Trace Augmentation: What Can Be Done Even Before Preprocessing in a Profiled SCA?

Sihang Pu[1(✉)], Yu Yu[1], Weijia Wang[1], Zheng Guo[1], Junrong Liu[1], Dawu Gu[1], Lingyun Wang[2], and Jie Gan[3]

[1] Shanghai Jiao Tong University, Shanghai, China
{push.beni,yyuu,aawwjaa,guozheng,liujr,dwgu}@sjtu.edu.cn
[2] Shanghai Viewsource Information Science and Technology Company, Shanghai, China
lingyun.wang@viewsources.com
[3] Beijing Smart-Chip Microelectronics Technology Co., Ltd., Beijing, China
ganjie@sgitg.sgcc.com.cn

Abstract. Preprocessing is an important first step in side-channel attacks, especially for template attacks. Typical processing techniques, such as Principal Component Analysis (PCA) and Singular Spectrum Analysis (SSA), mainly aim to reduce noise and/or extract useful information from raw data, and they are barely robust to tolerate differences between profiling and target traces. In this paper, we propose an efficient and easy-to-implement approach to preprocessing by applying the data augmentation method from deep learning, whose appropriate parameters can be efficiently determined using a simple validation. Our trace augmentation method, when added prior to existing profiling methods, significantly enhances robustness and improves performance of the attacks. Simulation-based experiments show that our approach not only results in a more robust profiling (even show an enhancement to the known robust profilings), but also works well in the ideal scenario (no distortions between profiling and target traces). The results of FPGA-based and software experiments are consistent to the ones of simulation-based counterparts. Thus, we conclude that the proposed augmentation method is an efficient performance-boosting add-on to profiled side-channel attacks in real world.

1 Introduction

1.1 Motivation

The crypto community has witnessed the fast development of Side-Channel Attacks (SCAs) since Kocher's original works [11–13], and various more efficient SCA methods are proposed for different scenarios. Profiled SCA, first proposed by Chari et al. [5], adds a profiling phase (prior to the online exploitation phase) to the original SCA and can be considered as the powerful class of power analysis. Since then, various profiled SCA methods have been introduced and studied (see [6,15,19,24–26] for an incomplete list).

T. Eisenbarth and Y. Teglia (Eds.): CARDIS 2017, LNCS 10728, pp. 232–247, 2018.
https://doi.org/10.1007/978-3-319-75208-2_14

Despite its excellent effectiveness, profiled SCA presumes high similarity between profiling device and target device, which might otherwise result in less desirable performance and thus limit their applicability in practice. This issue was noticed and studied by Standaert et al. [22], Elaabid and Guilley [10] and Choudhary and Kuhn [7]. Furthermore, Whitnall and Oswald [26] proposed a robust profiling method by applying clustering technique, and Wang et al. [24] proposed another robust profiling technique using the ridge regression method. To the best of our knowledge, all the existing solutions mainly focus on the profiling phase rather than preprocessing stage.

1.2 Preprocessing Techniques

Preprocessing techniques are widely used to increase the success rate of side-channel analysis. The most widely used technique is Principal Components Analysis (PCA), which was first introduced by Archambeau et al. [1] and extended by Batina et al. [3]. Standaert and Archambeau [21] compared PCA with a more contrived method named Fisher Linear Discriminant Analysis (LDA). Bruneau et al. [4] carried out a mathematical analysis of dimensionality reduction methods (i.e., PCA and LDA), and they concluded that LDA is asymptotically the optimal dimensionality reduction strategy. Choudary and Kuhn [6] compared several Points of Interest (POI) techniques and discussed the rules of selecting components. Recently, Merino Del Pozo and Standaert [17] used Singular Spectrum Analysis (SSA), a technique originally used in time series analysis, to ameliorate the Signal Noise Ratio (SNR) of raw traces. We stress that the aforementioned preprocessing techniques may not work well with deviated target devices, and they need to rely on subsequent robust profiling techniques. To resolve the issue, we propose a new preprocessing method based on data augmentation.

1.3 Data Augmentation

The term data augmentation refers to methods for constructing iterative optimization or sampling algorithms via the introduction of unobserved data or latent variables. This method was popularized in the deep learning community and it can produce a better profiling set to mitigate overfitting and build a more robust model. Simard et al. [20] first created a general set of elastic distortions that vastly expanded the size of the training set. Moreover, in deep learning, it is the easiest and most common method to artificially enlarge the dataset using label-preserving transformations (e.g., [8,9,18]) in order to reduce overfitting. Krizhevsky et al. [14] significantly reduced the error rate using data augmentation techniques.

1.4 Our Contributions

In this paper, we tackle the following problem:

What can be done during (or even before)
the preprocessing stage to make the subsequent attacks more robust?

We answer this question affirmatively. Borrowing ideas from deep learning, we introduce the “trace augmentation” technique (i.e., applying data augmentation in the SCA context), which shifts each trace to some random extent and then combine them all (both original and perturbed traces) to form an augmented trace set. Our trace augmentation method can be applied prior to existing preprocessing procedures (e.g., PCA, LDA) and significantly improve the robustness and performance of the attacks.

Further, we propose a very efficient method (named lazy validation) to find out an appropriate range of the shift, only based on the profiling traces. Informally speaking, this method splits the profiling traces into two partitions, profiles on one distorted partition, validates (by conducting attacks) on the other undistorted one, and chooses the largest range of distortion that doesn’t essentially impact the result of the attack. Intuitively, the resulting suggestion can be seen as a conservative one that at least doesn’t impact the effectiveness in the ideal setting (where there are no distortions between profiling and target trace).

At last, we conduct both simulation-based and practical experiments to verify our approach. They both show that trace augmentation not only improves the performance of the subsequent attack in scenarios where discrepancy exists between the profiling device and target device, but also works well in the ideal case (i.e., without distortions). In addition, simulated-based results suggest that the improvement of our method is related to the correlation among points of each trace. In the practical setting, we carry out the experiments on both software and FPGA implementations, whose results are consistent to the simulation.

2 Trace Augmentation

In this section, we present our trace augmentation method, and show how to determine the suitable parameters efficiently. We stress that trace augmentation can either work independently, or can be added prior to any other preprocessing techniques such as PCA, in order to produce a more robust trace set.

2.1 Trace Augmentation Through Random Shift

Generally speaking, this augmentation approach manages to increase the number of traces exploitable in profiling phase, in order to improve the robustness and performance in profiled SCA. This is achieved through random shift of each trace to form an expanded trace set (i.e. in this process we misalign the traces by shifting the trace entirely). We visualized this random-shift-based augmentation approach in Fig. 1 and we sketch its procedure as follows.

1. Shift each trace horizontally at random up to some extent (the shift ratio to be determined later).
2. Repeat step 1 several times (corresponding to the augmentation ratio) to yield many perturbed traces.
3. Append these new perturbed traces to original set.

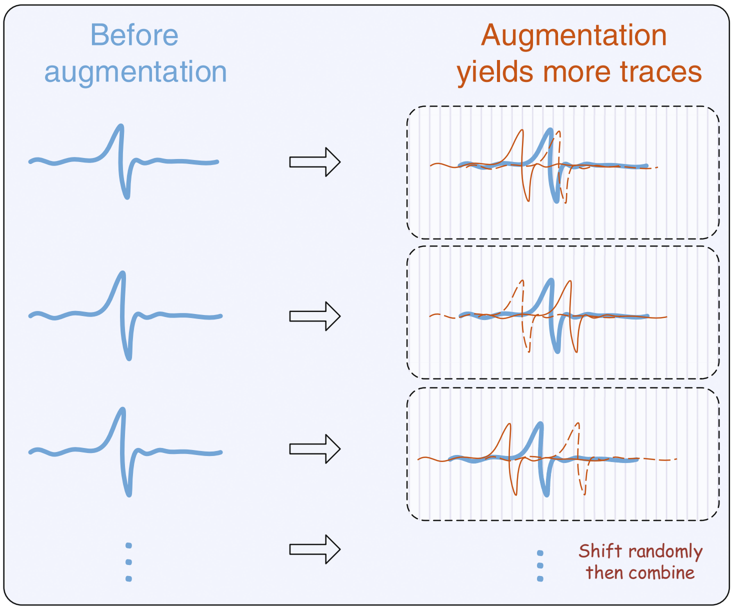


Fig. 1. A visual illustration of trace augmentation.

The above is parameterized by the shift ratio and augmentation ratio, denoted as γ and C respectively. The shift ratio γ, quantifies the extent of random shift: perturb each trace with a horizontal random shift drawn uniformly from $[-\gamma \cdot d, \gamma \cdot d]$, where d denotes the number of leakage points. The number of expanded traces is determined by the augmentation ratio C, shift ratio γ and the original trace number, that is, we have that $N_{new} = \gamma C \cdot N_{original}$, where N_{new} and $N_{original}$ are the numbers of original and new added traces respectively. Algorithm 1 presents this approach in formal details.

Intuitively, the more traces to exploit in the profiling phase the more effective the attack will be (against the target device). This motivates our approach which, given a fixed number of traces, augments the trace set (and exploit its information) as much as possible for a better performance in profiled SCA. The idea to enlarge the existing trace set is to apply perturbations since enlarging the dataset using such label-preserving method is the most straightforward yet efficient way to reduce overfitting. This approach works well especially when trace set is small. Moreover, we suggest to combine the trace augmentation approach with some data selecting (or points of interests) method such as PCA and LDA.

2.2 Search for Appropriate Parameter Though Lazy-Validation

As illustrated above, there are two undetermined parameters (i.e., γ and C), and it is somewhat tricky to define a general rule for to select them. We noted that the

Algorithm 1. Trace augmentation

```
Input: original trace set $\boldsymbol{L}$ ($N_{original}$ traces), number of leakage points $d$, shift
       ratio $\gamma$, augmentation ratio $C$
Output: augmented trace set $\boldsymbol{L}^{agmt}$ composed of $N_{original} + \gamma \cdot CN_{original}$
        traces
1 Append $\boldsymbol{L}$ to $\boldsymbol{L}^{agmt}$;
2 for $i = 1$ to $N_{original}$ do
3     Generate $\lfloor \gamma C \rfloor$ traces by randomly shift $\boldsymbol{L}_i$;
4     /* Note that each point of same trace shares a common horizontal
        shift                                                         */
5     Append the new traces to $\boldsymbol{L}^{agmt}$;
6 end for
7 return $\boldsymbol{L}^{agmt}$;
```

choice of γ affect the improvement of trace augmentation significantly, whereas (as verified in Appendix A) the choice of the other one (i.e., C) does not change the result much. We simply choose $C = 10$, and (as shown in Algorithm 2) design a lazy-validation method to yield a conservatively appropriate value for γ. We sketch the procedure as follows.

1. Select $C = 10$.
2. Split profiling traces into two (disjoint) partitions at random.
3. Perform trace augmentation on one partition with a certain shift ratio.
4. Build the template from this augmented partition.
5. Perform profiled SCA on the other partition with the template, and calculate a guessing entropy.
6. Repeat steps 3–5 with varied shift ratios, and obtain guessing entropy for each.
7. Select γ as the largest shift ratio that doesn't impact the result of the attack.

The underlying intuition of this procedure is that it is safe to perturb traces to some certain extent as long as it does not affect the performance in the ideal setting (no misalignment between profiling and target trace). Thus, the largest possible γ in respect of this condition can be seen as a conservative one. The optimal value of γ is highly specific to actual target trace set, whereas our conservative choice is in general an appropriate one universal for target traces with different levels of misalignment.

On the other hand, such lazy-validation might cost a little more time to yield the final parameter (γ). However, these procedures (including both augmentation and validation) are supposed to be finished at profiling stage, and attackers just end up getting a robust template. Therefore, attacking time will not be lengthened in practical.

Algorithm 2. Lazy-Validation

```
Input: original trace set $\boldsymbol{L}$ ($N_{original}$ traces), number of leakage points $d$, a
       vector of candidates of shift ratio $\Gamma$ in ascending order
Output: an appropriate shift ratio $\hat{\gamma}$
1  Split $\boldsymbol{L}$ into two random partitions $\boldsymbol{L}^{prof}$ and $\boldsymbol{L}^{valid}$;
2  Build template $T$ from $\boldsymbol{L}^{prof}$;
3  $ge_0 \leftarrow$ CalculateGuessingEntropy($T, \boldsymbol{L}^{valid}$);
4  $\hat{\gamma}$ is initialized to the first element of $\Gamma$ ;
5  for each $\gamma \in \Gamma$ do
6      $\boldsymbol{L}^{agmt} \leftarrow$ TraceAugmentation($\boldsymbol{L}^{prof}, d, \gamma, C = 10$);
7      Build template $T'$ from $\boldsymbol{L}^{agmt}$;
8      $ge \leftarrow$ CalculateGuessingEntropy($T', \boldsymbol{L}^{valid}$);
9      if $ge > ge_0$ then
10         break;
11     end if
12     $\hat{\gamma} \leftarrow \gamma$;
13 end for
14 return $\hat{\gamma}$;
```

3 Experimental Results

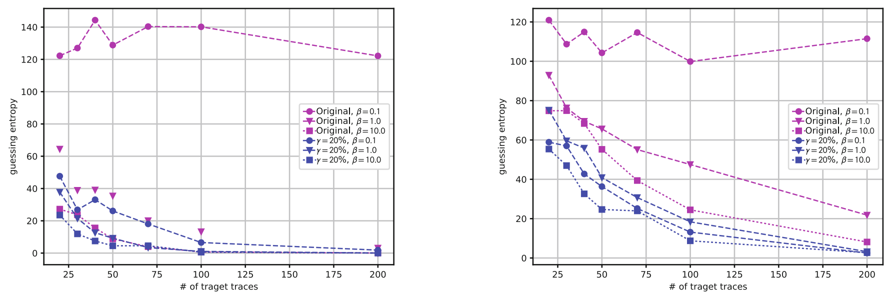
We test our approach through simulation-based and practical experiments. In simulation scenes, we disturb the power model of profiling and attacking stages. Whereas in practical scenes, since changing power model is knotty to control, we follow Whitnall and Oswald [26] and conduct the experiments by adding misalignment between profiling and target traces. We target on the first 8 bits of the AES's subkey and the output of the corresponding S-box in the first round. To evaluate the effectiveness of our method, we compute the guessing entropy [23] for comparison; in particular, we mount attacks for 100 times on different inputs and then compute the average rank of the correct key.

3.1 Simulation-Based Experiments

In simulation-based experiments, we assume that the leakage is subject to the multi-variant Gaussian distribution. Thus, we choose the mean of the distribution by randomly picking numbers and rely on a refined method named 'vine' [16] to simulate the covariance matrix. The 'vine' method is an efficient way to generate random correlation matrices, and correlations between leakage points are controlled by a single parameter β: higher β value corresponds to the more dependencies among points of each trace (please refer to Appendix B for more details). In order to simulate the imperfect case where profiling and attacking traces exist discrepancies, we perturb the (standardized) leakage function of the exploitation trace by imposing a 'noise' of uniform distribution $\mathcal{U}(-5, 5)$. And then we can generate the deviated exploitation trace using the perturbed leakage function.

Impacts of Correlation Among Points. Our augmentation method is followed by LDA (reduced to only one point) to extract the points of interest and the Gaussian template [5] building is used as the profiling phase. Figure 2 compares the guessing entropies of profiled attacks by varying the power model of profiling and target traces, using different correlation matrices (reflected by the value of β), with and without using trace augmentation. It shows that with augmentation the guessing entropies are declining much faster than those without augmentation ($\gamma = 0$), not only in imperfect cases with noise (as expected), but also in the ideal cases (no noise). A probable reason of this surprising result might be that more traces (although artificial one) can be accessed in profiling phase to overcome 'overfitting' issue.

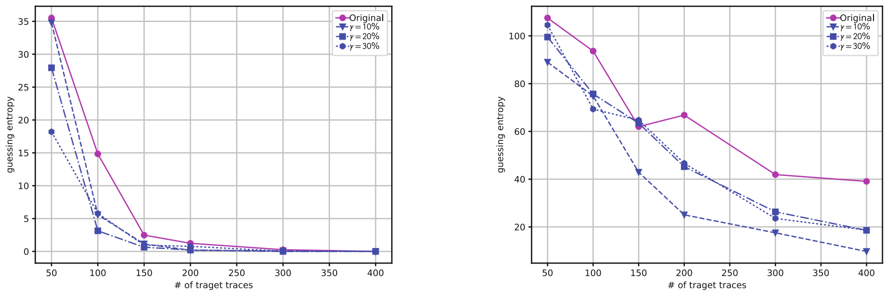
Further, we can see that attacks using augmentation are more 'insensitive' to β and in contrast, without augmentation, attacks are much more affected and become less effective while β decreases. We thus conclude that the effectiveness of this approach is enhanced while correlations among points are increasing (as β decreases). This means the power of this preprocessing method is correlated with the characteristics of the trace itself.



(a) perfect case (without discrepancy) (b) introducing noise as discrepancy

Fig. 2. The guessing entropies by varying the distribution between profiling and target traces (in terms of power model used), where simulation-implementation consists of 50 leaking points; 100 repetitions (to compute the guessing entropies) and 2000 profiling traces; using Gaussian templates

Enhancing the Robust Profiling Algorithm. We combine the our new method with the robust profiling algorithm proposed by Whitnall and Oswald [26]. We use K-means and Differential Cluster Analysis (DCA) introducted in [2] to perform attacks on target traces, in which the 'optimal' cluster number is according to the silhouette of the attack result. And other simulation facts are similar to the former simulation instance but β is fixed to 1.0. Figure 3 compares the guessing entropies of profiled attacks by different power models of profiling and target traces, with and without trace augmentation. It shows that with augmentation the guessing entropies are declining much faster than those without augmentation ($\gamma = 0\%$), even in the ideal cases (no discrepancy). We may conclude that even combining with robust algorithm (such as K-means), our trace augmentation technique can improve the performance in profiled attacks. Besides, we also provide comparison with another robust profiling algorithm (named ridge-based profiling [24]) in real FPGA-based contexts later (see Fig. 6).



(a) perfect case (without discrepancy)

(b) introducing noise as discrepancy

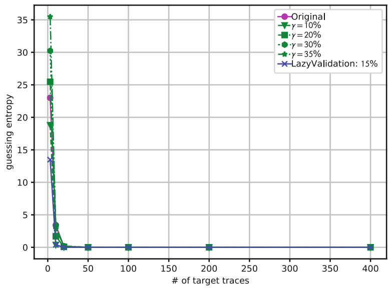
Fig. 3. The guessing entropies by varying the distribution between profiling and target traces (with different γ), where simulation-implementation consists of 50 leaking points; 100 repetitions (to compute the guessing entropies) and 2000 profiling traces; using cluster-based templates (K-means with DCA)

3.2 Software-Based Experiments

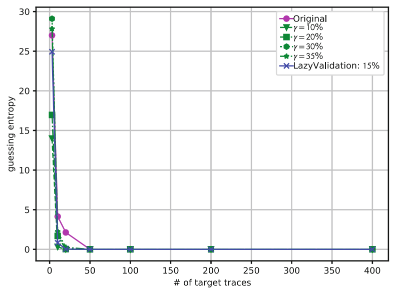
In software scenario, we target the AES implementation on Atmel ATMega-163 whose traces consist of 54 leakage points. Our augmentation method is followed by PCA (to reduce to 70% principal components [6]) to extract the points of interest and carry out the Gaussian templates building [5] as the profiling phase. Particularly, we use Hamming weight of target values as mean values in templates building, considering of its software implementation.

To provide a more comprehensive evaluation, we vary the parameter (of trace augmentation) γ from 10% to 35%, and the value $\gamma = 15\%$ is picked by lazy-validation of Sect. 2.2. For comparison, we also give the guessing entropies without trace augmentation. In such scenes, we follow Whitnall and Oswald [26] and conduct the experiments by adding misalignment between profiling and target traces. Note that this 'misalignment' (which misalign target traces with same points) should not be confused with 'shift' mentioned before.

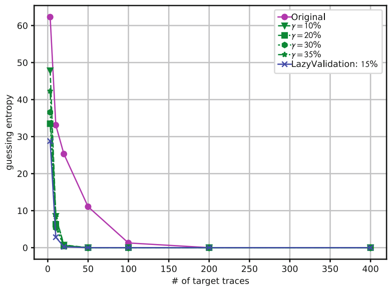
As shown in Fig. 4, with trace augmentation the performance of attacks have been improved in all settings even for those without misalignment. Note that misaligns of each trace are common for the same trace set. Further, we can see from following 6 sub-figures that the improvement of trace augmentation becomes more significant as the misalignment increases. Another observation is that, despite not always being the optimal, parameter γ chosen by lazy-validation is good enough for the attacks.



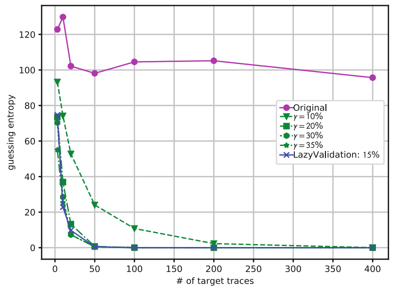
(a) no misalignment



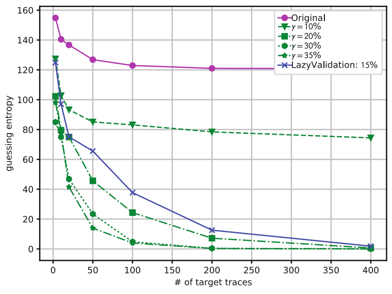
(b) misalignment = 5%



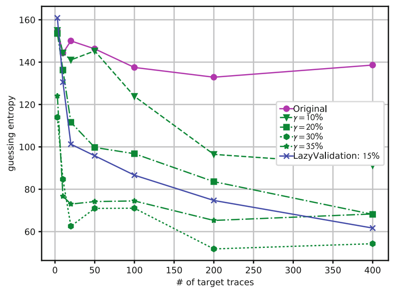
(c) misalignment = 10%



(d) misalignment = 15%

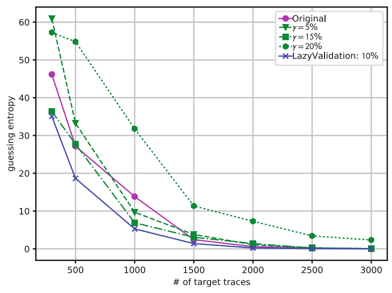


(e) misalignment = 20%

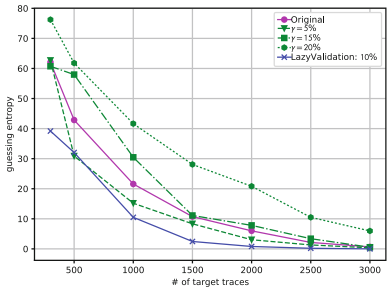


(f) misalignment = 25%

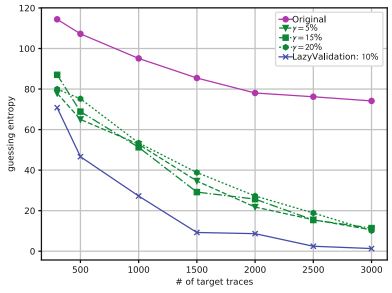
Fig. 4. The guessing entropies by varying the amount of deviation between profiling and target traces (in terms of misalignment), where software-implementation consists of 54 leaking points; 100 repetitions (to compute the guessing entropies) and 4000 profiling traces; using Gaussian templates



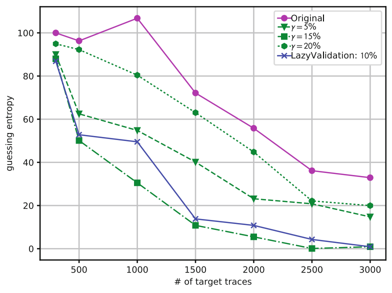
(a) no misalignment



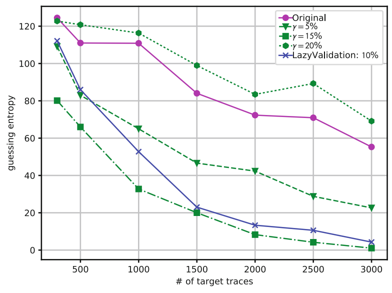
(b) misalignment = 5%



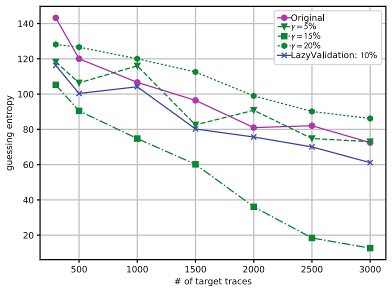
(c) misalignment = 10%



(d) misalignment = 15%



(e) misalignment = 20%



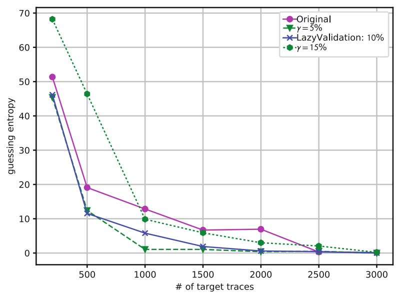
(f) misalignment = 25%

Fig. 5. The guessing entropies by varying the amount of deviation between profiling and target traces (in terms of misalignment), where FPGA-implementation consists of 20 leaking points; 100 repetitions (to compute the guessing entropies) and 5000 profiling traces; using linear-regression-based profiling

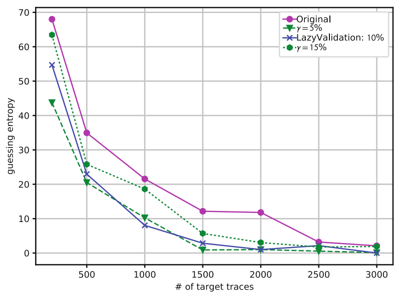
3.3 FPGA-Based Experiments

In hardware scenario, we target the AES implementation running on SAKURA-X board, whose traces contain 20 leakage points. Our attack strategy is similar to the one in software-based experiments, except for using linear-regression based profiling (in which the degree of power model is 1) from [19,25]. Compare to Gaussian template building, linear-regression based profiling can build up a model more efficiently with less number of measurements thus is more suitable to the FPGA scenario (with larger noise). In such scenes, we also conduct the experiments by adding misalignment between profiling and target traces as same as what we do in software experiments.

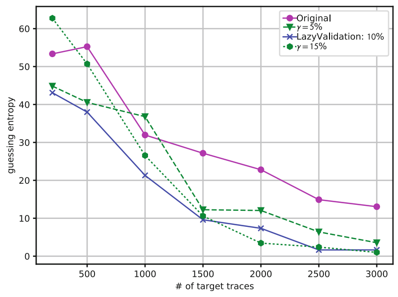
As shown in Fig. 5, the results of FPGA-based experiments is very similar to the software-based and simulation-based ones. Specifically, in the scenarios of high misalignment, the attacks without trace augmentation hardly distinguishes



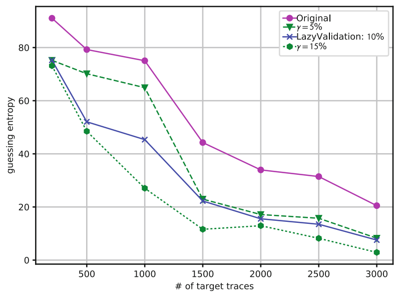
(a) no misalignment



(b) misalignment = 5%



(c) misalignment = 10%



(d) misalignment = 15%

Fig. 6. The guessing entropies by varying the amount of deviation between profiling and target traces (in terms of misalignment), where FPGA-implementation consists of 20 leaking points; 100 repetitions (to compute the guessing entropies) and 5000 profiling traces; using ridge-based profiling

between correct and incorrect keys, whereas the guessing entropies with trace augmentation still tend to zero.

The combination of trace augmentation with another known robust profiling method is also very interesting. Thus in Fig. 6 we present the guessing entropies of the attacks using ridge-based profiling (in which the degree of power model is 4) from [24] when combined with trace augmentation (and other experiments settings are same as the former one using linear regression based profiling). The result shows that our trace augmentation can also improve known robust profiling's performance.

4 Conclusion

In this paper, we show that trace augmentation based preprocessing can effectively improve the performance of profiled SCA, in both scenarios where the profiling device either deviates significantly from (or behaves close to) the target device. Further, we use a fast method called lazy-validation to obtain conservative but appropriate parameters. Finally, we provide simulation-based and practical experiments to confirm the effectiveness of our approach, and the former also indicates that the improvement of our approach (over other preprocessing techniques) depends on the correlation among points of each trace. We leave it as future work to explore other possible ways to augment the trace without using distortion, and whether such preprocessing strategies (if exist) can outperform trace augmentation.

Acknowledgment. This work is supported by the National Natural Science Foundation of China (Nos. 61472249, 61572149, 61572192, U1536103, 61402286, 61472250), the Major State Basic Research Development Program (973 Plan, 2013CB338004), the National Cryptography Development Fund MMJJ20170209, Industry & Education & Research Cooperation Program of Minhang District (2016MH310) and International Science & Technology Cooperation & Exchange Projects of Shaanxi Province (2016KW-038).

A The Impact of Augmentation Ratio C

Figure 7 shows the impact of augmentation ratio C in trace augmentation, and we can see that it is insignificant to the improvement.

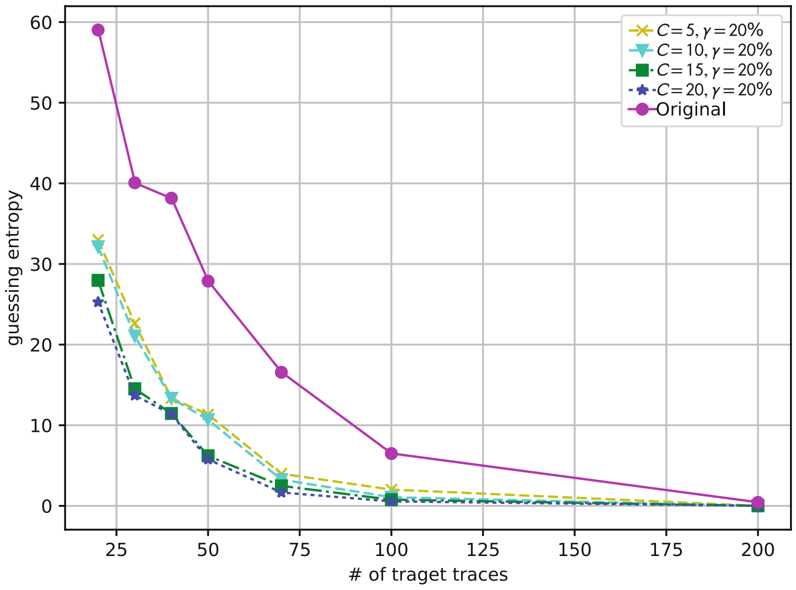
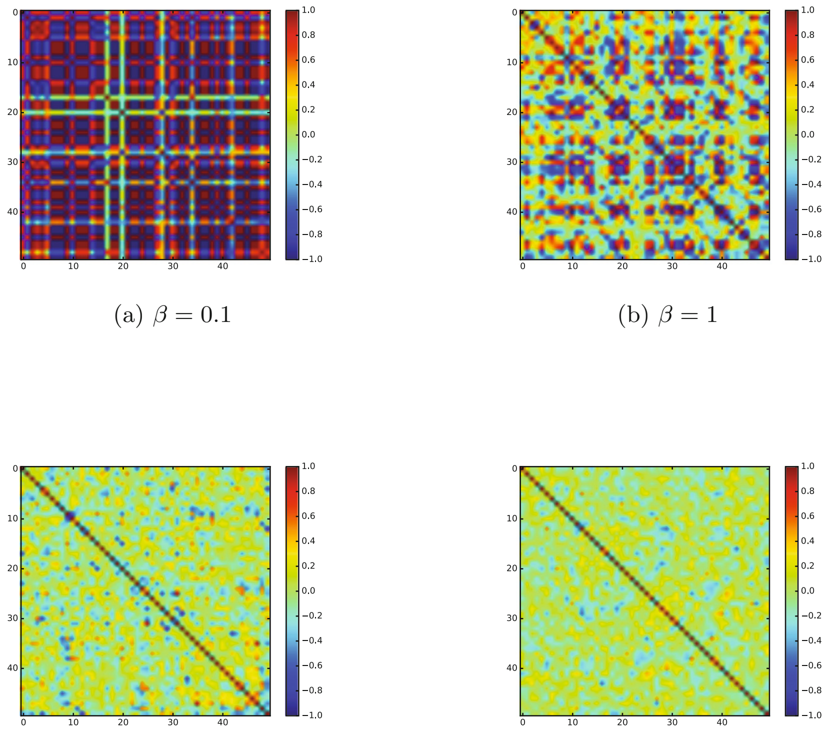


Fig. 7. The guessing entropies by varying augmentation ratio C in ideal scenario (no misalignment); simulation-based experiment containing 50 leakage points; 100 repetitions (to compute the guessing entropies) and 2000 profiling traces

B Correlation Matrices

'Vine' works in this way: off-diagonal values are derived from a beta distribution whose parameters satisfying $\alpha = \beta$, then perform a linear transform of these values to the interval $[-1.0, +1.0]$ (since beta distribution is defined on the interval $[0, 1]$). Correspondingly, values of correlation matrix are controlled by the single parameter β—higher β value corresponds to the less dependencies among points of each trace.

The correlation matrices of varied β value are provided as Fig. 8, colored according to correlations, from $[-1.0, +1.0]$. It is observed that correlations among points are enhanced as β decreasing.



(a) $\beta = 0.1$ (b) $\beta = 1$

(c) $\beta = 5$ (d) $\beta = 10$

Fig. 8. Correlation matrix (50 $\times$ 50) of each β parameter: 0.1, 1, 5, 10

References

1. Archambeau, C., Peeters, E., Standaert, F.-X., Quisquater, J.-J.: Template attacks in principal subspaces. In: Goubin, L., Matsui, M. (eds.) CHES 2006. LNCS, vol. 4249, pp. 1–14. Springer, Heidelberg (2006). https://doi.org/10.1007/11894063_1
2. Batina, L., Gierlichs, B., Lemke-Rust, K.: Differential cluster analysis. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 112–127. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04138-9_9
3. Batina, L., Hogenboom, J., van Woudenberg, J.G.J.: Getting more from PCA: first results of using principal component analysis for extensive power analysis. In: Dunkelman, O. (ed.) CT-RSA 2012. LNCS, vol. 7178, pp. 383–397. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27954-6_24
4. Bruneau, N., Guilley, S., Heuser, A., Marion, D., Rioul, O.: Less is more - dimensionality reduction from a theoretical perspective. In: Güneysu, T., Handschuh, H. (eds.) CHES 2015. LNCS, vol. 9293, pp. 22–41. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48324-4_2
5. Chari, S., Rao, J.R., Rohatgi, P.: Template attacks. In: Kaliski, B.S., Koç, K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 13–28. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36400-5_3

6. Choudary, O., Kuhn, M.G.: Efficient template attacks. In: Francillon, A., Rohatgi, P. (eds.) CARDIS 2013. LNCS, vol. 8419, pp. 253–270. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08302-5_17
7. Choudary, O., Kuhn, M.G.: Template attacks on different devices. In: Prouff, E. (ed.) COSADE 2014. LNCS, vol. 8622, pp. 179–198. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10175-0_13
8. Ciresan, D.C., Meier, U., Masci, J., Gambardella, L.M., Schmidhuber, J.: High-performance neural networks for visual object classification. CoRR abs/1102.0183 (2011)
9. Ciresan, D.C., Meier, U., Schmidhuber, J.: Multi-column deep neural networks for image classification. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012, pp. 3642–3649 (2012)
10. Elaabid, M.A., Guilley, S.: Portability of templates. J. Crypt. Eng. **2**(1), 63–74 (2012)
11. Kocher, P.C.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-68697-5_9
12. Kocher, P., Jaffe, J., Jun, B.: Differential power analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_25
13. Kocher, P.C., Jaffe, J., Jun, B., Rohatgi, P.: Introduction to differential power analysis. J. Crypt. Eng. **1**(1), 5–27 (2011)
14. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a Meeting Held December 3–6, 2012, Lake Tahoe, NV, USA, pp. 1106–1114 (2012)
15. Lerman, L., Bontempi, G., Markowitch, O.: A machine learning approach against a masked AES - reaching the limit of side-channel attacks with a learning model. J. Crypt. Eng. **5**(2), 123–139 (2015)
16. Lewandowski, D., Kurowicka, D., Joe, H.: Generating random correlation matrices based on vines and extended onion method. J. Multivar. Anal. **100**(9), 1989–2001 (2009)
17. Merino Del Pozo, S., Standaert, F.-X.: Blind source separation from single measurements using singular spectrum analysis. In: Güneysu, T., Handschuh, H. (eds.) CHES 2015. LNCS, vol. 9293, pp. 42–59. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48324-4_3
18. Sánchez, J., Perronnin, F.: High-dimensional signature compression for large-scale image classification. In: The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20–25 June 2011, pp. 1665–1672 (2011)
19. Schindler, W., Lemke, K., Paar, C.: A stochastic model for differential side channel cryptanalysis. In: Rao, J.R., Sunar, B. (eds.) CHES 2005. LNCS, vol. 3659, pp. 30–46. Springer, Heidelberg (2005). https://doi.org/10.1007/11545262_3
20. Simard, P.Y., Steinkraus, D., Platt, J.C.: Best practices for convolutional neural networks applied to visual document analysis. In: 7th International Conference on Document Analysis and Recognition (ICDAR 2003), Edinburgh, Scotland, UK, 3–6 August 2003, vol. 2, pp. 958–962 (2003)

21. Standaert, F.-X., Archambeau, C.: Using subspace-based template attacks to compare and combine power and electromagnetic information leakages. In: Oswald, E., Rohatgi, P. (eds.) CHES 2008. LNCS, vol. 5154, pp. 411–425. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85053-3_26
22. Standaert, F.-X., Koeune, F., Schindler, W.: How to compare profiled side-channel attacks? In: Abdalla, M., Pointcheval, D., Fouque, P.-A., Vergnaud, D. (eds.) ACNS 2009. LNCS, vol. 5536, pp. 485–498. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01957-9_30
23. Standaert, F.-X., Malkin, T.G., Yung, M.: A unified framework for the analysis of side-channel key recovery attacks. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 443–461. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01001-9_26
24. Wang, W., Yu, Y., Standaert, F.-X., Gu, D., Sen, X., Zhang, C.: Ridge-based profiled differential power analysis. In: Handschuh, H. (ed.) CT-RSA 2017. LNCS, vol. 10159, pp. 347–362. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-52153-4_20
25. Whitnall, C., Oswald, E.: Profiling DPA: efficacy and efficiency trade-offs. In: Bertoni, G., Coron, J.-S. (eds.) CHES 2013. LNCS, vol. 8086, pp. 37–54. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40349-1_3
26. Whitnall, C., Oswald, E.: Robust profiling for DPA-style attacks. In: Güneysu, T., Handschuh, H. (eds.) CHES 2015. LNCS, vol. 9293, pp. 3–21. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48324-4_1