Template Attacks

Federico Zanca

Computer Science and Engineering
Politecnico di Milano

Table of Contents

1. Template Attacks
2. Profiled SCAs: Machine Learning
3. Feature selection

Table of Contents

1 Template Attacks

2 Profiled SCAs: Machine Learning

3 Feature selection

Profiled Side-Channel Attacks (SCAs)

Overview

- A powerful class of SCAs designed to attack targets with minimal measurements (even down to a single one).
- Overcome countermeasures that restrict the number of measurements an attacker can acquire.
- Involve two main phases:
  1. **Profiling Phase:** Attacker builds an accurate model of the device's side-channel behavior.
  2. **Exploitation Phase:** The model is used to extract the unknown key from the target device.

Profiled SCAs: The Template Attack Advantage

The Profiling Advantage

- Performed on a device *identical* to the target, but fully controlled by the attacker.
- This allows setting the secret key value at will and performing unlimited measurements.
- Enables deriving a "perfect" side-channel behavior or "profile" for every possible value of the key bits.

Core Idea of Template Attacks

- TAs are considered the **most effective profiled attack** in an information-theoretic sense.
- They can, in principle, defeat masking and hiding countermeasures if enough measurements are collected during profiling.

Template Attacks: Statistical Modeling

Modeling Side-Channel Traces

- Each side-channel measurement trace is assumed to be affected by **additive Gaussian noise**.
- Each trace $\hat{T}^{(k_i)}$ (for a specific key bit k_i) is modeled as a random vector variable $T^{(k_i)}$.
- $T^{(k_i)}$ is assumed to follow a **multivariate Gaussian distribution**: $\mathcal{N}(\mu^{(k_i)}, \Sigma^{(k_i)})$.
  - $\mu^{(k_i)}$: Mean vector (average trace for key k_i).
  - $\Sigma^{(k_i)}$: Covariance matrix (describes how data points vary together for key k_i).
- The mean vector (μ) and covariance matrix (Σ) are computed over selected **points of interest** (POIs) in the trace, where significant leakage occurs.

Template Attack: Multivariate Gaussian PDF

The Probability Density Function (PDF)

For a trace x (composed of selected POIs) belonging to a key hypothesis k_i, its probability density function is given by:

$$\Pr(T^{(k_i)} = x) = \frac{\exp\left(-\frac{1}{2}(x - \mu^{(k_i)})^T (\Sigma^{(k_i)})^{-1}(x - \mu^{(k_i)})\right)}{\sqrt{(2\pi)^n \det(\Sigma^{(k_i)})}}$$

- n: Number of selected points of interest (dimension of the trace vector x).
- $\mu^{(k_i)}$: Mean vector for key hypothesis k_i.
- $\Sigma^{(k_i)}$: Covariance matrix for key hypothesis k_i.

Template Attack: Profiling Phase - Building Templates

Generating Templates

- The attacker, on their controlled device, sets a specific key bit value k_i (e.g., $k_i = 0$ or $k_i = 1$).
- For each possible value of k_i, a large number of side-channel traces are collected (from the POIs).
- From these measurements, *sample estimates* of the mean vector ($\hat{\mu}^{(k_i)}$) and covariance matrix ($\hat{\Sigma}^{(k_i)}$) are derived.
- These statistical models ($\hat{\mu}^{(k_i)}, \hat{\Sigma}^{(k_i)}$) form the **templates** for each key bit value.

Template Attack: Profiling Phase

Example: Targeting a Key Portion

- If targeting an 8-bit key byte, an attacker might build $2^8 = 256$ templates (one for each possible byte value).
- Alternatively, using a divide-and-conquer approach, they could target each bit individually, building two templates (for '0' and '1') for each bit.

Optimization: Grouping by Leakage Model

Instead of modeling the raw key value, we can build templates based on a chosen **leakage model**. This groups key values that are expected to produce similar side-channel leakage.

Common Leakage Models for Grouping

- **Hamming Weight (HW) Model:** Groups keys based on the number of '1's in a sensitive intermediate value (e.g., an S-box output). This reduces 256 key values to just 9 groups.
- **Hamming Distance (HD) Model:** Groups keys based on the number of bits that flip between two consecutive states.
- **Other models** can also be used, depending on the device's specific physical characteristics.

By grouping key hypotheses that are "close" in the leakage space, we drastically reduce the number of templates that need to be built, making the profiling phase far more practical.

Building the Template: The Mean Vector

Once the Points of Interest (POIs, more on them later) have been selected, we can build the statistical model for each key hypothesis k.

Step 1: Calculate the Mean Vector ($\hat{\mu}^{(k)}$)

- For a given key hypothesis k, collect a large number of profiling traces (T_k).
- For each selected POI s_i, calculate the average power consumption across all T_k traces:

$$\hat{\mu}_i^{(k)} = \frac{1}{T_k} \sum_{j=1}^{T_k} t_{j,s_i}$$

  where t_{j,s_i} is the measurement at POI s_i in trace j.
- The mean vector is simply the collection of these average values for all POIs.

$$\hat{\mu}^{(k)} = [\hat{\mu}_1^{(k)}, \hat{\mu}_2^{(k)}, \ldots, \hat{\mu}_n^{(k)}]^T$$

Building the Template: The Covariance Matrix

Step 2: Calculate the Covariance Matrix ($\hat{\Sigma}^{(k)}$)

- The covariance matrix captures how the POIs vary together.
- For each pair of POIs $(s_i, s_{i'})$, calculate the covariance:

$$c_{i,i'}^{(k)} = \frac{1}{T_k - 1} \sum_{j=1}^{T_k} (t_{j,s_i} - \hat{\mu}_i^{(k)})(t_{j,s_{i'}} - \hat{\mu}_{i'}^{(k)})$$

- The full $n \times n$ covariance matrix is constructed from these values. The diagonal elements $(c_{i,i})$ are the variances of each individual POI.

$$\hat{\Sigma}^{(k)} = \begin{pmatrix} c_{1,1} & c_{1,2} & \cdots \\ c_{2,1} & c_{2,2} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

The pair $(\hat{\mu}^{(k)}, \hat{\Sigma}^{(k)})$ forms the complete template for key hypothesis k.

Temlate Attacks: Matching Phase

Key Recovery Steps

- A single (or few) side-channel trace(s) $\hat{T}$ is acquired from the target device, where the key value is unknown
- Measurement conditions must be identical to the profiling phase
- For each possible key bit value, the likelihood of the acquired trace $\hat{T}$ belonging to that template's distribution is evaluated
- This is done by calculating the **a-posteriori probability** $\Pr(k_i|\hat{T})$, typically using Bayes' theorem:

$$\Pr(k_i|\hat{T}) = \frac{\Pr(\hat{T}|k_i) \cdot \Pr(k_i)}{\sum_j \Pr(\hat{T}|k_j) \cdot \Pr(k_j)}$$

- The key bit value k_i that maximizes this a-posteriori probability is selected as the most likely secret key bit.

Template Attacks

Statistical Efficacy

- The assumption of multivariate Gaussian distributions for power consumption has proven to be a very good practical approximation.
- Statistical tests are used to determine the "goodness of fit" of a model to the actual device behavior, accounting for noise.

Table of Contents

1 Template Attacks

2 Profiled SCAs: Machine Learning

3 Feature selection

Machine Learning for Profiled SCAs

Beyond Statistical Templates

- Machine Learning (ML) techniques offer an alternative to traditional Template Attacks.
- Instead of calculating probabilities, ML uses a **classification step** to identify the secret information
- A key advantage: ML provides a **nonparametric and data-driven approach**, removing assumptions about the statistical distribution of the leakage (like the Gaussian assumption in TAs)

Support Vector Machines

Support Vector Machines (SVMs)

- SVMs are a widely used and effective choice for profiled side-channel attacks.
- Their goal is to find an optimal hyperplane that best separates data points belonging to different classes (e.g., traces for key bit '0' vs. key bit '1')
- This hyperplane acts as a decision boundary

Support Vector Machines (SVMs): Detailed Flow

Profiling Phase (Training)

- Using the controlled device, the attacker collects many labeled side-channel traces (traces paired with their known key bit values)
- The SVM algorithm learns from this labeled data to compute the optimal hyperplane
- This process involves mapping the traces into a (potentially) higher-dimensional space where they become linearly separable, making it easier to find the separating hyperplane

Support Vector Machines (SVMs): Detailed Flow

Classification Phase

- A new, unlabeled side-channel trace is acquired from the target device (where the key is unknown)
- This new trace is then input into the trained SVM model
- The SVM classifies the trace by determining which side of the learned hyperplane it falls on, thereby predicting the most likely key bit value

Table of Contents

1 Template Attacks

2 Profiled SCAs: Machine Learning

3 Feature selection

Feature Selection: A Crucial Preprocessing Step

Why Feature Selection is Needed

- Side-channel traces can contain hundreds or thousands of samples (time points)
- Too many samples can introduce noise, increase computational complexity, and lead to numerical instability, especially for machine learning algorithms and covariance matrix inversion in Template Attacks

Feature Selection: Common Techniques

Feature selection aims to select a subset of trace samples that contain the most sensitive leakage information, often explicitly identifying them as **Points of Interest (POIs)**, or to transform the data into a smaller, more meaningful set of **features**

This process helps to filter out redundant or uncorrelated information, making the profiling and exploitation phases more efficient and effective.

- **Maximum Variance:** Selecting samples that show the most variation across traces
- **Sum of Squares of t-difference (SOST):** Identifies samples where the difference in means between classes (e.g., key hypotheses) is most statistically significant
- **Principal Component Analysis (PCA):** A dimensionality reduction technique that transforms original, correlated variables into a new set of uncorrelated variables (principal components), ordered by the amount of variance they explain

Sum-of-Differences

Instead of relying on complex statistics, we can use a simple and intuitive method to find the best Points of Interest (POIs).

Identify the exact moments in time where the side-channel traces for different key hypotheses are **most different** from each other.

This method involves three main steps, which we will now cover one by one.

Feature Selection: The Sum-of-Differences Method

A straightforward way to find the most interesting points in a trace is to find where the average traces for different key hypotheses **differ the most**.

Step 1: Calculate Average Traces

For each key hypothesis k (or leakage model group), calculate its average trace, M_k. For each sample point i in the trace, this is:

$$M_{k,i} = \frac{1}{T_k} \sum_{j=1}^{T_k} t_{j,i}$$

where T_k is the number of traces for hypothesis k.

Feature Selection: The Sum-of-Differences Method

Step 2: Create a Master Difference Trace

Create a single "master trace," D, by summing the absolute differences between all pairs of the average traces at each sample point i:

$$D_i = \sum_{k_1, k_2} |M_{k_1,i} - M_{k_2,i}|$$

From Difference Trace to Points of Interest

The master difference trace, D, will have large peaks at the time samples where the leakage is most pronounced. These peaks are our candidate Points of Interest (POIs).

Step 3: Pruning the Peaks

It is crucial to select points that are not only significant but also **separated in time**. This ensures they capture different aspects of the computation and improves the stability of the covariance matrix.

- A simple "pruning" method is to select a peak and then eliminate its nearest neighbors from consideration.
- Repeat until the desired number of POIs (e.g., 3 to 5) is reached.

The final set of pruned peaks gives us the vector of features to be used for building our templates.