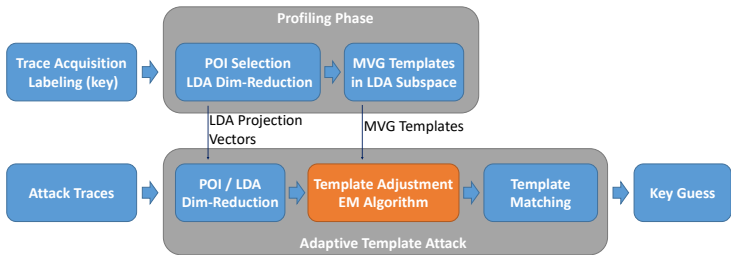
Adaptive template attack

Eric Chun-Yu Peng, Markus G. Kuhn (University of Cambridge)
cyp24@cl.cam.ac.uk, mgk25@cl.cam.ac.uk

UNIVERSITY OF CAMBRIDGE

The *adaptive template attack* [1] aims to improve the cross-device template accuracy by modifying the template models to fit the target device's leakage signal. Template attacks usually start with a profiling phase that builds multivariate Gaussian (MVG) leakage models of intermediate values in a cryptographic computation. They then use those models to classify leakage signals observed in an attack phase and infer intermediate values from that. However, this attack may not be practical as templates built on a profiling device may not match the actual leakage of an attacked device well enough and the attacked device may not be available for profiling.



We present here a new way of improving the template portability by combining *supervised* learning (MVG template building) with an *unsupervised* learning technique (template adjustment). Our processing pipeline starts conventionally, with point selection and LDA-based dimensionality reduction, to build the initial templates for a profiling device. We then reuse the same dimensionality-reducing projection in the attack phase, but follow it with the Expectation–Maximization (EM) algorithm for template adjustment. The EM algorithm is initialized with the template parameters from the profiling device, and then fed with unlabeled traces from the attacked device. This should tailor the resulting model to better fit the target device's leakage model.

The binomial sampler of CRYSTALS-KYBER turned out to be an attractive target, as it outputs a sequence of symbols chosen from one of five values, i.e. $K = \{-2, -1, 0, 1, 2\}$, resulting in an efficient parameter estimation problem. At the same time, this target has very high accuracy requirements for the classifier, as KYBER samples several hundred such values each time.

Table 1: Required template accuracy of the desired attack success rate (SR).

	50 % SR	90 % SR	99 % SR
AES-128 (16 bytes)	95.76 %	99.344 %	99.9372 %
AES-256 (32 bytes)	97.85 %	99.671 %	99.9686 %
Kyber-512	99.86 %	99.979 %	99.9980 %
Kyber-768	99.91 %	99.986 %	99.9986 %
Kyber-1024	99.93 %	99.990 %	99.9999 %

Adaptive templates and EM algorithm

When adjusting the profiled templates, we use the EM algorithm to characterize the attacked trace set in the LDA subspace as a Gaussian mixture model (GMM). This idea is similar to the Cross-Device Profiled Attack (CDPA) introduced by Cao [2], which adds a fine-tuning step to let their classifier neural network learn the target's leakage model. In our experiments, we observed that the commonly used dimensionality reduction technique (LDA) can produce well-separated data groups across different devices. Still, the projected attack data often drifted away from the MVG profiled template models due to cross-device signal variance, as shown in Figure 1(a). By employing the EM algorithm to adjust the templates, we can vastly reduce this model-data discrepancy.

GMM Modeling. A Gaussian mixture model is composed of $|K|$ Gaussian components θ_i, each with a mixture weight π_i that sums up to 1.

$|K|$: Number of Gaussian components, i.e. $K = \{k_1, \ldots, k_M\}$

$\theta_{i \in K}$: Gaussian components' parameters, i.e. $\theta_i = (\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$

$\pi_{i \in K}$: The mixture weight of the Gaussian components

In the profiling phase, we use the multivariate Gaussian model to characterize the traces of different intermediate values. By combining all the MVG templates, we can form a Gaussian mixture $\Theta = \{(\pi_{k_1}, \theta_{k_1}), \ldots, (\pi_{k_M}, \theta_{k_M})\}$ to characterize a target device's leakage. The weights π_i of the Gaussian components are governed by the targeted intermediate variable's probability distribution. For instance, the binomially distributed secret key variable S in CRYSTALS-KYBER will have the mixture weights $\Theta_S = \{(0.0625, \theta_{-2}), (0.25, \theta_{-1}), (0.375, \theta_0), (0.25, \theta_1), (0.0625, \theta_2)\}$.

EM Algorithm. The EM algorithm is useful in parameter estimation or to infer latent variables whose values are unknown. By iteratively updating the parameters to maximize the likelihood of the observed data while simultaneously re-estimating the probability distribution of the latent variables, EM facilitates parameter estimation for GMM.

E-step: Calculate the posterior distribution of the latent variable (intermediate variable) S, with the given attacked trace set $\boldsymbol{T}_\mathrm{a} = \{\boldsymbol{t}_1, \ldots, \boldsymbol{t}_N\}$ and GMM $\Theta^{(l-1)}$:

$$r_{i,j}^{(l)} = \frac{\pi_i p(\boldsymbol{t}_j | \theta_i^{(l-1)})}{\sum_{m \in K} \pi_m p(\boldsymbol{t}_j | \theta_m^{(l-1)})}$$

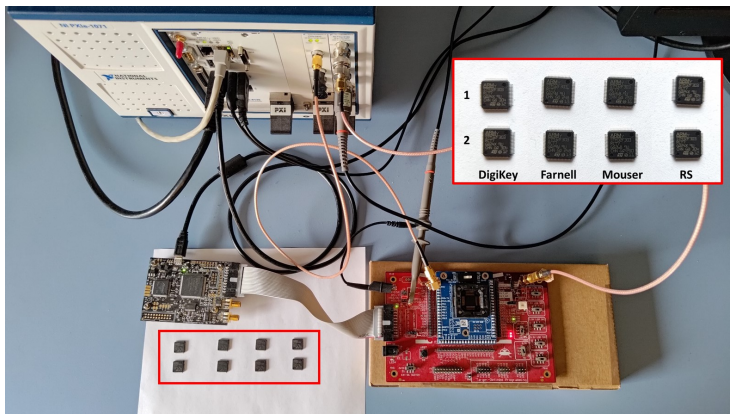
M-step: Find the optimal Θ by maximizing the expectation of the likelihood with respect to the posterior latent variable distribution from the E-step. Note that, since the Gaussian components' weights π_i are set by the probability distribution of the targeted intermediate variable as a known GMM parameter, we keep the π_i fixed during the M-step:

$$\boldsymbol{\mu}_i^{(l)} = \frac{\sum_{j=1}^{N} r_{i,j}^{(l)} \boldsymbol{t}_j}{\sum_{j=1}^{N} r_{i,j}^{(l)}} \qquad \pi_i = \begin{cases} 0.0625, & i \in \{-2, 2\} \\ 0.25, & i \in \{-1, 1\} \\ 0.375, & i \in \{0\} \end{cases}$$

$$\boldsymbol{\Sigma}_i^{(l)} = \frac{\sum_{j=1}^{N} r_{i,j}^{(l)} (\boldsymbol{t}_j - \boldsymbol{\mu}_i^{(l)})(\boldsymbol{t}_j - \boldsymbol{\mu}_i^{(l)})^\top}{\sum_{j=1}^{N} r_{i,j}^{(l)}}$$

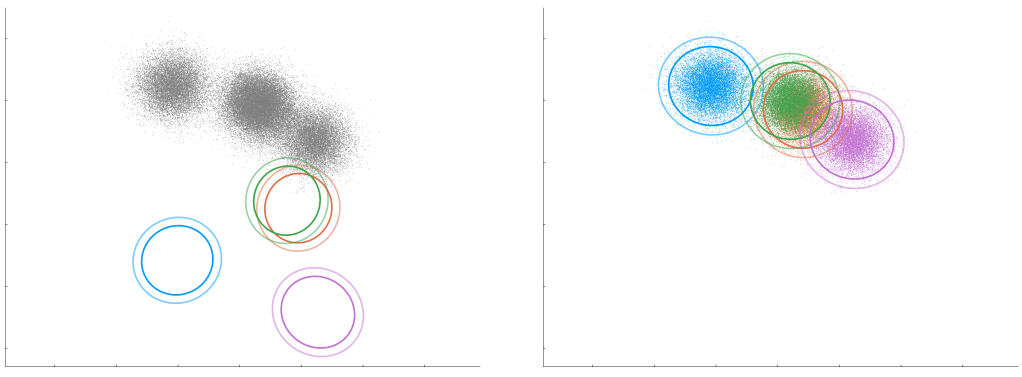
Measurement setup

We target the CRYSTALS-KYBER implementation from the pqm4 crypto library, compiled for ARM Cortex-M4 with `gcc -Os`. The target hardware setup uses a ChipWhisperer CW308T-STM32F-SOCKET board, which allows us to easily swap eight STM32F303RCT7 devices within the same measurement setup. We bought these MCUs from four different component distributors (DK, FN, MS, RS) to obtain samples of different batches with different manufacturing dates and thus ensure a wider spread in manufacturing variation. We supplied the targets with a 5 MHz external clock signal phase-locked with a 10-bit oscilloscope (NI PXIe-5160) connected across a 10-ohm resistor in the VCC line via a 5 MHz high-pass filter.



Result

Figure 1(a) demonstrates the cross-device mismatch between profiling-device templates and attack-device traces before the EM adjustment. The gray scattered dots are the attack trace set $\boldsymbol{T}_\mathrm{a}$ in the LDA subspace, and the multivariate Gaussian templates are shown as ellipses, color-coded by their respective intermediate values. Not only does $\boldsymbol{T}_\mathrm{a}$ have a systematic shift from the templates, but the relative position between different data groups is also shifted. Figure 1(b) illustrates the effect of our EM-based template adjustment. The adjustment relocated the templates and still identifies their respective data group with a well-fitted Gaussian model.



(a) Profiled templates. (b) EM adjusted templates.

Figure 1: Attacked traces and EM-based template adjustment in LDA subspace.

Table 2 lists the single-trace attack success rates on the KYBER binomial sampler target in a cross-device attack scenario. The left side of the table shows poor template portability with classical LDA-based templates, where most templates did not achieve a single-trace attack across different devices. In contrast, our EM-adjusted templates achieved a 100 % success rate for most cross-device attack scenarios. The result demonstrates the effectiveness of our adaptive template attack technique.

Table 2: Single-trace attack success rate of the binomial sampler in KYBER768.`KenGen`.

Profiling \ Target	DK1	DK2	FN1	FN2	MS1	MS2	RS1	RS2	DK1	DK2	FN1	FN2	MS1	MS2	RS1	RS2
	Template Attack								Adaptive Template Attack							
DK1	100.0%	99.8%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	100.0%	100.0%	100.0%	100.0%	100.0%	80.2%	100.0%	100.0%
DK2	94.1%	100.0%	0.0%	0.0%	0.4%	0.0%	25.1%	70.7%	100.0%	100.0%	100.0%	100.0%	100.0%	2.0%	44.7%	100.0%
FN1	0.0%	0.0%	100.0%	84.6%	12.6%	0.0%	0.0%	0.0%	100.0%	100.0%	100.0%	100.0%	100.0%	28.3%	100.0%	100.0%
FN2	0.0%	0.0%	87.6%	100.0%	74.1%	53.4%	0.0%	0.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
MS1	0.0%	0.2%	0.4%	86.1%	100.0%	92.2%	0.5%	14.2%	100.0%	100.0%	100.0%	100.0%	100.0%	99.9%	100.0%	100.0%
MS2	0.0%	0.0%	0.0%	41.2%	26.7%	100.0%	0.0%	0.4%	100.0%	97.2%	100.0%	100.0%	100.0%	100.0%	99.9%	100.0%
RS1	1.1%	29.0%	0.0%	0.0%	0.0%	18.7%	100.0%	99.9%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
RS2	0.0%	22.3%	0.0%	0.0%	0.0%	1.5%	99.9%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%

Conclusion

We have shown that by treating the templates as a Gaussian mixture model, we can use the EM algorithm to improve model fitness to the attacked device and increase cross-device template accuracy. This adjustment method allows us to reuse the LDA projection vectors, and build high-quality templates for an attacked device with unlabeled traces. Such portability improvement is crucial when dealing with a large number of subkeys, for example in post-quantum systems, like CRYSTALS-KYBER.

References

[1] Eric Chun-Yu Peng, Markus G. Kuhn: Adaptive Template Attacks on the Kyber Binomial Sampler. TCHES 2025(3), DOI: 10.46586/tches.v2025.i3.470-492

[2] Pei Cao, Chi Zhang, Xiangjun Lu, Dawu Gu: Cross-device profiled side-channel attack with unsupervised domain adaptation. TCHES 2021(4), DOI: 10.46586/tches.v2021.i4.27-56