Improving Leakage Exploitability in Horizontal Side Channel Attacks through Anomaly Mitigation with Unsupervised Neural Networks

Gauthier Cler[1,2], Sebastien Ordas[2], and Philippe Maurine[1]

[1] University of Montpellier LIRMM, Montpellier, France
[2] SERMA Safety & Security ITSEF, Pessac, France

Abstract. The success of horizontal side-channel attacks depends heavily on the correct extraction of points of interest, which are expected to contain relevant leakages, and on the quality of the traces. If the latter is not sufficient, this will consequently degrade the identification of leakage candidates and often render attacks inapplicable. This work aims to assess the relevance of neural networks in the unsupervised context of horizontal attacks by proposing two methods with alternative objectives to mitigate noise artefacts from the input signal. Their application results in better traces exploitability when using clustering-based horizontal attacks.

Keywords: Horizontal Attacks · Side Channel Analysis · Neural Networks · Unsupervised Learning

1 Introduction

Horizontal side-channel attacks are a family of attacks which rely on the use of a single acquired trace for exploitation. As the presence of masking or randomization makes the use of multiple traces irrelevant, they are often sought as the last attack vector. However, their applicability and success capability heavily relies on the quality of the provided patterns (processed trace). As no profiling phase nor leakage assessment is available in this attack scenario, it can often be hard to improve the patterns quality by reducing noise or digital filtering, without the risk of leakage degradation.

On the other side, neural networks are widely used for profiled side channel attacks [17,15,6,24,3]. They aim at building a model that characterizes the leakage manifestation on an open device, which can then be applied to a target device during the attack phase. While this approach has been shown to be efficient, it requires knowledge of the labels associated with the traces/patterns to be applied. In fact, this is mandatory for network training based on an error objective, since their weights are updated based on the error between predicted and ground truth labels.

This rise of neural network architectures in the supervised side-channel analysis domain leads to the question of their applicability in an unsupervised context, and more precisely for horizontal attacks. While proposed supervised methods have shown competitive performance against more traditional methods for profiled attacks (such as template attacks [7]), their applicability in an unsupervised context is far from straightforward. In fact, training objective such as error-based optimization cannot be considered since no training data (thus no true labels) is available. Instead, alternative training mechanisms that are applicable in this context should be explored.

Within this context, this work provides several contributions to improve the quality of side-channel traces and thus their exploitability. Among them are:

– the identification and quantification of abnormal values in the input data and their impact on the selection of points of interest (PoI). This includes the presence of outliers or extreme values. These latter values result in truncated statistical distributions at the end of traces capture because of a specific setting, intentional or unintentional, of the digital sampling oscilloscope.
– the correction of anomalies through the application of a robust autoencoder (RAE) framework [25,14], in a completely unsupervised manner.
– their mitigation in a self-supervised manner using anomalies models, based on generative networks (GAN) [12].

Correcting outliers and extreme values can be beneficial for many side-channel analyses. However, it is expected to be of particular interest in the context of horizontal attacks. In fact, due to the randomization of the scalar or secret exponent, only a single trace can be analyzed at a time to try to recover the secret. Thus, any outlier can cause the loss of a scalar or exponent bit to be recovered. This explains why focus is made on this context in the rest of the paper, and more particularly on clustering-based horizontal attacks.

To that aim, the remaining of the paper is organized as follows. Section 2 recalls the context of clustering-based horizontal attacks and derives the way adopted to check the soundness of the proposals. Section 3 deals with the identification of abnormal values, which include outliers and extreme (truncated) values. Section 4 illustrates the effect of such values on univariate PoI selection. Section 5 highlights the limits of traditionally used anomalies mitigation, while sections 6.1 and 6.2 propose two correction methods using either a Robust Auto-Encoder (RAE) or a specific Cycle-Generative Adversarial Network (CycleGAN), respectively. Section 7 presents and discusses the results obtained on the publicly available datasets considered. Finally, the 8 section concludes the paper.

2 Clustering-based horizontal attacks and figures of merit

This section describes the working foreground that have been adopted for the development and evaluation of the proposed contributions. It first recalls the backbone of clustering-based horizontal attacks. Then, the figures of merit considered to evaluate the benefits of the proposed solutions to mitigate outliers and extremes values are given and discussed. Finally, it ends by presenting the two publicly available datasets that have been selected to evaluate the soundness of the proposals.

2.1 Clustering-based horizontal attacks

Horizontal attacks based on unsupervised clustering are typically carried out in two or three steps, with the third step being optional. Considering, for example, the proposals of Perin *et al.* in [21,20], these steps are:

- **Step 1 : Selecting points of interest (PoI)**: Usually, the selection of the PoI is done thanks to a univariate analysis of each time sample of aligned patterns (traces). There are several ways to analyze each time sample. Some proposals recommend analyzing their statistical distribution, assuming that PoIs are expected to be a balanced Gaussian mixture of two normal distributions [19,9]. Others like [21] suggest to cluster each time sample and to compute the DoM (Difference of Means), or extended but related figures of merit [20] between each cluster to identify PoI.

- **Step 2 : Guessing the scalar or exponent bits**: After identifying and ranking the PoIs, a clustering algorithm (e.g., k-means or expectation maximization) is applied to a subset of the PoIs to obtain an estimate of the secret scalar or exponent.

- **Step 3 : Correcting the guess**: As a possible last step, one could apply a correction phase to the estimation of the scalar or exponent bits, using a neural network as described in [20,4]. However, the efficiency of these networks depends significantly on the percentage of bits that are already correct in the estimate provided by the previous step.

Of course, in a supervised context, the first step could be performed in a guided manner thanks to the knowledge of the secret scalar or exponent bits. In this way, one could use the t-value as a leakage metric.

2.2 Figures of merit

As described above, clustering-based horizontal attacks are performed in two or three successive steps. The effect of outliers and extreme values is not expected to be known on each of this step, and furthermore could be cumulative. It is probably the case, since it is showed that the quality of traces can drastically affect the quality and efficiency of such attacks [4].

The evaluation of the soundness of techniques that allow to mitigate the effects of outliers and extremes could thus be performed according to different points of view.

From the point of view of the end application, the success rate of an attack could be a good measure of the efficiency of such techniques. In this paper, it makes use of the Bit Recovery Rate (BRR) [9], which corresponds to the percentage of secret bits successfully recovered through the clustering partitioning process. This metric is obviously not available during the attack but is used here in a study purpose.

From a pure leakage point of view, considering leakage assessment metrics such as t-values, either applied in a supervised or unsupervised manner, might be a more meaningful solution.

Global indicators are also considered for measuring the global changes in the patterns, namely the Signal to Noise ratio (SNR) and the Mutual Information (MI) between the patterns and associated labels.

As a last criterion for evaluation, the evolution of the number of values identified as outliers or extremes has been considered as a natural indicator to characterize the efficiency of the proposed mitigation process. As the mitigation is conducted in an entirely unsupervised manner, it is important to ascertain the extent to which the proposed approaches can correctly identify such values.

In the remainder of this paper, these three criteria will therefore be considered in turn. First, the capability of a technique to identify and correct outliers and extremes will be assessed by considering the percentage of outliers before and after mitigation. Second, the proposed method will be evaluated to determine whether it emphasizes the leakage manifestation by computing the t-values in a supervised manner. Third, the bit recovery rate (BRR) obtained with the attacks described in [21] and [9] is considered.

2.3 Considered targets

Two public datasets that target Curve25519 curve from μNaCl library on a STM32F4 are used for this study, namely Cswap Pointer and Arith from Nascimento and Chmielewski work [19]. These datasets are known to be at the same time leaky but very noisy as depicted by Fig. 1. It shows random patterns from both datasets. The presence of outliers and extreme observable values (-128 or 127) at many time points is clearly visible and supports the choice of these datasets as a suitable target to demonstrate the soundness of this work proposals. The choice of these datasets is also sustained by the fact that several papers have reported the success rates of different horizontal attacks applied to these datasets [20,9,4] . This is interesting for a comparison purpose. For reference, Fig. 2 shows the supervised t-value for the two considered datasets.

3 Identification and quantification of outliers

The identification of outliers and extreme values is an old statistical topic that could potentially be revisited depending on the end application. In this paper, two methods have been considered to identify such abnormal values.

Extreme values During traces acquisition, the oscilloscope's vertical resolution during digital sampling determines the number of values a signal can take. For

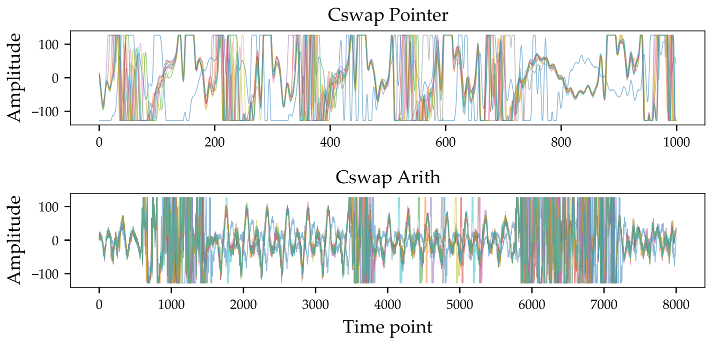

Fig. 1: Random patterns from Cswap Pointer and Cswap Arith datasets. [19]

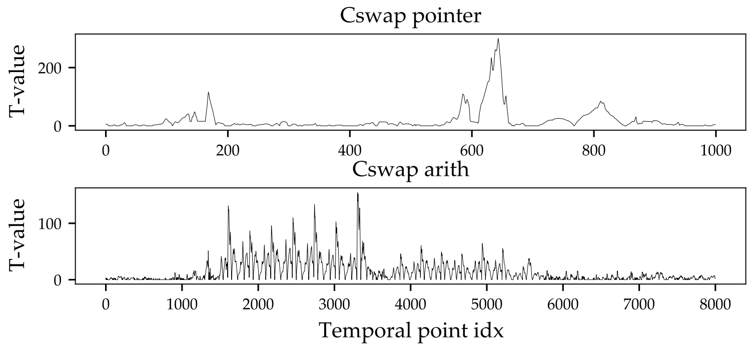

Fig. 2: t-value on each temporal point accross all patterns, for the two considered datasets.

example, a high-end oscilloscope with an 8-bit resolution can produce waveforms with a very large number of points, but all with values between -128 and 127. In case of a misconfiguration (intentional or not) of the acquisition hardware, the statistical distribution of some points could be truncated. This results in a large number of extreme values in the measurement of these points, which do not contain any valuable information. In this work, such values are called extremes. Considering signed values, saturated values for 8-bit and 12-bit resolution would respectively correspond to $\xi(8) = \{-128, 127\}$ and $\xi(12) = \{-2048, 2047\}$. For the rest of this work, only $\xi(8)$ is considered as the studied datasets are encoded on 8-bit. Thus a sample x would be flagged as saturated if $x \in \xi(8)$.

Interquartile range outliers Some noisy behaviors generated by the target or measurement environment can also introduce additional components into the distribution of time samples that can span the entire oscilloscope range up or down to extreme values. Values associated with these parasitic components are referred to as outliers in the remainder of this paper.

In the analysis of distributions, an outlier is defined as a value that is statistically significantly different from the others. If the interquartile range is considered (as in [19]), this corresponds to samples outside of the range $R = [Q_1 - \alpha\,\mathrm{IQR}, Q_3 + \alpha\,\mathrm{IQR}]$, where the $\mathrm{IQR} = Q_3 - Q_1$. Usually one considers $\alpha = 1.5$. Thus, all samples x with a high deviation from the quartiles such that $x \notin R$ are usually considered as outliers.

These definitions of extremes and outlier's models are interesting because they can be applied in an unsupervised context. Their application to the two datasets considered is thus straightforward. Fig. 3 shows the percentage of such values for each time point. As expected from a first glance at the traces, a significant number of time points show a high number of anomalies for both datasets. Some time points even show a percentage of detected anomalies (extreme values) close to 100%. This means that in the current state of trace quality, there is almost no exploitable leakage at these time points. Hence, the choice of these datasets to estimate the efficiency of any solution that allows to reduce outliers and extremes seems reasonable.

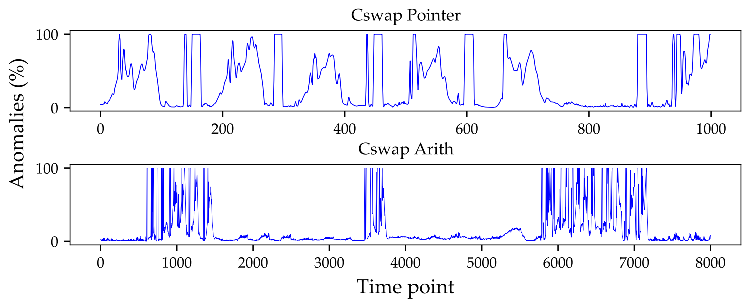

Fig. 3: Percentages of anomalies (outliers + saturated values) in Cswap Pointer and Arith datasets.

4 Impact of anomalies on the PoI selection process

When performing a clustering-based horizontal attack, the selection of the PoI (step 1) is done by applying a clustering algorithm to each time point of the aligned patterns and selecting some points highlighted by a distinguisher such as the DoM or the t-value. The relevance of the selected point is, of course, related to the quality of the traces. Indeed, the Gaussian noise, outliers and extreme values of the measurement could affect the efficiency of both the clustering algorithm and the distinguisher. Regarding outliers and extreme values, the impact could be huge, especially on the result provided by the clustering algorithm. Indeed, few outliers or extreme values could be sufficient to significantly shift the centroids and as a results bias the values provided by the distinguisher.

By way of illustration, Fig. 4 shows the effect of the extreme values on clustering results obtained with the k-means applied to sampled data from an univariate Gaussian mixture $p(x) = \pi_1\mathcal{N}(\mu_1, \sigma_1) + \pi_1\mathcal{N}(\mu_2, \sigma_2)$, with parameters

$\pi_1 = \pi_2 = 0.5$, $\mu_1 = 1$, $\mu_2 = 4$, $\sigma_1 = \sigma_1 = 0.5$. The x-axis indicates the percentage of values drawn from the underlying mixture that have been replaced by extreme values equal to 10. This process has been averaged over 10 experiments. As shown, the effect first remains moderated up to a certain value, in that case only 10%, above which there is a sudden and significant shift of the centroid values and of the associated DoM. Above this threshold, distance value involved in the k-means is mostly dominated by the presence of sufficient extreme values. In turn, this induced shift causes a rapid decrease in the bit recovery rate, i.e. the correctness of the guessed scalar or exponent bits. This trend (and more precisely the suddenness of the shift) clearly suggests that even correcting extreme values could be beneficial in some cases to exploit the leakage or bring back some time samples of traces in the exploitable domain.

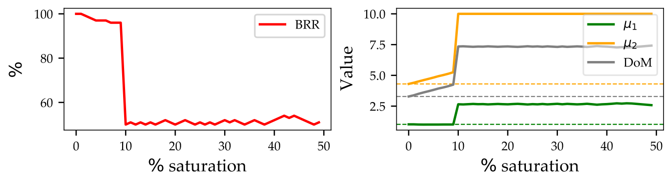

Fig. 4: Effect of extreme values on (Left): The BRR, (Right): the clustering centroid values and DoM.

In a similar fashion, the impact of outliers on the clustering capability has been studied. Considering the same data as before drawn from an univariate mixture distribution with same parameters, a percentage of random values from the distribution is replaced with $Q_3 + \alpha\,\mathrm{IQR}$. This is done with increasingly percentage and various α values. Results are showed on Fig. 5. Ones can notice the same behavior as in the previous experiment. Indeed, the BRR falls down towards 50% abruptly given a sufficient percentage of data on the right tail or alpha value. This shows that a sufficiently low α value should be used. In this work, $\alpha = 1.5$ is used.

Still, one may wonder if the presence of outliers and saturated values can negatively impact the PoI selection process on real data. Thus, some leaking points (in terms of high t-values) of the Cswap Pointer dataset have been corrupted. More precisely, some values of time samples with a significant leakage have been progressively replaced with extreme values the evolution of the BRR and centroids with respect to the percentage of corrupted values has been observed. Fig. 6 shows the results obtained considering only the first 5000 patterns of Cswap Pointer, for time points 640 and 168. A supervised leakage analysis using the t-test gave t-values of 300 (the highest value obtained across all points) and 116 respectively. The number of extreme values (either -128 or 127) before corruption was counted as 2 for both points that differs mainly by their mean and standard deviation. The figure clearly shows that a few extreme values (7%

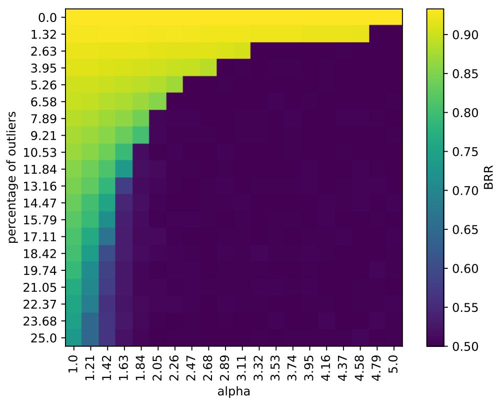

Fig. 5: Resulting BRR when replacing random values by right tail of the global distribution, with different percentage and alpha value.

and 2% respectively) are sufficient to completely deceive the k-means and cause a drastic reduction of the BRR from 92% and 70% respectively to values close to 50%. It seems that these few extreme values are indeed sufficient to significantly attract the centroids (at least one of the two) found by the k-means close to them. Hence, it is straightforward to see that noisy sets of traces as the ones considered for this study would be harder to exploit using a clustering based strategy. However, ones can wonder if the mitigation of such abnormal values would be beneficial for the exploitation.

5 Anomalies mitigation through ablation

As seen in the previous section, clustering process in the presence of saturated values or outliers can results in incorrect clusters estimation. From this, it would be necessary to take action to mitigate such values. Outliers mitigation is a well-studied topic in various statistical domains [10,1].

One proposal can be to replace identified anomalies values based on the values of other samples. For side channel analysis purpose in [19], considering the outliers model described above, authors proposed the following mitigation strategy. For each time point, it consists in replacing the value of identified outliers with the median of non-outliers on this time point. This could be applied as well for saturated values $x \in \xi(8)$. This can be a relevant approach if the percentage of anomalies is really low, but it has however no (or very low) beneficial effect on the quality of the PoI selection procedure in case of highly noisy context. Furthermore, this can have the consequence to degrade the classes discrimination through clustering, as the formed distribution tends towards an unimodal Gaussian distribution as the number of outliers increases.

Another straightforward approach can be to remove time points if the percentage of anomalies across all patterns is higher than a defined threshold. This,

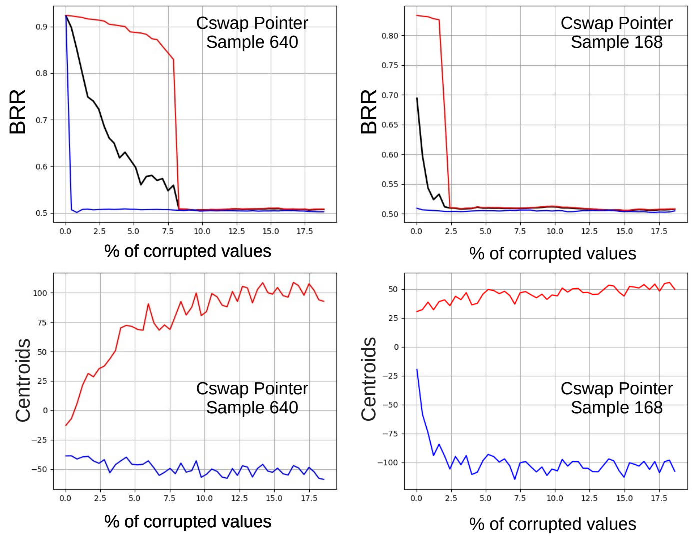

Fig. 6: Top: $Q_{0.99}$ (red), mean (black) and $Q_{0.01}$ (blue) of the BRR obtained by applying 100 times the k-means to time samples 640 and 168 from Cswap Pointer patterns. Bottom: Evolution of the centroid's values provided by the k-means applied to time samples 640 and 168.

however, can has a doubled-edged effect. Firstly, this can result in the removal of leaking time points depending on the arbitrary chosen threshold, thus in a decrease of leakage information in the patterns. Furthermore, in cases with high amount of anomalies (as seen on considered datasets in this work), this would have the effect of removing the majority of time points. For reference on Cswap Pointer and Arith datasets, a threshold above 5% of anomalies, would result in 34.8% and 53.65% respectively of time points removed.

From this status, the motivation behind this work lies as follow. Using a neural network unsupervised approach, it hopes to significantly increase the exploitability of the leakage present in some time samples by correcting outliers or extremes they may contain. Additionally, as the correction process is unsupervised, it should not have a degradation effect on the leakage.

6 Neural based anomalies mitigation

This sections aims at studying neural architectures candidate that could be relevant for anomalies mitigation in a horizontal attack context. To do so, said architecture would require to be trained in an unsupervised manner (or semi-supervised), without knowing the associated value to each pattern. For this work, two architectures have been studied, which training is based on alternative objective, applicable in the context of horizontal attacks. These methods are applied on the whole patterns, without discarding any time points.

6.1 Unsupervised mitigation with RAE

The first approach considered in this work to mitigate (or correct) outliers and extremes is an unsupervised approach. It consists of a variation of the autoencoder architecture. While the use of autoencoders for multivariate leakage extraction can give relevant results in noiseless (or low noise) contexts (as well as projections methods such as the Principal Component Analysis), it often fails in noisy ones [8]. Instead, a variant of the vanilla autoencoder, known as the robust autoencoder (RAE) [25,14] is considered in this paper. The RAE can be seen as non-linear generalization of the Robust Principal Component Analysis (RPCA) [5], trained using gradient descent. The idea behind the RPCA/RAE is to recover a low rank matrix L from corrupted matrix X and extract artifacts into an sparse noise matrix S. It can be described by the following decomposition:

$$X = L + S \tag{1}$$

where each term in this study context corresponds to:

- X, the input matrix, containing noisy artefacts, corresponding to the patterns obtained after the traces acquisition and pre-processing (verticalization and alignment procedure).
- L, the clean matrix, which should be free of abnormal values after the anomalies mitigation process,
- S, the noise matrix, from the difference $S = X - L$, which is a sparse matrix containing the positions of the extracted noise artefacts.

Rather than attempting to project relevant information into a low-dimensional subspace and thus focusing on its compression capability as traditional autoencoder would do, the RAE is used to mitigate anomalies in their reconstruction phase with an objective function consisting in two terms:

$$\mathcal{L}(\theta, \phi) = ||L - \mathcal{F}_\theta(\mathcal{E}_\phi(L))||_2 + \tau ||S||_1 \tag{2}$$

where the left term is the autoencoder reconstruction error optimized by gradient descent (with $\mathcal{E}_\theta$ and $\mathcal{F}_\phi$ the encoder and decoder networks respectively) and the right term is the proximal optimization of S. The proximal optimization of the ℓ_1 norm corresponds to the shrinkage operator:

$$\text{prox}_{\ell_1}(x, \tau) = \begin{cases} x - \tau & \text{if } x > \tau \\ x + \tau & \text{if } x < -\tau \\ x & \text{if } x \in [-\tau, \tau] \end{cases} \tag{3}$$

where the parameter τ defines the level of sparsity of S such that small τ will encourage more data to be identified as noise or outliers.

Since the combined objective of Eq. 2 cannot be optimized as a whole, it should be split into two sub-objectives that can be optimized by the alternating direction method of multipliers (ADMM) algorithm. It consists of alternately optimizing one sub-objective while keeping the other fixed. This leads to the following ADMM optimization procedure for the RAE.

1. Train the autoencoder to minimize the reconstruction error of $L = X - S$ using gradient descent (left term of Eq. 1).
2. Derive the noisy matrix $S = X - L$.
3. Minimize S by proximal optimization (right term of Eq. 1).

These three steps are repeated until convergence is reached, which is defined as the point at which no change is observed in the matrices. Details about network architecture and parameters are described in Appendix A. It is important to note that the mitigation process can have an impact on non-abnormal points. Indeed, as the RAE produces synthetic patterns during the reconstruction process, which are then used as new patterns, it can affect all values of all time points. In a side channel context, this could thus affect the observed leakage levels. This method was applied to the patterns of the two datasets considered. An identical network architecture was used for both datasets. The analysis of results that were obtained considering the cleaned matrix L are presented in Section 7.

6.2 Self-supervised mitigation with CycleGAN

As an alternative approach to mitigate outliers and extremes values, the application of a self-supervised framework may be a more interesting and controllable solution. Indeed, as described in section 3, it is possible to identify abnormal values in a completely unsupervised manner. Thus, it allows to control the learning process of the neural network used for anomalies mitigation to ensure that only the identified abnormal values are corrected, while the remaining data points remain unchanged. This approach thus differs from the one considered by the use of the RAE described previously. To that aim, an approach based on the CycleGAN architecture that trains on pairwise sets of data is therefore proposed. Such architecture is made up from a pair of two GAN, [12], that is two generators G_A, G_B and two discriminators D_A, D_B. GAN models are optimized through adversarial training, where a discriminator D aims at distinguish training samples as real or fake, while a generator G tries to fool the discriminator by generating convincing examples. The CycleGAN architecture has been widely applied for cross domains images translation [26]. The objective function of the CycleGAN is the following:

$$\begin{aligned}\mathcal{L}(G_A, G_B, D_A, D_B) = {} & \mathcal{L}_{\text{GAN}}(G_A, D_B, B, A) \\ & + \mathcal{L}_{\text{GAN}}(G_B, D_A, A, B) \\ & + \lambda \mathcal{L}_{\text{cyc}}(G_A, G_B)\end{aligned} \tag{4}$$

where each GAN objective, $\mathcal{L}_{\text{GAN}}$, is defined by:

$$\begin{aligned}\min_G \max_D \mathcal{L}_{\text{GAN}}(G, D, X, Z) = {} & \mathbb{E}_{x\sim X} \log D(x) \\ & + \mathbb{E}_{z\sim Z} \log[1 - D(G(z))]\end{aligned} \tag{5}$$

In addition, a consistency loss, $\mathcal{L}_{\text{cyc}}(G_A, G_B)$, is added to ensure mapping bijections between domains:

$$\begin{aligned}\mathcal{L}_{\text{cyc}}(G_A, G_B) = {} & \mathbb{E}_{a\sim A} ||G_B(G_A(a)) - a||_1 \\ & + \mathbb{E}_{b\sim B} ||G_A(G_B(b)) - b||_1\end{aligned} \tag{6}$$

with λ used to balance terms in Eq. 4. The proposed architecture for mitigating abnormal values differs from existing approaches in several ways. First, as a single dataset $X \in \mathcal{M}_{n,p}(\mathbb{R})$, ($n$ patterns of p time points) is available for horizontal attacks, it thus should be split into two subsets, designated as A and B. These subsets are constructed to contain samples with the greatest disparity in abnormal values. This is done by first computing the binary anomalies matrix $M \in \mathcal{M}_{n,p}(\{0,1\})$ such that each matrix element is:

$$m_{i,j} = \begin{cases} 1, & \text{if } x_{i,j} \in \xi(8) \vee x_{i,j} \notin R(x_{:,j}) \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

To recall, R corresponds to the interquantile range as defined in Sec. 3 Then, each pattern pair (a_i, b_i) is built such that it maximizes the Hamming distance between their abnormal values:

$$\underset{i,j \in \{1,\ldots,n\}}{\arg\max} \ \mathrm{HW}(m_{i,:} \oplus m_{j,:}), \quad i \neq j \tag{8}$$

Second, multiplexers (mux) are added between the output of generators and the input of discriminators. Multiplexers are defined as follows:

$$\begin{aligned} A'' &= \mathrm{mux}(A, G_B(B), M_A) = (M_A \wedge G_B(B)) \vee (\neg M_A \wedge A) \\ B'' &= \mathrm{mux}(B, G_A(A), M_B) = (M_B \wedge G_A(A)) \vee (\neg M_B \wedge B) \end{aligned} \tag{9}$$

where matrices M_A and M_B contains the positions of the anomalies (both outliers and saturated values) for subsets A and B. This does not affect the gradients propagation during the backward phase. The idea behind the application of the CycleGAN to outliers mitigation is to correct identified abnormal values in A from non outliers elements from B. This results in the fact that only samples in the original patterns flagged as abnormal are modified during training. For the optimization, this ensures that the discriminators D_A and D_B can train and backward propagate on the multiplexer outputs (rather than the whole synthetic patterns from the generators) to provide better judging capabilities. In parallel, the generators are specifically trained to generate convincing synthetic corrected points inplace of the anormal ones. The complete proposed architecture for the CycleGAN framework is shown in Fig. 7. A cycle of the mitigation process is then given as follows:

1. Calculate the anomalies matrix M (Eq. 7).
2. Split the original input data into two subsets A and B with highest Hamming distance between pairs of rows of M (Eq. 8).
3. Train the CycleGAN until convergence (Eq. 4).
4. Replace anomalies in initial patterns with generated points through the multiplexers (A'' and B'').

These steps are repeated until $||M||_0$ converges, that is when the number of anomalies reach zero or no changes are observed. Details of the CycleGAN architecture and optimization process are given in Appendix B. The results of application of this framework are given in the next section, where its performance is compared to the RAE correction capability.

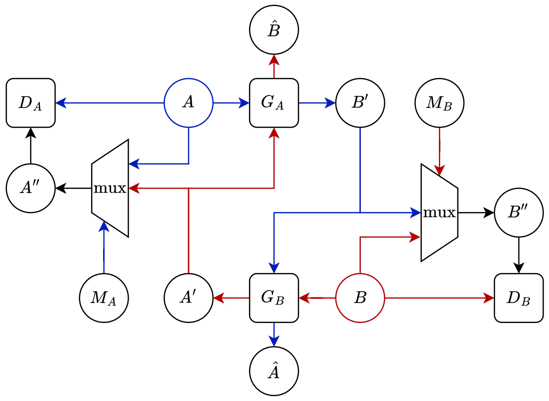

Fig. 7: Proposed CycleGAN architecture with multiplexers. Blue and red colors show data paths through the architecture.

7 Results & Discussion

In order to assess the efficiency of the proposed neural methods in mitigating outliers, several metrics were discussed and described in section 2. This section presents the results obtained for the two proposed methods, applied on the considered datasets.

7.1 Percentage of corrected outliers and extremes

Table 1 presents the percentage of anomalies identified using the IQR and extreme values methods before and after the application of the two proposed mitigation approaches.

	Cswap Pointer			Cswap Arith		
	Before	RAE	GAN	Before	RAE	GAN
Outliers (%)	4.32	5.36	**1.67**	5.15	4.73	**1.35**
Extreme (%)	30.27	**1.19**	10.22	12.88	**0.01**	5.19
Total (%)	33.39	**6.55**	11.85	16.54	**4.75**	6.49

Table 1: Percentage of outliers and extremes obtained on original patterns, after applying the RAE and the CycleGAN. Best results are highlighted in bold.

On the two datasets examined, the CycleGAN shows better outlier mitigation, while the RAE performs better at correcting extremes. Overall, the RAE shows slightly better results for total anomaly correction. However, the latter has a greater impact on the extreme than on the outliers in the mitigation process and changes all values of the patterns, which could be risky and alter the leakage.

7.2 Leakage assessment on time samples

Successful mitigation can be translated to no degradation in the observed leakage levels. In order to assess the capability of the considered approaches using RAE and CycleGAN, the Supervised t-test, as well as clustering using k-means and computing the BRR percentage is done. Fig. 8 displays the difference values obtained for each time point before and after application of the CycleGAN and RAE, in a *after minus before* fashion. In other words, positive values translates to improvement of the t-values (respectively BRR), while negative values means that the time point leakage exploitability has been degraded. For many time

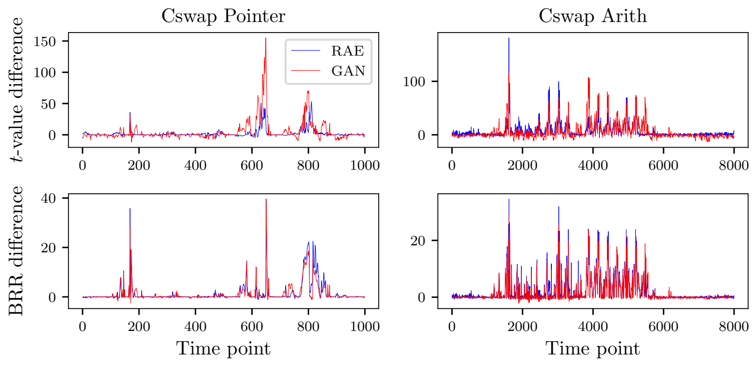

Fig. 8: Metrics difference before and after application of the RAE and CycleGAN. Top: t-values. Bottom: BRR. Positive values correspond to improvement, negative values to degradation.

points, the mitigation process produces an improvement, which means that the correction of anomalies allows a better partitioning of classes by the clustering process. Additionally, some new areas show high BRR values while they were close to 50% before the mitigation process. This is especially visible for the area between samples 3500 and 5500 of the Arith dataset. However, these observations do not allow us to conclude that the information contained in these time points has increased or is simply more exploitable, which is more likely. This point is discussed at the end of this section.

It also interesting to observe the distributions of time samples whose BRR has been drastically increased by the mitigation process. Fig. 9 shows the empirical p.d.f before and after application of the RAE and CycleGAN of time samples 650 and 168 of Pointer dataset and for time samples 1617 and 3028 of Arith dataset. The changes in the distribution caused by the application of the CycleGAN is not visible to the eye. This is to be expected, since only a few percentages of the points (outliers and extremes values) are corrected by the CycleGAN because of the multiplexers. Other points are left unchanged, thus preserving the overall shape and moments of the distributions.

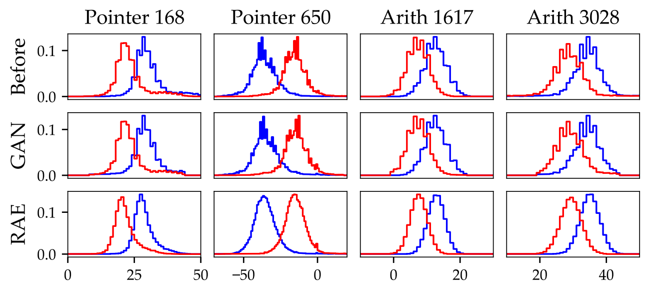

Fig. 9: Empirical p.d.f of four samples before and after application of the RAE and CycleGAN to mitigate abnormal values. Blue p.d.f corresponds to class $c = 0$ (resp. red $c = 1$).

On the other hand, generated samples from the RAE seems to produce smoother mixture of normal distributions. This is an expected behavior as the RAE decomposition produces a completely new and thus changed patterns without any guarantee that the moments and the shape of the distributions (and thus the leakage) are preserved.

Overall, both methods allow for leakage exploitation through the clustering process without significant degradation of the input patterns. While unsupervised, these approaches seem to preserve the leakage and seem thus applicable in the context of unsupervised horizontal attacks.

7.3 Success rate of attacks

To estimate the effect on the end application, the success rates of different attacks have been compared.

The first attacks applied, before and after outlier and extreme reduction, consisted of *supervised* selection and ranking of PoIs with the t-test and then applying the k-means. Thus, the set of selected PoIs can be assumed to be close to the optimal one. Fig. 10 shows the BRR versus the number of selected PoIs ranked according to the t-values. The application of RAE and GAN led to a slight increase in the BRR obtained on Cswap Pointer and Arith when the number of selected PoIs is below 50 and 100 respectively, because these points are only slightly affected by outliers. Above these numbers, the gain in BRR becomes more significant and is around 30% when the number of selected PoI remains below 150 and 200 respectively. This is an indication of the soundness of the proposed mitigation techniques, which delay the fall of the BRR by including more noisy and leaking points in the exploitable domain. The second attack considered is the unsupervised one proposed in [21]. The mitigation of outliers and extremes with the RAE and CycleGAN was applied either before the PoI selection (step 1) or immediately afterwards. In both cases, no improvement in the success rate of the attack was observed, regardless of the number of selected PoIs. It remained around 52%. Further analysis of the results showed

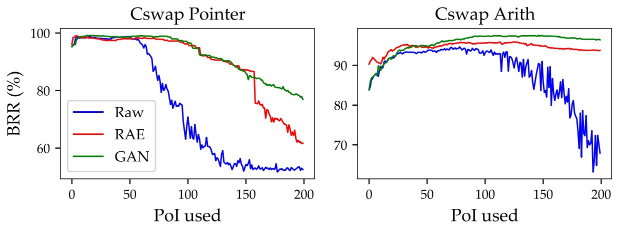

Fig. 10: BRR vs number of PoI considered when the mitigation of outliers is applied before PoIs selection and the selection and ranking of PoIs are done in a supervised manner.

that the effects of the mitigations were insufficient to significantly change the set of selected PoIs, which is not correct when using the DoM as a distinguisher on these trace sets. This was not a surprising result, as the number of points corrected with the RAE or CycleGAN is too limited to significantly change the DoM values.

Finally, the attack proposed in [9] has been considered. Similarly to what have been done for the attack reported in [21], the RAE or CycleGAN have been applied either before or after the selection of PoIs, thus considering a new set of PoI or the same than those found on raw traces.

Fig. 11 shows the success rate of the attack when the mitigation is done before the selection of PoIs thus with a new set of PoI. The effect of the mitigation of outliers obtained on Pointer dataset is limited, in average, to a slight increase of the success rate. For Arith, the same conclusion can be drawn except that the BRR values falls down to values close to 50% when more than 80 PoI, instead of 40, are considered. This indicates that some points that were unexploitable are now exploitable. The effect of mitigation would have been more valuable on a dataset where the first 40 PoIs were not present. This is not the case here but one could concede this could happen. Fig. 12 shows the success rate when the

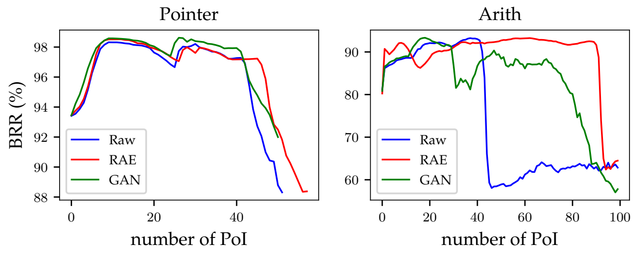

Fig. 11: BRR vs number of PoI considered when the mitigation of outliers is applied before PoIs selection and ranking with the method described in [9].

mitigation is applied after the selection of PoIs, i.e. considering the set of PoI found before mitigation. For Pointer datasets, there is no significant change in

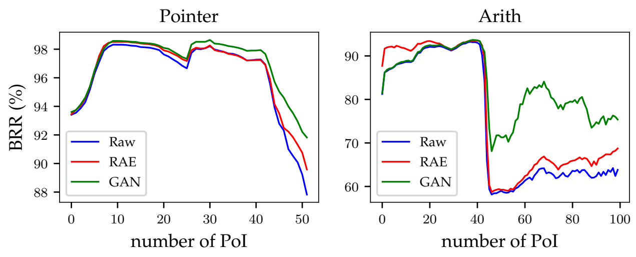

Fig. 12: BRR vs number of PoI considered when the mitigation of outliers is applied after PoIs selection and ranking with the method described in [9].

the results. Only a slight increase of the BRR is observed. For Cswap Arith, a similar trend can be observed, where PoIs with indexes 50 to 100 are more exploitable than before. However, these points are not the same than in previous the figure. This gives an indication that the mitigation of outliers and extremes was not sufficient to converge toward the best set of PoI which can be assumed to be one found in a supervised way.

Overall, the effects of mitigating outliers and extremes on the BRR of the supervised attack considered are significant, but are limited once PoI selection is done in an unsupervised manner (at least with the methods considered). This suggests that the proposed methods for reducing outliers are sound and that there is significant room for improvement in the way PoIs are selected, which appears to be the main limiting factor in the success rate of the attacks.

When comparing both approaches, it appears that mitigation before the PoI selection can produce more unreliable results during the PoI selection than applying the mitigation after a prior PoI selection. While identified PoI on raw patterns and after mitigation are almost the same (the overlap of their indexes is big), in some cases, new PoI which carry leakage are also identified if the selection is applied after the mitigation, especially matching indexes from Fig. 8. These side-effects from the unsupervised mitigation can affect the attack (positively or negatively) as showed on Fig. 11. Hence for these datasets a more robust strategy would be to firstly identify relevant PoI (even they are noisy) and then apply the cleaning process while keeping these PoI for the attack phase. Another possible strategy could also be for example to perform two attacks on both cases (mitigation before or after PoI selection) and combine the results. It is however necessary to apply the mitigation process on the whole patterns (and not only selected PoI) as relevant information can be also extracted from non identified PoI and neighboring points.

Finally, the overall attack complexity and remaining necessary exhaust after classes partitioning will depend on the used attack strategy. In this work, effort is put at showing the benefit of anomalies correction on the resulting leakage and exploitability rather than complete attack scheme. Furthermore, based on the successfully recovered part of the scalar, the effort for complete recovery will also depends on the exhaust strategy (see [23,18]). That is why only the BRR is considered as an indicator of the overall attack success. This allows to assess the relevance of proposed methods and is easily comparable between approaches.

Still, at this stage, one may wonder whether the mitigation preserves or restores the leakage, or simply limits the effect of outliers on the clustering process, thus allowing a more accurate estimation of centroids. This point is addressed in the next paragraphs.

7.4 Leakage recovery discussion

In order to obtain evidence on the ability of CycleGAN and RAE to restore (or preserve) the leakage when correcting an anomaly, the BRR values considering only the modified points by the networks have been extracted. This is depicted on 13. Using CycleGAN, only a few ($\leq 10\%$) of the points are corrected, while for RAE most of the points are modified, even if the changes are small. As can

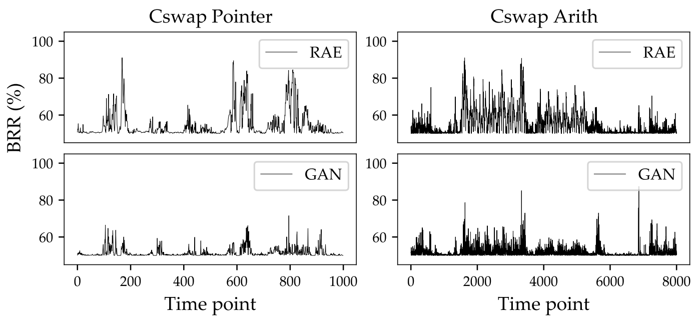

Fig. 13: BRR obtained for points of Cswap Arith whose values have been changed by the RAE or CycleGAN, on the two considered datasets.

be seen for Cswap Arith, in the leaky part of the patterns (time samples 1500 to 6000), many (between 55% and 80%) of the corrected time samples are classified in the correct cluster by the k-means after applying the RAE. The same can be observed with the CycleGAN, although the percentage of correctly classified bits is lower (between 55% and 70%). There are two reasons that could explain these results. Firstly, it is surprising but possible that both RAE and CycleGAN are able to recover some of the leakage thanks to the multidimensional treatment of the patterns. Secondly, the reduction of outliers and extremes limits their impact on the values of the centroids, allowing a better classification of all points.

The values of the two centroids before and after mitigation were therefore checked. Fig. 14 shows the absolute differences between the centroids provided by the k-means applied to the raw patterns and after mitigation of the outliers with

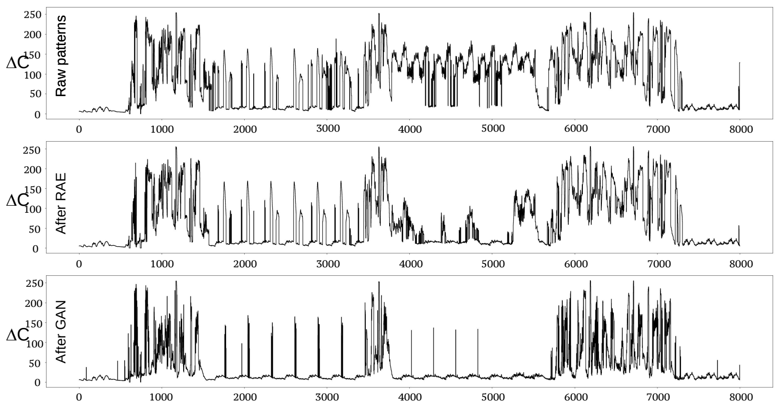

Fig. 14: Absolute differences between the centroids provided by the k-means without any mitigation of outliers and after application of the RAE or the CycleGAN, on the Cswap Arith dataset.

the RAE and the CycleGAN. Results are given only for Cswap Arith dataset, since the results for Cswap Pointer are similar. From the analysis of Fig. 8, it appears that the time samples where the BRR has significantly increased (time samples indexes 1500 to 6000) due to the reduction of outliers and extremes coincide with the time samples where the centroids have been significantly shifted (corrected) by the RAE or CycleGAN. Thus, the BRR improvements appear to be at least partially, perhaps entirely, due to centroid correction, i.e., limiting the impact of outliers on centroid estimates. If the gains in BRR were entirely due to the correction of centroids, this would suggest that there is no recovery of leakage.

7.5 Denoising and information recovery

Still, one would wonder what is the impact of the proposed mitigation processes on the patterns noise levels and underlying information. It is crucial to know if the proposed framework globally reduces the leakage information or not. To this aim, the global Signal-to-noise ratio (SNR) [22] has been computed before and after application of the considered methods. Results are showed in Table 2. An improvement over the measured SNR can be observed through the mitigation process. Indeed, the proposed methods allows to increase the SNR on Cswap Pointer by a factor of 1.7 using the CycleGAN, and by a factor of 2.3 on Cswap Arith using the RAE. This supports that the anomalies mitigation has a denoising effect on the treated patterns.

Finally, to decide on whether information was recovered or not, the mutual information (MI) between the correct scalar labels and the complete patterns was computed using MINE, a linearly scalable neural mutual information estimator [2,11] for high dimensionality estimation. Before the anomalies correction, the

Method / Dataset	Raw patterns	After RAE	After GAN
Cswap Pointer	0.030	0.037	**0.052**
Cswap Arith	0.009	**0.021**	0.020

Table 2: Signal to noise ratio. Best results are highlighted in bold.

information contained in Cswap Arith (resp. Pointer) was estimated to be close to 0.923 bits (resp. 0.933 bits). After correction with the RAE and CycleGAN, the information contained in Cswap Arith is now estimated to be 0.943 bits and 0.949 bits, respectively. For Pointer, it is about 0.922 bits and 0.936 bits. The MI values between the *complete* patterns and the secret bits were thus very little changed by the anomalies reduction process. This supports the claim that no information has been recovered, but that the better estimation of the centroids allows more information (that was already present in the raw patterns) to be correctly exploited.

However, MINE was also used to estimate the MI between labels and sub-parts of the patterns. The sub-parts of the patterns consisted in 100 bins, thus blocks of 10 and 80 consecutive time points for Cswap Pointer and Cswap Arith, respectively. Fig. 15 shows the difference in *local* MI before and after application of RAE or CycleGAN. Increases (or decreases) in MI are depicted in green (or red). Several increases in local MI of up to 0.3 bits for Arith and up to 0.2 bits for Cswap Pointer can be observed when applying the RAE. This corresponds to increases of about 20% of the *local* MI. The absolute MI gains are lower when the CycleGAN is used. Given that no *global* increase in MI was induced by the

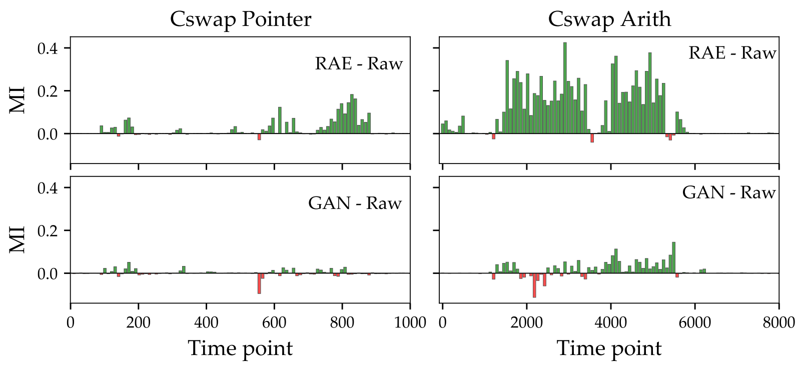

Fig. 15: Differences between local MI obtained after and before application of the RAE and CycleGAN to Pointer and Arith datasets.

application of either RAE or CycleGAN, and that significant increases in *local* MI were observed, it is suggested through this observation that both RAE and CycleGAN transpose some of the information contained in outlier-free parts of the patterns into altered subparts during the anomalies correction process, and that this transposition brings back some time points with too many outliers or extremes values into the exploitable domain for the attacks. This helps the k-means to provide better estimates of centroids and thus slight increases of the BRR obtained through the clustering process.

8 Conclusion

The presence of outliers and extremes values in side-channel datasets can seriously reduce the efficiency of attacks, especially horizontal attacks, as well as leakage assessment methods. Classical methods are often used to mitigate the impact of outliers in an unsupervised way, although they are insufficient in high noise contexts.

In this paper, the potential of deep learning based (and thus multidimensional) methods have been investigated to mitigate the effects of anomalous values. To this aim, two candidates has showed to be efficient architectures for unsupervised anomalies mitigation. The first relevant candidate is a robust autoencoder (RAE) that can be applied in a fully unsupervised manner. The only a prior required is the percentage of abnormal values, which can be easily estimated. The second candidate is a modified generative model (CycleGAN), which can also be applied in a fully unsupervised way, as its multiplexers only requires the positioning of anomalies values. These methods have been applied to two noisy public datasets considered for horizontal attacks. It has been found that some time samples that were not exploitable for attacks before mitigation became exploitable after application of RAE and CycleGAN, probably caused by a transposition of leakage from outlier-free parts of patterns to corrected parts of patterns. Furthermore, their application showed improvement over the Signal-to-noise ratio and no degradation of the underlying leakage information, quite the opposite. While it is possible that the process of applying the two methods subsequently could be beneficial in some cases, on the considered datasets, the mitigation using either the RAE or CycleGAN impact the same leakage areas and thus would not be seen as an added value if combined. Furthermore, If done, it would result in overly complexity for choosing appropriate architecture and parameters tuning of both methods for the application in the context of horizontal attack. Furthermore, some side-effect could arise from application of the two methods together. Instead, some benefit could be seen if the two methods would be applied separately on the same data and their results used for further analysis.

Finally, It should also be noted that no special effort has been made to optimize the network architectures used or to tune their parameters. Therefore, there is still room for improvement.

Finally, one should note that although the application of these techniques led to an increase in the exploitability of the leakage for the datasets studied, the result could vary and dependent on the used datasets, the considered outliers or noise model as well as their associated quantities. Indeed, it would be relevant for future work to consider alternative anomalies models than the ones used in this work.

References

1. Bakker, M., Wicherts, J.M.: Outlier removal, sum scores, and the inflation of the type i error rate in independent samples t tests: The power of alternatives and recommendations. Psychological Methods **19**(3), 409–427 (2014). https://doi.org/10.1037/met0000014
2. Belghazi, I., Rajeswar, S., Baratin, A., Hjelm, R.D., Courville, A.C.: MINE: mutual information neural estimation. CoRR **abs/1801.04062** (2018), http://arxiv.org/abs/1801.04062
3. Benadjila, R., Prouff, E., Strullu, R., Cagli, E., Dumas, C.: Deep learning for side-channel analysis and introduction to ascad database. Journal of Cryptographic Engineering **10**(2), 163–188 (Jun 2020). https://doi.org/10.1007/s13389-019-00220-8, https://doi.org/10.1007/s13389-019-00220-8
4. Boussam, S., Albillos, N.C.: Keep it unsupervised: Horizontal attacks meet simple classifiers. In: Bhasin, S., Roche, T. (eds.) Smart Card Research and Advanced Applications. p. 213–234. Springer Nature Switzerland, Cham (2024). https://doi.org/10.1007/978-3-031-54409-5_11
5. Candès, E.J., Li, X., Ma, Y., Wright, J.: Robust principal component analysis? J. ACM **58**(3) (Jun 2011). https://doi.org/10.1145/1970392.1970395, https://doi.org/10.1145/1970392.1970395
6. Carbone, M., Conin, V., Cornélie, M.A., Dassance, F., Dufresne, G., Dumas, C., Prouff, E., Venelli, A.: Deep learning to evaluate secure rsa implementations. IACR Transactions on Cryptographic Hardware and Embedded Systems p. 132–161 (Feb 2019). https://doi.org/10.13154/tches.v2019.i2.132-161, https://tches.iacr.org/index.php/TCHES/article/view/7388
7. Chari, S., Rao, J.R., Rohatgi, P.: Template attacks. In: Kaliski, B.S., Koç, c.K., Paar, C. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2002. p. 13–28. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (2003). https://doi.org/10.1007/3-540-36400-5_3
8. Cler, G.: Horizontal Side Channel Attacks on Noisy Traces. Theses, Université de Montpellier (Jul 2024), https://theses.hal.science/tel-04730413
9. Cler, G., Ordas, S., Maurine, P.: Bernoulli at the root of horizontal side channel attacks. In: Bhasin, S., Roche, T. (eds.) Smart Card Research and Advanced Applications. p. 107–126. Springer Nature Switzerland, Cham (2024). https://doi.org/10.1007/978-3-031-54409-5_6
10. Cowell, F.A., Victoria-Feser, M.P.: Robustness properties of inequality measures. Econometrica **64**(1), 77–101 (1996). https://doi.org/10.2307/2171925, https://www.jstor.org/stable/2171925
11. Cristiani, V., Lecomte, M., Maurine, P.: Leakage assessment through neural estimation of the mutual information. In: Zhou, J., Conti, M., Ahmed, C.M., Au, M.H., Batina, L., Li, Z., Lin, J., Losiouk, E., Luo, B., Majumdar, S., Meng, W., Ochoa, M., Picek, S., Portokalidis, G., Wang, C., Zhang, K. (eds.) Applied Cryptography and Network Security Workshops - ACNS 2020 Satellite Workshops, AIBlock, AIHWS, AIoTS, Cloud S&P, SCI, SecMT, and SiMLA, Rome, Italy, October 19-22, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12418, pp. 144–162. Springer (2020)

12. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial networks (arXiv:1406.2661) (Jun 2014). https://doi.org/10.48550/arXiv.1406.2661, http://arxiv.org/abs/1406.2661, arXiv:1406.2661 [cs, stat]
13. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks (2018), https://arxiv.org/abs/1611.07004
14. Kieu, T., Yang, B., Guo, C., Jensen, C.S., Zhao, Y., Huang, F., Zheng, K.: Robust and explainable autoencoders for unsupervised time series outlier detection—extended version (arXiv:2204.03341) (Apr 2022). https://doi.org/10.48550/arXiv.2204.03341, http://arxiv.org/abs/2204.03341, arXiv:2204.03341 [cs]
15. Kim, J., Picek, S., Heuser, A., Bhasin, S., Hanjalic, A.: Make some noise. unleashing the power of convolutional neural networks for profiled side-channel analysis. IACR Transactions on Cryptographic Hardware and Embedded Systems p. 148–179 (May 2019). https://doi.org/10.13154/tches.v2019.i3.148-179, https://tches.iacr.org/index.php/TCHES/article/view/8292
16. Mao, X., Li, Q., Xie, H., Lau, R.Y.K., Wang, Z., Smolley, S.P.: Least squares generative adversarial networks (2017), https://arxiv.org/abs/1611.04076
17. Masure, L., Dumas, C., Prouff, E.: Gradient visualization for general characterization in profiling attacks. In: Polian, I., Stöttinger, M. (eds.) Constructive Side-Channel Analysis and Secure Design. p. 145–167. Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-16350-1_9
18. Micheli, G.D., Heninger, N.: Recovering cryptographic keys from partial information, by example. Cryptology ePrint Archive, Paper 2020/1506 (2020), https://eprint.iacr.org/2020/1506
19. Nascimento, E., Chmielewski, L.: Applying horizontal clustering side-channel attacks on embedded ecc implementations (extended version) p. 23
20. Perin, G., Chmielewski, L., Batina, L., Picek, S.: Keep it unsupervised: Horizontal attacks meet deep learning. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2021**(1), 343–372 (2021)
21. Perin, G., Imbert, L., Torres, L., Maurine, P.: Attacking randomized exponentiations using unsupervised learning. vol. LNCS, p. 144–160 (Apr 2014). https://doi.org/10.1007/978-3-319-10175-0_11, https://hal-lirmm.ccsd.cnrs.fr/lirmm-01096039
22. Prouff, E., Strullu, R., Benadjila, R., Cagli, E., Dumas, C.: Study of deep learning techniques for side-channel analysis and introduction to ASCAD database. Cryptology ePrint Archive, Paper 2018/053 (2018). https://doi.org/10.1007/s13389-019-00220-8, https://eprint.iacr.org/2018/053
23. Schindler, W., Walter, C.D.: Optimal Recovery of Secret Keys from Weak Side Channel Traces, Lecture Notes in Computer Science, vol. 5921, p. 446–468. Springer Berlin Heidelberg, Berlin, Heidelberg (2009). https://doi.org/10.1007/978-3-642-10868-6_27, http://link.springer.com/10.1007/978-3-642-10868-6_27
24. Zaid, G., Bossuet, L., Habrard, A., Venelli, A.: Methodology for efficient cnn architectures in profiling attacks. IACR Transactions on Cryptographic Hardware and Embedded Systems p. 1–36 (2020). https://doi.org/10.13154/tches.v2020.i1.1-36, https://tches.iacr.org/index.php/TCHES/article/view/8391
25. Zhou, C., Paffenroth, R.C.: Anomaly detection with robust deep autoencoders. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 665–674. KDD ’17, Association for Computing Machinery, New York, NY, USA (Aug 2017). https://doi.org/10.1145/3097983.3098052, https://doi.org/10.1145/3097983.3098052

26. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks (arXiv:1703.10593) (Aug 2020). https://doi.org/10.48550/arXiv.1703.10593, http://arxiv.org/abs/1703.10593, arXiv:1703.10593 [cs]
27. Zhu, J.Y., Park, T., Wang, T.: CycleGAN and pix2pix in PyTorch , https://github.com/aitorzip/PyTorch-CycleGAN

A Robust autoencoder architecture

The complete optimization procedure for the RAE, based on [14] work, is described in Algorithm 1.

Algorithm 1 Robust autoencoder optimization procedure

```
Require: $X, \tau, \epsilon$
Ensure: $X = L + S$
  $L \leftarrow 0$
  $S \leftarrow 0$
  $X^* \leftarrow X$
  repeat
      $L \leftarrow X - S$
      $\theta, \phi = \text{backprop}(L, \mathcal{F}_\theta, \mathcal{E}_\phi)$    ▷ optimize left term of Eq. 2 using gradient descent
      $L \leftarrow \mathcal{F}_\theta(\mathcal{E}_\phi(L))$
      $S \leftarrow X - L$
      $S \leftarrow \text{prox}(S, \tau)$    ▷ update $S$ by minimizing $\tau||S||_1$ using Eq. 3
      $\epsilon_1 \leftarrow \frac{||X-L-S||_2}{||X||_2}$
      $\epsilon_2 \leftarrow \frac{||X^*-L-S||_2}{||X||_2}$
      $X^* \leftarrow L + S$
  until $\epsilon_1 \leq \epsilon$ or $\epsilon_2 \leq \epsilon$
  return $L, S$
```

Parameter ϵ corresponds to the stopping criterion for the optimization. The value of τ was chosen such that it reflects the amount of observed anomalies (see [14] for details). For the considered datasets, this parameter was set to $\tau = 0.04$ for Cswap Pointer and $\tau = 0.05$ for Cswap Arith respectively.

The network architecture was identical for the two datasets. The encoder network was built using four fully connected layers such that each layer size (the number of neurons) is the half of the previous one. The opposite was done for the decoder network. Each layer uses the ReLU activation, except the decoder last layer wich uses a hyperbolic tangent activation. The Adam optimizer was used with learning rate $\alpha = 10^{-4}$. The model forward/backward propagation is performed for 10 epochs at each optimization step using a batch size of 64 samples. Overall, no extensive architecture search nor parameter tuning has been performed, as the described configuration already exhibited sufficient efficiency. Indeed, a simple architecture allowed to produce satisfactory results. It is thus important to keep in mind that the aforedescribed architecture and tuning should not be treated as optimal for the given task.

B CycleGAN with multiplexers architecture

The following network structures and parameters tuning has been chosen in the application of the proposed framework on the two considered datasets. Hereafter, one dimensional convolution of a filters, b kernel size and c stride is denoted as $C(a,b,c)$. Similarly, the one dimensional convolution transpose is noted $C'(a,b,c)$. Bath normalization layers is noted as B and non linear activation as $\underset{d}{A}$ with d the activation function.

Firstly, the GANs generator use the LSGAN architecture [16], to prevent vanishing gradient that could occurs during training. G_A, G_B are built from convolutional autoencoders, composed of three downsampling blocks for the encoders

$$C(64,7,1) \rightarrow B \rightarrow \underset{ReLU}{A} \rightarrow C(128,3,2) \rightarrow B \rightarrow \underset{ReLU}{A} \rightarrow C(256,3,2) \rightarrow B \rightarrow \underset{ReLU}{A}$$

and the three upsampling ones for the decoders. ReLU activation are used on all layers, except for last layer of the decoder where tanh is used instead.

$$C'(256,3,2) \rightarrow B \rightarrow \underset{ReLU}{A} \rightarrow C'(128,3,2) \rightarrow B \rightarrow \underset{ReLU}{A} \rightarrow C'(64,7,1) \rightarrow B \rightarrow \underset{tanh}{A}$$

GANs dicriminators D_A, D_B are pixel discriminators [13], where each point is classified as fake or real instead of the whole pattern. They are built from three convolutional blocks:

$$C(64,1,1) \rightarrow \underset{ReLU}{A} \rightarrow C(128,1,1) \rightarrow \underset{ReLU}{A} \rightarrow C(1,1,1)$$

The discriminator loss is computed using the mean squared error between the output of last convolutional block and ground truth, which corresponds to the vector $\mathbf{1}_h$ if the input is real or $\mathbf{0}_h$ if it has been generated, where h match the size of the discriminators output. The architecture is trained for 10 epochs at each optimization step, with a learning rate $\alpha = 10^{-5}$ and a batch size of 16 samples.

C Details on implementation

The experimentations have been performed using Python 3.9. Implementation of the neural networks models has been done using the Pytorch library. For the proposed multiplexer CycleGAN, the implementation of the GAN networks is highly inspired from [27]. Based on this ressource, implementation of the described multiplexer CycleGAN should be more straightforward.

For both methods, the complete optimization procedure is performed under 30 minutes using a Nvidia RTX 3080 GPU and Intel i9-10900k CPU.

Data scaling is applied globally (not featurewise) on the input raw matrices so that points lie in the range $(-1, 1)$ before training through the networks. This is computed as follow:

$$X' = 2 \cdot \frac{X - \min(X)}{\max(X) - \min(X)} - 1 \tag{10}$$

This is mandatory since tanh activations are used for networks output (RAE and CycleGAN generators).