Experiment on Side-Channel Key-Recovery using a Real LPWA End-device

Kazuhide Fukushima[1], Damien Marion[2], Yuto Nakano[1], Adrien Facon[2], Shinsaku Kiyomoto[1] and Sylvain Guilley[2]

[1]*KDDI Research, Inc., 2-1-15 Ohara, Fujimino, Saitama 356-8502, Japan*

[2]*Secure-IC S.A.S., 80 avenue des Buttes de Coësmes Rennes, 35700, France*

Keywords: Internet of Things (IoT), Low-Power Wide-Area (LPWA), LoRaWAN, Side-channel Analysis, Correlation Power Analysis (CPA), Electromagnetic(EM)-leakage, AES.

Abstract: The Internet of things (IoT) has come into widespread use, and data protection and integrity are critical for connected IoT devices in order to maintain security and privacy. Low-power wide-area (LPWA) technologies for IoT wireless communication achieve data protection and integrity by using encryption and message authentication. However, side-channel analysis techniques exist that have the capacity to recover secret information from a device. In this paper, we apply a side-channel analysis technique to the payload encryption process and message authentication code generation process on a real LoRaWAN end-device. The entire AES-128 key for the payload encryption can be recovered with 260 electromagnetic(EM)-leakage traces and 12 bytes of the key for message authentication code generation can be recovered with 140 EM-leakage traces.

1 INTRODUCTION

The Internet of things (IoT) has been in widespread use for several years. IHS Markit (Howell, 2017) estimated that the number of IoT devices connected to the Internet is nearly 27 billion as of 2017, which will surge to 125 billion by 2030. IoT in the automobile and industry fields has attracted attention due to the projected growth rate. Moreover, Cisco (Bradley et al., 2013) has estimated there were 1.5 trillion IoT devices globally as of 2013, but 99.4% of these devices are still unconnected. This fact highlights the vast potential value of IoT.

Low-power wide-area (LPWA) is a term used todescribe communication technologies for IoT. These technologies are characterized by low power consumption, wide area coverage, and low cost. LPWA technologies can be roughly categorized into the licensed and unlicensed spectrum. LoRaWAN (LoRa Alliance$^{\text{TM}}$, 2017) and Sigfox (Sigfox, 2017) are typical examples protocols that run in the unlicensed spectrum, meaning that a license is not needed to build a network and provide services. LoRaWAN and LoRa are open standards developed by the LoRa Alliance. LoRaWAN defines the communication protocol and system architecture in the medium access control layer for the network while LoRa defines the physical layer or wireless modulation that enables long-range communication links. A LoRaWAN network can be built by buying equipment similar to wireless LAN. Conversely, only one company in each country can build a Sigfox network according to the policy of the French Sigfox company. LTE-M (GSMA, 2017) and NB-IoT (3GPP, 2016) operate over licensed spectrum, and mobile operators build a network and provide services. Their advantage is that existing LTE base stations can be used to build a new LPWA network.

Data protection and integrity are critical for connected IoT. For example, user privacy is compromised by location information and activity information acquired from a wearable device. As another motivating example, an air conditioner can be manipulated maliciously by modifying the value of the temperature sensor, which can lead to panic in crowded places, with possible fatalities. LPWA technologies achieve data protection and integrity by using encryption and message authentication. However, many attacks against the vulnerabilities of LPWA protocols have been proposed, and some of them are potential threats as they can extract the secret keys. Furthermore, side-channel analysis technologies exist that have the capacity to recover secret information from devices by using side-channel information including

Fukushima, K., Marion, D., Nakano, Y., Facon, A., Kiyomoto, S. and Guilley, S.
Experiment on Side-Channel Key-Recovery using a Real LPWA End-device.
DOI: 10.5220/0007259500670074
In *Proceedings of the 5th International Conference on Information Systems Security and Privacy (ICISSP 2019)*, pages 67-74
ISBN: 978-989-758-359-9

timing information, power consumption, electromagnetic leaks, sound, and heat.

Our Contributions. We apply a side-channel analysis technique to the payload encryption process and message authentication code generation process for data transmission on a real LoRaWAN end-device. The entire AES-128 key for the payload encryption can be recovered with 350 electromagnetic(EM)-leakage traces. The required number of traces can be reduced to 260 using a band-pass filtering technique. To the best of our knowledge, this is the first paper that describes extracting secret keys from a real LPWA device with less than 300 EM-leakage traces. Furthermore, the AES-128 key, except for the first four bytes for message authentication code generation, can be recovered with 140 EM-leakage traces and the remaining four bytes by a brute force search with 2^{32} computational complexity to recover the entire key.

2 RELATED WORK

Many studies have pointed out security issues, and attacks against LPWA protocols and have proposed solutions to improve security against such attacks. Most of them target the LoRaWAN protocol since it is an open technology and the specification is publicly available. Girard (Girard, 2015) pointed out the issues in key provisioning for end-devices. Zulian (Zulian, 2016) and Tomasin et al. (Tomasin et al., 2017) demonstrated the possibility of a replay attack against the join procedure in LoRaWAN due to the limitation in the variety of the DevNonce generated by an end-device, and theoretically and experimentally showed that random number generators in a real end-device are not secure. Na et al. (Na et al., 2017) argued that LoRaWAN was vulnerable to a similar replay attack and described countermeasures. Lee et al. (Lee et al., 2017) proposed a bit-flipping attack against an encrypted frame payload using AES-CTR and a countermeasure. Yang et al. (Yang et al., 2018) discovered several vulnerabilities of LoRaWAN and demonstrated five attacks: 1) replay attack leads to a selective DoS attack, 2) plaintext recovery attack, 3) malicious message modification, 4) falsification of delivery reports, and 5) battery exhaustion attack. Aras et al. (Aras et al., 2017) proposed a selective jamming attack against the LoRa physical layer and its countermeasure. Butun et al. (Butun et al., 2018) demonstrated five attacks: 1) RF jamming attack, 2) replay attack, 3) Beacon (Class B) synchronization attack, 4) network traffic analysis and 5) man-in-the-middle (MITM) attack against the latest version: LoRaWAN specification v1.1 released on Oct 11, 2017.

Side-channel analysis technologies to recover secrets are based on side-channel information such as sound, heat, timing information, power consumption, and electromagnetic-leakage. Some existing studies target IoT or resource-constrained devices. Kocher et al. (Kocher et al., 1999) were the first to propose a side-channel attack that leveraged a device's power consumption on a device and demonstrated that a DES key can be recover using the attack. Their attack includes a simple power analysis (SPA), differential power analysis (DPA), and higher-order DPA (HO-DPA). Messerges et al. (Messerges et al., 1999) theoretically derived the signal-to-noise ratio (SNR) in a DPA attack against DES proposed by Kocher et al. They improved the DPA to d-bit DPA by focusing on multiple bits in the S-Box of DES. Joye and Tymen (Joye and Tymen, 2001) proposed a DPA attack against the scalar multiplication on an elliptic curve-based cryptosystem (ECC), and Itoh et al. (Itoh et al., 2003) proposed a DPA attack focusing on the register address of an ECC. Brier et al. (Brier et al., 2004) were the first to study a correlation power analysis (CPA) based on a Hamming distance model. Komano (Komano et al., 2009) proposed a build-in determined sub-key CPA (BS-CPA) that finds a new sub-key by using the previously determined sub-keys recursively and demonstrated that it can recover a DES key with fewer power traces than the original CPA. Clavier et al. (Clavier et al., 2011) applied a CPA to first-order protected AES implementations and showed that the CPA requires fewer power traces than classical second-order DPA. Dinu and Kizhvatov (Dinu and Kizhvatov, 2018) demonstrated that a DPA can recover a partial AES-CCM key on a wireless microcontroller. Tawalbeh and Somani (Tawalbeh and Somani, 2016) evaluated the security of AES, ECC, and RSA against timing and fault side-channel attacks and showed countermeasures for IoT implementation. Moukarzel et al. (Moukarzel et al., 2017) proposed µLeech, a side-channel evaluation platform for IoT.

3 LoRaWAN PROTOCOL

We provide an overview of the LoRaWAN protocol. The end-device activation to a set AES-128 key for an end-device is described in Sect. 3.1, and data protection and integrity with AES-128 are described in Sect. 3.2.

3.1 End-device Activation

We have to personalize and activate the end-devices to connect them to a LoRaWAN network. There are two activation methods for an end-device: over-the-air activation (OTAA) and activation by personalization (ABP).

Over-the-Air Activation. In OTAA, an end-device must complete a join procedure to be able to make data exchanges with the network server. The join procedure requires the end-device to be personalized with a globally unique end-device identifier (DevEUI), an application identifier (AppEUI), and an AES-128 key (AppKey).

Activation by Personalization. End-devices can be activated by personalization (ABP). ABP directly associates an end-device to a LoRaWAN network without having to use the join procedure needed in OTAA.

Two session keys: NwkSKey and AppSKey, and DevAddr are stored in the end-device directly in ABP, while these keys are derived using the DevEUI, AppEUI, and AppKey in OTAA. The required information is preset to the end-device to connect a LoRaWAN network. Each end-device has a unique set of NwkSKey and AppSKey.

3.2 Data Protection and Integrity

Payload encryption using AES counter mode (AES-CTR) provides data protection of the frame payload for transmissions in the LoRaWAN protocol. AES-CMAC is used to generate a four-byte message integrity code (MIC) to maintain data integrity in payload transmissions and the OTAA procedure.

Data Protection. FRMPayload is encrypted before the MIC is calculated. The encryption key K depends on the FPort of the data message: If Fport is 0, then NwkSKey is used, and if Fport is in the range of 1, 2, ..., 255, then AppSKey is used. The encryption algorithm defines a sequence of blocks A_i. A block A_i contains one-byte `0x01`, followed by four-bytes `0x00000000`, one-byte direction field (Dir), four-byte identifier (DevAddr), four-byte FCntUp or FCntDown, one-byte `0x00`, and one-byte encoded i, or

$$A_i = \mathtt{0x01} \| \mathtt{0x00000000} \| \text{Dir} \| \text{DevAddr} \| \text{FCntUp or FCntDown} \| \mathtt{0x00} \| \mathsf{encode}(i).$$

Dir describes the direction field: 0 for uplink frames and 1 for downlink frames. The DevAddr identifies

Algorithm 1: Payload Encryption in LoRaWAN Protocol.

```
Input: FramePayload, Encryption key $K$
Output: EncrypredPayload
1 pld ← FRMPayload;
2 $k \leftarrow \lceil \mathsf{len}(\text{pld})/16 \rceil$;
3 for $i \leftarrow 1$ to $k$ do
4     $S_i \leftarrow$ AES-128($K$,$A_i$);
5 end
6 $S \leftarrow S_1 \| S_2 \| \ldots \| S_k$;
7 $T \leftarrow (\text{pld} \| \mathsf{pad}_{16}) \oplus S$;
8 EncryptedPayload ← First len(pld) bytes of $T$;
  /* Data protection using AES-CTR */
9 return EncryptedPayload;
```

the end-device in the current network. The frame counter FCntUp, incremented by end-devices, records the number of uplinks to the network server and FCntDown, incremented by the server, records the number of downlink frames from the server. Algorithm 1 shows the procedure of the payload encryption in detail. Note that the pad_{16} is a function that adds zero bytes so that the data length is a multiple of 16, and len returns the byte length of the data.

Data Integrity. All LoRa messages carry a PHY payload (Payload) consisting of one-byte MAC header (MHDR), a MAC payload (MACPayload), and a four-byte MIC. The MAC payload of the data messages starts with a frame header (FHDR) followed by an optional port field (FPort) and ends with an optional frame payload field (FRMPayload). The FHDR consists of the address of the end-device (DevAddr), a frame control byte (FCtrl), a frame counter (FCnt), and frame options (FOpts) to transport MAC commands. The MIC for payload calculated on the entire message is defined as

$$\text{msg} = \text{MHDR} \| \text{FHDR} \| \text{FPort} \| \text{EncryptedPayload}.$$

The block B_0 for the MIC calculation contains one-byte `0x49`, followed by four-bytes `0x00000000`, one-byte direction field (Dir), four-byte identifier (DevAddr), four-byte FCntUp or FCntDown, one-byte `0x00`, and one-byte len(msg), or

$$B_i = \mathtt{0x49} \| \mathtt{0x00000000} \| \text{Dir} \| \text{DevAddr} \| \text{FCntUp or FCntDown} \| \mathtt{0x00} \| \mathsf{len}(\text{msg}).$$

The MIC is calculated as

$$\text{MIC} = \text{AES-CMAC}(\text{NwkSKey}, B_0 \| \text{msg})[0 \ldots 3].$$

The join procedure in OTAA is started from an end-device by sending a join-request message. The join-request message contains AppEUI, DevEUI of the end-device, and a nonce of two bytes (DevNonce), or

$$\text{join-request msg} = \text{AppEUI}||\text{DevEUI}||\text{DevNonce}$$

AppEUI and DevEUI are a globally unique application ID of an end-device and an end-device ID in the IEEE EUI64 address space, respectively. DevNonce is a random value. The network server needs to keep the list of used DevNonce values for each end-device and ignores join requests with re-used DevNonce values to prevent replay attacks. The MIC for the join request in OTAA is calculated as

$$\begin{aligned}\text{MIC} = \text{AES-CMAC}(&\text{AppKey}, \text{MHDR}|| \\ &\text{join-request msg})[0\ldots3].\end{aligned}$$

4 KEY RECOVERY ATTACK

We propose a key recovery attack based on side-channel analysis, which applies to general LoRaWAN end-devices. The goal and assumptions are explained, and then the details of the attack are described.

4.1 Goal

The security of the LoRaWAN protocol is based on Advanced Encryption System (AES), a symmetric encryption algorithm. Data protection is ensured using AES-CTR, and message integrity is guaranteed by the computing of a MIC using AES-CMAC. Our key recovery attack thus targets AES-128 keys: AppSKey, NwkSKey, AppKey, stored in an end-device. AppSKey and NwkSkey are used to achieve data protection, and NwkSKey and AppKey are used to achieve integrity for data transmission and join messages, respectively. An attacker can decrypt or forge all messages and commands transmitted between the server and end-devices, or connect malicious end-devices to the LoRaWAN network by abusing these keys.

4.2 Assumptions

We assume an attacker as a security evaluator can access plaintext. The attacker does not have to control the plaintext and the corresponding ciphertext. This condition can be met if the attacker knows the data format and the data itself. For example, an end-device transmits recently measured temperatures periodically, and the attacker can guess the plaintext using a separate thermometer. Another way to meet the condition is to access an API for data transmission on an end-device. Some LoRaWAN libraries provide public APIs for data transmission that take plaintext messages or commands as input. Our side-channel key recovery attack is based on correlation power analysis that requires multiple pairs of plaintext and ciphertext. The attack is not applicable to fixed messages such as prefixed values in a protocol header since the Pearson correlation coefficient cannot be calculated. However, we can recover the keys and all the messages from the limited number of partial plaintext. In our experiments, we modify the source code of an end-device to set a trigger signal around the first round of AES-128. However, modification of the source code is not necessary if different EM-leakage traces can be appropriately aligned along the time axis.

The proposed attack is focused on the first round of AES-128, using the knowledge of the plaintext and guessing each byte of the AES-128 key independently. Guessing each byte of the first round key permits each byte of the output of the S-box to be recovered independently at the first round. The first round of AES-128 consists of four functions: AddRoundKey, SubByte, ShiftRow and MixColumn.

4.3 Key Recovery Attack

We now describe the key recovery attacks in detail. Our proposed attack consists of a leakage identification phase and key recovery phase.

4.3.1 Leakage Identification

The first phase of the attack is the identification of the EM-leakage traces produced by the execution of AES-128. The EM-leakage of one hundred executions with the same key and plaintext have been averaged. This permits an increase in the signal-to-noise ratio (SNR) defined as

$$\text{SNR} = \frac{P_{\text{AES-128}}}{P_{\text{Noise}}},$$

where $P_{\text{AES-128}}$ and P_{Noise} are respectively the power of AES-128 leakage and the noise. The noise P_{Noise} is considered to follow a Gaussian distribution $N(\mu, \sigma^2)$, which explains the increase in the SNR by averaging. The result of this recording is displayed in Fig. 1. This graph permits identify the ten rounds of AES-128 to be identified and we can manually delimit each round. This delimitation revealed a repetition of four events in each round (identified by four peaks) and corresponds to AddRoundKey, SubByte, ShiftRow and MixColumn. The x-axis represents time in terms of the number of samples, and the y-axis represents the electromagnetic range in volts.

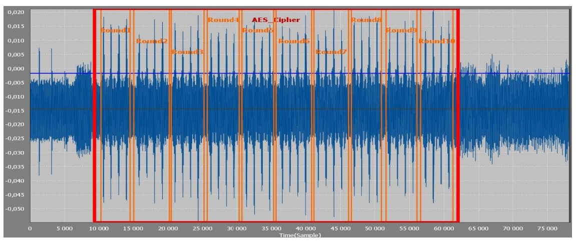

Figure 1: EM-leakage of hundred AES-128 executions.

Algorithm 2: Key Recovery Attack.

```
   Input: Plaintext $P_i$ $(0 \le i < Q)$
   Output: Guessed key $k^\star$
1  for $i \leftarrow 0$ to $Q-1$ do
2      for $d \leftarrow 0$ to $D-1$ do
3          $X^{d,i} \leftarrow$ EM-leakage of
             AES-128$(*, P_i)$;
4      end
5  end
6  for $i \leftarrow 0$ to $Q-1$ do
7      for $B \leftarrow 0$ to $16-1$ do
8          for $k \leftarrow 0$ to $256-1$ do
9              $Y_{i,k}[B] \leftarrow$
                 $\mathsf{HW}(\mathsf{SubByte}(P_i[B] \oplus k))$;
10         end
11     end
12 end
13 for $d \leftarrow 0$ to $D-1$ do
14     for $B \leftarrow 0$ to $16-1$ do
15         for $k \leftarrow 0$ to $256-1$ do
16             $r_{k,d}[B] \leftarrow \rho(Y_k[B], X^d)$;
17         end
18     end
19 end
20 for $B \leftarrow 0$ to $16-1$ do
21     $k^\star[B] \leftarrow \arg\max_{0 \le k < 256} \{\max_{0 \le d < D}\{r_{k,d}[B]\}\}$;
22 end
23 return $k^\star$;
```

4.3.2 Key Recovery

The second phase of the attack recovers the AES-key based on analysis of the EM-leakage traces. The key recovery attack uses correlation power analysis (Brier et al., 2004) focused on the output of the $\mathsf{SubByte}$ function in the first round. The leakage model applied is the Hamming weight (HW), which is justified by the fact that it is a software implementation. The following steps describe the procedure to compute the correlation between the EM-leakage and Hamming weight:

1. Record the EM-leakages during AES-128 encryption AES-128$(*, P_i)$ and store them to $X^{d,i}$, where $*$ is the unknown AES-128 key, P_i is the plaintext in ith trace, and $X^{d,i}$ $(0 \le d < D$ and $0 \le i < Q)$ is the d-th sample in the i-th trace out of Q traces of D samples.
2. Compute the 16-guessed distributions (one by key byte):
$$Y_{i,k}[B] = \mathsf{HW}(\mathsf{SubByte}(P_i[B] \oplus k))$$
for $0 \le k < 256$, $0 \le i < Q$, and $0 \le b < 16$ where $P_i[B]$ is the B-th byte of P_i, k is the guessed value of the key byte, and $Y_{i,k}[B]$ is a 16×256 distributions of Q elements.
3. Compare $X^{d,i}$ and all the 16×256 distribution $Y_{i,k}[B]$ using the Pearson correlation coefficient:
$$\begin{aligned} & r(k,d)[B] \\ &= \rho(X^d, Y_k[B]) = \frac{\mathrm{Cov}(X^d, Y_k[B])}{\sqrt{\mathrm{Var}(X^d)\,\mathrm{Var}(Y_k[B])}} \\ &= \frac{\sum_{i=0}^{Q-1}(X^{d,i} - \bar{X}^d)(Y_{i,k}[B] - \bar{Y}_k[B])}{\sqrt{\sum_{i=0}^{Q-1}(X^{d,i} - \bar{X}^d)^2 \sum_{i=0}^{Q-1}(Y_{i,k}[B] - \bar{Y}_k[B])^2}}, \end{aligned} \tag{1}$$
where
$$\bar{X}^d = \frac{1}{Q}\sum_{i=0}^{Q-1} X^{d,i} \text{ and } \bar{Y}_k[B] = \frac{1}{Q}\sum_{i=0}^{Q-1} Y_{i,k}[B].$$
for $0 \le d < D$, $0 \le b < 16$, and $0 \le k < 256$.

Algorithm 2 describes all the steps involved in the key recovery attack. The required number of traces to recover B-th byte of the key $N[B]$ is defined as the minimum q such that guessed key byte $k^\star[B]$ using q' traces is identical to that using q traces for all $q' > q$.

5 EXPERIMENTAL RESULTS

We describe the experimental result of our key recovery attack against a real LoRaWAN end-device. The experiment setup is sketched in Sect. 5.1 and results of a key recovery attack on the payload encryption key and MIC calculation key are shown in Sect. 5.2. Section 5.3 demonstrates a technique to reduce the number of EM-leakage traces to recover keys.

5.1 Experiment Setup

Our experiment used a LoRa Starter Kit. This starter kit is composed of two end-devices: an end-device with a plug-and-play LoRa module and an 868 MHz

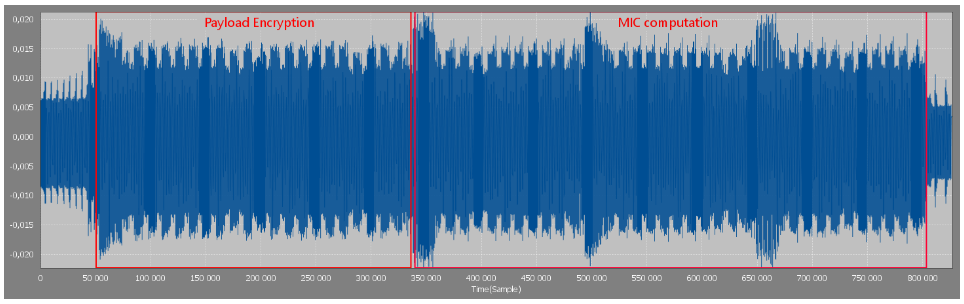

Figure 2: Identifications of the payload encryption and MIC computation.

antenna, and a gateway equipped with a LoRa module and an 868 MHz antenna. We used an API supplied by the end-device for our experiment to access to an implementation of AES-128. The source code of the program provided by the starter kit was modified to add a trigger signal at the first round of the AES.

5.2 Key Recovery

In our experiment, we targeted AppSKey for payload encryption and NwkSKey for MIC generation in the data transmission phase. The EM-leakage traces were recorded according to the following process.

1. The gateway generates a random plaintext P and send it to an end-device.
2. The end-device encrypts P using AppSKey and makes a ciphertext C.
3. The EM-probe gets the leakage information on AppSKey, and the oscilloscope records the information.
4. The end-device generates MIC for the payload containing C.
5. The EM-probe gets the leakage information on NwkSKey, and the oscilloscope records the information.
6. The end-device send a frame including C and MIC to the gateway.
7. Goto step 1 until a sufficient number of traces is captured.

In a key recovery attack, it is necessary to identify the distinct encryption phases in the EM-leakage traces to bring out the first round of AES-128 for the payload encryption or MIC computation. We can find patterns in the EM-leakage traces. Figure 3 show the measurement of the EM-leakage from a LoRaWAN end-device.

We can identify two distinct parts; the first part corresponds to the encryption of the application payload and the second part to the MIC computation. Figure 2 shows 20 similar patterns in the signal part

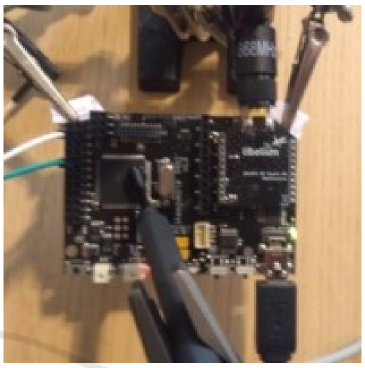

Figure 3: EM-leakage measurement from a end-device.

identified as the application payload encryption. The application payload is composed of 32 bytes, and AES-128 with ten rounds is executed twice; thus, 20 similar patterns appear. The same pattern can be identified three times in a row within the second part identified as the MIC computation. The MIC computation for 40-byte data executes AES-128 three times. Furthermore, inside each AES-128, the same pattern could be identified ten times corresponding to the number of rounds in a unique AES-128.

The key recovery attack is applied to the first round of AES-128. Figure 4 shows the result of the attack against the payload encryption process and plots $k^{\star}[0]$ with the highest correlation. These values with the highest correlation are identical to bytes of the AES-128 key, or $k^{\star}[B] = K[B]$ for all B. That is, the key recovery attack reveals all bytes of the key. In the sixteen traces of correlation, two peaks with an amplitude around 0.4 are identifiable. It suggests that the intermediate value $\{\mathsf{SubByte}(P[B] \oplus k^{\star}[B])\}$ $(0 \leq B < 16)$ is manipulated at least twice. The entire AES-128 key for the payload encryption key can be recovered with 350 electromagnetic (EM)-leakage traces. Table 1 shows the number of required traces $N[B]$ to recover each key-byte. On the other hand, 12 bytes of the AES-128 key for the MIC calculation can be recovered with 140 EM-leakage traces; however, the four bytes from the second byte to the fifth byte of the key never converge in the key recovery algorithm.

Table 1: Required number of traces to recover the key.

Byte (b)	Raw	De-noised	Improvement
0	140	90	-36%
1	220	130	-41%
2	200	80	-60%
3	310	120	-61%
4	130	70	-46%
5	200	260	+30%
6	150	110	-27%
7	260	110	-58%
8	230	210	-9%
9	320	180	-44%
10	350	130	-63%
11	230	200	-13%
12	180	200	+11%
13	80	80	± 0%
14	300	90	-70%
15	210	260	+23%
Maximum	350	260	-26%

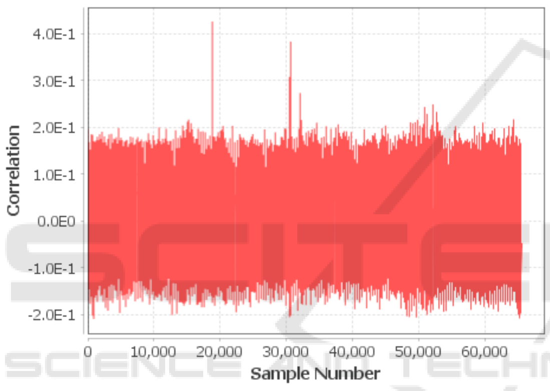

Figure 4: Correlation between the EM-leakage and Hamming weight for $k^{\star}[0]$.

The four bytes of the input to the first execution of AES-128 for the MIC calculation are DevAddr and constant. The variances of $\{Y_{i,k}[B]\}$ thus vanish for $0 \leq B < 4$, and we cannot compute the Pearson correlation coefficient in Eq. (1). One way to recover the four bytes is to use a brute force search with 2^{32} computational complexity. Alternatively, another leakage model, such as leakage during the computation of the MixColumn function, could be used .

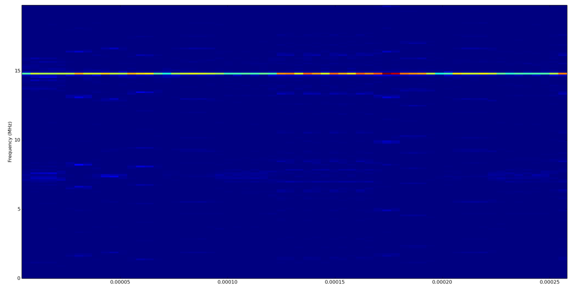

Figure 5: Spectrogram of an EM-leakage trace highlighting the activity of the microprocessor around 14-15 MHz.

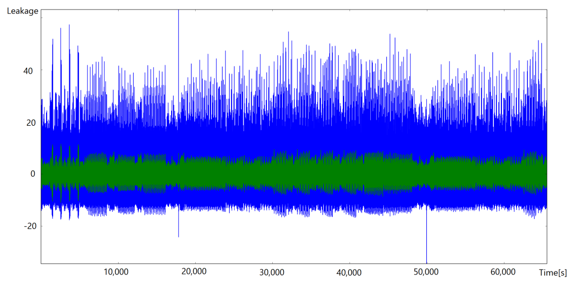

Figure 6: Band-pass filtered trace between 13 and 16 MHz in green and raw trace in blue.

5.3 Reduction of Required Traces

Uncorrelated noise produced by non-cryptographic circuits may increase the required number of EM-leakage traces. The targeted end-device has a frequency of 14 MHz. By computing the spectrogram of the recorded EM-leakage around this frequency, we obtain Fig. 5 where the colors gradient indicates the signal amplitude as a function of the time (x-axis) and of frequency (y-axis). This spectrogram shows activity around 14 to 15 MHz, which corresponds to the activity of the targeted microprocessor. We can thus apply a software band-pass filter between 13 and 16 MHz to remove low and high-frequency noise and improve the signal-to-noise ratio. Figure 6 illustrates the effect of the de-noising process, and a raw trace is plotted in blue and the associated de-noised trace in green. Furthermore, the de-noising procedure also facilitates visual analysis and sub-function separation that is both instrumental and suitable the side-channel analysis.

The application of band-pass filtering on the raw traces used in Sect. 5.2 improves the efficiency of the key recovery attack. We summarized the number of required traces $N[B]$ to recover each key-byte in Table 1 to compare both results. The column "improvement" shows the difference (as a percentage) between the number of traces required to recover each key byte with the raw traces and the de-noised trace. The filtering reduces the number of traces required to achieve the entire key by about 26%.

6 CONCLUSION

We applied a correlation power analysis to a real LoRaWAN end-device to recover secret keys used for the payload encryption process and message authentication code generation. The entire payload encryption key can be recovered with 350 EM-leakage traces, and 12 bytes out of the MIC calculation key can be recovered with 140 EM-leakage traces. The MIC calculation key can be completely recovered with 2^{32} computational complexity by specifying the remaining four bytes by a brute force search. Furthermore, we improved the efficiency of the key recovery attack by reducing the number of traces required to recover keys by 26% using a noise filtering technique. In our future work, we will try to recover the entire key for the MIC calculation by focusing on an AES-128 leakage during computation of the MixColumn function.

REFERENCES

3GPP (2016). Standardization of NB-IOT completed.

Aras, E., Small, N., Ramachandran, G. S., Delbruel, S., Joosen, W., and Hughes, D. (2017). Selective Jamming of LoRaWAN using Commodity Hardware.

Bradley, J., Barbier, J., and Handler, D. (2013). Embracing the Internet of Everything To Capture Your Share of \$14.4 Trillion .

Brier, E., Clavier, C., and Olivier, F. (2004). Correlation Power Analysis with a Leakage Model. In Joye, M. and Quisquater, J.-J., editors, *Cryptographic Hardware and Embedded Systems - CHES 2004*, volume 3156 of *Lecture Notes in Computer Science book series (LNCS)*, pages 16–29, Berlin, Heidelberg. Springer Berlin Heidelberg.

Butun, I., Pereira, N., and Gidlund, M. (2018). Analysis of LoRaWAN V1.1 Security: Research Paper. In *Proceedings of the 4th ACM MobiHoc Workshop on Experiences with the Design and Implementation of Smart Objects*, SMARTOBJECTS ’18, pages 5:1–5:6, New York, NY, USA. ACM.

Clavier, C., Feix, B., Gagnerot, G., Roussellet, M., and Verneuil, V. (2011). Improved Collision-Correlation Power Analysis on First Order Protected AES. In Preneel, B. and Takagi, T., editors, *Cryptographic Hardware and Embedded Systems – CHES 2011*, volume 6917 of *Lecture Notes in Computer Science book series (LNCS)*, pages 49–62, Berlin, Heidelberg. Springer Berlin Heidelberg.

Dinu, D. and Kizhvatov, I. (2018). EM Analysis in the IoT Context: Lessons Learned from an Attack on Thread. Cryptology ePrint Archive, Report 2018/076.

Girard, P. (2015). Low Power Wide Area Networks Security.

GSMA (2017). Long Term Evolution for Machines: LTE-M.

Howell, J. (2017). Number of Connected IoT Devices Will Surge to 125 Billion by 2030, IHS Markit Says.

Itoh, K., Izu, T., and Takenaka, M. (2003). Address-Bit Differential Power Analysis of Cryptographic Schemes OK-ECDH and OK-ECDSA. In Kaliski, B. S., Koç, ç. K., and Paar, C., editors, *Cryptographic Hardware and Embedded Systems - CHES 2002*, volume 2523 of *Lecture Notes in Computer Science book series (LNCS)*, pages 129–143, Berlin, Heidelberg. Springer Berlin Heidelberg.

Joye, M. and Tymen, C. (2001). Protections against Differential Analysis for Elliptic Curve Cryptography — An Algebraic Approach —. In Koç, Ç. K., Naccache, D., and Paar, C., editors, *Cryptographic Hardware and Embedded Systems — CHES 2001*, volume 2162 of *Lecture Notes in Computer Science book series (LNCS)*, pages 377–390, Berlin, Heidelberg. Springer Berlin Heidelberg.

Kocher, P., Jaffe, J., and Jun, B. (1999). Differential Power Analysis. In Wiener, M., editor, *Advances in Cryptology — CRYPTO’ 99*, volume 1666 of *Lecture Notes in Computer Science book series (LNCS)*, pages 388–397, Berlin, Heidelberg. Springer Berlin Heidelberg.

Komano, Y., Shimizu, H., and Kawamura, S. (2009). Built-in Determined Sub-key Correlation Power Analysis. Cryptology ePrint Archive, Report 2009/161.

Lee, J., Hwang, D., Park, J., and Kim, K.-H. (2017). Risk analysis and countermeasure for bit-flipping attack in LoRaWAN. In *2017 International Conference on Information Networking (ICOIN)*, pages 549–551.

LoRa Alliance^{TM} (2017). LoRaWAN^{TM}Specification v1.1.

Messerges, T. S., Dabbish, E. A., and Sloan, R. H. (1999). Investigations of power analysis attacks on smartcards. In *Proceedings of the USENIX Workshop on Smartcard Technology on USENIX Workshop on Smartcard Technology*, WOST’99, pages 17–17, Berkeley, CA, USA. USENIX Association.

Moukarzel, M., Eisenbarth, T., and Sunar, B. (2017). μLeech: A side-channel evaluation platform for IoT. In *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 25–28.

Na, S., Hwang, D., Shin, W., and Kim, K.-H. (2017). Scenario and countermeasure for replay attack using join request messages in lorawan. In *2017 International Conference on Information Networking (ICOIN)*, pages 718–720.

Sigfox (2017). Sigfox Technology Overview.

Tawalbeh, L. A. and Somani, T. F. (2016). More secure Internet of Things using robust encryption algorithms against side channel attacks. In *2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA)*, pages 1–6.

Tomasin, S., Zulian, S., and Vangelista, L. (2017). Security Analysis of LoRaWAN Join Procedure for Internet of Things Networks. In *2017 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pages 1–6.

Yang, X., Karampatzakis, E., Doerr, C., and Kuipers, F. (2018). Security Vulnerabilities in LoRaWAN. In *2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI)*, pages 129–140.

Zulian, S. (2016). Security threat analysis and countermeasures for LoRaWAN^{TM}join procedure. Master’s thesis, University of Padova.