ELECTRO-MAGNETIC SIDE-CHANNEL ATTACK THROUGH LEARNED DENOISING AND CLASSIFICATION
Florian Lemarchand, Cyril Marlin, Florent Montreuil, Erwan Nogues, Maxime Pelcat

To cite this version:

HAL Id: hal-02477654
https://hal.science/hal-02477654
Submitted on 13 Feb 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
ELECTRO-MAGNETIC SIDE-CHANNEL ATTACK
THROUGH LEARNED DENOISING AND CLASSIFICATION

Florian Lemarchand *, Cyril Marlin §, Florent Montreuil §,
Erwan Nogues *§, Maxime Pelcat *

* Univ. Rennes, INSA Rennes, IETR - UMR CNRS 6164
 § DGA-MI, Bruz

ABSTRACT
This paper proposes an upgraded Electro Magnetic (EM) side-channel attack that automatically reconstructs the intercepted data. A novel system is introduced, running in parallel with leakage signal interception and catching compromising data on the fly. Leveraging on deep learning and Character Recognition (CR), the proposed system retrieves more than 57% of characters present in intercepted signals regardless of signal type: analog or digital. The building of the learning database is detailed and the resulting data made publicly available. The solution is based on Software-Defined Radio (SDR) and Graphics Processing Unit (GPU) architectures. It can be easily deployed onto existing information systems to detect compromising data leakage that should be kept secret.

Index Terms— Electro-Magnetic Side-Channel, Denoising, Automation

1. INTRODUCTION
All electronic devices produce Electro Magnetic (EM) emanations that not only interfere with radio devices but also compromise the data handled by the information system. A third party may perform a side-channel analysis and recover the original information, hence compromising the system privacy. While pioneering work of the domain focused on analog signals [1], recent studies extend the eavesdropping exploit using an EM side-channel attack to digital signals and embedded circuits [2]. The attacker’s profile is also taking on a new dimension with the increased performance of Software-Defined Radio (SDR). With recent advances in radio equipment, an attacker can leverage on advanced signal processing to further stretch the limits of the side-channel attack using EM emanations [3]. With the fast evolution of deep neural networks, an attacker can extract patterns or even the full structured content of the intercepted data with a high degree of confidence and a limited execution time.

In this paper, a learning-based method is proposed with the specialization of Mask R-CNN [4] as a denoiser and classifier. A complete system is demonstrated, embedding SDR and deep-learning, that detects and recovers leaked information at a distance of several tens of meters. It provides an automated solution where the data is interpreted directly. The solution is compared to other system setups.

The paper is organized as follows. Section 2 presents existing methods to recover information from EM emanations. Section 3 describes the proposed method for automatic character retrieval. Experimental results and detailed performances are exposed in Section 4. Section 5 concludes the paper.

This work is supported by the "Pôle d’Excellence Cyber", initiative of the French Ministry of the Armed Forces and the Bretagne council.

2. RELATED WORK
This paper focuses on two areas: EM side channel attacks on information systems and learning-based techniques that can recover information from noisy environments.

Van Eck et al. [1] published the first technical reports revealing how involuntary emissions originating from electronics devices can be exploited to compromise data. While the original work of the domain targeted cathode ray tube (CRT) screens and analog signals, Kuhn et al. [2] propose to use side-channel attacks to extract confidential data from liquid crystal displays (LCDs) targeting digital data. Subsequently, other types of systems have been attacked. Vuagnoux et al. [5] extend the principle of EM side-channel attack to digital data. Subsequently, other types of systems have been attacked. Vuagnoux et al. [5] extend the principle of EM side-channel attack to digital data. When retrieving visual information from an EM signal, an important part of the original information is lost through the leakage and interception process. This loss leads to a drop of the Signal to Noise Ratio (SNR) and a deterioration of spatial coherence into the reconstructed samples in the case of image data. Hence, denoising methods are needed. Image denoising by signal processing techniques has been extensively studied since it is an important step in many computer vision applications. BM3D [6], proposed by Dabov et al., is the state-of-the-art methods for Additive White Gaussian Noise (AWGN) removal using non-learned processing. BM3D uses thresholding and Wiener filtering into the transform domain. Block-Matching 3D (BM3D) is used in the experiments of Section 4.

Deep learning algorithms have recently stood out from the crowd for solving many signal processing problems. These trained models have an extreme ability to fit complex problems. Recent Graphics Processing Unit (GPU) architectures have been optimized to support deep learning workloads and have fostered ever deeper networks, mining structured information from data and providing results where expert-based algorithms fail. The spread of deep learning has occurred in the domain of image denoising and several models initially developed for other applications have been turned into denoisers. Denoising Convolutional Neural Network
Target proposed method to enhance the EM side-channel attack. Especially when it comes to ensure the data privacy throughout the life-cycle, such approaches are costly solutions or technically hard to use in practice especially when it comes to nullify the emanations [16]. However, these approaches are not suitable for all cases and require a novel experimental setup.

Conventional approaches exist to protect devices from eavesdropping. Such approaches appear under different code names such as TEMPEST [15] or Emission Security (EMSEC) and consist of shielding devices [2] to nullify the emanations, or using fonts that minimize the EM emanations [16]. However, these approaches are either costly solutions or technically hard to use in practice especially when it comes to ensure the data privacy throughout the lifecycle of a complex information system. The next section details the proposed method to enhance the EM side-channel attack.

3. PROPOSED SIDE-CHANNEL ATTACK

3.1. System Description

Figure 1 shows the proposed end-to-end solution. The method automatically reconstructs leaked visual information from compromising emanations. The setup is composed of two main elements. At first, the antenna and SDR processing capture in the Radio Frequency (RF) domain the leaked information originating from the displayed video. Then, the demodulated signal is processed by the host computer, recovering a noisy version of the original image [3] leaving room for advanced image processing techniques. On top of proposing an end-to-end solution from the capture to the data itself, the method uses a learning-based approach. It exploits the capturing of compromising signals and recognized automatically the leaked data.

A first step based on a Mask R-CNN (Mask R-CNN) architecture embeds the following: denoising, segmentation, character detection/ localization, and character recognition. A second step post-processes the Mask R-CNN output. A Hough transform is done for text line detection and a Bitap algorithm [17] is applied to approximate match information. This setup detects several forms of compromising emanations (analog or digital) and automatically triggers an alarm if critical information is leaking. Next sections detail how the method is trained and integrated.

3.2. Training Dataset Construction

A substantial effort has been made on building a process that semi-automatically generates and labels datasets for supervised training. Each sample image is made up of a uniform background on which varied characters are printed. Using that process, an open data corpus of 123,610 labeled samples, specific to the problem at hand, has been created to further be used as training, validation and test datasets. This dataset is available online to train denoiser architectures in difficult conditions.

The proposed setup, to be trained, denoises the intercepted sample images and extracts their content, i.e. the detected characters and their positions. The input space that should be covered by the training dataset is large and three main types of interception variability can be observed. Firstly, interception induces an important loss of the information originally existing in the intercepted data. The noise level is directly linked to the distance between the antenna and the target. Several noise levels are generated by adding RF attenuation after the antenna. That loss itself causes inconsistencies in the rasterizing stage. Secondly, EM emanations can come from different sources, using different technologies, implying in turn different intercepted samples for the same reference image. The dataset covers Video Graphics Array (VGA), Display Port (DP), Digital Visual Interface (DVI) and High-Definition Multimedia Interface (HDMI) cables and connectors. Besides this unwanted variability, a synthetic third type of variability is introduced to solve the character retrieval. Many different characters are introduced in the corpus to be displayed on the attacked screen. They range from 11 to 70 points in size and they are both digits and letters, and letters are both upper and lower cases. Varied fonts, character colors and background colors, as well as varied character positions in the sample are used. Considering these different sources of variability, the dataset is built

Fig. 1: Experimental setup: the attacked system includes an eavesdropped screen (1) displaying sensitive information. It is connected to an information system (2). An interception chain including an SDR receiver (3) sends samples to a host computer (4) that implements signal processing including a deep learning denoiser and OCR.
3.3. Implemented Solution to Catch Compromising Data

In order to automate the interception of compromising data, the Mask R-CNN has been turned into a denoiser and classifier. The implementation is based on the one proposed by W. Abdulla. Mask R-CNN has been turned into a denoiser and classifier. The choice has been made to display on the target screen a sample containing patches of size 256 × 256 pixels (top-left image of Figure 2). For building the dataset, having multiple patches speeds the process up because smaller samples can be derived from a single screen interception and more variability can be introduced in the dataset. The main challenge when creating the dataset lies in the samples acquisition itself. Indeed, once intercepted, the samples are not directly usable. The interception process outputs samples such as the one of Figure 2 (middle-top) where intercepted characters are not aligned (temporally and spatially) with respective reference samples. An automated method is introduced that uses the porches, artificially colored in red in Figure 2 (middle-top), to align spatially samples. Porches are detected using brute-force search of large horizontal and vertical gradients (to find vertical and horizontal porches, respectively). A validation step ensures the temporal alignment, based on the insertion of a QRCode in the upper-left patch. If the QRCode is not directly usable. The interception process outputs samples such as the one of Figure 2 (middle-top) where intercepted characters are not aligned (temporally and spatially) with respective reference samples. An automated method is introduced that uses the porches, artificially colored in red in Figure 2 (middle-top), to align spatially samples. Porches are detected using brute-force search of large horizontal and vertical gradients (to find vertical and horizontal porches, respectively). A validation step ensures the temporal alignment, based on the insertion of a QRCode in the upper-left patch. If the QRCode is similar between the reference and the intercepted image, the image patches are introduced in the dataset.

Data augmentation [18] is used to enhance the dataset coverage area. It is done onto patches to add variability into the dataset and reinforce its learning capacity. Conventional methods are applied to raw samples to linearly transform them (Gaussian and median blur, salt and pepper noise, color inversion and contrast normalization).

Data augmentation [18] is used to enhance the dataset coverage area. It is done onto patches to add variability into the dataset and reinforce its learning capacity. Conventional methods are applied to raw samples to linearly transform them (Gaussian and median blur, salt and pepper noise, color inversion and contrast normalization).

The second stage, called head network, classifies the content and confidence information.

Character Recognition (OCR) or the Mask R-CNN classifier can directly infer the sample content (right) and propose some display and confidence information.

Fig. 2: A reference sample is displayed on the target screen (top-left). The interception module outputs uncalibrated samples. Vertical and horizontal porches (red) helps alignment and porch withdrawal (top-right). Samples are rescaled and split into patches to obtain the same layout than the reference set. The interception process outputs samples such as the one of Figure 2 (middle-top) where intercepted characters are not aligned (temporally and spatially) with respective reference samples. An automated method is introduced that uses the porches, artificially colored in red in Figure 2 (middle-top), to align spatially samples. Porches are detected using brute-force search of large horizontal and vertical gradients (to find vertical and horizontal porches, respectively). A validation step ensures the temporal alignment, based on the insertion of a QRCode in the upper-left patch. If the QRCode is similar between the reference and the intercepted image, the image patches are introduced in the dataset.

Data augmentation [18] is used to enhance the dataset coverage area. It is done onto patches to add variability into the dataset and reinforce its learning capacity. Conventional methods are applied to raw samples to linearly transform them (Gaussian and median blur, salt and pepper noise, color inversion and contrast normalization).

In order to automate the interception of compromising data, the Mask R-CNN has been turned into a denoiser and classifier. The implementation is based on the one proposed by W. Abdulla. Other learning-based and expert-based signal processing methods, discussed in Section 4.2, are also implemented to assess the quality of the proposed framework. Mask R-CNN is a framework adapted from the previous Faster R-CNN [19]. The network consists of two stages. The first stage, also known as backbone network, is a ResNet101 convolutional network extracting features out of the input samples. Based on the extracted features, a Region Proposal Network (RPN) proposes Region of Interests (RoIs) RoIs are regions in the sample where information deserves greater attention. The second stage, called head network, classifies the content and

returns bounding box coordinates for each of the RoIs. The main difference between Faster R-CNN and Mask R-CNN lies in an additional Fully Convolutional Network (FCN) branch running in parallel with the classification and extracting a binary mask for each RoI to provide a more accurate localization of the object of interest.

Mask R-CNN is not originally designed to be used for denoising but rather for instance segmentation. However, it fits well the targeted problem. Indeed, the problem is similar to a segmentation where signal has to be separated from noise. As a consequence, when properly feeding a trained Mask R-CNN network with noisy samples containing characters, one obtains lists of labels (i.e. characters recognition), as well as their bounding boxes (characters localization) and binary masks representing the content of the original clean sample. The setup of the classification branch allows to be language-independent and to add classes other than characters.

Two strategies can be employed to exploit Mask R-CNN components for the problem. The first idea is to draw the output masks of Mask R-CNN segmentation (Figure 3 left-hand side) and request an OCR to retrieve characters from the masks. A second possibility is to make use of the classification faculty of Mask R-CNN (Figure 3 right-hand side) and obtain a list of labels without using an OCR engine. The second method using the classifier of Mask R-CNN proves to be better in practice, as shown in Section 4.2.

The training strategy is to initialize the training process using pre-trained weights [22] for the MS COCO [23] dataset. First, the weights of the backbone are frozen and the head are trained together until convergence. This process is done to ensure the convergence and speed up training.

---

https://github.com/matterport/Mask_RCNN
4. EXPERIMENTAL RESULTS

4.1. Experimental Setup

The experimental setup is defined as follows: the eavesdropped display is 10 meters away from the interception antenna. A RF attenuator is inserted after the antenna. It ranges from 0 dB to 24 dB to simulate higher interception radius and generate a wide range of noise values. Compromising emanations are issued either by a VGA display, a DVI to DVI cable or an HDMI connector. The interception system is depicted in Figure 1: the antenna is bilog, the SDR device automatically recovering parameters is an Ettus X310 receiving with a 100 MHz bandwidth to recover the compromised information with a fine granularity. The host computer running post-processing has a Linux operating system, an Intel® Xeon® W-2125 Central Processing Unit (CPU) and an Nvidia GTX 1080 Ti GPU. The host computer rasters the compromising data using the CPU while the proposed learning-based denoiser/classifier runs also on the GPU.

4.2. Performance Comparison Between Data Catchers

The purpose of the exposed method is to analyze compromising emanations. Once a signal is detected and rasterized, intercepted emanations should be classified into compromising or not. Figure 4 illustrates the outputs of different implemented denoisers. More examples are available at. It is proposed to assess the data leak according to the ability of a model to retrieve original information. A ratio between the number of characters that a method correctly classifies from an intercepted sample, and the true number of characters in the corresponding clean reference is used as a metric.

The quality assessment method is the following. First, a sample containing a large number of characters is pseudo-randomly generated (similar to dataset construction). The sample is displayed on the eavesdropped screen and EM emanations are intercepted. The proposed denoising/retrieval is applied and the obtained results are compared to the reference sample. The method using Mask R-CNN produces directly a list of retrieved characters. Other methods, implemented to compare the efficiency of the proposal, use denoising in combination with the Tesseract. Tesseract is a well performing OCR engine, retrieving characters from images. It produces a list of characters retrieved from a denoised sample. As the output of Tesseract is of the same type as the output of Mask R-CNN classification, metrics can be extracted to fairly compare methods.

An end-to-end evaluation is used measuring the quality of character classification. A F-score classically used to evaluate classification model is computed using precision and recall. Precision is the number of true positives divided by the number of all positives. Recall is the number of true positives divided by the number of relevant samples, the set of relevant samples being the union of true positives and false negatives. For simplification and not use an alignment process, a true positive is chosen here to be the recognition of a character truly existing in the reference sample.

Table 1 presents the results of different data catchers on a test set of 12563 patches. All denoising methods are tested using Tesseract, and compared to Mask R-CNN classification used as OCR. Tesseract is first applied to raw (non-denoised) samples as a point of reference. BM3D is the only expert-based denoising solution tested. Noise2Noise, AutoEncoder, RaGAN and UNet are different deep learning networks configured as denoisers. As shown in Table 1, Mask R-CNN classification outperforms all other methods.

<table>
<thead>
<tr>
<th>Denoiser</th>
<th>OCR</th>
<th>F-Score</th>
<th>precision</th>
<th>recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw</td>
<td>Tesseract</td>
<td>0.04</td>
<td>0.20</td>
<td>0.02</td>
</tr>
<tr>
<td>BM3D</td>
<td></td>
<td>0.13</td>
<td>0.22</td>
<td>0.09</td>
</tr>
<tr>
<td>Noise2Noise</td>
<td></td>
<td>0.17</td>
<td>0.25</td>
<td>0.12</td>
</tr>
<tr>
<td>AutoEncoder</td>
<td></td>
<td>0.24</td>
<td>0.55</td>
<td>0.15</td>
</tr>
<tr>
<td>RaGAN</td>
<td></td>
<td>0.24</td>
<td>0.42</td>
<td>0.18</td>
</tr>
<tr>
<td>UNet</td>
<td></td>
<td>0.35</td>
<td>0.62</td>
<td>0.25</td>
</tr>
<tr>
<td>Mask R-CNN</td>
<td>Tesseract</td>
<td>0.55</td>
<td>0.82</td>
<td>0.42</td>
</tr>
<tr>
<td>Mask R-CNN</td>
<td></td>
<td>0.68</td>
<td>0.81</td>
<td>0.57</td>
</tr>
</tbody>
</table>

Table 1: Character recognition performance for several data catchers using either denoising and Tesseract, or Mask R-CNN (Mask R-CNN) classification. Mask R-CNN classifier outperforms others methods with a 0.68 F-score on the test set.

Table 2: Inference time for several data catchers using Tesseract or Mask R-CNN classification as OCR. Input resolution is 1200 × 1900 and it is processed using a split in 28 patches. Mask R-CNN classifier is slower than the autoencoder but still faster than BM3D.

<table>
<thead>
<tr>
<th>Denoiser</th>
<th>OCR</th>
<th>Inference Timing (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw</td>
<td>Tesseract</td>
<td>0.19</td>
</tr>
<tr>
<td>BM3D</td>
<td></td>
<td>21.8</td>
</tr>
<tr>
<td>Autoencoder</td>
<td></td>
<td>1.15</td>
</tr>
<tr>
<td>Mask R-CNN</td>
<td>Tesseract</td>
<td>4.22</td>
</tr>
<tr>
<td>Mask R-CNN</td>
<td></td>
<td>4.04</td>
</tr>
</tbody>
</table>

Table 2: Inference time for several data catchers using Tesseract or Mask R-CNN classification as OCR. Input resolution is 1200 × 1900 and it is processed using a split in 28 patches. Mask R-CNN classifier is slower than the autoencoder but still faster than BM3D.

Another key performance indicator of learning-based algorithms is inference time (Table 2). The proposed implementation using Mask R-CNN infers results from an input sample of resolution 1200 × 1900 in 4.04s in average. This inference time, although lower than BM3D latency, is admittance higher than other neural networks and hardly real-time. Nevertheless, the inference time of Mask R-CNN includes all the denoising process and provides a largely better retrieval score. In the context of a continuous listening of EM emanations, it provides an acceptable trade-off between processing time and interception performance. The optimization of the inference time could be considered as a future work with the recent advances in accelerating neural network inference.

5. CONCLUSIONS

Handling data while ensuring trust and privacy is challenging for information system designers. This paper presents how the attack surface can be enlarged with the introduction of deep learning in an EM side-channel attack. The proposed method uses Mask R-CNN as denoiser and it automatically recovers more than 57% of a leaked information for a wide range of interception distances. The proposal is software-based, and runs on the host computer of an off-the-shelf SDR platform.

https://github.com/opendenoising/extension
6. REFERENCES


