Grant agreement no: 644729

Secure Architectures of Future Emerging cryptography

State-of-the-Art in Physical Side-Channel Attacks and Resistant Technologies	
Deliverable	D7.1
Author(s)	Philip Hodgers (QUB), Francesco Regazzoni (USI), Richard Gilmore (QUB), Ciara Moore (QUB), Tobias Oder (RUB).
Version	1.0
Status	Approved
Date	11th February 2016
Classification	☒ **White – public** ☐ **Green – restricted to consortium members** ☐ **Yellow – restricted to access list given below** ☐ **Red – Highly sensitive information, access list only**
Access List	

Executive Summary

This report surveys the current state of the art in physical attacks and countermeasures for cryptographic devices and considers this in relation to the new class of lattice-based implementations. Although lattice-based primitives have been investigated for their resilience against quantum attacks, they will nevertheless require implementation in existing CMOS technology, and, as such, will be susceptible to the same physical attacks as those for existing cryptographies, namely that of side-channel attacks.

As new lattice-based designs emerge, and the number of deployments increase, we expect to see further new attacks described in the literature that exploit the particular characteristics of the lattice-based implementation. Countermeasures to address these specific threats, as they emerge, will be an important area of research going forward. However, we can expect that existing countermeasures such as masking, constant time, randomisation and fault detection will all have an important role to play in lattice-based security.

In terms of performance and security, the implementer will need to carefully consider the use of optimisations, since, for example, the use of lookup tables, early exiting of loops and branching based on secret data, can all lead to a non-constant time of operation. The re-use of libraries, which have been developed with security in mind, can help reduce the likelihood of introducing unintended vulnerabilities in the software context.

Of particular interest on microcontrollers is the use of on-board features, such as true random number generators, which can be used to output pseudo-random and uniformly distributed values. Specialist digital signal processing instructions, such as multiply-accumulate, can also be useful to speed up number theoretic transform calculations for more efficient implementations. For applications where the communication cost matters more than the processing time, it is beneficial to apply compression techniques (e.g. Huffman encoding) to signatures. On-board vector processing units such as AVX or Neon can also greatly increase performance; particularly linear algebraic operations, which are usually well suited for parallelisation. However, care should be taken to consider the potential side-channel leakages when employing any such features.

Table of Contents

Table of Figures

Glossary

AES	Advanced Encryption Standard
ASIC	Application Specific Integrated Circuit
AVX	Advanced Vector Extensions
CED	Concurrent Error Detection
CDT	Cumulative Distribution Table
CML	Current Mode Logic
CRT	Chinese Remainder Theorem
CMOS	Complementary Metal-Oxide Semiconductor
DPA	Differential Power Analysis
DEMA	Differential Electro-Magnetic Analysis
DES	Data Encryption Standard
DFT	Discrete Fourier Transform
DPA	Differential Power Analysis
DRP	Dual-Rail Pre-Charge (Logic)
DSP	Digital Signal Processing
EDA	Electronic Design Automation
EM	Electromagnetic
FFT	Fast Fourier Transform
FPGA	Field Programmable Gate Array
HD	Hamming Distance
HPC	Hardware Performance Counter
HW	Hamming Weight
IC	Integrated Circuit
LFSR	Linear Feedback Shift Register
LWE	Learning With Errors
LUT	Look Up Table (FPGA)
MD4	Message Digest Algorithm
MDPL	Masked Dual-Rail Pre-Charge Logic
NEON	Single Instruction, Multiple Data Accelerator for ARM Cortex-A Series
NMOS	N-Type CMOS
NN	Neural Network
NOP	No Operation instruction

NTRU	A Lattice-Based Public Key Cryptosystem
NTT	Number Theoretic Transform
PCA	Principle Component Analysis
PMOS	P-Type CMOS
RAM	Random Access Memory
RC4	Rivest Cipher 4
RF	Random Forest
R-LWE	Ring Learning With Errors
RNG	Random Number Generator
RSA	Rivest-Shamir-Adleman Public Key Cryptosystem
RSL	Random Switching Logic
SABL	Sense Amplifier Based Logic
S-Box	Substitution-Box
SCA	Side-Channel Attack
SOBER	A stream cipher
SNR	Signal-to-Noise Ratio
SPA	Simple Power Analysis
SVM	Support Vector Machine
TSE	Transient-Steady Effect
WDDL	Wave Dynamic Differential Logic
XOR	Exclusive-OR

1 Introduction

1.1 Purpose and Scope

With recent advancements in quantum technology, and the potential that practical quantum computers could become a reality at some future date, there has been renewed interest in the development of cryptographic technologies that are secure from quantum attacks, such as Shor's algorithm for integer factorisation. One promising candidate is cryptography based on the hardness of lattice problems. Although lattice based cryptography offers the prospect of quantum resistance, its deployment will nevertheless require implementation on the same physical platforms as those used for existing non-quantum resistant technologies. For this reason it is very likely that lattice based cryptography will also be vulnerable to the class of attacks known as physical, and in particular, side-channel attacks.

To date there has been little investigation into the resilience of lattice-based implementations from side-channel attacks (in particular advanced techniques) and from physical attacks in general, which an adversary could exploit to reveal information pertaining to the secret key. This document aims to address this issue by surveying the state-of-the-art in physical attacks and countermeasures.

Although most of the existing attacks have been developed for use against non-lattice-based cryptographies, a significant number of these attacks will be directly applicable in the lattice-based context, and it therefore serves as a solid platform to begin our investigations into their security. We will further consider the requirements of a lattice-based implementation and the consequent vulnerabilities that may be exposed because of them.

This document is organised as follows: Section 2 considers the types of attacks and their classifications, summarising the main ways in which physical attacks are carried out, such as power and electro-magnetic (EM) attacks, timing attacks, fault attacks, profiling, and machine learning attacks. Section 3 provides a summary of countermeasure techniques, broadly classified into the areas of masking and hiding. Section 4 then gives an overview of lattice-based implementations, relating this to the vulnerabilities presented in section 2, along with a survey of the current state-of-the-art of attacks and countermeasures for lattice-based cryptography.

2 Introduction to Physical Attacks

In this section, we provide an introduction to physical attacks on cryptographic devices, highlighting the vulnerabilities that make these attacks possible and summarise the most common types.

Cryptography provides confidentiality, integrity, authentication, and non-repudiation of data through the use of cryptographic primitives. A generic encryption algorithm usually takes two inputs, a plaintext and key, and produces a single output of encrypted ciphertext. The opposite transformation of decryption takes as input a key and ciphertext and reconstructs the original plaintext. Following Kerckhoffs' principle, restricting knowledge of the details of an encryption algorithm should not be the means by which security is underpinned, rather the algorithms should be assumed to be known to an adversary. Therefore, security for modern cryptographic schemes is determined through knowledge of the secret encryption key.

Encryption algorithms are carefully designed to be secure from theoretical cryptanalysis, often with mathematical proofs of their hardness against the most powerful of computing adversaries. However, real-world implementations generally require the deployment of these algorithms in some form of low-cost hardware, such as an electronic device that implements the encryption/decryption algorithms and stores, somewhere within itself, the secret key.

For low-cost portable devices, such as smart cards, or mobile computing platforms, such as phones and tablets, an adversary has full access to the device. As such, an adversary is free to not only control what data is passed into the system, perhaps aiding in cryptanalytic attacks, but more importantly an adversary is also able to closely monitor the device, observing its physical properties whilst it performs the cryptographic operations. These physical variables, such as timings required to perform computations, or the instantaneous power consumption during execution of the algorithm, may be acquired using low-cost equipment such as oscilloscopes, making the attacks readily accessible.

The unintended leakage of this side-channel information has a strong dependency on the device being used, the specific operations that are being performed, and on the data being processed. In the case of a cryptographic device, the operations of interest are those that manipulate data that has some relationship, whether direct or indirect, with the secret key. This information can then be compared against a hypothesised model of expected data/key values to determine the value of the secret key and therefore allow an attacker to decipher any past or future communications secured with that key.

2.1 Attack Classifications

Physical attacks can be classified according to various criteria. In this study, we follow the classification as described in the book of Mangard *et al.* [1], which is widely considered as the reference in the field of Power Analysis attacks. Here attacks are grouped according to whether they are performed in an active or passive sense and also in relation to the level of invasiveness required.

Passive Attack: during a passive attack, the adversary does not tamper with the device, and it continues to operate within its normal parameters. An attack is performed by observing and analysing its physical quantities, such as power consumption or execution time. For example, one can observe the differing time taken to compute a square or multiply operation in an exponentiation operation.

Active Attack: to perform these attacks, the adversary has to manipulate the cryptographic device, by modifying its inputs, its environment, or perhaps both together. The goal is to make the device

behave abnormally, and exploit this behaviour whilst performing the attack. For example, by inducing a fault during the processing of the key, it is possible to reveal whether an individual bit of the secret key is a '1' or a '0'.

Attacks can be further classified depending on the level of tampering required to access the device, i.e. non-invasive, semi-invasive, or fully invasive.

Non-invasive Attack: implies that the attacker does not tamper with the device and therefore no trace or evidence of an attack is left behind.

Semi-invasive Attack: may include some activities to gain better access to a device, e.g. open the enclosure of a device to enable closer positioning of an electromagnetic probe, but generally we assume that the device itself is not permanently damaged or modified.

Invasive Attack: implies the greatest level of tampering, for example the de-packaging of a chip, or soldering wires to points on a circuit. This type of attack clearly requires more time and effort and would often leave evidence that an attack has occurred. This kind of attack may also permanently damage the device under test.

There are multiple leakage vectors that can potentially be exploited, such as timing analysis [2], computational fault analysis [3], acoustic analysis [4], and optical analysis [5]. However, one of the major threats comes from the analysis of power consumption measurements, obtained during power analysis [6] and electromagnetic attacks [7][8][9].

2.2 Timing Attacks

Timing attacks exploit the differences in time required by a device to perform specific operations, such as the non-constant time to execute two different instructions e.g. a division or multiplication, the time needed to fetch data due to a cache memory hit or miss, program behaviour due to branching or conditional statements and optimisations which lead to the skipping of unnecessary operations.

In the particular case of a cryptographic device, these performance characteristics will have a relationship with the secret key and/or the input data. Although it may be assumed that timing characteristics leaked in this manner would only reveal limited information from the device, it was shown by Kocher in [2] that the timings for modular multiply operations in exponentiation operations, and modulo reductions of the Chinese Remainder Theorem (CRT) optimisation in RSA, could lead to the discovery of the entire key. Dhem et al. followed this in [40], where a more generalised approach was demonstrated on a smart card implementation. The approach of Dhem *et al.*, which targeted the squaring operation of Montgomery multiplication, did not require the calculation of partial timings for known parts of the key, and therefore required a less-detailed knowledge of the system.

Timing attacks are not only restricted to localised devices. In [41], Bernstein demonstrated a timing attack of OpenSSL AES, on a UNIX x86 server, across a network. The attack consisted of first profiling the server timing responses with a known key to determine what the maximum timing characteristics were for given plaintext values, and then during the attack phase, by sending plaintexts to the server and comparing the timing responses against the profiled references, thus enabling key byte values to be calculated. The underlying reason for the vulnerability was highlighted as the non-constant timing profile of table lookup operations. At the same time, Brumley and Boneh reported remote timing attacks against OpenSSL in [42]. Here, the attack exploited the factorisation of the RSA modulus by progressively improving the guess of the factor q, by iteratively determining its bit values through timing analysis.

The theoretical use of cache hits and misses was proposed by Page in [43], with cache-timing attacks demonstrated in [44] by Tsunoo *et al.*, where it was shown that DES could be broken using 2^{23}

known plaintexts and 2^{24} calculations, with a success rate > 90%. Cache attacks on AES were considered by Bernstein in [41] and more recently in [45] by Tromer *et al.*, where full AES key extraction using DM-CRYPT disk encryption on Linux was accomplished with 800 accesses to an encrypted file, in 65 ms of measurements and 3 seconds of analysis. The OpenSSL library was also attacked in as little as 13 ms, with only 300 encryptions required.

2.3 Power Analysis Attacks

Power analysis attacks exploit the fact that electronic devices consume power during their operation. The logic style used to realise the majority of existing integrated circuits is CMOS, and among the reasons that made CMOS a popular choice are its robustness against errors and its reduced power consumption, which is particularly low in its static, non-switching, state.

Combinatorial logic cells of CMOS are built using PMOS and NMOS transistors, arranged in a complementary structure. PMOS transistors are used to compose the pull-up network, whilst NMOS transistors form the pull-down network. The two networks are designed in such a way that they avoid conducting at the same time. For this reason, the static power dissipation of CMOS logic is minimised and relatively constant during steady-state conditions.

When the output of the cell transitions state, there is a brief short-circuit developed that causes a spike of power consumption. However, the predominant power consumption within a CMOS device is generated when the output of the gates transition from either a ‘0’ to ‘1’ or a ‘1’ to ‘0’ logic state. During a ‘0’ to ‘1’ transition, the consumption is primarily due to the charging of the capacitive loads between the internal and external elements of circuitry. Whilst, during the transition from ‘1’ to ‘0’, the stored capacitances are discharged. It is observed that the current drawn by a CMOS transistor is slightly higher when transitioning from a ‘0’ to ‘1’ value.

It is the aggregation of many logic cells charging and discharging together that combine to give a measurable dynamic power signature that is useful in power analysis attacks. As a consequence, the instantaneous power consumption of a cryptographic device has a strong relationship with the data and computational operations being performed.

Two main classes of power analysis attacks are distinguished; simple power analysis and differential power analysis. In the following sections we introduce both types and discuss their properties.

2.3.1 Simple Power Analysis

In simple power analysis (SPA), an adversary attempts to derive the secret key using a small set of power traces (possibly as few as one), with the relevant information obtained directly from the trace pattern. A possible target for SPA attacks are cryptographic devices in which the execution path depends on the key. For example, in the case of a software-implementation, branching to different instructions may occur when the secret key, or some component of it, has a specific value. Examples of key-dependent branching can be found in cryptographic routines dealing with operations such as key scheduling, permutations, comparisons, multiplications, and exponentiations.

As a pre-requisite, the attacker must first capture the waveform information in a readable form; a task usually performed with an oscilloscope. This requires precise referencing of the signal, i.e. triggering of the source, and some level of pre-processing, usually in the form of filtering, to remove high frequency noise components, or any other artefacts that would have a significant effect on the pattern’s readability. Mounting a successful SPA also requires a detailed knowledge of the instructions or routines being processed on the device, information that is discerned through some prior profiling activity which records, or characterises, the waveform patterns generated by each instruction. Therefore, if a specific instruction (or set of instructions) is executed, dependent on some value of the secret key, it now becomes possible to derive the entire key by simply reviewing the waveform pattern and inferring the key values that are being used.

2.3.2 Differential Power Analysis

Differential power analysis (DPA) is particularly accessible from an attacker's point of view since it can reveal the secret key without requiring any detailed knowledge of the device implementation. Typically, only knowledge of the cryptographic algorithm is sufficient. Due to the averaging methods inherent within DPA, it is often successful even when the collected power traces contain considerable amounts of noise.

DPA attacks are based on a divide and conquer strategy, the general approach being that the attacker selects a small portion of the key, makes a hypothesis of its value, and then compares the hypothesis against the measured power traces. By repeating the process, for all sub-key candidates, each sub-key value can be determined and thus the full key recovered.

Power consumption is typically modelled by estimating the number of '1''s in a register via a Hamming weight or Hamming distance power model. Several differing methods of statistically comparing the modelled versus measured power consumptions are commonly used, such as difference of means, distance of means and Pearson's correlation coefficient [10].

A DPA attack can be summarised in the following five steps:

1. *Choose an intermediate result of the executed algorithm.*
2. *Measure the power consumption.*
3. *Calculate hypothetical intermediate values.*
4. *Map intermediate values to hypothetical power consumption values.*
5. *Statistically compare the hypothetical power values with the measured power traces.*

To improve DPA attacks, the adversary can target several intermediate values and hypothesise all of them. Typically, the number of values used while formulating the hypothesis is used to classify the specific attack. Thus the standard DPA is referred as a first-order attack, whilst attacks using more values are called higher-order attacks. The most common higher-order DPA uses a hypothesis based on two points and is called second-order DPA, as introduced in [11], [12] and [13]. The working principles of higher-order DPA's are the same as those for first-order DPA, with the exception being that a preliminary pre-processing phase is often required. This pre-processing combines, by means of some joint function, a number of points to hypothesise intermediate values, which can then be compared against the power measurements.

One potential drawback of DPA can be the large number of samples needed to mount an effective attack, which can require a significant time to complete. Thus an adversary will usually need to have full, unrestricted, access to the device during this phase of the attack.

2.3.3 Collision Attacks

Collision attacks target the scenario where two encryptions, computed using differing inputs and an unknown key, can generate the same intermediate values. The collision can be exploited if an adversary is able to measure the power consumption of the two encryption operations that create this event. Since it is only possible for two inputs to generate a collision for certain key values, each identified collision enables the attacker to reduce the search space for key recovery.

Schramm *et al.* introduced the attack in the context of power analysis in [14] and [15], where intermediate values of DES and AES were targeted using power analysis to recover the key. Linear and algebraic collision attacks were introduced by Bogdanov in [16] and [18], and by Bogdanov and Pyshkin in [17], with attacks on AES. Their approach further reduced the number of measurements required to detect collisions, and therefore recover the key.

2.3.4 Zero Value Attacks

Zero value attacks exploit the fact that certain cryptographic operations that process zero values can cause the leakage of information that reveals the secret key. In [19] Golić and Tymen showed that a zero value attack could overcome the multiplicative masking countermeasure of an AES block cipher; since multiplying a random mask with a zero value will result in the same, unmasked, zero value. Whilst in [20] Goubin showed that scalar multiplication is insufficient to protect elliptic curves against DPA attacks. Their attack supposes that the curve $E(K)$ contains a special point $P_0 \neq \Theta$, such that one of the (affine or projective) co-ordinates is equal to 0 in K. Where this is the case, key bit guesses are shown to result in noticeable power consumption peaks, enabling the key to be recovered. Akishita and Takagi extended the attack in [21], where it was pointed out that even if a point lacks a zero-value co-ordinate, the auxiliary registers might still have zero-values, thus enabling an attack.

2.3.5 Profiling Attacks

Profiling attacks were first introduced by Fahn in [22] with Inferential Power Analysis (IPA), an attack against DES on a smartcard, and later with the template attack of Chari *et al.* in [23], an attack against an RC4 implementation on a smart card. These approaches require an identical test device for which the attacker knows the key and plaintext sequence. The device under test is supplied successive plaintexts, with the recorded power consumption values used to compile a power profile for the device. This profile can subsequently be compared with the measured power consumptions of an identical device to determine which key values were used in the target device. In [24], Rechberger and Oswald, and subsequently Gierlichs in [25], looked at different ways to find the points of interest in the power trace to further improve the efficiency of the template. The profiling phase of a template attack necessitates an intimate knowledge of the implementation in order to enable the operations to be profiled accurately, which may require a large number of power traces to be gathered. However, if this level of information is available, the template attack can be a very powerful side-channel attack, since it can potentially reveal the key with as little as one power trace, as shown in [23].

Another, more recent, example is the profiling of hardware performance counters (HPCs), investigated by Bhattacharya and Mukhopadhyay in [26], as a means of obtaining side-channel information. The author's targeted 1024-bit RSA on an Intel platform by monitoring the state of the HPCs to determine the number of branch misses on the square any multiply operations of the Montgomery exponentiation algorithm. By comparing the online HPC values against previously profiled HPC simulation values, they were able to successfully determine the RSA key bit values.

2.3.6 Machine Learning Techniques

The application of machine learning as a method to perform SCA is a relativity new idea with only a few papers exploring the concept. To date most of the work has centred on the use of support-vector machines to perform attacks, as well as random forests. Comparison between different machine learning attacks that have currently been used is difficult as they have been applied to different architectures and have each employed differing attack models.

2.3.6.1 Supervised Learning

Supervised learning requires labelling of the data, i.e. knowledge of what the correct class for a particular input should be. The learner is then trained by testing the output of the algorithm against the expected value. A cost is added when the learner incorrectly identifies a sample and this is repeated iteratively until the learner can identify the sample, or until no significant improvement in identification occurs. In most SCA labelling the data with the correct key is fairly trivial and the majority of research into using machine learning for SCA has been through the use of supervised

learning. However, in these cases the features selection stage has either been manual or had limited processing carried out on the data. It is here that unsupervised learning has true potential, to identify suitable features from traces that have countermeasures and where leakage identification is not trivial.

The specific machine learning algorithm used is often less important for a successful attack than the feature selection and data set size. That being said, there are differences in how the algorithms operate; i.e. whether the algorithm can handle non-linear relationships and whether the resulting model is black box or can easily be interpreted.

2.3.6.2 Support Vector Machines (SVM)

SVMs have been the most widely studied algorithm for SCA. Such attacks are described in the works of [27], [28] and [29]. SVMs typically employ one-versus-all classification and use multiple binary classifiers to separate multi-class data. Dividing the problem into multiple binary classification problems is the most common way to approach multi-class problems, for a problem with M classes this leads to M classifiers. This makes SVMs more suited to attacks using a bit or Hamming weight model, since a byte model would require 256 classifiers, as described by Lerman *et al.* in [30].

2.3.6.3 Random Forest (RF)

The authors in [30] also explored the use of RF classifiers to attack DES. RF classifiers use an ensemble of decision trees to classify data. A simple decision tree can lead to overfitting of training data and poor generalisation on test data. To overcome overfitting RFs average over multiple decision trees, each trained on a random subset of the features given.

2.3.6.4 Neural Networks (NN)

NNs map input data to output classes using one or more hidden layers. These hidden layers can learn non-linear relationships between the data, NNs are inherently multi-class. “Shallow” NNs, those with one or two hidden layers, perform similarly to SVMs and are mathematically similar. Deep neural networks, those with a large number of hidden layers, have only recently become viable due to the large computational cost in training such a network. Gilmore *et al.* [31] used a two stage neural network to attack a masked version of AES.

2.3.6.5 Unsupervised Learning

Unsupervised learning does not require any labelling of data, instead clustering algorithms attempt to group data based solely on similarities found within the data. It is often possible to label these clusters afterwards and then use the clustered data to carry out supervised learning. Principle component analysis and independent component analysis are two methods used for unsupervised learning that use the statistical properties of the samples to group data. Chou *et al.* [32] use an unsupervised learning model for SCA; this paper is an initial exploration of the topic, with room for more extensive research in this subject area.

Principle component analysis (PCA) has been used as a pre-processing technique for DPA [33]. PCA has been used in numerous fields extending beyond that of SCA, with the assumption that the high-order eigenvectors contain all the relevant information, with the low order eigenvectors containing noise, enabling the dimensionality to be reduced, thus increasing the SNR of the desired features. This theory however has been challenged in [34] and [35], where it was shown that when information was clustered in a localised small set of points in a large data set, then this information was actually in a particular set of low order eigenvectors. In SCA the points of interest are typically a small localised set amidst a large number of uninteresting points; this was tested and shown to be an effective technique in [36], with the theory further developed in [37]. The application of PCA is still an active area of research and advances in this field may yield further insights into how to

improve its usage with SCA. Moreover, it has only been put to limited use as a feature extraction technique for use with machine learning [29] and for whitening in [31].

2.4 Electromagnetic Attacks

Electromagnetic (EM) attacks were first introduced in the literature in [7], as the differential electromagnetic analysis (DEMA) technique. DEMA is a variant of the DPA attack of Kocher [6], differing in the method of obtaining the power consumption readings. In EM attacks, an electric or magnetic field probe is used to convert the fields generated by current flows from within the device circuitry into voltage signals, for recording with an oscilloscope. EM fields are usually sensed with near field probing methods. However, as Agrawal *et al.* showed in [9], it is also possible to use far field antennas, with AM demodulation techniques, to extract the side-channel information.

The techniques of measuring side-channel EM emissions are not new. Research in the 1960's was undertaken by the US Department of Defence in the TEMPEST programme. This came after the discovery at Bell Labs that their electro-mechanical signalling machines, used for covert communications by the US army and intelligence services, were found to emanate EM pulses that could be used to reveal which characters had been typed at the terminals [38]. Later, other EM side-channel attacks were developed, relating to the reconstruction of computer monitor screens by capturing the vertical and horizontal scan patterns of cathode ray tubes.

An EM acquisition has several attributes that may offer advantages over that of traditional power analysis attacks.

Reduced Invasiveness: In a power analysis attack, the adversary usually attaches a resistive load between the power supply, or ground, of the target device. Measurements are then recorded using an oscilloscope, registering the changes in the voltage drop across the resistor as the current flow drawn by the device fluctuates. The resistor may require soldering in-circuit, leaving permanent evidence of the attack. In contrast, the electric and magnetic fields radiating from a device permit measurements to be taken with a non-contact probe. Skorobogatov showed in [39] that with the use of invasive techniques, such as decapsulation, that it is possible to gain closer access to the IC core. With modern sensitive EM probes, and through use of pre-amplifiers, placing the probe tip on the chip surface will often provide sufficient signal strength to mount a successful attack. The EM probe can therefore potentially offer a fully non-invasive approach to side-channel measurement. A typical EM measurement is shown in figure 2-1.

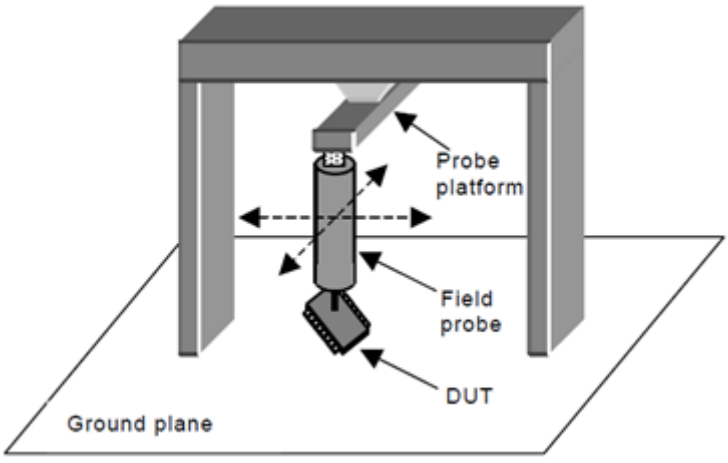

Figure 2-1 EM scanning platform as defined in IEC 61967-3 [146].

Improved Localisation: In a power analysis attack, the position of the load resistor is fixed, with the power consumption of the entire device generally being recorded. This macroscopic power

consumption consists of contributions, not only from the targeted encryption processing core, but also from other regions of the device that may be carrying out other tasks or generating electronic noise from unrelated system switching activities. This statistically unrelated information can lead to anomalies in the side-channel analysis, a feature termed as 'false peaks' by Gandolfi *et al.* in [8]. In contrast, the EM probe may be positioned at any desired location, and with the use of a high resolution, fine-tipped probe, the power consumption contributions from a small localised area of a device may be recorded, thus minimising the power consumption contributions from other unrelated areas of the device.

The quality and strength of the acquired signals, with respect to noise, is an important factor in determining the success of a side-channel attack, although pre-processing techniques such as alignment, averaging and pattern matching can be used to improve the result.

Enhanced Pattern Matching: The localised nature of EM power consumption measurements can also assist with pattern extraction techniques, since the acquired data will more closely resemble the characteristic power consumption patterns that are being profiled. For example, the ten successive rounds of AES will often produce the corresponding pattern of ten processing peaks on the oscilloscope's display. Passing an EM probe across the surface of a device that is processing an encryption algorithm will show a varying pattern at different locations across the IC's surface. The areas which display the clearest pattern of the ten rounds of AES will have the best signal-to-noise characteristics and thus offer the most advantageous position to mount the attack.

2.5 Fault Attacks

In fault attacks, the adversary purposely induces a fault and exploits the erroneous behaviour of the circuit to gain some information about the secret key. These errors are typically transient in nature, meaning that their effects are reversible. As such, once the fault has propagated through the circuit, the device will continue to operate normally. This approach is advantageous since if the device is not permanently damaged, the attacker can continue to perform many repeated experiments, sufficient to generate and observe the desired effects.

Although the chance of faults occurring spontaneously within a device are very small, Anderson and Kuhn showed in [46] and [47] that faults may be intentionally introduced in smart cards by varying the supply voltage, system clock speed or ambient temperatures. A further class of invasive attacks were introduced by Skorobogatov in [5] and [39], with the use of destructive ion beams and semi-permanent optical fault injection techniques; the faults being shown to induce effects such as changing the values of internal registers, incorrect branching of the program or the skipping of program instructions.

Fault attacks presented in the literature target both public and private key algorithms. The attack of Boneh *et al*. [48] targets the Chinese remainder theorem computation of the RSA signature scheme and is particularly efficient since, to be successful, it requires only one correct and one faulty signature. Fault analysis techniques were subsequently applied to attacks on other signatures schemes in [49], [50] and [51].

Among the attacks that target block ciphers, the attacks on AES are of particular interest. One example is the work in [52], which describes a differential fault attack against the substitution permutation network. The work of Kim and Quisquater in [53] targets the key schedule of AES, and to be successful requires only two faults to occur in the 9th round. Concerning active attacks, the most common is the fault injection attack of [54].

Fault attacks relating to stream ciphers were first introduced by Hoch *et al.* in [55], where the linearity of the LFSR component is exploited, with attacks on LILI-128 and SOBER. An attack against RC4 is also demonstrated, with random faults introduced into the S-table and the first output byte of RC4 after initialisation is targeted.

A description of several ways to induce faults in cryptographic devices is reported in [56] by Barenghi *et al.* The authors classified the attacks in two main classes; low cost, and high cost (with the differentiator being a cost of more than 3000 $USD for the attack to be carried out). An additional ranking of the attacks is then provided, based on the skills needed by the adversary, which can be low, moderate, high, or complete. Among the simplest attacks (which are low cost and that can be carried out with limited skills) are underfeeding and heating. Both approaches were shown to be an effective way to successfully mount a fault attacks.

The resistance against a successful fault attack can be significantly affected by the shrinking of the technology and through the use of aggressive power saving techniques. For example, in [57], Barenghi *et al.* describe an in-depth characterisation of a chip implementing AES, manufactured with a 65nm low power library and working in the subthreshold voltage range. The authors show that it is possible to inject faults that would enable an attack to be performed against the cipher. The authors also addressed the problem of predicting such faults at design time, employing standard EDA tools. They concluded that it is possible to predict which lines are more likely to fail, through a static time analysis.

An attack on block ciphers, using the insertion of clock glitches to induce faults, was proposed by Ren *et al.* in [58]. Their transient-steady effect (TSE) attack exploits the fact that a combinatorial circuit will often have a transient state before transitioning to a correct output state. The attack is performed by injecting a clock glitch into the system, with the output state monitored to capture any key related information which is briefly leaked during the transient stage of the combinatorial circuit's operation. The authors state that an advantage of this attack approach, over other fault attacks, is that it does not require a large number of encryptions in order to build the statistical model.

2.6 Photonic Emission Analysis

In photonic emission analysis, such as the work of Schlösser *et al.* in [147], the backside of an IC can be observed and photon emissions recorded with a very high spatial resolution. Although the described attack was made more accessible through thinning of the backside substrate, the authors pointed out that many advanced security IC's, such as those contained within modern smart cards, tend to have thinner substrates. In any case, more advanced InGaAs cameras, which are sensitive above a 1µm wavelength, are able to detect the photonic emissions readily, since, at these frequencies, the silicon becomes transparent with respect to their propagation.

The approach of photonic emission analysis, although invasive and requiring a custom scanning platform to be developed, has the potential to enable the determination of values held within individual transistors on a device. The authors demonstrate that it is possible to read s-box values directly form a composited image of the photon emissions of SRAM memory cells, in a manner equivalent to a simple power analysis.

It is claimed by the authors that the cost of such a system is comparable to that of a mid-range oscilloscope. However, due to the bespoke nature of its construction, and level of invasiveness required (the device needs to be completely removed from the system), it is considered to be a more specialised approach for mounting an attack.

2.7 Concluding Remarks

Physical attacks have been an active research topic for nearly 20 years now, not only because of the high level of threat posed by such attacks, but also because they are constantly evolving. As soon as new countermeasures are published, then new attacks begin to be developed, and whilst there continues to be a strong incentive to break protected systems, this activity will surely continue.

As we have seen, there are a large number of possible attack vectors that the cryptographic engineer needs to be aware of. Over time these attacks have become more accessible to the adversary, with the continued lowering in cost of equipment such as oscilloscopes and probes. The processing of large side-channel data sets may also require significant computing resources. With the advent of cloud services, it now becomes much easier for a scaleable, on-demand, processing resource to be applied, without investment in upfront infrastructure costs.

The power and sophistication of attacks has improved through the advancement of analysis techniques, developing from simple comparators such as the difference of means, through differential analysis using Pearson's correlation, to the more recent mathematical treatments and statistical methods such as information analysis. As improvements in the sophistication of attacks continue, both new and existing countermeasures will require ongoing evaluation, ensuring that they continue to offer the required levels of protection. Indeed, there is now a wide array of countermeasures available for the cryptographic engineer to choose from, many of which can be used in combination, supporting a multi-layered approach to their implementation. In the next section, we will consider the various countermeasures that have been proposed to combat the ongoing threat from physical attacks.

3 Countermeasures

The implementation of cryptographic algorithms onto electronic devices has the unfortunate consequence that it also leads to the unintentional leakage of side-channel information, exposing vectors of attack that can reveal the secret key and thus compromise system security. Countermeasures are the means by which cryptographic devices are protected in order to minimise leakage and thwart attacks. In this section we present the general ideas behind the various countermeasures and their implementation.

Power analysis is generally considered the most powerful physical attack presented so far. It requires little information about the implementation, with knowledge of the encryption algorithm usually sufficient to mount a successful attack. As such, this research topic has attracted significant interest and we therefore start with a survey of power analysis countermeasures and dedicate a large portion of this chapter to this topic.

3.1 Countermeasures Against Timing Attacks

As discussed in section 0, timing attacks exploit the differences in the time taken to process information that has some relationship to the secret data. Vulnerabilities have been described in the literature for both implementations in the hardware and software contexts and for the various standard public key and symmetric encryption algorithms. For the designer, one of the main pitfalls to be weary of is that of optimisation. Well-intentioned efforts at improving performance through the use of pre-computed lookup tables, or early exits from loops, for example, whilst reducing execution time, will often lead to the leakage of timing information. Care should also be taken when considering the implementation of a given design across differing platforms, since leakages are commonly device specific and closely related to the physical characteristics of the device.

There have been various countermeasures proposed to thwart timing attacks. As already discussed in section 3.2.2, masking countermeasures will change the intermediate values, so that even if their values are leaked, they will not directly reveal the key data. However, their implementation cost may be high and therefore impractical on the constrained device with limited resources.

In Kocher's seminal paper on timing attacks [2], it was discussed that one option is to try and make all operations execute in a constant time. Although conceptually straight-forward, in practise this may not be so easy to accomplish. As Kocher noted, this was a difficult task because of issues such as compiler optimisations, RAM cache hits, and variances in instruction timings; since these aspects are generally outside the control of the designer, particularly in the context of a software implementation. Kocher further suggested the possibility of inserting random delays (with a clock-skipping countermeasure later patented [83]). However, it was noted that this approach had the effect of adding noise, which could be overcome by gathering more traces to average out its effect; with the number of samples required increasing approximately as the square of the timing noise. Kocher recommended the use of blinding to protect RSA, a concept originally proposed by Chaum in [84], coupled with the additional masking of the exponent with a random value before each modular exponentiation.

In [40], where Dhem *et al.* targeted the squaring operation of the Montgomery multiplier of RSA, it was suggested that a simple fix was to make the multiplier operation constant time by adding an additional subtraction operation, even if the result was to be discarded. It was pointed out, however, that the smart card implementation, broken in their demonstrated attack, already included this countermeasure, but that the implementation had a flaw, such that there was still a small time variation between the timings of whether the result was discarded or copied at the end of the

operation. This demonstrates clearly that careful consideration must also be given to the implementation of countermeasures. At the same time, in [85], Schindler proposed the use of dummy operations within the Montgomery multiplier in order to hide the leakage of timing information.

In the context of smart card implementations, the option to disable cache flushing, was proposed by Page in [86].

The work of Tromer *et al.* in [45] considers various countermeasures against cache attacks. Some possible options are:

1. *Avoid the use of memory accesses by replacing lookups with equivalent logical operations. This is a possibility for algorithms such as AES. However, there will be a performance trade-off.*
2. *Use of bit-slicing approach, such as described in [87].*
3. *Use of a cache no-fill mode, where memory is accessed from the cache during a hit and serviced from memory when there is a cache miss.*
4. *Dynamic table storage, where the contents of the table lookup is cycled around in memory during encryption operations in order to de-correlate it.*

For some guidance on generic coding standards, in relation to cryptographic implementations in software, the reader is referred to the resource of [88]. For example, in the context of timing attacks, it is recommended:

1. *Do not compare secret values on a byte-by-byte basis*
2. *Avoid branching predicated on secret data*
3. *Avoid the use of lookup tables indexed by secret data*
4. *Avoid loops that are bounded by a secret value.*

The software implementer can leverage existing libraries, such as NaCl [89], written specifically with security in mind.

Some processors also include custom instruction sets dedicated to cryptography, such as the AES-NI instructions [90], included in the Intel processors since the Westmere architecture. The AES-NI instruction set consists of seven instructions that perform several portions of the AES algorithm. AESENC performs ShiftRows, SubBytes, MixColumns and AddRoundKey, whilst AESENCLAST processes the last round of the algorithm, skipping MixColumn. Similarly, the decryption is accelerated by AESDEC and AESDECLAST. Two instructions are also used for fast key generation, namely AESKEYGENASSIST and AESIMC. A seventh instruction, PCLMULQDQ, aids in carry-less multiplication. According to the manufacturer [90], use of these processor instructions can boost performance, ranging from a factor of x2 to x10, over that of purely software-based implementations. Since these instructions enable an implementation avoiding table lookups, it is also claimed in [91] that security against cache timing attacks is improved.

3.2 Countermeasures Against Power Analysis

The main goal of a countermeasure against power analysis attacks is to make the power consumptions independent from the processed secret data. It is important to note that it is not necessary to reach independence from all processed data in the device, but rather specifically from data that would allow the attacker to verify the intermediate secret values, for example, from the inputs or outputs of the s-boxes in AES, or the values of the exponentiation of RSA.

Counteracting SPA is a more straight-forward prospect; since the attacker has to visually explore the traces, it is sufficient to protect the values directly related to the secret key that affect the program

execution or its behaviour. For example, concerning conditional branches, if the programmer is able to ensure the absence of conditional branching that depends on the secret data, the adversary has limited chances to gain useful information from the inspection of the power traces. Another approach is to increase noise levels to try and hide the signals during the data dependent processing.

Protecting a device from DPA, by contrast, is a much more difficult task, since this attack uses advanced statistical techniques to extract information from a large number of traces.

Countermeasures can be classified into two broad groupings: those that aim to hide the data and those that are designed to mask the data [1]. These concepts are generally valid in both the hardware and software contexts and depend upon the particular methodology adopted to achieve protection. Furthermore, although the two approaches are independent from one another, they are complimentary and thus may be combined, providing a multi-layering of countermeasure implementations.

3.2.1 Hiding

Fundamentally, a hiding countermeasure does not change the intermediate data values that are processed in an encryption algorithm, rather it attempts to hide those values in amongst the other processing activities. Typically, hiding is achieved in one of two ways: by randomising the power consumption or by making the power consumption constant during all processing operations.

A random power consumption can be achieved by changing, in each iteration of the algorithm, the time at which a given cryptographic operation is executed. If the targeted intermediate value is processed in a differing instant in time, then the attack will become more difficult since a more complicated trace re-alignment step is required. Two possible ways for randomising the power consumption are the random insertion of dummy instructions and the shuffling of the operations. Both methods may also be combined. This temporal misalignment between successive side-channel acquisitions reduces the effectiveness of an adversary's statistical analysis.

Pre-processing techniques such as integration [59] can assist in reducing this effect, but are only effective to a certain extent, limited by $\sqrt{l}$, where l is the length of the integrating window [1]. In [60] and [61] Homma *et al.* proposed a technique called phase-only correlation to improve the re-alignment of traces. An alternative approach, based on frequency analysis, was published by Gebotys *et al.* in [62], where the captured data was first transformed into the frequency domain using a discrete Fourier transform (DFT), creating a series of spectrograms for the analysis. Although this approach overcomes misalignments, since it is time-shift invariant, the size of the spectral windowing has an effect on the noise, and a balance therefore needs to be made between a large enough window to capture the common processing operations and the desire to minimise the window width, therefore avoiding the inclusion of unnecessary noise, as considered by Hodgers *et al.* in [63]. Pattern matching techniques that can identify the processing region of interest, and realign the traces based on those characteristic features, is therefore another possible method open to an attacker, as described in [64].

The use of random insertions can have a significant impact on the performance of the system, since dummy operations are purely redundant processing activities. It also offers only limited protection, since the attacker can potentially take many measurements to average out the effects of the randomisation.

The second approach for counteracting power analysis attacks aims at equalising the power consumed by each instruction of the device. As Agrawal *et al.* observed in [9], the instructions of a processing device, such as a micro-controller, can use differing amounts of power with some 'bad instructions' consuming significantly more power than others. Approaches in software to minimise the variance of the power consumption include techniques such as choosing processor instructions with similar power consumption profiles. Hardware approaches have focussed on features such as

incorporating on-board power filters, to reduce exploitable power leakage, through to adding noise generating engines on the device to increase the noise floor. An alternative approach has focussed on implementations at the level of the logic cells, namely with the use of dual-rail pre-charge (DRP) logic styles. DRP uses two wires that are complementary for each signal, as shown in figure 3-1. Care is taken on the interconnection between DRP cells to ensure that the capacitive loads are balanced, as this aspect tends to determine the overall effectiveness of the system.

In [65] Tiri *et al.* proposed the use of Sense Amplifier Balanced Logic (SABL) to provide resistance against DPA. SABL has the property that it charges and discharges the same capacitance (internal and external) every clock, regardless of its state. This was followed with proposals for Current Mode Logic (CML) in [66], [67] and [68], and Wave Dynamic Differential Logic (WDDL) in [69]. WDDL uses standard cells, avoiding the time and cost of implementing a full custom design, though its capacitive balancing, and therefore DPA resistance, is not perceived to be as good as SABL.

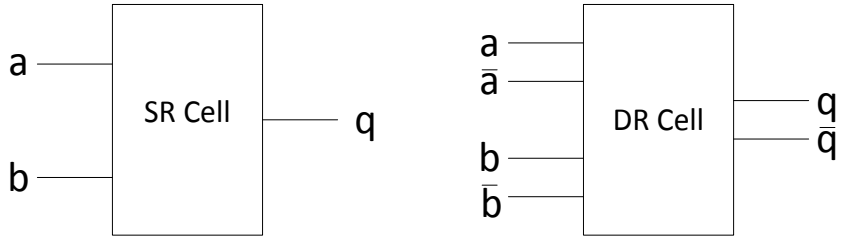

Figure 3-1 A single and dual rail cell. The dual rail cell has complementary input and output lines to balance the power consumption.

3.2.2 Masking

Masking is a technique based on secret-sharing, where the original message is divided into parts called shares and the message can only be reconstructed if a sufficient number of shares is known. The idea of applying secret-sharing to counteract power analysis was first proposed by Chari *et al.* [70] and further improved by Messerges in [11]. With masking, the intermediate values that are processed generate a power consumption that is uncorrelated with the secret key, even though the power consumption characteristics of the device remain unchanged.

When considering an algorithm for masking, such as AES, the linear operations of ShiftRows, MixColumns and AddRoundKey are all easily dealt with since a liner operation has the property of $f(x \oplus m) = f(x) \oplus f(m)$. However, for the non-linear SubBytes operation, $S(x \oplus m) \neq S(x) \oplus S(m)$. For this reason, masking schemes can be complex and require careful consideration during their design. It is possible to divide the masking schema in two groups, based on the operations used to calculate the masked value, namely Boolean and arithmetic masking. Boolean masking uses a bit-wise exclusive-or operation to mask the intermediate value and is particularly suitable for masking linear functions of cryptographic algorithms. Arithmetic masking uses operations such as modular addition or modular multiplication to mask the intermediate values and is particularly suitable when used to mask the non-linear functions, but for other operations it has one major problem; not all the intermediate values can be masked.

An aspect that can cause masking schemes to become complicated is the potential requirement to switch between Boolean and multiplicative masking. Akkar and Giraud proposed an efficient method to facilitate this process in [71]. Multiplicative masks do have a major disadvantage in that they cannot conceal the intermediate value if it is zero. Oswald *et al.* proposed a scheme that took this weakness into account using a combination of additive and multiplicative masks in [72].

There have also been developments in the hardware implementation of masking, with the introduction of masked logic styles by Trichina [73]. This approach employs the principle that, since a

Boolean mask is a linear operation, the values can be processed in a masked cell without requiring the unmasking of the data. The encryption algorithm can therefore be implemented in an un-masked manner, with the masking taken care of automatically at the logic level. The implementation requires an additional line to carry the mask bits associated with each input and output from the cell, as shown in figure 3-2.

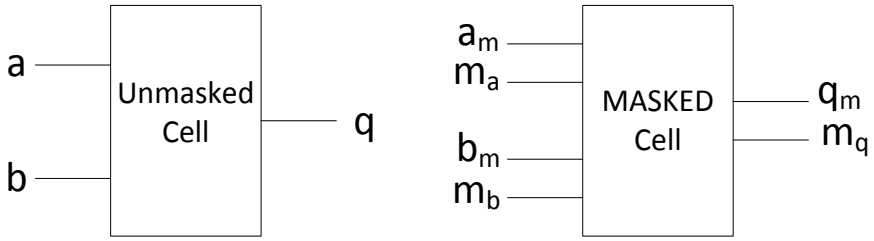

Figure 3-2 An unmasked and masked cell. The masked cell has additional mask lines for the inputs and outputs and can process data in a masked state.

An important consideration for masked cells is the avoidance of glitches, as highlighted in [74], where it was shown that glitches can lead to strong data dependency between the un-masked data and the power consumption. They further proposed masked dual-rail pre-charged logic (MDPL) to overcome issues with glitches. An alternative masked logic cell style, termed random switching logic (RSL), was proposed by Suzuki *et al.* in [75] and [76], where a leakage model was proposed based on transition or switching probabilities and also a scheme that equalised the probabilities for each of the transitions. However, it was shown by Shaumont and Tiri in [77] that both RSL and MDPL could be broken by considering the average power consumption and whether an individual measurement was above or below this threshold, indicating the state of the mask bit and thus determining the value of the underlying key.

In [78] Moradi *et al.* also showed that collision attacks could be used to break some implementations of masking. Whilst in [79] and [80], Oswald *et al.* proposed the combination of additive and multiplicative masking to defend against zero-value attacks.

3.3 Physical Countermeasures for EM Probing and Photonic Emissions

The countermeasures previously discussed provide general protection against both power and EM analysis. However, for non-invasive attacks with an EM probe, or more invasive attacks with photonic emission analysis, physical shielding countermeasures can also offer some further resistance. When the first attacks were reported in the middle-to-late 1990's, chip manufacturers introduced various physical countermeasures to improve the tamper resistance of their devices, with features such as random noise generators, power filters, active grids and metallisation layers [81].

The suppression of EM waves for near field probing is a more difficult task, since the generation of electric and magnetic fields are a natural consequence of the current flows within a device. Electric fields can be mitigated to some extent with the use of metallisation layers on the device core, or through encapsulation of the device; however, a surface cap can be easily removed with depackaging techniques [39]. Magnetic shielding was investigated for its application to resisting EM attacks by Yamaguchi *et al.* in [82]. The authors applied a thin magnetic film shield over the core of the device and reported a 6dB reduction in detected EM signals with a sensor probe.

In the case of photonic emissions, the authors of [147] proposed the use of an active shield or mesh on the backside of the device.

A cryptographic module may also include active anti-tampering countermeasures to monitor important system parameters such as supply voltage, operating temperature and clocking frequencies and suspend module operation if it detects such anomalies.

3.4 Countermeasures Against Fault Attacks

Countermeasures against fault injection attacks have also been proposed. One approach is to use error detection codes, which have been traditionally used in the area of data transmission when dealing with noisy channels. Several classical codes have been adapted to the needs of cryptographic applications, for example the use of parity checking. Additionally, some new solutions based on concurrent error detection (CED) techniques have been proposed. CED works to suppress the normal execution of the algorithm whenever an error is detected, thus preventing an attacker form viewing and analysing the faulty output. One possible means of checking the validity of the output is through the duplication of hardware. The results produced by two identical circuits are compared, with no output produced if they are not equal. This duplication roughly doubles the area needed by the circuit, and therefore is a rather expensive approach. An alternative method is to re-use the same circuit and re-compute the result a second time before comparing. In this case, the area requirements are kept low, but the execution time is doubled.

In addition to these general approaches, some works focus on a particular cryptographic algorithm or class of algorithms. In [92], Wolter *et al.* presented an implementation of the IDEA algorithm in which the data is first encrypted and then, as a check, decrypted with the result compared to the original plaintext. Gaubatz and Sunar analysed public key algorithms in [93], where the authors suggested the provision of error detection and correction by means of redundant arithmetic based on finite rings. Although comprehensive, the proposed implementation is complex and results in a higher area overhead compared to other approaches. In [94] Karri *et al.* proposed a CED that is tailored to substitution-permutation network ciphers, comparing the modified parity of the input with the parity of the output. The CED scheme proposed for AES by Bertoni in [95] uses one parity bit for every internal state byte of AES. This scheme, which requires a limited amount of area to be implemented, detects all odd errors, and in many cases, even errors as well. Due to its simplicity and low overhead, this approach offers an attractive solution.

There have been several proposals to protect RSA signature computations against CRT targeted attacks. In [96] Shamir computed the arguments of the CRT using efficient redundancy, which enabled verification of the values before RSA combination. This approach added minimal timing overhead, compared to the prior approaches that would require full redundancy and a doubling of timing overhead. Kim and Quisquater introduced higher order fault attacks in [97], demonstrating the breaking of first order countermeasures for RSA. Their approach consisted of inducing a first fault during one of the exponentiations and then a second fault to cause the skipping of the CRT error checking routine. In [98], Yen *et al.* showed that inducing a fault into a status register flag could bypass the conditional checking of countermeasures, thus introducing the concept of infective computation.

3.5 Robustness Metrics

The problem of measuring resistance against specific attacks is still an open problem. Among the several metrics presented in the literature, one of the most objective is the one proposed in [99] by Standaert *et al.* which evaluates the resistance of a cryptographic implementation against the strongest possible power analysis attack. The metric establishes a relationship, i.e. mutual information, between the secret key that is used for encryption and the power traces.

The T-test and difference-of-means between two sets of power traces can also be used to detect information leakage, as proposed by Cryptographic Research in [100]. While performing a T-test, the evaluator has to partition the traces based on the value of a selected bit of an intermediate state of

the target algorithm. To assess the specific bit for leakage, an evaluator must compute the intermediate values for the chosen state, using a set of chosen messages. Having recorded the encryption or decryption of the chosen messages, the resulting traces can be partitioned into two sets, depending on the value of the bit of the intermediate state. The results of the statistical test are then used to determine whether a difference between the means exists. The t-test by design can only detect differences contained within the mean of the leakage samples, and assumes that the populations being compared are normally distributed.

A detailed analysis of leakage assessment techniques was proposed by Schneider and Moradi in [101]. The paper provides a study on how the T-test can be applied in a higher-order setting. Additionally, techniques are described on how to carry out the test efficiently and on how to optimise the measurement setup.

3.6 Automatic Application of Countermeasures

Recently, the community has begun to develop, and extend, dedicated toolsets that support the automated application of physical attack countermeasures. Previous works were devoted primarily to the automated application of countermeasures against timing or power attacks.

Most of the automation introduced for side-channel analysis and protection focus on hardware countermeasures, rather than software countermeasures. For example, in [102], Tiwari *et al.* introduced a method for gate-level information-flow tracking, by composing complex logical structures which propagates the trustworthiness of each bit along with its value. Others, such as Tiri *et al.* in [103], Guilley *et al.* in [104], and Regazzoni *et al.* in [105], proposed methodologies to automate the application or analysis of some hardware countermeasures.

A handful of projects have looked at power analysis attacks from perspectives other than hardware design, including the Computer Aided Cryptography Engineering (CACE) project of [106]. In [107] Barbosa *et al.* analysed the effects of a compiler on elliptic curve cryptography, whilst in [108] Bayrak *et al.* proposed a framework to automate the application of power analysis countermeasures. In [109] Cleemput *et al.* proposed compiler techniques to defend against timing attacks on x86 processors.

3.7 Effects of Countermeasures on Other Attacks

The implementation of countermeasures designed to thwart one type of attack may in-themselves have the unfortunate consequence of generating other leakages that can be exploited by an attacker. In the works of [110], [111] and [112] Regazzoni *et al.* show the effect that an error detection circuit may have on the resistance to a power analysis attack, of hardware implementations of cryptographic s-boxes. The authors' show that the presence of error detection/correction circuitry increases the total amount of information available to an attacker, which may then be exploited depending on the particular attack hypothesis used. As a result, when incorporating fault detection or correction circuitry into implementations of cryptographic algorithms, it is important to be aware of the possible side-effects that this added circuitry may have on robustness against power analysis attacks. This may lead to the requirement to add additional protections for the additional circuitry e.g. additional protection for error-check bits.

3.8 Concluding Remarks

Since the first attacks of Kocher [2], which revealed the side-channel vulnerabilities of cryptographic devices, there has been a strong commercial need for the development of new countermeasures to protect against the constantly evolving threat of attacks. The use of countermeasures, as a security mechanism contained within cryptographic devices, has systemic importance in areas such as secure information technology, commerce and the myriad of electronic transactions that underpin our financial system.

From the point-of-view of the cryptographic engineer, there is a large number of potential threats to consider but also a wide variety of countermeasures to choose from. Unfortunately, however, there is no silver bullet to provide a countermeasure that is both low cost, for deployment at the consumer scale, and that can also be guaranteed to provide 100% protection, not only for the current known attacks, but also for the as yet unknown attacks of the future; since some devices may need to be deployed in active service for many years before they are withdrawn or replaced.

For technical and commercial reasons, the extent to which a countermeasure can be implemented may also not be ideal. For example, a countermeasure may take up excessive area on silicon, consume too much power, degrade system throughput, or require the development of expensive new fabrication technologies. There are also considerations dependent on the platform that the countermeasure is targeted at, for example a design for ASIC will have the most scope for unrestricted implementation (with a large up-front financial cost), whereas designs targeted for FPGAs may have constraints based on the available space and technology for specific physical countermeasures, whilst the software based countermeasure will have most limitations placed upon what is possible to achieve and will primarily rely on algorithmic-level approaches. These considerations generally lead to a trade-off in security level versus cost/performance for the designer, with the aim to make an attack infeasible, or at least impractical, in terms of its financial cost and time required for an attack.

With the possible arrival of quantum computing, there is a now a drive to develop new cryptographic technologies that will be secure against quantum algorithms. With these new designs, come new challenges in terms of meeting implementation requirements but also the possibility of introducing new or unforeseen vulnerabilities to be exploited by an attacker. New countermeasures will inevitably have to be developed.

The attacks and countermeasures considered in the previous two sections relate to the differing types of cryptographic technology that currently exist. In the next section we consider the implementation of lattice-based cryptography, the implications in terms of side-channel vulnerabilities and survey the current state-of-the-art in lattice-based attacks and countermeasures proposed thus far.

4 Lattice-Based Crypto Implementations and Side Channel Vulnerabilities

In the previous sections we considered the many physical attacks and vulnerabilities that exist against cryptographic devices, and the corresponding countermeasures that have been developed in response to those threats. Although lattice-based cryptography is deemed to be secure against existing quantum attacks, such as use of Shor's algorithm, their use nevertheless requires implementation on existing CMOS technology, and as such, are considered vulnerable to existing side-channel attacks.

The body of literature in side-channels is largely related to attacks on the currently used crop of block ciphers and asymmetric algorithms. Very little work has been done in the field of lattice-based cryptography, primarily because it has not been widely deployed. We envisage that this situation will change rapidly as new lattice-based implementations become commonplace. As a new type of cryptographic implementation, attackers will inevitably start targeting these systems to probe for weaknesses and to exploit any vulnerabilities that may be inherent within their design.

Lattice based cryptography can typically be reduced down to the implementation of the following two requirements: the calculation of a series of linear algebraic operations such as addition/subtraction, division/multiplication, and the sampling of values from a discrete Gaussian-distributed random source. We now consider these aspects of implementation and survey the current state-of-the art in lattice-based attacks and countermeasures.

4.1 Linear Algebraic Operations

Learning with errors (LWE) usually requires matrix-vector multiplication, which can be carried out using traditional multiplication methods. The majority of research into lattice-based cryptosystems has focused on ring learning with errors (R-LWE), whereby the use of ideal lattices requires polynomial multiplication. Usually a number theoretic transform (NTT) multiplier is employed for this purpose. The NTT is in essence a discrete Fourier transform over finite fields. In some schemes, one of the operands is a polynomial consisting of only a few non-zero coefficients. In those cases, an optimised schoolbook multiplication is usually more efficient [113]. Efficient implementations and several optimised designs for polynomial multiplication have been proposed in [114], [115], [116] and [117].

When applying the NTT, there are several design choices to be made, for example the NTT can be implemented using various butterfly constructions. The most common, when dealing with lattice-based cryptography, are the Cooley-Tukey and the Gentleman-Sande butterfly. Both butterflies can be implemented to take naturally ordered input and produce bit-reversed output or vice-versa. One way to efficiently choose from the differing butterfly constructs was shown in [118]. Depending on the underlying ring, the NTT has to compute the positive or negative wrapped convolution to avoid a costly subsequent reduction of the resulting polynomial.

There are different techniques available to implement modular reduction in finite fields. Especially on architectures with no hardware divider, it is advisable not to rely on compiler-generated modular reduction routines. One straight-forward approach to realise modular reduction in finite fields is the subtract-and-shift algorithm. The idea behind this algorithm is subtracting multiples of the modulus until the result is smaller than the modulus. Another well-known algorithm is called Barrett reduction [119]. While the subtraction of multiples of the modulus from the input is performed systematically in the subtract-and-shift algorithm, the Barrett reduction includes an estimation on how often the modulus has to be subtracted to get a result not larger than the modulus. In case the architecture does not support division, but floating point multiplication, a very efficient approach is

to precompute the division of 1 by the modulus and store the result. For the actual reduction, one now has to multiply the input by the inverse of the modulus and round off the result to find out how many times the modulus has to be subtracted from the input.

4.2 Discrete Gaussian Sampling

Most lattice-based cryptosystems require sampling from a discrete Gaussian distribution. In addition to the traditional method of rejection sampling, several sampling techniques have been proposed or adapted for use in lattice-based cryptosystems: Bernoulli [120], Cumulative Distribution Table (CDT) sampling (also known as inversion sampling) [121], Knuth-Yao sampling [122], rejection sampling [120] and discrete Ziggurat sampling [123]. Over the last few years, there has been an increase in research activity into the development of optimal discrete Gaussian samplers, since sampling is a core component in both lattice-based encryption and signature schemes. We therefore first present the common issues associated with all discrete Gaussian samplers, followed by the most common sampling techniques. For further details, and a more mathematical description on discrete Gaussian sampling within lattice-based cryptography, the reader is referred to the review paper by Dwarakanath and Galbraith [122].

Since no finite computation can produce a perfect discrete normal distribution [122], distributions are used, such that they are indistinguishable from normal to a certain degree. To verify this, the statistical distance between two distributions can be used, where the difference should be negligible. This distance is typically measured in bits (i.e. $2^{-\lambda}$ for target security of λ bits) and is known as the precision parameter. In recent work by Saarinen [124], where the precision of Gaussian samplers was considered, it was argued that such a level of precision is excessive and that in most cases, half of the precision of this security parameter is almost always sufficient. This leads to faster and more compact implementations; potentially halving the size in both hardware and software implementations.

Another important measure is the Rényi divergence. Recent works of [124] and [130] favour the Rényi divergence since it allows for more efficient parameter choices, e.g. a smaller standard deviation. Indeed, *parameter selection* is vital for all sampling methods; parameters required for each sampler include the tail-cut, precision and standard deviation. Another consideration to be noted is that parameter selection is dependent on the target application, i.e. encryption or signature generation.

An algorithmic optimisation, proposed by Peikert in [121], uses the *convolution property* of distributions; given two independent random variables, the sum of these variables is equal to the convolution of their respective distributions. Thus, smaller standard deviations can be used to generate two or more samples, which have the required larger standard deviation when combined using basic additions and scalar multiplications. This has been used, for example, in [125] to improve performance of samplers.

There are several approaches available when considering the implementation of Gaussian samplers. We now discuss the common methods and comment on their efficiency and potential vulnerabilities.

Rejection sampling involves using a random sample from some simple distribution and using probabilities to decide whether to reject or accept each given sample. One issue with rejection sampling is that many samples may be required before one is accepted. Several of the following techniques are optimisations of basic rejection sampling, with the aim of reducing the likelihood of rejection. Rejection sampling has a non-constant execution time, but still the timing leakage from rejection sampling is not considered to be exploitable by an attacker since the output value is independent from the number of previous rejections.

CDT or inversion sampling uses stored, precomputed tables which are then searched to generate suitable discrete Gaussian samples. A hardware design of CDT, targeting the Xilinx Spartan-6 FPGA platform, was proposed by Pöppelmann *et al.* in [125] and compared with the Bernoulli approach. Both are incorporated into hardware designs of the BLISS signature scheme in [120]. Several optimisations are proposed, such as an optimised binary search using shortcut tables, reduction of table sizes by using the Kullback-Leibler divergence and, as previously mentioned, the use of the convolution property to reduce the precomputed table sizes. Results show that CDT outperforms Bernoulli in terms of both operations per second and in area usage. To implement the CDT approach in constant time, the input value has to be compared with all table entries. This is quite expensive, since a result is usually found after searching only a fraction of the table.

Bernoulli sampling is an optimised method of rejection sampling that incorporates the use of the Bernoulli distribution, which is a discrete distribution with only two outcomes, 0 and 1. The advantage of the Bernoulli method is that the rejection probability can be reduced in comparison to traditional rejection sampling. In [129], Oder *et al.* proposed an implementation of three Gaussian sampling methods (Bernoulli, Knuth-Yao, and Ziggurat) on a Cortex-M4F Microcontroller. This research indicates that Bernoulli performs better than Knuth-Yao and Ziggurat, in terms of both memory requirements and speed. As with other methods, naïve implementations of the Bernoulli sampler will leak timing information.

Knuth-Yao sampling uses a Knuth-Yao random walk or a binary tree to generate samples. One advantage of this method is that the Knuth-Yao algorithm has the aim of requiring, as close as possible to, the minimum number of inputs from a uniform distribution [122]. Roy *et al.* proposed a hardware design of Knuth-Yao sampling in [126] for use in LWE encryption schemes. The proposed optimisations include implementation of a discrete distribution generating tree, optimised storage of a probability matrix and also column-wise compression of zeros present in the probability matrix.

The Knuth-Yao random walk does not operate in constant time and therefore is vulnerable to timing attacks. The only known constant time Gaussian sampler proposed is by Roy *et al.* [127]; the authors perform an SPA attack on their initial design, with random shuffling of samples used to counter timing attacks.

Discrete Ziggurat sampling is adapted by Buchmann *et al.* [123] from an original method of continuous Ziggurat sampling, as proposed by Marsaglia and Tsang in [128]. It is again an optimised form of rejection sampling, which divides the area under the curve into several rectangles of equal area, where the left hand side of each rectangle is aligned with the y-axis and the right hand side of each rectangle aligns with the curve of the Gaussian distribution. This structure can then be used to optimise rejection sampling of random points from a uniform distribution. The increase in the number of rectangles then decreases the probability of rejection. Buchmann *et al.* [123] have proposed a C++ implementation with several optimisations, and compared Ziggurat with alternative sampling methods to show that Ziggurat is suitable for use when large standard deviations are required. Like rejection sampling, discrete Ziggurat sampling has a non-constant running time. Due to the more complex structure of the algorithm, it is harder to say whether an attacker can exploit this timing leakage or not. Further research in this field is necessary.

In practise, most discrete Gaussian samplers use a form of *rejection sampling* and thus, such rejections can affect performance and also cause discrete Gaussian samplers to perform in non-constant time, which could ultimately be exploited through timing attacks.

A brief mention of an area related to sampling is that of random number generation (RNG), required as the source of entropy for Gaussian samplers; a topic that is often ignored or assumed when Gaussian samplers are discussed in the literature. Although software based systems will most likely already have RNG resources available, for the constrained device, or custom implementation, a secure RNG source is also an important consideration.

4.3 Device Constraints and Optimisation Techniques

As stated in section 4.1, the polynomial multiplication in ring-based schemes is usually performed by transferring the operands into the NTT domain and computing a point-wise multiplication of the transformed polynomials. Since in many cases one operand is a key (either public or secret), it makes sense to already store the keys in their NTT representation. Similarly, a careful selection of when to apply NTT transformations might yield some performance gain. There are schemes in which the result of a multiplication during one operation is used as an input for a multiplication during the next operation. For instance, in RLWEEncrypt the ciphertext is the result of a multiplication and an addition. To decrypt, the ciphertext gets multiplied by the secret key. Since it is possible to perform the addition in the NTT domain, rather than computing a backwards transform on the result of the first multiplication, it is instead beneficial to transform the other operand of the addition into the NTT domain as well and then output the ciphertext in NTT representation. By doing so, one can save one NTT transformation during decryption.

To speed up the modular reduction, one should consider transforming the coefficients of the polynomials into the Montgomery domain. Unfortunately, a Montgomery transformation is quite costly on microcontrollers, since it requires an additional point-wise multiplication with subsequent modular reduction. However, depending on the parameter set, and the target platform, the benefits are likely to outweigh the cost. On some architectures it is useful to only apply the modular reduction after every few operations, so that the result can be larger than the modulus, but still fit into one data word.

Of particular interest on microcontrollers is the use of on-board features, such as true random number generators, which can be used to output pseudo-random and uniformly distributed values. DSP instructions, such as multiply-accumulate, can also be useful to speed up the NTT. For applications where the communication cost matters more than the processing time, it is beneficial to apply compression techniques (e.g. Huffman encoding) to signatures.

On-board vector processing units such as AVX or Neon can greatly increase the performance of an implementation. Especially linear algebraic operations, which are usually well-suited for parallelisation. Efficient memory access schemes, such as those described in [131], help to exploit the full potential of vectorisation.

4.4 State-of-the-Art Attacks and Countermeasures for Lattice-Based Crypto

There has been limited research on the side-channel vulnerabilities of lattice-based implementations, with the majority of work focussing on the NTRU public key cryptosystem [132], formalised in the IEEE standard P1363.1-2008 [133]. In this section we present a summary of the state-of-the-art in physical lattice-based attacks.

Roy *et al.* proposed a compact implementation of the Knuth-Yao sampler in [127], having a limited statistical distance to a true discrete Gaussian distribution. Two of them are optimised for performance (area and time), while the third one is robust against power and timing attacks. The area-optimised implementation of the bit-scan operation based Knuth-Yao sampler requires on average 17 cycles to generate a sample and can be implemented in 30 slices on a Xilinx Virtex 5 FPGA. A further optimised implementation, which uses a precomputed table to map the initial random bits into samples with very high probability, requires 35 slices and generates a sample in approximately 2.5 cycles. Side-channel resistance is achieved using random shuffling, which protects the Gaussian distributed polynomial. This implementation occupies 52 slices and requires 420 cycles to generate a polynomial of 256 coefficients.

Reparaz *et al.* presented a masked ring-LWE decryption implementation resistant to first-order side-channel attacks in [134]. The main idea of this work is to carry out the entire computation in the masked domain, which is obtained by using a dedicated masked decoder. The hardware

implementation of the architecture, measured on a Virtex-II FPGA, occupies around 2000 LUTs; which is approximately 20% more than the reference unprotected implementation. Performance-wise, the protected implementation requires 7478 cycles to complete a computation, which is approximately 2.6 times slower than its unprotected counterpart.

Silverman and Whyte described a timing attack on the implementation of NTRUEncrypt in [135].The attack relies on the fact that decryption of different (possibly bogus) ciphertexts may require a different number of calls to a hash function. To mount the attack, the adversary performs a variable number of precomputations, and then submits a relatively small number of specially constructed ciphertexts for decryption, measuring the decryption times. Comparison of the decryption times with the precomputed data enables the attacker to recover the key in a much reduced time compared to standard attacks on NTRUEncrypt. Reported results show that for specific parameter sets, an attacker can recover a single key with approximately k/2 bits of effort. The work highlights possible ways to prevent the attack by ensuring a constant number of SHA calls.

In [136], Vizev exploited the differing number of hash calls to mount timing side-channel attacks. The proposed countermeasure consists of a padding scheme, which helps ensure a constant timing of operations. A constant time sampler was also used in [145] for key-exchange in the transport layer security (TLS) protocol, based on a R-LWE implementation.

Atici *et al.* presented the first power analysis attacks on NTRU in [137], targeting implementations for RFIDs. This was followed by the paper of Lee *et al.* [138], which considered first and second order DPA attacks on NTRU. The attacks were based on the leakage of Hamming distance information, generated during the computation of the convolution product. The authors proposed three countermeasures, with the aim to thwart both first and second order attacks. The first countermeasure proposed a random initialisation of every register used in the convolution operation, with the random value then subtracted at the end of processing the final result. The second proposal was to blind the intermediate convolution steps, each with a separate random integer value. The final countermeasure was to randomise the order of the array holding the b coefficients, the non-zero polynomials.

In [139], Wang *et al.* considered the countermeasure proposals of [138] and suggested that blinding during the computation of the convolution products was not sufficient, since it could eventually leak information on the real 't' values with enough DPA measurements. A chosen plaintext attack was described to exploit the calculation of intermediate values. Even though illegal intermediate values would be generated, and these values would be prohibited from being output, their calculation during convolution processing stages would still be recordable in the power consumption measurements. The authors then proposed alternative countermeasures, based on random delay insertions, an alternative masking scheme and the use of dummy operations. The countermeasures were implemented in a software-environment by inserting a random number of NOP instructions before the beginning of sensitive operations and by incorporating a number of dummy convolution products and modular arithmetic operations. The authors did not give any indication relating to the performance loss due to the incorporation of these countermeasures.

In [140], Sheng *et al.* showed that a collision attack could break the combined countermeasures proposed in [138] and that their attacks were more efficient than the second order attacks demonstrated in [138]. Three countermeasures were proposed, the insertion of dummy operations, the insertion of random delays between processing operations, and the random cyclical rotation of the key and ciphertext.

A fault analysis attack against NTRUEncrypt was presented in [141] by Kamal and Youssef, where the fault model assumes the attacker is able to inject faults into the coefficients of the second step of the decryption process. Where NTRUEncrypt is implemented with parameters (N, p, q), the attack is shown to be successful with probability$\approx 1 - \frac{1}{p}$. In [143], Kamal and Youssef proposed methods for

strengthening hardware implementations of NTRUEncrypt against fault analysis attacks using error detection codes and duplication of the decryption operation, using a rotated version of the ciphertext in a redundant computation.

A scan based side-channel attack on NTRUEncrypt was demonstrated in [144], where the scan chain structure of the polynomial multiplication circuits was extracted, thus enabling the secret key to be retrieved.

Kamal and Youssef also presented a fault analysis of the NTRUSign digital signature scheme in [142], where it is assumed that the attacker is able to inject a transient fault into the coefficients of the polynomials in the signing algorithm. When the attacker is also able to skip the norm-bound signature check, the attack requires only one fault for success.

Errors can be intentionally introduced to a system via fault attacks, but they may also arise as a natural consequence of the mathematical constructs of the implementation, for example, calculations that have permissible error rates. When such events occur, an error recovery mechanism will detect the error and instantiate a new calculation. It is unclear at present whether such errors would make attacks more difficult, since the errors would be uncorrelated with attack models, or whether it is possible that such errors, and associated error recovery features, could then leak exploitable information. Naturally one should err on the side of caution, and we therefore recommend that, where relevant, the aspect of permissible error rates, and associated error recovery, should be a factor for consideration when developing lattice-based countermeasures.

4.5 Concluding Remarks

We have seen that lattice-based implementations are primarily concerned with high levels of linear algebraic operations and the sampling of values from a discrete Gaussian distribution. In terms of side-channel leakages, power will be an important consideration, but also timing since many of the algorithms proposed for sampling from the discrete Gaussian distributions would have timing vulnerabilities if implemented in a naïve sense.

Implementations that target constant timing may suffer performance loss due to the need for all operations to complete in the same length of time as the slowest operation. This conflicts with the requirement to provide optimal throughput, so the designer will have to consider what other optimisations can be incorporated to mitigate these effects, for example through the use of on-board vector processing units, such as AVX or Neon, and through the use of pre-computations or parallelisation activities, in a trade off with area on hardware-based platforms.

In the previous section we surveyed the existing known attacks, primarily they were based on NTRU and elements such as the Gaussian sampler. We can see that the existing side-channels of power, timing and fault analysis have already started to be exploited and, as lattice-based crypto deployments increase in number, we expect there to be many more attacks described. We can expect that countermeasures such as masking, constant time, randomisation and fault detection will be of importance for lattice-based security.

5 Conclusions

In this report we have considered the threats from physical attacks to both the existing class of cryptographic devices and to the new class of lattice-based implementations. Although lattice-based primitives have been investigated for their resilience from quantum attacks, such as Shor's algorithm, they will nevertheless require implementation in existing CMOS technologies, and will therefore be susceptible to the same class of physical attacks as existing cryptographies, namely side-channel attacks. These kinds of attacks are attractive and accessible to the adversary due to their increasingly low cost, and often, the low level of *a-priori* knowledge required of the implementation.

A survey has been presented on the various existing attacks, such as power analysis, EM analysis, timing attacks, fault attacks, collision attacks, profiling attacks, multi-channel attacks, and machine learning attacks. We consider that all of these attacks will be a potential threat to lattice-based implementations. As new lattice-based designs emerge, and the number of deployments increase, we expect to see further new attacks described in the literature that exploit the particular characteristics of the lattice-based implementation.

In terms of performance and security, the implementer will need to carefully consider the use of optimisations, since, for example, the use of lookup tables, early exiting of loops and branching based on secret data, can all lead to a non-constant time of operation. The re-use of libraries, which have been developed with security in mind, can help reduce the likelihood of introducing unintended vulnerabilities in the software context.

Of particular interest on microcontrollers is the use of on-board features. For instance, some boards feature a true random number generator that can be used to output pseudo-random and uniformly distributed values. DSP instructions, such as multiply-accumulate, can also be useful to speed up the NTT. For applications where the communication cost matters more than the processing time, it is beneficial to apply compression techniques (e.g. Huffman encoding) to signatures. On-board vector processing units such as AVX or Neon can greatly increase the performance; particularly linear algebraic operations, which are usually well-suited for parallelisation. However, care should be taken to consider the potential side-channel leakages when employing any such features.

Countermeasures to address specific threats, as they emerge, will be an important area of research going forward. However, the general countermeasure approaches, already developed and in-use, will offer a level of protection against the existing known threats and these will undoubtedly be useful when considering future designs of countermeasures for lattice-based crypto. We can expect that existing countermeasures such as masking, constant time, randomisation and fault detection will all have an important role to play in lattice-based security. As improvements in the sophistication of attacks continue, both new and existing countermeasures will require ongoing evaluation, ensuring that they continue to offer their claimed levels of protection.

6 References

[1] S. Mangard, E. Oswald, and T. Popp. ”Power Analysis Attacks: Revealing the Secrets of Smart Cards.” Springer, 2007.

[2] P. Kocher. “Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and other Systems.” in Advances in Cryptology (CRYPTO ’96). Lecture Notes in Computer Science, Vol. 1109, pp. 104-113. Springer, 1996.

[3] D. Boneh, R. DeMillo, and R. Lipton. “On the importance of checking cryptographic protocols for faults” (Eurocrypt ’97), Lecture Notes in Computer Science, Vol. 1233, pp. 37–51. Springer, 1997.

[4] E. Tromer "Acoustic cryptanalysis: on nosy people and noisy machines." (Eurocrypt2004) Rump Session, 2004.

[5] S. Skorobogatov and R. Anderson. "Optical fault induction attacks." In Cryptographic Hardware and Embedded Systems (CHES 2002). Lecture Notes in Computer Science, Vol. 2523, 31-48. Springer, 2003.

[6] P. Kocher, J. Jaffe and B. Jun, “Differential Power Analysis” in Advances in Cryptology (CRYPTO ’99). Lecture Notes in Computer Science, Vol. 1666, pp. 388-397. Springer, 1999.

[7] J. J. Quisquater, D. Samyde. “ElectroMagnetic Analysis (EMA): Measures and Counter-Measures for Smart Cards.” In Smart Card Programming and Security (E-smart 2001). Lecture Notes in Computer Science, Vol. 2140, pp.200-210. Springer, 2001.

[8] K. Gandolfi, C. Mourtel, and F. Olivier. “Electromagnetic Attacks: Concrete Results,” In Cryptographic Hardware and Embedded Systems (CHES 2001). Lecture Notes in Computer Science, Vol. 2162, pp. 251-261. Springer, 2001.

[9] D. Agrawal, B. Archambeault, J. R. Rao, and P Rohatgi. “The EM Side-Channel(s),” In Cryptographic Hardware and Embedded Systems (CHES 2002). Lecture Notes in Computer Science, Vol. 2523, pp 29-45. Springer, 2002.

[10] E. Brier, C. Clavier, and F. Olivier. "Correlation power analysis with a leakage model." In Cryptographic Hardware and Embedded Systems-CHES 2004, pp. 16-29. Springer Berlin Heidelberg, 2004.

[11] S. Messerges. "Using second-order power analysis to attack DPA resistant software" In Cryptographic Hardware and Embedded Systems (CHES 2000), Lecture Notes in Computer Science, Vol. 1965, pp. 238-251. Springer, 2000.

[12] J. Waddle and D. Wagner. "Towards efficient second-order power analysis." In Cryptographic Hardware and Embedded Systems (CHES 2004). Lecture Notes in Computer Science, Vol. 3156, pp. 1-15. Springer, 2004.

[13] E. Oswald, S. Mangard, C. Herbst, and S. Tillich. "Practical second-order DPA attacks for masked smart card implementations of block ciphers." In Topics in Cryptology–CT-RSA 2006, pp. 192-207. Springer Berlin Heidelberg, 2006.

[14] K. Schramm, T. Wollinger, and C. Paar, “A new class of collision attacks and its application to DES.” In Fast Software Encryption pp. 206-222. Springer Berlin Heidelberg, 2003.

[15] K. Schramm, G. Leander, P. Felke, and C. Paar, “A Collision-Attack on AES: Combining Side Channel- and Differential-Attack” CHES 2004: pp. 163-175, 2004.

[16] A. Bogdanov. "Improved side-channel collision attacks on AES." In Selected Areas in Cryptography, pp. 84-95. Springer Berlin Heidelberg, 2007.

[17] A. Bogdanov and Andrei Pyshkin, "Algebraic Side-Channel Collision Attacks on AES." http://eprint.iacr.org/2007/477.pdf. Cryptology ePrint Archive, 2007.

[18] A. Bogdanov, “Multiple-Differential Side-Channel Collision Attacks on AES”, CHES 2008: pp 30-44, 2008.

[19] J. Golić and C. Tymen. "Multiplicative masking and power analysis of AES." In Cryptographic Hardware and Embedded Systems-CHES 2002, pp. 198-212. Springer Berlin Heidelberg, 2003.

[20] L. Goubin. "A refined power-analysis attack on elliptic curve cryptosystems." In Public key cryptography—PKC 2003, pp. 199-211. Springer Berlin Heidelberg, 2002.

[21] T. Akishita and T. Takagi. “Zero-value point attacks on elliptic curve cryptosystem.” Springer Berlin Heidelberg, 2003.

[22] P. Fahn and P. Pearson. "IPA: A new class of power attacks." In Cryptographic Hardware and Embedded Systems (CHES 1999). Lecture Notes in Computer Science, Vol. 1717, pp. 173-186. Springer, 1999.

[23] S. Chari, J. Rao and P. Rohatgi "Template Attacks" In Workshop on Cryptographic Hardware and Embedded Systems (CHES 2002). Lecture Notes in Computer Science, Vol. 2523, pp. 13-28. Springer, 2003.

[24] C. Rechberger and E. Oswald. "Practical template attacks," 5th International Workshop on Information Security Applications (WISA 2004). Lecture Notes in Computer Science, vol. 3325, pp. 443–457. Springer, 2004.

[25] B. Gierlichs. "Signal theoretical methods in differential side channel cryptanalysis." Ruhr-Universität Bochum, 2006.

[26] S. Bhattacharya and D. Mukhopadhyay. "Who watches the watchmen?: Utilizing Performance Monitors for Compromising keys of RSA on Intel Platforms." In Cryptographic Hardware and Embedded Systems--CHES 2015, pp. 248-266. Springer Berlin Heidelberg, 2015.

[27] G. Hospodar, B. Gierlichs, E. De Mulder, I. Verbauwhede and J. Vandewalle. “Machine learning in side-channel analysis: a first study.” In Journal of Cryptographic Engineering, volume 1, number 4, pages 293—302, 2011.

[28] A. Heuser and M. Zohner. “Intelligent Machine Homicide: Breaking Cryptographic Devices Using Support Vector Machines.” W. Schindler and S.A. Huss, ed., Constructive Side-Channel Analysis and Secure Design -- COSADE 2012, volume 7275 of series LNCS, pages 249—264, 2012.

[29] L. Lerman, G. Bontempi and O. Markowitch. “Side Channel Attack: an Approach based on Machine Learning.” Constructive Side-Channel Analysis and Secure Design -COSADE 2011, 2011.

[30] L. Lerman, S. F. Medeiros, G. Bontempi, and O. Markowitch, “A Machine Learning Approach Against a Masked AES,” in Smart Card Research and Advanced Application Conference — CARDIS 2013, ser. Lecture Notes in Computer Science, vol. to appear. Springer-Verlag, 2013.

[31] R. Gilmore, N. Hanley, M. O’Neill. “Neural Network Based Attack on a Masked Implementation of AES.” IEEE International Symposium on Hardware Orientated Security and Trust, 2015.

[32] J. Chou, M. Chu, Y. Tsai, and Y. Jin. “An Unsupervised Learning Model to Perform Side Channel Attack.” Advances in Knowledge Discovery and Data Mining Lecture Notes in Computer Science Volume 7818, 2013, pp 414-425, 2013.

[33] L. Bohy, M. Neve, D. Samyde, and J.-J. Quisquater. “Principal and independent component analysis for crypto-systems with hardware unmasked units.” In Proceedings of e-Smart 2003, 2003.

[34] M. Cucuringu, V. Blondel, and P. Van Dooren. "Extracting spatial information from networks with low-order eigenvectors." Physical Review E 87, no. 3 (2013): 032803, 2013

[35] M., Cucuringu, M.W., Mahoney: Localization on low-order eigenvectors of data matrices. Technical Report, 2011.

[36] L. Batina, J. Hogenboom, J. van Woudenberg. “Getting More from PCA: First Results of Using Principal Component Analysis for Extensive Power Analysis.” In Dunkelman, O. (ed.) CT-RSA 2012. LNCS, vol. 7178, pp. 383–397. Springer Heidelberg, 2012.

[37] D. Mavroeidis, L. Batina, and T. Marchiori. “PCA, Eigenvector Localization and Clustering for Side-Channel Attacks on Cryptographic Hardware Devices.” In Machine Learning and Knowledge Discovery in Databases Lecture Notes in Computer Science Vol. 7523, 2012, pp 253-268, 2012.

[38] “TEMPEST: a Signal Problem” The story of the discovery of various compromising emanations from communications and Comsec equipment. http://www.nsa.gov/public_info/_files/cryptologic_spectrum/tempest.pdf, 2007.

[39] S. Skorobogatov. "Semi-invasive attacks - a new approach to hardware security analysis." In technical report, University of Cambridge, Computer Laboratory, 2005.

[40] J.-F Dhem, F. Koeune, P.-A. Leroux, P. Mestré, J.-J. Quisquater, and J.-L. Willems. "A practical implementation of the timing attack." In Smart Card Research and Applications, pp. 167-182. Springer Berlin Heidelberg, 2000.

[41] D. Bernstein "Cache-timing attacks on AES." 2005.

[42] D. Brumley and D. Boneh. "Remote timing attacks are practical." Computer Networks 48, no. 5, pp. 701-716, 2005.

[43] D. Page. "Theoretical Use of Cache Memory as a Cryptanalytic Side-Channel." IACR Cryptology ePrint Archive, http://eprint.iacr.org/2002/169.pdf, 2002.

[44] Y. Tsunoo, T. Saito, T. Suzaki, M. Shigeri, and H. Miyauchi. "Cryptanalysis of DES implemented on computers with cache." In Cryptographic Hardware and Embedded Systems-CHES 2003, pp. 62-76, Springer Berlin Heidelberg, 2003.

[45] E. Tromer, D. Osvik, and A. Shamir. "Efficient cache attacks on AES, and countermeasures." Journal of Cryptology 23, no. 1 (2010): 37-71, 2010.

[46] R. Anderson and M. Kuhn. “Tamper resistance—a cautionary note” In Proc. of Second USENIX Workshop on Electronic Commerce, pp. 1–11, 1996.

[47] R. Anderson and M. Kuhn. "Low cost attacks on tamper resistant devices" In Proc. of 5th Security Protocols Workshop, Lecture Notes in Computer Science, Vol. 1361, pp. 125–136. Springer, 1997.

[48] D. Boneh, R. DeMillo, and R. Lipton. "On the importance of eliminating errors in cryptographic computations." Journal of cryptology 14, no. 2 (2001): 101-119, 2001.

[49] I. Biehl, B. Meyer, and V. Müller. "Differential fault attacks on elliptic curve cryptosystems." In Advances in Cryptology—CRYPTO 2000, pp. 131-146. Springer Berlin Heidelberg, 2000.

[50] J. Biernat and M. Nikodem. "Fault cryptanalysis of ElGamal signature scheme." In Computer Aided Systems Theory–EUROCAST 2005, pp. 327-336. Springer Berlin Heidelberg, 2005.

[51] C. Giraud, E. Knudsen, and M. Tunstall. "Improved fault analysis of signature schemes." In Smart Card Research and Advanced Application, pp. 164-181. Springer Berlin Heidelberg, 2010.

[52] G. Piret and J.-J. Quisquater. "A differential fault attack technique against SPN structures, with application to the AES and KHAZAD." In Cryptographic Hardware and Embedded Systems-CHES 2003, pp. 77-88. Springer Berlin Heidelberg, 2003.

[53] C. Kim and J.-J. Quisquater. "New differential fault analysis on AES key schedule: two faults are enough." In Smart Card Research and Advanced Applications, pp. 48-60. Springer Berlin Heidelberg, 2008.

[54] H. Bar-Eli, H. Choukri, D. Naccache, M. Tunstall, and C. Whelan. "The sorcerer's apprentice guide to fault attacks." Proceedings of the IEEE 94, no. 2 (2006): 370-382, 2006.

[55] J. Hoch and Adi Shamir. "Fault analysis of stream ciphers." In Cryptographic Hardware and Embedded Systems-CHES 2004, pp. 240-253. Springer Berlin Heidelberg, 2004.

[56] A. Barenghi, L. Breveglieri, I. Koren, and D. Naccache, "Fault Injection Attacks on Cryptographic Devices: Theory, Practice, and Countermeasures" Proceedings of the IEEE 100(11): 3056-3076, 2012.

[57] A. Barenghi, C. Hocquet, D. Bol, F.-X. Standaert, F. Regazzoni, and I. Koren. "A combined design-time/test-time study of the vulnerability of sub-threshold devices to low voltage fault attacks." Emerging Topics in Computing, IEEE Transactions on 2, no. 2 (2014): 107-118, 2014.

[58] Y. Ren, A. Wang, and L. Wu. "Transient-Steady Effect Attack on Block Ciphers." In Cryptographic Hardware and Embedded Systems--CHES 2015, pp. 433-450. Springer Berlin Heidelberg, 2015.

[59] C. Clavier, J. Coron, and N. Dabbous. "Differential power analysis in the presence of hardware countermeasures" In Cryptographic Hardware and Embedded Systems (CHES 2000). Lecture Notes in Computer Science, Vol. 1965, pp.252-263. Springer, 2000.

[60] N. Homma, S. Nagashima, Y. Imai, T. Aoki, and A. Satoh. "High-resolution side-channel attack using phase-based waveform matching." In Cryptographic Hardware and Embedded Systems (CHES 2006). Lecture Notes in Computer Science, Vol. 4249, pp. 187-200. Springer, 2006.

[61] N. Homma, S. Nagashima, T. Sugawara, and A. Satoh. "A high-resolution phase-based waveform matching and its application to side-channel attacks". In IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 91(1), pp. 193-202, 2008.

[62] C. Gebotys, S. Ho, and A. Tiu. "EM Analysis of Rijndael and ECC on a Wireless Java-Based PDA." In Workshop on Cryptographic Hardware and Embedded Systems (CHES 2005). Lecture Notes in Computer Science, pp. 250-264. Springer, 2005.

[63] P. Hodgers, K.-H. Boey and M. O'Neill. "Variable window power spectral density attack." In Information Forensics and Security (WIFS 2011), IEEE International Workshop on, pp. 1-6. IEEE, 2011.

[64] P. Hodgers, N. Hanley, and M. O'Neill. "Pre-processing power traces to defeat random clocking countermeasures." In Circuits and Systems (ISCAS), 2015 IEEE International Symposium on, pp. 85-88. IEEE, 2015.

[65] K. Tiri, M. Akmal, and I. Verbauwhede. "A dynamic and differential CMOS logic with signal independent power consumption to withstand differential power analysis on smart cards." In Solid-State Circuits Conference (ESSCIRC 2002). Proceedings of the 28th European, pp. 403-406. IEEE, 2002.

[66] F. Mace, F.-X. Standaert, I. Hassoune, and J.-D. Legat. “A Dynamic Current Mode Logic to Counteract Power Analysis Attacks.” In proceedings of Design of Circuits and Integrated Systems (DCIS 2004), pp. 186-191, 2004.

[67] Z. Toprak and Y. Leblebici. "Low-power current mode logic for improved DPA-resistance in embedded systems." In Circuits and Systems (ISCAS 2005). IEEE International Symposium on pp. 1059-1062, 2005.

[68] I. Hassoune, F. Macé, D. Flandre, and J. D. Legat. "Low-swing current mode logic (LSCML): A new logic style for secure and robust smart cards against power analysis attacks" Microelectronics Journal, 37(9), pp. 997-1006, 2006.

[69] K. Tiri and I. Verbauwhede. "A logic level design methodology for a secure DPA resistant ASIC or FPGA implementation." In Proceedings of the conference on Design, automation and test in Europe-Volume 1 p. 10246). IEEE Computer Society, 2004.

[70] S. Chari, C. Jutla, J. Rao, and P. Rohatgi. "Towards sound approaches to counteract power-analysis attacks" In Advances in Cryptology (CRYPTO ’99). Lecture Notes in Computer Science, Vol. 1666, pp. 398-412. Springer, 1999.

[71] M.-L. Akkar and C. Giraud. "An implementation of DES and AES, secure against some attacks" In Cryptographic Hardware and Embedded Systems (CHES 2001). Lecture Notes in Computer Science, Vol. 2162, pp. 309-318. Springer, 2001.

[72] E. Oswald, S. Mangard, and N. Pramstaller. "Secure and Efficient Masking of AES - A Mission Impossible?" http://eprint.iacr.org/2004/134.pdf. Cryptology ePrint Archive, 2004.

[73] E. Trichina. "Combinational logic design for AES sub-byte transformation on masked data." http://eprint.iacr.org/2003/236.pdf. Cryptology ePrint Archive, 2003.

[74] S. Mangard, T. Popp, and B. Gammel. "Side-channel leakage of masked CMOS gates." Topics in Cryptology (CT-RSA 2005), vol. 3376, pp. 351-365. Springer, 2005.

[75] D. Suzuki, M. Saeki, and T. Ichikawa. "Random switching logic: A countermeasure against DPA based on transition probability." http://eprint.iacr.org/2004/346.pdf. Cryptology ePrint Archive, 2004.

[76] D. Suzuki, M. Saeki, and T. Ichikawa. "Random switching logic: A new countermeasure against DPA and second-order DPA at the logic level." In IEICE transactions on fundamentals of electronics, communications and computer sciences, 90(1), 160-168, 2007.

[77] P. Schaumont and K. Tiri. "Masking and Dual Rail Logic Don’t Add Up." In Cryptographic Hardware and Embedded Systems (CHES 2007). Lecture Notes in Computer Science, vol. 4727. pp. 95–106. Springer, 2007.

[78] A. Moradi, O. Mischke, and T. Eisenbarth. "Correlation-enhanced power analysis collision attack." In Cryptographic Hardware and Embedded Systems, CHES 2010, pp. 125-139. Springer Berlin Heidelberg, 2010.

[79] E. Oswald, S. Mangard, and N. Pramstaller. "Secure and efficient masking of AES-a mission impossible?" IACR Cryptology ePrint Archive 2004: 134, 2004.

[80] E. Oswald, S. Mangard, N. Pramstaller, and V. Rijmen. "A side-channel analysis resistant description of the AES S-box." In Fast Software Encryption, pp. 413-423. Springer Berlin Heidelberg, 2005.

[81] O. Kömmerling and M. Kuhn. "Design principles for tamper-resistant smartcard processors" In Proceedings of the USENIX Workshop on Smartcard Technology on USENIX Workshop on Smartcard Technology, pp. 2-2, 1999.

[82] M. Yamaguchi, H. Toriduka, S. Kobayashi, T. Sugawara, N. Homma, A. Satoh, and T. Aoki. "Development of an on-chip micro shielded-loop probe to evaluate performance of magnetic film to protect a cryptographic LSI from electromagnetic analysis." In Electromagnetic Compatibility (EMC), International Symposium on, pp. 103-108. IEEE, 2010.

[83] P. Kocher, J. Jaffe and B. Jun. "Differential power analysis method and apparatus." U.S. Patent 7,587,044, issued September 8th, 2009.

[84] D. Chaum. "Blind signatures for untraceable payments." In Advances in cryptology, pp. 199-203. Springer, 1983.

[85] W. Schindler. "A timing attack against RSA with the Chinese remainder theorem." In Cryptographic Hardware and Embedded Systems—CHES 2000, pp. 109-124. Springer Berlin Heidelberg, 2000.

[86] D. Page. "Defending against cache-based side-channel attacks." In Information Security Technical Report, vol. 8, issue 8, 2003.

[87] E. Biham. "A fast new DES implementation in software." In Fast Software Encryption, pp. 260-272. Springer Berlin Heidelberg, 1997.

[88] Cryptocoding.net, "Cryptographic Coding Standards" https://cryptocoding.net/index.php/Cryptography_Coding_Standard, 2013.

[89] NaCl: Networking and Cryptography library, https://nacl.cr.yp.to/.

[90] Advanced Encryption Standard New Instructions (AES-NI). https://software.intel.com/en-us/articles/intel-advanced-encryption-standard-instructions-aes-ni/.

[91] Intel Advanced Encryption Standard (AES) New Instructions Set, White Paper. https://software.intel.com/sites/default/files/article/165683/aes-wp-2012-09-22-v01.pdf

[92] S. Wolter, H. Matz, A. Schubert, and R. Laur. "On the VLSI implementation of the international data encryption algorithm IDEA." In Circuits and Systems, 1995. ISCAS'95., 1995 IEEE International Symposium on, vol. 1, pp. 397-400. IEEE, 1995.

[93] G. Gaubatz and B. Sunar. "Robust finite field arithmetic for fault-tolerant public-key cryptography." In Fault Diagnosis and Tolerance in Cryptography, pp. 196-210. Springer Berlin Heidelberg, 2006.

[94] R. Karri, G. Kuznetsov, and M. Goessel. "Parity-based concurrent error detection of substitution-permutation network block ciphers." In Cryptographic Hardware and Embedded Systems-CHES 2003, pp. 113-124. Springer Berlin Heidelberg, 2003.

[95] G. Bertoni, L. Breveglieri, I. Koren, P. Maistri, and V. Piuri. "Error analysis and detection procedures for a hardware implementation of the advanced encryption standard." Computers, IEEE Transactions on 52, no. 4 (2003): 492-505, 2003.

[96] A. Shamir. "Method and apparatus for protecting public key schemes from timing and fault attacks." U.S. Patent 5,991,415, issued November 23, 1999.

[97] C. Kim and J.-J. Quisquater. "Fault attacks for CRT based RSA: New attacks, new results, and new countermeasures." In Information Security Theory and Practices. Smart Cards, Mobile and Ubiquitous Computing Systems, pp. 215-228. Springer Berlin Heidelberg, 2007.

[98] S.-M Yen, S. Kim, S. Lim, and S.-J. Moon. "RSA speedup with Chinese remainder theorem immune against hardware fault cryptanalysis." Computers, IEEE Tras. on 52, no. 4 (2003): 461-472, 2003

[99] F.-X. Standaert, T. Malkin, and M. Yung. "A unified framework for the analysis of side-channel key recovery attacks." In Advances in Cryptology-EUROCRYPT 2009, pp. 443-461. Springer Berlin Heidelberg, 2009.

[100] G. Goodwill, B. Jun, J. Jaffe, and P. Rohatgi, "A testing methodology for side-channel resistance validation." In NIST Non-invasive attack testing workshop, 2011.

[101] T. Schneider and A. Moradi. “Leakage Assessment Methodology - A Clear Roadmap for Side-Channel Evaluations.” CHES 2015: 495-513, 2015.

[102] M. Tiwari, X. Li, H. Wassel, B. Mazloom, S. Mysore, F. Chong, and T. Sherwood, “Gate-level information flow tracking for secure architectures”, IEEE Micro, vol. 30, no. 1, pp. 92–100, 2010.

[103] K. Tiri and I. Verbauwhede. “A digital design flow for secure integrated circuits.” IEEE Trans. Computer-Aided Design Integration. Circuits Systems, vol. 25, no. 7, pp. 1197–1208, 2006.

[104] S. Guilley, P. Hoogvorst, Y. Mathieu, and R. Pacalet, “The backend duplication method.” In Proc. Cryptographic Hardware Embedded Syst. (CHES 05), Aug. 2005, vol. 3659, pp. 383–397, 2005.

[105] F. Regazzoni, A. Cevrero, F.-X. Standaert, S. Badel, T. Kluter, P. Brisk, Y. Leblebici, and P. Ienne, “A design flow and evaluation framework for DPA-resistant instruction set extensions.” In Proc. Cryptographic Hardware Embedded Syst. (CHES 09) , pp. 205–219, 2009.

[106] CACE European Project, Computer Aided Cryptography Engineering. http://www.cace-project.eu

[107] M. Barbosa, A. Moss, and D. Page, “Constructive and destructive use of compilers in elliptic curve cryptography.” J. Cryptol., vol. 22, no. 2, pp. 259 –281, 2009.

[108] A. Bayrak, F. Regazzoni, P. Brisk ,F.-X. Standaert, and P. Ienne, “A first step towards automatic application of power analysis counter-measures.”, in Proc. 48th ACM/EDAC/IEEE Design Autom. Conf.(DAC’11), pp. 230–235, 2011.

[109] V. Cleemput, B. Coppens, and B. de Sutter, “Compiler mitigations for time attacks on modern x86 processors.”, ACMTrans. Archit. Code Optim. , vol. 8, no. 4, pp. 1–20, 2012.

[110] F. Regazzoni, T. Eisenbarth, J. Grossschadl, and L. Breveglieri. "Power attacks resistance of cryptographic s-boxes with added error detection circuits." In Defect and Fault-Tolerance in VLSI Systems, 2007. DFT'07. 22nd IEEE International Symposium on, pp. 508-516. IEEE, 2007.

[111] F. Regazzoni, T. Eisenbarth, L. Breveglieri, P. Ienne, and I. Koren. "Can knowledge regarding the presence of countermeasures against fault attacks simplify power attacks on cryptographic devices?" In Defect and Fault Tolerance of VLSI Systems, 2008. DFTVS'08. IEEE International Symposium on, pp. 202-210. IEEE, 2008.

[112] F. Regazzoni, L. Breveglieri, P. Lenne, and I. Koren. "Interaction Between Fault Attack Countermeasures and the Resistance Against Power Analysis Attacks." In Fault Analysis in Cryptography, pp. 257-272. Springer Berlin Heidelberg, 2012.

[113] T. Güneysu, V. Lyubashevsky, and T. Pöppelmann. "Practical lattice-based cryptography: A signature scheme for embedded systems." In Cryptographic Hardware and Embedded Systems–CHES 2012, pp. 530-547. Springer Berlin Heidelberg, 2012.

[114] T. Pöppelmann and T. Güneysu. "Towards efficient arithmetic for lattice-based cryptography on reconfigurable hardware." In Progress in Cryptology–LATINCRYPT 2012, pp. 139-158. Springer Berlin Heidelberg, 2012.

[115] A. Aysu, C. Patterson, and P. Schaumont. "Low-cost and area-efficient FPGA implementations of lattice-based cryptography." In Hardware-Oriented Security and Trust (HOST), 2013 IEEE International Symposium on, pp. 81-86. IEEE, 2013.

[116] S. Roy, F. Vercauteren, N. Mentens, D. Chen, and I. Verbauwhede. "Compact ring-LWE cryptoprocessor." In Cryptographic Hardware and Embedded Systems–CHES 2014, pp. 371-391. Springer Berlin Heidelberg, 2014.

[117] D. Chen, N. Mentens, F. Vercauteren, S. Roy, R. Cheung, D. Pao, and I. Verbauwhede. "High-speed polynomial multiplication architecture for ring-LWE and SHE cryptosystems." Circuits and Systems I: Regular Papers, IEEE Transactions on 62, no. 1 (2015): 157-166, 2015.

[118] T. Pöppelmann, T. Oder, and T. Güneysu. "High-Performance Ideal Lattice-Based Cryptography on 8-Bit ATxmega Microcontrollers." In Progress in Cryptology--LATINCRYPT 2015, pp. 346-365. Springer International Publishing, 2015.

[119] P. Barrett. “Implementing the Rivest Shamir and Adleman Public Key Encryption Algorithm on a Standard Digital Signal Processor.” In Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 311–323. Springer, Heidelberg, 1987.

[120] L. Ducas, A. Durmus, T. Lepoint, and V. Lyubashevsky. “Lattice signatures and bimodal Gaussians”, CRYPTO 2013, 2013.

[121] C. Peikert. “An Efficient and Parallel Gaussian Sampler for Lattices”, CRYPTO 2010, 2010.

[122] N. Dwarakanath and S. Galbraith, “Sampling from discrete Gaussians for lattice-based cryptography on a constrained device” App. Algebra Eng. Commun. Comput. 25(3): 159-180, 2014.

[123] J. Buchmann, D. Cabarcas, F. Göpfert, A. Hülsing, and P. Weiden. “Discrete Ziggurat: A time-memory trade-off for sampling from a Gaussian distribution over the integers“ SAC 2013: 402-417, 2013.

[124] M.-J Saarinen. “Gaussian Sampling Precision in Lattice Cryptography”, https://eprint.iacr.org/2015/953.pdf, Cryptology ePrint Archive, 2015.

[125] T. Pöppelmann, L. Ducas, and T. Güneysu. “Enhanced Lattice-based Signatures on Reconfigurable Hardware” CHES 2015: 353-370, 2015.

[126] S. Roy, F. Vercauteren, and I. Verbauwhede, “High Precision Discrete Gaussian Sampling on FPGAs”, SAC 2013, 2013

[127] S. Roy, O. Reparaz, F. Vercauteren, and I. Verbauwhede, “Compact and Side Channel Resistant Discrete Gaussian Sampling” http://eprint.iacr.org/2014/591.pdf, Cryptology ePrint Archive, 2014.

[128] G. Marsaglia and W. Tsang, “The Ziggurat Method for Generating Random Variables”, Journal of Statistical Software, 2000.

[129] T. Oder, T. Pöppelmann, and T. Güneysu, “Beyond ECDSA and RSA: Lattice-based Digital Signatures on Constrained Devices”, DAC 2014, 2014

[130] K. Takashima and A. Takayasu, “Tighter Security for Efficient Lattice Cryptography via the Rényi Divergence of Optimized Orders”, https://eprint.iacr.org/2015/1132.pdf, Cryptology ePrint Archive, 2015.

[131] Ö. Dagdelen, R. El Bansarkhani, F. Göpfert, T. Güneysu, T. Oder, T. Pöppelmann, A. Sánchez, and P. Schwabe, “High-speed signatures from standard lattices”, In Progress in Cryptology-LATINCRYPT 2014, pp. 84-103, Springer International Publishing, 2014

[132] J. Hoffstein, J. Pipher, and J. Silverman. "NTRU: A ring-based public key cryptosystem." In Algorithmic number theory, pp. 267-288. Springer Berlin Heidelberg, 1998.

[133] IEEE Standard P1363.1, “IEEE standard specification for public key cryptographic techniques based on hard problems over lattices”, 2009.

[134] O. Oscar, S. Roy, F. Vercauteren, and I. Verbauwhede. "A masked ring-LWE implementation." In Cryptographic Hardware and Embedded Systems--CHES 2015, pp. 683-702. Springer Berlin Heidelberg, 2015.

[135] J. Silverman and W. Whyte. “Timing attacks on NTRUEncrypt via variation in the number of hash calls.” In Topics in Cryptology, CT-RSA 2007, volume 4377 of Springer LNCS, pages 208-224, 2006.

[136] N. Vizev. “Side Channel Attacks on NTRUEncrypt.” PhD thesis, Bachelors thesis, Technical University of Darmstadt, Germany, 2007.

[137] A. Atici, L. Batina, B. Gierlichs, and I. Verbauwhede. "Power analysis on NTRU implementations for RFIDs: First results." In Proceedings of RFIDSec 2008, 2008.

[138] M.-K. Lee, J. Song, D. Choi, and D.-G HAN, "Countermeasures against power analysis attacks for the NTRU public key cryptosystem." IEICE transactions on fundamentals of electronics, communications and computer sciences 93, no. 1 (2010): 153-163, 2010

[139] A. Wang, X. Zheng, and Z. Wang. “Power analysis attacks and countermeasures on NTRU-based wireless body area networks.” In KSII Transactions on Internet and Information Systems (TIIS), 7(5):1094-1107, 2013.

[140] X. Zheng, A. Wang, and W. Wei. “First-order collision attack on protected NTRU cryptosystem.” In Microprocessors and Microsystems, 37(67):601-609, 2013.

[141] A. Kamal and A. Youssef. “Fault analysis of NTRUEncrypt.” IEICE transactions on fundamentals of electronics, communications and computer sciences, E94-A(4), 1156-1158, 2011.

[142] A. Kamal and A. Youssef. “Fault analysis of the NTRUSign digital signature scheme.” Cryptography and Communications, 4(2):131{144, 2012.

[143] A. Kamal and A. Youssef. “Strengthening hardware implementations of NTRUEncrypt against fault analysis attacks.” Journal of Cryptographic Engineering, 3(4):227-240, 2013.

[144] A. Kamal and A. Youssef. "A scan-based side channel attack on the NTRUEncrypt cryptosystem." In Availability, Reliability and Security (ARES), 2012 Seventh International Conference on, pp. 402-409. IEEE, 2012.

[145] J. Bos, C. Costello, M. Naehrig, and Douglas Stebila, “Post-quantum key exchange for the TLS protocol from the ring learning with errors problem”, https://eprint.iacr.org/2014/599.pdf, Cryptology ePrint Archive, 2015.

[146] IEC/TS 61967-3 ed1.0 “Integrated circuits - Measurement of electromagnetic emissions, 150 KHz to 1 GHz - Part 3: Measurement of radiated emissions - Surface scan method”, 2006.

[147] A. Schlösser, D. Nedospasov, J. Krämer, S. Orlic, and J-P Seifert. "Simple photonic emission analysis of AES." In Cryptographic Hardware and Embedded Systems–CHES 2012, pp. 41-57. Springer Berlin Heidelberg, 2012.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 644729

End of Document