Side-Channel Attacks on the Yubikey 2 One-Time Password Generator

David Oswald, Bastian Richter, and Christof Paar

Horst Görtz Institute for IT Security
Ruhr-University Bochum, Germany
david.oswald@rub.de, bastian.richter@rub.de, christof.paar@rub.de

Abstract. The classical way of authentication with a username-password pair is often insufficient: an adversary can choose from a multitude of methods to obtain the credentials, e.g., by guessing passwords using a dictionary, by eavesdropping on network traffic, or by installing malware on the system of the target user. To overcome this problem, numerous solutions incorporating a second factor in the authentication process have been proposed. A particularly wide-spread approach provides each user with a hardware token that generates a One-Time Password (OTP) in addition to the traditional credentials. The token itself comprises a secret cryptographic key that, together with timestamps and counters, is used to derive a fresh OTP for each authentication. A relatively new yet widespread example for an OTP token is the Yubikey 2 produced by Yubico. This device employs an open-source protocol based on the mathematically secure AES and emulates a USB keyboard to enter the OTP in a platform-independent manner. In this paper, we analyse the susceptibility of the Yubikey 2 to side-channel attacks. We show that by non-invasively measuring the power consumption and the electro-magnetic emanation of the device, an adversary is able to extract the full 128-bit AES key with approximately one hour of access to the Yubikey 2. The attack leaves no physical traces on the device and can be performed using low-cost equipment. In consequence, an adversary is able to generate valid OTPs, even after the Yubikey 2 has been returned to the owner.

Keywords: Yubikey, side-channel analysis, one-time passwords, hardware token, implementation attack, embedded systems security, hardware vulnerabilities

1 Introduction

Considering the steadily increasing risk due to, e.g., phishing and malware, normal authentication schemes like username and password are not sufficient anymore for high-security online, especially cloud-based services. Therefore, additional means to strengthen the authentication by introducing an additional "factor" are mandatory. A popular example of these techniques are OTPs generated by a hardware token. These tokens are common in high-security commercial

applications, but not for private use, often because of their high price and the need for additional sever infrastructure.

Attacks on two-factor authentication systems, most prominently the breach of RSA's SecurID system [5, 2], were until now mostly based on weaknesses in the cryptographic design of the protocol or the backend network. In contrast, attacks on the actual (hardware) implementation are assumed to have much higher requirements with respect to the capabilities of an adversary. Indeed, "classical" invasive attacks on modern devices use expensive equipment, e.g., microprobes or a Focused Ion Beam (FIB) that can only be operated by an experienced semiconductor engineer. However, in the past few years, side-channel attacks have been shown to be an effective method to non-invasively extract secrets from embedded cryptographic devices. Side-Channel Analysis (SCA) utilises information leaked via channels that were not intended by the developer, for example, via the power consumption or the electro-magnetic (EM) emanation. Often, these attacks can be carried out with relatively cheap equipment and without the need for a highly sophisticated lab.

Therefore, the question arises if OTP tokens are susceptible to these methods. In this paper, we use the example of the Yubikey 2, a USB-based device manufactured by Yubico Inc. [32]. As a side note, the reason why we chose the Yubikey 2 as our target is that we were contacted by a member of a large computer user's group that employs the Yubikey 2 for two-factor authentication. We are currently in the process of evaluating tokens of other vendors with respect to similar physical attacks

The Yubikey 2 differs from most other OTP tokens with its focus on simplicity and an open-source software backend. The question arises if high-security requirements can be fulfilled by such a low-cost device and how well the token protects the 128-bit AES key used for the OTP generation. . Yubico has several security-sensitive reference customers (that use the Yubikey, e.g., for securing remote access) listed on their website [27], for example, Novartis, Agfa, and U.S. Department of Defense Contractors. The U.S. Department of Defense Contractors even switched from RSA's SecureID system to the Yubikey [30], even though the Yubikey 2 is not certified for governmental standards.

1.1 Two-Factor Authentication

As mentioned above, the "normal" way of authentication by means of username and password is not sufficient in many cases. The credentials can often be obtained, e.g., by social engineering or due to protocol weaknesses (cf. [21] for a recent example). Thus, an additional security factor is needed. An established solution for this problem are OTPs. An OTP is generated by a hardware (or sometimes software) token and provided in addition to the normal credentials. The token generates a value which is valid for a single use, sometimes also only for a short period of time. Now, the user has to *know* the username and password and additionally has to *own* the token to successfully perform an authentication. The OTP is usually derived based on usage counters, timestamps, and a secret

key securely stored on the token, by, e.g., hashing or encrypting the respective values.

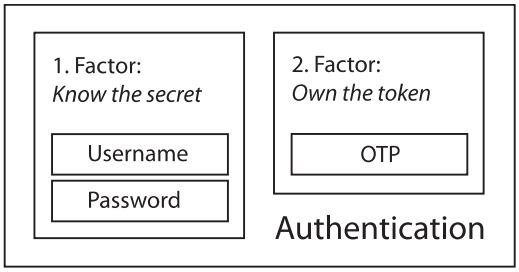


Fig. 1: Authentication with two factors

Of course, if an adversary manages to obtain both the physical token and the credentials of the user, he is able to gain unauthorised access. However, as soon as the token is, for instance, returned to the owner in order to conceal the attack, the adversary is no longer able to impersonate the rightful user.

1.2 Adversary Model

In this paper, we assume an adversary gaining physical access to the token for a limited amount of time (in the range of a few hours), e.g., when a user left his token at his desk. Besides, a token could also be stolen and returned without the owner noticing. Especially in the light of, for example, the attack on Lockheed Martin presumably being the motivation for the intrusion into RSA's network [5], this scenario is less hypothetical than it initially sounds. Organisations specialised in industrial espionage go to great lengths to overcome protection mechanisms, and obtaining a user's token for a limited amount of time seems conceivable.

In contrast to just using the token to login and then returning it, we focus on an attack that actually extracts the cryptographic secret from the device. This allows an adversary to create indistinguishable clones of the original device, usable for an unlimited amount of time. Apart from having direct access to the device, no modifications or invasive steps, e.g., the decapsulation of the token, are required. The SCA described in this paper is based on the non-invasive, passive observation of the token's behaviour and hence does not leave physical traces that can be detected later.

1.3 Side-Channel Attacks

A side-channel attack is usually performed in two steps. First, the adversary has physical access to the target device and acquires a side-channel signal (e.g.,

the power consumption or the EM emanation) during the cryptographic computation. This is repeated N times with different input data M_i, yielding N time-discrete waveforms $x_i(t)$ called *traces*. To recover the cryptographic key, the traces are then statistically processed in the evaluation phase, e.g., using the Pearson correlation coefficient when performing a Correlation Power Analysis (CPA) [4]. The adversary fixes a (small) subset $\mathcal{K}_{cand} \subseteq \mathcal{K}$ (e.g., the 256 possible 8-bit subkeys entering one S-box of the AES) and considers all key candidates $k \in \mathcal{K}_{cand}$. Then, for each $k \in \mathcal{K}_{cand}$ and for each $i \in \{0, \ldots, N-1\}$, a hypothesis $V_{k,i}$ on the value of some intermediate (e.g., the output of one 8-bit AES S-box) is computed. Using a power model f, this value is then mapped to $h_{k,i} = f(V_{k,i})$ to describe the process that causes the side-channel leakage. In practice, a Hamming Weight (HW) or Hamming Distance (HD) power model is often suitable for CMOS devices like Microcontrollers (µCs) [15]. In order to detect the dependency between $h_{k,i}$ and $x_i(t)$, the correlation coefficient $\rho_k(t)$ (for each point in time t and each key candidate $k \in \mathcal{K}_{cand}$) is given as

$$\rho_k(t) = \frac{cov(x(t), h_k)}{\sqrt{var(x(t))\, var(h_k)}}$$

with $var(\cdot)$ indicating the sample variance and $cov(\cdot, \cdot)$ the sample covariance according to the standard definitions [26]. The key candidate $\hat{k}$ with the maximum correlation $\hat{k} = \arg\max_{k,t} \rho_k(t)$ is assumed to be the correct secret key. When for instance attacking an implementation of the AES, this process is performed for each S-box separately, yielding the full 128-bit key with a much lower complexity of $\mathcal{O}\left(16 \cdot 2^8\right)$ compared to $\mathcal{O}\left(2^{128}\right)$ for an exhaustive search.

1.4 Related Work

Beginning with the first paper on Differential Power Analysis (DPA) published in 1999 [12], a multitude of methods for SCA has been introduced, for example, CPA [4] or the use of the EM emanation instead of the power consumption [1]. A comprehensive overview on the field of side-channel attacks is given in [15].

However, until 2008, there was no report of a successful side-channel attack on a real-world system. This changed with the break of the KeeLoq hopping code scheme [8]. Subsequently, several wide-spread products were attacked by means of SCA, e.g., the Mifare DESFire MF3ICD40 contactless smartcard [19] or the bitstream encryption schemes of Xilinx and Altera Field Programmable Gate Arrays (FPGAs) [16–18].

The security of–today heavily outdated–USB tokens was analysed in [11], describing hardware and software weaknesses but not covering side-channel attacks. In [10], it is stated that newer devices are harder to attack and that a "lunchtime attack [is] likely not possible". For the SecurID tokens manufactured by RSA, there are reports on both attacks on the backend [5] and flaws on the protocol level [2]. However, the real-world relevance of the latter attack is denied by RSA [7].

The cryptanalytical security of parts of the protocol used for the Yubikey was analysed in [25], and no severe formal vulnerabilities were found. Yubico mentions the threat of side-channel attacks in a security evaluation on their website [31], however, apparently did not further investigate this issue.

1.5 Contribution and Outline

The remainder of this paper is organised as follows: in Sect. 2, we describe the OTP generation scheme and analyse the underlying hardware of the Yubikey 2. The measurement setup for automatically acquiring power consumption and EM traces for our SCA is presented in Sect. 3. In Sect. 4, we detail on the initial side-channel profiling of the Yubikey 2, leading to the full-key recovery attack shown in Sect. 5. We conclude in Sect. 6, discussing suitable countermeasures and describing the reaction of the vendor Yubico, which we informed ahead of time as part of a responsible disclosure process.

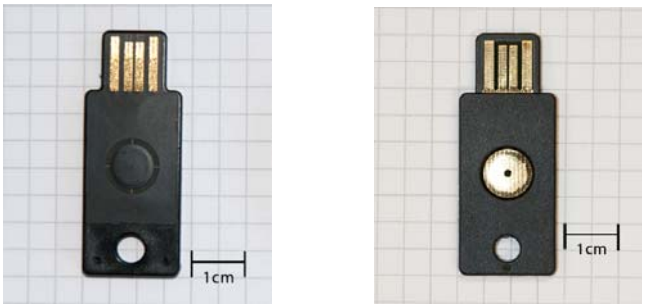
The novelty of this paper is the practical application of side-channel attacks in the context of authentication tokens. We demonstrate that physical attacks on such tokens can be used to extract secret keys and thus allow an adversary to duplicate the second authentication factor. Since the attacks in this paper were conducted with a relatively low-cost setup and mainly required experience in the field of SCA, it is likely that well-funded organisations could reproduce (or have already developed) similar techniques. Thus, we emphasize the need for additional countermeasures in the backend and system design, e.g., the use of key diversification et cetera.

2 The Yubikey 2

In this paper, we analyse the current version 2 of the Yubikey Standard (in the following sometimes referred to as Device Under Test (DUT)) with the firmware version 2.2.3. The predecessor Yubikey 1 (cf. Fig. 2a) was introduced in 2008 but already replaced by the current Yubikey 2 (Fig. 2b) in 2009 [32]. Apart from the Yubico-specific OTP generation, the Yubikey 2 can also be used to store a static password. Besides, the Yubikey 2 can be used as a token for generating HMAC-based One Time Password (HOTP) specified by the Initiative of Open Authentication (OATH) [27]. However, we do not further examine these additional features in this paper and focus on the default OTP mechanism.

2.1 Typical Use

Both the Yubikey 1 and the Yubikey 2 appear as a normal USB keyboard to the user's computer to enable direct input of the OTP. An advantage of this technique is that it does not require an extra driver installation and works with default keyboard drivers available on virtually every relevant operating system. When the user presses the button on top of the DUT, the token generates a OTP, encodes it in a specific format described in Sect. 2.2, and enters it using



(a) Yubikey 1 (b) Yubikey 2

Fig. 2: The two versions of the Yubikey Standard

simulated keyboard inputs. The intended way of using the OTP is depicted in Fig. 3. The user first enters his credentials and then gives focus to an additional input field on the login form before pressing the Yubikey's button.

Fig. 3: Typical Yubikey login form.

2.2 OTP Structure

The OTP generated by the Yubikey 2 is based on several counters, random bytes, a secret ID, and a checksum which are concatenated to a 16-byte value and subsequently encrypted using the AES with a 128-bit key.

UID The private ID is 6 byte long and kept secret. It can be used as another secret parameter or to distinguish users when a common encryption key is used.

useCtr The non-volatile usage counter is 2 byte long and increased at the first OTP generation after power-up. Additionally, the counter is incremented when the session usage counter wraps from `0xff` to `0x00`.

tstp The 3-byte timestamp is initialized with random data on startup. After this, the timestamp is incremented with a frequency of approximately 8 Hz.

sessionCtr The 1-byte session counter starts with `0x00` at power-up and is incremented on every OTP generation.

rnd 2 additional byte of random data.

crc A 2-byte CRC16 checksum computed over all previous data fields.

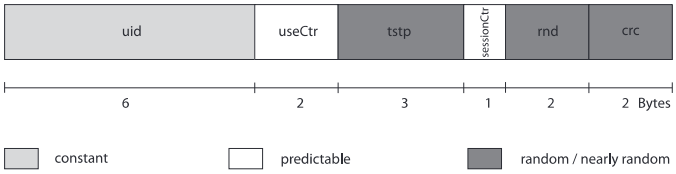


Fig. 4: Structure of a Yubikey OTPs

Figure 4 gives an overview of the structure of the OTPs and indicates which fields are static, predictable, or random.

All data fields are concatenated and then AES-encrypted using the secret 128-bit key programmed into the Yubikey 2. Usually, this key is set once by, e.g., the system administrator using the configuration utility [28] before the Yubikey 2 is handed to the user. The resulting ciphertext of the AES encryption is encoded using a special encoding called "Modhex" to avoid problems with different keyboard layouts by limiting the simulated keypresses to alphanumeric characters that have the same keycode in most locales. To identify the Yubikey, a Modhex-encoded 6-byte public ID is prepended to the encoded ciphertext.

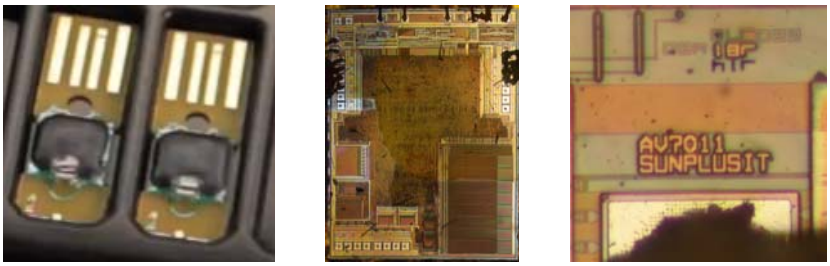
To verify the OTP, the server-side software, e.g., the open-source validation server provided by Yubico, undoes the Modhex encoding, retrieves the AES key stored for the respective public ID, decrypts the OTP, and validates the resulting data. More precisely, the following steps are performed for the verification of an OTP:

1. Identify the Yubikey by the public ID and retrieve the corresponding AES key
2. Decrypt the OTP with the corresponding AES key
3. Check the CRC checksum
4. Check if the private ID is correct
5. Compare the counters to the last saved state:
   - Accept the OTP if the session counter has been incremented or
   - If the session counter has not been incremented, accept the OTP if the usage counter has increased

If the OTP does not meet one of the previous conditions, it is considered invalid and the authentication fails. The conditions for the counters are in place to avoid replay attacks.

2.3 Hardware of the Yubikey 2

The Yubikey 2 is mono-block molded and thus hermetically sealed. To find out which kind of µC is used in the Yubikey 2, we dissolved the casing with fuming nitric acid to gain access to the silicon die (cf. Fig. 5a). The position of the µC was known from a promotional video about the production of the Yubikey [29], from which we extracted the picture of the Printed Circuit Board (PCB) shown in Fig. 5a. On the die, we found the label "SUNPLUSIT" (cf. Fig. 5c) which seems to belong to Sunplus Innovation Technology Inc. based in Taiwan [24]. We were unable to exactly find out which controller was used, as there is no Sunplus part related to the label "AV7011". However, all Human Interface Device (HID) µCs produced by Sunplus employ an 8-bit architecture. This fact is important when searching for a suitable power model for the SCA.



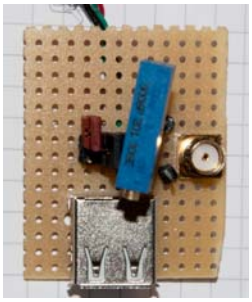
(a) PCB from [29] (b) Complete die (c) Die label

Fig. 5: Die of the µC in the Yubikey 2

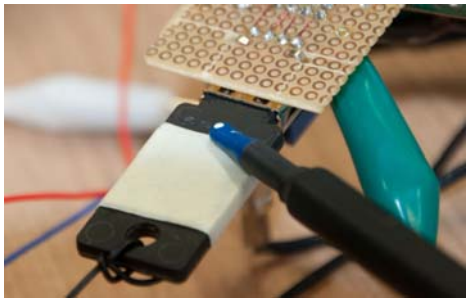
3 Measurement Setup

To record power traces for an SCA, we built a simple adaptor to get access to the USB power and data lines, cf. Fig. 6a. Note that the developed measurement adaptor is not specific for the Yubikey, but can be used in general for power measurements of USB devices. The basic setup gives simple access to the USB lines and provides a pin to insert a shunt resistor for power measurements. The `D+` and `D-` lines are directly connected to the PC's USB port. A 60 Ω resistor was inserted into the ground line to measure the power consumption of the Yubikey.

In our first experiments, we used `Vcc` provided by the USB port as power supply for the Yubikey, however, this resulted in a high amount of measurement noise. Therefore, an external power supply was added to reduce the noise



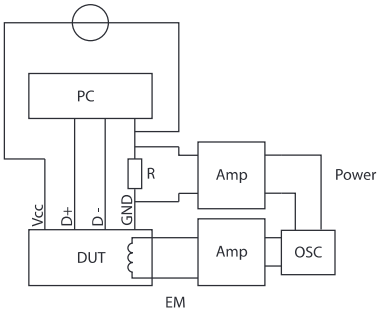
(a) USB adaptor



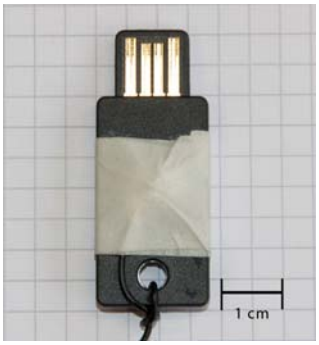
(b) EM probe

Fig. 6: Measurement methods: USB adaptor with shunt resistor and EM probe at the position with maximal signal amplitude on the Yubikey 2

caused by the PC's power supply. Figure 7a depicts the overall structure of the measurement setup.



(a) Schematic



(b) Yubikey 2 with wire for simulating button presses

Fig. 7: Setup for measuring the power consumption and EM emanation

A custom amplifier was added to amplify the measured voltage drop over the resistor. This was necessary because the measured (unamplified) voltage was too low to fill the minimal input range of ±100 mV of the utilized oscilloscope, a Picoscope 5204 Digital Storage Oscilloscope (DSO) [22]. All measurements were recorded at a sample rate of 500 MHz. Initially, to perform the profiling of the DUT described in Sect. 4, we focused on the power consumption measured via the shunt resistor. However, in subsequent experiments and for improving the key recovery described in Sect. 5, we also recorded the EM emanation of the DUT by placing a commercially available near-field probe [13] on an experimentally

determined position on the package of the Yubikey 2. The resulting signal was amplified by 30 dB using an amplifier made by Langer EMV [14]. The EM probe on the casing to the Yubikey is depicted in Fig. 6b. The overall cost for the setup used in this paper is approximately $ 3000. Hence, the attack described in Sect. 5 can be performed at low cost without sophisticated, expensive lab equipment.

3.1 Controlling the Yubikey

To initiate an encryption on the Yubikey, a capacitive button on top of the token has to be pressed. This button is basically a open plate-type capacitor whose capacitance changes when a finger is placed on top. For our purposes of automatic measurements, manually pressing the button is not an option. However, the finger can be "simulated" by connecting the blank metal contact on top of the Yubikey 2 to ground. For this purpose, we used a MOSFET transistor controlled by an ATXMega µC. The Yubikey with the controlling wire is depicted in Fig. 7b. Note that this setup is not fully stable. This can lead to false button presses or failures to press the button at all. Thus, the measurement software was prepared to handle these problematic cases.

4 Side-Channel Profiling

The data acquisition of the DSO was triggered using a large drop within the power consumption of the device caused by the status LED of the Yubikey being turned off. A level dropout trigger–firing when the signal has been below a certain level for a defined period of time–was employed. Note that the DUT needs at least 2.6 seconds to "recover" after a button press. Incidentally, this significantly slows down the measurement process (and thus the overall attack) because the speed of the data acquisition is limited by this property of the Yubikey.

There are glitches regularly occuring in the power traces. These glitches are apparently generated by the DUT and do not occur when simply measuring the supply voltage without the DUT being connected. They follow a constant interval of 1 ms, but do not have a constant offset to the voltage drop. Because of this, they might be caused by the USB Start Of Frame (SOF) packets that are sent by the PC in an 1 ms interval actively polling the DUT. These glitches turned out to be problematic because they have a large influence on the amplitude of the trace and disturb the statistical methods used in the subsequent analysis. In order to solve this problem, a MATLAB function was developed to detect these wide glitches and discard the respective power trace. As a result, the effective number of power traces usable for SCA is approximately 65 % of the overall number of recorded traces.

4.1 Locating the AES Encryption

When initially examining the power trace of the DUT, the significant voltage drop caused by shutting off the LED was used as a reference point. Right before

the voltage drop, a pattern can be observed that resembles a structure with ten rounds, each approximately 200 µs long, cf. Fig. 8.

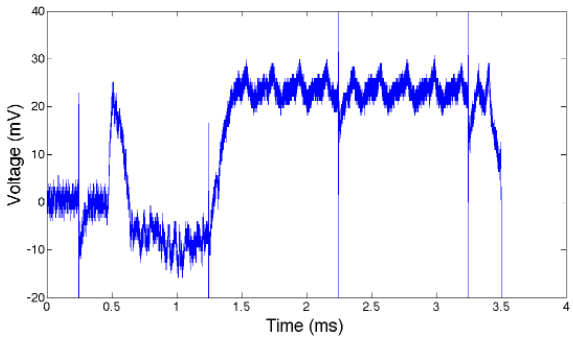


Fig. 8: Ten-round pattern in the power traces before the LED being shut off

Since the AES-128 employed on the Yubikey has ten rounds, it is likely that this part of the trace belongs to the AES encryption. This is further confirmed by Fig. 9 showing an average trace computed using 1000 power traces. The "rounds" are clearly visible, and even different operations are distinguishable within one round. Note that, however, we were unable to observe single instructions within one round, rather, it appears the traces are in some way low-pass filtered. This may, for instance, be due to a voltage regulator of the µC or decoupling capacitors. Additionally, the tenth round at approximately 2.1 ms is 70 µs shorter than the others, which agrees with the fact that the final round of the AES algorithm misses the `MixColumns` step.

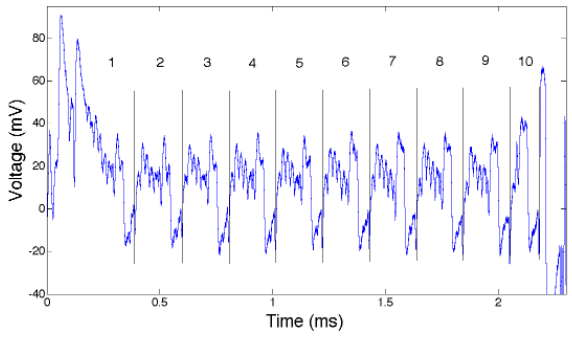


Fig. 9: Average over 1000 amplified traces of the part suspected to belong to the AES encryption

We recorded 20,000 traces of the part presumably belonging to the AES operation. The 128-bit AES key was set to `ad 5c 43 c5 2f 25 a7 4a 94 41 c2 1f 35 5b 43 09`. The used sample rate was 500 MHz as mentioned in Sect. 3. Experimentally, we found that (digitally) downsampling the traces by a factor of ten does not affect the success rate of the subsequent attack presented in Sect. 5. Hence, to reduce the data and computation complexity, all experiments described in the following included this pre-processing step.

We tested different models for the power consumption of the device. An 8-bit HW model for single bytes of the intermediate values within the AES turned out to be suitable, confirming the assumption that an 8-bit µC is used in the Yubikey 2. To identify the different AES operations within the rounds, a CPA using the HW of certain output bytes of the S-boxes in round nine and of certain input bytes to round ten as the power model (cf. Sect. 1.3) was performed. The correlation results (after 6,400 power traces) can be exemplarily seen for byte 13 and 16 in Fig. 10.

The horizontal blue lines at ±0.05 indicate the expected "noise level" of $4/\sqrt{\#traces}$. A correlation exceeding this boundary is considered significant, i.e., means that the DUT performs a computation involving the predicted value (in this case state bytes in round nine and ten) at the respective point in time. The rationale for this condition is given in [15]: For wrong predictions, the correlation coefficient follows a Gaussian distribution with standard deviation $\sigma = 1/\sqrt{\#traces}$. 99.99% of the samples taken from this distribution are within $\pm 4\sigma$, which yields the boundary of $4/\sqrt{\#traces}$.

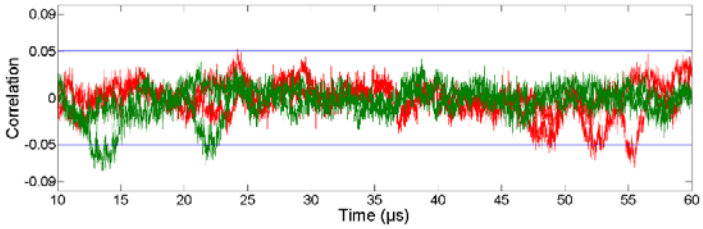


Fig. 10: Correlation for byte 13 and 16, HW of the S-box output in round nine (green, 10 . . . 25 µs) and HW of the input to round ten (red, 50 . . . 60 µs) using 6,400 traces

4.2 EM Measurements

As mentioned in Sect. 3, we also captured the EM emanation of the DUT at the same time as the power consumption in subsequent experiments. The EM traces mainly showed a clock signal at a frequency of 12 MHz. However, digitally amplitude-demodulating [23] this signal yielded a trace not exhibiting the low-pass filtered shape observed for the power consumption traces. Figure 11 depicts

a power consumption trace (blue, bottom) and the corresponding demodulated EM trace (green, top). In both cases, the round structure is discernible. Yet, the EM trace allows to separately observe every clock cycle, while the power consumption trace only shows the overall round structure.

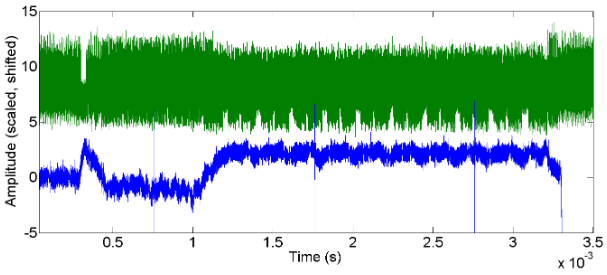


Fig. 11: Power consumption trace (blue, bottom) and demodulated EM trace (green, top). Vertical scaling and offset changed to compare general signal shape

Similar to the power consumption traces, we also observed distorted EM traces. However, the overall number of "usable" traces was higher compared to the power consumption measurement: only 25 % of the EM traces had to be discarded, compared to about 35 % for the power consumption traces.

5 Practical Attack: Extracting the AES Key

Having analysed the round structure and identified the points in time when the leakage occurs, we continued with trying to recover the secret AES key. Initially, we used the power traces, but switched to EM traces later to reduce the number of required measurements and thus the time needed for the attack.

5.1 Key Recovery using Power Consumption Traces

We computed the correlation coefficient for all 256 candidates for each key bytes using 10,000 traces. The hypothetical power consumption h_i (cf. Sect. 1.3) was computed as $h_i = HW\left(SBOX^{-1}\left(C_i \oplus rk\right)\right)$, with C_i a ciphertext byte (for measurement i) and rk the corresponding byte of the round key (dropping the byte index for better readability). The correlation coefficients for the first, second, eighth and ninth key byte are exemplarily shown in Fig. 12. Evidently, the maximum absolute value for the correlation coefficient occurs for the correct key candidate. This observation also holds for the remaining bytes, for which the results are not depicted in Fig. 12.

To get an estimate of how many traces are needed to clearly distinguish the correct key candidate from the wrong ones, the maximum correlation coefficient

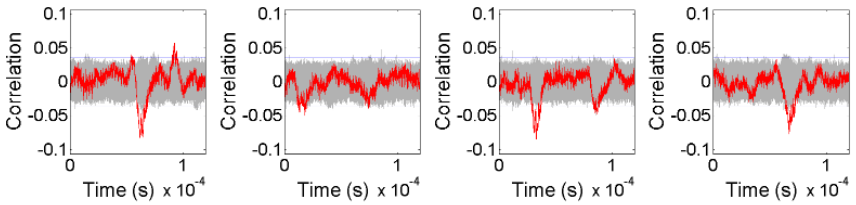


Fig. 12: Correlation coefficient for all candidates for the key bytes 1, 2, 8, and 9 (left to right) after 10,000 traces. Red: correct key candidate, gray: wrong key candidates

(at the point of leakage) for each candidate after each trace was saved. The result after 10,000 traces is exemplarily depicted in Fig. 13 for the first, second, eighth and ninth key byte. The maximum correlation for all key bytes is shown in Fig. 15 in Appendix 6.2.

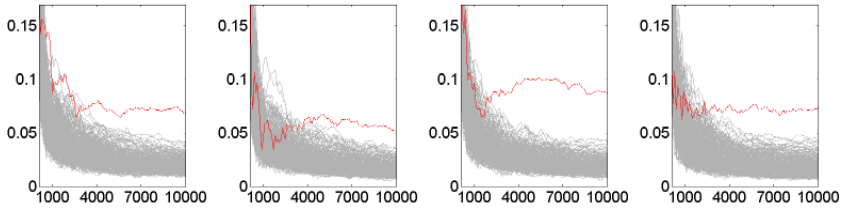


Fig. 13: Evolution of the maximum correlation (vertical axis) over the number of used traces (horizontal axis) for key bytes 1, 2, 8, and 9 (left to right). Red: Correct key candidate

To estimate the number of traces needed to recover the key, we used the ratio between the maximum correlation for the correct key candidate and the highest correlation for the "second best" wrong candidate as a metric, cf. for instance [20]. We then used the number of traces for which this ratio is greater than 1.1 as the minimum number of required traces given in Table 1.

We were able to clearly determine the full 128-bit AES key using approximately 4,500 traces. In this regard, it turned out that the number of traces needed to recover a key byte differs: For byte 1, 4, and 16, less than 1,000 traces were sufficient. For byte 8, 9, 10, 11, 13, and 14, less than 3,000 traces sufficed to determine the correct value. For byte 2, 3, 5, 6, 7, 12, and 15, a number between 3,100 and 4,500 traces lead to the correct key byte being found.

Note that the pre-selection of the traces necessary due to the glitches mentioned in Sect. 4 effectively requires more traces to be recorded: for 4,500 usable traces, approximately 7,000 traces had to be measured in total. With our current

Key byte	1	2	3	4	5	6	7	8
# Required traces	700	4,400	3,300	200	4,100	4,200	4,300	2,200
Key byte	9	10	11	12	13	14	15	16
# Required traces	2,800	2,100	2,300	4,500	1,400	1,100	3,100	500

Table 1: Approximate number of required traces to recover respective bytes of the AES key using power consumption traces. Metric: Ratio between correlation for correct key candidate and second highest correlation greater than 1.1

measurement setup, 1,000 traces can be acquired in about 1.5 h, i.e., at a rate of 11.1 $^{\text{traces}}/_{\text{min}}$. Thus, to obtain 7,000 traces in total, approximately 10.5 h of access to the DUT were necessary.

The "spread" correlation peak with a width of 8.3 µs would translate to a clock frequency of approximately 120 kHz. At this clock frequency, the execution time of about 2.5 ms (cf. Fig. 9) would imply that the AES is performed in only 300 clock cycles. Considering that even highly optimized AES implementations require about 3,000 cycles on similar (and probably more powerful) 8-bit µCs [3], it appears that the leakage is distributed over several clock cycles, presumably due to the low-pass characteristic mentioned in Sect. 4. Hence, we continued our analysis using the EM traces that give a higher resolution in this regard.

5.2 Key Recovery using EM Traces

We performed the identical attack as in Sect. 5.1 on the (digitally demodulated) EM traces. The resulting correlations after 800 traces for all candidates for the first, second, eighth and ninth key byte are exemplarily shown in Fig. 14. In contrast to the power consumption traces, the correlation for the correct key candidate clearly exceeds the one for the wrong candidate after already less than 1,000 traces. Besides, the correlation peak is limited to a short instant of approximately 160 ns, which corresponds to a clock frequency of about 6.25 MHz. Thus, it is likely that this correlation is for one or a few instructions of the µC only.

Again, we estimated the number of required traces to recover respective key bytes in analogy to Sect. 5.1. The results are given in Table 2. Figure 15b in Appendix 6.2 shows the evolution of the maximum correlation which was used to derive the numbers given in Table 2.

As evident in Table 2, a maximum number of 500 traces is sufficient to fully recover the 128-bit AES key. Due to approximately 25 % of the EM traces being unusable, this translates to an overall number of 666 traces. Thus, only 1 h of access to the Yubikey 2 is sufficient to recover the key with EM measurements, compared to 10.5 h that would be required when using the power consumption traces.

Besides, a tradeoff between computation time and the number of traces could be applied. An adversary may, for instance, decide to only record 300 traces, so

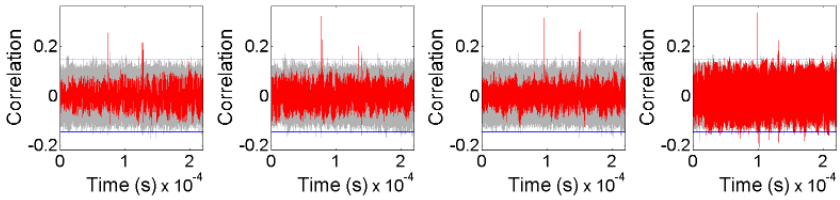


Fig. 14: Correlation coefficient for all candidates for the key bytes 1, 2, 8, and 9 (left to right) after 800 traces. Red: correct key candidate, grey: wrong key candidates

Key byte	1	2	3	4	5	6	7	8
# Required traces	400	300	400	200	300	200	300	200
Key byte	9	10	11	12	13	14	15	16
# Required traces	200	200	200	200	300	500	300	300

Table 2: Approximate number of required traces to recover respective bytes of the AES key using EM traces. Metric: Ratio between correlation for correct key candidate and second highest correlation greater than 1.1

three key bytes (1, 3, and 14) would not be (fully) recoverable. However, these remaining three bytes, i.e., 24 bit, could be easily determined using an exhaustive search on a standard PC within minutes. In this case, the measurement time is reduced to 36 min for effectively 400 traces in total.

6 Conclusion

Using a non-invasive side-channel attack, we are able to extract the full 128-bit AES key stored on a Yubikey 2 with approximately 500 EM traces. The necessary equipment has a cost of less than $ 3000 in total. Given the AES key, an adversary is able to generate an arbitrary number of valid OTPs and thus to impersonate the legitimate owner given that the traditional credentials have been obtained, e.g., by means of eavesdropping, phishing, or malware. To acquire the required number of traces, an adversary needs less than one hour of physical access to the Yubikey. Thus, the attack could for instance be carried out during the lunch break.

Note that a standard CPA was sufficient to mount our attack with a number of traces small enough to pose a threat in the real world. Hence, we did not further investigate more complicated (profiled) SCA techniques like template attacks [6]. Such methods could further reduce the number of required traces, however, come with additional difficulties due to the need for a separate training device, cf. for instance [9]. Hence, we decided to use the more “robust” CPA, an approach that turned out to be sufficient in this specific case.

The attack leaves no physical traces on the DUT. The only means by which the attack could be detected is a (relatively high) increase of the usage counters, cf. Sect. 2.2. Due to the fact that the volatile session counter has to reach 256 first before the non-volatile usage counter is incremented, the EM-based attack only increases the usage counter by two when recording 500 traces. Thus, the presented attack does not lead to a "suspicious" change of this counter and is very unlikely to be detected in this way.

6.1 Countermeasures

To mitigate the consequences of the attack described in this paper, countermeasures both on the hardware level and for the (organisation of the) backend should be implemented. In this regard, as part of the process of responsible disclosure, we discussed feasible approaches with the vendor Yubico.

In general, the Yubikey should of course always be treated as a second factor and never be used as the sole means of authentication. Secondly, it should be ensured that no two Yubikeys have the same AES key. Otherwise, obtaining the AES key from one device would render all other devices with the same key insecure as well. Using only the 6-byte private ID mentioned in Sect. 2.2 to distinguish Yubikeys is hence not advisable in our opinion. Besides, especially for sensitive applications, users should be trained to keep their Yubikey with them at all times and report lost or stolen devices instantly so that they can be blocked and replaced.

On the level of the hardware and embedded software of the Yubikey 2, specific countermeasures against SCA can be realised: established techniques, for instance, randomising the execution order and the timing by shuffling the S-boxes and inserting dummy operations [15] are likely to make the presented attack much more difficult and to considerably increase the number of required traces. This in turn would reduce the threat posed by the attack: the longer the device has to be in the hands of the adversary, the more likely it is that the attack is noticed by the legitimate user. Due to the limitations of the 8-bit µC used on the Yubikey 2, it is unclear whether SCA countermeasures such as masking that involve a higher space and time overhead can be implemented.

One interesting alternative–especially for high-security applications–is the Yubikey Neo also produced by Yubico [33]. Instead of a standard µC, the Yubikey Neo employs a Common Criteria certified smartcard controller that was specifically designed to withstand implementation attacks and thoroughly tested in this regard. In our opinion, to protect sensitive services and data, the double price of $ 50 compared to $ 25 for the Yubikey 2 may be a reasonable investment.

6.2 Reaction of the Vendor

Having discovered the security problem, before publication, we contacted the vendor Yubico as mentioned before. Yubico acknowledged our results and has taken measures to mitigate the security issues. We examined an updated firmware (version 2.4) and found that our attacks do not apply to this improved version.

Several attempts to circumvent the new mechanisms implemented by the vendor were unsuccessful. Thus, the resistance of the DUT against SCA seems to have increased significantly. This likely rules out low-complexity attacks (in terms of the equipment and the required time for the measurements) as presented in this paper. The following statement summarizes the reaction of the vendor Yubico:

> "Yubico takes security seriously and we welcome analysis of our products, and are happy to engage on a technical basis for the benefit of our customers. While the YubiKey Standard was not intended to resist physical attacks, we aspire to exceed expectations. After being informed about preliminary results, we worked with the research team to implement mitigations. We have incorporated this in our currently manufactured product. We wish to stress that the YubiKey NEO and the YubiKey Standard used in OATH or challenge response mode is not affected. We look forward to continue work with researchers and improve our products."

Acknowledgments

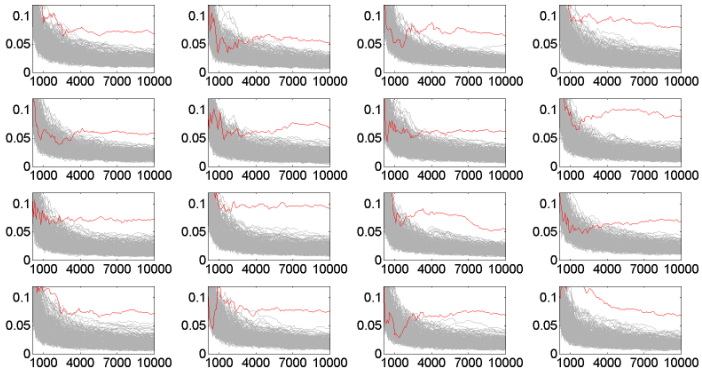
We would like to thank Christoph Wegener for his remarks and contributions in the course of our analysis.

References

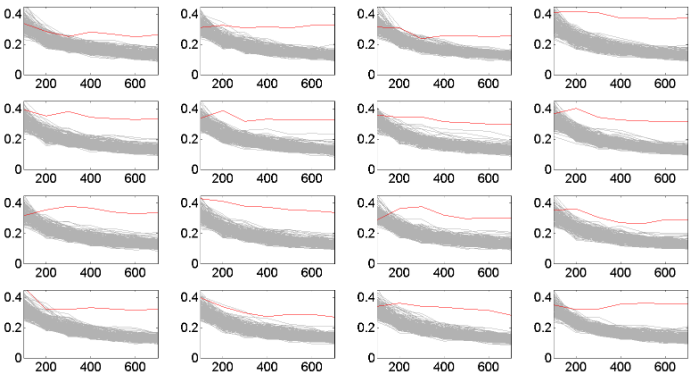
1. D. Agrawal, B. Archambeault, J. R. Rao, and P. Rohatgi. The EM Side-Channel(s). In *Cryptographic Hardware and Embedded Systems – CHES 2002*, LNCS, pages 29–45. Springer, 2003.
2. R. Bardou, R. Focardi, Y. Kawamoto, L. Simionato, G. Steel, and J.-K. Tsay. Efficient padding oracle attacks on cryptographic hardware. In R. Safavi-Naini and R. Canetti, editors, *Advances in Cryptology - CRYPTO 2012*, volume 7417 of *LNCS*, pages 608–625. Springer, 2012.
3. J. W. Bos, D. A. Osvik, and D. Stefan. Fast Implementations of AES on Various Platforms. *IACR Cryptology ePrint Archive*, 2009:501, 2009.
4. E. Brier, C. Clavier, and F. Olivier. Correlation power analysis with a leakage model. In *CHES 2004*, 2004.
5. P. Bright. RSA finally comes clean: SecurID is compromised, June 2011.
6. S. Chari, J. R. Rao, and P. Rohatgi. Template Attacks. In B. S. K. Jr., Çetin Kaya Koç, and C. Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES'02*, volume 2523 of *LNCS*, pages 13–28. Springer, 2002.
7. S. Curry. Don't Believe Everything You Read... Your RSA SecurID Token is Not Cracked. blog entry, June 2012.
8. T. Eisenbarth, T. Kasper, A. Moradi, C. Paar, M. Salmasizadeh, and M. T. M. Shalmani. On the Power of Power Analysis in the Real World: A Complete Break of the KeeLoq Code Hopping Scheme. In *Advances in Cryptology – CRYPTO 2008*, volume 5157 of *LNCS*, pages 203–220. Springer, 2008.
9. M. A. Elaabid and S. Guilley. Portability of templates. *Journal of Cryptographic Engineering*, 2(1):63–74, 2012.
10. J. Grand. Hardware Token Compromises. Presentation at Black Hat USA 2004, 2004.
11. Kingpin. Attacks on and Countermeasures for USB Hardware Token Devices.
12. P. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In *Proceedings of CRYPTO 1999*, pages 388–397, 1999.
13. Langer EMV-Technik. LF1 Near Field Probe Set. Website.

14. Langer EMV-Technik. Preamplifier PA 303. Website.
15. S. Mangard, E. Oswald, and T. Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer-Verlag, 2007.
16. A. Moradi, A. Barenghi, T. Kasper, and C. Paar. On the vulnerability of FPGA bitstream encryption against power analysis attacks: extracting keys from Xilinx Virtex-II FPGAs. In *CCS 2011*, pages 111–124. ACM, 2011.
17. A. Moradi, M. Kasper, and C. Paar. Black-Box Side-Channel Attacks Highlight the Importance of Countermeasures - An Analysis of the Xilinx Virtex-4 and Virtex-5 Bitstream Encryption Mechanism. In *CT-RSA 2012*, volume 7178 of *LNCS*, pages 1–18. Springer, 2012.
18. A. Moradi, D. Oswald, C. Paar, and P. Swierczynski. Side-channel attacks on the bitstream encryption mechanism of Altera Stratix II: facilitating black-box analysis using software reverse-engineering. In *Proceedings of the ACM/SIGDA international symposium on Field programmable gate arrays*, FPGA '13, pages 91–100, New York, NY, USA, 2013. ACM.
19. D. Oswald and C. Paar. Breaking Mifare DESFire MF3ICD40: Power Analysis and Templates in the Real World.
20. D. Oswald and C. Paar. Improving side-channel analysis with optimal linear transforms. In S. Mangard, editor, *Smart Card Research and Advanced Applications*, volume 7771 of *LNCS*, pages 219–233. Springer, 2013.
21. K. Paterson and N. AlFardan. On the Security of RC4 in TLS. Website, March 2013.
22. Pico Technology. PicoScope 5200 USB PC Oscilloscopes, 2008.
23. K. S. Shanmugam. *Digital & Analog Communication Systems*, chapter 8.3.2. Wiley-India, 2006.
24. Sunplus Innovation Technology Inc. http://www.sunplusit.com.
25. L. Vamanu. Formal Analysis of Yubikey. Master's thesis, INRIA, 2012.
26. E. W. Weisstein. Variance. Mathworld - A Wolfram Web Resource, December 2010. `http://mathworld.wolfram.com/Variance.html`.
27. Yubico. www.yubico.com.
28. Yubico. Download of personalisation tool. http://www.yubico.com/products/services-software/personalization-tools/.
29. Yubico. How YubiKeys are manufactured. https://www.youtube.com/watch?v=s8_I1-ErZSQ.
30. Yubico. Yubico Reference Customers: Department of Defense. http://www.yubico.com/about/reference-customers/department-defence/.
31. Yubico. *Yubikey Security Evaluation Version 2.0*.
32. Yubico. *The YubiKey Manual*. Yubico, May 2012.
33. Yubico. YubiKey NEO. Website, 2013.

Correlation for All Key Bytes



(a) Power measurements



(b) EM measurements

Fig. 15: Evolution of the maximum correlation (vertical axis) over the number of used traces (horizontal axis) for all key bytes (left to right, top to bottom). Red: Correct key candidate