Contents lists available at ScienceDirect

Forensic Science International: Digital Investigation

journal homepage: www.elsevier.com/locate/fsidi

DFRWS 2020 EU — Proceedings of the Seventh Annual DFRWS Europe

Cutting Through the Emissions: Feature Selection from Electromagnetic Side-Channel Data for Activity Detection

Asanka Sayakkara [a, *], Luis Miralles-Pechuán [b], Nhien-An Le-Khac [a], Mark Scanlon [a, b]

[a] Forensics and Security Research Group, University College Dublin, Ireland
[b] Centre for Applied Data Analytics Research (CeADAR), University College Dublin, Dublin, Ireland

ARTICLE INFO

Article history:

Keywords:
Digital forensics
Electromagnetic side-channels
Feature selection
Internet-of-things (IoT)
Machine learning

ABSTRACT

Electromagnetic side-channel analysis (EM-SCA) has been used as a window to eavesdrop on computing devices for information security purposes. It has recently been proposed to use as a digital evidence acquisition method in forensic investigation scenarios as well. The massive amount of data produced by EM signal acquisition devices makes it difficult to process in real-time making on-site EM-SCA infeasible. Uncertainty surrounds the precise information leaking frequency channel demanding the acquisition of signals over a wide bandwidth. As a consequence, investigators are left with a large number of potential frequency channels to be inspected; with many not containing any useful information leakages. The identification of a small subset of frequency channels that leak a sufficient amount of information can significantly boost the performance enabling real-time analysis. This work presents a systematic methodology to identify information leaking frequency channels from high dimensional EM data with the help of multiple filtering techniques and machine learning algorithms. The evaluations show that it is possible to narrow down the number of frequency channels from over 20, 000 to less than a hundred (81 channels). The experiments presented show an accuracy of 0.9315 when all the 20, 000 channels are used, an accuracy of 0.9395 with the highest 500 channels after calculating the variance between the average value of each class, and an accuracy of 0.9047 when the best 81 channels according to Recursive Feature Elimination are considered.

1. Introduction

Electronic devices, including computers, are known to inadvertently produce noisy electromagnetic (EM) radiation when operating (Sayakkara et al., 2018a). With the emergence of Internet of Things (IoT), the number of EM noise sources in both indoor and outdoor environments are rapidly increasing. It has been shown that EM radiation from processors of computing devices tend to leak information that can be captured from a reasonable distance. EM side-channel analysis (EM-SCA) is the study of eavesdropping on such devices using these byproduct emissions. The behaviour of software running on the processor, such as specific machine instructions being executed and associated variables being manipulated by the software, have been shown to be revealed through EM-SCA techniques (Nazari et al., 2017; Sayakkara et al., 2018b). Both the information security and digital forensics domains have useful applications of EM-SCA, including the potential ability to overcome one of the biggest challenges facing digital forensics; encryption (Lillis et al., 2016).

The application of EM-SCA requires EM observations made close to a specific device, referred to as *device under test* (DUT). An EM signal observed for a particular period of time and saved into a file is called an EM trace. Typically, in most practical EM-SCA scenarios, a large number of EM traces are saved first and later analysed to extract embedded EM information leaking signal. However, there exist special cases where EM signals are needed to be analysed on-the-fly as they are being observed. For example, when an IoT device found in a crime scene is being analysed for potential clues of its involvement in a case, it is often necessary to complete the analysis as soon as possible (Sayakkara et al., 2019a). Remote wiping of the data stored on the device or any other malicious activity can harm the analysis of the device — potentially sabotaging the EM-SCA. Therefore, EM-SCA techniques should be ideally applied to the EM data as they are being collected.

Processing EM data in real-time poses a challenge for several reasons. EM signals are typically observed through software-defined radio (SDR) hardware or oscilloscopes, both featuring large sampling rates. As a result, EM trace files with sizes of several

* Corresponding author.
E-mail addresses: asanka.sayakkara@ucdconnect.ie (A. Sayakkara), luis.miralles@ucd.ie (L. Miralles-Pechuán), an.lekhac@ucd.ie (N.-A. Le-Khac), mark.scanlon@ucd.ie (M. Scanlon).

https://doi.org/10.1016/j.fsidi.2020.300927
2666-2817/© 2020 Elsevier Ltd. All rights reserved.

gigabytes are produced when observing a DUT for time periods as short as 1 min. As a consequence, when capturing and analysing EM data on-site, large data storage space, memory, and processing power are required on the analyst's computer. Besides the hardware requirements, the specific information leaking EM frequencies of a computing device is not always known. The most common approach taken to counteract this unknown is to listen to a wider band of frequencies around the processor's clock frequency and process that entire bandwidth using EM-SCA techniques. For example, an IoT device that has a processor running at $1.4GHz$ can be observed with a bandwidth of $20MHz$ using an SDR device. If a Short-Time Fourier Transform (STFT) was applied to this data with a 1 ms long non-overlapping time window, it produces $20,000$ potential frequency channels that can contain leaked information. STFT of a signal is produced by taking segments of the time domain EM signal and converting each of them into frequency domain. This high dimensionality of the EM data often makes it unfeasible to process in real-time.

While there is a large number of frequency channels that can potentially carry information about the activity of a device processor, typically a small subset of them are useful. There can be channels that carry redundant information while some others may not leak any information at all. Therefore, the identification of frequency channels that are useful out of the large number of available channels can improve the efficiency of performing EM-SCA. In order to address this problem, this work presents a systematic methodology to identify information leaking frequency channels from high dimensional EM data.

Contribution of this work

The contribution of this work can be summarised as follows:

- Experimental evaluation of multiple filtering methods to select a manageable number of frequency channels from a high dimensional EM data set.
- Introduction of a methodology using a Random Forest classifier to identify information-leaking frequency channels from high dimensional EM side-channel data.
- Demonstration of the effectiveness of the channel selection methodology by classifying software activities performed on a representative IoT device by observing its EM emissions.

2. Related work

The analysis of unintentional electromagnetic radiation has been identified as a method to eavesdrop on electrical and electronic devices for decades (Sayakkara et al., 2019b). It has been demonstrated that early analogue computer video displays with cathode ray tubes (CRT) leak sufficient amount of information to reconstruct the content being displayed through their EM radiation (Van Eck, 1985) and the same technique holds true for digital LCD monitors (Sayakkara et al., 2018a). With the availability of off-the-shelf hardware capable of capturing weak EM signals combined with computers with sufficient processing power to analyse the data, EM-SCA on a variety of computing devices has recently become more viable. As a result, a multitude of research has been conducted on various EM-SCA techniques including software anomaly detection and cryptographic key recovery (Stone and Stone, 2015; Nazari et al., 2017; Camurati et al., 2018; Sayakkara et al., 2019b).

Kocher et al. demonstrated that power consumption of a computing device can be used as a side-channel to extract cryptographic keys (Kocher et al., 1999, 2011). When the CPU of the computing device performs cryptographic operations, each value assigned to its registers is reflected in its power consumption. By collecting a sufficient number of power consumption traces during cryptographic operations with the same key, it is possible to reveal the key using techniques like differential power analysis (DPA). The power consumption of the CPU is directly associated with the EM radiation of the device, which opens up the EM side-channel. Therefore, variants of power analysis algorithms, such as differential electromagnetic analysis (DEMA), were introduced later in order to recover cryptographic keys using EM traces (Quisquater and Samyde, 2001; Gandolfi et al., 2001). The success of a cryptographic key recovery attack using DEMA heavily depends on the precise alignment of consecutive EM traces. Due to this reason, EM traces are acquired by instrumenting the target device in order to precisely synchronise the signal acquisition device and the target device.

In addition to the information security aspects of EM-SCA techniques, digital forensics is another field that can greatly benefit. Souvignet and Frinken suggested that power analysis attacks can be used to extract data from smart-cards used by malicious skimmer devices as a method to identify victims in a forensic investigation. However, it requires physically tapping into the device being investigated leading to potential inadvertent tampering of evidence (Souvignet and Frinken, 2013). In contrast, EM-SCA techniques are more suitable for digital evidence acquisition as they do not require any physical alterations to the device being investigated (Sayakkara et al., 2019a). Unfortunately, cryptographic key recovery is still a challenging task with EM-SCA due to the fact that EM traces have to be acquired with precise alignment, which typically requires physical instrumentation of the device.

When using machine learning algorithms to detect patterns in time series data, such as EM emissions, recurrent neural networks (RNN) can play a major role. An RNN is a specific type of neural network where the parameters generated during one-time instance are reused as input to the network again in consecutive time instance (Giles et al., 1994). It enables the learning of patterns that occur in data sequences along the time domain. Wang et al. evaluated the effectiveness of long-short-term-memory (LSTM), a variant of RNN, against traditional multi-layer perceptron (MLP) neural networks in classifying EM emission signals from various embedded devices (Wang et al., 2018). They show that both MLP and LSTM networks perform well in detecting the behaviour of the firmware running on their target device. Meanwhile, Han et al. demonstrated the potential of using LSTM networks in identifying control flow of programmable logic controllers (PLC) in industrial environments (Han et al., 2017). Their work indicates that sliding window sampling of EM signals can be effectively used to track the control flow of a program with sufficient resolution to identify malfunctions.

3. Electromagnetic side-channel data

EM radiation is the result of time-varying electrical currents that travel through a conductor. Digital electronics (computers in particular) rely on high-frequency electrical pulses. Due to these fast electrical pulses, running computing devices result in strong EM radiation. Among the various components of a computer, such as network controllers, memory, data bus lines, video graphic controllers, etc., the strongest and most important EM source is the CPU. A CPU consists of a system clock that runs at a high frequency in a scale of gigahertz. Therefore, an EM emission that has a frequency equal to the frequency of the system clock can be observed from a running CPU. When a CPU executes instructions and manipulates data on its internal registers, the hamming weight of those bits have been shown to modulate the EM emission of the system clock (Callan et al., 2014; Prvulovic et al., 2016). Therefore, the EM signals of the CPU open up the opportunity to perform EM-SCA on the internal software activity of a computing device.

The observation of EM side-channel emissions for EM-SCA requires special hardware equipment and software tools. Software-defined radios (SDRs) are becoming popular as an easy-to-handle EM signal observation platform among researchers and radio enthusiasts and are significantly more affordable than their digital oscilloscope counterparts. An SDR platform consists of a minimal hardware front-end to capture EM signals and produce digitised samples at a fast rate. Once digitised, stages of signal processing of the samples are performed entirely in software with the help of associated SDR software libraries. Due to the weak nature of EM emissions from CPUs, it is necessary to keep the signal acquisition device as close to the CPU chip as possible. This can be achieved by using an H-probe antenna and placing it directly on top of the CPU chip of the DUT.

When capturing EM emissions, it is necessary to use as fast a sampling rate as possible. The faster the sampling rate, the more accurate the information captured and saved from the original signal. However, faster sampling rates result in larger EM trace files. For example, consider a *HackRF* SDR device that has a maximum sampling rate of 20*MHz*. This particular device produces EM samples each with a size of 32 bytes. Therefore, when capturing EM signals for a period of 60 s using this device, it produces nearly 9*Gb* of EM data. As a result of these large volumes of EM data produced for very small observation time periods, processing them with EM-SCA techniques is a challenging task that requires large storage space, memory and processing power.

When listening to the EM emissions of the CPU of a computing device, the system clock frequency is typically considered as the key focus. However, it is not possible to predetermine the exact frequency channel that leaks information. In most practical scenarios, multiple frequencies closer to the CPU clock frequency can leak information. Due to this uncertainty of the information leaking frequency, it is necessary to observe EM emissions over a wider bandwidth around the CPU clock frequency. For example, consider a scenario of listening to the CPU of an IoT device running at 288*MHz*. Signals are sampled using an SDR device by tuning it to 288*MHz* and setting the bandwidth to 20*MHz*. Consequently, the captured EM traces include signals from 278*MHz* to 298*MHz*, as shown in Fig. 1. Among this wide band of signals, some will contain useful information leakages while a majority are unlikely to contain anything useful whatsoever. When listening to EM emissions of the CPU of a device, it is important to have an SDR device that supports the frequency in question. In the case of HackRF SDR that we use in this work, it is possible to tune to any frequency between 10*MHz* to 6*GHz*, which is sufficient in most practical scenarios (Ossmann, 2016).

The identification of information-leaking channels from a wide band of channels is a challenge that an attacker needs to overcome in order to efficiently perform EM-SCA. One potential approach is plotting randomly selected channels and visually identifying the channels that have apparent changes over time. Fig. 2 illustrates some of such randomly selected channels of EM data acquired from an IoT device. However, due to the availability of a large number of channels to inspect, this is an arduous task and not realistically feasible across the entire frequency range. Another potential method of reducing the number of channels to inspect is breaking the frequency domain into equally-sized groups and then averaging the signals within each group (Sayakkara et al., 2019a). The weakness of this method is the possibility of having multiple information leaking channels in the same group and getting them averaged. This likely results in a loss of valuable information leaking channels.

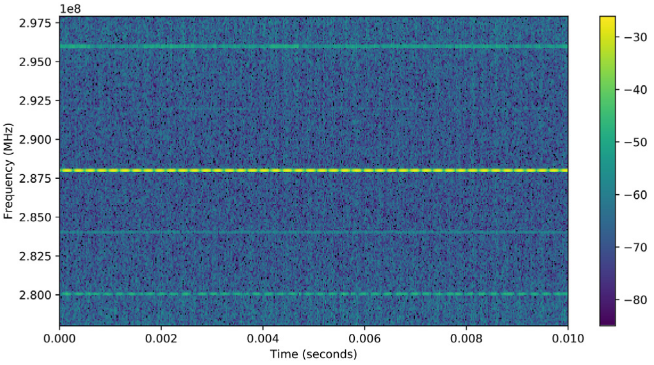


Fig. 1. Spectrogram of the observed EM signal from DUT.

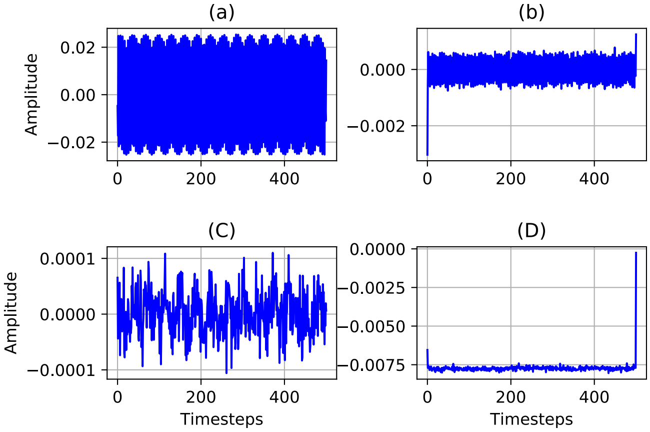


Fig. 2. Waveform of some randomly selected channels of the EM data set.

This work experimentally proposes to reduce the number of EM channels from a large bandwidth through multiple filtering techniques and feature selection with a random forest classifier. Fig. 3 illustrates the workflow to generate EM traces, identify channels, and finally perform EM-SCA with selected channels. Throughout the experimentation of this work, an *Arduino Leonardo* device is used as a representative DUT. Though its system clock runs at 16*MHz*, a higher harmonic observed at 288*MHz* was used when acquiring data in order to avoid external noise sources. Initially, EM traces were acquired while the device is running 10 different programs repetitively. Each EM trace file contains a 500 ms long observation with a sample rate of 20*MHz*. Collected EM traces are converted to the frequency domain through STFT function with a window size of 1 ms. The resulting data set contains $20,000$ individual frequency channels representing each of the 10 software activities of the IoT device across time. This data set is fed into the channel selection methodology presented in Section 4 to identify a limited set of information-leaking channels. When the channel selection phase is complete, real-time EM signals can be captured from the DUT and only the identified useful subset of channels are used to perform EM-SCA.

4. Experimentation and results

In order to identify which activity a user is performing on a device, the EM data of $20,000$ channels emitted by the Arduino Leonardo device was gathered. The objective of this work is to substantially reduce the number of considered channels. To predict which of the ten activities the device is executing, a supervised machine learning model for multi-classification was trained. Using a large number of channels will increase the size of the model, the amount of stored information, the required time to make predictions and also the energy consumption and time to transmit the signals. Additionally, a large number of features does not guarantee (and in some cases will negatively affect) the performance of the built models because some channels may be redundant or induce noise. This section describes the data set as well as the conducted experiments for selecting the best channels to create the model.

4.1. Description of the dataset

The data set was produced by capturing the EM emissions of an Arduino Leonardo device with a 20*MHz* sample rate using a HackRF SDR. The time-domain EM data were passed through an STFT function using a 1 ms Fourier transformation window. The resulting data set consists of $20,000$ frequency channels in one dimension and the time steps in the other dimension. A total of $1,002$ steps were available on the time dimension where the 10 different software activities are represented with equally spaced time periods. The 10 software activities that ran on the Arduino device are named as `Activity 0` up to `Activity 9`. Listing 1 illustrates an example program used to produce the data set. The activities differ from each other from the number of `for` loops in the program, however, keeping the time complexity at $O(n)$ on each.

The choice of software activities was made with the goal of exploring the possibility of detecting even minor variations in the code. Therefore, the difference between two consecutive software activities is maintained at a minimum. Since the scope of this work limits to IoT devices, the number of possible software activities that can occur on a device is limited. Therefore, this experiment assumes that the number of software activities that are being searched for are known in the first place. However, when dealing with general purpose computers, this may not be viable due to the complex nature of software behaviours that are possible on high-end computers. In future experiments, the feature selection methods will be tested with real-world IoT device firmware applications. To conduct the experiments, a MacBook Pro with a 2.3 GHz Intel Core i5 processor and a memory of Âă8Gb was used.

```
/* Arduino test program */
void setup(){
}
void loop(){
    for(int i=0, i<20, i++) { delay(10); }
    for(int j=0, j<20, j++) { delay(10); }
    /* further loops */
}
```

Listing 1: An example *Arduino* program used to produce EM data for the experiments.

4.2. Experimentation methodology

The main goal of this work is to substantially reduce the number of channels processed by the supervised model to a more manageable number without compromising the accuracy of the classifiers. As shown in Fig. 4, the prediction of the activity involves several steps. In the first step, the information of the channels is collected using the SDR device. In the second step, the relevant channels are selected. Later, there are two optional steps: reducing the number of features by applying Recursive Feature Elimination, and/or applying a time window where features in the time and frequency domains are extracted. The last two steps are the creation and evaluation of a classification machine learning model to predict the activity.

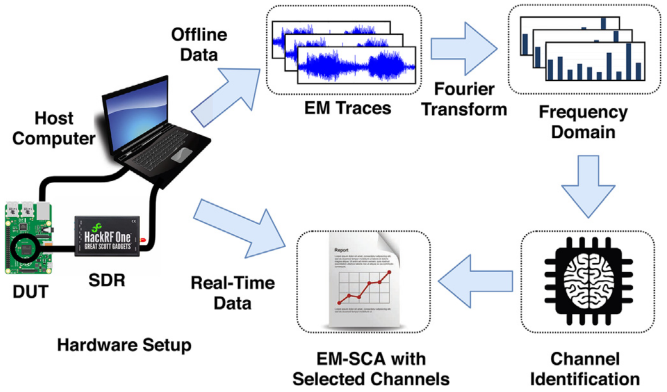


Fig. 3. The workflow to generate EM traces, identify channels, and finally perform EM-SCA with selected channels.

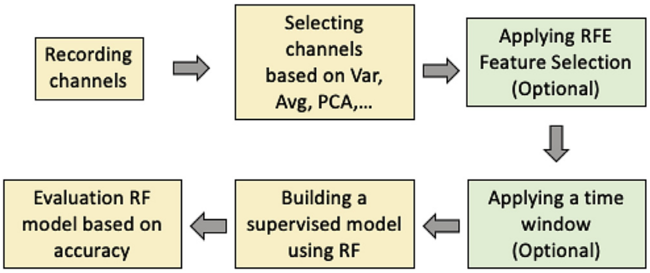

Recording channels
Selecting channels based on Var, Avg, PCA,...
Applying RFE Feature Selection (Optional)
Applying a time window (Optional)
Building a supervised model using RF
Evaluation RF model based on accuracy

Fig. 4. The steps to find out the optimal channels to perform experiments.

4.3. Experimentation

In order to compare the proposed solutions, 100 channels out of $20,000$ (which represents 0.5% of the total channels) were selected. Additionally, all models were created using the well-known machine learning method *Random Forests* (RF). Random Forests are highly suitable for this kind of application since they are fast and very accurate in prediction (Liaw et al., 2002). RF can process datasets with noise and `NaN` (Not a Number) values. RF has two main parameters: the number of created trees (estimators) and the depth of those trees. The final prediction is the majority vote among all the created trees. In these experiments, all the RF methods use 500 estimators and the trees have a maximum depth of 50 levels. The accuracy of the tested models was the average after applying cross-validation with five partitions and ten repetitions.

Whenever the average and variance were calculated the outliers values were removed. Outliers were considered as those points that had a Z-score absolute value higher than 3.[1] After taking a closer look at the values of the channels, most channels were found to contain no useful information. That is to say, almost all their values are very close to zero. Experimentation demonstrated two things: first, most channels have small variance and second, the average of all the sample points of each channel is, in general, very low. With these first hand insights, various experiments conducted and their results are outlined in the following subsections.

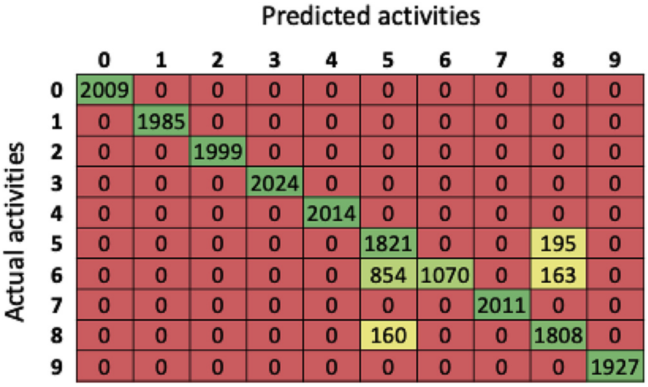
4.3.1. Experiment I: using 20,000 channels

In this first experiment, a supervised model was created using all the available channels. The reason to do this is to have a baseline

[1] Z-scores are used in statistics to measure an observation's deviation from the group's mean value (Altman et al., 2000). They are also known as Altman Z scores due to their developer, Edward Altman. According to the normal distribution table, 99% of the values will have an absolute value of less than 3.

Table 1
Average accuracy per class using the entire 20, 000 channels.

Class	Accuracy
0	1
1	1
2	1
3	1
4	1
5	0.9043
6	0.5129
7	1
8	0.9199
9	1


Predicted activities
Actual activities

	0	1	2	3	4	5	6	7	8	9
0	2009	0	0	0	0	0	0	0	0	0
1	0	1985	0	0	0	0	0	0	0	0
2	0	0	1999	0	0	0	0	0	0	0
3	0	0	0	2024	0	0	0	0	0	0
4	0	0	0	0	2014	0	0	0	0	0
5	0	0	0	0	0	1821	0	0	195	0
6	0	0	0	0	0	854	1070	0	163	0
7	0	0	0	0	0	0	0	2011	0	0
8	0	0	0	0	0	160	0	0	1808	0
9	0	0	0	0	0	0	0	0	0	1927

Fig. 5. The confusion matrix of classifying activities using all the channels.

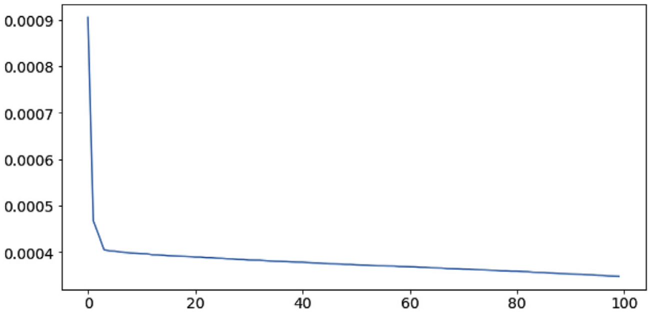


Fig. 6. Variance (y-axis) of the top 100 eigenvalues (x-values) when applying principal component analysis.

for performance comparison of the built models with a fewer number of channels against the performance of the models using all channels. Due to the fact that the number of channels is very high, 5,000 trees were used as opposed to the 500 used in the rest of the experiments. The average accuracy of the experiments conducted is 0.9315, and the time to predict 2,004 samples was 7.7435 s.

The results of the experiments are shown in Table 1 and the confusion matrix of the results is shown in Fig. 5. The results demonstrate that the accuracy for classes (0, 1, 2, 3, 4, 7 and 9) were 100% correct, for classes (5 and 8) are acceptable, but the accuracy for class 6 is very low. The algorithm confuses class 6 with classes 5 and 8 quite often (25% for each class). If class 6 was not considered the overall accuracy would be 0.9804.

4.3.2. *Experiment II: principal component analysis*

Since 20, 000 is a large number for applying wrapper methods (methods that implement supervised models such as *Non-dominated Sorting Genetic Algorithm* (NSGA-II) (Deb et al., 2000) and *Recursive Feature Elimination* (RFE) (Guyon et al., 2002)), a filter method was applied as a first step.[2] Normally, when the number of variables is very large, as a first step, filter methods are applied to discard variables with low variation and subsequently wrapper methods are applied to create a more accurate selection.

In this second experiment, we applied Principal Component Analysis (PCA) (Jolliffe, 2011), which applies a linear combination of weighted variables to drastically reduce the number of features. The new features are linear combinations of the previous ones and are called *eigenvectors* (also called *principal components*). Each of the eigenvectors has an eigenvalue and they are ordered according to this value in such a manner that the first components have more information than the last ones (which can be rejected). Ideally, PCA will reduce the feature space without losing significant information, which allows the creation of models in less time and with a higher accuracy. In Fig. 6, the top 100 eigenvalues are presented that were used to create a predictive model. PCA is well-known for being fast and effective.

After running the experiments, the average accuracy of PCA was found to be 0.1870, which is very poor. This is likely due to a combination of the high number of features, and most features being constant with low values. Also, because of the high number of considered variables. The results by class are displayed in Table 2. The low performance of PCA was found to be not favourable and was discarded.

4.3.3. *Experiment III: channel selection based on variance*

In this experiment, the variance for each channel was calculated and subsequently, the highest 100 were selected. In order to calculate the variance, the outliers were removed. After calculating the variance, the spectrum for all the channels is represented in Fig. 7. As can be seen, the variance is very low for most channels. In order to increase the visibility of the lower peaks, the limit of y-axis is set to 0.000005 in this figure. However, the variance of 5th and 6th channels are 0.000282 and 0.000222 respectively that goes beyond the y-axis limit.

To create the experiments, the variance threshold was set to select at least the top 100 channels with the highest variance. The threshold was set to 1.0632×10^{-08} to finally select 103 channels. The selected channels were saved in a matrix along with their class and a Random Forest classifier was trained to predict the results. The average accuracy of the experiments is 0.5431. This is substantially higher than PCA experiment (0.1870) but still quite far from acceptable performance. The time for building each RF model with 100 features was 1 min and 37 s. The performance per class is shown in Table 3.

4.3.4. *Experiment IV: channels selection based on the average*

After observing that the results of applying channel selection based on the variance were not sufficient, a selection based on the average was explored (removing the outliers values as before). The logic behind this principle is that channels that are active tend to have high values while non-active channels will be most of the times close to zero. Fig. 8 depicts the different average values for the channels. After calculating the average, a threshold of 6.9936×10^{-05} was set to select the 100 highest channels. The

[2] Wrapper methods are generally more accurate than filters because they implement supervised models that consider the relationships between the variables. The downside of wrapper methods is that they require a much higher processing time than filters (Chandrashekar and Sahin, 2014).

Table 2
Average accuracy per class for the best 100 PCA components.

Class	Accuracy
0	0.3438
1	0.2823
2	0.3882
3	0.1457
4	0.1373
5	0.0935
6	0.1768
7	0.2459
8	0.1375
9	0.1555

Table 3
Average accuracy per class of the highest 103 channels ordered by variance.

Class	Accuracy
0	0.6473
1	0.5965
2	0.5391
3	0.5882
4	0.8027
5	0.2684
6	0.3845
7	0.4830
8	0.6272
9	0.5058

accuracy of the Random Forest algorithm was 0.5423, which was very similar to the selection made on the basis of variance. The performance per class is shown in Table 4.

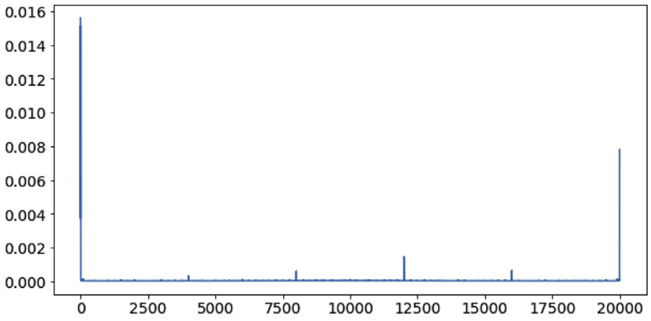


Fig. 8. Average (y-axis) for each of the 20, 000 channels (x-axis).

4.3.5. Experiment V: applying average per class and variance between the classes

In this experiment, a new and original approach was used. First, the average value of each channel per activity was calculated, removing the outlier values in such a way that the original matrix has 20, 000 rows (where each row represent a channel) and 10, 020 columns (where each column has the timestamp values of each channel for a given activity). The new matrix had the same number of rows (channels) but only 10 columns (one per activity). The second step was to calculate the variance between the average of the ten classes for all the channels, and the result is shown in Fig. 9.

We can see that the results of the spectrum seem to have much more diversity than the previous experiments where we used only the variance or only the average, although the variance values are close to zero (as it was in the previous experiments). The logic behind this procedure was that sought after channels were those that have very different values between activities (inter-class) whilst variance within the class (intra-class) was not the focus. A variance threshold of 0.000033 was used to select the highest 100 channels.

Subsequently, the accuracy of the predictions was calculated using the created Random Forest model. As can be seen in Table 5, the results of the experiments are much better than in previous experiments. For this experiment, three different numbers of channels were used. The average accuracy with the 10 higher channels is 0.5753, with the 100 higher channels is 0.9047 and with the 500 higher channels is 0.9395.

4.3.6. Experiment VI: applying recursive feature selection

Recursive feature elimination (RFE) is a wrapper method, i.e., implements supervised models during its execution, for selecting features proposed by Guyon et al. (2002). RFE initially creates a model using all the possible attributes of the data set. Then, each attribute is ranked according to its importance. RFE rejects the weakest attributes and creates a new model, whose performance is again evaluated. RFE repeats the process until it reaches the minimum number of required features. In order to have more reliable results, RFE applies cross-validation. As shown in Fig. 10, RFE gives a list of feature sets along with the corresponding model's performance. The optimal subset of features is that results in a model with the highest performance according to the selected metric. This is generally accuracy, ROC or F1-Score (in our case it was accuracy). The time to process the RFE selection was approximately 16 h.

The average accuracy of the RF model created with 81 channels is 0.9047. However, if we remove class 6, the performance improves to 0.9503. The instances of class 6 in the data set presents very similar values to those of classes 5 and 8. That is the reason for RF algorithm to find it very difficult to make clear rules to distinguish one from the another. The fact that we are having such a high

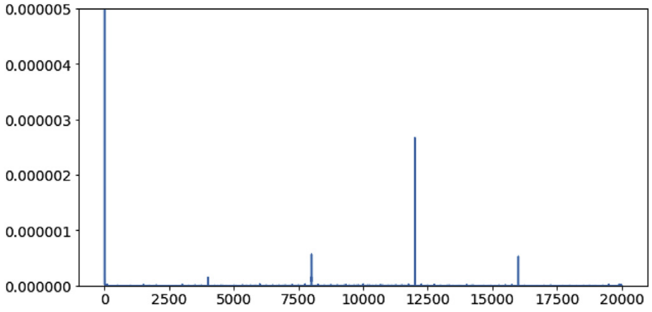


Fig. 7. The variance (y-axis) of the 20, 000 channels (x-axis). The limit of y-axis is set to 0.000005 in order to visualise lower values. However, the variance of 5th and 6th channels are 0.000282 and 0.000222 respectively.

Table 4
Average accuracy per class of the highest 100 channels ordered by average.

Class	Accuracy
0	0.6555
1	0.6067
2	0.5197
3	0.5770
4	0.8025
5	0.2694
6	0.3696
7	0.4939
8	0.6288
9	0.5196

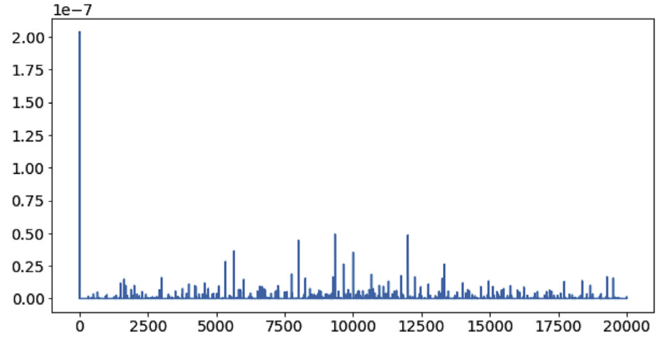


Fig. 9. Variance between the average of each of the classes for all the channels.

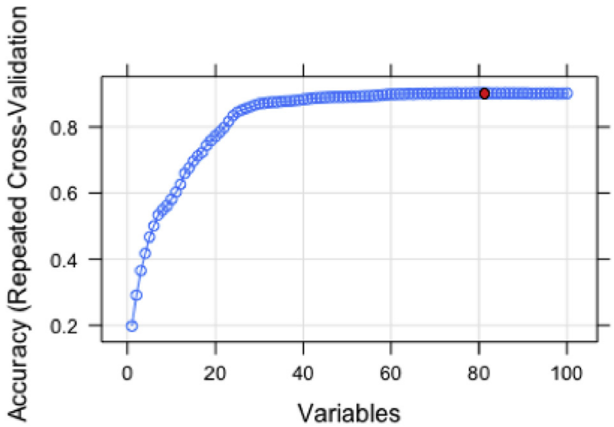


Fig. 10. The optimal number of features (model with highest performance) for the RFE algorithm is 81 (marked with a red dot).

accuracy even when the activities in the data set are so similar makes us very optimistic about being able to accurately differentiate real-world IoT firmware activities in the future. Fig. 11 illustrates the confusion matrix of the experimental results. Table 6 illustrates the classification accuracy for the 10 classes when channels are selected with RFE.

4.3.7. *Experiment VII: using a time window of 50 timestamps*

The application of a time window is common practice when dealing with time-series data and has been applied to other domains, such as human activity recognition (Ponce et al., 2016). Theoretically, if a decision is being taken only considering a timestamp, it will be more difficult to predict that when a time window, containing several timestamps, is used. As a result, the premise is to extract features in both the time domain and the frequency domain.

The extracted features for applying the time window were:

- **Time domain:** mean, standard deviation, root mean square, maximal amplitude, minimal amplitude, median, number of zero-crossing, skewness, kurtosis, first-quartile, third-quartile, autocorrelation.
- **Frequency domain:** mean frequency, median frequency, entropy, energy, principal frequency, spectral centroid.

Table 7 illustrates the classification accuracy results when using a time window of 50 timestamps. The average accuracy of the experiments were 0.8000. The low accuracy was probably due to the fact that the model needs more samples to be trained. Applying a time window reduces the number of samples.

5. Discussion

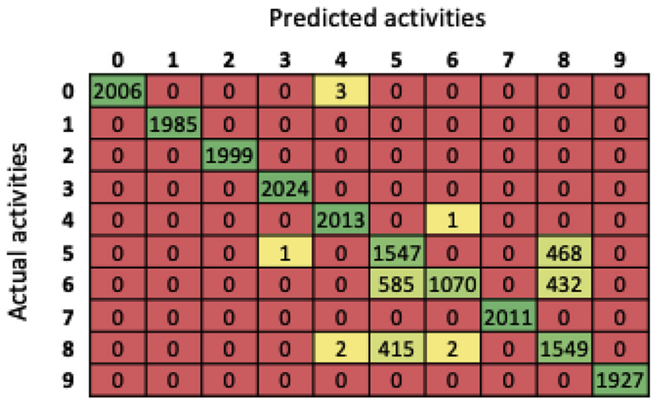
There exists an important potential in applying EM-SCA to analyse computing devices in digital forensic cases. With the recent emergence of IoT devices, they are increasingly being used for malicious purposes. For example, certain IoT smart bulbs have shown to be prone to attacks that reprogram them remotely (Ronen et al., 2017). Once re-programmed, the attacker may switch the devices on and off at their will making it difficult to distinguish a legitimate action from a malicious action involving light bulbs in a building. Upon the arrival of law-enforcement in such a crime scene, a quick EM-SCA can help to verify the firmware running on the devices thus revealing any malicious modifications. The reduction of the number of information-leaking channels that needs to be analysed makes such on-the-spot analysis with portable computing devices, e.g., laptops, possible.

Table 5
Average accuracy per class after calculating the variance between the average per activity.

Class	10 ch.	100 ch.	500 ch.
0	0.8220	0.9995	1
1	0.7364	1	1
2	0.8019	1	1
3	0.9090	1	1
4	0.5613	0.9990	1
5	0.3531	0.7749	0.9351
6	0.3592	0.5135	0.5179
7	0.4028	1	1
8	0.3828	0.7603	0.9422
9	0.4247	1	1
Avg	**0.5753**	**0.9047**	**0.9395**

The experiments presented in this work are performed in an exploratory manner by starting with the entire EM frequency channels and applying different selection techniques to detect the best channels. The reasoning behind the elimination of channels with variance and average was the fact that if a channel does not show any considerable difference among software activities, then such channels are not sufficiently distinguishable between each other. It is important to note that the final set of channels selected by the methodology in our evaluations are suitable only to distinguish between considered software activities running on the experimental device. The scope of our evaluation does not focus on selecting channels that are suitable to perform other forms of EM-SCA, such as cryptographic key retrieval.

The presented methodology is to identify the specific frequency channels from an EM data set and subsequently using those channels to classify real-time EM data. The experimental


Predicted activities
Actual activities

	0	1	2	3	4	5	6	7	8	9
0	2006	0	0	0	3	0	0	0	0	0
1	0	1985	0	0	0	0	0	0	0	0
2	0	0	1999	0	0	0	0	0	0	0
3	0	0	0	2024	0	0	0	0	0	0
4	0	0	0	0	2013	0	1	0	0	0
5	0	0	0	1	0	1547	0	0	468	0
6	0	0	0	0	0	585	1070	0	432	0
7	0	0	0	0	0	0	0	2011	0	0
8	0	0	0	0	2	415	2	0	1549	0
9	0	0	0	0	0	0	0	0	0	1927

Fig. 11. The confusion matrix for the 10 Activities and 81 features, i.e., the optimal number for RFE.

Table 6
Average accuracy per class of the selected 81 channels by RFE.

Class	Accuracy
0	0.9986
1	1
2	1
3	1
4	0.9994
5	0.768
6	0.5129
7	1
8	0.7868
9	1

Table 7
Result of the experiments when applying a time window of 50 samples.

Class	Accuracy
0	0.8857
1	0.9800
2	0.9500
3	0.9175
4	1.0000
5	0.5867
6	0.4255
7	0.8583
8	0.5467
9	0.9750

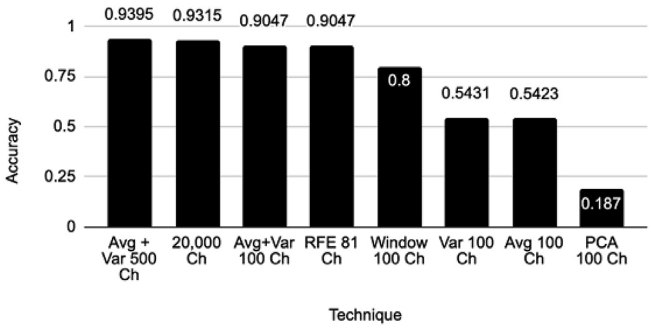


Fig. 12. Summary of the experimental results with different techniques.

evaluations performed in this work are focused on the channel selection phase. The identified channels are used to classify the software activities in captured EM traces. It is obvious that in the second phase of the methodology, the identified channels can be used to classify data captured in real-time using buffering with a fixed window size. Future experiments will use this technique with real-time data for further evaluations.

The number of software activities used in this experiment is limited to 10. Therefore, the channels identified by the experiments represent the best channels that leak information about those 10 specific software activities. It is possible that for a different set of software activities, the set of channels to be selected may be different. This means that in order to identify a specific set of software activities running on a particular computing device, a unique set of channels should be identified by following the methodology presented in this work. That means, in order to identify specific previously known software activities, a specific previously identified channels can be used. In practical investigation scenarios with EM-SCA, the investigator has to specifically use those predetermined channels depending on what target software activity they are looking for.

While `Activity 6` underperform in classifications compared to others, it was not possible to identify the exact reason behind that. The activity only contains a difference of a single `for` loop from its neighbouring activities, i.e., `Activity 5` and `Activity 7`. It is necessary to further study the factors of a software program that cause significantly distinguishable emission pattern in unique channels. The difficulties of distinguishing certain software programs such as `Activity 6` is only explainable with a better understanding on such factors. Future studies will focus on that aspect.

For the same set of software activities, the information leaking frequency channels can vary across different computing devices due to the differences such as CPU chip, the design of printed circuit board (PCB), etc. Therefore, the identification of specific frequency channels using the methodology presented in this work should be used to profile a specific type of computing device. For example, multiple variants of Arduino devices exist that contains the same CPU model. The identified channels are not interchangeable across different makes and models of devices.

6. Conclusion and future work

This paper presented a methodology to identify the frequency channels from a highly dimensional EM side-channel data set that leak software behaviour-related information. The choice of the Random Forests machine learning method enables the fast classification of software activities. Multiple avenues were explored experimentally to identify the best performing method in order to reduce EM channels (see Fig. 12). Channel selection with variance and average did not result in a sufficient classification accuracy for the considered software activities. However, the average per class and variance between classes proved to be effective in classifying software activities with an accuracy of 94% by reducing the channel space to 500. Furthermore, when applying the RFE method, the 10 software activities were identified with a 90% classification accuracy. The performance of the classification is seriously undermined by `Activity 6` that achieved 51% classification accuracy. If we do not consider this class, the accuracy of the models after applying the average together with the variance will be 98.6% when using 500 features and 94.81% when using 100.

These results indicate that high dimensional EM side-channel data can be reduced drastically to a manageable set of dimensions that are sufficient to accurately identify software activities running on a computing device. The evaluation of this work uses a limited number of firmware as target software activities for classification. It is important to further evaluate the feature selection methodology with real-world IoT applications, such as data encryption operations, file system read/write operations, networking protocols, etc. Multiple IoT devices of the same type needs to be used to cross-validate the applicability of identified channels on the same type of devices. Furthermore, when filtering outliers of EM channels, use of statistical mode is another potential approach to investigate in the future. Finally, it is important to apply and evaluate the potential of deep learning models in classification of software activities with selected channels in the future.

References

Altman, E.I., et al., 2000. Predicting Financial Distress of Companies: Revisiting the Z-Score and Zeta Models. Stern School of Business, New York University, pp. 9–12.

Callan, R., Zajić, A., Prvulovic, M., 2014. A practical methodology for measuring the side-channel signal available to the attacker for instruction-level events. In: Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture. IEEE Computer Society, pp. 242–254.

Camurati, G., Poeplau, S., Muench, M., Hayes, T., Francillon, A., 2018. Screaming channels: when electromagnetic side channels meet radio transceivers. In: Proceedings of the 25th ACM Conference on Computer and Communications Security (CCS), pp. 163–177. CCS '18; ACM.

Chandrashekar, G., Sahin, F., 2014. A survey on feature selection methods. Comput. Electr. Eng. 40 (1), 16–28.

Deb, K., Agrawal, S., Pratap, A., Meyarivan, T., 2000. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. In: International Conference on Parallel Problem Solving from Nature. Springer, pp. 849–858.

Gandolfi, K., Mourtel, C., Olivier, F., 2001. Electromagnetic analysis: concrete results. In: International Workshop on Cryptographic Hardware and Embedded Systems (CHES). Springer, pp. 251–261.

Giles, C.L., Kuhn, G.M., Williams, R.J., 1994. Dynamic recurrent neural networks: theory and applications. IEEE Trans. Neural Network. 5 (2), 153–156.

Guyon, I., Weston, J., Barnhill, S., Vapnik, V., 2002. Gene selection for cancer classification using support vector machines. Mach. Learn. 46 (1–3), 389–422.

Han, Y., Etigowni, S., Liu, H., Zonouz, S., Petropulu, A., 2017. Watch me, but don't touch me! contactless control flow monitoring via electromagnetic emanations. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, pp. 1095–1108. ACM.

Jolliffe, I., 2011. Principal Component Analysis. Springer.

Kocher, P., Jaffe, J., Jun, B., 1999. Differential power analysis. In: Advances in Cryptology (CRYPTO '99). Springer, 789–789.

Kocher, P., Jaffe, J., Jun, B., Rohatgi, P., 2011. Introduction to differential power analysis. J. Cryptographic Eng. 1 (1), 5–27.

Liaw, A., Wiener, M., et al., 2002. Classification and regression by randomforest. R. News 2 (3), 18–22.

Lillis, D., Becker, B., O'Sullivan, T., Scanlon, M., 2016. Current challenges and future research areas for digital forensic investigation. In: The 11th ADFSL Conference on Digital Forensics, Security and Law (CDFSL 2016). ADFSL, Daytona Beach, FL, USA, pp. 9–20.

Nazari, A., Sehatbakhsh, N., Alam, M., Zajic, A., Prvulovic, M., 2017. Eddie: Em-based detection of deviations in program execution. In: 2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA). IEEE, pp. 333–346.

Ossmann, M., 2016. Software defined radio with hackrf. Great Scott Gadgets. https://greatscottgadgets.com/sdr.

Ponce, H., Miralles-Pechuán, L., Martínez-Villaseñor, M., 2016. A flexible approach for human activity recognition using artificial hydrocarbon networks. Sensors 16 (11), 1715.

Prvulovic, M., Zajić, A., Callan, R.L., Wang, C.J., 2016. A method for finding frequency-modulated and amplitude-modulated electromagnetic emanations in computer systems. IEEE Trans. Electromagn C. 59 (1), 34–42.

Quisquater, J.J., Samyde, D., 2001. Electromagnetic analysis (EMA): measures and counter-measures for smart cards. In: International Conference on Research in Smart Cards. Springer, pp. 200–210.

Ronen, E., Shamir, A., Weingarten, A.O., O'Flynn, C., 2017. IoT goes nuclear: creating a ZigBee chain reaction. In: 2017 IEEE Symposium on Security and Privacy (SP). IEEE, pp. 195–212.

Sayakkara, A., Le-Khac, N.A., Scanlon, M., 2018a. Accuracy enhancement of electromagnetic side-channel attacks on computer monitors. In: Proceedings of the 13th International Conference on Availability, Reliability and Security. ACM, New York, NY, USA, ISBN 978-1-4503-6448-5, pp. 15:1–15:9. https://doi.org/10.1145/3230833.3234690. ARES 2018.

Sayakkara, A., Le-Khac, N.A., Scanlon, M., 2018b. Electromagnetic side-channel attacks: potential for progressing hindered digital forensic analysis. In: Proceedings of the International Workshop on Speculative Side Channel Analysis (WoSSCA 2018). ACM, pp. 138–143.

Sayakkara, A., Le-Khac, N.A., Scanlon, M., 2019a. Leveraging electromagnetic side-channel analysis for the investigation of IoT devices. Digit. Invest. 29, 94–103. https://doi.org/10.1016/j.diin.2019.04.012.

Sayakkara, A., Le-Khac, N.A., Scanlon, M., 2019b. A survey of electromagnetic side-channel attacks and discussion on their case-progressing potential for digital forensics. Digit. Invest. 29, 43–54. https://doi.org/10.1016/j.diin.2019.03.002.

Souvignet, T., Frinken, J., 2013. Differential power analysis as a digital forensic tool. Forensic Sci. Int. 230 (1–3), 127–136.

Stone, B.D., Stone, S.J., 2015. Radio frequency based reverse engineering of microcontroller program execution. In: 2015 National Aerospace and Electronics Conference (NAECON). IEEE, pp. 159–164.

Van Eck, W., 1985. Electromagnetic radiation from video display units: an eavesdropping risk? Comput. Secur. 4 (4), 269–286.

Wang, X., Zhou, Q., Harer, J., Brown, G., Qiu, S., Dou, Z., Wang, J., Hinton, A., Gonzalez, C.A., Chin, P., 2018. Deep learning-based classification and anomaly detection of side-channel signals. In: Cyber Sensing 2018, vol. 10630. International Society for Optics and Photonics, p. 1063006.