Differential Power Analysis (DPA)

Preparation: Assume an adversary that is able to send a number of challenges to an implementation of which he is able to predict a data dependent and key dependent intermediate value $\mathrm{f}(d,k)$, where d is known data and k is a part of the key. The adversary is additionally able to sample the power consumption while the intermediate value is processed.

Acquiring data: The adversary sends D challenges with the known data d_l, $l \in \{1,...,D\}$. He stores all data values in the data vector $\mathbf{d} = (d_1,...,d_l,...,d_D)^T$. For each challenge d_l, the adversary records one power trace $\mathbf{t}_l = (t_{l,1},...,t_{l,j},...,t_{l,T})$, where T is the number of samples $t_{l,j}$, $j \in \{1,...,T\}$ for each trace. All power traces of all challenges are stacked rowwise to form the $(D \times T)$ matrix $\mathbf{T}$.

$$\mathbf{T} = \begin{pmatrix} \mathbf{t}_1 \\ \mathbf{t}_l \\ \mathbf{t}_D \end{pmatrix} = \begin{pmatrix} t_{1,1} & \cdots & t_{1,T} \\ \vdots & t_{l,j} & \vdots \\ t_{D,1} & \cdots & t_{D,T} \end{pmatrix}$$

Calculating hypothetical intermediate results: The partial key k can take K possible values. All key hypotheses k_i, $i \in \{1,...,K\}$, are stored in the vector $\mathbf{k} = (k_1,...,k_i,...,k_K)$. The adversary is now able to compute the hypothetical intermediate values $v_{l,i} = \mathrm{f}(d_l,k_i)$, which again form a matrix $\mathbf{V}$ of size $(D \times K)$.

$$\mathbf{V} = \begin{pmatrix} v_{1,1} & \cdots & v_{1,K} \\ \vdots & v_{l,i} & \vdots \\ v_{D,1} & \cdots & v_{D,K} \end{pmatrix}$$

Recall that the goal of the adversary is to find the correct hypothetical key, hence to choose the correct column of this matrix.

Mapping intermediate values to hypothetical power consumption values: The adversary will now map $\mathbf{V}$ to $\mathbf{H}$, where $\mathbf{H}$ is the hypothetical power consumption of the device in the moment the the intermediate value $v = \mathrm{f}(d,k)$ is processed. The adversary needs to know how the target device leaks information. The chosen power model in this case is the Hamming Weight $\mathrm{HW}(v)$, leading to $\mathbf{H} = \mathrm{HW}(\mathbf{V})$.

Finding the 'best' key hypothesis: The adversary compares each column of the the hypothetical power consumption $\mathbf{H}$ (each column corresponds to one key hypothesis k_i) to each column of the actual power consumption $\mathbf{T}$ (each column of $\mathbf{T}$ corresponds to one point in time, j). Hence the adversary browses through all keys *and* through all points in time to find (i) when the

data leakage occurs and (ii) for which key it occurs. In other words he checks when and for which key he was able to 'best' predict the power consumption of the device. He stores the results of this comparison in a matrix $\mathbf{R}$ of size $(K \times T)$, where each element $r_{i,j}$ is a metric to measure a relation between the predicted and the actual power consumption.

DPA using correlation: A strong metric for the relation of two variables is the correlation. In DPA, the power consumption at j (the jth column of $\mathbf{T}$) is correlated to the hypothetical power consumption of key guess i (the ith column of $\mathbf{H}$). Both vectors have the same length D.

Use the Pearson correlation coefficient

$$r_{xy} = \frac{\sum x_i y_i - n\overline{x}\,\overline{y}}{(n-1)s_x s_y}$$

The expression for $r_{i,j}(\mathbf{h}_i, \mathbf{t}_j)$ can be used to calculate all values of the correlation matrix $\mathbf{R}$.

The adversary changes his mapping from $\mathbf{V}$ to $\mathbf{H}$. Instead of the Hamming Weight $\mathrm{HW}(v)$, he only uses a single bit of v, hence $h_{l,i} = \mathrm{HW}(v_{l,i} \cdot 2^m)$, where m is the chosen position of that bit (e.g. 0 for the LSB). Assume all possible values of v to be equiprobable.

DPA using difference of means: In the first publication of DPA, a different mapping from $\mathbf{V}$ to $\mathbf{H}$ was proposed. Instead of a correlation, the difference of means of two sets is calculated. The decision to which set a measurement point $t_{l,j}$ is added depends on a single bit of v, like above $h_{l,i} = \mathrm{HW}(v_{l,i} \cdot 2^m)$. Hence for each key guess i and for each point in time j we get two sets ($h_{l,i} \in \{0, 1\}$):

$$M_0 = \sum_{l=1}^{D} t_{l,j} \cdot (1 - h_{l,i}) \qquad\qquad M_1 = \sum_{l=1}^{D} t_{l,j} \cdot h_{l,i}$$

The adversary calculates $r_{i,j} = \overline{M_1} - \overline{M_0}$ (hence generates $\mathbf{R}$), chooses the biggest value, where index i of that value corresponds again to the 'best' key hypothesis.