Profiling Methods in the Context of Side Channel Attacks

Elisabeth Oswald

University of Klagenfurt and University of Bristol

Overview

Part 1

- Why Profile?
- Types of Power Models
- Measures for Model Quality
- Check your understanding

Part 2

Part 3

Differential attacks

The concept of profiling was initially mentioned in the “Differential Power Analysis” paper by Kotcher et al.

You know by now that they proposed a very *generic* attack approach: using a simple t-test and a 0/1 power model.

A simple 0/1 power model has the advantage that you don’t make many assumptions about the actual leakage behaviour of a device.

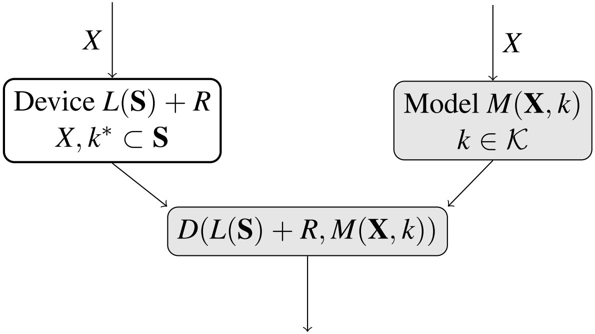
Go to https://answergarden.ch/1762609 to submit your responses.

What assumptions about the device leakage do you make with a 0/1 power model?

Differential Attacks: Recap

The adversary captures observations $L + r$ and produces input and key guess dependent model predictions M.

Using a statistical function (a distinguisher), the adversary decides, which key dependent model prediction “fits” better with the observations, thereby assigning a “score” to each key guess.



X

Device $L(\mathbf{S}) + R$

$X, k^* \subset \mathbf{S}$

X

Model $M(\mathbf{X}, k)$

$k \in \mathcal{K}$

$D(L(\mathbf{S}) + R, M(\mathbf{X}, k))$

$$\mathbf{D} = \{D(L + R, M(\mathbf{X}, k = 0)), D(L + R, M(\mathbf{X}, k = 1)), \ldots, D(L + R, M(\mathbf{X}, k = m))\}; \mathbf{r} \in \mathbf{R} \sim \mathcal{N}(0, 1)$$

The key guess with the highest score is likely to correspond to the true key value k^*.

Differential attacks, leakage model

With a 0/1 power model we make the assumption that the leakage of a device depends at least on the leakage of a single bit, i.e.

$$L(\mathbf{S}) = \beta_i s_i + L'(\mathbf{S}),$$

with $\beta_i \in \mathbb{R}$, $L'(\mathbf{S})$ excluding the term $\beta_i s_i$.
(The adversary choose a bit of the state, the bit is denoted by s_i, and they can give a weight to it, this is called β_i. In the Kocher et al. paper, $\beta_i = 1$.)

The “leakage model" thus in the Kocher attack is:

$$M(X) = s_i.$$

(X denotes the target value, which is a function of parts of the input and the key. X is contained within $\mathbf{S}$: $X \subset \mathbf{S}$).

This rather generic leakage model is a poor approximation of the actual leakage.

Differential attacks, leakage model

Submit your answers at https://answergarden.ch/1762614

The purpose of profiling is to derive a leakage model that:

- is better for attacks (submit A)
- explains most of the device leakage (submit B)
- fits with a proof for an implementation (submit C)

PROFILING IN CONTEXT OF DIFFERENT PURPOSES

Attacks: we seek to find a model that enables us to accurately predict the target device’s leakage for a specific intermediate state, with the aim to minimise the number of leakage observations that are necessary to significantly reduce the *rank* of a subkey (and therefore overall key).

Simulators: we seek to find models that enable us to accurately predict the target device’s leakage for arbitrary states, with the aim to enable accurate *relative* assessments of the leakage of different code sequences and with the aim of helping to explain leakage sources in code.

Proofs: we seek to validate proof assumptions by finding models that explain the sources of leakage with the aim of checking that certain proof assumptions are valid for a device.

Qualities of models in relation to use for all/many/one device(s) of a type

Clearly different applications of profiling indicate that models can have/should have different qualities:

Generic: a model that is used in a proof needs to apply *to all devices* of the same "type", thus it won't (or shouldn't) capture manufacturing or integration/setup specific features. It should capture architecture specific features.

Portable: a model that is used for attacks on *different devices* of the same "type" should capture enough device specifics that it improves attack efficiency, but it must not be too specific to the devices/setup that it was derived from because then it would not be portable to other devices of the same type.

Specific: a model that is used only on *one device* to illustrate the "worst case adversary" should be highly specific too that device on the expense that it is likely to fail (or not lead to the same best results) on other devices of the same type.

Classifying Models according to Steven's Levels of Measurements

Direct approximation $M \approx L$ (c.f. the 'ratio scale'), e.g. Bayesian templates and stochastic models.

Proportional approximation $M \approx \alpha L$ (c.f. the 'interval scale'), e.g. via linear regression (stochastic models withou the variance estimation), weighted Hamming weight; suitable for use in correlation DPA

Ordinal approximation $\{z|M(z) < M(z')\} \approx \{z|L(z) < L(z')\}\ \forall z' \in \mathcal{Z}$ (c.f. the 'ordinal scale'); e.g. HW, or derived via (un)supervised learning (e.g. cluster analysis); suitable for use with (e.g.) Spearman's rank correlation coefficient.

Nominal approximation $\{z|M(z) = M(z')\} \approx \{z|L(z) = L(z')\}\ \forall z' \in \mathcal{Z}$ (c.f. the 'nominal scale'); e.g. 0/1, "generic models" like the identity model, other clustering based approaches, can be used e.g. with 'partition-based' distinguishers of Standaert et al. (ISISC '08), and Mutual Information, Kolmogorov-Smirnov.

Types of Models vs Use Cases

Type	Proof	Simulation	Attack (portable)	Attack (worst case)
Direct				o
Proportional			o	
Ordinal		o	o	
Nominal	o	o	o	
	Explanatory power		Predictive Power	

Any model that offers a proportional or direct approximation of the leakage is particularly good for attacks: it can predict “new” leakages well.

Any model that offers an ordinal or nominal approximation of the leakage relates particularly well to proofs and is good for simulations: it can explain the leakage well.

Simulation models arguably should strike a balance between their predictive power and their explanatory power.

Judging Model Quality

We need to define what we are after:

Predictive power: important for attacks, in particular if we want to demonstrate a "worst case adversary", we need to judge how "close" model predictions are to "new observations": R^2, cross validation, key rank

Explanatory power: important for simulations and proofs, we need to judge how much of the leakage our model can predict: $F-$test

Data complexity requirements: important for estimation accuracy; the amount of data implicates how complex a model can be.

Typically one trades off one property for the other: I know of no method that would lead to a model that can achieve very high explanatory power AND predictive power simultaneously.

Quiz Time …

Question 1

When designing a profiled attack on a device for which you have an identical one (in the same test harness) available for profiling you would consider using:

1. a proportional model
2. an ordinal model
3. the Identity function as a model
4. a weighted Hamming weight as a model
5. a model that explains the power consumption very well

(Select all answers that you believe to be correct.)

Quiz Time …

Question 2

A predictive model

1. is a proportional approximation of the device leakage function
2. is good at helping pinpoint the sources of the leakage
3. is bad for simulations
4. is able to predict the validity of proofs for a device
5. is device specific

(Select all answers that you believe to be correct.)

Quiz Time ...

Question 3

A model that is meant for a target device that has the same architecture as the profiling device should

1. be portable
2. be a direct approximation of the profiling device's leakage function
3. be based on a nominal model
4. consider trading of genericity and explanatory power
5. be useable with a distinguisher such as correlation

(Select all answers that you believe to be correct.)

Overview

Part 1

Part 2

- Methods for deriving predictive models
- Classical templates
- Regression based templates
- Comparing classical templates and regression based templates
- Deep Learning
- Check your understanding

Part 3

Predictive Modelling

The goal is to derive proportional or direct models: we aim to be as close to the observed power consumption as possible.

With a fixed/limited amount of training data.

The training data comes with *labels*: we have full control over the training device, i.e. inputs, key, and any randomness. This is also called *supervised learning*.

(Perhaps in practice this implies doing a sucessfull unprofiled DPA attack first, perhaps this implies faulting the randomness generation process, perhaps this implies some degree of reverse engineering.)

In a Common Criteria based security evaluation scheme, this form of profiling is always attempted. In FIPS 140-3 this is out of scope.

Classical approaches to predictive modelling

I will stick to the parametric setting, and Gaussian assumption.

‘Classical’ templates (with/without noise estimation):

- Separate (multivariate) Gaussian models for each key-dependent value
- Covariance matrix estimated for each key-dependent value

Linear regression-based templates (with/without noise estimation):

- Linear regression model fitted to the pooled data at each time point
- Covariance matrix estimated for pooled data (2^{nd}, independent sample)

For both approaches it is possible to estimate the mean and variance for trace points independently. Any dependencies are then captured via the covariance matrix.

Profiles that do not incorporate any covariance information or even variance information are called “reduced templates” in the DPA book nomenclatura.

ML/DL Approaches to predictive modelling

Using templates to update priors is also a classical ML technique: (naive) Bayes. (“Naive” implies that the trace points are assumed to be independent.)

Many techniques exist, but approaches such as clustering (supervised or not) do not create proportional or direct approximations of the data: they create ordinal and nominal approximations. Thus they are useful for creating profiles if e.g. the device is too complex (and classical approaches lead to poor results) or if portability is of importance.

DL techniques such as MLPs or CNNs have proven to be useful in the context of predictive modelling. In particular CNNs (these are, roughly speaking, MLPs with a preceeding convolution layer) deal with misaligned traces with little user intervention.

“Classical templates”

$\mathbf{Y}_x$ =L + r= $\{Y_t|X = x\}_{t=1}^{T}$ is the random vector representing the leakage over time given that the associated intermediate target takes the value $x \in X$, and $X \subset S$.

The assumption is that the observed leakage Y follows a normal distribution:
$\mathbf{Y}_x \sim \mathcal{N}(\boldsymbol{\mu}_v, \Sigma_v)$.

The model M is fitted by finding the $T \times 1$ sample mean $\hat{\boldsymbol{\mu}}_x$ and the $T \times T$ sample covariance $\hat{\Sigma}_x$ from N_x measurements $\{\mathbf{y}_{x,n}\}_{n=1}^{N_x}$ observed on the profiling device.

Templates can be built for pairs of input and key, or target function values (these strategies are equivalent iff the target function satisfies the “equal images under differen subkeys” (EIS) property).

Linear Regression based Templates

The approach proposed by Schindler et al. is to fit a linear regression model to the *pooled data* at each point in time: $Y_t = \sum_{j=0}^{p} \beta_{j,t} g_j(X) + r$.

$\{g_0, \ldots, g_p\}$ are $p + 1$ functions of the intermediate value which form the *covariate set* (the elements in this set are also called the "explanatory variables" for the model).

In practice, g_0 is usually a constant (i.e. 1) and the remaining g_j are monomials of the form $\prod_{i \in \mathcal{I}} v[i]$ where $v[i]$ denotes the i^{th} bit of v and $\mathcal{I} \subset \{1, \ldots, m\}$ (with m the number of bits needed to represent V in binary), so that the model specification is of the form of a polynomial in function of the bits of the intermediate value, e.g. for a 2-bit value of X we have the form $Y = \beta_0 + \beta_1 x_0 + \beta_2 x_1 + \beta_3 x_0 x_1 + r$.

Linear Regression based Templates

Ordinary Least Squares (OLS) is used to obtain the coefficients $\hat{\beta}_{j,t}$ and subsequently the model fitted values $\hat{Y}_t = \sum_{j=0}^{p} \hat{\beta}_{j,t} g_j(V)$.

If all the influential terms are included in the model, the fitted values coincide asymptotically with the conditional means obtained via ‘classical’ templating ($\hat{\boldsymbol{Y}} = \boldsymbol{\mu}_v$).

The noise profiling stage consists of estimating a single (pooled) covariance matrix $\hat{\Sigma}$ from the model residuals observed in a second independent sample.

Linear Regression vs. Classical

- Classical templates have fixed complexity: 2^m conditional mean vectors, 2^m covariance matrices.
- Linear regression has adjustable complexity: an intercept, coefficients on all the equation terms, and one covariance matrix.
  - Potentially large reduction in profiling traces needed (e.g. linear model expression requires only $m + 1$ coefficients).
  - Potentially substantial degradation in model quality if simplifying assumptions are not correct.
- Linear regression models coincide with classical (in complexity and quality of deterministic part) once all possible monomial terms are included in the equation.

How do we utilise predictive models?

We have to choose a distinguisher, and decide upon one of the key hypothesis:

Choose the key hypothesis which maximises the *log-likelihood* of the observed traces.

OR (ignoring noise):

Choose the key hypothesis which maximises the *correlation* between the model fitted values and the observed traces.

Thus profiled attacks fit neatly into the “general DPA style attack” framework. The only difference is that we derive the leakage model from observations, rather than some a priori assumption.

Initial Analysis/Comparison between Methods

Templates vs. Stochastic Methods, B. Gierlichs, K. Lemke-Rust, C. Paar. *CHES 2006, LNCS 4249: 15–29, Springer.*

- LR templates recover key with fewer (profiling) traces but classical achieve higher success rates once profiling sample is large.
- Analysis primarily experimental: true distributions unknown so difficult to comment on model quality.
- Tested scenarios limited and favourable to LR (close to HW).

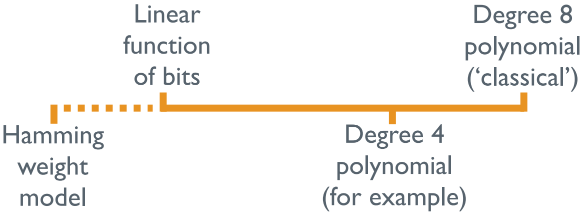
How to Compare Profiled Side-Channel Attacks?, F.X. Standaert, F. Koeune, W. Schindler. *ACNS 2009, LNCS 5536: 485–498, Springer.*

- Information theoretic metric can be used to quantify model quality.
- Analysis geared more towards theory (establishing an evaluation framework).
- Tested scenarios limited to simulated HW leakage – LR has big advantage; comparative findings do not extend to general case.

In-depth Analysis

Profiling DPA: Efficiency and Efficacy Tradeoffs, C. Whitnall, E. Oswald. *CHES 2013, LNCS 8086: 37–54, Springer.*

- Explore trade-offs in a **wider range of scenarios**, including those *not* well-suited to low-degree approximations.
- **Theoretic** (rather than experimental) evaluation where possible.
- Hypothetical scenarios with **fully-specified leakage distributions** give concrete benchmarks for model quality/performance.

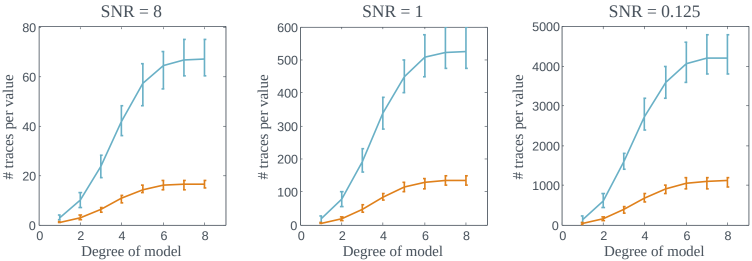

Linear function of bits
Degree 8 polynomial (‘classical’)
Hamming weight model
Degree 4 polynomial (for example)

Some Example Scenarios

We considered different types of leakage functions and different amounts of added Gaussian noise.

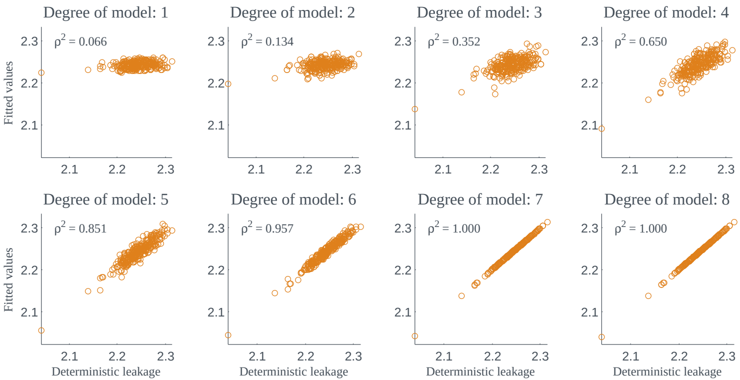
1. The leakage function is proportional to the **Hamming weight**, as motivated by typical behaviour of CMOS technology.
2. Adjacent wires interact so that the leakage is proportional to the **Hamming weight plus quadratic terms** involving adjacent bits of the intermediate value.
3. The leakage is a **highly nonlinear** function of the intermediate bits such as that arising from hardware implementations of AES.

Profiling Complexity



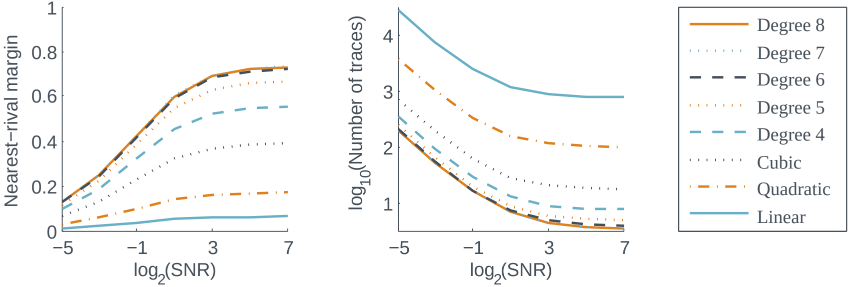
- Affects all leakage scenarios similarly.
- Estimation of maximum degree polynomials requires around 30 times traces than for linear models.
- Little change in complexity between degree 6 and degree 8 models; real savings only from degree 5 or lower.
- More accurate model fit (blue) comes at an exponentially increasing rate compared to not so accurate fit (orange).

HIGHLY NONLINEAR LEAKAGE:



- Linear model inadequate to approximate the leakage – captures just 6% of the variation.
- Degree 4 model accounts for about two thirds of the variation, with less than half the number of parameters required for the classical model.

HIGHLY NONLINEAR LEAKAGE:



- Very little difference in distinguishing power between the degree 5 and classical models.
- Linear and quadratic models are able to recover the key, but by very small margins and requiring lots of traces – over a hundred times as many in the case of the linear model.
- Degree 4 model requires around twice as many traces.

What to use?

- Linear regression is an excellent alternative to classical profiling when the true leakage function is simple.
- Over-simplified assumptions when the leakage is complex can substantially diminish attack performance; but trade-offs are possible and useful in particular if only limited training data is available.
- Device evaluation perspective:
  - Classical profiling remains the best way to test for vulnerability against the strongest possible adversary; assuming that enough data for profiling is available
- Attacker perspective:
  - In our example, degree 4 models offer a promising trade-off between profiling and attack complexity.
  - Even minimal profiling can substantially *increase* attack performance relative to standard assumptions (such as Hamming weight leakage) when those assumptions do not hold.

DL for predictive Modelling

Both MLPs and CNNs have been experimentally studied in the context of DPA style attacks. It is important to bear a few facts in mind:

The **universal approximation theorem** states that “networks with two hidden layers and a suitable activation function can approximate any continuous function on a compact domain to any desired accuracy” (Cybenko,Hornik et al.) .

However, by only having two hidden layers, one would require an exponentially large number of hidden neutrons per layer relative to the input size.

Neural networks are able to solve this problem by trading off the number of hidden layers for the number of nodes within each layer. Previous work shows that by having fewer nodes per hidden layer, ‘natural functions’ can be learnt quickly (Telgarsky).

DL for predictive Modelling

Both MLPs and CNNs have been experimentally studied in the context of DPA style attacks. It is important to bear a few facts in mind:

The **no free lunch theorem** states that a truly best learning algorithm cannot exist in general [12].

The NFL says that we cannot know what we have not seen: any learning algorithm only knows the “structure” of the training data, but, assuming that the training data does not represent the full input space, it cannot be expected to generalise to the unseen test data.

Specifically theorem 2 from [12] says that for a number of probabilistic measures of “error” (on a data set distinct from the training set), the average performance of any (pair of) algorithms is the same.

Thus: any algorithm that turns out to generalise well for some particular dataset will perform badly for some other dataset.

DL for predictive Modelling

There are many papers now around that experiment with both MLPs and CNNs. There are a few clear results/pointers:

- MLPs train more quickly than CNNs. Their representation is also more memory efficient.
- CNNs can learn how to correct some "misalignment" because they have some convolution layer. However, with decent pre-processing, any MLP can also cope with misaligned data.
- Finding the "best" set of hyperparameters for a device/setup is a trial and error process. No help is on the horizon.
- Papers find conflicting results and there is no way to conclude that DL approaches are better than classical templating/linear regression.

But if your classical templating/LR fails, it is well worth trying an MLP if you have nice data, and CNNs if you suspect that your data is misaligned and you cannot improve things via classical methods (filtering, elastica alignment, PCA/LDA).

Quiz Time ...

Question 4

Classical templates differ from linear regression based models because they

1. also characterise the noise
2. include covariances rather than just variances
3. are less configurable
4. are based on the assumption that the leakage is Gaussian
5. produce direct approximations instead of proportional approximations of the leakage

(Select all answers that you believe to be correct.)

Quiz Time ...

Question 5

Templates (classical ones and LR based ones)

1. can be plugged into a correlation based distinguisher
2. are suitable for DPA style attacks
3. must be used with a log-likelihood distinguisher
4. cannot approximate linear distributions very well
5. are bit-linear functions of the data

(Select all answers that you believe to be correct.)

QUIZ TIME …

Question 6

Deep learning based profiling

1. is provably more trace efficient than classical profiling methods
2. ensures portability of profiles
3. requires carefull tuning of the hyperparameters
4. cannot guarantee to produce profiles that outperform all other profiling methods on a device
5. fits into a general DPA style attack framework

(Select all answers that you believe to be correct.)

Overview

Part 1

Part 2

Part 3

- Modelling for Portability
- Clustering based approaches
- Check your understanding

Direct Models are not often Portable

Direct approximations of the device leakage are designed to **capture the scale, location and shape of the traces** used for profiling.

Thus **any deviation between the profiling and the attack traces potentially produces a mismatch** which renders the model unfit for purpose.

We may need to profile becaus the device model is not well approximated by anything simple such as the HW or 0/1 model (i.e. we profile in order to get *some* attack to work rather than improve an attack).

Clearly this calls for models that are nominal/ordinal approximations of the data, whereby we emphasise the predictive power of these models for the purposes of attacks.

(UN)SUPERVISED LEARNING

So far we have looked purely at supervised learning approaches: we assign traces from the profiling set to some "categories", which we define via the intermediate values.

In contrast *unsupervised* techniques—in particular, unsupervised clustering algorithms, attempt to learn categories implied by the profiling traces. In other words, clustering algorithms aim to find a meaningful arrangement of objects with no *a priori* knowledge about the number or characteristics of the underlying classes.

Clustering is the task of grouping objects (in this case, observed power consumption traces) in such a way that the objects inside any given group are *similar* to one another whilst objects in different groups are *dissimilar*.

Use within DPA style attacks

Nominal models can be used within a so-called **partition-based DPA attack** such as

- Mutual Information (MI) [7],
- Kolmogorov-Smirnov (KS) [10],
- the Variance Ratio (VR) [9], and its multivariate extension in the context of
- Differential Cluster Analysis (DCA) [3].

Be mindful that if the target function is injective (e.g. the AES S-box), the ‘trivial’ nominal power model which treats each intermediate value as a distinct class *invariably* fails to distinguish between key hypotheses in *any* partition-based DPA (see [9, 11]). Therefore, a meaningful non-trivial grouping is required.

Deriving Nominal Models via Unsupervised clustering

Task: Arrange objects s.t. those inside a given group are *similar* whilst those in different groups are *dissimilar*.

Assumption: Number or characteristics of the underlying classes are *a priori* unknown (unlike supervised classification).

Method: Large selection of iterative trial-and-error solutions:

- Cluster models vary: hierarchical, centroid-based, density- or distribution-based, graph-based …
- ‘Similarity’ measures vary: Euclidean distance, correlation, Hamming, Manhattan …

N.B.: Notoriously difficult to match the best-suited learning algorithm to a given problem; no learning algorithm is ‘best’ across different problems.

Deriving and using Nominal Models

General strategy

1. Partition the profiling traces according to the intermediate values and compute the means $\{\bar{\mathbf{y}}_x\}_{x \in \mathcal{X}}$.
2. Obtain a mapping $M : \mathcal{X} \longrightarrow \mathcal{M}$ by clustering the mean traces.
   - Values in $\mathcal{X}$ not represented in the profiling dataset are mapped to cluster $C + 1$ (i.e. an 'other' category).
3. Use M as the (nominal) power model in 'partition-based' DPA against the target traces.

The mapping in the second step thus associates each intermediate value (as represented by its' mean leakage value from the trace) to a "label" or model category.

There is no distance measure, or any other quantitative relationship assumed between the model categories. Thus they give us a nominal model.

Example instantiation

Robust Profiling for DPA Attacks, C. Whitnall, E. Oswald. *CHES 2015, LNCS 9293: 3–21, Springer.*

Best configuration

Clustering algorithm: Principal component analysis followed by *k*-means clustering.

DPA distinguisher: Univariate and multivariate variance ratio (thus VR and DCA).

Benchmark: Correlation DPA using the first principal component to approximate a 'proportional' power model.

(Hierarchichal clustering was less effective than k-means clustering; We used the shilouette index to judge the clustering quality.)

Experimentally evaluating the Robustness/Portability

Data

Software: 10,000 traces from an unprotected AES implementation on an ARM microcontroller.

Hardware: 5,000 traces from an unprotected AES implementation on an RFID-type system.

Experimental approach

1. Randomly draw (disjoint) profiling and attack samples from the full dataset.
2. Derive nominal and proportional power models from the profiling subsample.
3. Modify the attack subsample to simulate a variety of discrepancies.
4. Perform correlation- and univariate/multivariate VR-based DPA.
5. Repeat to estimate guessing entropies (average rank of correct subkey).

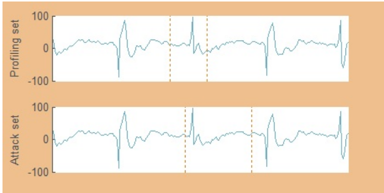
DISTORTIONS

- Width and location of “attack windows”
- Trace resolution
- Varying noise
- Inconsistent filtering
- Misalignment due to clock frequency changes

These distortions were independently applied to each attack subset. Attacks on the last category failed alltogether.

Attacks were also applied to trace sets without any distortions to confirm the validity of the attack principle.

Discrepancy in window width and location

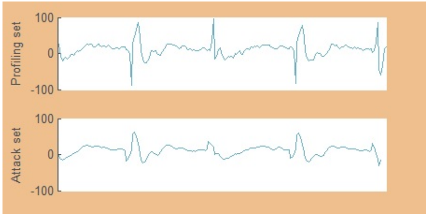


Scenario: Attacker roughly knows the interesting 'windows' but cannot match them precisely.

	Attack sample size ⟶	Software						Hardware					
		DCA(M_{KM})		**VR**(M_{KM})		**Corr**(M_{P1})		**DCA**(M_{KM})		**VR**(M_{KM})		**Corr**(M_{P1})	
		50	**400**	**50**	**400**	**50**	**400**	**50**	**400**	**50**	**400**	**50**	**400**
Offset	$-\lfloor \mathbf{w}/\mathbf{2} \rfloor$	53	1	87	1	15	1	121	65	68	1	22	1
	$-\lfloor \mathbf{w}/\mathbf{4} \rfloor$	37	1	65	1	3	1	51	1	66	1	20	1
	$\mathbf{0}$	34	1	72	1	1	1	15	1	65	1	21	1
	$\lfloor \mathbf{w}/\mathbf{4} \rfloor$	27	1	83	1	1	1	25	1	76	1	24	1
	$\lfloor \mathbf{w}/\mathbf{2} \rfloor$	74	4	109	1	22	1	66	1	113	3	90	1

- Software attacks vulnerable to this; larger samples help to compensate.
- Hardware attacks vulnerable to the most extreme shifts.

DISCREPANCY IN TRACE PRE-PROCESSING



Scenario: Training traces have been pre-processed in a manner not precisely known to the attacker.

Simulated distortion: Apply additional filtering to the attack subsample (moving averages).

Attack		**Software**						**Hardware**					
sample		**DCA**(M_{KM})		**VR**(M_{KM})		**Corr**(M_{P1})		**DCA**(M_{KM})		**VR**(M_{KM})		**Corr**(M_{P1})	
size ⟶		**50**	**400**	**50**	**400**	**50**	**400**	**50**	**400**	**50**	**400**	**50**	**400**
Smoothing window	**1**	43	1	96	1	16	1	19	1	62	1	19	1
	2	44	1	75	1	5	1	24	1	59	1	17	1
	4	51	1	104	1	5	1	74	1	100	4	79	1
	8	77	1	106	1	16	1	111	32	121	54	100	17
	16	115	5	123	3	53	1	112	82	118	94	113	64

- Software attacks robust; smoothing pairwise even improves outcomes.
- Hardware attacks less robust (fewer clock cycles; raw traces are already shorter and more coarsely sampled).

ROBUST PROFILING SUMMARISED . . .

- Unsupervised clustering can recover nominal power models for use in effective ‘partition-based’ DPA.
  - Requirements in profiling phase are minimal relative to full profiling.
  - Robustness to discrepancies between profiling and attack traces is considerably greater.
- Proportional power models can recovered under the same circumstances, for use in correlation DPA.
  - More efficient, in the case of software experiments; slightly less in the case of hardware experiments.
  - Almost as robust.

At the moment, a fair number of papers are published that advocate and explore the use of deep learning in the context of misaligned traces. As research papers are biased towards reporting positive results, most papers that use CNNs conclude that they are able to deal with some degree of misalignment.

Quiz Time …

Question 7

Nominal profiling methods

1. can fit into a standard DPA style attack framework
2. have the advantage of potentially coping with differences between profiling and attack data
3. can be built based on DL, ML and classical profiling techniques
4. can provably compensate for misalignment in traces
5. require PCA as pre-processing method

(Select all answers that you believe to be correct.)

Quiz Time ...

Question 8

Robustness is important

1. if the profiling traces are not from the same device as the attack traces
2. if countermeasures impact on the trace acquisitions
3. because differences in the properties of traces may change over time
4. to assess the potential for real world/outside of the lab attacks
5. because profiling is mandated by some evaluation regimes

(Select all answers that you believe to be correct.)

Thank You for Listening!

Some slides were created by Carolyn Whitnall. Much of this research resulted through my collaboration with her.

Reading Material I

TAMPRES: Tamper Resistant Sensor Nodes.
http://www.tampres.eu, 2009–2013.

C. Archambeau, E. Peeters, F.-X. Standaert, and J.-J. Quisquater.
Template Attacks in Principal Subspaces.
In L. Goubin and M. Matsui, editors, *CHES 2006*, volume 4249 of *LNCS*, pages 1–14. Springer, 2006.

L. Batina, B. Gierlichs, and K. Lemke-Rust.
Differential Cluster Analysis.
In C. Clavier and K. Gaj, editors, *CHES 2009*, volume 5747 of *LNCS*, pages 112–127. Springer, 2009.

Reading Material II

L. Batina, J. Hogenboom, and J. van Woudenberg.
Getting More from PCA: First Results of Using Principal Component Analysis for Extensive Power Analysis.
In O. Dunkelman, editor, *CT-RSA 2012*, volume 7178 of *LNCS*, pages 383–397. Springer Berlin / Heidelberg, 2012.

S. Chari, J. Rao, and P. Rohatgi.
Template Attacks.
In B. Kaliski, Ç. Koç, and C. Paar, editors, *CHES 2002*, volume 2523 of *LNCS*, pages 51–62. Springer Berlin / Heidelberg, 2003.

G. Cybenko.
Approximation by superpositions of a sigmoidal function.
Mathematics of Control, Signals and Systems, 2:303–314, 1989.

Reading Material III

B. Gierlichs, L. Batina, P. Tuyls, and B. Preneel.
Mutual Information Analysis: A Generic Side-Channel Distinguisher.
In E. Oswald and P. Rohatgi, editors, *CHES 2008*, volume 5154 of *LNCS*, pages 426–442. Springer–Verlag Berlin, 2008.

K. Hornik, M. B. Stinchcombe, and H. White.
Multi-layer feedforward networks are uni-versal approximators.
1988.

F.-X. Standaert, B. Gierlichs, and I. Verbauwhede.
Partition vs. Comparison Side-Channel Distinguishers: An Empirical Evaluation of Statistical Tests for Univariate Side-Channel Attacks against Two Unprotected CMOS Devices.
In P. Lee and J. Cheon, editors, *ICISC 2008*, volume 5461 of *LNCS*, pages 253–267. Springer Berlin / Heidelberg, 2009.

Reading Material IV

- N. Veyrat-Charvillon and F.-X. Standaert.
  Mutual Information Analysis: How, When and Why?
  In C. Clavier and K. Gaj, editors, *CHES 2009*, volume 5747 of *LNCS*, pages 429–443. Springer Berlin / Heidelberg, 2009.

- C. Whitnall, E. Oswald, and F.-X. Standaert.
  The Myth of Generic DPA...and the Magic of Learning.
  In J. Benaloh, editor, *CT-RSA*, volume 8366 of *LNCS*, pages 183–205. Springer, 2014.

- D. Wolpert.
  The supervised learning no-free-lunch theorems.
  01 2001.