Side-Channel Attacks Detection Methods: A Survey

Joanna Assaeedi[1]† and Hatim Alsuwat[1]†,

[1] Department of Computer Science, College of Computer and Information Systems, Umm Al-Qura University, Saudi Arabia

Summary
Side-channel attacks are a quiet mighty type of attack that targets specific physical implementations vulnerabilities. Even though several researchers have examined diverse means and methods of detecting side-channel attacks, at the present time a systematic review of these approaches does not exist. The purposes of this paper are to give an extensive analysis of literature on side-channel attack detection and offer intuitiveness from past research studies. In this study, a literature survey is conducted on articles related to side-channel attack detection between 2020 and 2022 from ACM and IEEE digital libraries. From the 10 publications included in the study, it appears they target either a single type of side-channel attacks or multiple types of side-channel attacks. Therefore, a vital review of each of the two categories is provided, as well as possible prospective research in this field of study.
Keywords:
Side-channel attacks, Survey, Cryptanalysis, Detection of side-channel attacks.

1. Introduction

The significance of computer security is evident in our day-to-day lives. Currently, we live in a time where all sensitive and valuable information and data is passed and distributed through computer networks. To protect these sensitive pieces of information, a wide variety of cryptographic algorithms and protocols have been developed and implemented. Cryptography is defined as the art and science of generating a secret code or cipher by enciphering or deciphering a message [1]. In the past, Cryptographers have measured the strength of the cipher, which is a method to conceal the meaning of a message by transforming it [2], by the difficulty presented in breaking down the algorithm used to conceal the message. However, in modern times a new liability has emerged due to the usage of electronics.

Encryption systems are constructed to scramble data and hide them from prying eyes; however, the implementation of these systems is more complicated in practice. Researchers have revealed that some cryptographic devices are vulnerable to implementation-specific physical attacks known as Side-Channel Attacks [3].

Side-channel attacks primarily focus on extracting confidential intelligence from a computer system by examining the physical characteristics [3,4]. The analysis aims to use the relationship between the encryption process taking place on a device and the power consumption, processing time, electromagnetic emanation, or other phenomena [2].

Even though researchers have examined a number of methods and approaches that can detect side-channel attacks, there is an absence of a systematic literature review on detection techniques.

This paper intends to demonstrate a systematic review of studies on detection methods of side-channel attacks and exhibit insights from previous literature and their methodologies to provide an outlook on detection methods. The rest of the paper compromise of background information to assist in understanding the main aspects of the study in section 2, a presentation of the methodology that is applied for the literature collection in section 3, summarization of the literature found in the results in section 4, and a recap of the study and pointing out future research directions in section 5.

2. Background

Side-channel attacks are tightly linked to the physically observable phenomenon that occurs due to present time microelectronic executing computational tasks. For instance, microprocessors must expend time and power to complete their assigned tasks. As well as making some noise, depleting heat, and radiating electromagnetic fields [5]. Acquiring sensitive information using these side-channels is a branch of cryptography known as side-channel cryptanalysis [3]. Unlike other forms of cryptanalysis, it doesn’t attack the algorithms and the possible computational problems, rather it targets the inevitable leakage of information from any electronic device [6]. In short, side-channel attacks cryptanalysis attack a physical implementation and not the mathematical foundation of an algorithm [3].

Side-channel attacks that use exact knowledge of the device and the algorithm implementation, as well as architecture and their effect on a few observed measurement samples, are known as simple power analysis side-channel. While differential power side-channel attacks use many observations and precise insight into the architecture along

Manuscript received June 5, 2022
Manuscript revised June 20, 2022
https://doi.org/10.22937/IJCSNS.2022.22.6.38

with the algorithm and device implementation and their effect on the observed measurement samples [7].
Timing Attacks, Optical Attacks, Electromagnetic attacks, Power Analysis Attacks, Fault Induction Attacks, and Traffic Analysis are the various possible side-channel attacks [3], with electromagnetic side-channel attacks, power analysis side-channel attacks, and timing side-channel attacks being the most prominent in the literature.

2.1 Timing attacks

One of the earliest articles regarding side-channel attacks was published by Kocher [8] and it presented a way to retrieve an RSA private key solely by seeing how much time it takes the device to decipher a message. The reason it worked was due to RSA and alternative public-key cryptosystems working with big figures such as 2,048 bits while CPUs have a lower word capacity. To deal with the difference in size, crypto implementations used multi-precision arithmetic, which is adapting a collection of words and a loop in order to move from one word to the latter to represent a large number [9].
Cryptosystems such as RSA regularly used an optimization known as square-and-multiply whenever a multi-precision number had to be raised to an exponent. The optimization works by decomposing exponentiation into a chain of squiring, x^2, and conditional multiplication that occurs if the bit being processed is one. This is akin to traditional multiplication done on paper, in which the result is shifted a single decimal position to the left whenever there are trailing zeros, while the nonzero digits multiplication is combined with the results.
Since the multiplication phase is optional, the assailant can deduce the total number of one bit by measuring the time taken for decryption. By using various input messages and computing the total time taken by the device to perform multi-precision exponentiation, the attacker can ultimately uncover the whole private key or a sufficient amount of it to brute force the remainder of it in a timely manner.
Timing attacks have constantly improved over the decade, going as far as being performed on a Secure Sockets Layer (SSL) over a network [9].
To summarize, numerous sources of timing leakages are due to careless implementations [10], and attackers analyze the time taken by the cryptographic device to process a set of messages to recover some secrete parameters [11].

2.2 Power Analysis Side-Channel

Nowadays most encryptions depend on electronic machinery manipulation of ones and zeros. This is done by either administering or cutting electric power to devices known as transistors, to either hold a value or execute an operation on the value [2]. By applying or removing a current to or from each transistor, a zero is changed to a one or vice versa.
The measurement of electronic device power consumption contains information about the circuit's calculation due to the relationship between the amount of power used and the processed data, even a single transistor operation appears as a weak correlation in power consumption measurement as it was first studied by Kocher et al. [12]. It was discovered that an attacker manages to uncover secret knowledge by observing the data-dependent power usage of a device processing cryptographic secret [2].

2.2.1 Simple Power Analysis Side-Channel

Simple Power Analysis (SPA) is the most straightforward side-channel power analysis attack, which attempts to directly interpret the power consumption as cryptographic operations are performed to infer information about the procedure [2,13].
A trace is known as a set of power measurements taken throughout interesting time often being a full cryptographic operation. SPA makes use of visual inspection of the leakage traces to identify power fluctuations that might disclose cryptographic operations [13]. For example, in a case where the square and multiplication operations are implemented differently from each other, as this will allow for faster code for the square operation, the visual traces and consumption patterns for the exponentiation operation will look quite different to the point where the secrete exponent's value can be directly realized. In most cases, any programs that contain conditional processes that depend on a secrete parameter are at risk [13].

2.2.2 Differential Power Analysis Side-Channel

The visual inspections of SPA are intriguing; however, it is quite difficult to automate them not to mention they are subjected to different interpretations. In addition, it is rather challenging to directly observe and distinguish the information about the secret key from within the traces.
In the Differential Power Analysis (DPA), the adversary adapts mathematical and statistical principles to figure out the secret information within the power consumption patterns. In 1999, Kocher et al. [12] developed the idea of model-based side-channel attack in a paper considered to have built the foundation of research in the side-channel attacks field. By creating a selection function, traces are put into two different classes or sets of data. First, it must be determined if the two classes are statistically different from one another, to do that a static is chosen and used to compare the two classes. In the second step of classical DPA, the master trace is determined for each class by using the mean or the first moment to reduce all traces in each class. Finally, the class master traces are measured against every point in the trace, to decide whether these points are substantially different from each other [2].

2.3 Electromagnetic Attack

Electromagnetic emission is the fundamental science for various wireless communications [14]. It is a well-known reality that as an aftereffect of electronic machines internal activities, Electromagnetic radiation is generated on unintentional frequencies [15]. In the USA these unintended Electromagnetic emissions are standardized by government departments like the 'Federal Communications Commission' (FCC) due to the possibility of health issues that might affect the users of the device, as well as the probable intrusion they might make on valid wireless communication, however, complete avoidance of such radiation is impossible [14].
Electromagnetic side-channel attacks have revealed promising outcomes, as they involve the lowest amount of physical manipulation of the inspected machine [14]. The electromagnetic radiation of an electric machine can be passively observed to derive both the processed data and the internal processes [16]. Electromagnetic ambushes have been proved to have higher effectiveness than power analysis attacks on some cards [16]. And they can be performed alongside power analysis attacks to lower the possibilities of failure and raise the overall precision of the disclosed information [14].

3. Literature Collection

To conduct a valuable and methodical literature review, the guidelines known as Preferred Reporting Items for 'Systematic Reviews and Meta-Analysis' (PRISMA) [17] are adopted. A four-step sequential process is utilized that consists of first eligibility criteria, second information sources and searches, third paper selection, and fourth the results.

3.1 Eligibility Criteria

In this study, electronic databases known to publish peer-reviewed journals and conferences proceedings are elected to choose the final two databases from them. The two databases selected are IEEE Xplore and ACM. The elected papers must be related to side-channel attacks detection, and they must include the methods or mechanisms used for the detection.

3.2 Information Sources and Searches

Both IEEE Xplore and ACM provide an advanced search function. In IEEE Xplore digital library the advanced search allows keyword searching in various areas such as all metadata, full text, publication title, and abstract as well as data range for publication year. As for ACM digital library, the advance search enables searching for a keyword within different areas such as the title, publication title, full text, abstract, and a custom time scope. Both IEEE Xplore and ACM implement an area for command search or query search, which supports more control by typing the search query using specific keywords and expressions.
The search aims for articles published between 2020 and 2022 in the English language. The selected keywords for the initial search are "side-channel attack", "detection", and "method". And the title and abstract are used as filters for the first stage of the systematic review.

3.3 Paper Selection

To select the relevant papers, a three-step procedure is applied. In the first step, papers that have the keywords 'side channel' with either the keyword 'detection' or 'method' in the title are selected. Then in the second step, the selected papers will be filtered furthermore by reading each paper's abstract and ascertaining whether they are related to the search research topic or not. Finally, in the last step, the remaining articles' full text is skimmed to determine their relevance to the research subject.
Table 1 presents the outcome of each step used to identify the suitable publication from the two digital databases between 2020 and 2022. After the first step, filtering by the specified keywords in the title, a total of 31 papers were identified. Afterward, the abstracts of the 31 articles were read to ascertain their relevance, and a total of 19 papers were found to be related to the research topic. Finally, after skimming the full text of the documents, out of the 19 papers 10 were found to not duplication and connected to the topic.
For each of the qualified publications, a documentation of their publication year, the digital library, the type of side-channel the method is detecting, the name of the method, whether the method uses machine learning or not, and the evaluation method used to test the performance of the method is presented in table 2.

Table 1: Outcome of the steps adopted in the literature review

Sources	***Filtered by Keywords in Title***	***Filtered by Abstract***	***Filtered by Full Text***
ACM	14	9	6
IEEE	17	10	4
Total	31	19	10

4. Results

The eligible papers are analyzed based on the year of publication, the type of side-channel attack they deal with, and whether they implement machine learning methods or not.

4.1 Year of Publication

Table 2: Outline of the 11 qualified papers

Ref.	*Digital Library*	*Year*	*Side-channel Type*	*Method*	*Machine Learning*	*Evaluation Method*
[18]	IEEE	2020	Timing	Lurking Eyes	×	F-measure
[19]	IEEE	2020	Most of them	kurtosis-based consistency check	×	Compared to TVLA
[20]	IEEE	2020	Most of them	Phased-Guard	✓	DTW, F-measure
[21]	IEEE	2021	Most of them	Machine Learning Framework	✓	Simulation
[22]	ACM	2020	Power	RO-based mechanism	×	Real-time detection
[23]	ACM	2020	Timing	Automatic Framework	×	Dataset
[24]	ACM	2021	Padding Oracles	Automatic Framework	✓	Testing set, Real-time detection
[25]	ACM	2021	Most of them	generalized detection mechanism	✓	KL Divergence values, Random Forest Classifier
[26]	ACM	2020	cache access pattern	Hybrid-Shield	✓	Detection accuracy, false alarm rate
[27]	ACM	2020	Timing	SCFMSP	×	Simulation against benchmarks

Figure 1 shows the distribution of the 10 eligible articles based on the year of publication. The figure shows that seven papers were published about side-channel in 2020 in both digital libraries. While only 3 papers were published on the topic in both libraries in the year 2021 and zero

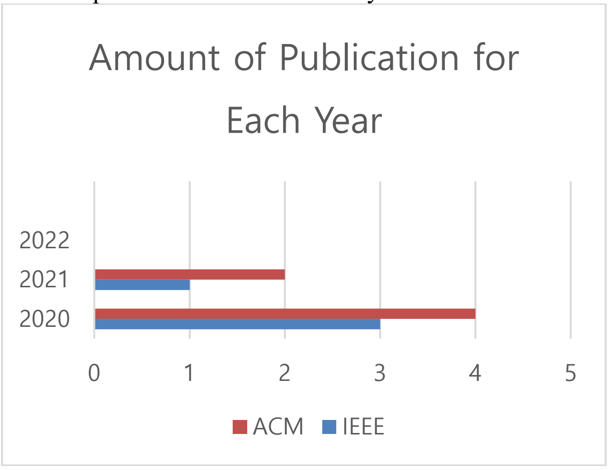

Fig. 1 Distribution of publication from 2020 to 2022.

articles have been published to date in the year 2022. This unfortunately shows a decline in the number of publications related to side-channel detection.

4.2 Side-Channel Type

At present, there is no obvious classification in the literature for side-channel detection methods. From the literature review, it appears that all literature either focus on dealing with one side-channel attack or attempt to find a general method for multiple side-channel attacks. Figure 2 demonstrates the distribution of the different side-channel attacks that the eligible papers detect from 2020 to 2022. Therefore, the detection methods will be classified into one of two categories. The categories are single side-channel attacks and multiple side-channel attacks.
The single side-channel attack category can be further divided into timing, electromagnetics, power, and cache access pattern. While the multiple side-channel attacks category is further classified into using machine learning or not using machine learning.

4.2.1 Single Side-Channel Attacks

The single side-channel attack category consists of multiple subcategories. This paper will focus on four of them since these four categories are present in the reviewed publications. The four categories are timing, power, cache access pattern, and padding oracles.

4.2.1.1 Timing Side-Channel Attacks

From the eligible papers, the authors of [18],[23], and [27] methods detect timing side-channel attacks. Timing attacks are accomplished by examining the time a cryptographic

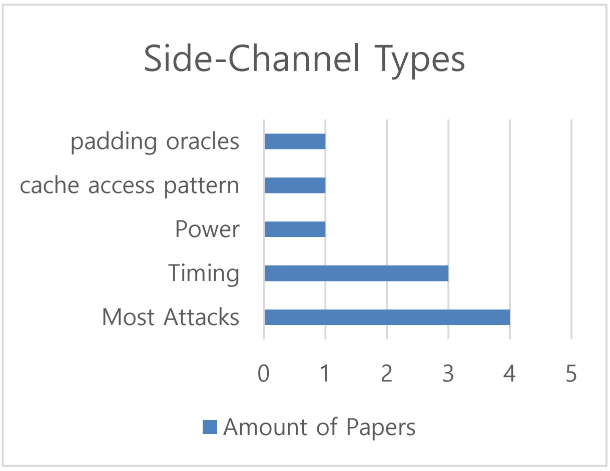

Fig. 2 Classifications of different detected side-channel attacks.

device takes to handle a set of messages to retrieve some secret parameters.

First, Mazaheri et al. [18] detection method named Lurking Eyes is designed to detect timing attacks targeting JavaScript and WebAssembly within different web pages. The main advantage this method has over the past methods is abolishing the necessity of disabling some useful JavaScript features. The JavaScript features that might have needed to be disabled are sensor API, multithreading, shared data, accurate timing, and memory addresses. The way the Lurking Eyes method work is by first inspecting the web page HTML code. Afterward, the web page JavaScript code will be obtained. Then, hunting for the features that may signify a timing side-channel presence. Lastly, alert the user of the findings using one of the seven designated alerts. The features and functions that may indicate an attack are obtained through assessing and replicating the attacks in different conditions. The authors evaluated Lurking Eyes by testing the top 3350 pages in Alexa and Majestic. Then, calculate the value of accuracy, error rate, specificity, precision, and recall. Finally, the F-measure was calculated, and it was found to be 0.983. However, an attacker might be able to deceive the method in different ways such as using lowercase and uppercase letters. Therefore, the authors have tried to identify as many various methods as possible and provide a solution for them. Second, Brennan et al. [23] automatic detection method is intended to detect timing attacks that emerge in certain optimizations presented through just-in-time (JIT) compilation in Java programs. These JIT-induced side-channels are caused by attempting to optimize paths. Whenever a program is called repeatedly the JIT compiler will generate optimized native code in order to speed up the execution of the most commonly executed path. Even though this process would speed up the execution, any calls to the program using unknown input can leak data about the path the input went through. The automatic detection method works by going through two main steps. The initial one is known as the input generation phase, and the second one is known as Java Virtual Machine (JVM) fuzzing. For the input generation phase, it first generates pairs of partition cells by analyzing six different programs behavior patterns and recognizing which input partitions are prone to the side-channel attack. Then the input generation phase generates a set of prime input for the JVM to favor specific program paths to initiate the timing side-channel attack. As for the second phase, JVM fuzzing uses the two results from the first phase to perceive if a timing side-channel attack occurs or not. The authors evaluated the automatic detection method by testing it on three datasets known as Blazer, Themis, and DiffFuzz containing benchmark and unsafe programs data. The automatic detection method was able to detect JIT-induced timing side-channel attacks in previously studied and labeled safe benchmarks.

Pouyanrad et al. [27] detection method named SCF^{MSP} is created to detect timing attacks aimed at the MSP430 assembly program. The MSP430 architecture is a 16-bit microcontroller composed of embedded devices with low power consumption. The timing attack might occur in the MSP430 architecture due to the time to execute the instruction on the MSP430 microcontroller being deterministic. The SCF^{MSP} tool consists of three components. The first component, the container initializer, receives a list of starting function arguments as an input and it assigns each memory and registers a security level. Then the container initializer links the argument and the matching register according to the calling conventions of MSP430. As for the second component, the syntax converter takes the inputted ELF files and converts them into a form close to a syntax defined by the authors. Then the third component, control flow analysis, uses the output of the syntax converter as an input and uses it to build a definite MSP430 assembly program control flow graph. The control flow graph aims to achieve two goals by extracting all execution point successors and predecessors. The two goals are, limiting the information flow of the following execution points, and capturing later points in need of analysis. For the fourth and final component, SC verify, it collects the output from the container initializer component, and the control flow component and wields them to administer a sequence of static analyses to recognize side-channel leaks. Whenever the start-to-end secret dependent if/else running times are different from non-secret if/else then the SCF^{MSP} will report it as a timing leaks vulnerability. On the other hand, if at least one instruction in if/else takes a distinct execution time from the rest of the program then the SCF^{MSP} will report a Nemesis vulnerability. Lastly, if a secret input results in hidden information being observed in the output, then the SCF^{MSP} will report it as an Information Leakage.

The authors evaluated the SCF^{MSP} tool by testing it against hand-crafted benchmarks and it was able to find a variety of side-channel vulnerabilities.

4.2.1.2 Power Side-Channel

From the eligible papers, the authors of the [22] method detect power side-channels. The power side-channel attack is accomplished by gauging electronic device power consumption.

Gattu et al. [22] detection method is constructed to detect power side-channel attacks. The primary contribution of this method is the low overhead of the detection due to not using machine learning in the process. This Ring Oscillator (RO) based method works by comparing the voltage of the victim node with a neighboring attack-free node. The RO sensor is chosen due to its considerable sensitivity to millivoltage level of variation, the sensor's low consumption overhead, its speed of detection, and not being disruptive to the general functionality of the underlying chip. The RO and various counters serve to sample the N_{rising} from every power node. When all the N_{rising} that is neighboring RO are compared and the N_{rising} is calculated to be high enough, then the chip will be considered under attack. N_{rising} which is equal to the clean RO deducted by RO under a side-channel attack is the detection metric in this method. The authors evaluated this approach by comparing the purposed technique to a machine learning-based technique and it was found to be faster with a speed of 2 µs and no test or storage overhead. However, the purposed approach does face some limitations which are its effectiveness for detecting electromagnetic side-channel attacks and the possibility of errors due to noise present in the sampling clock.

4.2.1.3 Cache Access Pattern Side-Channel

From the eligible papers, the authors of the [27] method detect cache access pattern side-channels. The cache access pattern side-channel attacks are accomplished by exploiting and observing the cache access patterns [28][29].

Wang et al. [27] detection method named Hybrid-Shield is designed to detect cache access pattern side-channel attacks targeting Microarchitecture. The main advantage of this method is having a lower false alarm rate. The way Hybrid-Shield work is by first collecting the data, extracting the features, training, testing the machine learning model, and lastly minimizing the false alarm rate. The data collection step collects the Hardware Performance Counter (HPC) data that are under attack and not under attack to build the final dataset. Then for feature vector extraction, the raw data is received from the data collection step and the Greedy Forward Selection algorithm is applied to determine the most prominent features. Then, the Correlation Attribute Evaluation is employed due to the limited number of HPCs available for collection simultaneously. In machine, learning classifiers step numerous types of machine learning classifiers, OneR, MLP, DT, J48, and BayesNet algorithms are implemented as the classification models from separate fields of machine learning. Finally, to minimize the rate of false alarms since the side-channel attacks are partial to the under-attack category. The false alarm minimization approach is to delay the decision of reporting under attack until a certain amount of continuous intervals occurs. The authors evaluated Hybrid-Shield by calculating the detection accuracy and the rate of the false alarms. The Hybrid-Shield approach and traditional classifiers attain about 80% detection accuracy while utilizing few HPCs. As for the false alarm rate, Hybrid-Shield and the false alarm rate plunge from the 87% rate of traditional classifiers to a 4.7% rate.

4.2.1.4 Padding Oracles Side-Channel Attacks

From the eligible papers, the authors of the [24] method detect padding oracles side-channel attacks. The padding oracles attacks are accomplished by querying a padding oracle on whether a ciphertext has valid padding or not [30].

Drees et al. [24] automatic detection method is developed to detect padding oracles side-channel attacks targeting cryptographic protocols. The automatic detection method works by executing four sequential stages. In the first stage, manipulated TLS client, a manipulated client establishes a connection with a TLS server to test it. Then, the manipulated client chooses at random an alteration to administer to the ciphertext. Afterward, it uses the altered ciphertext to execute a handshake with the TLS server. And this process is executed a pre-configured number of times while a network tap logs all the messages traded amidst the manipulated client and the TLS server with the help of tcpdump. Then in the second stage, feature extraction, the raw data from stage one is transformed into constant-sized training features. The transformation is done by converting all the protocol fields to floating-point and integers values and assigning any lost values with negative one (-1). Afterward, in the third stage, Classification Model Learning, the dataset from stage 2 is employed to train the machine learning model to classify ciphertext padding into valid or invalid based on the server rection. However, a single binary classification proved ineffective, therefore the method uses a group of binary classifiers to differentiate between authentic and inauthentic padding. Finally in stage 4, report generation, a vulnerability verdict is issued by comparing the t-test score of the approach with simple Random Guessing (RG) algorithm. Also, a final report contains details information concerning any detected side-channel as feedback to software developers. The authors evaluated the automatic approach by applying the methodology to various TLS servers' implementations.

First, the method is tested against an insecure TLS server and a secure TLS server. The test confirmed the approach's ability to detect side-channels in the insecure implementation and confirm the security of the secure implementation. Afterward, the method was applied to real-world side-channels such as OpenSSL Version 0.9.7a and ROBOT. Then finally it was tested on open-source implementation that doesn't show any unidentified vulnerabilities.

4.2.2 Multiple Side-Channel Attack

The category of multiple side-channel attacks can encompass two subcategories that are present in the reviewed publications. The two categories are detection methods that support machine learning, and detection methods that do not use machine learning.

4.2.2.1 Machine Learning Methods

From the qualified papers, the authors of [20],[21], and [25] methods detect multiple forms of side-channel attacks. The approaches the authors used employed the use of machine learning.

First, Wang et al. [20] detection method known as Phased-Guard is designed to detect and identify various zero-day microarchitecture side-channel attacks. The major contribution of this method is detecting unknown zero-day attacks, as well as identifying the type of vulnerability. The idea behind how Phased-Guard work is by going through a three-phase framework. The framework consists of data collection and feature representation, side-channel attack detections, and side-channels identification. For the data collection process, 16 Hardware Performance Counters (HPC) were collected from 4590 records of the under no attack sample, and the 10,000 records of the under three different side-channel attacks sample. Afterward, to prepare the training dataset the similarity metric is calculated between the HPC's sequences using the dynamic time warping (DTW) for all the 16 HPC features. And due to the limitation of the ability to collect HPC registers simultaneously in modern microprocessors correlation attribute evaluation is applied to identify the most significant HPC. As for the side-channel attacks detection phase, binary classification is applied to detect no attack or attacked conditions. For this reason, four different machine learning classifiers, OneR, MLP, J48, and BayesNet, were used. Finally, for the type identification phase, the BayesNet algorithm is used to classify the data into one of three predetermined classes after phase one reports an under attack condition. The authors evaluated Phased-Guard by calculating the detection accuracy using three different testing and training cases. The results show above 99% accuracy for both Phased-Guard and established machine learning detection mechanism when testing the two categories, the under no attack victims and known attacks victims. While for the victims under unknown attack, the common HPC-based detectors fall to lower than 20% to 15% while Phased-Guard maintains the 99% accuracy.

Second, Lescisin et al. [22] detection method is a layered group of functional components that provide various side-channel test scenarios. The main lead this method over past methods is its capability of collecting side-channel data from many sources, as well as recovering leaked information by merging various side-channel transmitter sources. The layered architecture consists of five different layers that must be followed in order of data gathering layer, then the feature extraction layer, then the machine learning layer, then the threat modeling layer, and lastly the reactive layer. As for the data collection layer, it contains four different tools that are essential for accumulating raw data that have to include perceptible traits and affiliated private information. The data gathering tools are traffic tagger for tunneled VNC, instrumentable testbench virtual machine, bash shell UDP tagger, and Firefox addon UDP tagger. Then the feature extraction layer uses four modular components to generate feature vectors by performing preparatory data processing on the data collected in the previous layer. The four modular components in this layer are, the time density filter, UDP labeled sequence extractor, NBurst filter, and SSH labeled sequence extractor. Afterward is the machine learning layer which consists of three functional components that use the extracted features from the previous layer as well as the private event label to generate a probabilistic model which predicts every scenario in the encapsulated event space. The three functional components are decision tree builder, balanced label data splitter, and entity histogram creator. Then the threat modeling layer uses four functional components that use the machine learning model from the previous layer to conclude whether a side-channel leakage exists or not by checking whether it breaches the security model set by the system. The four functional components are string entropy calculator, SSH command prediction evaluator, VNC keypress prediction evaluator, and HTTPS page load prediction evaluator. Finally, the reactive tier uses five functional components compares the results of the threat modeling layer and the system's security model and provides the next step to mitigate the information leak. The five functional components are report webpage generator, bar graph renderer, polar gauge renderer, entropy target renderer, and warning logger. The authors evaluated their layered test scenario framework by deploying four different test scenarios, and the framework proved to be successful on all four assessments.

Lastly, Alam et al. [25] generalized detection method is designed to uncover distinct micro-architectural side-channel attacks. The main benefit of this mechanism is being a generalized mechanism, as well as removing any false positives from the discovered results. The way this approach work is by implementing a two-phase, offline phase and online phase, methodology. First is the offline stage that consists of two modules, the data collection module, and the classification module. In the data collection module of the offline phase, the data is collected from a system executing an encryption algorithm. When collecting the data, the module considers four different restricted execution environments. The four different environments are regular behavior, cache intensive process in the background, branch intensive programs in the background, and programs influencing the RAM in the background. It is important to note that to keep the method generalized, the data collected is connected to the performance counter events of applications developing anomalous observations. Then the classification module takes the data from the previous module and uses it to build a classifier that knows which performance counter events relate to the diverse classes of anomalies. The second is the online phase which consists of three modules, the data collection module, classification module, and correlation Module. The data collection module in the online stage works similarly to the one in the offline step. The classification module uses the data collected from the previous module and uses it in order to identify the character of the HPC values and classifies them as irregular behavior or natural behavior, as well as to distinguish the brand of anomaly. Then the correlation module is implemented to abolish the false positives by computing the correlation value between the trace from the trace dataset that relates to the category from the classification module and the trace found by observing the anomalous behavior. The authors evaluated their method by calculating the detection accuracy of the method. It shows the proposed method has the second-highest accuracy of 98.7% and the lowest false positive percentage of 0.11% when matched against the state-of-the-art approaches.

4.2.2.2 Non-Machine Learning Method

From the eligible papers, the authors of [19] methods were able to detect multiple side-channel attacks. The method employed does not use machine learning for the detection process.

Yang et al. [19] detection method is designed to detect different s detection methods as complicated calculations, leakage models, and the requirement restrictions on inputs. The approach works by collecting side-channel traces. Side-channel traces consist of leakage points, which contain key-dependent information, and non-leakage points, which do not include any key-dependent information. Then the non-leakage mark is considered variable parameters channel in communication channels since non-leakage data can be perceived as arbitrary noise. While the leakage point is regarded as constant parameters channel in the communication channel. Then using specific values and equations kurtosis value is computed for every point of traces. Afterward, the 'generalized extreme studentized deviate' (GESD) evaluation is applied to a set of kurtosis values to establish the uppermost and the lowermost constrains. Finally, the point of leakage is considered to be any kurtosis value higher than the upper bound. The authors evaluated their method by collecting 30,000 power remnants, extracted from a hardware implementation of an AES-128, and an additional 5,000 power particles from a software implementation of an AES-128. The method is then implemented on the collected power traces and the results are compared to a Test Vector Leakage Assessment (TVLA) detection result. The author's approach proves to outperform the TVLA approach in leakage traces discovery in single-channel and multi-channel leakage exposure.

5. Conclusions and Future Work

This paper examines the current side-channel attacks detection method found in the literature. The research also introduces a new taxonomy for side-channel attacks detection methods based on the type of attack being detected. The classification categorizes side-channel attacks methods into two main categories, single side-channel attacks detection, and multiple side-channel attacks detection. Single side-channel detections consist of methods designed to detect a single type of side-channel attack. This category is then classified into timing attacks detection, power attacks detection, cache access pattern attacks detection, and padding oracles attacks detection. While multiple side-channel attacks detection consists of methods devised to detect multiple types of side-channel attacks. This category is further divided into methods that either employ machine learning for the detection or it does not. Future studies could cover a larger set of digital libraries and more research years.

References

[1] Merriam-Webster. (n.d.). Cryptography definition & meaning. Retrieved March 12, 2022, from https://www.merriam-webster.com/dictionary/cryptography.

[2] Randolph, M., & Diehl, W. (2020). Power side-channel attack analysis: A review of 20 years of study for the layman. *Cryptography, 4* (2), 15.

[3] Joy Persial, G., Prabhu, M., & Shanmugalakshmi, R. (2011). Side channel attack-survey. *Int J Adva Sci Res Rev, 1*(4), 54–57.

[4] Wang, H., Sayadi, H., Rafatirad, S., Sasan, A., & Homayoun, H. (2020). Scarf: Detecting side-channel attacks at real-time using low-level hardware features. In *2020 ieee 26th international symposium on on-line testing and robust system design (iolts)* (pp. 1–6).

[5] Verbauwhede, I. M. (2010). *Secure integrated circuits and systems*. Springer.

[6] Okeya, K., & Sakurai, K. (2003). A multiple power analysis breaks the advanced version of the randomized addition-subtraction chains countermeasure against side channel attacks. In *Proceedings 2003 ieee information theory workshop (cat. no. 03ex674)* (pp. 175–178).

[7] Rahaman, M. Z., & Hossain, M. A. (2008). Side channel attack prevention for aes smart card. In *2008 11th international conference on computer and information technology* (pp. 376–380).

[8] Kocher, P. C. (1996). Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In *Annual international cryptology conference* (pp. 104–113).

[9] Lawson, N. (2009). Side-channel attacks on cryptographic software. *IEEE Security & Privacy,* 7(6), 65–68.

[10] Fan, J., Guo, X., De Mulder, E., Schaumont, P., Preneel, B., & Verbauwhede, I. (2010). State-of-the-art of secure ecc implementations: a survey on known side-channel attacks and countermeasures. In *2010 ieee international symposium on hardware-oriented security and trust (host)* (pp. 76–87).

[11] Quisquater, J.-J., & Koene, F. (2009). Side channel attacks: State of the art, october 2002. *Available from http://www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/1047 Side Channel report. pdf.*

[12] Kocher, P., Jaffe, J., & Jun, B. (1999). Differential power analysis. In *Annual international cryptology conference* (pp. 388–397).

[13] Sayakkara, A., Le-Khac, N.-A., & Scanlon, M. (2019). A survey of electromagnetic side-channel attacks and discussion on their case-progressing potential for digital forensics. *Digital Investigation, 29*, 43–54.

[14] Getz, R., & Moeckel, B. (1996). Understanding and eliminating emi in microcontroller applications. *National Semiconductor*.

[15] Sayakkara, A., Le-Khac, N.-A., & Scanlon, M. (2018). Electromagnetic side-channel attacks: potential for progressing hindered digital forensic analysis. In *Companion proceedings for the issta/ecoop 2018 workshops* (pp.138–143).

[16] Pongaliur, K., Abraham, Z., Liu, A. X., Xiao, L., & Kempel, L. (2008). Securing sensor nodes against side channel attacks. In *2008 11th ieee high assurance systems engineering symposium* (pp. 353–361).

[17] Moher, D., Liberati, A., Tetzlaff, J., Altman, D. G., & Group*, P. (2009). Preferred reporting items for systematic reviews and meta-analyses: the prisma statement. *Annals of internal medicine, 151*(4), 264–269.

[18] Mazaheri, M. E., Taheri, F., & Sarmadi, S. B. (2020). Lurking eyes: A method to detect side-channel attacks on javascript and webassembly. In *2020 17th international isc conference on information security and cryptology (iscisc)* (p. 1-6). doi: 10.1109/ISCISC51277.2020.9261920.

[19] Yang, W., Zhang, H., Gao, Y., Fu, A., & Wei, S. (2020). Side-channel leakage detection based on constant parameter channel model. In *2020 ieee 38th international conference on computer design (iccd)* (p. 553-560). doi: 10.1109/ICCD50377.2020.0009.

[20] Wang, H., Sayadi, H., Kolhe, G., Sasan, A., Rafatirad, S., & Homayoun, H. (2020). Phased-guard: Multi-phase machine learning framework for detection and identification of zero-day microarchitectural side-channel attacks. In *2020 ieee 38th international conference on computer design (iccd)* (p. 648-655). doi: 10.1109/ICCD50377.2020.00111.

[21] Lescisin, M., & Mahmoud, Q. H. (2021). A machine learning based monitoring framework for side-channel information leaks. *IEEE Open Journal of the Computer Society, 2*, 139-151. doi: 10.1109/OJCS.2021.3061445.

[22] Gattu, N., Khan, M. N. I., De, A., & Ghosh, S. (2020). Power side channel attack analysis and detection. In *Proceedings of the 39th international conference on computer-aided design*. New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3400302.3415692 doi: 10.1145/3400302.3415692.

[23] Brennan, T., Saha, S., & Bultan, T. (2020). Jvm fuzzing for jit-induced side-channel detection. In *Proceedings of the acm/ieee 42nd international conference on software engineering* (p. 1011–1023). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3377811.3380432 doi:10.1145/3377811.3380432.

[24] a Drees, J. P., Gupta, P., Hullermeier, E., Jager, T., Konze, A., Priesterjahn, C., . . . Somorovsky, J. (2021). Automated detection of side channels in cryptographic protocols: Drown the robots! In *Proceedings of the 14th acm workshop on artificial intelligence and security* (p. 169–180). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3474369.3486868 doi:10.1145/3474369.3486868.

[25] Alam, M., Bhattacharya, S., & Mukhopadhyay, D. (2021, jan). Victims can be saviors: A machine learning–based detection for micro-architectural side-channel attacks. *J. Emerg. Technol. Comput. Syst., 17* (2). Retrieved from https://doi.org/10.1145/3439189 doi:10.1145/3439.

[26] Wang, H., Sayadi, H., Sasan, A., Rafatirad, S., & Homayoun, H. (2020). Hybrid-shield: Accurate and efficient cross-layer countermeasure for run-time detection and mitigation of cache-based side-channel attacks. In *Proceedings of the 39th international conference on computer-aided design*. New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3400302.3418783 doi:10.1145/3400302.3418783.

[27] Pouyanrad, S., Muhlberg, J. T., & Joosen, W. (2020). Scf^{msp}: Static detection of side channels in msp430 programs. In *Proceedings of the 15th international conference on availability, reliability and security*. New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3407023.3407050 doi:10.1145/3407023.3407050

[28] Sayadi, H., Wang, H., Miari, T., Makrani, H. M., Aliasgari, M., Rafatirad, S., & Homayoun, H. (2020). Recent advancements in microarchitectural security: Review of machine learning countermeasures. In *2020 ieee 63rd international midwest symposium on circuits and systems (mwscas)* (pp.949–952).

[29] Wang, H., Sayadi, H., Mohsenin, T., Zhao, L., Sasan, A., Rafatirad, S., & Homayoun, H. (2020). Mitigating cache-based side-channel attacks through randomization: A comprehensive system and architecture level analysis. In *2020 design, automation & test in europe conference & exhibition (date)* (pp. 1414–1419).

[30] Bleichenbacher, D. (1998). Chosen ciphertext attacks against protocols based on the rsa encryption standard pkcs# 1. In *Annual international cryptology conference* (pp. 1–12).