Noname manuscript No.
(will be inserted by the editor)

Side Channel Cryptanalysis Using Machine Learning

Using an SVM to recover DES keys from a smart card.

Hera He · Josh Jaffe · Long Zou

December 14, 2012

Abstract Cryptographic devices use power as they operate. Variations in the amount of power consumed during an operation may leak information about data values being processed. Methods such as differential power analysis (DPA) [1] have been used to recover keys by exploiting such leaks. But DPA is tougher to apply to leakage that depends only on the key. For this project we attacked leakage from an 8-bit smart card performing DES[1] decryptions with a 56-bit key. We explored using an SVM to exploit the side channel leakage.

Using an SVM classifier we reached near 100% classification accuracy using features drawn from measurements during the DES key schedule, after training with at least 1000 samples out of our set of 11000. This is the first publication in which an SVM has been used to recover an entire key.

1 Introduction

Many devices today use cryptography to perform security tasks. For example, most automobiles can be unlocked using a wireless key fob. Parking meters in San Francisco accept payment from smart cards. SIM cards in mobile phones authenticate the phone to the base tower. The security of such systems requires 'tamper resistance' to prevent attackers from simply opening the device, reading all the secrets from its memory, and then using them to create clones.

But tampering is not the only threat. Physical circuits use power and generate electromagnetic emissions as they operate. Power consumption is a property that an attacker can measure—for example, by connecting an oscilloscope to a target device's ground pin and monitoring the voltage as it rises or falls as the amount of current through the chip varies with time. These variations carry information beyond what is communicated by the ciphertext. So the power measurements constitute a side channel—and this has traditionally been exploited with attacks such as simple power analysis (SPA) [1] and DPA. While DPA is often applied as a 'black box' attack method, template attacks (TA) [2] are a related approach that use a characterization stage requiring known keys.

Related work. It has recently been noted [4] [5] that the TA model is amenable to supervised learning. One paper [4] used a Least Squares SVM to extract information from a byte operation during AES decryption. Although not oriented towards key-recovery, the authors built a one byte classifier that achieved up to 99.3% accuracy for certain parameter choices. Another paper [5] looked at 3DES in an FPGA, and found that an SVM could reliably recover a few bits of the key—but would require a 70 bit search to complete the attack on a 112 bit key. Both papers found feature selection to be critically important.

In this project were interested to see whether supervised learning could automate the process of decoding SPA leaks and key dependent statistical leaks. We also revisited the question of feature selectionand found, in contrast to previous results, that aggressive feature selection was not necessarily helpful.

The rest of this paper is structured as follows. In section 2 we give an overview of the target device and data collection process, and describe the features that were used for supervised learning. In sections 3 and 4 we discuss the problem formulation and how we arrived at the final SVM model. In section 5 we present the results of our model. We conclude with some remarks

`yhe1@stanford.edu`, `joshj@stanford.edu`, `pz73@stanford.edu`

[1] Much of our analysis also applies to 3DES.

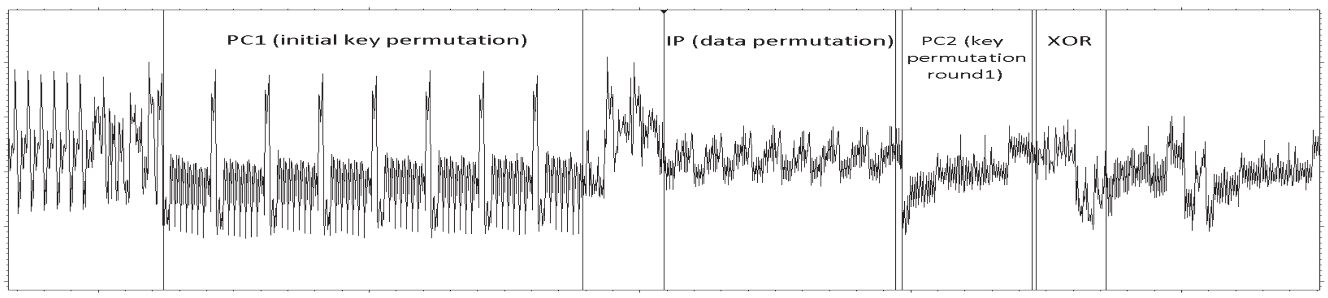

Fig. 1: Start of DES decryption. Each point represents 6 clock cycles. PC1, IP, PC2, and XOR regions indicated.

about the significance of these results, lessons learned, and future steps that could make this publishable in a top tier side-channel-crypto conference.

2 Data Set Overview: the DES smart card

This project focused on extracting a key from an 8-bit Atmel smart card performing decryptions using the Data Encryption Standard (DES)[3]. DES is a block cipher that uses a 56-bit key and has a 64-bit block size. For this project, it is relevant to know that DES performs bit permutations on the input message ("IP") and key ("PC1"), and then applies a round function 16 times in a Feistel structure. The round function performs a second key permutation ("PC2"), then computes the exclusive or of a 48-bit round key with some bits of the round input ("XOR"). A 6-bit to 4-bit substitution ("SBOX") is performed on the XOR output. The DES **key schedule** includes PC1 and PC2. This project focused on data-dependent variations in power measurements during clock cycles in which the PC1, PC2, and XOR operations were performed.

Power measurement traces were collected using:

- A ZeitControl BASIC card version 3.9 (programmed for DES decryption with chosen key)
- A DPA Workstation smart card reader with 5Ω resistor in series with ground provided to smart card
- A Tektronix DPO7104 oscilloscope connected to measure the voltage drop across the ground resistor

Figure 1 shows a trace of power measurements over one DES decryption.

The smart card ran at 4MHz. Power measurements were recorded at 200MHz for 50 samples per clock cycle—or over one million power measurements per trace. We stored a subset of each trace covering the DES initialization and first four decryption rounds. The data collection spanned 4000 unique keys and 16000 ciphertexts[2]. Each key and ciphertext combination was repeated five times. The traces were then aligned and each clock cycle was synchronized with a reference (to remove timing jitter). The data were further reduced by averaging together all traces that correspond to a single key and ciphertext combination, then selecting the 12th sample of each clock cycle as representative for that cycle.[3]

3 Problem Formulation

3.1 Problem Description

We collected $N = 11000$ instances and held out $N_1 = 1000$ of them to be test data, leaving $N_0 = 10000$ to be training examples. Each training example consisted of Y, C, X_1, X_2, X_3, where the key Y is a binary vector of length $p_k = 56$, input ciphertext C is a binary vector of length $p_c = 64$, X_1, X_2, X_3 are power trace measurements in PC1 region, PC2 region and XOR region, vectors with length $p_1 = 3039, p_2 = 904, p_3 = 516$ respectively. We used SPA and DPA to interpret the portions of the power traces as labeled in Figure 1 and determine exactly where the PC1, PC2, and XOR operations occur. We hope to recover keys Y in the test set.

We expect leaks during the PC1 and PC2 stages to depend directly on bits of the key, while the power measurements in the XOR stage may depend on bits of input ciphertext as well as bits of key. The analysis focused on these regions separately, i.e. we seek to find three separate multi-label classifiers $Y = G_1(X_1)$, $Y = G_2(X_2)$ and $Y = G_3(C, X_3)$.

[2] We primarily worked with the first 11000.

[3] We also performed a comparison between this '12th cycle' approach and that of using the average of all 50 measurements over a cycle as representing that cycle. Differences were minor.

3.2 Problem Transformation: From Multi-Label Classification To Binary Classification

We are looking for a classifier that assigns each instance to a set of $d = 56$ labels with binary classes. Two main methods have been proposed for tackling such problems [6]: the **binary relevance** (BR) approach and the **label power set** (LP) approach. In the BR approach, the multi-label problem is transformed into d separate binary classification problems. This breakdown of the classes may overlook possible interactions among the labels, however. The label power set method transforms the multi-binary-label problem into single-compound-label problem where the single label has 2^d possible classes. Here the interactions are encoded to the classification implicitly, but the exponential increase of the number of classes makes the computation intractable for large d.

The DES block cipher tends to manipulate the key primarily in 1, and 6 bit groupings. We decided to use BR first, figuring that operations that involve only one bit of the key are likely to be have minimal interaction and may be sufficient to recover the key. The problem was therefore transformed into finding 56 separate binary label classifiers for each of the three regions.[4]

3.3 Choice of evaluation metric

Ideally we want to find a model that is accurate for most target bits. Preliminary trials with the SVM showed similar performance trend on all key bits, suggesting that performance for any one bit was an acceptable approximation for performance over all 56. Hence to reduce the training time, during the model selection stage we evaluated models based only on the accuracy for the first bit.

4 Evolution of our SVM model

4.1 Overview of SVM results

Once the problem has been broken into finding a binary classifier for each bit of the key, we could use standard classification methods like SVM. With SVM, we were able to recover most of the key bits with high accuracy, achieving near 100% prediction accuracy in PC1 and PC2 regions, and above 90% accuracy in XOR region.

In order to get an effective SVM classifier it is essential to choose an effective kernel function, find a good set of kernel parameters, optimize the soft margin C, and identify an informative set of features. In this section, we will discuss our choice of these variables and how we arrived at our final SVM model.

Our first step was to fit a linear kernel SVM with L2 regularization to all three regions. We quickly figured out that standardizing the features before feeding into the SVM classifier both reduced the model training time and improved the prediction accuracy. Standardizing is especially important in the XOR region to normalize the binary valued ciphertexts and real valued power trace measurements to the same scale.

This model is very successful for PC1 and PC2 regions. We reached 100% prediction accuracy with about 1000 training samples (see Figure 2). However, SVM with linear kernel is not as successful in the XOR region. As we increase the sample size towards 10000, the XOR training accuracy and test accuracy converge to around 65% (see Figure 2). This indicates an inadequacy of the linear kernel for XOR region with just standardized feature matrix $[C\ X_3]$.

4.2 Kernels and parameters

The failure of the linear kernel in XOR indicates possible nonlinear relationship between the labels and the input features. Hence we experimented with different kernels for SVM, including polynomial kernels with different degrees, Gaussian kernel and sigmoid kernel. Table 1 shows the hold-out cross validation accuracy for different kernel. Polynomial kernel with degree 2 has slightly better performance than the linear kernel, while other kernels made it worse—little better than random guessing. We also tried optimizing the kernel parameters and the soft margin parameter C by grid searching and cross validation, but the search in a wide range still didn't save us from the poor performance. Hence we conclude the inappropriateness of all kernels other than the polynomial kernel of degree 2.

4.3 Feature engineering for XOR region

Our next goal was to push prediction accuracy in the XOR region to higher than 69.5%. Polynomial kernel with 2 degrees expands the dimension of the feature space from $p_3 + p_c = 636$ to $(p_3 + p_c)^2 > 4 \times 10^5$, where features are the pairwise product of the original inputs. With only 10000 training samples, overfitting is clearly

[4] Note that DES is unusual in manipulating individual bits. With other ciphers, bit groupings of 8 or more bits may be more common. When operations involve several bits of a key simultaneously, more complex models should probably be applied to capture the interactions among different classes. An intermediate form of LP that focuses on k bit groupings limits the exponential increase in classes to 2^k.

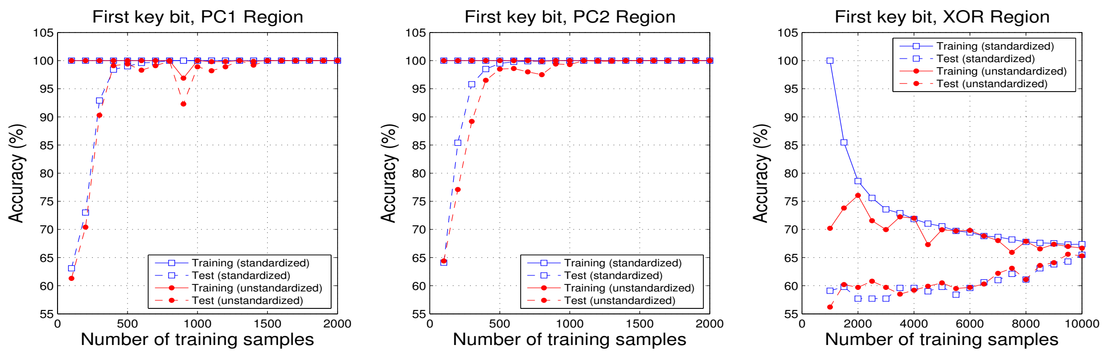

Fig. 2: Training and Test accuracy of SVM with linear kernel for PC1, PC2, and XOR regions respectively. Features for PC1 and PC2 only contains power traces X_1 and X_2 respectively. For XOR regions features contain both ciphertext C and power traces X_3

Kernel	NVEC	TRAIN ACC	TEST ACC
Polynomial d = 2	9014	100%	69.5%
Polynomial d = 3	9966	100%	59.2%
Polynomial d = 4	9995	100%	57.8%
Gaussian	10000	100%	52%
Sigmoid	6247	100%	50.2%

Table 1: Comparisons of SVM with different kernels. NVEC = # of support vectors. $N_0 = 10000, N_1 = 1000$.

a problem: we have training accuracy of 100%, and 90% of our inputs became support vectors. The high dimension of the new feature space makes the data set linearly separable—but clearly contains a lot of noise that is being fit by the model. We should not criticize this feature mapping too much, because 69.5% is significantly better than random guessing. Some of new features are indeed informative. But we may be able to obtain a better classifier by retaining the informative features while removing some of the others. We accomplished this with some judicious feature engineering.[5]

The engineered features are obtained in the following way. Beginning with the X_3 power measurements standardized to have zero mean and unit variance, and using ciphertext bits $C_{0..63}$, we computed: $\phi(C, X_3) = \phi'(C_0, X_3) \,||\, \phi'(C_1, X_3) \,||\, ... \,||\, \phi'(C_{63}, X_3)$. The function ϕ' is defined by $\phi'(1, X) = X$ and $\phi'(0, X) = -1 \cdot X$.

The result has $64 \cdot 572 > 36000$ elements, and corresponds to all terms in the degree 2 polynomial that are a product of one bit of ciphertext and one power measurement. Training with this feature vector, however, was extremely slow due to the large number of features—and prediction accuracy remained low!

So we next worked separately with each engineered feature matrix $\phi'(C_i, X_3)$. Training each bit of the key with all 64 engineered feature matrices reveals that for each key bit only one engineered feature matrix is informative. This fact is expected, given the structure of the DES algorithm.[6]

This success of this engineered feature matrix can also be justified in the following way. After the XOR operation $I_h = C_i \oplus K_j$ the power features may leak information about I_h. If the power consumption is linearly proportional to I_h, this could be expressed as:

$$P_t = a \cdot (K_j \oplus C_i) + b \tag{1}$$

for some power measurement (P_t) at time t. Using the following fact[7] for binary K_j and C_i:

$$K_j \oplus C_i = \frac{1}{2}(1 - (2K_j - 1) \cdot (2C_i - 1)) \tag{2}$$

and solving (1) for K_j reveals that:

$$K_j = -\frac{1}{a}\left(\left(P_t - \frac{a}{2} - b\right) \cdot (2C_i - 1) - \frac{a}{2}\right).$$

Substituting the unknown $\frac{a}{2} + b$ with $\langle P_t \rangle$ gives

$$K_j \propto (P_t - \langle P_t \rangle) \cdot (2C_i - 1) + \epsilon.$$

But calculating the product of the centered power trace $(P_t - \langle P_t \rangle)$ and $(2C_i - 1)$ is exactly the same as flipping the sign of the standardized power trace according to the value of C_i—which is how we obtained our engineered feature matrix. So if the power measurements X_3 can be linearly separated by an SVM according to the value of a bit I_h, then the feature matrix engineered with bit C_i can be linearly separated by an SVM to reveal $K_j = C_i \oplus I_h$.

[5] We could have used standard feature selection techniques, based on forward search or looking at mutual information. Instead, we chose a feature engineering strategy based on rationale that we will lay out towards the end of this section.

[6] We also expect only 32 of the engineered feature matrices to be informative, corresponding to the bits of 'R' in round 1. And we expect only a 48-bit round key (not the full 56-bit key) to be be used in XOR operations.

[7] For explanation, see the extended version of this paper

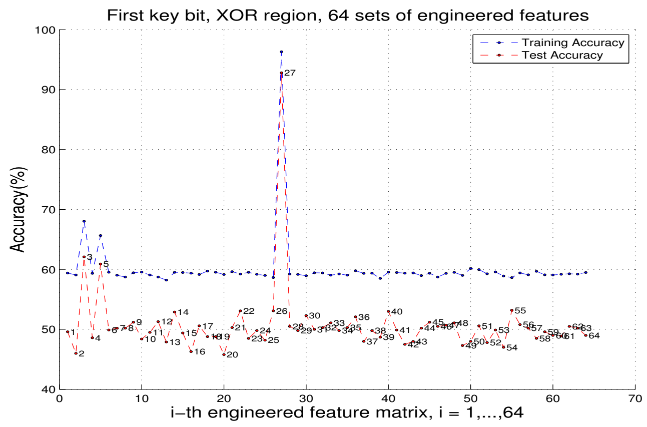

Fig. 3: Training and Prediction Accuracy for the first key bit based on 64 engineered feature matrices, the numbers being displayed in the plot correspond to the index of engineered feature matrix, being x in this case.

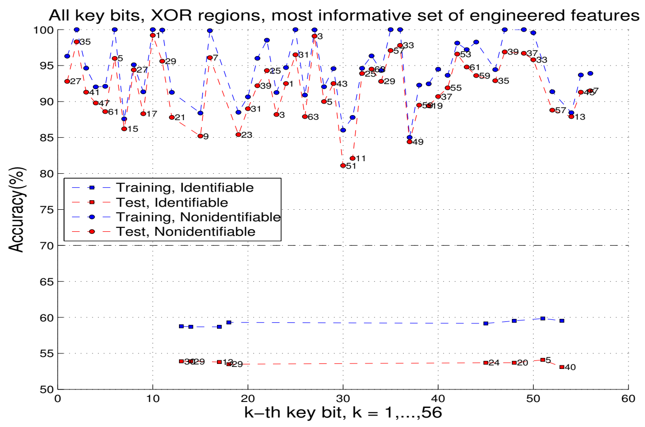

Fig. 4: Training and Prediction Accuracy for all key bits based on the most informative engineered feature matrix. The 48 key bits that are part of the decryption round key are all predicted with greater than 80% accuracy.

5 Results With Engineered Features In XOR Region

We present the result of training for the 1st key bit with our engineered feature matrix in Figure 3. All models were trained with 10000 samples and tested on 1000 samples. As shown in the plot, for the 1st key bit, the 27th ciphertext is leaking information, giving a prediction accuracy over 90%, while results trained from engineered feature matrices based on other bits of ciphertexts are basically random. This analysis was repeated for all bits of the key. The leakiest bit of ciphertext corresponding to each key was identified, and the results are reported in Figure 4. Note that in the XOR region, 8 bits of the key are not used. This is in accordance with our result of 8 unidentifiably key bits with test accuracy below 55%.

6 Conclusion

Initial power analysis of the DES card shows high amplitude power leaks during the PC1 and PC2 operations. Such leaks are challenging to exploit using traditional DPA (because it can't vary the keys) and SPA methods (because the signals are overlapping and complicated). Template attacks provide an alternative, but traditional TA methods are also complicated. In this project we found that a very straightforward SVM was able to decode these leaks and recover the keys with near perfect accuracy—after a relatively short training process.

Our results are compare favorably to previous research, in that we extracted a key, and also found feature selection to be less important (in the PC1 and PC2 regions). In the XOR region we found that either using a degree 2 polynomial kernel or using judicious feature engineering could draw out the relationship between key bit, ciphertext bit, and power measurement.

This work can be continued by applying these techniques to recover a key from a more modern device (such as the ASIC in an FPGA bitstream encryption core). Another interesting area to explore is applying these techniques to evaluating leakage using a sequence of specific test vectors.

References

1. Kocher, Paul, Josh Jaffe, Benjamin Jun. "Differential Power Analysis," CRYPTO 1999, pages 388-397.
2. Chari, Suresh, Josyula Rao, and Pankaj Rohatgi. "Template attacks." CHES 2002 (2003): 51-62.
3. FIPS PUB 46-3, "Data Encryption Standard," National Bureau of Standards, US Dept. of Commerce, Jan. 1977.
4. Hospodar, Gabriel, Benedikt Gierlichs, Elke de Mulder, Ingrid Verbauwhede, Joos Vandewalle. "Machine Learning in Side-Channel Analysis: A First Study." JCEN 2011, Vol 1, Issue 4, 293-302.
5. Lerman, L., Bontempi, G., Markowitch, O. "Side channel attack: an approach based on machine learning." COSADE (2011).
6. Tsoumakas, G., and I. Katakis. "Multi-label classification: An overview."
7. R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A Library for Large Linear Classification, Journal of Machine Learning Research 9(2008), 1871-1874.
8. Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1–27:27, 2011. IJDWM 3, no. 3 (2007): 1-13.