CHAPTER

SIDE-CHANNEL ATTACKS

8

CONTENTS

Hardware Security. https://doi.org/10.1016/B978-0-12-812477-2.00013-7

8.1 INTRODUCTION

Side-channel attacks (SCA) is a noninvasive attack that is based on targeting the implementation of a cryptographic algorithm rather than analyzing its statistical or mathematical weakness. These attacks exploit physical information leaking from various indirect sources or channels, such as, the target device's power consumption, electromagnetic (EM) radiation, or the time taken for a computation. These channels are referred to as "side channels". The information embedded in side-channel parameters depend on the intermediate values computed during the execution of a crypto-algorithm, and are correlated with the inputs and the secret key of the cipher [1]. An adversary can effectively extract the secret key by observing and analyzing side-channel parameters with relatively cheap equipment, and within a very short time span, ranging from a few minutes to a few hours. Due to these reasons, SCA poses a major threat to cryptographic devices, especially smart cards and IoT devices, for which an attacker can have easy access to these physical parameters.

Figure 8.1 illustrates how a device leaks side-channel information while operating. Common side-channel attacks, such as power attacks, monitor the device's power consumption. Typically, this is done by incorporating a current path at V_{dd} or Gnd pin of a chip, which is performing the cryptographic operation, to record power dissipation for such an operation. The device's power consumption captures switching activity of the relevant transistors, which depends on inputs to a cryptographic function, such as the plaintext and the key. While the device is in operation, the power consumption can be measured using an oscilloscope, and the relation between the power consumption and the secret key is analyzed in various ways. Simple power analysis (SPA) is a technique to directly interpret the collected traces of power consumption for a set of inputs. It requires relatively detailed knowledge about the implementation of a cryptographic algorithm and a skilled adversary to interpret secret key information by visually examining the power consumption. Figure 8.2 provides the overview of the process of SCA. In contrast, differential power analysis (DPA) is a statistical analysis approach that does not require detailed knowledge of the target hardware implementation, which may be considered as a black box. DPA has been shown to be effective in finding a correlation between power consumption and processed data related to the secret key by statistical methods. In order to perform DPA successfully, however, often large number of power measurements are required.

8.2 BACKGROUND ON SIDE-CHANNEL ATTACKS

The first reported instance of SCA was an attack on a government agency in 1965. The attack was applied to a cryptographic machine that produced ciphertexts using a key that was reset every day [6].

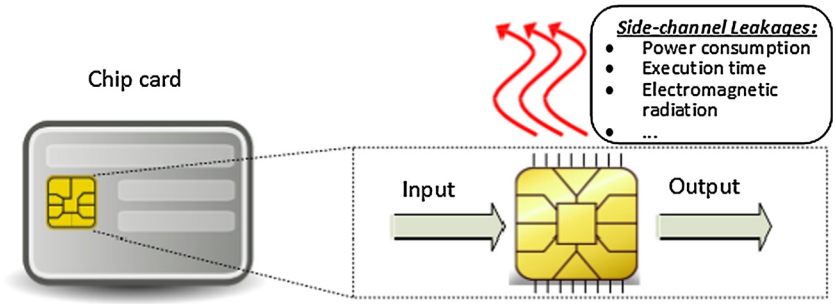


FIGURE 8.1

Side-channel leakages while a cryptographic hardware is in operation.

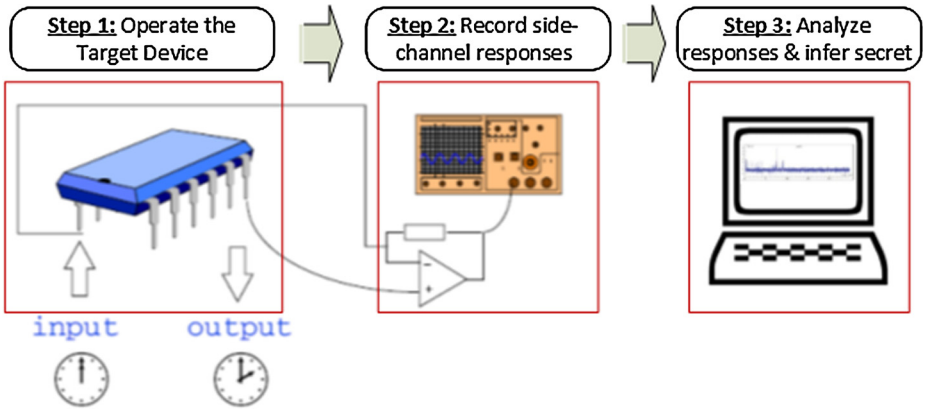


FIGURE 8.2

The data collection process of side-channel attacks, where a controlled attack scheme is enforced, and the measurements are provided back to the processing unit in order to perform the side-channel analysis. The process is usually iterative to ensure a wider functional coverage and optimized results.

By recording the sound of the module, attackers were able to derive the secret key. They related the number of click sounds, which were produced by the machine, to the value of the key. Since then SCAs have greatly evolved and relied on several other parameters, e.g., power, timing, and EM. The timeline in Fig. 8.3 shows the evolution of SCAs over past five decades.

8.2.1 TAXONOMY OF SIDE-CHANNEL ATTACKS

Based on the level of control that an attacker may have on a device prior to performing SCAs, they can be classified into passive and active attacks. Passive attacks (such as power, timing, or EM SCAs) do not require an attacker to interfere with the functionality or the operation of the device under attack [10]. The attack is usually launched in a manner that allows the system to behave normally as if the

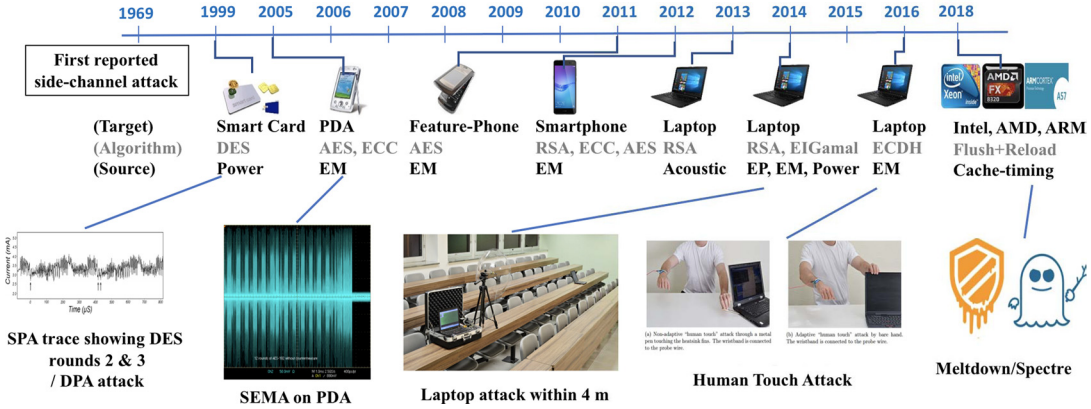


FIGURE 8.3

Evolution of side-channel attacks.

attack is not in effect. On the other hand, active attacks aim to interfere with the operation of the device under attack, where an attacker tends to influence how the device behaves, and what operation it performs. By actively controlling the behavior of the device, an attacker gains the advantage of selectively extracting side-channel information that can help break a cryptographic module, or extract the secret key.

Each side-channel attack can be done in many ways. Typically, a simple nonexhaustive approach has been introduced first, and then a refined and more complex approach is developed to enhance the amount and quality of extracted side-channel information. In case of power analysis attacks, as mentioned earlier, an adversary can perform a simple analysis, where a power signal is simply visually inspected. In a more sophisticated version of the attack, namely, DPA, multiple power traces are statistically analyzed to derive more robust information about the secret key.

Figure 8.4 shows the taxonomy of SCAs. Depending on the general source of side-channel information, there are several forms of SCA. They are: power SCA, EM SCA, fault injection attack, and timing SCA. Each SCA can be classified according to specific attack method: applied analysis methods, such as simple observation and statistical methods; side-channel signal generation methods, such as voltages and clocks; or analysis granularity, such as microarchitecture and system level analysis. [11].

8.2.2 UNCOMMON SIDE-CHANNEL ATTACKS

Besides the common ones described earlier, there are several other side-channel signals that can leak information about stored secrets in a hardware. These signals include emitted sound, temperature, and vibration. The analysis of these signals to extract secret information is not widely researched. One example of these uncommon SCAs is acoustic side-channel analysis [22]. It resembles the first reported SCA in 1965 in terms of the side-channel signal used in the attack [6]. The attack focuses on systems that produce sounds while being operated (such as, 3D printers), where program information can be extracted from the leaked acoustic signals. The captured sound signal is run through a series

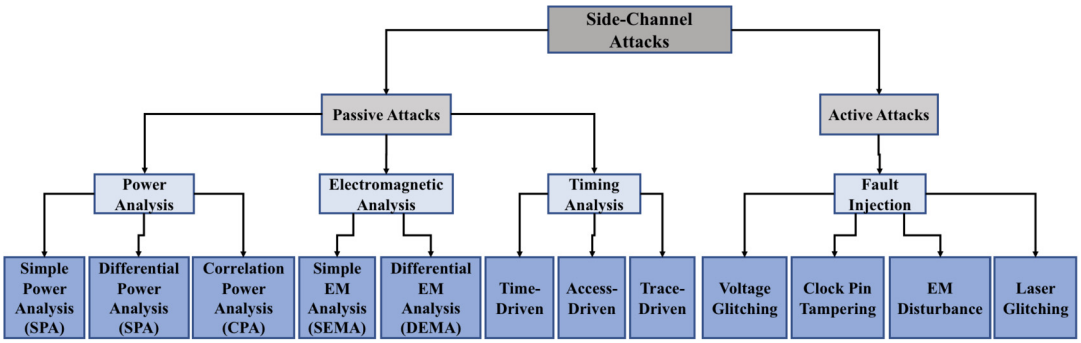


FIGURE 8.4

Taxonomy of general side-channel attacks.

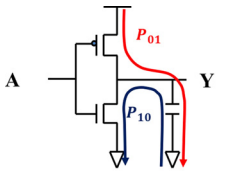
of signal processing and machine-learning stages that can accomplish reconstructing the operation and producing an output similar to that of the device under attack. Other uncommon side-channels, such as, temperature and vibration, can also leak a significant amount of critical information about the device under attack. In order to build secure systems, all forms of side-channels need to be considered as a valid threat to information leakage, and adequate countermeasures need to be incorporated.

8.3 POWER ANALYSIS ATTACKS

The basic idea of power analysis attacks is to reveal secret information from a device by analyzing its power consumption [12]. The attack is non-invasive, and it requires physical access to the device, due to the need to capture current signatures that are produced while the device is undergoing an operation. Power analysis attacks are mainly used to extract the secret key of cryptographic systems, as it has been used to successfully break the Advanced Encryption Standard (AES) in a few minutes.

Power analysis attacks have been studied extensively by academic and industrial researchers as a dominant form of SCA. Various power analysis attacks, such as, SPA, DPA, and correlation power analysis (CPA) have been developed to reveal critical information about the device under attack. A set of power measurements is required for each side-channel analysis to be applied; these sets vary in scope and form, depending on the type of attack, the complexity of the design, and the accuracy of the data collection process. Each power signal captured during the analysis is called a power trace. An attacker usually needs to use a large number of power traces in all attack modes before applying the power analysis attack.

In this section, we explore what causes power signals to exist, and what factors would affect the shape of these signals. We also discuss the types of power signals that are generated during the device's operation, and how to capture them accurately. We explain the types of attacks that can be applied, and what information is extracted from a successful attack. Finally, we discuss the countermeasures, and what can be done to prevent attackers from performing SCAs.



A	Y	Power
$0 \rightarrow 0$	$1 \rightarrow 1$	P_{11}
$\mathbf{0 \rightarrow 1}$	$\mathbf{1 \rightarrow 0}$	$\boldsymbol{P_{10}}$
$\mathbf{1 \rightarrow 0}$	$\mathbf{0 \rightarrow 1}$	$\boldsymbol{P_{01}}$
$1 \rightarrow 1$	$0 \rightarrow 0$	P_{00}

FIGURE 8.5

Dynamic power of an inverter.

8.3.1 ORIGINS OF SIDE-CHANNEL LEAKAGE IN POWER CONSUMPTION

Two factors affect the power consumption of a device; the first factor is the dynamic power, which is caused by the switching activities of transistors within the device. The second factor is the leakage power, which is an unwanted behavior of a transistor, associated with the leakage current generated in its off state. An adversary is usually interested in capturing the dynamic power signals as they are directly related to the functional behavior of the device, i.e., specific operations going inside a device. For example, the dynamic power of an inverter is related to the switching activities of the input and the output, as shown in Fig. 8.5. Let P_{ij} be the power consumption, where the output value of the inverter is changed from i to j, for $i, j \in \{0, 1\}$. P_{01} and P_{10} are much greater than P_{00} and P_{11}, since the capacitor connected to the output is charged or discharged when the output value is switched; P_{00} and P_{11} are almost zero since there is no charging or discharging activity. Based on this characteristic, an adversary can estimate the status of the output or input by measuring the power of an inverter. If the input of an inverter originates from a secret key, the power side-channel leakage gives an adversary a clue about the secret key.

8.3.2 ACQUISITION OF POWER SIGNALS

The process of capturing power signals is straightforward and easily accomplished by a capturing equipment with high sampling rates, for example, an oscilloscope, which can be obtained at reasonably small cost. The process of power acquisition requires basic knowledge about the functionality of a device, where an input pattern is applied, and the power traces are captured during the processing of those patterns.

The power signals are captured by measuring the change in current levels in the voltage supply transmission lines. Usually, an oscilloscope measures the voltage drop across a precision sense resistor connected between the power rail (e.g., output of a voltage regulator that delivers power to a PCB) and V_{dd} or *Gnd* pin of the target device. Figure 8.6 presents an overview of a power analysis setup, where a cryptographic system is being controlled by a computer that applies input patterns, observes the outputs, measures the power consumption, and performs necessary analysis to extract the secret key.

Collected power traces include noise, which consists of algorithmic and natural electrical noise. The algorithmic noise is caused by the switching activity of other modules, and the natural electrical noise results from various environmental effects, e.g., electromagnetic inference. To eliminate noise, an adversary can pass the acquired power traces through a filtering phase. This filter removes the natural

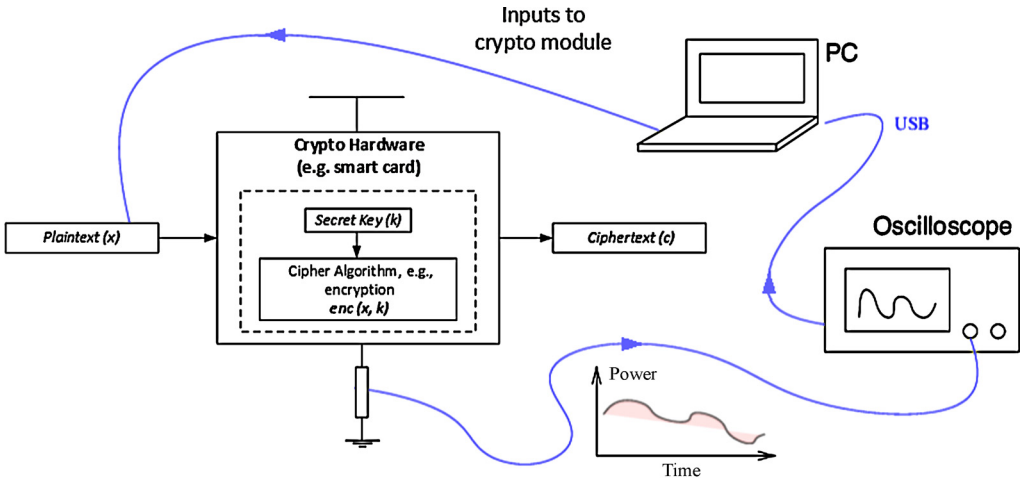


FIGURE 8.6

Typical setup for a power analysis attack on a crypto hardware.

noises that are caused by the device and other surroundings. A process of identifying noise levels and frequency bands can be applied to design the noise reduction filter, and accurately capture the power trace. A number of power traces can be averaged to remove noise and smoothen the signal. Depending on the device, the implementation of the algorithm, and type of attack, the number of power traces can vary from a single trace to millions of traces.

8.3.3 TYPES OF POWER ANALYSIS

We introduce three types of power side-channel analysis: simple power analysis (SPA), differential power analysis (DPA), and correlation power analysis (CPA).

8.3.3.1 Simple Power Analysis (SPA)

SPA is a technique that aims to observe power measurements obtained while the device under attack is in operation mode. This type of analysis does not require any advanced or statistical processing stages. SPA attacks are usually applied to devices with limited accessibility, where one or few power traces are available. SPA can be applied to a single power trace, where the attacker attempts to observe critical information or secret keys from that trace. When SPA is applied to multiple power traces captured from numerous occurrences, these traces are averaged to remove noise. In both cases, the attacker can only apply a successful attack if the recorded power consumption can lead to critical information about the device being revealed [15].

Visual inspection of power traces is considered the primary form of SPA attack, where a power trace shows a sequence of patterns that can lead to identifying key bits, instructions, or functions. Each instruction in a processor causes a specific pattern that can be visually identified in the power trace. Hence, visual inspection can prove useful when looking for clear patterns in the device under attack.

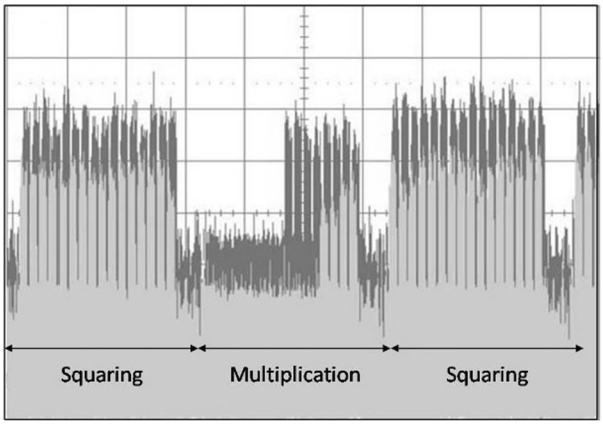


FIGURE 8.7

Part of a modular exponentiation power traces in RSA decryption [18].

Figure 8.7 shows a sequence of patterns corresponding to squaring and multiplication operations for a modular exponentiation in the RSA decryption.

Template attacks are a more advanced form of SPA, where known and recognized patterns in the power trace are characterized and stored as templates. Power traces collected from the target device are matched to the templates, and then corresponding operations are recognized. SPA can also be used as the first step to a more advanced approach, where power traces are used to identify the length of pipelines, processor load, microarchitectural events (such as, cache misses and hits), and the different functions that may trigger when changing the input patterns.

8.3.3.2 Differential Power Analysis (DPA)

DPA attacks are the most common type of side-channel attacks, due to the fact that attackers are not required to have prior knowledge about the hardware architecture of the device under attack to perform the analysis. Additionally, DPA has been proven very effective in obtaining high-quality signals in a noisy environment. Compared to SPA, DPA typically requires larger number of traces; more data collection makes DPA more powerful. DPA is widely used to reveal secret keys of cryptographic systems by obtaining power traces while the system is encrypting or decrypting data blocks [8].

In DPA, an adversary can successfully exploit data dependency of the power consumption, which gives them the ability to observe internal transitions, and extract secret keys and critical information. Implementation of DPA attack requires two phases: data collection and data analysis. In the data collection phase, different input patterns are applied to the device while recording the power traces in a high sampling rate. Averaging the measured traces, and applying a bandpass filter that is tuned to remove noise, can help improve the quality of traces. In the data analysis phase, statistical analysis, such as the difference of means is applied. Figure 8.8 shows an example of DPA being applied to an AES block. Based on the decision function, for example, the MSB of the substitution box (SBOX) [11] operation in Fig. 8.8, power traces are classified into two sets, and then the difference between means

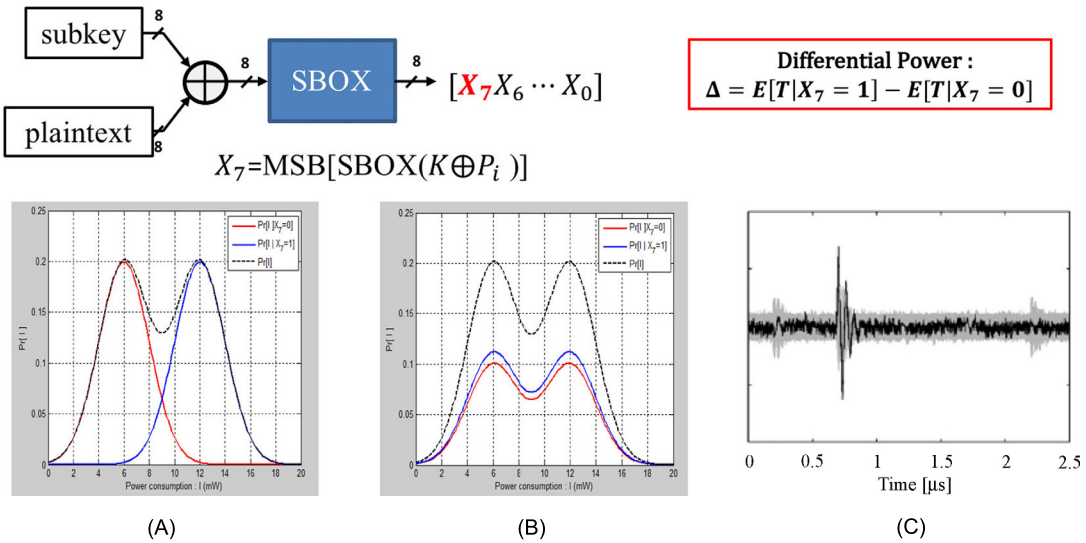


FIGURE 8.8

DPA attack to reveal a secret key. The probability density function for a correct key guess is shown in (A), the probability density function for a wrong key guess is shown in (B), the difference of means is shown in (C).

of the two sets is calculated as follows:

$$\Delta = \frac{\sum_{i=1}^{m} D(K, P_i)T_i}{\sum_{i=1}^{m} D(K, P_i)} - \frac{\sum_{i=1}^{m}(1 - D(K, P_i))T_i}{\sum_{i=1}^{m}(1 - D(K, P_i))}.$$

In the equation above, $D(K, P_i) = MSB(SBOX(K \oplus P_i))$, K is the guessed key of an adversary, P_i is the plaintext, and T_i is a collected power trace, when the device is performing SBOX operation. If the guessed key is correct, then the conditional probability density functions given by $D = X_7 = 0$ and $D = X_7 = 1$, are completely different as shown in Fig. 8.8A. Otherwise, the conditional probability density functions are similar, as shown in Fig. 8.8B. Therefore, in case of the correct guessed key, the difference of means is the largest and in other cases, the difference of means is almost zero, as shown in Fig. 8.8C [9].

High-order DPA is a technique that utilizes multiple points in the entire power trace, or high-order statistics of a point, such as the second, third, and higher moments (i.e., variance, skewness, and kurtosis). This technique can successfully exploit the device's vulnerabilities, and bypass the traditional power analysis countermeasures.

DPA targets the correlated regions of the device's power consumption, which gives an adversary the ability to automate the analysis, and even train it to adapt to a device and environmental variations. The amount of information, the low cost, and the noninvasive nature of the attack make it one of the most powerful side-channel analysis attacks. It is worth emphasizing that DPA has been used to successfully attack many devices [24].

8.3.3.3 Correlation Power Analysis (CPA)

CPA is an advanced form of SCA that exploits the correlation between the power consumption, and the Hamming distance or Hamming weight of the target function, for example, the output of the SBOX operation. The first step of the CPA attack is to determine the intermediate value of the cryptographic algorithm executed by the device under attack, that is, the target function, which is denoted by $v_i = f(d_i, k^*)$, where d_i is the ith plaintext or ciphertext, and k^* is the hypothesis of a component of the secret key [16].

The second step is to measure the power consumption of the cryptographic device while it encrypts or decrypts D, different data inputs, including the target function at the first step. We denote the power trace as $\vec{t_i} = (t_{i,1}, t_{i,2}, \ldots, t_{i,t^*}, \ldots, t_{i,L})^T$, corresponding to input d_i, where L denotes the length of the trace, and t_{i,t^*} is the power consumption when the target function at the first step is performed. An adversary measures a trace for each of the D data inputs, and hence, the traces can be written as matrix $\mathbf{T}$ of size $D \times L$: $\mathbf{T} = (\vec{t_1}, \vec{t_2}, \ldots, \vec{t_{t^*}}, \ldots, \vec{t_L})$, where $\vec{t_j}$ for $j = 1, \ldots, L$ is a column vector of size $D \times 1$.

The third step is to calculate a hypothetical intermediate value for all possible k : $v_{i,j} = f(d_i, k_j)$ for $i = 1, \ldots, D$ and $j = 1, \ldots, K$.

The fourth step is to map the hypothetical intermediate values to the hypothetical power consumption values: $h_{i,j} = g(v_{i,j}) = g(f(d_i, k_j))$ for $i = 1, \ldots, D$ and $j = 1, \ldots, K$. The most commonly used power consumption models are the Hamming-distance and the Hamming-weight models. The $D \times K$ matrix $\mathbf{H}$ is made at this step: $\mathbf{H} = (\vec{h_1}, \ldots, \vec{h_K})$, where $\vec{h_i}$ for $i = 1, \ldots, K$ is a vector of size $D \times 1$.

The fifth step is to compare the hypothetical power consumption model with the measured power traces. In order to measure the linear relationships between the two vectors $\vec{h_i}$ and $\vec{t_j}$ for $i = 1, \ldots, K$ and $j = 1, \ldots, T$, the correlation coefficient is calculated:

$$r_{i,j} = \frac{\sum_{d=1}^{D}(h_{d,i} - \overline{h_i})(t_{d,j} - \overline{t_j})}{\sqrt{\sum_{i=1}^{D}(h_{d,i} - \overline{h_i})^2 \sum_{i=1}^{D}(t_{d,j} - \overline{t_j})^2}}$$

where $\overline{h_i}$ and $\overline{t_j}$ denote the mean values of the vector $\vec{h_i}$ and $\vec{t_j}$, respectively. If r_{k^*,t^*} of the correct key k^* and the specific time t^* has the distinct peak value, then the CPA attack is successful.

8.3.4 POWER SIDE-CHANNEL ATTACK COUNTERMEASURES

In order to remove dependency between power consumption and intermediate values of the executed cryptographic algorithm, the cryptographic hardware can be implemented with secure primitive logic cells (such as sense-amplified-based logic (SABL) [19], wave dynamic differential logic (WDDL) [20], and t-private logic circuit [21]) at the design stage. These secure logic styles use different methods to make the power consumption of the performed operation independent of the processed data values, thus preventing leakage of secret information (i.e., key) in power traces. SABL and WDDL consume equal amounts of power in each clock cycle, but t-private logic circuit randomizes amounts of power consumption in each clock cycle by masking each bit with t random bits. In other words, SABL and WDDL implement the *hiding countermeasure*, and t-private logic circuit implements the *masking countermeasure*.

Table 8.1 Secure logic style

	SABL	WDDL	t-private logic
SCA resistance	✓	✓	✓
Probing resistance	×	×	✓
Method	Hiding	Hiding	Random masking
Design	Full custom	Semicustom	Semicustom
Area	Medium	Low	High
Power	Medium	High	Low

Whereas all these secure cells have varying level of robustness against SCAs, only t-private logic circuit prevents the probing attack, which allows an adversary to observe only t-limited number of internal nodes per each clock cycle. In terms of their implementation, t-private logic circuit and WDDL are implemented with the general CMOS digital cell library, but each SABL cell should be full-customized. Of these secure logic design styles, t-private has the largest circuit area, but the power consumption of t-private logic circuit is the smallest. Since SABL and WDDL have two-phase (the pre-charge phase and the evaluation phase), during each clock cycle in which phase signals are switched, the power consumptions of SABL and WDDL are larger than that of t-private logic circuit. Table 8.1 shows the summary of these secure logic styles.

8.3.5 HIGHER-ORDER SIDE-CHANNEL ATTACKS

Higher-order side-channel attacks exploit multiple leakages, corresponding to several intermediate values during execution of a cryptographic algorithm [26]. The $(n+1)$st-order SCA is effective in the nth-order masking countermeasures, in which an intermediate value is masked with n random values. When an intermediate value is masked with a random value in the cryptographic device, second-order DPA or CPA attacks can be performed in order to reveal the secret key. For example, we assume that the input and output of AES SBOX operations are concealed with the same mask r as follows: $v = (p \oplus k) \oplus r, u = \text{SBOX}(p \oplus k) \oplus r$, and two intermediate values are stored in a register. The SBOX input and output are targeted in the second-order CPA attack. For the attack, adversary's hypothetical function h is defined as the Hamming distance between two intermediated values: $h_{k_i} = HW(v \oplus u) = HW((p \oplus k_i) \oplus \text{SBOX}(p \oplus k_i))$, and two points l_1 and l_2 of collected power traces when two intermediate values are stored in a register are combined as the absolute-difference function: $t = |l_1 - l_2|$. The reason to use the absolute-difference function among other functions, such as summation, subtraction, or square of summation to combine two points, is that the absolute-difference function has a higher correlation to the hypothetical function [26].

If the guessed key is equal to the correct key, the correlation between the hypothetical function and the combined function has the maximum value: $k^* = \arg\max_{k_i \in \mathcal{K}} \rho(h_{k_i}, t)$, where k^* is the correct key. Consequently, the correct key can be revealed by comparing the hypothetical power consumption with the combined power trace.

8.3.6 POWER SCA SECURITY METRIC

A device's security against SCAs needs to be evaluated using appropriate metric. There are different methods to measure the level of protection of the device under attack. These methods assess the difficulty of performing a successful SCA, and the time required to successfully extract the critical information from the device.

The test vector leakage assessment (TVLA) is a common assessment approach to measure how easy it is to detect any data leakages in a device. This assessment is done by applying a predefined set of test inputs, detecting leakage, and evaluating the ability to extract significant information from the traces. The leakage assessment is based on Welch's t-test, which is used to test the hypothesis that two populations have equal means when two samples have unequal variance and unequal sample size. In the side-channel evaluation process, n side-channel measurements are collected while the device under the test operates with a secret key. The n measurements $\bar{\mathbf{p}}^i = [p_0^i, \ldots, p_{m-1}^i]$ for $i = 1, \ldots, n$, where m is the number of the sampling points, are classified into two sets by the determinant function D: $S_0 = \{\bar{\mathbf{p}}^i | D = 0\}$, and $S_1 = \{\bar{\mathbf{p}}^i | D = 1\}$. If the t-test statistic

$$t = \frac{\mu_0 - \mu_1}{\sqrt{\frac{\sigma_0^2}{N_0} + \frac{\sigma_1^2}{N_1}}}$$

is out of the confidence interval, $|t| > C$, then the null hypothesis, $H_o : \mu_0 = \mu_1$ is rejected. This means that two groups are distinguishable and the implementation has high probability to leak information. Thus, it does not pass the leakage assessment test. Let us assume that the threshold value C is chosen as 4.5, which leads to a confidence of > 0.99999 to reject the null hypothesis, and the determinant function D is defined as

$$D = \begin{cases} 0 & \text{if plaintext is random,} \\ 1 & \text{if plaintext is fixed,} \end{cases}$$

which is referred to as *nonspecific fixed-vs-random* test.

Another method used to measure side-channel vulnerability is to apply the attack success rate analysis. This success rate is defined as the number of successful attacks (i.e., the key is derived through the attack) divided by the total number of performed attacks, where the maximum rate of 100% means the device is attacked successfully every time a side-channel analysis is applied, and the minimum of 0%, which means the device is protected against all attacks. This assessment can also reflect the time needed for the attacker to extract critical information from the device.

8.4 ELECTROMAGNETIC (EM) SIDE-CHANNEL ATTACKS

EM SCA focuses on measuring electromagnetic waves that are emitted from ICs in operation. These EM waves are defined as synchronized oscillations of electric and magnetic fields that propagate at the speed of light through a vacuum [14].

In this section, we discuss the origin of different EM signals, and the equipment needed to capture them. We differentiate between intentional and unintentional EM signals. We also address the amount and type of information that could be leaked through EM side-channel. We describe a data acquisition

process that can capture low energy, yet critical, EM signals. We explain various types of EM based side-channel attacks that could be applied using side-channel analysis, such as simple electromagnetic attacks (SEMA), and differential electromagnetic attacks (DEMA). Finally, we outline possible countermeasures to protect devices against EM side-channel attacks.

8.4.1 ORIGIN OF EM SIGNALS

The EM waves are produced as current flows across a device, where transistor and interconnect switching activities occur with changing input patterns. This current flow results in the EM signals. EM signals of specific current flow may not only be affected by the physical or functional structure of the device, but can also be affected by the EM waves of other components, and their current flows.

An adversary usually aims to capture EM signals that are produced by current flows of data processing stages, where most waves occur, due to the switching activity of a device while performing a data processing operation. These waves are usually considered unintentional, and they allow critical information to be leaked naturally during operation. When applying EM side-channel analysis, switching activities can be easily captured and translated into a series of events and instances that occur in each clock cycle. This type of attack is similar to the power side-channel analysis, where a one-dimensional view of current activity is used to extract critical secret from a device. Power analysis attacks, such as DPA, however, cannot extract any spatial information, e.g., the location of a specific current activity. On the other hand, an EM side-channel attack can also identify the location of an EM signal, which makes it a powerful attack vector.

8.4.2 EM EMANATIONS

EM emanation is defined as the process that causes the target device to generate EM signals. There are two types of EM emanations: intentional and unintentional emanations. Next, we describe these two broad types.

8.4.2.1 Intentional Emanations

Intentional EM emanation results from current flows that are applied to cause the device to emit an electromagnetic response [17]. These current flows are usually in the form of short bursts and sharp rising edges, which cause a high-power emanation that would be readily observable across a full frequency band. Often, the applied current flow targets a higher frequency band to quickly capture the response due to noise and other interfering emissions in a lower frequency band. The objective of an attacker in this type of emanation is to isolate the EM response of the targeted critical data path. In order to do this, a tiny and sensitive EM probe is needed. Delayering of the device may also help improve the captured signal quality.

8.4.2.2 Unintentional Emanations

When an adversary applies EM side-channel analysis, focusing on unintentional emanation can help identify critical paths and acquire their data values. The increased complexity and reduced size of modern ICs result in electric and electromagnetic coupling between components, which is an uncontrolled phenomenon that can generate a compromising signal. These components may act as modulators; they generate a carrier signal that can be intercepted and post-processed to acquire the carried data.

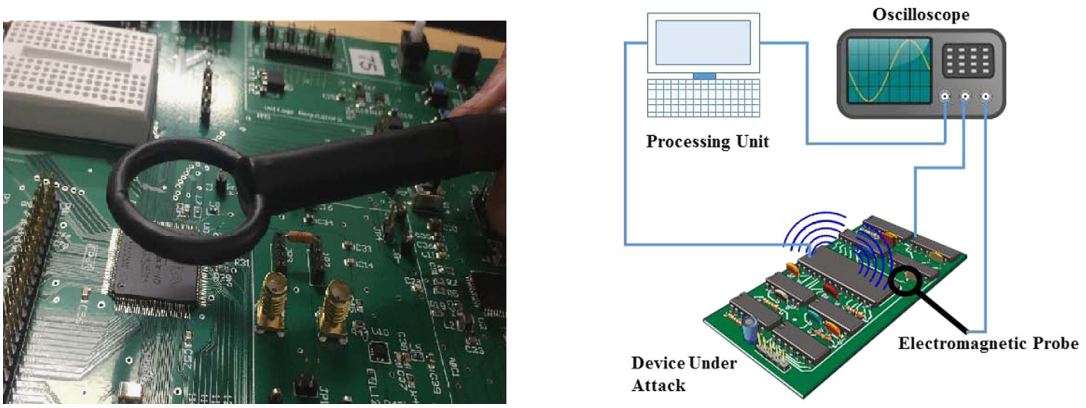


FIGURE 8.9

(A) A picture of an EM probe being placed on an FPGA; (B) EM Side-channel analysis setup.

Modulation of the signals can be either an amplitude modulation (AM), or a frequency modulation (FM). In AM, the coupling between the carrier signal and the data signal result in an AM emanation; the data signal can be extracted by demodulating the AM signal using a tuned receiver. In FM, the coupling results in a frequency shifted signal; this signal can be demodulated by using an FM receiver.

8.4.3 ACQUISITION OF EM SIGNALS

EM signals often propagate through conduction and radiation; these signals can be intercepted using sensors, such as a near-field probe or an antenna. Using these sensors allows the EM signal to be transferred into a current signal, which is post-processed to remove noise, and limit the frequency band in order to apply the EM analysis. The quality of the received signal is usually improved if the used sensor is shielded from unwanted frequency bands, or other EM interferences.

Post-processing of the signal may include filtering frequency bands that are not related to the targeted critical data path, which requires prior knowledge of the frequency band that holds the information. In order to obtain that knowledge, a spectrum analyzer is commonly used to identify carriers and noise; then a post-processing filter can be tuned only to allow critical information to pass. Figure 8.9 shows an illustration of a measurement setup for the EM attack.

8.4.4 TYPES OF EM ANALYSIS

There are two main types of EM analysis: Simple and differential electromagnetic analysis, SEMA and DEMA, respectively.

8.4.4.1 Simple Electromagnetic Analysis (SEMA)

In SEMA, an attacker obtains a single time domain trace to observe, and gain knowledge about the device directly. The attack is only valid when there is prior knowledge about the device's architecture, or the security policy, applied. The primary objective in SEMA is to obtain critical information via visual inspection of the EM signal trace, where a sequence of transitions at the startup of the system may include information about the secret key used to encrypt/decrypt data. The use of SEMA is usually the first step of EM SCA, where the necessary information can be observed to carry on a more detailed analysis using DEMA.

8.4.4.2 Differential Electromagnetic Analysis (DEMA)

The attacker applies DEMA to the device to exploit information that cannot be visually observable. DEMA generally utilizes a self-referencing approach, which compares the analyzed signal with an equivalent one in a different area of the device (spatial referencing), or in a different time (temporal referencing). DEMA does not require much knowledge about the device under attack; most information can be exploited when obtaining different forms of EM signals in different places and times. The analysis in DEMA can help identify functional and structural details of the target device. It can also track a process flow and determine how a signal propagates inside the device. These details obtained by DEMA can help reverse engineer the device, or give the attacker the ability to disable the security policy of the system physically.

8.4.5 EM SCA COUNTERMEASURES

To protect against EM SCAs, many countermeasures can help add a layer of protection, while maintaining the device's performance and quality of service. Redesigning the circuit to reduce the coupling issue is one of the primary countermeasures. Additionally, adding a layer of shielding to the device to prevent EM signals from propagating is another significant measure. Introducing nonfunctional modules that produce EM noise can also prevent critical information from being easily intercepted due to the high amount of noise being applied in the same frequency band.

Other functional countermeasures may hide critical processes from being detected, such as introducing a crucial nonlinear processing sequence when using a cryptographic system. By injecting dummy instructions or operations between the stages of a cryptographic process, the adversary can be prevented from differentiating between key-bits and dummy-bits, even when performing successful EM side-channel attacks. Due to the existence of several methods to intercept uncontrolled EM signals, many attacks have been successful in extracting critical information, even when encryption is applied. Hence, countermeasures should be introduced as early as possible in the design phase for the device to be adequately protected against EM SCAs.

8.5 FAULT INJECTION ATTACKS

Unlike power analysis attacks, fault injection attacks are active attacks, where a crypto-device is intentionally injected with a fault that leads to a leakage of the secret key [1,7]. The injected fault is designed to introduce a temporary malfunction during the device operation. This malfunction is typically a disturbance of a few memory or register bits. As the execution continues, the disturbance, i.e.,

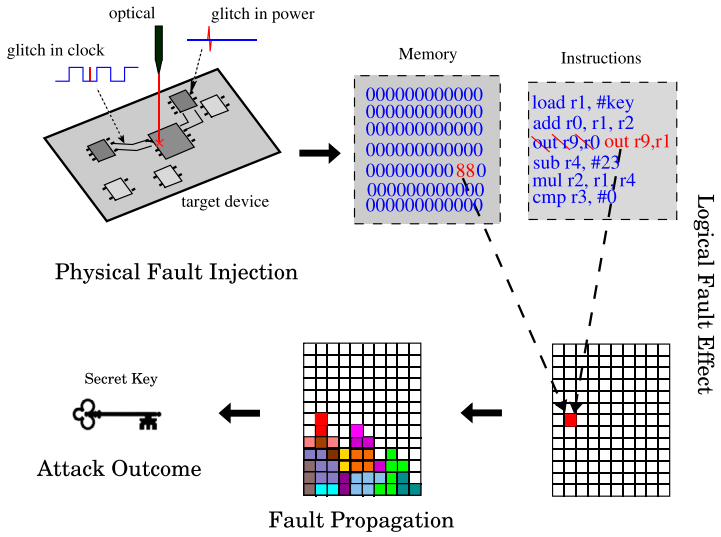


FIGURE 8.10

In a fault injection attack, or fault attack, a physical fault is deliberately injected in a device during its operation with the objective of leaking critical information. Such a fault can be injected by disturbing the clock or voltage source, or by using a laser beam, in order to modify memory or register locations, or to induce other fault effects (e.g., skip an instruction). In a crypto device, such a disturbance often gets propagated to other locations during the execution, and eventually results in a faulty ciphertext. The faulty ciphertext can then be evaluated to retrieve the secret key.

single/multiple memory bit-flip, propagates to other memory locations and eventually results in a corrupted output. We call this corrupted output a faulty ciphertext. If the fault is injected precisely and has specific properties, an attacker can use the faulty ciphertext to derive the secret key. Figure 8.10 shows the entire process involved in a fault attack. Several encryption schemes, such as AES, RSA, and ECC have been demonstrated to be vulnerable to this attack.

A typical attack starts with injecting a fault to an operational device, which can either be a voltage or a clock source glitch [4,5]. Other techniques, such as electromagnetic radiation, or physical probing, or passing a laser beam through the device can also be used for the fault injection. The device is then analyzed by observing the faulty outputs; these outputs can potentially help extract the secret key. The attack is considered semi-invasive, as the physical modification is sometimes needed for the fault to be properly injected.

The attack requires prior knowledge about the design in order for it to be successful. Choosing the type of fault, the location and time of the injection process, is not possible when the device is treated as a black box. An example of a simple fault injection attack on AES is shown in Fig. 8.11. Let k_0 represent the bit 0 of the AES key. Now, consider that a stuck-at-zero fault is injected in k_0 during the initial Key Addition operation, which forces the value of k_0 to be zero. There are two possible outcomes. First, if $k_0 = 0$ prior to fault injection, then the fault induced has no effect on the output. On the other hand, if $k_0 = 1$, then the fault toggles the value of $p_0 \oplus k_0$. This is a disturbance, which

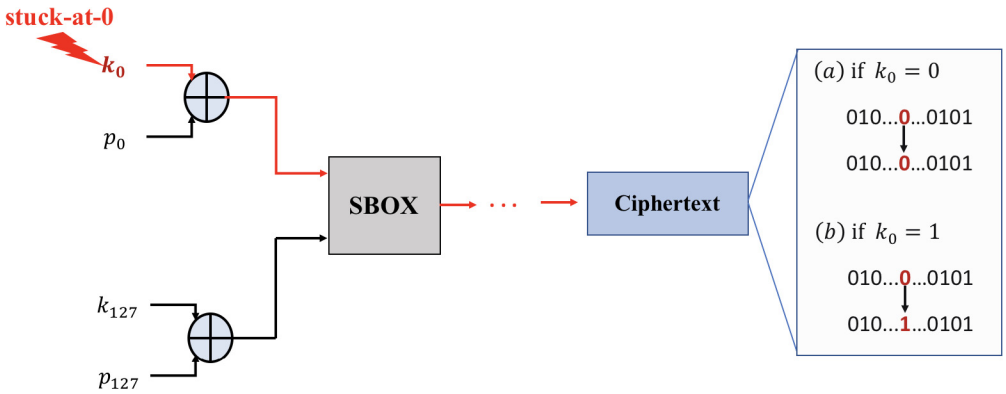


FIGURE 8.11

A simple fault injection attack on AES; the target bit is k_0.

then gets propagated as execution progresses, resulting in a faulty ciphertext. In this case, the faulty ciphertext will be different from that of the fault-free ciphertext, generated from the same plaintext. The attacker, thus, identifies the value of k_0.

In a similar way, by independently injecting faults in key bits k_1 to k_{127}, the attacker can retrieve the entire AES key. This attack—though easy to understand—is very difficult to perform in practice. This is due to the fact that 128 precisely placed and timed faults are required to be generated by an attacker. Each fault should exactly force one bit of the key to zero. Since this simple attack, many stronger and powerful fault attacks have been proposed. The strongest attack can recover the entire secret key with just a single fault injection. This attack requires a fault to be injected at the 8th round of AES. The attack also relaxes the fault injection requirements; the fault only needs to randomly modify a single byte in the 8th round.

8.5.1 FAULT INJECTION TECHNIQUES

Fault injection techniques can vary based on the type of device, and the amount of information available to the attacker. The following subsections explain the main types of fault injection techniques.

8.5.1.1 Voltage Glitching

Voltage glitching is considered the basic fault injection technique, where a device is supplied with a lower than normal voltage level. By running the device in this state, faults would start to appear at the output of the device. The precision and type of fault are controlled by the level of the supplied voltage; a more faulty behavior is obtained as the voltage supply is lowered further. This technique is not invasive and it does not require specific timing patterns. When applying this attack, faults will propagate uniformly across the device, which can give an advantage to the attacker in terms of accessibility to rarely-activated nodes and registers in the device.

8.5.1.2 Tampering With Clock Pin

Another basic fault injection technique involves an attacker subjecting the device with a faulty clock signal. The fault can either be a glitch in the clock, or the voltage level of the signal. This noninvasive attack forces the device to generate faulty outputs across all clocked operations. The attacker needs to have access to the clock signal of the device in order to precisely control the fault, which is a relatively easy task.

8.5.1.3 EM Disturbances

EM Disturbances can be applied to a device to inject faults. Generating EM signals and directing them to a device can cause the operation of the system to be compromised. By controlling the EM signal, different types of behavioral changes in functionality can be observed, and with the right input patterns and enough iterations, the device can leak the secret key of the cryptographic module. Since EM signals are applied to the entire device, the attack affects the system uniformly, as the faults can be injected at any location in the device.

8.5.1.4 Laser Glitching

Applying a laser beam to a specific area of a device can cause a fault to be injected [3]. Data in registers and states can be modified when intentionally applying a strong laser beam. Beams can be controlled in terms of strength and polarization, which allows the attacker to either inject faults to a specific area, or to the entire device. Injected errors can propagate to the output of the design, and successfully leak the secret key of the cryptographic module.

8.5.2 FAULT INJECTION COUNTERMEASURES

Countermeasures for fault injection attacks are limited, as tools and equipment used to inject errors are easy to obtain. The level of accessibility to main ports of the design (such as, the power and clock lines) make these attacks very difficult to combat. Nevertheless, there are some measures that can be taken to avoid leaking critical information when faults are introduced.

One of the most common fault injection attack countermeasures is based on replication of critical operations, which is a popular solution for fault-tolerant computing [2]. Here, the crypto operations are repeated, and the two outputs are compared. If found different, the system assumes that a fault has been injected, and appropriate actions are taken. The replication can be done spatially or temporally. Spatial replication, especially applicable in hardware crypto accelerators, has redundant circuit blocks to recompute specific crypto operations. Temporal replication reuses the same circuit blocks to perform the recomputation at different time. Whereas the spatial countermeasure does not affect the execution time for crypto operation (but adds area overhead), the temporal countermeasure does not affect the area requirements (but adds delay overhead). The former is, therefore, suited for high-speed applications, whereas the latter is suited for small devices.

An alternate protection approach is based on error-detection schemes, such as parity checks. The scheme adds a detection mechanism that disables the critical functionality of the device when it is operating in a faulty environment. Compared to replication, they have typically less overheads. However, they are not very efficient in detecting multiple fault injections. Overhead goes up significantly with the need to detect and/or correct multiple faults. These protection methods are also inspired by similar techniques used in building fault-tolerant systems. Anti-tamper protection modules are also an option.

These can be used to reduce the impact of fault injection attacks. Anti-tamper protection modules can act as scanning tools that look for and report any physical modification attempts. These modules are limited to physical and semi-invasive attacks.

8.6 TIMING ATTACKS

Timing analysis is an SCA that is used to extract critical information about the device under attack by analyzing the execution time of each operation under different setups and input patterns [13]. Every operation performed in a silicon-based device takes a certain amount of time to complete. This time can vary due to the type of operation, the input data, the technology used to build the device, and the properties of the environment, in which the device is operating.

In this section, we show how timing attack is applied, methods used to increase the accuracy of the analysis, what information can be obtained, and countermeasures that could prevent the attack from being successful.

8.6.1 TIMING ATTACKS ON CRYPTOGRAPHIC HARDWARE

An adversary often applies timing analysis on cryptographic systems to extract the secret key, where timing analysis can help the attacker determine which subsets of the key are correct, and which subsets are not. The way an adversary measures the delay of a signal is by applying a change in the input, and recording the delay that occurs before the output is updated. Other techniques include focusing on power or EM signals to analyze the delay; this is mainly used when the device under attack has a sequential circuit, or uses pipelines. Environmental conditions may help an adversary perform effective timing analysis, where different operating temperatures may affect the speed of data flow. For example, higher temperature typically causes a slower data flow, which can help differentiate between parallel or high-speed operations.

Figure 8.12A shows an overview of a naive modular exponentiation algorithm, and Fig. 8.12B shows the total time of 10,000 executions of 3 different modular-exponentiation software implementations: (1) straightforward, (2) square-and-multiply, and (3) Montgomery with square-and-multiply implementations. As shown in this figure, the execution time depends on the exponent. In the case of straightforward implementation, as the exponent increases, the execution time also increases linearly. In other cases, the execution time is related to the number of 1's in the binary exponent number, that is, the Hamming weight of the exponent.

Timing attacks are usually applied along with other side-channel attacks, since more information can be extracted when different analysis methods are employed. Power analysis is one example that works well with timing attacks; the power trace does not only show the pattern in which the operation performed is correlated to, but also how long it took before the operation is completed. The order of operation is also revealed when applying timing analysis to power signals; this order can help identify the type of process the device is running, and may even allow the adversary to reverse engineer the device.

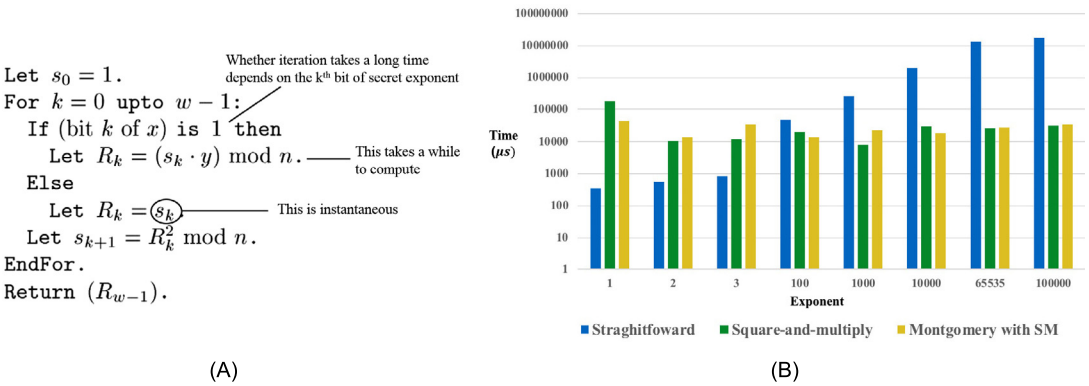


FIGURE 8.12

(A) Square and multiply algorithms used for a modular exponentiation operation, and (B) the total time of 10,000 executions of 3 different modular-exponentiation software implementations.

8.6.2 CACHE-TIMING ATTACK IN PROCESSOR

Another powerful timing analysis attack, called cache-timing attack, is applied to the cache memory of a processor. The main objective of the cache-timing attack is to measure the time for cache access and then to relate the timing values to the information being processed. The cache access time is different for the following two cases: (1) when the requested data by the processor is available in the cache (i.e., cache hit), and (2) when the cache does not have the data available and requests it from the main memory (i.e., cache miss). Retrieving data from the main memory, or from cache levels closer to the main memory in the memory hierarchy of a processor, takes longer time than that from cache levels closer to the processor core. These timing differences have been exploited in SCAs. To measure the time, an attacker can flush the monitored memory line from the cache hierarchy (FLUSH phase), and then wait to allow the victim program to access the memory line (WAIT phase). An attacker then reloads the memory line, measuring the time to load it (RELOAD phase). If the victim accesses the memory line during the wait phase, the reload operation takes a shorter time. Otherwise, the requested line needs to be brought from the memory, and the reload takes significantly longer time. Figure 8.13 shows the timing of the attack phases with and without the victim access. This attack is called `Flush+Reload` attack [23].

8.6.3 TIMING ATTACKS COUNTERMEASURES

To protect devices against timing attacks, designers can do the following: (1) randomize the delay of different operations, or (2) make all operations take the same time, thus preventing information leakage through timing channel. While constant-time implementations can guarantee security against timing attacks, they are not easily achievable in practice. Randomization on the other hand, for instance, by adding random delays to the execution of a task, is easier to accomplish. While it makes the attack more difficult, it cannot, however, guarantee the security of an implementation against timing attacks. Randomization is done by creating various execution paths and adding different delays to different

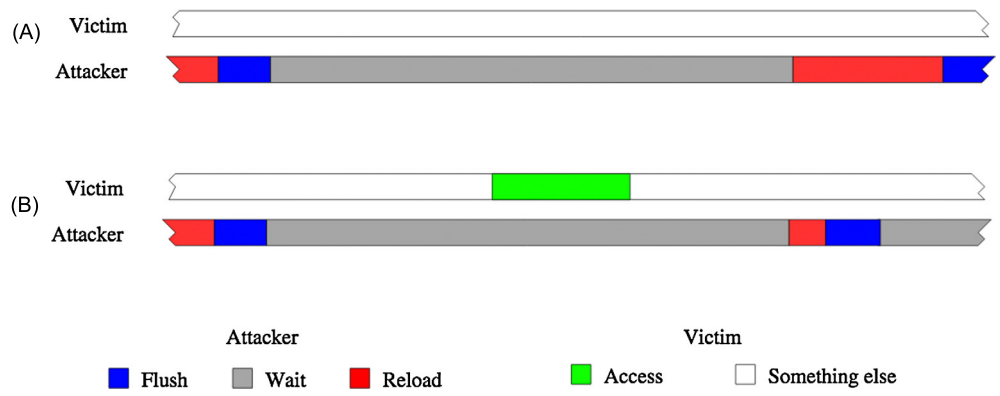


FIGURE 8.13

Timing of `Flush+Reload` attack: (A) without victim access; (B) with victim access [23].

paths. One way to apply delay to a path is to place a series of buffers in the path during circuit design, where the number of buffers can be controlled by the designer to maintain the desired delay.

8.6.4 TIMING LEAKAGE METRICS

In order to evaluate key-dependent timing leakage in a crypto module, test vector leakage assessment methodology can be used. We define timing leakage assessment based on Welch's t-test. Here, it is relevant to repeat that the latter evaluation method is used to test the hypothesis that two populations have equal means when two samples have unequal variance and unequal sample size. For example, two sets of RSA decryptions are defined as the following:

Set 1: n timing measurements during decrypting the same n random ciphertexts with a fixed key: $X_1 = \{t_i | t_i = \#cycle(C_i^{K^*} \mod N), i = 1, \ldots, n\}$.

Set 2: n timing measurements during decrypting the same n random ciphertexts with n random keys: $X_2 = \{t_i | t_i = \#cycle(C_i^{K_i} \mod N), i = 1, \ldots, n\}$. Let the means and variances of X_1 and X_2 be $\bar{X}_1$, $\bar{X}_2$, S_1^2, and S_2^2. If the t-test statistic

$$t = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{S_1^2}{n} + \frac{S_2^2}{n}}} \tag{8.1}$$

is out of the confidence interval, $|t| > T$, then the null hypothesis, $H_0 : \bar{X}_1 = \bar{X}_2$ is rejected. This means that two sets are distinguishable and the implementation has high probability to leak timing information. Thus, it does not pass the leakage assessment test. This is referred to as *nonspecific fixed-vs-random* test. There are other timing leakage tests available in the literature.

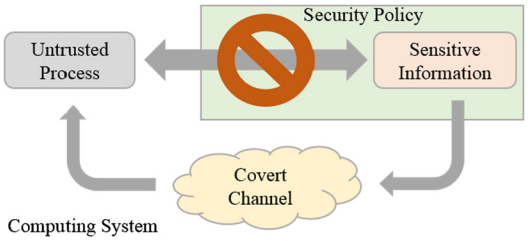


FIGURE 8.14

An overview of a covert channel that bypasses the system security policy and forms an unauthorized communication channel with an untrusted process.

8.7 COVERT CHANNELS

Similar to side channels, covert channels are information leakage channels. However, there are significant differences between their information leakage mechanisms. A covert channel is one that allows communication between software processes that are not authorized to communicate within a system [25]. These communication channels are often not monitored, as security policies may not be able to recognize them. Figure 8.14 illustrates the formation of covert channels. Different types of covert channels have been discovered in modern computing systems. The most common type of covert channels is the storage covert channel, which utilizes the headers of data packets to transfer data. Another type is the timing covert channel, where the communication occurs by modulating allocated resources.

Covert channels can also exist in hardware implementation, where a stealthy malicious circuit with a very rare triggering condition is injected into the system during design phase. When triggered, this circuit can leak sensitive information to the primary outputs of the system, and only the attacker would know how to trigger this circuit.

Compared to SCAs, the exploitation of covert channels is usually more difficult, as an attacker needs to have knowledge about the type and location of sensitive information to leak. This can be accomplished through an attacker's access to the detailed knowledge about the implementation or ability to modify a design at development stage in order to create a covert channel. On the other hand, SCAs typically do not require access to such knowledge. Further, they do not require any design modification, since side-channel signals are naturally generated. Although covert channels are harder to implement, they can typically leak much more sensitive information than side channel analysis.

The detection process of covert channels is very challenging as the number of possible methods for communication can be very large. However, there are mitigation methods that can reduce the number of covert channels. The commonly used method is a validation process that detects and eliminates all malicious functions in the system. The other method is to limit the access to stored data to one process at a time so that other processes cannot form a channel and receive the leaked data.

Side-channel attacks pose a significant threat to the semiconductor industry. Many side-channel analysis attacks cannot be prevented since the origin of these side-channel signals are natural. Many techniques have been evolving in the past decade as the attacker's capabilities have dramatically increased. Table 8.2 shows a summary of the side-channel attacks discussed in this chapter.

Table 8.2 Summary of side-channel attacks

Side-Channel Attack	Measured Parameters	Analysis Methods	Countermeasures
Power Analysis	Current signature and power consumption patterns	Simple power analysis (SPA) Differential power analysis (DPA) Correlation power analysis (CPA)	Power consumption masking Power consumption hiding
EM Analysis	Intentional and non-Intentional electromagnetic emission	Simple EM analysis (SEMA) Differential EM analysis (DEMA)	EM emission shielding EM noise generation modules
Fault Analysis	Invalid outputs, underpowered behavior, and Laser/UV Glitching Responses	Comparative approach to analyze responses before and after fault insertion	Error detection schemes Anti-tamper protection modules
Timing Analysis	Operation delays, time elapsed when different input patterns are applied	Analysis to relate operation delay to the nature of the function	Randomized operational delay Fixed operational delay

The decreasing effort to perform these attacks, and diminishing cost of the measurement instruments, make it easy to exploit side-channel vulnerabilities and help break traditional cryptographic systems. Many countermeasures have been proposed to prevent SCAs. These countermeasures have also evolved over time in their effectiveness and cost, and are being increasingly deployed in new devices to protect against diverse forms of SCAs.

8.8 HANDS-ON EXPERIMENT: SIDE-CHANNEL ATTACK

8.8.1 OBJECTIVE

This experiment is designed to help students explore different types of side-channel attacks for a cryptomodule. The experiment allows students to perform noninvasive side-channel attacks using the HaHa platform.

8.8.2 METHOD

First part of the experiment illustrates the power side-channel attacks, where simple and differential power analysis (SPA and DPA) are applied to an AES design (simplified version) mapped to the FPGA. Students will apply the power analysis while the encryption process is running. Next, the students will capture enough power traces to extract the encryption key of the AES. The second part of the experiment focuses on the fault injection attack, where students will deliberately inject faults into the module in order to leak the encryption key.

8.8.3 LEARNING OUTCOME

By performing the specific steps of the experiments, the students will understand how secret information can leak from inside a chip through different modes of side-channel attacks. They will understand the

side-channel signal measurement and analysis steps, and the physical mechanisms for fault injection. They will also explore the level of information that can be extracted through side-channel analysis.

8.8.4 ADVANCED OPTIONS

Additional exploration on this topic can be done through investigation on how different countermeasures (for example, power balancing or masking) can help mitigate these attacks.

More details about the experiment are available in the supplementary document. Please visit: http://hwsecuritybook.org.

8.9 EXERCISES

8.9.1 TRUE/FALSE QUESTIONS

1. The device under attack does not need to be physically available to the attacker in order to apply the side-channel analysis.
2. Attackers are not often able to perform side-channel attacks due to the high cost of analysis equipment.
3. No prior knowledge of the device's functionality is required to perform power analysis attacks.
4. SPA will always give more information about the device than DPA.
5. Post processing is a step that aims to take out noise and unwanted parts of the power signal.
6. EM emanation is always intentionally done to perform EM analysis attacks.
7. When done properly, EM analysis can provide more information about the device than power analysis.
8. Fault analysis are invasive attacks that usually destroy the device when performed.
9. Timing analysis is only applicable to designs that are purely sequential.
10. Side-channel countermeasures can be applied to the device at any part of its lifetime.

8.9.2 SHORT-ANSWER TYPE QUESTIONS

1. Describe the main idea of side-channel analysis.
2. What information can be gained when successfully applying a side-channel attack?
3. Explain an invasive and noninvasive attack.
4. Explain SPA and DPA. What are the differences between them?
5. Describe what tools and equipment are required to perform power analysis attacks.
6. Describe approaches of protecting a chip against power analysis attacks.
7. What is EM emanation, and how is it used to apply EM analysis attacks?
8. Explain the main objective of fault analysis attacks and how they are performed.
9. Describe the main idea of timing attacks and what tools are required to perform it.
10. How can side-channel attack countermeasures protect the device, and how would these countermeasures affect the performance of the system?

8.9.3 LONG-ANSWER TYPE QUESTIONS

1. Side-channel attacks are applied to any silicon-based system, where a variety of system architectures may be used in the device under attack. How can an attacker perform side-channel analysis on a system that performs parallel-based operations compared to sequentially-based operations?
2. Suppose an attacker is attempting to reverse engineer a device using a side-channel attack. What process should the attacker take to accurately obtain internal information about the device, and what side-channel analysis techniques should be used?
3. Applying EM analysis to capture critical information can be tricky, many factors can affect the quality of the signal. What measures should the attacker take to successfully perform the attack in a noisy environment, and what are the pros and cons of applying EM analysis over power analysis in this case?
4. Attackers can force a device to change a specific internal value using advanced techniques, such as laser injection processes. When applying these techniques to large and complex designs, what measures should the attacker take before applying fault attacks? Where in the design should the attacker start injecting faults, and what information can the attacker leak using this type of attacks?
5. Analyzing the delay of the output of a device can be used to gain knowledge about the design; one countermeasure used is to randomize the delays every time an operation is performed. How can this random delay be implemented? What are the challenges that the designer may face when applying this countermeasure?

REFERENCES

[1] A. Barenghi, L. Breveglieri, I. Koren, D. Naccache, Fault Injection Attacks on Cryptographic Devices: Theory, Practice, and Countermeasures, Proceedings of the IEEE 100 (11) (2012) 3056–3076.
[2] C. Giraud, DFA on AES, in: International Conference on Advanced Encryption Standard, Springer Berlin Heidelberg, 2004.
[3] S. Skorobogatov, R. Anderson, Optical Fault Induction Attacks, in: International Workshop on Cryptographic Hardware and Embedded Systems, Springer Berlin Heidelberg, 2002.
[4] A. Barenghi, G. Bertonit, L. Breveglieri, M. Pellicioli, G. Pelosi, Fault Attack on AES with Single-Bit Induced Faults, in: Information Assurance and Security (IAS), 2010 Sixth International Conference on, IEEE, 2010.
[5] M. Agoyan, J. Dutertre, D. Naccache, B. Robisson, A. Tria, When Clocks Fail: On Critical Paths and Clock Faults, in: International Conference on Smart Card Research and Advanced Applications, Springer Berlin Heidelberg, 2010.
[6] F. Standaert, Introduction to Side-Channel Attacks, in: Secure Integrated Circuits and Systems, Springer, Boston, MA, 2010, pp. 27–42.
[7] Y. Zhou, D. Feng, Side-Channel Attacks: Ten Years After Its Publication and the Impacts on Cryptographic Module Security Testing, IACR Cryptology ePrint Archive 2005 (2005) 388.
[8] P. Kocher, J. Jaffe, B. Jun, Differential Power Analysis, in: Annual International Cryptology Conference, Springer Berlin Heidelberg, 1999.
[9] E. Prouff, DPA Attacks and S-Boxes, in: International Workshop on Fast Software Encryption, Springer Berlin Heidelberg, 2005.
[10] S. Guilley, L. Sauvage, J. Danger, D. Selmane, R. Pacalet, Silicon-level Solutions to Counteract Passive and Active Attacks, in: Fault Diagnosis and Tolerance in Cryptography, 2008. FDTC'08. 5th Workshop on, IEEE, 2008.
[11] S. Guilley, P. Hoogvorst, R. Pacalet, J. Schmidt, Improving Side-Channel Attacks by Exploiting Substitution Boxes Properties, in: International Conference on Boolean Functions: Cryptography and Applications (BFCA), 2007.
[12] W. Hnath, Differential Power Analysis Side-Channel Attacks in Cryptography, Diss., Worcester Polytechnic Institute, 2010.
[13] P. Kocher, Timing Attacks on Implementations of Diffie–Hellman, RSA, DSS, and Other Systems, in: Advances in Cryptology CRYPTO96, Springer, 1996, pp. 104–113.
[14] J. Quisquater, D. Samyde, ElectroMagnetic Analysis (EMA): Measures and Counter-measures for Smart Cards, in: Smart Card Programming and Security, 2001, pp. 200–210.

[15] C. Clavier, D. Marion, A. Wurcker, Simple Power Analysis on AES Key Expansion Revisited, in: International Workshop on Cryptographic Hardware and Embedded Systems, Springer, 2014, pp. 279–297.
[16] E. Brier, C. Clavier, F. Olivier, Correlation Power Analysis with a Leakage Model, in: International Workshop on Cryptographic Hardware and Embedded Systems, Springer, 2004, pp. 16–29.
[17] D. Strobel, F. Bache, D. Oswald, F. Schellenberg, C. Paar, SCANDALee: A Side-ChANnel-based DisAssembLer using Local Electromagnetic Emanations, in: Proc. Design, Automation, and Test in Europe Conf. and Exhibition (DATE), Mar. 2015, pp. 139–144.
[18] J. Courrege, B. Feix, M. Roussellet, Simple Power Analysis on Exponentiation Revisited, in: CARDIS, 2010.
[19] K. Tiri, M. Akmal, I. Verbauwhede, A Dynamic and Differential CMOS Logic with Signal Independent Power Consumption to Withstand Differential Power Analysis on Smart Cards, in: Solid-State Circuits Conference, 2002. ESSCIRC 2002. Proceedings of the 28th European, 2002, pp. 403–406.
[20] K. Tiri, I. Verbauwhede, A VLSI Design Flow for Secure Side-Channel Attack Resistant ICs, in: Proceedings of the Conference on Design, Automation and Test in Europe – Volume 3, DATE '05, IEEE Computer Society, Washington, DC, USA, 2005, pp. 58–63.
[21] Y. Ishai, A. Sahai, D. Wagner, Private Circuits: Securing Hardware against Probing Attacks, in: Advances in Cryptology – CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17–21, 2003, Proceedings, in: Lecture Notes in Computer Science, vol. 2729, Springer, 2003, pp. 463–481.
[22] M. Faruque, S. Chhetri, A. Canedo, J. Wan, Acoustic Side-Channel Attacks on Additive Manufacturing Systems, in: Cyber-Physical Systems (ICCPS), 2016 ACM/IEEE 7th International Conference, Vienna, Austria, 2016, 2016.
[23] Y. Yarom, K. Falkner, FLUSH+RELOAD: A High Resolution, Low Noise, L3 Cache Side, in: Proceedings of the 23rd USENIX Conference on Security Symposium (SEC'14), USENIX Association, Berkeley, CA, USA, 2014, pp. 719–732.
[24] S. Mangard, E. Oswald, T. Popp, Power Analysis Attacks: Revealing the Secrets of Smart Cards, 1st ed., Springer Publishing Company, Incorporated, 2010.
[25] B. Lampson, A Note on the Confinement Problem, Communications of the ACM (1973) 613–615.
[26] M. Rivain, E. Prouff, Provably Secure Higher-Order Masking of AES, in: Cryptographic Hardware and Embedded Systems, CHES 2010, Springer Berlin Heidelberg, 2010, pp. 413–427.