

Evaluating the Robustness of Adversarial Defenses in Malware Detection Systems

Mostafa Jafari and Alireza Shameli-Sendi

Faculty of Computer Science and Engineering, Shahid Beheshti University (SBU), Tehran, Iran

Email: {most.jafari@mail.sbu.ac.ir,a_shameli@sbu.ac.ir}

Abstract—Machine learning (ML) has become a vital tool in Android malware detection, effectively identifying malicious patterns in applications to combat the growing threat of malware. However, ML-based systems remain highly vulnerable to evasion attacks, where subtle perturbations can bypass detection mechanisms. Despite efforts to develop adversarially robust malware detection systems, the lack of comprehensive evaluation frameworks for state-of-the-art defenses in this domain limits the thorough understanding of their effectiveness. This study introduces two key contributions to advance adversarial evaluation in binary-constrained domains. First, we propose *Prioritized Binary Rounding*, a technique that efficiently transforms continuous adversarial perturbations into binary spaces while preserving high attack success rates and minimizing perturbation size. Second, we develop the σ -binary attack, a novel adversarial attack specifically tailored for binary domains, designed to achieve adversarial objectives with minimal perturbations while maintaining compatibility with binary feature constraints. Extensive evaluations on the Malscan dataset demonstrate the superiority of the proposed approach. σ -binary consistently outperforms existing adversarial methods, including CW, σ -zero, Mimicry, and PGD-based attacks, across both robust and non-robust defenses in terms of attack success rate and perturbation size. Furthermore, our analysis of state-of-the-art defenses using σ -binary reveals critical vulnerabilities. Defenses equipped with adversary detectors, such as KDE, DLA, DNN⁺, and ICNN, exhibit significant brittleness, with attack success rates exceeding 90% for fewer than 10 feature modifications and reaching 100% with a 20-feature perturbation budget. Adversarially trained defenses, including AT-rFGSM^k and AT-MaxMA, exhibit improved robustness at lower budgets but remain highly susceptible to unrestricted perturbations, with attack success rates of 99.45% and 96.62%, respectively. Although PAD-SMA has been demonstrated to maintain a high accuracy above 83.45% (equivalent to an attack success rate below 16.55%) against 15 gradient-based attacks without perturbation limits, our analysis reveals significant vulnerabilities. The σ -binary attack achieves a 36.34% success rate against PAD-SMA with fewer than 10 modified features, increasing to 94.56% under unrestricted perturbations. These findings underscore the importance of precise evaluation tools, such as σ -binary, to uncover latent weaknesses in existing defenses and guide the development of more robust malware detection systems.

Index Terms—Machine Learning, Android Malware Detection, Adversarial Malware Detection, Evasion Attack

I. INTRODUCTION

The surge in mobile device attacks has escalated into a critical global concern. In 2023 alone, such attacks increased by 50% compared to the previous year. According to Kaspersky, their systems blocked 33.8 million instances of malware, adware, and riskware targeting mobile devices, with Android

remaining the primary focus of cybercriminals [1]. This significant increase in Android malware underscores the urgent need for advanced detection strategies to counter increasingly sophisticated threats.

Machine learning (ML) has become a cornerstone of Android malware detection, significantly improving scalability and detection precision [2]–[6]. Among ML-based approaches, static analysis methods, such as Drebin [4] and MaMaDroid [7], have demonstrated notable efficiency. These methods analyze application features without requiring execution, enabling rapid and scalable security assessments. Consequently, static analysis has become a foundational component of modern malware detection pipelines.

Despite these advancements, ML-based malware detectors are inherently vulnerable to adversarial attacks. These attacks exploit weaknesses in detection models by introducing carefully crafted perturbations to evade detection. Such attacks, which undermine reliability, are broadly categorized into evasion and poisoning attacks. Evasion attacks manipulate test examples during the deployment phase to bypass detection [8]–[12], whereas poisoning attacks compromise training datasets to degrade detector performance [13], [14]. This study focuses on evasion attacks because of their immediate implications for operational systems.

Evasion attacks can be further divided into problem-space and feature-space attacks. Problem-space attacks modify malware applications directly, such as injecting benign "gadgets" or altering application components to obscure malicious intent [15]–[17]. In contrast, feature-space attacks modify feature vectors extracted from malware while preserving the functionality of the original sample [8], [9]. This study focuses exclusively on feature-space attacks because of their controlled evaluation capabilities and practical relevance.

Defending against adversarial attacks remains a significant challenge. Adversarial training, which incorporates adversarial samples into training datasets, has shown promise in improving robustness [8]. However, these methods often fail to generalize to novel attack strategies, impose high computational costs, and risk overfitting to specific attacks [18]. Other defense mechanisms such as ensemble methods [19]–[21], auxiliary models such as Variational Autoencoders (VAEs) [22], and hybrid frameworks such as PAD-SMA [23] offer varying degrees of effectiveness. However, these approaches frequently involve trade-offs, including increased complexity and reduced detection accuracy.

A. Our Contributions

Evaluating the adversarial robustness of malware defenses in binary space presents significant challenges. Existing approaches encounter two critical limitations. First, gradient-based attacks originally designed for continuous spaces are often adapted to binary domains using naive binarization techniques such as Deterministic and Randomized Binary Rounding, which significantly reduce their effectiveness (e.g., [9]). Additionally, the absence of a gradient-based attack designed for binary spaces arises from the inherent complexity of optimization within a non-convex and non-differentiable constraint space, further exacerbating this limitation. Second, these methods rely on subjective methodologies and generalized attack strategies, employing low-iteration gradient-based attacks with fixed hyperparameters across all defenses without optimizing them to individual models (e.g., [20], [22]–[25]). This combination of attack incompatibility with binary-constrained domains and insufficient optimization prevents these methods from effectively exposing vulnerabilities in defense mechanisms.

To address the challenges of evaluating adversarial robustness in binary-constrained domains, this study introduces a novel framework tailored for thorough assessment of defenses. The key contributions of our research are as follows:

First, we propose *Prioritized Binary Rounding*, an innovative technique for transforming gradient-based attacks from continuous to binary spaces. Extensive evaluations on the Malscan dataset [26], encompassing four defenses (DNN, AT-rFGSM, AT-MaxMA, and PAD-SMA) and two attack scenarios (σ -zero and CW attacks, optimizing ℓ_0 -norm and ℓ_2 -norm, respectively), demonstrate its superiority over Deterministic and Randomized Binary Rounding in terms of both attack success rate and perturbation size IV-B. For the σ -zero attack, Prioritized Binary Rounding achieves results nearly identical to continuous outputs, minimizing feature modifications while maintaining high attack success rates: 100% on DNN (compared with 94.88% for the second-best method), 85.95% on AT-rFGSM (vs. 78.64%), 92.56% on AT-MaxMA (vs. 81.31%), and 79.64% on PAD-SMA (vs. 69.91%). For the CW attack, it effectively addresses the challenge of binarizing small fractional perturbations, achieving 100% success on DNN (vs. 49.45%), 68.36% on AT-rFGSM (vs. 26.10%), 65.37% on AT-MaxMA (vs. 37.18%), and 36.82% on PAD-SMA (vs. 8.25%).

Second, we introduce *σ -binary attack*, a novel adversarial method specifically designed for binary-constrained domains. This attack leverages a differentiable approximation of the Hamming distance, which is a key metric for binary vectors, to facilitate an effective gradient-based optimization. Furthermore, it incorporates dynamic confidence adjustment, enabling the optimization process to efficiently explore regions near the decision boundary and enhance the likelihood of identifying viable binary solutions. By seamlessly integrating Prioritized Binary Rounding, σ -binary maintains strict adherence to binary feature constraints while effectively achieving adversarial objectives.

Our experimental evaluations demonstrate the superior per-

formance of σ -binary IV-C. Against the non-robust baseline (DNN), σ -binary achieves an optimal attack success rate of 100%, consistent with most other attacks except Mimicry. On AT-rFGSM^k, it attains 99.45%, surpassing PGD- ℓ_2 (89.21%) and σ -zero (85.95%). Against AT-MaxMA, σ -binary achieves 96.62%, outperforming σ -zero (92.56%) and PGD- ℓ_2 (91.75%). For PAD-SMA, the most robust defense, σ -binary attains 94.56%, significantly exceeding PGD- ℓ_2 (79.90%) and σ -zero (79.64%).

Further analysis of attack success rates across varying perturbation budgets confirms that σ -binary consistently outperforms all other attacks across all evaluated defenses (Figure 2), underscoring its efficacy in binary-constrained domains.

Finally, we apply the σ -binary attack to evaluate the robustness of the eight defenses on the Malscan dataset IV-C. The baseline DNN and defenses with adversary detectors, such as KDE, DLA, DNN⁺, and ICNN, exhibit significant vulnerability, with attack success rates exceeding 90% for fewer than 10 feature modifications and reaching 100% with a 20-feature perturbation budget. In contrast, adversarially trained defenses such as AT-rFGSM^k and AT-MaxMA show improved robustness at lower budgets, but remain susceptible under unrestricted perturbations, with success rates of 99.45% and 96.62%, respectively. PAD-SMA, considered the most robust defense, demonstrates strong resilience in prior evaluations [23], achieving an accuracy above 83.45% (equivalent to an attack success rate below 16.55%) against 15 different gradient-based attacks without perturbation limits. However, our evaluation reveals vulnerabilities, with σ -binary achieving 36.34% success with fewer than 10 modified features, 89.79% at a 50-feature budget, and 94.56% under unrestricted perturbations.

Furthermore, we evaluate defenses under both non-attack conditions and oblivious attacks to provide a comprehensive comparison. Our findings indicate that while PAD-SMA exhibits marginally greater robustness under high perturbation levels compared to the second most robust defense (AT-MaxMA), this comes at the cost of significant computational overhead and reduced accuracy for benign samples. These findings emphasize the trade-offs among various evaluation criteria and underscore the necessity to align defenses with specific threat models and operational requirements.

The source code for this study is publicly available at <https://github.com/mostafa-ja/sigma-binary> to ensure reproducibility and facilitate further research.

B. Paper outline.

The remainder of the paper is organized as follows: Section II provides a review of the preliminaries. Section III details the proposed methods, including Prioritized Binary Rounding and σ -binary. Section IV describes the experimental setup and presents results. Section V reviews related studies on evasion attacks and defenses. Finally, Section VI presents the key findings and outlines directions for future research.

II. PRELIMINARIES

In this section, we introduce the notations used throughout the paper and describe the integrated malware and adversary

detectors, focusing on the key concepts that form the basis of our work.

A. Notations

To formalize the concepts and methods discussed in this paper, we define the following key notations:

- Let Z represent the problem space, where each $z \in Z$ corresponds to an Android application sample.
- A feature extraction function $\phi : Z \rightarrow X$ maps samples from the problem space to a d -dimensional discrete feature space $X \subset \mathbb{R}^d$.
- A malware detector $f : Z \rightarrow Y$ assigns a label $y \in Y = \{0, 1\}$, where 0 represents benign and 1 indicates malicious. The detector is defined as $f(z) = \varphi_\theta(\phi(z))$, where $\varphi_\theta : X \rightarrow Y$ is a ML model with parameters θ .
- An adversary detector $g : Z \rightarrow \mathbb{R}$ identifies adversarial examples. It operates in the feature space as $g(z) = \psi_\vartheta(\phi(z))$, where ψ_ϑ is a secondary model with learnable parameters ϑ . A sample is flagged as adversarial if $g(z) > \tau$, where τ is a predefined threshold.
- For a sample-label pair (z, y) , the feature representation is $x = \phi(z)$. An adversarial example is defined as $z' = z + \delta_z$, where δ_z denotes perturbations in the problem space. Correspondingly, $x' = \phi(z')$ represents the perturbed feature-space sample, expressed as $x' = x + \delta_x$ where δ_x denotes feature-space perturbations.

The loss function for the malware detector is denoted as $\mathcal{L}_\varphi(\theta, x + \delta_x, y)$, capturing the discrepancy between the model's predictions and true labels.

B. Integrated Malware and Adversary Detectors

To enhance the robustness of malware detection, a secondary adversary detector g is integrated with the primary malware detector f . The models are defined as:

$$f(z) = \varphi_\theta(\phi(z)), \quad (1)$$

$$g(z) = \psi_\vartheta(\phi(z)), \quad (2)$$

where θ and ϑ are the parameter sets of the respective models. The prediction process is as follows:

$$\text{predict}(z) = \begin{cases} f(z), & \text{if } g(z) \leq \tau, \\ 1, & \text{if } g(z) > \tau \text{ and } f(z) = 1, \\ \text{further action}, & \text{if } g(z) > \tau \text{ and } f(z) = 0. \end{cases} \quad (3)$$

When $g(z) > \tau$ and $f(z) = 0$, the sample is classified as suspicious and handled in one of two ways: (i) **Deferred**, where the input is excluded from classification and reserved for further analysis; or (ii) **Conservative**, where the input is labeled as malicious to prioritize safety. This integrated approach ensures robust detection while addressing uncertainties introduced by adversarial examples.

C. Evasion Attacks

Evasion attacks target both the problem and feature spaces. In the problem space, an adversarial sample z' satisfies:

$$f(z') = 0, \quad g(z') \leq \tau. \quad (4)$$

In the feature space, this translates to perturbations $x' = x + \delta_x$ such that:

$$\varphi_\theta(x') = 0, \quad \psi_\vartheta(x') \leq \tau, \quad x' \in [\underline{u}, \bar{u}], \quad (5)$$

The interval $[\underline{u}, \bar{u}]$ defines the feature space boundaries, where \underline{u} and \bar{u} represent the lower and upper limits, respectively. These constraints ensure that perturbations remain within the valid feature range, maintaining the feasibility of adversarial examples.

To address the disconnect between the feature space and the problem space, an approximate inverse mapping function, $\tilde{\phi}^{-1}$, is utilized [27]. This function maps perturbations from the feature space back to the problem space, ensuring practical feasibility.

D. Oblivious vs. Adaptive Attacks

Adversarial attacks targeting malware detection systems can be broadly classified into oblivious and adaptive attacks, based on the knowledge the attacker has of the adversary detector g .

Oblivious attacks operate without any knowledge of the adversary detector g . These attacks aim solely to bypass the malware detector f by ensuring that adversarial inputs are misclassified as benign. As such, oblivious attacks overlook the presence of g , making them less effective when g is actively present. The adversary detector g can flag such inputs as adversarial, limiting their ability to evade the overall detection framework.

Adaptive attacks, on the other hand, explicitly account for the adversary detector g , represented as $g(z') = \psi_\vartheta(\phi(z'))$. These attacks require that the adversarial input not only evades the malware detector f but also satisfies $g(z') \leq \tau$. This dual requirement makes adaptive attacks more complex and challenging to defend against.

The optimization problem for adaptive attacks is formulated as:

$$\min_{x' \in [\underline{u}, \bar{u}]} \mathcal{L}_\varphi(\theta, x', 0) \quad \text{s.t.} \quad \psi_\vartheta(x') \leq \tau \quad \text{and} \quad x' \in X. \quad (6)$$

This formulation substitutes the condition $\varphi_\theta(x') = 0$ with minimizing $\mathcal{L}_\varphi(\theta, x', 0)$, addressing the issue of non-differentiability.

A critical challenge in this optimization process lies in the non-linear nature of the constraint $\psi_\vartheta(x') \leq \tau$, which precludes the use of standard gradient descent techniques. Unlike oblivious attacks, the problem cannot be reformulated by simply swapping the objective and constraint. The non-linear constraint imposed by g requires a more sophisticated approach to ensure that adversarial inputs evade both f and g while adhering to distortion bounds.

To address this difficulty, existing literature adopts a Lagrangian relaxation approach, previously applied to constructing minimum-distortion adversarial examples [28]. The optimization objective is reformulated as:

$$\min_{x' \in [\underline{u}, \bar{u}]} \mathcal{L}_\varphi(\theta, x', 0) + \lambda \psi_\vartheta(x') \quad (7)$$

where $\lambda \geq 0$ is a penalty factor that adjusts the relative importance of evading f versus reducing the response of g . The optimal value of λ is determined through a binary search,

ensuring an effective trade-off between these competing objectives [29].

This penalty-based formulation enables adaptive attacks to effectively bypass both f and g , while addressing the inherent complexities of optimizing over a constrained, non-linear adversarial detection model.

E. Binary Rounding Methods

Binary rounding is essential for generating adversarial examples that respect the binary feature constraints inherent in malware detection. During optimization, gradient-based methods often yield intermediate solutions with continuous feature values, violating these constraints. To address this, Previous studies have explored two primary approaches:

- **Deterministic Binary Rounding:** This method applies a fixed threshold, typically 0.5, to convert continuous values into binary features. Values equal to or greater than 0.5 are rounded to 1, while values below 0.5 are rounded to 0.
- **Randomized Binary Rounding:** This probabilistic approach rounds continuous values based on their magnitude. For example, a value of 0.7 is rounded to 1 with a probability of 70% and to 0 with a probability of 30%.

III. PROPOSED APPROACH

This section presents the proposed methods for crafting adversarial examples in binary feature spaces. The approach addresses two key challenges: (i) ensuring perturbations conform to binary constraints and (ii) achieving dual objectives—evading malware detection and bypassing adversarial defense mechanisms.

A. Threat Model and Design Objective

We assume a white-box attack setting, where the attacker has full knowledge of both the malware detector f and the adversary detector g , including their architectures, trained parameters, and feature representations [28], [30]. This assumption allows for worst-case evaluations of the robustness of ML-based defenses against adversarial attacks.

The attacker’s objective is to generate adversarial examples that evade f while remaining undetected by g . An approximate inverse mapping function ϕ^{-1} is employed, as proposed in prior studies [20], [23], to translate feature-space perturbations into the problem space, ensuring the adversarial examples retain their functional validity.

It is worth noting that in oblivious attacks, the attacker is unaware of the existence of g and, consequently, does not consider g in the process of generating adversarial examples.

B. Incorporating Distance Metrics

An appropriate distance metric is essential for quantifying the similarity between original and adversarial samples. In binary feature spaces, the *Hamming distance* (d_H) is a natural

choice, capturing the number of differing features between two binary vectors:

$$d_H(x, x') = \sum_{i=1}^d \mathbb{I}(x_i \neq x'_i). \quad (8)$$

where $\mathbb{I}(\cdot)$ is the indicator function. The Hamming distance aligns directly with the ℓ_0 -norm for binary vectors and serves as a structured and interpretable measure of dissimilarity.

C. Problem Formulation

The adversarial attack is formulated as an optimization problem in the binary feature space. For a given malware instance-label pair (x, y) , where $x = \phi(z)$ and $y = 1$, the attacker seeks to determine the minimal feature modification δ^* , quantified by the Hamming distance d_H , such that the resulting adversarial example $x^* = x + \delta^*$ satisfies:

$$\delta^* \in \arg \min_{\delta} d_H(x, x + \delta), \quad (9)$$

$$\text{s.t. } \varphi_{\theta}(x + \delta) = 0, \quad (10)$$

$$\psi_{\vartheta}(x + \delta) \leq \tau, \quad (11)$$

$$x + \delta \in \mathcal{X}, \quad (12)$$

$$x + \delta \in [\underline{u}, \bar{u}], \quad (13)$$

Due to the intractability of solving this problem directly, we reformulate it using a smooth surrogate objective:

$$\delta^* \in \arg \min_{\delta} \mathcal{L}(\theta, \vartheta, x + \delta) + \frac{1}{d} \tilde{d}_H(x, x + \delta), \quad (14)$$

$$\text{s.t. } x + \delta \in \mathcal{X}, \quad x + \delta \in [\underline{u}, \bar{u}], \quad (15)$$

Where $\tilde{d}_H(x, x + \delta)$ represents a differentiable approximation of the Hamming distance, normalized by the number of features d to ensure its value lies within the interval $[0, 1]$. This normalization removes the need for hyperparameter tuning to balance the trade-off between the loss and the perturbation size, thereby avoiding computationally expensive line searches for each input sample [31].

The total loss \mathcal{L} is defined as:

$$\mathcal{L}(\theta, \vartheta, x + \delta) = \mathcal{L}_{\varphi}(\theta, x + \delta, 0) + C \cdot \mathcal{L}_{\psi}(\vartheta, x + \delta). \quad (16)$$

where:

$$\mathcal{L}_{\varphi}(\theta, x + \delta, 0) = \max(\varphi_{\theta}^1(x + \delta) - \varphi_{\theta}^0(x + \delta), -\kappa_1). \quad (17)$$

$$\mathcal{L}_{\psi}(\vartheta, x + \delta) = \max(\psi_{\vartheta}(x + \delta) - \tau, -\kappa_2). \quad (18)$$

In these equations:

- $\mathcal{L}_{\varphi}(\theta, x + \delta, 0)$ represents the loss function of the malware detector f , with a target label of zero.
- $\mathcal{L}_{\psi}(\vartheta, x + \delta)$ represents the loss function of the adversary detector g .
- φ_{θ}^k represents the logit output of f for class $k \in \{0, 1\}$.
- $C \geq 0$ is a penalty weight optimized via binary search to balance the loss contributions of f and g .
- κ_1 and κ_2 are confidence margins for f and g , ensuring robust misclassification and detection avoidance.

We define the loss for the malware detector f using the differences between logits, a method whose rationale is extensively discussed in [28]. Furthermore, independently bounding

Algorithm 1: σ -binary Attack with Prioritized Binary Rounding

Input: $x \in [0, 1]^d$: input sample;
 θ : malware detector f ; ϑ : adversary detector g ;
 $N, N_{\text{binary_search}}$: max iterations; η_0 : initial step size;
 σ : smoothing parameter; C_{init} : initial penalty factor;
 γ_0 : initial sparsity threshold; t : sparsity adjustment factor;
 $\kappa_{1,\text{init}}, \kappa_{2,\text{init}}$: initial confidence values;
 α_1, α_2 : confidence bounds; ϵ : loss convergence threshold.
Output: x^* : binary adversarial example.
 $\text{Loss}(\theta, \vartheta, x, \delta) = \mathcal{L}(\theta, \vartheta, x + \delta) + \frac{1}{d} \tilde{d}_H(x, x + \delta)$;
Initialize: $\delta^* \leftarrow \infty, C \leftarrow C_{\text{init}}, \text{best_dist} \leftarrow \infty$; **for**
OuterStep $\leftarrow 1$ **to** $N_{\text{binary_search}}$ **do**
Initialize: $\delta \leftarrow 0, \gamma \leftarrow \gamma_0, \eta \leftarrow \eta_0, \kappa_1 \leftarrow \kappa_{1,\text{init}},$
 $\kappa_2 \leftarrow \kappa_{2,\text{init}};$
for $i \leftarrow 1$ **to** N **do**
 $\nabla g \leftarrow \nabla_{\delta} \text{Loss}(\theta, \vartheta, x, \delta);$
 $\nabla g \leftarrow \frac{\nabla g}{\|\nabla g\|_{\infty}};$
 $\eta \leftarrow \text{cosine_annealing}(\eta_0, i);$
 $\delta \leftarrow \text{clip}(x + (\delta - \eta \cdot \nabla g)) - x;$
 $\delta \leftarrow \Pi_{\gamma}(\delta);$
if $(x + \delta)$ evades both f and g **then**
 $\gamma \leftarrow \gamma + t \cdot \eta;$
else
 $\gamma \leftarrow \gamma - t \cdot \eta;$
 $\delta_{\text{binary}} \leftarrow R_{\text{prioritized_binary}}(x, x + \delta, \theta, \vartheta);$
if $d_H(\delta_{\text{binary}}, 0) < \text{best_dist}$ **and**
 $(x + \delta_{\text{binary}})$ evades both f & g **then**
 $\delta^* \leftarrow \delta_{\text{binary}};$
 $\text{best_dist} \leftarrow d_H(\delta_{\text{binary}}, 0);$
if $|\text{Loss} - \text{previous_Loss}| < \epsilon$ **then**
if $(x + \delta)$ evades both f & g **then**
 $\kappa_1 \leftarrow -\alpha_1, \kappa_2 \leftarrow -\alpha_2;$
else
 $\kappa_1 \leftarrow \alpha_1, \kappa_2 \leftarrow \alpha_2;$
 $\text{Update } C \text{ via binary search};$
Return: $x^* \leftarrow x + \delta^*;$

the loss functions ensures that minimizing one loss term does not inherently satisfy the constraints for both detectors, thereby preserving the distinct objectives of f and g .

1) *Hamming Distance Approximation:* To enable gradient-based optimization, the Hamming distance d_H is approximated as:

$$\tilde{d}_H(x, x + \delta) = \sum_{i=1}^d \frac{\delta_i^2}{\delta_i^2 + \sigma}. \quad (19)$$

Here, $\sigma > 0$ serves as a smoothing parameter to ensure differentiability. However, this approximation may lead to non-sparse solutions. To mitigate this, an adaptive projection operator Π_{γ} enforces sparsity by setting perturbation components below a threshold γ to zero, promoting both sparsity and adversarial effectiveness [31].

2) *Binary Rounding:* To convert continuous solutions into binary values, we employ *Prioritized Binary Rounding* (detailed in Subsection III-D), which prioritizes feature modifications based on their impact on model decisions. This method ensures compliance with binary constraints while minimizing perturbations.

3) *Dynamic Confidence Adjustment:* In adversarial optimization, the solution typically converges to an optimal result in the continuous-valued space. However, during the binary rounding process, where continuous values are discretized into binary form, suboptimality can be introduced. This suboptimality arises because the process of forcing discrete values can disrupt the fine-tuned structure of the continuous solution. To address this issue, we employ the *Dynamic Confidence Adjustment* method. After the initial convergence of the optimization process, the confidence parameter κ is dynamically adjusted within a bounded range $[-\alpha, +\alpha]$. This adjustment enables the optimization process to explore regions near the decision boundary, facilitating the discovery of a binary solution that more effectively satisfies the adversarial objectives.

4) *Algorithm Details for the σ -binary Attack:* The σ -Binary Attack, outlined in Algorithm 1, generates binary adversarial examples with minimal perturbations to evade the malware detector f and, if applicable, the adversary detector g . Key parameters, such as the penalty factor C , sparsity threshold, and confidence margins κ_1 and κ_2 , are initialized with κ_1 and κ_2 set to small values to prioritize early boundary crossing for effective evasion.

The algorithm employs two nested loops: an outer binary search loop to adjust C for balancing loss terms, and an inner optimization loop to minimize a loss function combining the evasion objective and a smooth approximation of the Hamming distance. Perturbations are iteratively updated via a cosine annealing learning rate, clipped gradient descent, and enforced sparsity. Non-binary perturbations are converted to binary form using *prioritized binary rounding* to maintain functionality and input constraints.

Once the total loss converges, the thresholds are dynamically adjusted throughout the optimization process based on evasion success. The algorithm ultimately outputs the best adversarial example, denoted $x^* = x + \delta^*$, which minimizes the Hamming distance while successfully bypassing detection.

D. Prioritized Binary Rounding

To efficiently map continuous perturbations to binary feature vectors, the proposed σ -binary attack framework introduces a novel technique called *Prioritized Binary Rounding*. This method ensures compliance with binary constraints while minimizing unnecessary feature modifications, balancing adversarial effectiveness and sparsity.

Traditional binary rounding methods often use uniform deterministic or randomized thresholds, which overlook variations in feature importance and adversarial impact. In contrast, Prioritized Binary Rounding selects features for modification based on two critical factors:

- **Perturbation Magnitude:** Features exhibiting larger deviations from their original values are prioritized for modification.

- **Feature Importance:** Features that significantly influence the model’s decision boundary are prioritized to enhance adversarial effectiveness.

Algorithm 2: Prioritized Binary Rounding

Input: $\mathbf{x}_{\text{orig}} \in [0, 1]^d$: original input;
 $\mathbf{x}_{\text{adv}} \in [0, 1]^d$: adversarial input;
 f_θ : target model; τ : perturbation threshold.
Output: \mathbf{x}^* : binary adversarial example.
 Initialize: $\mathbf{x}^* \leftarrow \mathbf{x}_{\text{orig}}$;
if $f_\theta(\mathbf{x}^*)$ is benign **then**
 return \mathbf{x}^* ;
 Compute perturbation magnitude: $\Delta \leftarrow |\mathbf{x}_{\text{adv}} - \mathbf{x}_{\text{orig}}|$;
 Generate mask for significant perturbations:
 $\mathbf{M} \leftarrow \mathbb{1}(\Delta > \tau)$;
 Sort indices by descending perturbation magnitude:
 FeatureOrder $\leftarrow \text{argsort}(\Delta, \text{desc})$;
for $i \leftarrow 1$ **to** $\sum(\mathbf{M})$ **do**
 Update feature $j \leftarrow \text{FeatureOrder}[i]$:

$$\mathbf{x}^*[j] \leftarrow \begin{cases} 1, & \text{if } \mathbf{x}_{\text{adv}}[j] > \mathbf{x}_{\text{orig}}[j], \\ 0, & \text{otherwise.} \end{cases}$$

 if $f_\theta(\mathbf{x}^*)$ is benign **then**
 return \mathbf{x}^* ;
return \mathbf{x}^* ;

The algorithm operates as follows:

First, the perturbation magnitude for each feature is computed as the absolute difference between the adversarial and original feature values. Features with perturbations exceeding a predefined threshold τ are marked as candidates for modification using a binary mask. Candidate features are sorted in descending order based on perturbation magnitude and relative importance, determined by their position in the feature space, to prioritize the most impactful changes.

Each selected feature is iteratively updated by rounding it to one if the adversarial feature value is greater than the original; otherwise, it is set to zero. After each modification, the adversarial example is evaluated to determine whether it successfully evades detection by the malware detector. If the detector classifies the sample as benign, the rounding process terminates early, and the rounded result is returned. If early termination does not occur, the algorithm continues updating all significant features before outputting the final binary adversarial example.

By leveraging this method, the σ -binary attack achieves higher success rates with minimal perturbations, making it a robust and efficient solution for adversarial attacks in malware detection scenarios.

IV. EXPERIMENTS

To evaluate the effectiveness of our proposed methods, we conducted extensive experiments across various scenarios. Our analysis addresses two primary categories of research questions (RQs): (i) comparing the proposed methods with

prior work (RQ1 and RQ2), and (ii) using these methods to evaluate the performance of existing defenses (RQ3, RQ4, and RQ5). Specifically, we sought answers to the following:

- **RQ1: Effectiveness of Prioritized Binary Rounding.** How effective is the proposed Prioritized Binary Rounding technique in comparison to existing binary rounding methods?
- **RQ2: Effectiveness of the σ -binary attack.** How does our proposed σ -binary attack perform relative to other attack strategies in binary feature spaces?
- **RQ3: Baseline performance of defenses.** How effective are defense mechanisms under non-adversarial conditions?
- **RQ4: Robustness against oblivious attacks.** How resilient are defenses against oblivious attacks, where attackers are unaware of the adversary detector g ?
- **RQ5: Robustness against adaptive attacks.** How robust are the defenses against adaptive attacks? This evaluates performance under worst-case scenarios.

A. Experimental Framework and Configuration

This section provides a detailed overview of the dataset, feature extraction process, defense mechanisms, attack methods, evaluation metrics, and experimental configurations utilized in our study.

1) *Dataset:* We utilized the *MalScan* dataset [26], a widely recognized collection comprising 23,196 Android applications, with 11,583 malicious and 11,613 benign samples. The dataset spans applications from 2011 to 2018. The data was randomly split into training (60%), validation (20%), and testing (20%) subsets.

2) *Feature Extraction:* Feature extraction was performed using Drebin [4], which analyzes Android application packages (APKs) to construct a binary feature space. Features were extracted from the Android manifest and the disassembled dexcode using the Androguard tool. These features were organized into eight categories, including hardware components, permissions, intents, API calls, and class names. Each APK is represented as a binary feature vector, where each dimension indicates the presence (1) or absence (0) of a specific feature. Consistent with prior work [23], we exclude easily modifiable features (e.g., package names) and retain the 10,000 most frequent features, ensuring robust representation for evaluation.

3) *Defenses Considered for Comparative Analysis:* In this study, we evaluate a range of defenses drawn from prior research, categorized into three distinct types: (i) methods that enhance the resilience of malware detectors through adversarial training, (ii) methods that integrate malware detection with adversarial example detection mechanisms, and (iii) methods that combine both adversarial training and adversary detection mechanisms for comprehensive defense.

Below, we provide a detailed overview of each defense:

- **DNN [8]:** A baseline deep neural network model for malware detection. This model does not incorporate specific countermeasures against evasion attacks and serves

as a reference for comparing the robustness of advanced defense mechanisms.

- **DNN⁺** [32]: An extension of the DNN model that incorporates a secondary detector. This detector introduces an additional outlier class to identify adversarial examples.
- **KDE** [33]: Combines a DNN with a Kernel Density Estimation (KDE)-based secondary detector. The KDE mechanism identifies adversarial examples by analyzing deviations in activations of the DNN’s penultimate layer.
- **DLA** [34]: Dense Layer Analysis (DLA) incorporates a secondary detector alongside the DNN. By evaluating activations across all dense layers, DLA distinguishes between normal and adversarial examples, adding a robust layer of defense.
- **AT-rFGSM^k** [9]: A defense mechanism that strengthens the DNN through adversarial training using the FGSM^k attack. This approach employs randomized rounding projections to increase the model’s resilience to adversarial perturbations.
- **AT-MaxMA** [20]: Enhances DNN robustness by incorporating adversarial training with a Mixture of Attacks (MaxMA) strategy. This method combines multiple attack techniques to construct a more comprehensive and resilient defense.
- **ICNN** [23]: Integrates a DNN with an Input Convex Neural Network (ICNN) as a secondary detector. The ICNN operates independently of the DNN’s architecture and analyzes the feature space to detect adversarial examples.
- **PAD-SMA** [23]: integrates a DNN-based malware detector with an ICNN-based adversary detector. Both components are fortified through adversarial training using the Stepwise Mixture of Attacks (SMA), making this approach a representative example of defenses that combine adversarial training with adversary detection mechanisms.

This comprehensive classification facilitates a thorough analysis of the robustness of various defenses against the proposed σ -binary attack.

4) Evasion Attack Methods for Comparative Analysis:

To benchmark our proposed σ -binary attack, we compare it against four well-established adversarial attack methods: CW, σ -zero, Mimicry, and PGD. Below, we detail each method:

a) *Projected Gradient Descent (PGD) Attacks*: PGD [35] is an iterative optimization method that identifies perturbations maximizing adversarial loss while adhering to predefined constraints. The update rule is expressed as:

$$\delta_x^{(t+1)} = \text{Proj}_{[u-x, u-x]} \left(\delta_x^{(t)} + \alpha \nabla_{\delta_x} F(\theta, x + \delta_x^{(t)}, 1) \right),$$

where $\alpha > 0$ is the step size, $\nabla_{\delta_x} F$ denotes the gradient of the loss function, and Proj ensures that perturbations remain within constraint bounds. To address small gradients and enhance adversarial success, researchers employ normalization in different norms, such as the ℓ_∞ -norm, ℓ_2 -norm, or ℓ_1 -norm. Each norm identifies the steepest direction, which is then utilized to compute effective perturbations [36].

b) *CW Attack*: The Carlini & Wagner (CW) attack [28] minimizes the ℓ_2 -norm of perturbations while ensuring misclassification. It achieves this by solving a carefully crafted op-

timization problem through gradient-based methods, resulting in highly effective evasion with minimal feature modification.

c) *σ -zero Attack*: The σ -zero attack [31] is designed to minimize the number of modified features by approximating the ℓ_0 -norm and applying adaptive projections. It is effective in generating highly sparse perturbations, challenging defenses that assume adversarial examples require extensive feature modifications.

d) *Mimicry Attack*: Mimicry [27], [37] is a gradient-free adversarial attack that perturbs malware samples to resemble benign samples. By iteratively querying the target model and adjusting perturbations, it effectively bypasses detection mechanisms without requiring knowledge of model parameters.

5) *Evaluation Metrics*: To rigorously assess the performance and robustness of defenses, we utilize a comprehensive suite of evaluation metrics. These metrics capture both the classification accuracy and resilience to adversarial attacks, and are defined as follows:

- **False Positive Rate (FPR)**: The proportion of benign examples that are incorrectly classified as malicious. This reflects the model’s tendency to overestimate threats.
- **False Negative Rate (FNR)**: The proportion of malicious examples that are mistakenly classified as benign, indicating the model’s vulnerability to evasion by malware.
- **Accuracy (Acc)**: The overall ratio of correctly classified examples to the total number of samples, representing the model’s general predictive capability.
- **F1 Score**: The harmonic mean of precision and recall, offering a robust evaluation for imbalanced datasets by balancing the trade-off between false positives and false negatives.
- **Attack Success Rate (ASR_k)**: The fraction of adversarial attacks that successfully evade detection, where the Hamming distance between the original and adversarial samples satisfies $d_H(\mathbf{x}, \mathbf{x}^*) \leq k$. This metric measures the model’s resistance to perturbations of varying magnitudes.
- **Median Perturbation ($\tilde{d}_{H, \text{median}}$)**: The median Hamming distance of successful adversarial examples, quantifying the typical modification magnitude required to deceive the detector.
- **Excluded Samples (ExS)**: For deferred detection models, this metric represents the fraction of inputs that are deferred by the adversary detector g .

These metrics collectively provide a robust framework for evaluating both the predictive accuracy and adversarial robustness of the proposed defenses.

6) *Experimental Setup and Configuration*: To evaluate the effectiveness of gradient-based attacks, we generate adversarial examples using all 3,104 malicious samples from the test dataset. Each attack is executed with 10,000 iterations of gradient descent, ensuring a thorough exploration of the attack surface. While convergence often occurs earlier, this approach allows for a comprehensive evaluation [28]. For the CW attack, 8 iterations of binary search are conducted to optimize the penalty weight, maximizing the impact of adversarial perturbations.

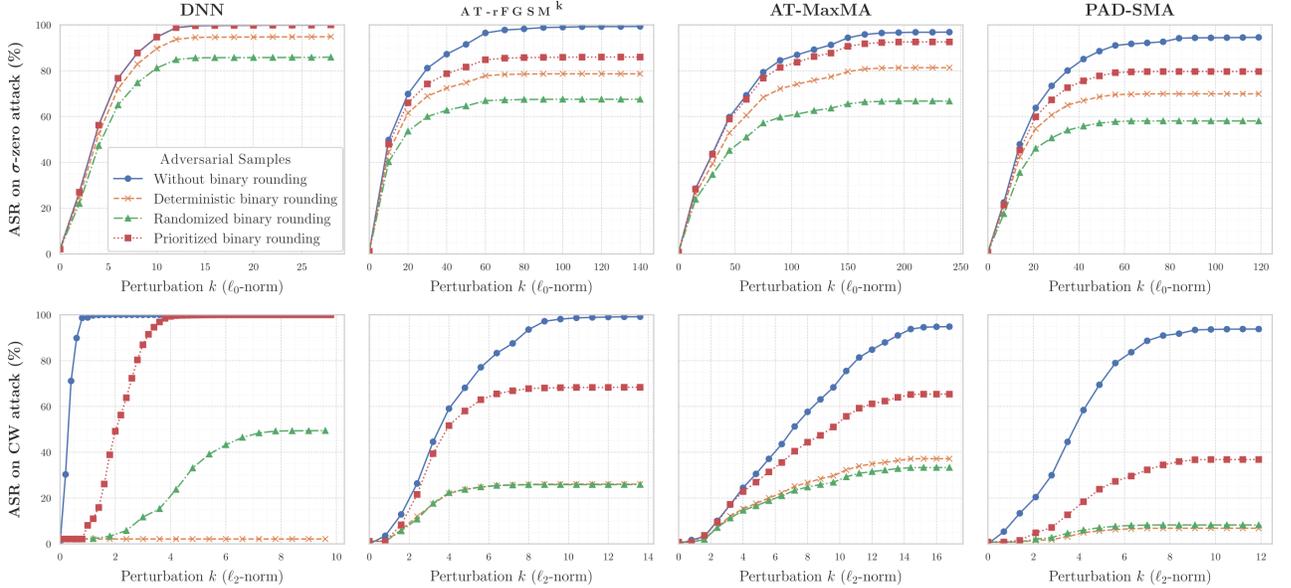


Figure 1: Robustness comparison of defense models under various binary rounding methods (ASR vs. perturbation budget k).

As gradient-based attacks produce continuous-valued outputs, the Prioritized Binary Rounding method is employed to enforce binary feature constraints. This post-processing step ensures the generated adversarial examples adhere to the constraints of the domain while maintaining their effectiveness. Hyperparameters for each attack are meticulously tuned for alignment with the defense models under evaluation, ensuring fairness and consistency in the experimental results.

For defenses equipped with adversary detectors, the process is adjusted to use 4 iterations of binary search to identify the smallest effective penalty factor c . Additionally, thresholds for identifying suspicious samples are computed by excluding the top 5% of validation examples with the highest confidence, as recommended by [23] and [34]. This strategy ensures that the thresholds are both practical and minimally disruptive to benign classification performance.

B. RQ1: Effectiveness of Prioritized Binary Rounding

This section evaluates the effectiveness of the proposed Prioritized Binary Rounding method by comparing it with two widely used techniques: Deterministic Binary Rounding and Randomized Binary Rounding. The analysis examines adversarial examples crafted by two distinct and representative attacks: the CW attack, which is an optimization-based method focused on minimizing the ℓ_2 -norm of perturbations, and the σ -zero attack, a sparsity-driven approach specifically targeting the ℓ_0 -norm to induce minimal feature modifications. Since the outputs of these attacks are continuous-valued, binary rounding is essential to satisfy the binary feature constraints inherent to binary-constrained domains.

Four defense models were evaluated to ensure comprehensive analysis. The DNN serves as a baseline model without

specialized defenses, while AT-rFGSM^k, AT-MaxMA, and PAD-SMA (in conservative mode) represent robust defenses known for their effectiveness against adversarial attacks. These models span a spectrum of robustness levels, as further discussed in IV-F.

To ensure a clear comparison, binary-rounded adversarial examples were evaluated alongside their original continuous outputs. The ℓ_2 -norm perturbation (characterizing the CW attack) and ℓ_0 -norm perturbation (representing the σ -zero attack) were employed to evaluate the effectiveness of various rounding methods. Figure 1 presents a comparative analysis of the performance of each rounding technique across the defense models.

As depicted in Figure 1, Prioritized Binary Rounding significantly outperforms Deterministic and Randomized Binary Rounding in the σ -zero attack scenario. It achieves results closely aligned with the original continuous outputs, particularly on the DNN model, where its performance nearly overlaps with the continuous baseline. This effectiveness stems from its alignment with the ℓ_0 -norm, which the σ -zero attack optimizes, and its ability to preserve the adversarial success rate by minimizing the number of feature modifications. Conversely, Randomized Binary Rounding demonstrates the weakest performance, failing to maintain the effectiveness of adversarial examples generated by the σ -zero attack.

In the CW attack, a more noticeable gap exists between the original continuous outputs and all binary rounding methods. This is attributable to the CW attack’s focus on minimizing the ℓ_2 -norm, which often results in numerous small fractional changes, complicating binary conversion. Despite this challenge, Prioritized Binary Rounding still surpasses Deterministic and Randomized Binary Rounding, demonstrating superior robustness across both ℓ_0 -norm and ℓ_2 -norm-based attacks.

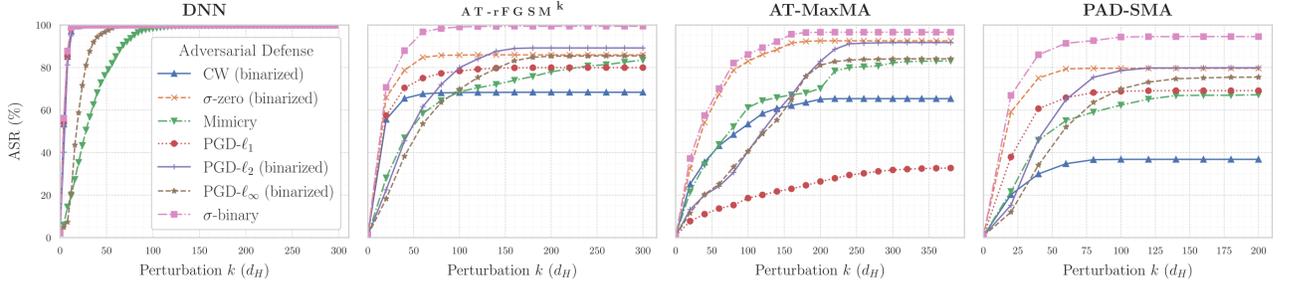


Figure 2: Robustness comparison of defense models under various attack methods (ASR vs. perturbation budget k).

To answer RQ1, Prioritized Binary Rounding outperforms Deterministic and Randomized Binary Rounding by closely preserving the success rates of adversarial attacks across different distance metrics. Its versatility allows for the effective conversion of a broad range of continuous adversarial examples into binary formats, making it a robust and general-purpose solution for binary-constrained domains.

C. RQ2: Effectiveness of σ -binary attack

This section evaluates the effectiveness of our proposed σ -binary attack in comparison with several adversarial attack methods, including the CW attack (ℓ_2 -norm-based), the σ -zero attack (ℓ_0 -norm-based), and the Mimicry attack (a gradient-free approach). Additionally, we examine the performance of PGD- ℓ_1 , PGD- ℓ_2 , and PGD- ℓ_∞ , which represent attacks that do not explicitly minimize perturbation. For simplicity, throughout this subsection, PAD-SMA refers exclusively to its configuration in conservative mode.

The results in Table I and Figure 2 demonstrate the superior performance of the σ -binary attack across all evaluated defenses. Against the baseline DNN, all attacks—except Mimicry—, including σ -binary, achieve ASR_∞ of 100%, highlighting the vulnerability of DNNs to adversarial perturbations. However, significant differences arise when robust defenses such as AT-rFGSM k , AT-MaxMA, and PAD-SMA are evaluated.

For AT-rFGSM k , σ -binary achieves an impressive ASR_∞ of 99.45%, followed by PGD- ℓ_2 (89.21%) and σ -zero (85.95 %). On AT-MaxMA, σ -binary reaches ASR_∞ of 96.62%, outperforming σ -zero (92.56%) and PGD- ℓ_2 (91.75%). Against the strongest defense, PAD-SMA, σ -binary achieves an ASR_∞ of 94.56%, significantly surpassing PGD- ℓ_2 (79.90%) and σ -zero (79.64%). In contrast, PGD- ℓ_1 struggles to adapt to robust defenses, with a low ASR_∞ of 32.76% on AT-MaxMA, indicating limited versatility.

The CW attack’s poor performance after binarization is noteworthy. Designed to minimize ℓ_2 -norm perturbations, CW produces adversarial examples near the decision boundary, making it particularly vulnerable to binarization, which disrupts its finely tuned perturbations. In contrast, PGD- ℓ_2 and PGD- ℓ_∞ avoid this limitation by producing perturbations that push further beyond the decision boundary, resulting in higher success rates post-binarization.

Table I: Accuracy (%) of defenses under adaptive attacks.

Attack name	ASR_∞ (%)			
	DNN	AT-rFGSM k	AT-MaxMA	PAD-SMA
CW (binarized)	100.00	68.36	65.37	36.82
σ -zero (binarized)	100.00	85.95	92.56	79.64
Mimicry	99.81	83.92	83.09	67.20
PGD- ℓ_1	100.00	79.96	32.76	69.14
PGD- ℓ_2 (binarized)	100.00	89.21	91.75	79.90
PGD- ℓ_∞ (binarized)	100.00	85.47	84.09	75.52
σ -binary	100.00	99.45	96.62	94.56

As shown in Figure 2, σ -binary consistently achieves higher ASRs at smaller perturbation budgets compared to other attacks, demonstrating its efficiency in generating effective adversarial examples with minimal input modifications. Although PGD- ℓ_2 and PGD- ℓ_∞ improve slightly at larger perturbation budgets, they remain less effective than σ -binary, especially against PAD-SMA. Mimicry, a gradient-free attack, performs poorly against PAD-SMA due to the robust adversarial detection capabilities of PAD-SMA, particularly against high-perturbation attacks.

Answer to RQ2: The σ -binary attack is highly effective across a wide range of defenses, including PAD-SMA. It consistently achieves higher ASRs with smaller perturbation budgets, outperforming all other evaluated attacks and establishing itself as a robust and efficient adversarial attack method.

D. RQ3: Baseline performance of defenses

This subsection evaluates the performance of defense models in the absence of adversarial attacks. The defense categorized into two groups: (1) defenses without adversary detectors (DNN, AT-rFGSM k , AT-MaxMA) and (2) those with adversary detectors (DNN $^+$, KDE, DLA, ICNN, PAD-SMA), operating in either *deferred* or *conservative* modes.

As summarized in Table II, models with adversary detectors in deferred mode consistently outperform their conservative counterparts. Deferred models exclude a small percentage of challenging samples, which simplifies classification and boosts accuracy. Notably, DNN $^+$ (Deferred) achieves the highest accuracy (98.12%) and F1 score (98.12%), excluding 4.98% of samples. Similarly, DLA (Deferred) and KDE (Deferred) achieve competitive results with F1 scores of 97.86% and

Table II: Effectiveness (%) of detectors in the absence of attacks.

Defense	FNR (%)	FPR (%)	Acc (%)	F1 (%)	ExS (%)
DNN	2.13	3.13	97.38	97.42	-
AT-rFGSM ^k	1.13	6.45	96.24	96.37	-
AT-MaxMA	0.77	9.61	94.86	95.12	-
KDE (Deferred)	2.24	3.17	97.29	97.30	3.71
KDE (Conservative)	2.13	5.53	96.19	96.29	-
DLA (Deferred)	1.53	2.61	97.91	97.86	5.54
DLA (Conservative)	1.39	4.28	97.18	97.25	-
DNN ⁺ (Deferred)	1.54	2.23	98.12	98.12	4.98
DNN ⁺ (Conservative)	1.45	6.22	96.19	96.32	-
ICNN (Deferred)	2.05	3.24	97.37	97.48	5.11
ICNN (Conservative)	2.00	10.63	93.73	94.05	-
PAD-SMA (Deferred)	0.86	7.86	95.62	95.73	4.56
PAD-SMA (Conservative)	0.81	10.43	94.43	94.74	-

97.30%, respectively, while excluding 5.54% and 3.71% of samples.

Conservative-mode models prioritize strict classification, treating all suspicious samples flagged by the detector g as malicious. This approach results in elevated false positive rates (FPRs), reducing overall performance. For instance, ICNN (Conservative) has an FPR of 10.63%, compared to 3.24% in its deferred counterpart. Similarly, PAD-SMA (Conservative) shows an accuracy of 94.43%, lower than its deferred counterpart (95.62%).

Adversarially hardened models, such as AT-rFGSM^k, AT-MaxMA, and PAD-SMA, exhibit lower false negative rates (FNRs), highlighting their ability to detect malicious samples effectively. However, this robustness against adversarial attacks comes at the cost of performance under benign conditions. For example, AT-rFGSM^k achieves an FNR of 1.13% and an FPR of 6.45%, resulting in an accuracy of 96.24%. PAD-SMA (Deferred) achieves an FNR of 0.86% and an FPR of 7.86%, with an accuracy of 95.62%, lagging behind other deferred models due to its reliance on adversarial training, which biases its detector g and increases FPR [23].

Answer to RQ3, In the absence of attacks, deferred-mode defenses such as DNN⁺, DLA, and KDE achieve superior accuracy and F1 scores by excluding challenging samples, whereas conservative-mode models and adversarially hardened defenses face trade-offs between robustness and performance, with higher FPRs and lower accuracy when there are no adversarial attacks.

E. RQ4: Robustness against oblivious attacks

In this subsection, we assess the robustness of five defenses—KDE, DLA, DNN⁺, ICNN, and PAD-SMA—against oblivious attacks using the proposed σ -binary attack. These defenses integrate an adversary detector g , which is not explicitly targeted in this evaluation.

Oblivious attacks operate under the assumption that attackers are unaware of the presence or functionality of the detector g , focusing solely on evading the main classifier. This scenario provides a baseline for evaluating the intrinsic robustness of g and the overall defense system.

While non-adaptive attack evaluations are necessary, they are insufficient to fully gauge robustness. Success against such

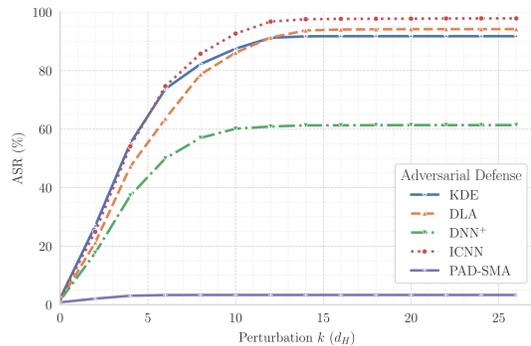
Figure 3: Robustness evaluation curves (ASR vs. perturbation budget k) against oblivious attacks.

Table III: Robustness against Oblivious Attacks

Defense	ASR _∞ (%)	ExS (%)	$\tilde{\ell}_{0,median}$
KDE (Deferred)	100.00	8.25	4.00
KDE	91.75	-	4.00
DLA (Deferred)	100.00	5.83	4.00
DLA	94.17	-	4.00
DNN ⁺ (Deferred)	100.00	38.63	4.00
DNN ⁺	61.37	-	4.00
ICNN (Deferred)	100.00	2.09	4.00
ICNN	97.91	-	4.00
PAD-SMA (Deferred)	100.00	96.68	2.00
PAD-SMA	3.32	-	2.00

weaker adversaries does not guarantee resilience to adaptive attacks [18]. However, these evaluations can still highlight vulnerabilities in defenses under minimal adversarial threats.

The robustness evaluation considers two configurations: *deferred* and *conservative*. In the deferred mode, samples flagged as suspicious by g are excluded from the ASR calculation, isolating the vulnerabilities of the main classifier. In contrast, the conservative mode assumes all suspicious samples flagged by g are malicious, incorporating them into the ASR calculation. Unless explicitly labeled as *Deferred*, results represent the conservative configuration.

Table III summarizes the evaluation results. In the deferred mode, all defenses exhibit ASR_∞ of 100%, underscoring the inability of the malware detector to resist attacks independently. However, the conservative mode reveals substantial variations in robustness, highlighting the critical role of g in mitigating adversarial threats.

Among the evaluated defenses, PAD-SMA exhibits exceptional robustness, achieving an ASR of just 3.32% in its conservative configuration. The flat ASR curve shown in Figure 3 highlights its consistent resilience across varying perturbation budgets. This superior performance, compared to ICNN—which shares a similar defense architecture—can be attributed to PAD-SMA’s incorporation of adversarial training, which significantly enhances the robustness of the adversary detector g .

ICNN exhibits a high ASR of 97.91%, with its steep ASR curve, as depicted in Figure 3, underscoring substantial vulnerability to small perturbations. Similarly, KDE and DLA perform poorly, with ASRs of 91.75% and 94.17%, respectively, in the conservative configuration. DNN⁺ shows

moderate improvement, achieving an ASR of 61.37%, yet it remains significantly less robust than PAD-SMA in resisting oblivious attacks.

Answer to RQ4: PAD-SMA demonstrates unparalleled robustness against oblivious attacks, outperforming other defenses by a wide margin. The considerable vulnerability of ICNN, KDE, and DLA emphasizes the indispensable role of adversarial training in crafting effective defense mechanisms.

F. RQ5: Robustness Against Adaptive Attacks

This subsection evaluates the robustness of defenses against adaptive attacks using our proposed σ -binary attack. Unlike oblivious attacks, adaptive attacks assume full knowledge of the adversary detector g , enabling attackers to craft perturbations that evade both the primary model and g , presenting a greater challenge for defenses.

Table IV and Figure 4 present the robustness of each defense under adaptive attacks. DNN, KDE, DLA, DNN⁺, and ICNN exhibit high vulnerability, with median perturbation ($\tilde{\ell}_{0,median}$) of 4.00 and ASR₁₀ exceeding 90%, indicating susceptibility to small perturbations.

In contrast, adversarially trained defenses such as PAD-SMA, AT-MaxMA, and AT-rFGSM^k show significantly enhanced robustness. AT-MaxMA achieves the strongest resistance at low perturbation budgets, with ASR₁₀ = 23.32%, outperforming PAD-SMA (36.34%) and AT-rFGSM^k (51.35%). This highlights AT-MaxMA’s capability to mitigate low-budget attacks effectively.

At moderate perturbation budgets, as indicated by ASR₅₀, AT-MaxMA maintains its superiority, achieving the lowest value (65.17%), followed by PAD-SMA (89.79%) and AT-rFGSM^k (92.01%). However, for high perturbation budgets, PAD-SMA demonstrates greater robustness, achieving the lowest ASR_∞ = 94.56%, slightly outperforming AT-MaxMA (96.62%) and other defenses. PAD-SMA’s performance reflects its ability to restrict the success of high-budget attacks, as evidenced by its flattened ASR_k curve for $k \geq 50$.

These findings highlight a trade-off between robustness across different perturbation budgets and practical considerations. AT-MaxMA is well-suited for scenarios where small perturbations are prevalent, demonstrating superior robustness at low budgets while maintaining lower false positive rates (FPR) and higher benign classification accuracy. In contrast, PAD-SMA excels against high-budget attacks, showcasing enhanced robustness in such scenarios. However, this comes at the cost of increased FPR and computational overhead, as noted in [23]. Furthermore, PAD-SMA demonstrates significant resilience when attackers are unaware of the adversary detector g , as evidenced in Table III.

Answer to RQ5: Adaptive attacks expose varying degrees of robustness among defenses. DNN, KDE, DLA, DNN⁺, and ICNN exhibit high brittleness, with ASR₁₀ exceeding 90%. While AT-rFGSM^k outperforms these simpler defenses, it falls short compared to AT-MaxMA and PAD-SMA. AT-MaxMA demonstrates exceptional robustness at low and moderate perturbation budgets, achieving ASR₁₀ = 23.32% and ASR₅₀ = 65.17%. Conversely, PAD-SMA excels against

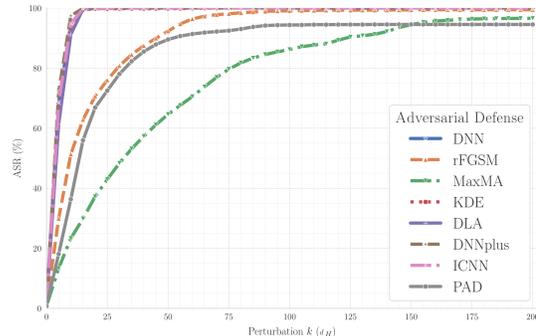


Figure 4: Robustness evaluation curves (ASR vs. perturbation budget k) against adaptive attacks.

Table IV: Robustness against Adaptive Attacks

Defense	ASR ₁₀ (%)	ASR ₅₀ (%)	ASR _∞ (%)	$\tilde{\ell}_{0,median}$
DNN	94.75	100.00	100.00	4.00
AT-rFGSM ^k	51.35	92.01	99.45	10.00
AT-MaxMA	23.32	65.17	96.62	30.00
KDE	93.07	100.00	100.00	4.00
DLA	91.20	100.00	100.00	4.00
DNN ⁺	97.45	100.00	100.00	4.00
ICNN	94.75	100.00	100.00	4.00
PAD-SMA	36.34	89.79	94.56	13.00

high-budget attacks, achieving the lowest ASR_∞ = 94.56%. These results underscore the trade-offs between robustness across different perturbation levels, false positive rates, and computational demands, emphasizing the importance of aligning defenses with specific threat models and operational requirements.

V. RELATED WORK

This section reviews two critical aspects of adversarial research in malware detection: the development of evasion attack strategies and the design of corresponding defense mechanisms. To provide a comprehensive overview, the discussion is divided into *Evasion Attacks in Malware Detection* and *Defenses Against Evasion Attacks*.

A. Evasion Attacks in Malware Detection

Adversarial attacks in Android malware detection can be broadly categorized into *problem-space attacks*, which directly modify applications, and *feature-space attacks*, which manipulate extracted feature representations. Both strategies exploit vulnerabilities in machine learning-based malware detectors while preserving the original functionality of the malware.

Problem-space attacks alter application components such as the manifest file, Dalvik bytecode, or inject benign artifacts to bypass detection. Android HIV [16] perturbs Dalvik bytecode to evade detectors while maintaining functionality. Pierazzi et al. [15] introduced the injection of benign bytecode “gadgets” to fool detection models. Tang et al. [17] expanded this with DapAdv, leveraging a hierarchical attention mechanism to optimize code slicing for adversarial repackaging. Advanced techniques include reinforcement learning-based attacks like HRAT [38] and hybrid approaches using generative adversarial

networks (GANs) [39] with co-evolution algorithms [10] have further improved the effectiveness of problem-space attacks. Black-box methods such as EvadeDroid [11] demonstrate success with minimal feature knowledge, relying on random perturbations in benign code.

Feature-space attacks focus on manipulating extracted feature vectors without altering the application itself. Gradient-based methods [8], [9] optimize perturbations to evade detection. Ensemble-based techniques such as [20] combine diverse manipulation strategies for enhanced efficacy. GAN-based approaches, such as E-MalGAN [40] and the method proposed by Shahpasand et al. [41], craft adversarial examples that evade both malware detectors and adversarial detection systems.

B. Defenses Against Evasion Attacks

Developing effective defenses against adversarial attacks has been a critical area of research, focusing on enhancing model robustness.

Adversarial training is a well-established defense strategy that enhances model resilience by incorporating adversarial examples into the training process. While it proves effective against the perturbations encountered during training [8], this approach faces limitations, including vulnerability to novel attack styles, high computational overhead, and a tendency to overfit to specific attack methods [18].

Ensemble methods aim to enhance robustness by combining multiple classifiers or detectors. Smutz and Stavrou [19] proposed mutual agreement analysis, flagging uncertain predictions based on classifier disagreement. Ficco [21] introduced a dynamic ensemble system combining generic and specialized detectors. However, applying ensemble strategies to deep learning models remains challenging due to difficulties in coordinating diverse model predictions.

Defenses with *auxiliary models* utilize separate frameworks to identify adversarial examples or purify inputs. Li et al. [22] employed a Variational Autoencoder (VAE) to detect adversarial malware via reconstruction errors, while Li et al. [42] combined hash transformations with denoising autoencoders to mitigate perturbations. Despite initial success, these approaches often struggle against adaptive attacks.

Recently, *hybrid approaches* have gained prominence, combining multiple defense strategies to address the limitations of individual methods. For instance, PAD-SMA [23] integrates a malware detection system with an adversary detector, both strengthened through adversarial training with a diverse range of attacks. While this approach enhances robustness and generalization, it also incurs significant computational overhead and exhibits lower accuracy on benign samples, limiting its practicality in real-world scenarios.

The ongoing advancements in both attack and defense mechanisms underscore the dynamic nature of this field, necessitating continuous innovation to address emerging threats effectively.

VI. CONCLUSIONS AND FUTURE WORK

This study introduces a robust adversarial attack framework in binary space and a comprehensive evaluation methodology

for assessing the robustness of ML-based Android malware detection against adversarial attacks in feature space. By proposing the σ -binary attack and Prioritized Binary Rounding, we address key limitations in adapting gradient-based attacks—which typically generate continuous adversarial examples—to binary feature domains. Our results emphasize the critical importance of adversarial training in enhancing robustness. However, they also reveal that even advanced defenses, such as PAD-SMA, remain susceptible to adaptive attacks, highlighting vulnerabilities that were previously underestimated.

This work further identifies inherent trade-offs in robust defenses, including increased false positive rates and computational overhead, underscoring the importance of aligning defense mechanisms with specific operational needs and threat scenarios. These insights provide valuable guidance for designing more effective and efficient defense systems.

Future research directions could focus on expanding the application of the σ -binary attack and Prioritized Binary Rounding to other binary-constrained domains, such as fraud detection and network intrusion detection, to demonstrate their broader applicability. Another critical area is the development of standardized benchmarks for evaluating adversarial robustness in binary domains, which would enable consistent and fair comparisons across studies and foster greater transparency in the field.

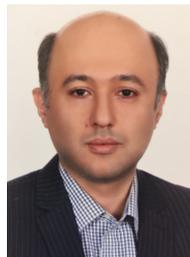
REFERENCES

- [1] kaspersky. (2024) Attacks on mobile devices significantly increase in 2023. [Online]. Available: <https://www.kaspersky.com/about/press-releases/attacks-on-mobile-devices-significantly-increase-in-2023>
- [2] C. Gao, G. Huang, H. Li, B. Wu, Y. Wu, and W. Yuan, "A comprehensive study of learning-based android malware detectors under challenging environments," in *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*, 2024, pp. 1–13.
- [3] J. Liu, J. Zeng, F. Pierazzi, L. Cavallaro, and Z. Liang, "Unraveling the key of machine learning solutions for android malware detection," *arXiv preprint arXiv:2402.02953*, 2024.
- [4] D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, K. Rieck, and C. Siemens, "Drebin: Effective and explainable detection of android malware in your pocket." in *Ndss*, vol. 14, 2014, pp. 23–26.
- [5] Y. He, Y. Liu, L. Wu, Z. Yang, K. Ren, and Z. Qin, "Msdroid: Identifying malicious snippets for android malware detection," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 3, pp. 2025–2039, 2022.
- [6] Y. Liu, C. Tantithamthavorn, L. Li, and Y. Liu, "Deep learning for android malware defenses: a systematic literature review," *ACM Computing Surveys*, vol. 55, no. 8, pp. 1–36, 2022.
- [7] E. Mariconti, L. Onwuzurike, P. Andriotis, E. De Cristofaro, G. Ross, and G. Stringhini, "Mamadroid: Detecting android malware by building markov chains of behavioral models," *arXiv preprint arXiv:1612.04433*, 2016.
- [8] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel, "Adversarial examples for malware detection," in *Computer Security—ESORICS 2017: 22nd European Symposium on Research in Computer Security, Oslo, Norway, September 11-15, 2017, Proceedings, Part II* 22. Springer, 2017, pp. 62–79.
- [9] A. Huang, A. Al-Dujaili, E. Hemberg, U.-M. O'Reilly et al., "Adversarial deep learning for robust detection of binary encoded malware," *arXiv preprint arXiv:1801.02950*, 2018.
- [10] H. Li, Z. Cheng, B. Wu, L. Yuan, C. Gao, W. Yuan, and X. Luo, "Black-box adversarial example attack towards {FCG} based android malware detection under incomplete feature information," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 1181–1198.
- [11] H. Bostani and V. Moonsamy, "Evadedroid: A practical evasion attack on machine learning for black-box android malware detection," *Computers & Security*, vol. 139, p. 103676, 2024.

- [12] W. Hu and Y. Tan, "Generating adversarial malware examples for black-box attacks based on gan," in *International Conference on Data Mining and Big Data*. Springer, 2022, pp. 409–423.
- [13] S. Chen, M. Xue, L. Fan, S. Hao, L. Xu, H. Zhu, and B. Li, "Automated poisoning attacks and defenses in malware detection systems: An adversarial machine learning approach," *computers & security*, vol. 73, pp. 326–344, 2018.
- [14] N. Bala, A. Ahmar, W. Li, F. Tovar, A. Battu, and P. Bambarkar, "Droidenemy: battling adversarial example attacks for android malware detection," *Digital communications and networks*, vol. 8, no. 6, pp. 1040–1047, 2022.
- [15] F. Pierazzi, F. Pendlebury, J. Cortellazzi, and L. Cavallaro, "Intriguing properties of adversarial ml attacks in the problem space," in *2020 IEEE symposium on security and privacy (SP)*. IEEE, 2020, pp. 1332–1349.
- [16] X. Chen, C. Li, D. Wang, S. Wen, J. Zhang, S. Nepal, Y. Xiang, and K. Ren, "Android hiv: A study of repackaging malware for evading machine-learning detection," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 987–1001, 2019.
- [17] J. Tang, S. Zhou, T. Peng, and W. Tian, "Dapadv: Differentiated adversarial perturbation generation method in problem space for android malware detection," *Available at SSRN 5052646*, 2024.
- [18] N. Carlini, A. Athalye, N. Papernot, W. Brendel, J. Rauber, D. Tsipras, I. Goodfellow, A. Madry, and A. Kurakin, "On evaluating adversarial robustness," *arXiv preprint arXiv:1902.06705*, 2019.
- [19] C. Smutz and A. Stavrou, "When a tree falls: Using diversity in ensemble classifiers to identify evasion in malware detectors," in *NDSS*, 2016.
- [20] D. Li and Q. Li, "Adversarial deep ensemble: Evasion attacks and defenses for malware detection," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3886–3900, 2020.
- [21] M. Ficco, "Malware analysis by combining multiple detectors and observation windows," *IEEE Transactions on Computers*, vol. 71, no. 6, pp. 1276–1290, 2021.
- [22] H. Li, S. Zhou, W. Yuan, X. Luo, C. Gao, and S. Chen, "Robust android malware detection against adversarial example attacks," in *Proceedings of the Web Conference 2021*, 2021, pp. 3603–3612.
- [23] D. Li, S. Cui, Y. Li, J. Xu, F. Xiao, and S. Xu, "Pad: Towards principled adversarial malware detection against evasion attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 2, pp. 920–936, 2023.
- [24] Y. Zhou, G. Cheng, Z. Chen, and S. Yu, "Malpurifier: Enhancing android malware detection with adversarial purification against evasion attacks," *arXiv preprint arXiv:2312.06423*, 2023.
- [25] D. Li, Q. Li, Y. Ye, and S. Xu, "A framework for enhancing deep neural networks against adversarial malware," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 1, pp. 736–750, 2021.
- [26] Y. Wu, X. Li, D. Zou, W. Yang, X. Zhang, and H. Jin, "Malscan: Fast market-wide mobile malware scanning by social-network centrality analysis," in *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 2019, pp. 139–150.
- [27] N. Šrđić and P. Laskov, "Practical evasion of a learning-based classifier: A case study," in *2014 IEEE symposium on security and privacy*. IEEE, 2014, pp. 197–211.
- [28] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 IEEE symposium on security and privacy (sp)*. IEEE, 2017, pp. 39–57.
- [29] O. Bryniarski, N. Hingun, P. Pachuca, V. Wang, and N. Carlini, "Evading adversarial example detection defenses with orthogonal projected gradient descent," *arXiv preprint arXiv:2106.15023*, 2021.
- [30] B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 2154–2156.
- [31] A. E. Cinà, F. Villani, M. Pintor, L. Schönherr, B. Biggio, and M. Pelillo, " σ -zero: Gradient-based optimization of ℓ_0 -norm adversarial examples," *arXiv preprint arXiv:2402.01879*, 2024.
- [32] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel, "On the (statistical) detection of adversarial examples," *arXiv preprint arXiv:1702.06280*, 2017.
- [33] T. Pang, C. Du, Y. Dong, and J. Zhu, "Towards robust detection of adversarial examples," *Advances in neural information processing systems*, vol. 31, 2018.
- [34] P. Sperl, C.-Y. Kao, P. Chen, X. Lei, and K. Böttinger, "Dla: dense-layer-analysis for adversarial example detection," in *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2020, pp. 198–215.
- [35] A. Madry, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.
- [36] D. Li, Q. Li, Y. Ye, and S. Xu, "Enhancing deep neural networks against adversarial malware examples," *arXiv preprint arXiv:2004.07919*, 2020.
- [37] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrđić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III 13*. Springer, 2013, pp. 387–402.
- [38] K. Zhao, H. Zhou, Y. Zhu, X. Zhan, K. Zhou, J. Li, L. Yu, W. Yuan, and X. Luo, "Structural attack against graph based android malware detection," in *Proceedings of the 2021 ACM SIGSAC conference on computer and communications security*, 2021, pp. 3218–3235.
- [39] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.
- [40] H. Li, S. Zhou, W. Yuan, J. Li, and H. Leung, "Adversarial-example attacks toward android malware detection system," *IEEE Systems Journal*, vol. 14, no. 1, pp. 653–656, 2019.
- [41] M. Shahpasand, L. Hamey, D. Vatsalan, and M. Xue, "Adversarial attacks on mobile malware detection," in *2019 IEEE 1st International Workshop on Artificial Intelligence for Mobile (AI4Mobile)*. IEEE, 2019, pp. 17–20.
- [42] D. Li, R. Baral, T. Li, H. Wang, Q. Li, and S. Xu, "Hashtran-dnn: A framework for enhancing robustness of deep neural networks against adversarial malware samples," *arXiv preprint arXiv:1809.06498*, 2018.



Mostafa Jafari completed his B.Sc. in Mechanical Engineering from K.N.Toosi University of Technology, followed by his M.Sc. in Software Engineering from Shahid Beheshti University (SBU) in Tehran, Iran. His current research interests revolve around machine learning, adversarial robustness and anomaly detection.



Alireza Shameli-Sendi is currently an Associate Professor at Shahid Beheshti University. Before joining SBU, he was a Postdoctoral Fellow at Ericsson, Canada and Postdoctoral at ETS and McGill universities in collaboration with Ericsson. He received his Ph.D. degree in computer engineering from Montreal University (Ecole Polytechnique de Montreal), Canada. He obtained his B.Sc. and M.Sc. from Amirkabir University of Technology. His primary research interests include cloud computing and network/information security. He is a recipient of

Postdoctoral Research Fellowship Award and Industrial Postdoctoral Fellowship Award from Canada. In addition, he received the best researcher awards, at SBU, in 2018 and 2022.