

Output Policies for Anomaly Detection with Homomorphic Encryption: Threshold Resolution, Decision Preservation, and Numerical Stability

Subeen Cho¹[0009–0004–4352–8021], Jiwon Bang¹[0009–0006–5453–1722],
Minseo Kim¹[0009–0008–1979–5778], Seungwon Lee¹[0009–0009–9583–1387], and
Hwajeong Seo¹[0000–0003–0069–9061]

¹Department of Convergence Security,
Hansung University, Seoul (02876), South Korea,
{chosubin1208, bgjiiw0n754, msqoo0212, dkajdfhd1, hwajeong84}@gmail.com

Abstract. Homomorphic encryption computes anomaly scores without exposing inputs, but released scores and decisions can reveal a private threshold. We evaluate whether output policies limit numerical threshold information while preserving existing decisions, and whether this limitation also impedes decision prediction on other inputs. Policies distinguish internal-score and released-score decisions and share decision-loss and score-distortion constraints. Thresholds were estimated in all 45 plaintext experiments combining three intrusion-detection datasets, three scoring models, and five seeds. All 45 CKKS experiments combining the same datasets, five autoencoder seeds, and three independent keys yielded threshold-containing intervals after revalidating supplied starting intervals. A 16-query cap stopped every plaintext baseline search but served only 16 of 512 separate usage requests, reducing availability. Under random splitting, released-score quantization preserved 99.864% of decisions on average while limiting numerical threshold precision. Surrogates trained on 128 decision-only responses achieved mean balanced accuracy 0.817. With training-only preprocessing and a CIC-IDS2017 weekday split, model-wise balanced accuracies were 0.816–0.911. Separating time and Flow IDs across five resampled evaluations reduced model-wise means to approximately 0.5, showing that prediction transfer depends on data splitting. Threshold-dependent reselection of quantization width changed responses that matched under fixed widths. Selected CKKS boundary tests yielded one released-decision mismatch in 57 queries. Numerical threshold information and service decision prediction therefore require separate protection goals. Output policies must assess decision preservation and response rate alongside policy selection, data splitting, and numerical boundary conditions.

Keywords: Homomorphic encryption · Anomaly detection · Output policy · Threshold leakage · Decision preservation

1 Introduction

Network anomaly detection identifies abnormal activity from network-flow features. Fully homomorphic encryption (FHE) allows anomaly scores to be computed without exposing sensitive inputs to the computation server [6]. Research on CKKS-based log anomaly detection has evaluated detection performance and the computation and communication costs of encrypted inference [18]. However, returning both scores and normal/anomalous decisions allows a requester to observe which scores are treated as normal. Services that keep their thresholds private therefore need to assess the operational criteria revealed by public responses separately from input confidentiality.

Prior work has continued to address information exposure and functional replication through prediction-API model extraction, utility-aware output perturbation, and model protection in encrypted inference [17,15,4]. However, threshold-based detection requires an additional distinction because limiting detailed numerical information about a threshold does not necessarily impede prediction of service decisions on other inputs. Numerical threshold information and service decision prediction should therefore be evaluated as separate protection goals.

The score used for decisions also matters. Transforming a released score while retaining the internal-score decision differs from deciding on the transformed score. Even with the same transformation, these rules change the relation between observed scores and decisions, affecting both threshold information and decision preservation.

This study examines whether numerical threshold information can be limited while preserving existing decisions, and whether that limitation also impedes service decision prediction. We compare quantization, noise, and decision-only outputs under the same utility criteria. For query limits, we also measure the response rate for ordinary use. Threshold-dependent policy selection, chronological data splitting, and CKKS approximation error are then examined to determine whether the results hold beyond fixed evaluation conditions. Through this analysis, we clarify the need to distinguish numerical threshold protection from service decision protection and to evaluate information exposure together with utility preservation when designing output policies.

1.1 Main contributions

- **Decision-source semantics.** We distinguish internal-score and released-score decisions and identify when threshold-interval containment and fixed-grid response equivalence hold. This shows that the same output transformation can expose different information depending on the decision source.
- **Utility-constrained output-policy evaluation.** Across three datasets and three scoring models, we separately measure threshold resolution and decision prediction under common utility constraints. Query controls are evaluated through observation cost and response rate, showing the need to consider both protection goals and service utility.

- **Deployment-sensitive analysis.** We assess fixed-policy results under threshold-dependent reselection, training-only preprocessing, and time/Flow ID separation. Actual CKKS errors are also compared with threshold and quantization-boundary margins to show the need for deployment-specific validation.

2 Background and Related Work

2.1 Anomaly Scores and Threshold Decisions

Score-based anomaly detection quantifies the degree of abnormality and uses a threshold to determine whether an input is normal or anomalous [5]. A network flow can be represented by a feature vector x , with an anomaly score $s(x)$ defined so that larger values indicate stronger signs of anomaly. Even when the scoring function is unchanged, moving the threshold can change false positives and false negatives.

In reconstruction-based detection, an autoencoder (AE) computes an anomaly score from the discrepancy between the input and its reconstruction [13]. Centroid-based and Mahalanobis scores measure deviation from normal training data, with the Mahalanobis score also accounting for covariance [10]. To compare these three approaches, we use scores normalized by input dimension p in Eq. (1).

$$\begin{aligned} s_{\text{AE}}(x) &= \frac{1}{p} \|x - \hat{x}\|_2^2, \\ s_{\text{C}}(x) &= \frac{1}{p} \|x - \mu\|_2^2, \\ s_{\text{M}}(x) &= \frac{1}{p} (x - \mu)^\top (\Sigma + \rho I)^\dagger (x - \mu). \end{aligned} \tag{1}$$

In Eq. (1), $\hat{x}$ is the AE reconstruction, while μ and Σ are the mean and covariance of normal training data. The regularization coefficient is $\rho > 0$, and $\dagger$ denotes the pseudoinverse used in numerical computation. Because the three scores differ in computation and distribution, absolute error alone is not used to compare estimation difficulty across models.

Detection quality is evaluated against ground-truth labels, whereas decision agreement measures how well the original detector decisions are preserved after applying an output policy. Response rate is the fraction of all requests that return a score or decision. Therefore, high decision agreement does not imply high detection quality or a high response rate.

2.2 Encrypted Score Computation with CKKS

CKKS supports approximate addition and multiplication over encrypted real or complex values [6]. An input vector is encoded with scale Δ , encrypted under a

public key, and evaluated as an arithmetic circuit f . The secret-key holder then decrypts and decodes the result, as shown in Eq. (2).

$$\begin{aligned} c_x &= \text{Enc}_{pk}(\text{Encode}_\Delta(x)), \\ \text{Decode}(\text{Dec}_{sk}(\text{Eval}_f(c_x))) &\approx f(x). \end{aligned} \quad (2)$$

In Eq. (2), pk and sk denote the public and secret keys. Approximation error arises from encoding and homomorphic evaluation. CKKS also supports SIMD-style slot packing and rotations for matrix–vector computation.

Ciphertext multiplication increases the scale, while rescaling divides by a modulus-chain factor q_ℓ and consumes one modulus level. The scale updates are given in Eq. (3).

$$\Delta_{\text{mul}} = \Delta_1 \Delta_2, \quad \Delta_{\text{rescale}} = \frac{\Delta_1 \Delta_2}{q_\ell}. \quad (3)$$

Scale and modulus-chain parameters must therefore balance numerical precision and multiplicative depth [6].

AE linear layers, square activations, and mean squared reconstruction error can be evaluated as a polynomial circuit over ciphertexts. Let $s(x)$ denote the plaintext score of the same circuit and $d(x)$ the decrypted score. Their difference is $e(x) = d(x) - s(x)$. Because even small errors may affect outputs near a threshold or quantization boundary, Section 3 evaluates both error magnitude and distance to the boundary.

2.3 Output Policies and Protection Goals

An output policy determines the type and precision of information released by the service and thus what an external requester can observe. Raw-score release returns the decrypted score, quantization maps it to representative values, and noise addition perturbs it randomly. Decision-only release returns only the normal/anomalous decision. In contrast, query caps and similarity-based rejection restrict which requests receive responses rather than modifying returned values. Output transformations may therefore alter released scores or decisions, whereas request controls reduce observation opportunities at the cost of a lower response rate.

Public responses reveal information about model outputs and decision behavior, so repeated observations can support internal-parameter recovery or functional replication [17,15]. In threshold-based detection, exact threshold recovery and decision reproduction on other inputs are distinct objectives. Multiple thresholds can induce the same decisions, so service behavior may remain predictable even when the exact threshold cannot be distinguished. We therefore evaluate threshold information and decision prediction separately while also considering decision preservation, added errors, score distortion, and response rate.

2.4 Related Work

Prior work has mainly addressed anomaly detection in encrypted environments, model-extraction defenses for public outputs, and observation control under repeated queries. SigML [18] evaluates anomaly detection on CKKS-encrypted logs together with computation and communication costs. Byun et al. [4] study model-extraction protection in HE/MPC inference while comparing accuracy and protection. These studies address encrypted computation and model protection, but they do not directly separate numerical information about a private threshold from prediction of service decisions after decryption.

Output and query controls have also been studied to reduce information exposure. ModelGuard [15] constructs model-extraction defenses, including quantization, under output-distortion and decision-preservation constraints and analyzes conditions where online quantization reveals additional information. Kuppa et al. [8] use binary anomaly-detector responses to study decision boundaries and evasion. Blacklight [9] detects and rejects similar queries to limit repeated observation. These studies focus on model extraction, boundary observation, or query rejection, but they do not compare numerical threshold resolution and service decision prediction as separate protection goals.

Table 1 compares the mechanisms and primary focus of representative studies with this study. We distinguish internal-score and released-score decisions and evaluate numerical threshold resolution and service decision prediction as separate protection goals. We also compare output policies under utility constraints and analyze threshold-dependent quantization-width reselection. Time/Flow ID separation and actual CKKS boundary errors are further examined to assess the scope of results obtained under fixed conditions. These analyses show that output-policy protection should be interpreted together with decision reproducibility, service utility, and deployment conditions.

Table 1. Mechanisms and primary focus of representative studies

Study	Encrypted inference	Output transform	Query rejection	Primary focus
SigML [18]	✓	–	–	Encrypted anomaly detection
Kuppa et al. [8]	–	–	–	Boundary observation and evasion
ModelGuard [15]	–	✓	–	Model extraction
Byun et al. [4]	✓	✓	–	HE/MPC model extraction
Blacklight [9]	–	–	✓	Similar-query rejection
This study	✓	✓	△	Threshold and decision inference

✓: evaluated; △: plaintext reference; –: outside scope.

3 Service Model and Output Policy Analysis

3.1 Trust Boundaries and Observation Scope

In this study, a requester is an external user who submits encrypted inputs to the service and receives either a released score with a decision or the decision alone, depending on the output policy. The requester can issue repeated queries through the normal service interface and record both submitted inputs and public responses. The adversary is assumed to have the same privileges as an ordinary requester. Its goal is not to recover the secret key or the model itself, but to narrow numerical information about the private threshold or predict service decisions on new inputs from repeated responses. We therefore assume that the adversary may know the output transformation and its configuration, but cannot access the secret key, model weights, prerelease scores, or audit records. We also assume that the computation server and trusted modules do not collude.

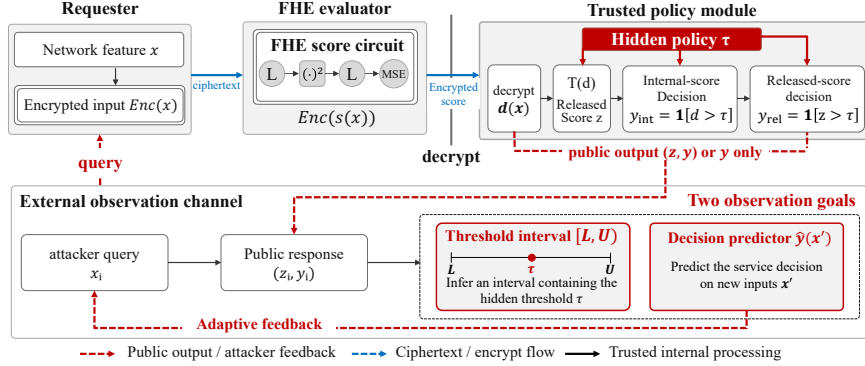


Fig. 1. Service model, decision sources, and two observation goals. Encrypted inputs are evaluated by the FHE scoring circuit, and the trusted policy module decrypts the score, applies the output transformation, and produces decisions using either the internal or released score with the hidden threshold τ . Public responses are analyzed through separate observation records for threshold-interval inference and decision prediction on new inputs.

Figure 1 shows the trust boundary and service flow. The requester encrypts a network feature vector x , and the FHE evaluator computes the scoring circuit on ciphertexts to produce an encrypted score. The score is decrypted only inside the trusted domain to obtain $d(x)$, and the output transformation $T(d)$ produces the released score z . The private threshold τ is used only inside the trusted policy module. An internal-score decision compares $d(x)$ with τ , whereas a released-score decision compares z with τ . Thus, even under the same output transformation, the relation between released scores and decisions can depend on which score is used for the decision.

The external observation analysis separates two protection goals. The first narrows an interval $[L, U]$ that may contain the private threshold by repeatedly

observing public responses. The second uses observed decisions to predict service decisions on new inputs. Because these analyses have different objectives and evaluation criteria, they use separate observation records. A requester holding both the secret key and raw-score ciphertext could decrypt the score before output transformation, so this condition is excluded. This study focuses on how public responses and CKKS numerical errors affect decisions. Malicious ciphertexts, side channels, key recovery, operating-system isolation, and distributed-protocol security require separate attack models or implementation-level validation and are therefore outside the evaluation scope.

3.2 Distinguishing the Scores Used for Decisions

Let $s(x)$ denote the plaintext reference score, $d(x) = s(x) + e(x)$ the decrypted score, and $z = R(d(x); \omega)$ the released score under transformation R with randomness ω . Thus, $d - s$ represents numerical error from encrypted computation, while $z - d$ represents the change introduced by the output transformation. Equation (4) defines the internal-score and released-score decision rules using the private operating threshold τ , where 0 denotes normal and 1 denotes anomalous. Here, $\mathbf{1}[\cdot]$ is the indicator function, which returns 1 when its condition is true and 0 otherwise.

$$b_{\text{int}} = \mathbf{1}[d(x) > \tau], \quad b_{\text{rel}} = \mathbf{1}[z > \tau]. \quad (4)$$

In Eq. (4), b_{int} uses the decrypted score $d(x)$, so the decision for a given decrypted score is preserved even when the release transformation changes. In contrast, b_{rel} uses the transformed score z , so the output transformation can change the decision itself. Quantization of width $h > 0$ is defined by $Q_h(d) = h \text{round}(d/h)$, with half-integer ties rounded away from zero. The noise policy uses $z = d + \eta$, where $\eta \sim \mathcal{N}(0, \sigma^2)$. The decision-only policy returns only the internal-score decision.

For released-score decisions, normal responses satisfy $z_i \leq \tau$, while anomalous responses satisfy $z_i > \tau$. Let b_i denote the observed decision for the i th public response. We define L as the largest released score with $b_i = 0$ and U as the smallest released score with $b_i = 1$. Therefore, the largest normal-response score cannot exceed the threshold, while the smallest anomalous-response score must exceed it, yielding Eq. (5).

$$L = \max_{i: b_i=0} z_i, \quad U = \min_{i: b_i=1} z_i, \quad L \leq \tau < U. \quad (5)$$

In Eq. (5), L and U are the largest normal-response and smallest anomalous-response released scores, respectively (Fig. 2). For released-score decisions, this containment relation holds even when noise is applied because the released score and decision use the same score. It also holds for raw-score release with internal decisions because $z = d$. However, when the released score is transformed while the decision remains based on $d(x)$, the ordering between released scores and decisions may change. In that case, containment of τ in $[L, U)$ is not generally guaranteed.

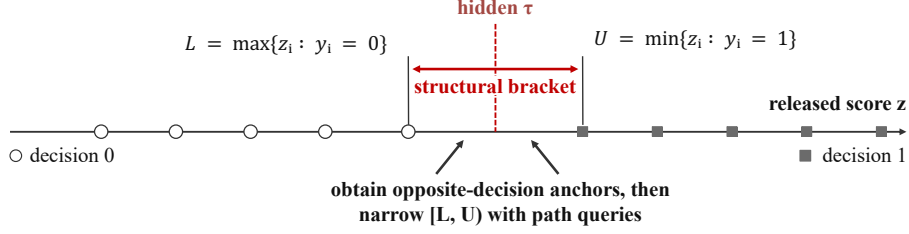


Fig. 2. Threshold interval induced by released scores and decisions. Opposite-decision anchors define the structural bracket $[L, U]$ around the hidden threshold τ , which is narrowed through path queries.

3.3 Threshold Resolution and Decision Prediction

With quantization and released-score decisions, returned scores lie on the grid $h\mathbb{Z}$, where $h\mathbb{Z}$ denotes integer multiples of the quantization width h . If τ and τ' lie in the same interval $[kh, (k+1)h)$, they produce the same decision for every released grid value z . When the model, grid width, and metadata are fixed and no other response depends on the threshold, thresholds in the same interval therefore produce identical public responses. Public records alone cannot distinguish their precise values.

However, utility-constrained selection can make h depend on τ because decision disagreement and added errors for each candidate width are evaluated against threshold-dependent baseline decisions. Changing the threshold can therefore change the selected width and the resulting public responses. This does not contradict fixed-grid response equivalence. Instead, it changes the assumption that the grid remains fixed. We evaluate this policy-reselection sensitivity using the same calibration data and candidate widths.

Boundary search first obtains two inputs x_0 and x_1 with opposite decisions, then repeatedly queries the midpoint of the path $x(\alpha) = (1 - \alpha)x_0 + \alpha x_1$ or a constraint-projected path. Each response updates the endpoints while preserving opposite decisions. Search terminates when the α -interval width is at most 10^{-6} , which treats the input-space boundary as localized to that precision. When both scores and decisions are released, the observations also update L and U . Once a numerical interval is formed, we use $\hat{\tau} = (L + U)/2$ as the threshold estimate because no additional location information is available within the interval and the midpoint provides an unbiased representative of its endpoints.

Threshold estimation error is measured as $|\hat{\tau} - \tau|$, which directly captures the distance between the estimate and the actual threshold. Observation resolution is evaluated using the normalized interval width $(U - L)/a$ and whether the interval contains the actual threshold, accounting for differences in score scale across models. If L and U cannot be formed, $\hat{\tau}$ is unavailable and the estimation error is undefined. Input-path termination means that the location where the decision changes has been narrowed sufficiently in input space. Feature projection can

map nearby inputs to different valid feature values, so a narrow input interval does not necessarily imply a narrow score interval.

Decision prediction measures how well a surrogate trained on public responses reproduces service decisions on separate inputs. We use balanced accuracy (BA) to give equal weight to prediction performance on normal and anomalous service decisions. When both decisions occur, a predictor that always returns one class has BA 0.5. Decision-only policies do not release a numerical score z , so a numerical threshold interval cannot be formed. However, the observed normal/anomalous decisions can still serve as labels for evaluating decision prediction.

3.4 Output Policy Selection under Utility Constraints

The threshold is set from normal validation scores, while output policies are selected on a labeled calibration set D_{cal} . Overall decision agreement alone can hide errors newly introduced within a specific class. We therefore constrain disagreement, added false negatives, and added false positives separately. Let $b_0 = \mathbf{1}[s > \tau]$ denote the plaintext reference decision before applying an output policy, let b denote the decision after applying the policy, and let $y \in \{0, 1\}$ be the ground-truth anomaly label. The losses are defined in Eq. (6).

$$\begin{aligned}\ell_{\text{dis}} &= \mathbf{1}[b \neq b_0], \\ \ell_{\text{FN}} &= \mathbf{1}[b_0 = 1, b = 0] \quad (y = 1), \\ \ell_{\text{FP}} &= \mathbf{1}[b_0 = 0, b = 1] \quad (y = 0).\end{aligned}\tag{6}$$

In Eq. (6), disagreement is evaluated over all calibration rows. Added false negatives and added false positives are evaluated within truly anomalous and truly normal rows, respectively. These losses measure decision changes newly introduced by the output policy rather than the detector’s original errors. This separation prevents high overall agreement from masking class-specific utility loss.

A policy may preserve decisions while substantially changing released scores. We therefore measure score distortion separately using normalized root mean squared error (NRMSE), which accounts for score-scale differences across models [12]. Equation (7) defines this measure.

$$\text{NRMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n \left(\frac{z_i - s_i}{a} \right)^2}.\tag{7}$$

In Eq. (7), n is the number of score pairs and $a = \max\{\text{IQR}_v, \sigma_v, \epsilon_{\text{mach}}\}$. The interquartile range IQR_v and standard deviation σ_v are computed from normal validation scores. Using the larger scale reduces sensitivity to unusually small score dispersion, while ϵ_{mach} prevents unstable normalization near zero. NRMSE is undefined for decision-only output because no released score pairs are available.

The $M = 27$ candidates comprise 24 combinations of two output transformations, two decision rules, and six strengths, together with raw-score, decision-only, and score-only baselines. Score-only output is excluded from final selection because decision preservation cannot be evaluated, but it remains part of the predefined candidate family. To reduce the chance of selecting a candidate that appears favorable only on the calibration sample, we follow the Learn then Test risk-control perspective [1]. Each binary loss is evaluated with a one-sided Clopper–Pearson upper bound [7], and Bonferroni correction accounts for the three losses across all candidates. Equation (8) gives the upper bound for k errors among $n > 0$ samples.

$$u(k, n) = \begin{cases} F_{\text{Beta}(k+1, n-k)}^{-1}(1 - \alpha_*), & k < n, \\ 1, & k = n, \end{cases} \quad \alpha_* = \frac{0.05}{3M}. \quad (8)$$

In Eq. (8), F^{-1} is the beta-distribution quantile function. The familywise significance level 0.05 controls the statistical test, while 0.01 is the allowed upper limit for each binary loss. A candidate is feasible when all three upper bounds are at most 0.01 and empirical NRMSE is at most 0.05. Within each transformation–decision group, we select the largest feasible quantization width or noise standard deviation to impose the strongest transformation that still satisfies the utility constraints. All candidates are checked directly because monotonic loss behavior is not assumed.

Threshold setting, policy selection, and final utility evaluation use separated data so that policy selection does not rely on the same rows used for final evaluation. The Clopper–Pearson bounds apply to the three binary decision losses, whereas NRMSE is retained as a separate empirical utility constraint for continuous score distortion. This design controls policy-induced decision changes and score distortion under distinct criteria.

3.5 CKKS Error and Quantization Boundaries

If CKKS error satisfies $|d - s| \leq \varepsilon$, then $|s - \tau| > \varepsilon$ ensures agreement between the internal-score and plaintext decisions. For round-to-nearest quantization, $|Q_h(d) - s| \leq \varepsilon + h/2$, so $|s - \tau| > \varepsilon + h/2$ is sufficient for agreement with the plaintext raw-score decision. When the same quantization is applied to plaintext and CKKS scores, the distance to the nearest rounding boundary gives the grid-margin condition in Eq. (9).

$$m_h(s) = \min_{k \in \mathbb{Z}} |s - (k + \frac{1}{2})h|, \quad |d - s| < m_h(s) \implies Q_h(d) = Q_h(s). \quad (9)$$

In Eq. (9), $m_h(s)$ is the distance from the plaintext score to the nearest quantization boundary. If CKKS error is smaller than this margin, the quantized output and decision are preserved. Crossing a grid boundary does not necessarily change the decision unless the transformed value also crosses the threshold. We therefore compare measured CKKS errors with threshold and grid margins separately.

4 Experiments and Results

4.1 Evaluation Design and Configuration Rationale

The evaluation addresses three questions. We examine the threshold information remaining in public responses, the utility costs of output transformations and query controls, and the effects of policy selection, data splitting, and CKKS approximation.

The datasets serve different evaluation roles. NSL-KDD [16] reduces the duplicate-connection problem of KDD’99 and provides a baseline setting for score-based evaluation. UNSW-NB15 [11] contains normal traffic and nine attack types, allowing output policies to be compared under a different feature and attack composition. CIC-IDS2017 [14] provides weekday, timestamp, and Flow ID information. These fields support training-only preprocessing, chronological splitting, and removal of recurring Flow IDs. This additional evaluation reduces reliance on a single random-row split, which can affect conclusions in security-oriented machine learning [2]. Because all three datasets were collected in experimental environments, the evaluation scope is limited to these datasets and the selected models.

Table 2 summarizes the common evaluation configuration.

Table 2. Datasets and common evaluation configuration.

Category	Configuration
Datasets	NSL-KDD [16], UNSW-NB15 [11], CIC-IDS2017 [14]
Input	20-D after categorical encoding, Min–Max normalization, and PCA
Models	AE (20–24–12–24–20, square), Centroid, Mahalanobis
Training	30 epochs, Adam, 10^{-3} learning rate, full batch, seeds 0–4
Thresholds	Primary: $\tau = \mu_v + 2\sigma_v$. Alternative: normal-validation 99th percentile
Strengths	h/a , $\sigma/a \in \{0.001, 0.003, 0.01, 0.03, 0.1, 0.3\}$

The primary threshold $\tau = \mu_v + 2\sigma_v$ uses only normal validation scores and reflects both their location and dispersion. It is an operating rule rather than an assumption of normality or a specified false-positive rate. The normal-validation 99th percentile is evaluated as an alternative to test whether the observations depend on a single threshold rule. Policy strengths are normalized by the score scale a and fixed from weak to strong transformations so that the same candidate levels can be compared across scoring models rather than optimized separately for each condition.

4.2 Detection Quality and Baseline Threshold Leakage

Before evaluating output policies, we first examine baseline detection quality and sensitivity to threshold selection. Table 3 reports the baseline quality of the three scoring models, while Table 4 compares TPR and FPR at two AE operating

points. Mahalanobis scoring shows relatively high performance across the three datasets, whereas AE F1 is 0.129 on UNSW-NB15 and 0.117 on CIC-IDS2017. At the 99th-percentile threshold, FPR decreases on these two datasets, but TPR also decreases. Together, the two tables show that changing the AE threshold shifts the trade-off between detection and false positives rather than improving the underlying detection quality.

Table 3. Detection quality of three scoring models across three intrusion-detection datasets, reported as five-seed means using ground-truth anomaly labels.

Dataset	Scoring model	AUROC	AP	F1
NSL-KDD [16]	AE	0.788	0.873	0.634
	Centroid	0.872	0.917	0.741
	Mahalanobis	0.924	0.932	0.819
UNSW-NB15 [11]	AE	0.562	0.652	0.129
	Centroid	0.792	0.853	0.397
	Mahalanobis	0.835	0.879	0.546
CIC-IDS2017 [14]	AE	0.796	0.426	0.117
	Centroid	0.841	0.455	0.150
	Mahalanobis	0.923	0.772	0.524

AUROC: area under the receiver operating characteristic curve. AP: average precision. F1: harmonic mean of precision and recall.

Table 4. AE detection performance before output policies under two threshold rules, reported as five-seed medians using ground-truth anomaly labels.

Dataset	$\mu_v + 2\sigma_v$: TPR / FPR	99th percentile: TPR / FPR
NSL-KDD [16]	0.418 / 0.020	0.055 / 0.002
UNSW-NB15 [11]	0.068 / 0.033	0.040 / 0.008
CIC-IDS2017 [14]	0.075 / 0.041	0.009 / 0.010

TPR: true-positive rate. FPR: false-positive rate.

Table 5 evaluates how closely the private threshold can be estimated from released scores and decisions under these baseline conditions. All 45 dataset-model conditions met the 10^{-6} path-width termination criterion, with mean absolute errors ranging from 8.27×10^{-8} to 2.71×10^{-5} . These results show that repeated public responses can narrow the numerical location of the threshold even when baseline detection quality differs across models. Because score scales differ by model, absolute errors are not used to directly compare estimation difficulty.

Table 5 also reports the observation cost of threshold estimation. Total query counts average 126.2–150.8 and include 101 path-diagnostic queries. A separate

Table 5. Plaintext threshold estimation error and query cost across three datasets and scoring models, reported as five-condition means for runs meeting the termination criterion.

Dataset	Scoring model	Mean absolute error	Total queries	Estimation queries
NSL-KDD [16]	AE	9.82×10^{-8}	126.6	25.6
	Centroid	9.18×10^{-8}	126.6	25.6
	Mahalanobis	3.81×10^{-6}	126.2	25.2
UNSW-NB15 [11]	AE	1.13×10^{-7}	150.8	49.8
	Centroid	2.07×10^{-7}	137.4	36.4
	Mahalanobis	2.71×10^{-5}	130.2	29.2
CIC-IDS2017 [14]	AE	8.27×10^{-8}	140.0	39.0
	Centroid	9.97×10^{-8}	140.0	39.0
	Mahalanobis	8.19×10^{-6}	134.2	33.2

Total queries: estimation plus 101 diagnostic queries.

Estimation queries: diagnostics excluded. Error: model-specific score units.

estimation run without diagnostics uses 25.2–49.8 queries on average. Thus, the total count represents evaluation cost including diagnostics rather than a minimum query requirement. To test whether the estimated threshold preserves decisions with the scoring function fixed, we replaced only the threshold on a separate candidate set. Decision agreement was 1.0 in every condition, showing that the estimated thresholds reproduced the original decisions on that candidate set.

4.3 Service Costs of Query Controls and Fixed Noise

This experiment compares how query controls suppress threshold observation at the cost of response availability and how fixed output noise affects preservation of existing decisions. Query caps, similarity-based rejection, and fixed noise are evaluated under the same boundary-search setting across 45 plaintext conditions. Similarity rejection uses the L_2 distance between a new input and up to 512 recently accepted inputs and rejects requests within a specified radius. Because this mechanism directly uses feature vectors, it is evaluated as a separate plaintext reference condition from the scalar output module in Fig. 1. Response rate is measured on 512 usage requests separate from the estimation sessions, and decision agreement is computed over returned responses.

A 16-query cap stopped all 45 baseline searches but returned only 16 of 512 usage responses, corresponding to 3.125%. With a 128-query cap, the response rate increased to 25%, but all 45 searches also completed. These results show that a low query cap can suppress repeated observation while sharply reducing ordinary response availability. Relaxing the cap improves availability but can also restore the ability to complete boundary search.

Table 6. Search completion, response rate, and agreement under query controls and fixed noise. Completion indicates meeting the termination criterion. Response rate is measured on 512 separate usage requests. Agreement is computed over returned responses only.

Defense setting	Baseline completion	Adaptive completion	Response rate (%)	Agreement (%)
No defense	45/45	—	100.00	100.00
Fixed noise $\sigma = 0.1$	27/45	—	100.00	88.83
16-query cap	0/45	—	3.125	100.00
128-query cap	45/45	—	25.00	100.00
Similarity $\delta = 10^{-8}$	0/45	45/45	99.22	100.00
Similarity $\delta = 10^{-5}$	0/45	45/45	98.44	100.00
Similarity $\delta = 10^{-3}$	0/45	45/45	96.94	100.00

—: excluded comparison.

Similarity-based rejection stopped all baseline searches while maintaining response rates of 96.94–99.22%. However, when the rejection radius was known and queries were adapted to rejected responses, all 135 experiments across the three radii formed intervals containing the actual threshold. Mean attempts increased from 35.67 to 45.13 as the radius increased. The baseline search can also terminate when a request that rechecks a starting input is rejected. The results therefore reflect both the rejection mechanism and how the search uses radius information and rejected responses. Search termination alone does not characterize the information remaining in accepted responses.

Under fixed noise, the response rate remained 100%, but only 27 of 45 baseline searches completed and decision agreement fell to 88.83%. Thus, preserving response availability does not ensure preservation of the service’s original decisions when released scores are perturbed. Here, $\sigma = 0.1$ is fixed in raw-score units and is distinct from the model-scaled noise selected under utility constraints in the following subsection.

4.4 Output Policies and Decision Observation under Utility Constraints

This experiment evaluates three questions in sequence: how well utility-constrained output policies preserve existing decisions, how much numerical threshold information remains, and whether limiting that information also reduces service decision prediction. Test rows are randomly divided into policy calibration, utility evaluation, observation candidates, and final decision evaluation, with approximately one quarter assigned to each. Policies are selected on the calibration rows and then fixed for subsequent evaluation. The six groups include raw-score and decision-only outputs, together with quantization and

noise using either internal-score or released-score decisions. PCA-space and constraint-projected paths are evaluated for each group, yielding 540 sessions in total. Constraint projection enforces valid ranges, integer features, and one-hot representations, but does not establish packet realizability. Each session uses a 160-query cap and a path-width termination criterion of 10^{-6} .

Table 7 reports decision preservation and added errors for policies selected under the utility constraints at the primary operating point $\tau = \mu_v + 2\sigma_v$. Released-score quantization preserved 99.864% of decisions on average, while released-score noise preserved 99.823%. All three added losses remained within 1% in every condition. These results show that released scores can be transformed while retaining most of the service’s existing decisions. Repeating the noise policy over 200 randomness trials produced a similar mean agreement of 99.822%. However, at the 99th-percentile operating point, released-score noise was feasible in only 44 of 45 conditions. The utility of the noise policy was therefore stable across repeated randomness but its feasibility still depended on the operating threshold.

Table 7. Held-out evaluation of policies selected under utility constraints, reported as means over 45 conditions. All evaluated policies have a 100% response rate.

Output	Decision	Agreement (%)	Added false negatives (%)	Added false positives (%)	NRMSE
Raw score	Internal	100.000	0.000	0.000	0.0000
Decision only	Internal	100.000	0.000	0.000	—
Quantization	Internal	100.000	0.000	0.000	0.0293
	Released	99.864	0.099	0.031	0.0265
Noise	Internal	100.000	0.000	0.000	0.0299
	Released	99.823	0.151	0.036	0.0264

Added FN/FP: policy-induced errors within each ground-truth class. —: score distortion not applicable.

Table 8 compares the numerical threshold information that remains under these utility-preserving policies and the observation cost required to obtain it. On constraint-projected paths, raw-score release had a median normalized interval width of 8.68×10^{-6} , whereas released-score quantization increased it to 0.1. Their mean query counts were similar at 31.40 and 31.22. Quantization therefore reduced numerical threshold resolution without substantially increasing observation cost. Released-score noise produced a median width of 0.00152 but required 133.78 queries on average. Under internal-score decisions, the ordering of released scores and decisions could also become inconsistent. Threshold-interval interpretation must therefore consider both the output transformation and the score used for the decision.

Table 8. Termination, public-score intervals, and observation cost on constraint-projected paths across 45 conditions. All sessions use a 160-query cap, and noise settings include additional observations. Interval widths are omitted for internal-score decisions because threshold containment is not guaranteed.

Output	Decision	Path termination	$L < U$	Median width/ a	Mean queries
Raw score	Internal	45/45	45/45	8.68×10^{-6}	31.40
Decision only	Internal	45/45	N/A	—	33.18
Quantization	Internal	45/45	10/45	—	28.69
	Released	45/45	45/45	0.100	31.22
Noise	Internal	45/45	8/45	—	136.20
	Released	44/45	45/45	0.00152	133.78

$L < U$: conditions with an ordered public-score interval.

N/A: numerical interval not applicable. —: width not reported.

To determine whether reduced numerical information also limits service decision prediction, we use separate observation-candidate and final-evaluation sets. The surrogate model is trained only on 20-dimensional PCA inputs and returned binary decisions. Released scores and ground-truth anomaly labels are not used for surrogate training. Figure 3 shows decision-prediction performance as the observation budget increases. With 128 observations, decision-only output achieved mean BA 0.817, while released-score quantization achieved 0.816. Thus, numerical threshold information could be restricted, or numerical scores omitted entirely, while service decisions remained predictable from observed decision responses.

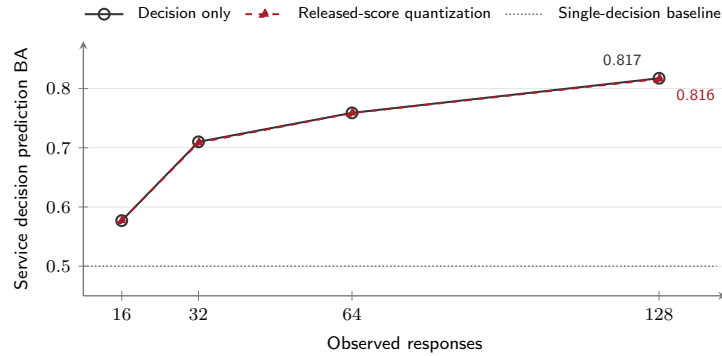


Fig. 3. Decision prediction versus observation count under random row splitting. Values are means over 45 experiments, with lines connecting measured points. Targets are service decisions, not ground-truth anomaly labels.

Finally, we evaluate whether fixed-grid response equivalence remains valid when the output policy is reselected as the threshold changes. For each of 45 conditions, we use the original threshold and five alternative thresholds at relative positions 0.05, 0.25, 0.5, 0.75, and 0.95 within the selected quantization cell, giving $45 \times 6 = 270$ settings. With the quantization width fixed, all 1,000 held-out scores not used for policy selection produced identical responses for thresholds within the same cell. We then reselected the policy for each threshold using the same calibration data and 27 candidates. The selected width changed in 35 settings, and the resulting quantization grid changed released-score positions enough to alter decisions in 32 settings. At least one width change occurred in 21 of 45 conditions, with a maximum decision-change rate of 8.6%. These results show that response equivalence under a fixed width does not automatically extend to a threshold-dependent policy-selection procedure.

4.5 Effects of Data Splitting and Temporal Change

Additional evaluations examine whether the decision-prediction performance under random-row splitting depends on duplicate rows or the split itself. Removing held-out rows identical to observed rows changed BA only slightly, to 0.817052. Separating model-training and threshold-setting data from later evaluation data still yielded decision-only BA of 0.816–0.911. Thus, the earlier prediction performance cannot be explained solely by exact duplicates or a simple random split.

Under a stricter setting, data are ordered by collection time and flows already observed in earlier intervals are removed when they reappear in later evaluation intervals. Decision agreement remains high at 99.35–99.63%, but decision-only BA decreases to 0.500–0.501. Table 9 compares decision preservation and prediction performance under these split conditions. Because temporal ordering, flow-overlap removal, and evaluation-interval composition change together, the BA decrease cannot be attributed to a single factor. The results nevertheless show that service decision prediction is sensitive to how observation and later evaluation data are separated. Prediction performance from random-row splits should therefore not be directly generalized to deployment conditions.

Table 9. CIC-IDS2017 split sensitivity for decision preservation and prediction. Week-day results use seed 0, while Time/Flow ID results are means over five resampling seeds.

Model	Weekday		Time/Flow ID		Path termination
	Agreement	Decision BA	Agreement	Decision BA	
AE	99.800	0.911	99.626	0.500	1/5
Centroid	99.987	0.816	99.627	0.500	1/5
Mahalanobis	99.867	0.853	99.354	0.501	4/5

Weekday: temporal split. Time/Flow ID: recurring flows removed. Agreement: preservation (%). BA: 128-response balanced accuracy.

4.6 Threshold Estimation and Released-Decision Stability in CKKS

The CKKS evaluation examines how approximation error affects threshold estimation and released decisions. Encrypted evaluation uses the square-activation AE scoring circuit implemented with TenSEAL 0.3.16 [3]. We set the polynomial degree to $N = 32768$, the scale to 2^{40} , and the coefficient-modulus bit sizes to $(60, 40, 40, 40, 40, 40, 40, 60)$.

Table 10 reports conditional threshold-estimation error and request latency when search continues from supplied intervals. For each of the three intrusion-detection datasets, five AE seeds are combined with three independent keys, yielding 15 conditions per dataset and 45 conditions in total. Under each new key, the supplied interval is first revalidated for containment of the actual threshold before search continues. All final intervals contained τ , with path width 9.54×10^{-7} and dataset-wise mean errors of 1.04×10^{-7} – 2.42×10^{-7} . Synthetic data are evaluated separately in 15 additional conditions to examine numerical behavior without mixing them with the intrusion-detection results. These results therefore characterize conditional estimation from supplied intervals rather than complete same-key sessions beginning with anchor acquisition.

Table 10. Conditional CKKS estimation error and request latency from supplied intervals. Latencies are means over 45 fresh-encryption requests.

Dataset	Completed/ contained	Mean absolute error	Latency (s)		
			Encryption	HE eval.	Total
Synthetic	15/15	2.82×10^{-7}	0.048	0.591	0.833
NSL-KDD [16]	15/15	2.42×10^{-7}	0.155	4.092	4.904
UNSW-NB15 [11]	15/15	1.04×10^{-7}	0.155	4.098	4.910
CIC-IDS2017 [14]	15/15	1.08×10^{-7}	0.155	4.149	4.956

Completed/contained: termination and threshold containment both satisfied.
HE: homomorphic encryption.

To evaluate numerical stability of released decisions, we fix the trained AE model and the quantization policy selected from the calibration data in 15 conditions formed by three datasets and five AE seeds. Two independent keys, four inputs, and two fresh encryptions per condition yield 240 ordinary-input requests. Internal-score decisions and released-score decisions under the same quantization matched their plaintext counterparts in all requests, with a maximum decryption error of 4.734×10^{-6} . Public responses are restricted to the scores and decisions permitted by the policy so that the evaluation isolates the effect of CKKS numerical error on released results.

For 120 near-threshold requests, no internal-decision mismatch occurred, and the minimum threshold-margin-to-error ratio was 1.611. In three representative conditions selected near quantization boundaries, one released-decision mismatch occurred among 57 requests. A separate NSL-KDD condition acquired starting points from public responses and retained the same key, producing one mis-

match among 23 requests. Both boundary evaluations terminated at path width 7.63×10^{-6} because they were designed to test decision stability near numerical boundaries rather than precise threshold estimation. The observed 1/57 and 1/23 mismatches therefore demonstrate that disagreement can occur in selected boundary cases and are not interpreted as operational error rates. Table 11 summarizes these numerical validation settings.

Table 11. Distinct numerical validation settings for actual CKKS. Rows are separate evaluations, and request counts are not pooled.

Evaluation input or path	Requests	Mismatches	
		Internal	Released
Ordinary inputs	240	0 / 240	0 / 240
Near-threshold inputs	120	0 / 120	—
Boundary paths (3 conditions)	57	—	1 / 57
Public-response start, same key (1 condition)	23	—	1 / 23

Internal: compared with plaintext raw-score decisions.

Released: compared under the same deterministic quantization. —: not reported.

Figure 4 compares CKKS error with the quantization-boundary margin in the two mismatch cases. In the 57-request case, the margin was 4.674×10^{-7} and the error was 9.609×10^{-7} . In the public-response-started case, they were 6.820×10^{-8} and 2.343×10^{-6} , respectively. In both cases, CKKS error exceeded the remaining margin to the quantization boundary and a released-decision mismatch was observed. The results show that even a small absolute error can alter a released result when the score lies closer to a quantization boundary than the error magnitude. Numerical stability should therefore be assessed relative to boundary margins rather than by average error alone.

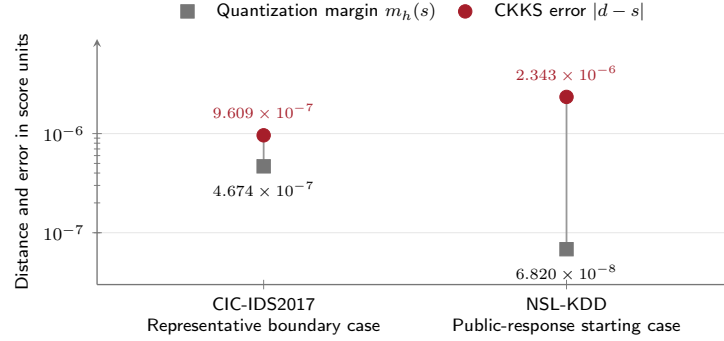


Fig. 4. Numerical relationships in two CKKS cases with released-decision mismatches. In both cases, decryption error exceeded the quantization-boundary margin. Points denote selected distances, not error frequencies or operational probabilities.

Runtime measurements examine which stage dominates the cost of processing CKKS requests. Experiments run on a shared Linux host with an Intel Core i7-10875H, approximately 15.4 GiB of memory, and Python 3.12.3. Across 240 ordinary-input requests, the median latencies for encryption and serialization, evaluation and serialization, and decryption and deserialization were 1.235 s, 12.491 s, and 0.002 s, respectively. The median total latency was 13.728 s. Most request cost therefore came from encrypted evaluation rather than output transformation, while the public evaluation context in the representative boundary evaluation was approximately 850 MiB. Key and context generation, networking, and authentication are excluded from these measurements, so deployment cost must also account for encrypted computation and context transfer.

5 Conclusion

This study separated numerical threshold information from service decision prediction in homomorphically encrypted anomaly detection. Distinguishing internal-score and released-score decisions clarified when public responses define a threshold-containing interval. Plaintext and conditional CKKS evaluations showed that input encryption does not remove operational information exposed through released scores and decisions. Fixed-grid equivalence can limit precise numerical identification of the threshold, but it does not imply that service decisions on other inputs are difficult to predict.

Released-score quantization preserved 99.864% of existing decisions on average under random-row splitting, while KNN trained only on decision responses achieved mean BA 0.817. These results show that service utility can remain high even when decision functionality remains reproducible from public responses. Output policies should therefore evaluate numerical threshold protection and decision-function protection separately, while also considering policy-induced errors and the response rate of query controls. Policy reselection and alternative data splits further showed that these results depend on evaluation and deployment conditions. The CKKS experiments also showed that decision stability depends not only on the absolute approximation error but on its size relative to threshold and quantization-boundary margins. Output-policy protection should therefore be assessed jointly through decision preservation, response rate, prediction transfer, and numerical boundary conditions rather than by reduced score precision alone.

The present evaluation is limited by its use of the specified datasets and scoring models, feature-space inputs, supplied starting intervals for conditional CKKS estimation, and selected boundary cases. In particular, the selected CKKS boundary cases are insufficient to estimate general operational error probabilities. Future work should extend evaluation under broader temporal and flow-separated conditions, acquire starting inputs from public responses throughout complete CKKS sessions, and further examine numerical stability near decision boundaries under diverse deployment conditions.

References

1. Angelopoulos, A.N., Bates, S., Candès, E.J., Jordan, M.I., Lei, L.: Learn then test: Calibrating predictive algorithms to achieve risk control. *The Annals of Applied Statistics* **19**(2), 1641–1662 (2025). <https://doi.org/10.1214/24-A0AS1998>
2. Arp, D., Quiring, E., Pendlebury, F., Warnecke, A., Pierazzi, F., Wressnegger, C., Cavallaro, L., Rieck, K.: Dos and don'ts of machine learning in computer security. In: 31st USENIX Security Symposium (USENIX Security 22). pp. 3971–3988. USENIX Association (2022), <https://www.usenix.org/conference/usenixsecurity22/presentation/arp>
3. Benaïssa, A., Retiat, B., Cebere, B., Belfedhal, A.E.: TenSEAL: A library for encrypted tensor operations using homomorphic encryption (2021). <https://doi.org/10.48550/arXiv.2104.03152>, arXiv preprint
4. Byun, J., Choi, Y., Lee, J., Park, S.: Privacy-preserving inference resistant to model extraction attacks. *Expert Systems with Applications* **256** (2024). <https://doi.org/10.1016/j.eswa.2024.124830>, article 124830
5. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. *ACM Computing Surveys* **41**(3), 1–58 (2009). <https://doi.org/10.1145/1541880.1541882>
6. Cheon, J.H., Kim, A., Kim, M., Song, Y.: Homomorphic encryption for arithmetic of approximate numbers. In: *Advances in Cryptology – ASIACRYPT 2017. Lecture Notes in Computer Science*, vol. 10624, pp. 409–437. Springer (2017). https://doi.org/10.1007/978-3-319-70694-8_15
7. Clopper, C.J., Pearson, E.S.: The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika* **26**(4), 404–413 (1934). <https://doi.org/10.1093/biomet/26.4.404>
8. Kuppa, A., Grzonkowski, S., Asghar, M.R., Le-Khac, N.A.: Black box attacks on deep anomaly detectors. In: *Proceedings of the 14th International Conference on Availability, Reliability and Security*. pp. 1–10. ACM (2019). <https://doi.org/10.1145/3339252.3339266>, article 21
9. Li, H., Shan, S., Wenger, E., Zhang, J., Zheng, H., Zhao, B.Y.: Blacklight: Scalable defense for neural networks against query-based black-box attacks. In: 31st USENIX Security Symposium (USENIX Security 22). pp. 2117–2134. USENIX Association (2022), <https://www.usenix.org/conference/usenixsecurity22/presentation/li-huiying>
10. Mahalanobis, P.C.: On the generalized distance in statistics. *Proceedings of the National Institute of Sciences of India* **2**(1), 49–55 (1936)
11. Moustafa, N., Slay, J.: UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: *2015 Military Communications and Information Systems Conference*. pp. 1–6. IEEE (2015). <https://doi.org/10.1109/MilCIS.2015.7348942>
12. Oh, S., Kang, D.D., Brock, G.N., Tseng, G.C.: Biological impact of missing-value imputation on downstream analyses of gene expression profiles. *Bioinformatics* **27**(1), 78–86 (2011). <https://doi.org/10.1093/bioinformatics/btq613>
13. Sakurada, M., Yairi, T.: Anomaly detection using autoencoders with nonlinear dimensionality reduction. In: *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*. pp. 4–11. ACM (2014). <https://doi.org/10.1145/2689746.2689747>
14. Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: *Proceedings of the 4th International Conference on Information Systems Security and Privacy*. pp. 108–116. SciTePress (2018). <https://doi.org/10.5220/0006639801080116>

15. Tang, M., Dai, A., DiValentin, L., Ding, A., Hass, A., Gong, N.Z., Chen, Y., Li, H.H.: ModelGuard: Information-theoretic defense against model extraction attacks. In: 33rd USENIX Security Symposium (USENIX Security 24). pp. 5305–5322. USENIX Association (2024), <https://www.usenix.org/conference/usenixsecurity24/presentation/tang>
16. Tavallae, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the KDD CUP 99 data set. In: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications. pp. 1–6. IEEE (2009). <https://doi.org/10.1109/CISDA.2009.5356528>
17. Tramèr, F., Zhang, F., Juels, A., Reiter, M.K., Ristenpart, T.: Stealing machine learning models via prediction APIs. In: 25th USENIX Security Symposium (USENIX Security 16). pp. 601–618. USENIX Association (2016)
18. Trivedi, D., Boudguiga, A., Triandopoulos, N.: SigML: Supervised log anomaly with fully homomorphic encryption. In: Cyber Security, Cryptology, and Machine Learning. Lecture Notes in Computer Science, vol. 13914, pp. 372–388. Springer (2023). https://doi.org/10.1007/978-3-031-34671-2_26