





Decimated in Linear Time; Single Power Trace, Full Key Recovery Attack on Toeplitz Hash Privacy Amplification

Niall Canavan^{*} , Tuan Hoang[†] , Ayesha Khalid[‡]  and Máire O'Neill[§] 

Centre for Secure Information Technologies (CSIT), Queen's University Belfast, UK

Email: ^{*}ncanavan815@qub.ac.uk, [†]t.hoang@qub.ac.uk, [‡]a.khalid@qub.ac.uk, [§]m.oneill@ecit.qub.ac.uk

Abstract—We demonstrate a template power side channel attack that recovers the full secret key on an implementation of a radix-2 Decimation in Time Fast Fourier Transform (DIT-FFT) optimised Toeplitz Hashing based privacy amplification, commonly used in quantum key distribution. The duration of the attack is linear in time with respect to the input key length when the target platform is an ARM Cortex-M4 and is successful with a single power trace. We demonstrate that key leakage is present in an Artix-7 FPGA power trace, but high noise makes a single trace attack difficult.

Index Terms—Privacy Amplification, Toeplitz Hashing, Side Channel Analysis, Power Analysis, Quantum Key Distribution

I. INTRODUCTION

Quantum Key Distribution (QKD) has been proposed as a solution to the threat that quantum computers pose to classical cryptography [1]. In theory, QKD protocols are secure as long as the laws of quantum mechanics as we know them remain true. However, in practice an adversary can exploit imperfections in QKD implementations to attack the protocol and recover secret key material [2].

Implementation security in the quantum phase of QKD (quantum state generation, transmission and measurement) is relatively well studied [3], [4], while the classical phase (device control and post processing) receives less attention. This discrepancy can be attributed to the common assumption that the classical processing devices are themselves secure and protected from attack in the QKD device. ETSI frames this as: the QKD device is contained within a security perimeter in which no adversary can enter [5]. In this work, we consider attacks that are possible if the security perimeter is broken. A break in the perimeter can be as obvious as an attacker opening the QKD device, or more subtle, through covert channels [6]. In this case, then it is necessary to understand points of vulnerability in the classical phase such that possible attacks are known and security measures may be put in place to add a level of redundancy to the security perimeter notion. To this end, we investigate an implementation of the final classical post-processing step, Privacy Amplification (PA), under a power side channel analysis (SCA) attack.

A. Toeplitz Hashing Based Privacy Amplification

PA is carried out symmetrically between two genuine parties (Alice and Bob), such that an eavesdropper (Eve) has negligible information on their identical shared private key, \mathbf{K}_{PA} . From the previous steps of QKD Alice and Bob derive an identical, error corrected key, \mathbf{K}_{EC} , which is n bits long. Eve derives a correlated key \mathbf{K}_{Eve} , which

has e bits of information on \mathbf{K}_{EC} . In terms of Shannon entropy, $H(\mathbf{K}_{EC}|\mathbf{K}_{Eve}) \geq n - e$ is Eve's uncertainty on \mathbf{K}_{EC} . It is then the aim of PA to apply some function, $f : \{0, 1\}^n \rightarrow \{0, 1\}^r$, such that $H(\mathbf{K}_{PA}|f, \mathbf{K}_{Eve}) = r$, where $f(\mathbf{K}_{EC}) = \mathbf{K}_{PA}$. That is, Eve cannot distinguish \mathbf{K}_{PA} from random despite knowing \mathbf{K}_{Eve} and f .

Toeplitz matrix based PA schemes are among the most popular in QKD [7]. In this class of schemes, a Toeplitz matrix, $\mathbf{T}(\mathbf{S})$, is constructed from a random seed vector, $\mathbf{S} \in \{0, 1\}^{n+r}$. $\mathbf{T}(\mathbf{S})$ is a diagonal constant matrix of the form:

$$\mathbf{T}(\mathbf{S}) = \begin{pmatrix} s_r & s_{r+1} & s_{r+2} & \dots & s_{r+n} \\ s_{r-1} & s_r & s_{r+1} & \dots & s_{r+n-1} \\ s_{r-2} & s_{r-1} & s_r & \dots & s_{r+n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ s_3 & s_4 & s_5 & \dots & s_{n+2} \\ s_2 & s_3 & s_4 & \dots & s_{n+1} \\ s_1 & s_2 & s_3 & \dots & s_n \end{pmatrix} \quad (1)$$

where s_i is the i^{th} element from \mathbf{S} . The seed \mathbf{S} is generated by Alice or Bob and shared to the other over a public channel. Therefore, Eve also knows \mathbf{S} , and by extension, $\mathbf{T}(\mathbf{S})$. Privacy amplification ($H(\mathbf{K}_{PA}|f, \mathbf{K}_{Eve}) = r$) is achieved when we consider f to be *hashing* \mathbf{K}_{EC} upon $\mathbf{T}(\mathbf{S})$. This hashing operation is simply the matrix vector multiplication in eq.2:

$$\mathbf{K}_{PA} = \mathbf{T}(\mathbf{S})\mathbf{K}_{EC} \quad (2)$$

If I_{pwr} is information leaked from the power consumption of the device performing PA, from which l bits of \mathbf{K}_{EC} can be derived, then Eve's uncertainty on \mathbf{K}_{EC} is $H(\mathbf{K}_{EC}|I_{pwr}, f, \mathbf{K}_{Eve}) = n - e - l$. If $l = n - e$, Eve can fully recover \mathbf{K}_{EC} , and from there she can derive \mathbf{K}_{PA} . Alternatively, if the power consumption leaks information on \mathbf{K}_{PA} , then Eve's uncertainty on \mathbf{K}_{PA} is $H(\mathbf{K}_{PA}|I_{pwr}, f, \mathbf{K}_{Eve}) = r - l$. If $l = r$, \mathbf{K}_{PA} is fully revealed to Eve. Such leakage scenarios are catastrophic for the theoretical security guarantees set out by QKD.

B. Related Work

To the best of our knowledge, there has been no investigation into the power leakage of any PA scheme. However, Nikiforov et al. investigate cache side channels of PA schemes, where they found no secret key information from cache side channels on an Intel Skylake architecture [8]. More broadly in QKD classical phase implementation

security, the error correction step has received some attention. Canavan et al. [9] present a theoretical power analysis attack upon the Cascade error correction scheme [10], which builds upon a practical vulnerability, uncovered by Kim et al. [11], against a generic parity checking algorithm. The LDPC error correction algorithm is also the subject of an investigation by Park et al. [12], where the full key was successfully recovered in a practical power analysis attack.

The review carried out by Luo et al. [7] provides a detailed discussion of the current state of the art of QKD post processing, including PA implementations. One take away from the study is that Toeplitz hashing based methods are the most popular scheme. In high performance Toeplitz schemes, matrix-vector multiplication optimisations are always used such as the Fast Fourier Transform (FFT), Number Theoretic Transform (NTT) or Linear Shift Feedback Registers (LSFR). The FFT is the most common optimisation.

In Post Quantum Cryptography (PQC), implementation security of algorithms is a major concern. To this end, there have been investigations around the security of the NTT, [13], [14], used throughout ML-KEM [15] and ML-DSA [16]. The FFT is less commonly seen in cryptographic applications, but is featured in National Institute of Standards and Technology (NIST) backed Falcon algorithm. Successful attacks have been demonstrated, [17], which targets floating point multiplications in the FFT domain.

C. Contributions

In section.II of this paper we give background information and a description of the Decimation in Time Fast Fourier Transform (DIT-FFT) optimisation commonly used in PA. Section.III gives a detailed description our attack, which aims to recover a full secret key, \mathbf{K}_{PA} , from a single power trace. Finally, section.IV evaluates the performance of this attack on MCU/FPGA platforms (ARM Cortex-M4 & Artix-7 FPGA resp). Section.V discusses future work and conclusions.

II. BACKGROUND

In practice, calculating eq.2 using standard matrix-vector multiplication is an $O(n^2)$ operation. This performance scaling is unacceptable for realistic QKD key sizes. Fortunately, if $\mathbf{T}(\mathbf{S})$ is a *circulant* matrix (a special subclass of Toeplitz matrices, where each row is a right shift of the previous by one index, where the first value is equal to the last value from the previous row) performance optimisations exist. If $\mathbf{T}(\mathbf{S})$ is not already circulant, there is a trivial transformation for which the correctness and security of PA is preserved. We denote the circulant matrix for a seed \mathbf{S} as $\mathbf{C}(\mathbf{S})$. Both \mathbf{K}_{EC} and $\mathbf{C}(\mathbf{S})$ can be projected into the Fourier domain via a Discret Fourier Transform (DFT), where pointwise multiplication between projections is equivalent to matrix vector multiplication in the original domain. The result is projected back to the original domain to give \mathbf{K}_{PA} . This approach scales in $O(n \log n)$.

A. DFT in Toeplitz Hashing

Circulant matrices satisfy $\mathbf{C}(\mathbf{S}) = \mathbf{F}^{-1} \text{diag}(\mathbf{F}\mathbf{S})\mathbf{F}$, where \mathbf{S} is the seed and also the first row of $\mathbf{C}(\mathbf{S})$, \mathbf{F} is the DFT matrix and $\text{diag}(\mathbf{v})$ is the square diagonal matrix of entries from a vector \mathbf{v} , non diagonal elements are 0.

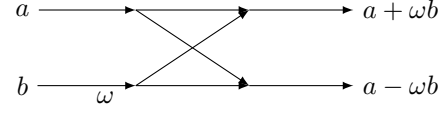


Figure 1: Radix-2 Butterfly operation on inputs a and b , with twiddle factor ω . We denote the operation as $B(a, b, \omega) = (a + \omega b, a - \omega b)$.

Moreover, multiplying \mathbf{F} by a vector, \mathbf{x} , is equivalent to computing the DFT of \mathbf{x} . Using this, eq.3 demonstrates how to compute $\mathbf{C}(\mathbf{S})\mathbf{x}$ efficiently, where $\text{DFT} \equiv \mathbf{F}$, and the Inverse Discrete Fourier transform, $\text{IDFT} \equiv \mathbf{F}^{-1}$.

$$\begin{aligned} \mathbf{C}(\mathbf{S})\mathbf{x} &= \mathbf{F}^{-1} \text{diag}(\mathbf{F}\mathbf{S})\mathbf{F}\mathbf{x} \\ &= (\mathbf{F}^{-1}) \text{diag}(\mathbf{F}\mathbf{S})(\mathbf{F}\mathbf{x}) \\ &= \text{IDFT}(\text{diag}(\text{DFT}(\mathbf{S}))\text{DFT}(\mathbf{x})) \quad (3) \\ &= \text{IDFT}(\text{diag}(\mathbf{v})\mathbf{y}) \\ &= \text{IDFT}(\mathbf{v} \circ \mathbf{y}) \end{aligned}$$

In other words, calculating $\mathbf{C}(\mathbf{S})\mathbf{x}$ is equivalent to finding $\mathbf{v} = \text{DFT}(\mathbf{S})$ and $\mathbf{y} = \text{DFT}(\mathbf{x})$, taking the pointwise product (\circ) of \mathbf{v} and \mathbf{y} and finally computing the inverse DFT on this result. In practice, the DFT/IDFT algorithm used is actually the FFT/IFFT (or NTT/INTT).

The more popular choice of FFT is reflected in the Toeplitz Hashing algorithm, Alg.1, where n is the input key length, r is the output key length ($n > r$), \mathbf{K}_{EC} is the original key, \mathbf{S} is the public seed and \mathbf{K}_{PA} is the hashed output key.

Algorithm 1: FFT Toeplitz Hash

```

1 Input:  $\mathbf{K}_{EC}, \mathbf{S}, n, r$ 
2 Output:  $\mathbf{K}_{PA}$ 
3  $\mathbf{v} \leftarrow \text{FFT}(\mathbf{S})$ 
4  $\mathbf{y} \leftarrow \text{FFT}(\mathbf{K}_{EC})$ 
5  $\mathbf{u} \leftarrow \mathbf{v} \circ \mathbf{y}$ 
6  $\mathbf{K}_{PA} \leftarrow \text{IFFT}(\mathbf{u}) \bmod 2$ 
```

B. Decimation In Time FFT

The Decimation In Time FFT (DIT-FFT) variation of the FFT algorithm computes FFT coefficients in place, iteratively, whereas the recursive variant (textbook version) requires significantly more memory overhead. Hence, DIT-FFT is universally found in high-performance requirement situations, such as QKD-PA [7].

The foundation of the radix-2 DIT-FFT is the radix-2 butterfly operation, represented graphically in Fig.1. The butterfly operation takes as input three complex numbers, a , b and ω . The values of a and b are arbitrary, while ω is a complex root of unity ($\omega_q^p = e^{-2\pi i p/q}$). The results, $a + \omega b$ and $a - \omega b$, are stored in the same memory location as the inputs a and b were retrieved from respectively, which is memory efficient.

Many butterfly operations make up the entire DIT-FFT procedure. Fig.2 shows how an input, $\mathbf{x} = \{x_0, \dots, x_7\}$, is projected into the FFT domain as $\mathbf{X} = \{X_0, \dots, X_7\}$. Generally, the entire procedure consists of $\log_2(n)$ stages,

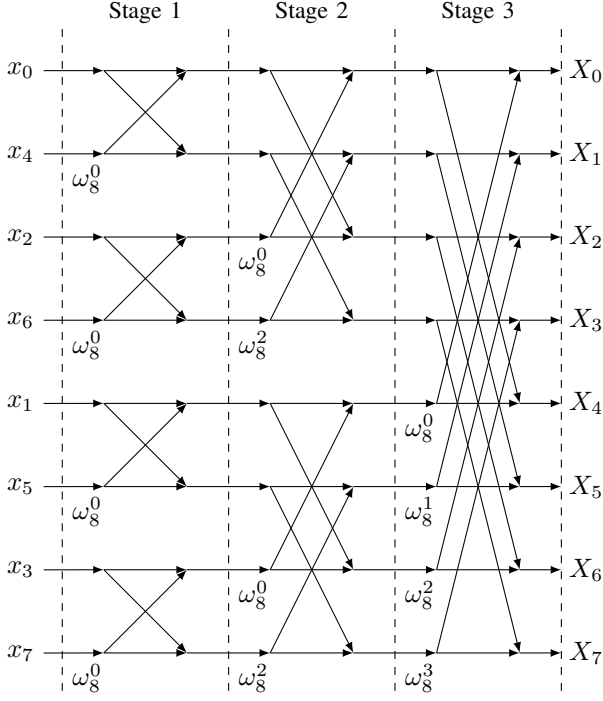


Figure 2: 8-point radix-2 DIT-FFT diagram for input $x = \{x_0, \dots, x_7\}$ and output $X = \{X_0, \dots, X_7\}$

each containing $n/2$ butterfly operations, where n is the input length and n is a power of 2. The mathematical reasoning behind how the butterfly structure in Fig.2, results in a valid FFT domain projection is given by Cooley and Tukey [18].

III. PROPOSED ATTACK

In Alg.1, lines 4, 5 & 6 present themselves as potential targets for attack due to their direct operation over the secret key, \mathbf{K}_{EC} , intermediate FFT domain secret states, \mathbf{y} & \mathbf{u} , as well as the output key \mathbf{K}_{PA} . If any of these states can be recovered, the final key is either immediately available or can be trivially calculated.

At a high level, the attack we develop in this work focuses on line 4, where \mathbf{K}_{EC} is converted into the FFT domain. The aim of the attack is to recover \mathbf{K}_{EC} by observing the power consumption side channel [19], where the adversary hopes that I_{pwr} gives $H(\mathbf{K}_{EC}|I_{pwr}) < n$, and ideally $H(\mathbf{K}_{EC}|I_{pwr}) = 0$. Note that we simplify the considerations for this attack by assuming that the attacker uses information from I_{pwr} only to recover the key, i.e $e = 0$. An attack where $e > 0$ is possible and would certainly be advantageous for an adversary but is outside the scope of this study.

In particular, in this attack we focus on the first stage of butterfly operations (see Fig.2) for the \mathbf{K}_{EC} FFT projection (Alg.1, line 4). Recovery of the inputs to a single butterfly operation of this type results in the adversary knowing two secret key bits. If each of the stage one butterfly operation inputs can be recovered, the adversary knows \mathbf{K}_{EC} fully.

A. Threat Model

We make several assumptions about the adversary's capabilities for this attack. The adversary has the ability to measure the power consumption (power trace) of the

(a) IEEE 754 Representation					
a	b	$B(a, b, \omega)$	HW In	HW Out	HD
0	0	(0,0)	0	0	0
0	1	(1,-1)	7	15	8
1	0	(1,1)	7	14	7
1	1	(2,0)	14	1	13

(b) Q8.8 Fixed-Point Representation					
a	b	$B(a, b, \omega)$	HW In	HW Out	HD
0	0	(0,0)	0	0	0
0	1	(1,-1)	2	10	8
1	0	(1,1)	2	4	5
1	1	(2,0)	4	1	3

Table I: Full set of possible stage 1 butterfly input and outputs, with associated combined Hamming weights (HW) and Hamming distance (HD) for IEEE-754 floating point and Q8.8 fixed point representation

target QKD post-processing device (victim device) used by a genuine party. Victim device power traces are sent to the adversary for processing. However, the adversary has no control over the information processed on the victim device. The adversary is in possession of an identical device (adversarial device), of which they have arbitrary control of the information processed and power consumption measurements.

In a normal QKD operation, the PA algorithm is run only once per QKD key. Therefore, we assume that an adversary can measure only one power trace per $(\mathbf{K}_{EC}, \mathbf{S})$ on the victim device. However, to assess information leakage more generally, we will consider the case where an arbitrary number of measurements on the victim device is possible.

B. Butterfly Operation Vulnerability

The input to the DIT-FFT, \mathbf{K}_{EC} , is binary, where the bit values are assumed to be in a uniform random distribution. Therefore, each individual stage 1 butterfly operation has four equiprobable secret key bit inputs: $(a, b) \in \{(0, 0), (0, 1), (1, 0), (1, 1)\}$. Moreover, in the first stage $\omega = 1$ is constant in each operation. Hence, there are only four possible computations for each of the target butterfly operations. Each of the four possible butterfly operations, $B(a, b, \omega = 1)$ are listed with their respective output in Table.I.

As the DIT-FFT operates on the set of complex real numbers \mathbf{K}_{EC} must be in an appropriate format, typically the IEEE-754 floating point standard (common on MCU architecture) or the $QM.N$ fixed point representation (common on FPGA architecture). When examining the Hamming distance between the input and output of $B(a, b, \omega = 1)$, in IEEE-754 and Q8.8 format, we find that each of the four inputs has a distinct Hamming distance. The results are captured in Table.I. Hamming distance is correlated to power consumption [20]. Therefore, we expect the average power consumption to be different for each of the four inputs.

C. Attack Description

We have developed an attack, described in Alg.2, which exploits hypothetical stage 1 butterfly operation leakages. First, the adversary records the power trace for Alg.1 on the victim device. We label this power trace as \mathbf{t} , which is input to Alg.2. From \mathbf{t} , the adversary isolates each section of the power trace that corresponds to individual butterfly

operations and retains these sections only. The adversary then has a list of sub-traces $\{\mathbf{t}_0, \mathbf{t}_1, \dots, \mathbf{t}_{n \log_2(n)/2}\}$ from \mathbf{t} , where it is assumed that butterfly operations are computed in top down order from Fig.2. For the currently developed attack, only the first $n/2$ sections that correspond to stage 1 butterfly operations are needed. The remaining sub-traces may be discarded (see line 3, Alg.2).

The adversary initialises their guess for the key, \mathbf{K}'_{EC} with all bit values as 0. Next, on the adversarial device, traces are gathered for all possible inputs to the first butterfly operation (the operation involving x_0 and x_4 in stage 1 in Fig.2). The possible key space for this segment is 4 keys, as in Table.I. The input target bit hypothesis values are given by $G_{1,2,3,4}$. For each of the i target butterfly operations, the adversary repeats the same process. Templates for each hypothesis, $\mathbf{T}_{1,2,3,4}$, are generated along with a score, $s_{1,2,3,4}$, for how well the template matches the target trace \mathbf{t}_i .

Templates are generated by recording and calculating an average power trace of the i^{th} butterfly operation over L samples for the current guess of \mathbf{K}'_{EC} (see Alg.2, lines 8 & 9). A statistical test comparing the template to the target trace results in the score s_j . We use a Pearson correlation test in our results.

The guess G_j , which yields the best score (the interpretation of the best depends on the comparison test used), is then assumed to be the correct bit values for \mathbf{K}'_{EC} . These correct values are then retained within the adversary's guess for \mathbf{K}'_{EC} on subsequent butterfly targets. This reduces the opportunity for incorrect key bits in \mathbf{K}'_{EC} , prior to the target bits, to negatively affect the quality of templates.

After each of the $n/2$ sub-traces have been examined, the adversary has a complete guess for the correct key which produced \mathbf{t} .

Algorithm 2: DIT-FFT Template Attack

```

1 Input:  $\mathbf{t}, n, L$ 
2 Output:  $\mathbf{K}'_{EC}$ 
3  $\{\mathbf{t}_0, \mathbf{t}_1, \dots, \mathbf{t}_{n/2}\} \leftarrow \mathbf{t}$ 
4  $\mathbf{K}'_{EC} \leftarrow \{0\}^n$ 
5  $G_1 \leftarrow (0, 0), G_2 \leftarrow (0, 1), G_3 \leftarrow (1, 0), G_4 \leftarrow (1, 1)$ 
6 for  $i$  in  $n/2$  do
7   for  $j$  in 4 do
8      $\mathbf{K}'_{EC} \leftarrow \text{setGuess}(\mathbf{K}'_{EC}, G_j, i)$ 
9      $\mathbf{T}_j \leftarrow \text{genTemplate}(\mathbf{K}'_{EC}, L, i)$ 
10     $s_j \leftarrow \text{compare}(\mathbf{T}_j, \mathbf{t}^i)$ 
11   $G_{best} \leftarrow \text{best}((G_1, s_1), (G_2, s_2), (G_3, s_3), (G_4, s_4))$ 
12   $\mathbf{K}'_{EC} \leftarrow \text{setGuess}(\mathbf{K}'_{EC}, G_{best}, i)$ 

```

IV. RESULTS

A. Experimental Setup

Power traces were collected and processed with the ChipWhisperer framework [21]. We used ChipWhisperer Husky to record traces from the CW312 (ARM Cortex-M4 MCU) and CW305 (Artix-7 XC7A100T FPGA) target boards. The same boards are used to simulate both the victim and adversarial devices. For the CW312 experiments we set a target clock rate of 7.36MHz and a sampling rate of

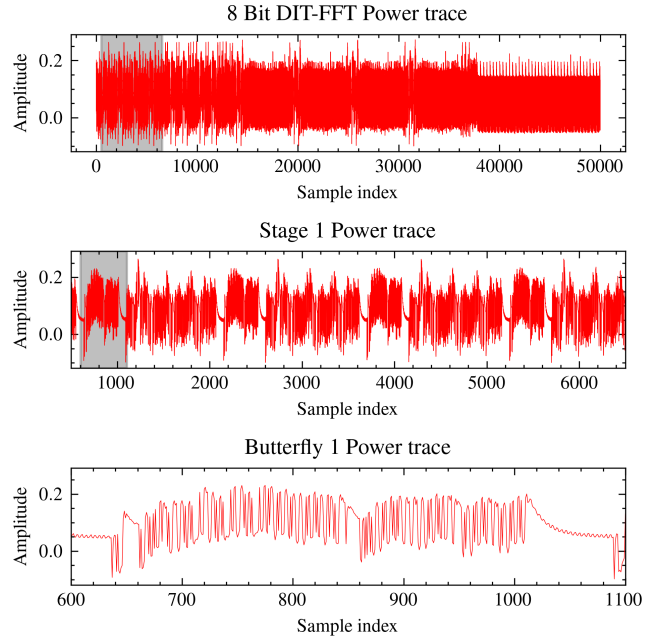


Figure 3: Full power trace of 8-bit DIT-FFT recorded on CW312 (top). The grey region in full trace corresponds to stage 1, seen in full in the middle plot. The grey region in the stage 1 plot corresponds to the first individual butterfly operation (bottom).

29.45MHz. For the CW305, the target clock rate is set at 10MHz while the sampling rate is 4GHz.

Custom software designed for DIT-FFT analysis was written in C for the Cortex-M4 and in Verilog for the Artix-7, along with custom python scripts for interfacing with ChipWhisperer and implementing Alg.2 [22]. Both sets of designs feature a sequential butterfly operation. IEEE-754 floating point numbers are used in the MCU software. The FPGA design uses Q8.8 fixed point numbers.

B. Leakage Assessment

We show in Fig.3 that from the full power trace of an 8-bit DIT-FFT on the CW312 device, it is possible to see twelve distinct segments in the trace. The first four segments correspond to the first stage. Individual butterfly operations can be isolated from here. Proper alignment of power traces must be carried out for statistical tests to be carried out afterward.

We evaluated the information leakage contained within the power traces for both target boards using a Test Vector Leakage Assessment (TVLA) test, Fig.4. The TVLA test identifies differences between two sets of power traces by computing Welch's t-test on the two sets. One set has a fixed input across all samples, the other is random in each sample. In both cases, the input is an 8-bit key. The number of samples that we use is 100. A t-score, $|t|$, close to 0, at a certain point, implies that the fixed and random sets have an approximate mean value. A large t-score implies that the mean of the fixed and random sets is significantly different, perhaps due to the data dependance that we wish to exploit. Although a high t score does not guarantee that leakage can be easily exploited, an adversary has an advantage with a higher t score in the test. At some arbitrary point, if

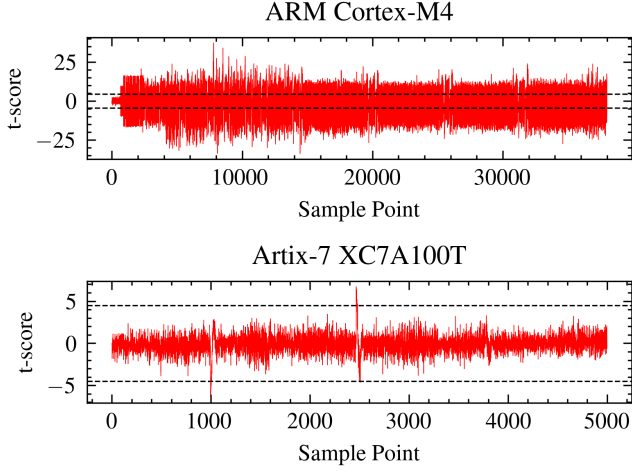


Figure 4: TVLA test for ARM Cortex-M4 (CW312) and Artix-7 XC7A100T (CW305) platforms running the DIT-FFT algorithm. Black dashed lines at $|t| = 4.5$

$|t| < 4.5$, generally there is no significant leakage, while if $|t| \geq 4.5$, this implies significant leakage that can perhaps be exploited.

The T-value boundaries are represented in the dashed black lines in Fig. 4. On the Cortex-M4 board there are many sections of the power trace that will potentially leak the key value on a given butterfly operation, as most sample points past point ~ 1750 have $|T| \geq 4.5$. In contrast, for the Artix-7 board, there are only two small sections of the trace that satisfy the leakage potential criteria, at point ~ 1000 and point ~ 2500 . The immediate take away is that the MCU implementation is much more vulnerable to a power analysis attack than the FPGA, under our current experimental setup, though both implementations do leak information about their input.

C. ARM Cortex-M4 Leakage

A detailed observation of the different power trace templates (an average trace over 50 samples), for each of the four possible butterfly inputs, along a section of interest, can be found in Fig. 5. When comparing each of the templates, it can be seen that each is significantly distinct from all the others. This shows that the leakage implied in Fig. 4 is easily exploitable by matching the victim trace to the template. This is true only if the noise in the trace from the victims device is low. If noise is too high, the trace will appear more random and all scores from Alg. 2, line.10 will all be relatively unpromising. However, in our current experimental setup, the noise is low in the CW312 traces.

D. Artix-7 XC7A100T Leakage

A series of the 75 most distinct points from a sample set of 5000 points between the template traces recorded on the Artix-7 XC7A100T target is shown in Fig. 6. The most significant points of interest occur between sample points 25-35, where the templates are clearly distinct for each of the four inputs. The remaining sample points do not have any significant difference in template values. This shows that the Artix-7 XC7A100T device does leak K_{EC} in the power trace, which is consistent with the TVLA test. However, the distinction between templates is much less pronounced

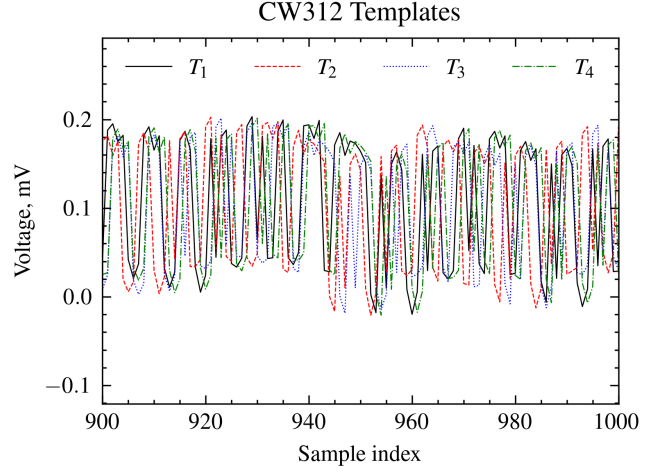


Figure 5: Template power traces across a selected sample range recorded on the CW312 board (Arm Cortex-M4).

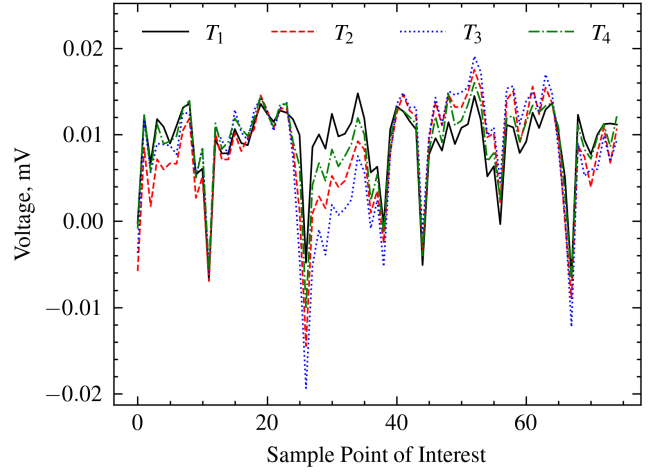


Figure 6: Template power traces across a selected set of points of interest recorded on the CW305 board (Artix-7 XC7A100T).

in the CW305 board compared to the CW312, under our current experimental setup. This makes an attack difficult, as a small amount of noise in the victim device power trace can result in the target trace not matching the correct template any more significantly than the incorrect templates.

E. Attack Performance

The attack, when performed on the Cortex-M4 platform is successful. A successful match of template to target trace for a butterfly operation takes on average 1.042 seconds. This is with limited performance optimisations, therefore further improvement is possible. Since the attack is a sequence of $n/2$ individual butterfly input recoveries, the full attack can be expected to take approximately $n/2$ seconds. This linear time attack is viable for exploiting PA of arbitrary key size.

A section of the output log for the attack can be found in Fig. 7. This snippet is focused on the statistical results from comparing each of the four templates to the target trace on the first butterfly operation (Segment 0). The correct input in this case was $(x_0 = 1, x_4 = 1)$. Guess 0, through Guess 3 are $(x_0 = 0, x_4 = 0)$, $(x_0 = 0, x_4 = 1)$, $(x_0 = 1, x_4 = 0)$ and $(x_0 = 1, x_4 = 1)$ in that order. The Mean Squared Error

```

Segment 0
Guess 0. MSE: 0.026930091417854338,
      PC: 0.4978433287673936
Guess 1. MSE: 0.03299193977045687,
      PC: 0.3856896788187195
Guess 2. MSE: 0.04104327327827073,
      PC: 0.2338513997655182
Guess 3. MSE: 0.00018392346275788736,
      PC: 0.9965736472717602

Segment 1
...

[1 0 1 1 1 1 1 0] : Eve's Guess
[1 0 1 1 1 1 1 0] : Correct Key
Successfull Key Recovery
Time Taken: 4.05157208442688s

```

Figure 7: Snippet from output log from key recovery attack on ARM Cortex-M4 platform. Snippet features statistics calculated between each template and the first target butterfly operation, as well as final result and timing.

(MSE) is significantly the smallest for the correct guess and the Pearson Coefficient (PC) is significantly the largest. Both are valid statistical measures for comparing templates with the target trace. The full key, of 8 bits, was successfully recovered in just over 4 seconds.

Currently, our attack on the Artix-7 platform is not successful, despite the identified leakage observed in Fig.6 and Fig.4. Since the leakage from the Artix-7 device is small, any noise on a single power trace has a significant impact on the ability of the statistical methods we use to determine which template best matches the target trace effectively. As such, successful key recovery is not yet possible. Despite this, we believe that since leakage exists on an average power trace and is seen in the TVLA plot, an attack on the Artix-7 platform may yet prove successful with a refined experimental setup.

V. CONCLUSION & FUTURE WORK

This work highlights sensitive information leakage is present in the power consumption of Toeplitz hashing PA in both MCU and FPGA platform. We show and that a linear time key recovery attack can be carried out against an ARM Cortex-M4. This work should concern those interested in the use of practical QKD, as the assumption that the secure perimeter will always remain secure is not strong. For example, a Hardware Trojan could be constructed to leak information via covert channels. Even simpler, in the context of space based QKD such as the Eagle-1 [23] or SAGA [24] missions, a malicious, trusted human actor in the ground station has potential to interfere with the QKD device with the intention of recording and leaking power consumption of the unprotected QKD device outside the security perimeter.

Research into countermeasures appropriate to ensure that a single power trace attack on any realistic platform is infeasible is now required. This will form future work following on from this investigation. Any countermeasure must have the condition that it protects from single trace attacks as

well as having low performance overhead such that high throughput PA can be maintained. Moreover, this uncovered vulnerability is by no means the only point of weakness in Toeplitz hashing based PA; other vulnerabilities identified in sec.III warrant investigation themselves. On a broader scale, further investigation of QKD classical processing is required to ensure QKD remains secure against classical SCA threats.

ACKNOWLEDGEMENTS

This PhD project receives funding from the Cyber AI Hub Doctoral Training Program at CSIT, Queen's University Belfast, supported by the UK Government as part of the New Deal for Northern Ireland, administered through Innovate UK/UKRI.

REFERENCES

- [1] C. H. Bennett and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing," en, *Theoretical Computer Science*, vol. 560, pp. 7–11, Dec. 2014.
- [2] S. Pirandola, U. L. Andersen, *et al.*, "Advances in quantum cryptography," en, *Advances in Optics and Photonics*, vol. 12, no. 4, p. 1012, Dec. 2020.
- [3] V. Zapatero, Á. Navarrete, *et al.*, "Implementation Security in Quantum Key Distribution," en, *Advanced Quantum Technologies*, Oct. 2023.
- [4] V. Makarov, A. Abrikosov, *et al.*, "Preparing a commercial quantum key distribution system for certification against implementation loopholes," en, *Physical Review Applied*, vol. 22, no. 4, p. 044076, Oct. 2024.
- [5] ETSI, "Quantum Key Distribution (QKD); Part 5: Security specifications of QKD Components," Tech. Rep. ETSI GS QKD 005 V1.1.1, May 2010.
- [6] M. Curty and H.-K. Lo, "Foiling covert channels and malicious classical post-processing units in quantum key distribution," en, *npj Quantum Information*, vol. 5, no. 1, p. 14, Feb. 2019.
- [7] Y. Luo, X. Cheng, *et al.*, "An Overview of Postprocessing in Quantum Key Distribution," en, *Mathematics*, vol. 12, no. 14, p. 2243, Jul. 2024.
- [8] O. Nikiiforov, A. Sauer, *et al.*, "Side-Channel Analysis of Privacy Amplification in Postprocessing Software for a Quantum Key Distribution System," *Tech. Rep., Technical Report TUD-CS-2018-0024*, 2018, Mar. 2021.
- [9] N. Canavan, A. Khalid, *et al.*, "Cascade Error Correction Attack; Exploiting Implicit and Side Channel Information Leakage," Mar. 2025, pp. 502–510.
- [10] J. Martinez-Mateo, C. Pacher, *et al.*, "Demystifying the Information Reconciliation Protocol Cascade," en, *Quantum Information and Computation*, Dec. 2014.
- [11] G. Kim, D. Park, *et al.*, "Side Channel Vulnerability in Parity Computation of Generic Key Reconciliation Process on QKD," en, Jeju Island, Korea, Republic of: 2021 International Conference on Information and Communication Technology Convergence (ICTC), Oct. 2021, pp. 257–261.
- [12] D. Park, G. Kim, *et al.*, "Single trace side-channel attack on key reconciliation in quantum key distribution system and its efficient countermeasures," en, *ICT Express*, vol. 7, no. 1, pp. 36–40, Mar. 2021.
- [13] Z. Qiao, Y. Liu, *et al.*, "When NTT Meets SIS: Efficient Side-channel Attacks on Dilithium and Kyber," en, *Cryptology {ePrint} Archive*, 2023.
- [14] R. Primas, P. Pessl, *et al.*, "Single-Trace Side-Channel Attacks on Masked Lattice-Based Encryption," en, in W. Fischer and N. Homma, Eds., vol. 10529, Cham: Springer International Publishing, 2017, pp. 513–533.
- [15] National Institute of Standards and Technology (US), "Module-lattice-based key-encapsulation mechanism standard," en, National Institute of Standards and Technology (U.S.), Washington, D.C., Tech. Rep. NIST FIPS 203, Aug. 2024, NIST FIPS 203.
- [16] National Institute of Standards and Technology (US), "Module-lattice-based digital signature standard," en, National Institute of Standards and Technology (U.S.), Washington, D.C., Tech. Rep. NIST FIPS 204, Aug. 2024, NIST FIPS 204.
- [17] E. Karabulut and A. Aysu, "FALCON Down: Breaking FALCON Post-Quantum Signature Scheme through Side-Channel Attacks," Dec. 2021, pp. 691–696.

- [18] J. W. Cooley and J. W. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series," en, *Mathematics of Computation*, 1965.
- [19] P. C. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems," N. Koblitz, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, pp. 104–113.
- [20] S. Mangard, E. Oswald, *et al.*, *Power analysis attacks: revealing the secrets of smart cards* (Advances in information security 31), en. New York: Springer, 2007.
- [21] C. O'Flynn and Z. Chen, "ChipWhisperer: An Open-Source Platform for Hardware Embedded Security Research," en, in E. Prouff, Ed., vol. 8622, Cham: Springer International Publishing, 2014, pp. 243–260.
- [22] N. Canavan, "Toeplitz Attack", Oct. 2025. [Online]. Available: https://github.com/ncanavan1/Toeplitz_attack (accessed: accessed: 1. Oct. 2025).
- [23] ESA, "Eagle-1 Mission". [Online]. Available: https://www.esa.int/Applications/Connectivity_and_Secure_Communications/Eagle-1 (accessed: accessed: 1. Oct. 2025).
- [24] ESA, "SAGA Mission". [Online]. Available: <https://connectivity.esa.int/ultrasecure-communications-saga> (accessed: accessed: 1. Oct. 2025).