the 0.5 threshold (3.0% and 0.5%), we also compared them at the threshold that flags 1% of normal scored requests. There chi-square fusion of the five features detected 8.8% of multi-weak requests against 1.5% for max fusion

(14% against 1.2% held out), and for rate with parameter count chi-square fusion detected the extra-parameter attack better (94% against 70%). At a matched false-alarm rate, then, the sum rule did gain where evidence was spread, as Section 4.2 predicts. But parameter count alone detected 14% of multi-weak requests at the same rate, so fusion did not turn four weak cues into a usable detection. With token age included, max fusion appeared to detect 36% of multi-weak requests, but at a benign rate of 42%, and its AUC (0.56) shows that these flags did not distinguish the attack.

Table 5. Max and chi-square fusion. Development suite: normal / mean attack requests flagged, and mean AUC; J on the real log.

Configuration	Features	Max: normal / attack flagged	Max: AUC	Max: J (real)	Chi-square: normal / attack flagged	Chi-square: AUC	Chi-square: J (real)
As shipped	All six	97% / 92%	0.57	0.14	87% / 79%	0.54	−0.02
	No token age	58% / 75%	0.64	0.14	36% / 46%	0.69	0.04
	Rate + parameters	1.9% / 67%	0.84	0.30	1.5% / 60%	0.84	0.24
W = 30	All six	42% / 75%	0.82	0.33	10% / 40%	0.84	0.00
	No token age	3.0% / 68%	0.94	0.33	0.5% / 31%	0.93	0.04
	Rate + parameters	1.1% / 67%	0.94	0.33	0.2% / 59%	0.94	0.26
Gate off	All six	8.5% / 54%	0.82	0.10	1.8% / 11%	0.83	−0.02
	No token age	1.9% / 40%	0.85	0.10	0.2% / 3.8%	0.85	−0.02
	Rate + parameters	0.6% / 39%	0.79	0.12	0.1% / 23%	0.79	0.02

Table 6. The multi-weak attack: requests flagged and AUC for each single feature and for fused sets, on the development and held-out suites.

Configuration	Features and rule	Flagged (dev)	AUC (dev)	Flagged (held-out)	AUC (held-out)
W = 30	Rate	15%	0.72	13%	0.70
	Entropy	0.0%	0.84	0.0%	0.85
	Concentration	0.0%	0.50	0.0%	0.52
	Token age	19%	0.41	23%	0.41
	Parameters	2.5%	0.72	2.7%	0.72
	Payload	2.7%	0.52	2.8%	0.53
	All six, max	36%	0.56	40%	0.53
	All six, chi-square	7.1%	0.65	6.3%	0.62
	No token age, max	20%	0.80	18%	0.80
	No token age, chi-square	0.5%	0.84	0.5%	0.84
Gate off	Rate	7.3%	0.66	6.3%	0.65
	Entropy	0.0%	0.75	0.0%	0.75
	Concentration	0.0%	0.39	0.0%	0.40
	Token age	6.2%	0.45	8.0%	0.48
	Parameters	1.8%	0.67	2.1%	0.67
	Payload	1.7%	0.48	1.7%	0.48

Configuration	Features and rule	Flagged (dev)	AUC (dev)	Flagged (held-out)	AUC (held-out)
	All six, max	16%	0.64	18%	0.64
	All six, chi-square	2.0%	0.68	2.5%	0.68
	No token age, max	10%	0.72	9.6%	0.71
	No token age, chi-square	0.3%	0.74	0.4%	0.74

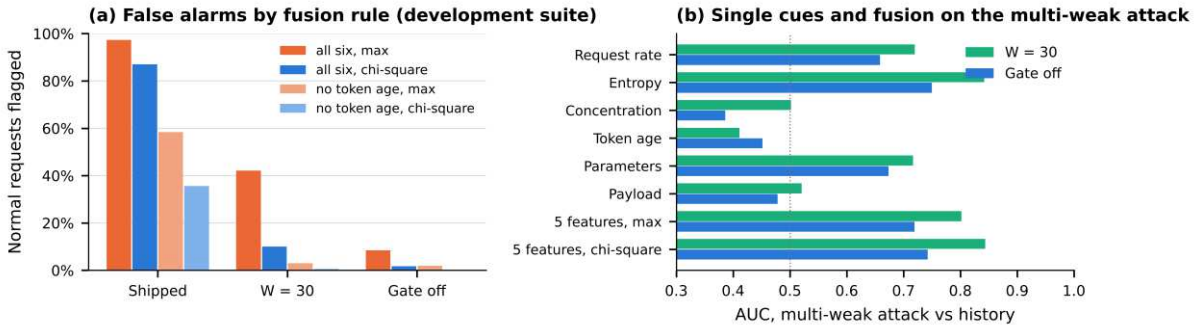

Fig. 5. (a) Normal requests flagged on the development suite with all six features and without token age, under max and chi-square fusion. (b) Separation of the multi-weak attack (AUC on scored requests) by each single feature and by the five features other than token age (four of them carrying the attack's weak cues) fused with either rule.

7. Discussion

7.1 Implications for design

A feature must earn its false alarms. Under max fusion every scored feature adds its own false alarms (Eq. (2)), and a feature whose normal values drift during a baseline's life adds many. Token age on the simulated clients and the endpoint-histogram features on real sessions were such features. Token age looked useful in isolation, because it detected the stale-token attack aimed at it; the histogram pair did not even do that, since the crawl aimed at it was rarely detected. Both cost more in false alarms on normal traffic than they returned in detection on the full suite. A feature should be admitted only after its benign false-alarm cost has been measured on representative traffic, and non-stationary features should be detrended before they are scored, for example token age as a fraction of the token's lifetime.

Small sets are enough for this engine. Request rate with parameter count matched or beat all six features on two simulated suites and on real traffic, under every configuration with the learning gate on. This is a statement about this engine, these features and these attacks, and it does not license deleting features in production. It does show that the default feature set is not the best one, that the difference is large, and that it can be measured with an offline harness of the kind we release.

Attributions need checking. Max fusion makes every alarm explainable by one feature, and an engine can report that feature. Our attribution shows that the explanation is often incomplete or misleading. Exfiltration was credited to request rate, a secondary effect, while the oversized bodies that characterise it were hidden by the payload cap. Crawls were credited to token age, which the crawl does not change. An analyst who read only these attributions would have misjudged the attack. Caps make this worse, because a capped feature can never be named once any uncapped feature crosses its threshold. Reporting the full vector of capped z-scores, rather than the largest alone, would let an analyst see when several features moved.

Choose the fusion rule for the threat. Max fusion is the better rule when attacks move one feature strongly, which was the case for most of our attacks and for the real ones. Sum rules are better when attacks move several features weakly and keep the false-alarm rate fixed as features are added. Neither rule turned our four weak cues into a usable detection. Weak signals need either features that are individually stronger for the threat in question, or longer windows over which weak evidence can accumulate, as in cumulative-sum monitoring [19].

7.2 Generality

Eq. (2) and Section 4.2 hold for any detector that fuses per-feature z-scores by their maximum, and the cap argument of Section 4.3 for any detector that caps them. The size of the effects depends on the features and traffic. The finding that a non-stationary feature dominates the false alarms should transfer to any per-identity detector that scores time-since-event features or accumulating histograms; the ranking of the particular features does not need to.

7.3 Threats to validity

Scenario design. The simulated clients and attacks are our choices. We generated the suite twice with disjoint seeds and chose subsets on one, but both suites share the same generator, so the held-out suite tests overfitting to seeds, not to design. The real log is the stronger test. It covers one day of one browser-based application, has no request bodies or token headers, and its attacks were injected by us.

Masks are counterfactual. AI-Sentinel has no setting that removes a feature from scoring. Masked and chi-square scores come from our reference implementation, which reproduces the engine exactly when nothing is masked and max fusion is used.

Selection criterion. J weighs detection and false alarms equally, averages over attacks with equal weight and uses a single threshold of 0.5, the engine's evaluation threshold. Other criteria,

such as a cost-weighted rate or detection at a fixed false-alarm rate, could rank subsets differently. We report AUC and the per-attack rates so that readers can apply their own.

Single version. All results concern one commit of one engine with its default floors and caps.

8. Conclusion

Adding a behavioural feature to a per-identity detector is not free. Under max fusion each feature adds its own false alarms, and a feature whose normal values drift adds many. In AI-Sentinel, removing one such feature, token age on simulated clients or the endpoint-histogram pair on real sessions, gave the largest reduction in false alarms in eight of nine combinations of data set and configuration. With the learning gate on, two features, request rate and parameter count, matched or beat all six on two simulated suites and on a day of real traffic. Alarms were often attributed to a feature other than the one the attack targeted, because noisy and uncapped features win the maximum. Fusing evidence by a sum rule kept false alarms in check and, at a matched false-alarm rate, detected spread evidence better than max fusion, but it did not make weak cues detectable at a usable threshold, did no better than the best single cue, and lost to max fusion on attacks concentrated in one feature. Feature sets and fusion rules should be chosen by measurement on held-out traffic, with their false-alarm cost reported next to their detections.

CRediT authorship contribution statement

Collin Kabwama did all the steps of this paper from Conceptualization, Methodology, Software, Formal analysis, Investigation, Writing – original draft to Writing – review & editing

Declaration of competing interest

Author declares no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The scenario-suite generator, the request-level harness, the masked and fused reference scorer, the analysis scripts and all summary tables will be made available upon acceptance of the paper or upon request by reviewers. All randomness is seeded, so both simulated suites can be regenerated from the code. The real access log is the Charles University SIS Access Log

Dataset, available from Zenodo under a CC BY 4.0 licence [18]; it is not redistributed with our material.

Ethics, Consent to Participate, and Consent to Publish declarations: not applicable**.**

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

[1] OWASP Foundation. "OWASP top 10 API security risks - 2023," 2023. [Online]. Available: https://owasp.org/API-Security/editions/2023/en/0x11-t10/.

[2] Kabwama, C. "AI-Sentinel: Application/API-layer behavioral-risk evaluation framework, version 0.4.0," 2026. [Online]. Available: https://github.com/CollinKabwama/ai-sentinel.

[3] Šidák, Z. "Rectangular confidence regions for the means of multivariate normal distributions," *Journal of the American Statistical Association* vol. 62 pp. 626–633, 1967, doi: https://doi.org/10.1080/01621459.1967.10482935.

[4] Loughin, T. M. "A systematic comparison of methods for combining p-values from independent tests," *Computational Statistics & Data Analysis,* pp. 47 467–485. , 2004, doi: https://doi.org/10.1016/j.csda.2003.11.020.

[5] Pinyi, E. K., Zeyeum, C. A., Oluwabukola, J. N. and Ikwuogu, H. P. "Federated IAM for Critical Systems: A Blockchain-Based Decentralized Identity Framework Enhanced by Federated Learning for Cross-Sector CNI Trust," *International Journal of Computer Applications Technology and Research,* vol. 14, no. 09, pp. 28 – 45, 2025, doi: https://doi.org/10.7753/IJCATR1409.1005.

[6] Olugbade, A. A., Eni, E., Titilayo, F., Kabwama, D. H. and Agbesi, C. A. "Hybrid Quantum Machine Learning for Anomaly Detection in Critical Infrastructure Sensor Networks," *Global Journal of Engineering and Technology Advances,* vol. 23, no. 03, pp. 333-361, 2025, doi: https://doi.org/10.30574/gjeta.2025.23.3.0198.

[7] Malingu, C. J. K., Businge, C. A., Agaba, P., Ankunda, I. A., Mugalu, I. A., Ariho, B. and Musinguzi, J. G."Application of LLMs to Fraud Detection," *World J. Adv. Res. Rev,* vol. 26, pp. 178-183, 2025, doi: https://doi.org/10.3390/a18120792.

[8] Kabwama, C. B., Malingu, C. J., Atuhaire, I. J., Ankunda, I. J., Ariho, J. G., Mugalu, B. and Musinguzi, D., "Graph attention networks for credit card fraud detection: a relational learning approach," *World Journal of Advanced Research and Reviews,* vol. 26, no. 3, pp. 2580-2585, 2025, doi: https://doi.org/10.30574/wjarr.2025.26.3.2400.

[9] BusingePius, A. I. A., Atuhaire, J. I., Nayebale, F. I., Ariho, J. G., Mugalu, B., Musinguzi, D., Malingu, C. J. and Kabwama, C. A."Adaptive financial fraud detection using graph neural networks and reinforcement learning," *World Journal of Advanced Research and Reviews,* vol. 28, no. 02, pp. 842-847, 2025.

[10] Ambroise, G. J. M. C. "Selection bias in gene extraction on the basis of microarray gene-expression data, Proceedings of the National Academy of Sciences " vol. 99 pp. 6562–6566. , 2002, doi: https://doi.org/10.1073/pnas.102102699.

[11] Cawley, G. C. and Talbot, N. L. C. "On over-fitting in model selection and subsequent selection bias in performance evaluation," *Journal of Machine Learning Research* pp. 11 2079–2107., 2010.

[12] Guyon, I. and Elisseeff, A. "An introduction to variable and feature selection," *Journal of Machine Learning Research* pp. 3 1157–1182, 2003.

[13] Kohavi, R. and John, G. H. " Wrappers for feature subset selection," *Artificial Intelligence* pp. 97 273–324, 1997, doi: https://doi.org/10.1016/S0004-3702(97)00043-X.

[14] Sommer, R. and Paxson, V. "Outside the closed world: On using machine learning for network intrusion detection," *IEEE Symposium on Security and Privacy,* pp. 305–316, 2010, doi: https://doi.org/10.1109/SP.2010.25.

[15] Arp, D., Quiring, E., Pendlebury, F., Warnecke, F., Pierazzi, F., Wressnegger, C., Cavallaro, L. and Rieck, K. , " Dos and don'ts of machine learning in computer security," *31st USENIX Security Symposium (USENIX Security 22),* pp. 3971–3988. , 2022, doi: https://www.usenix.org/conference/usenixsecurity22/presentation/arp.

[16] Hotelling, H. "The generalization of Student's ratio. "The generalization of Student's ratio," *Annals of Mathematical Statistics,* pp. 2 360–378. , 1931, doi: https://doi.org/10.1214/aoms/1177732979.

[17] Bolton, R. J. and Hand, D. J. "Statistical fraud detection: A review," *Statistical Science,* pp. 17 235–255. , 2002, doi: https://doi.org/10.1214/ss/1042727940.

[18] Maňásek, M. and Tůma, P., "Charles university SIS access log dataset," 2019, doi: https://doi.org/10.5281/zenodo.3241445.

[19] E.S. Page, "Continuous inspection schemes," *Biometrika,* pp. 41 100–115, 1954, doi: https://doi.org/10.1093/biomet/41.1-2.100.