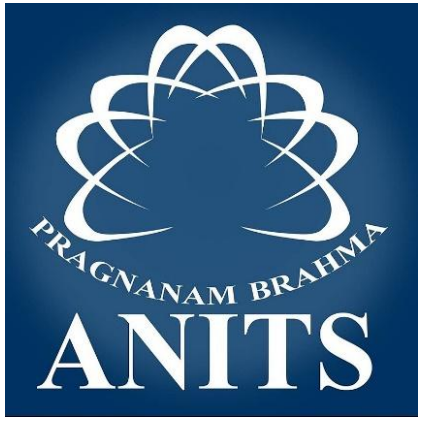
SIDE-CHANNEL ATTACK AGAINST AES ON RASPBERRY PI 4 USING EM ANALYSIS



UNDER GUIDANCE :

AKASH GUPTA
SCIENTIST 'E'

DONE BY :

SAI TARUN BARATAM
INTERN SAG, DRDO

ADVANCED ENCRYPTION STANDARD (AES)

What is Cryptography ??

- Cryptography is the art of protecting information by transforming the original message , called plaintext into an encoded message , called a cipher or ciphertext.

- ABC (meaningful message) -> ZYX (cipher)

What is AES ??

- AES is an encryption standard chosen by the National Institute of Standards and technology (NIST), USA to protect classified information. It has been accepted world wide as a desirable algorithm to encrypt sensitive data.
- It is a block cipher which operates on block size of 128 bits for both encrypting as well as decrypting.
- Each Round performs same operations.

Why AES ??

- In 1990’s the cracking of DES algorithm became possible.
- Around 50hrs of brute forcing allowed to crack the message.
- NIST started searching for new feasible algorithm and proposed its requirement in 1997.
- In 2001 Rijndael algorithm designed by Rijment and Daemon of Belgium was declared as the winner of the competition.
- It met all Security, Cost and Implementation criteria.

How does it works ??

- AES basically repeats 4 major functions to encrypt data. It takes 128 bit block of data and key and gives a ciphertext as outptut. The functions are:

I. Sub Bytes
II. Shift Rows
III. Mix Columns
IV. Add Key

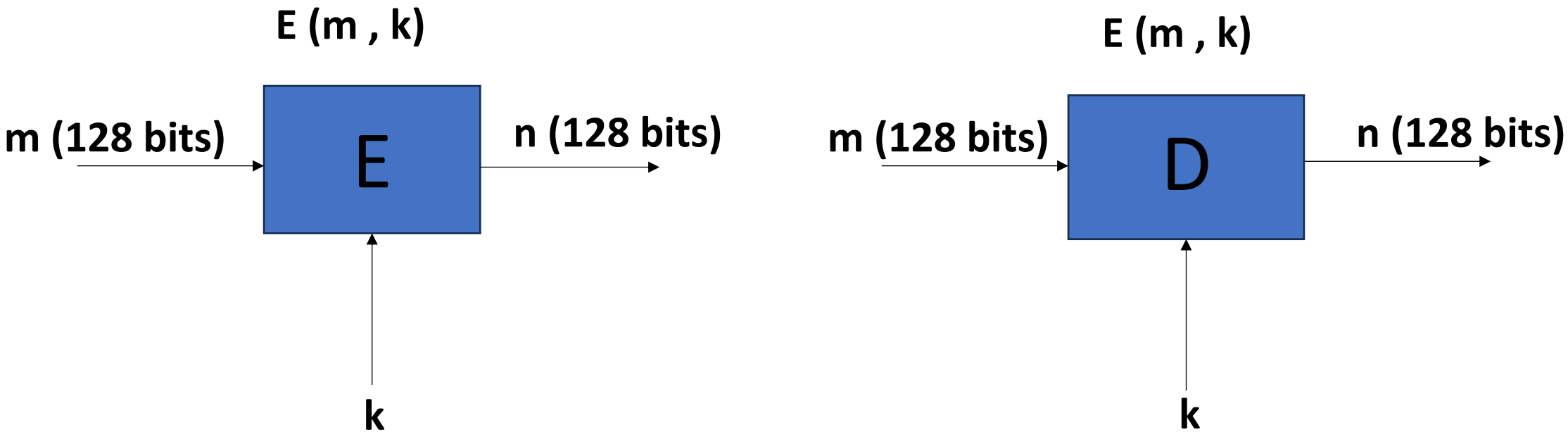
How does it works ??

- The number of rounds performed by the algorithm strictly depends on the size of key.
- The following table gives overview of no. of rounds performed with the input of varying key lengths :

Key Size (in bits)	Rounds
128	10
192	12
256	14

- The larger the number of keys the more secure will be the data. The time taken by software to encrypt will increase with no. rounds.

How does it works ??



Here , E = encryption function for a symmetric block cipher

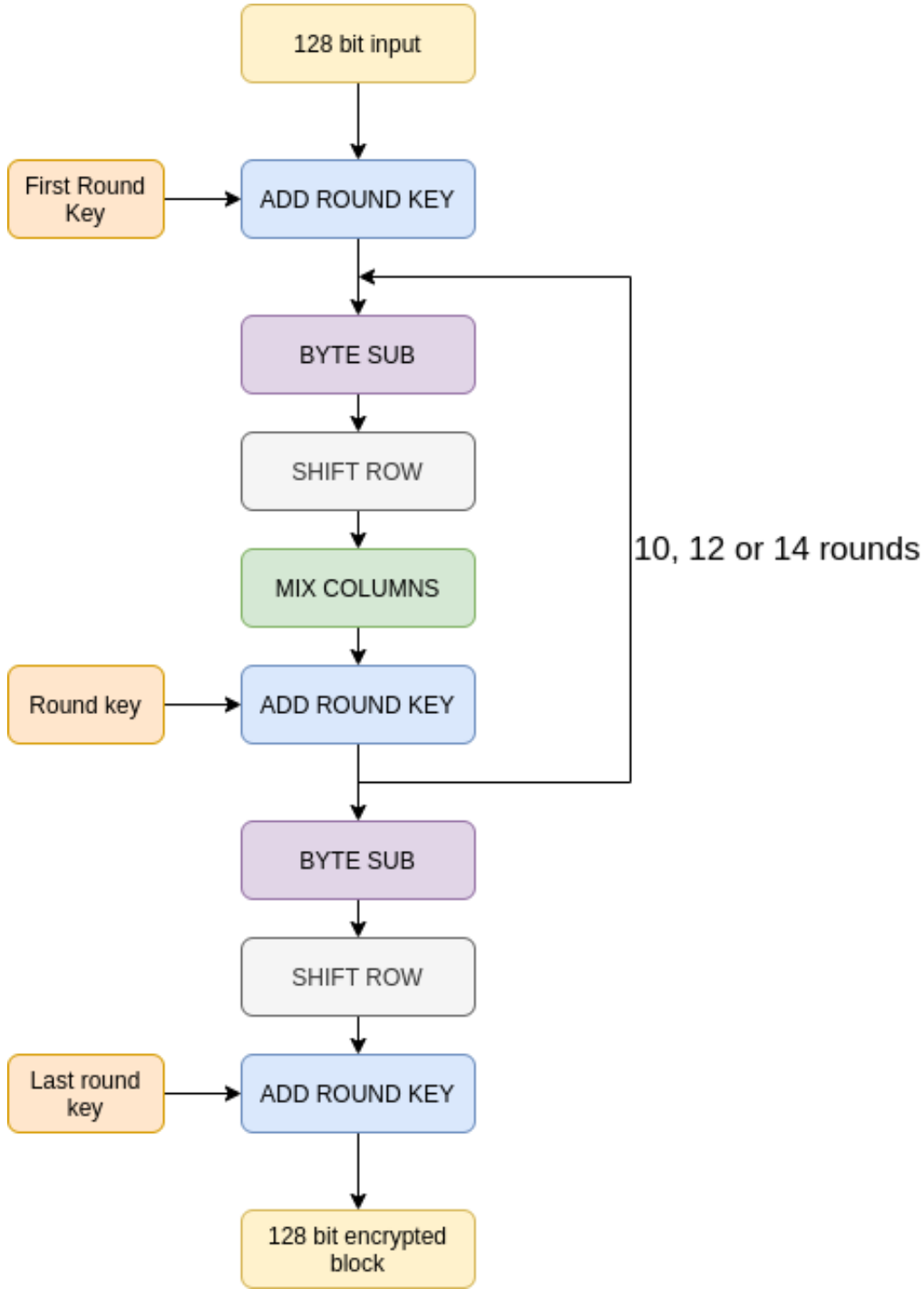
m = plaintext message of size 128 bits

n = ciphertext

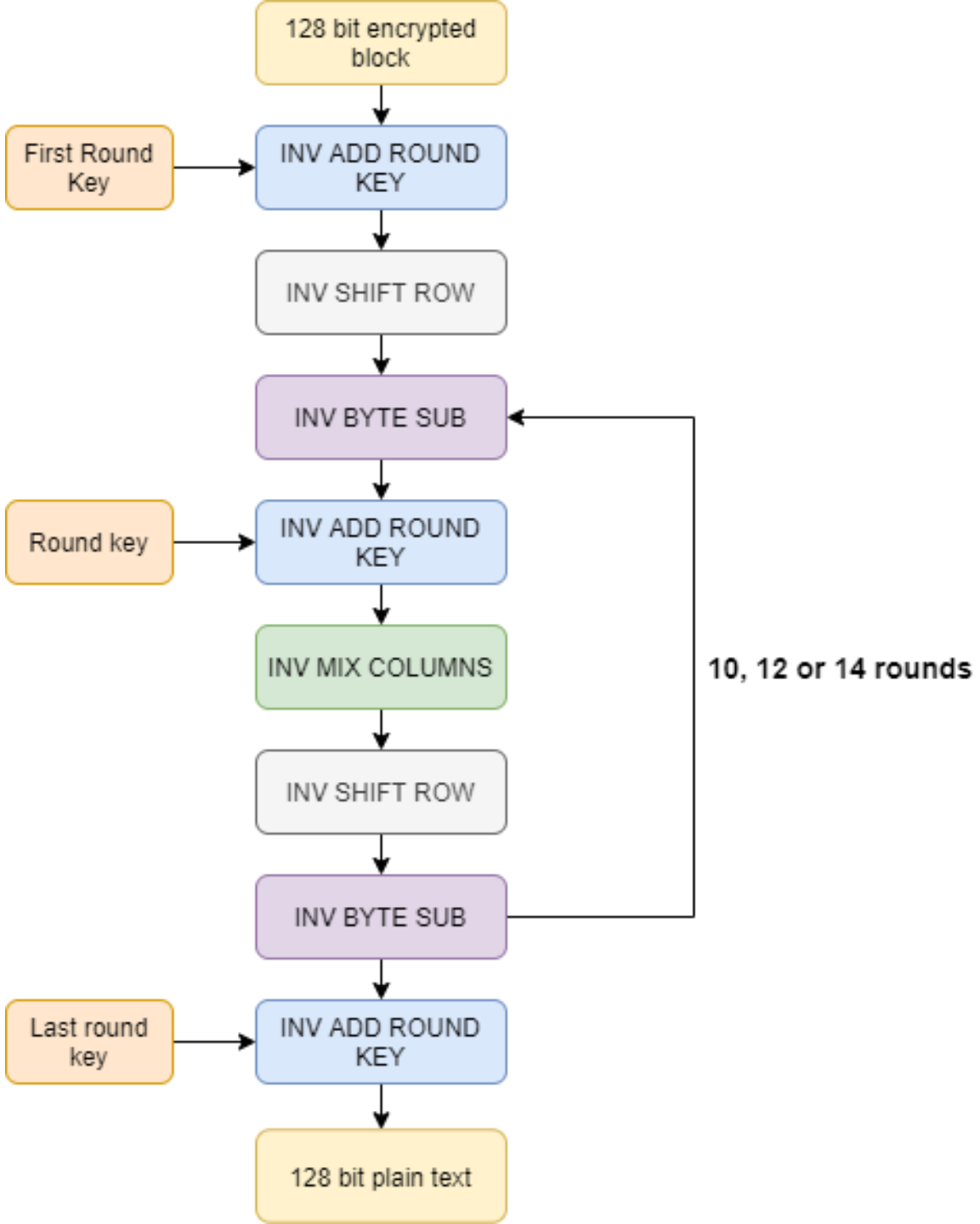
k = key of size 128bits which is same for both encryption and decryption

D = decryption function for symmetric block cipher

Steps for Encryption



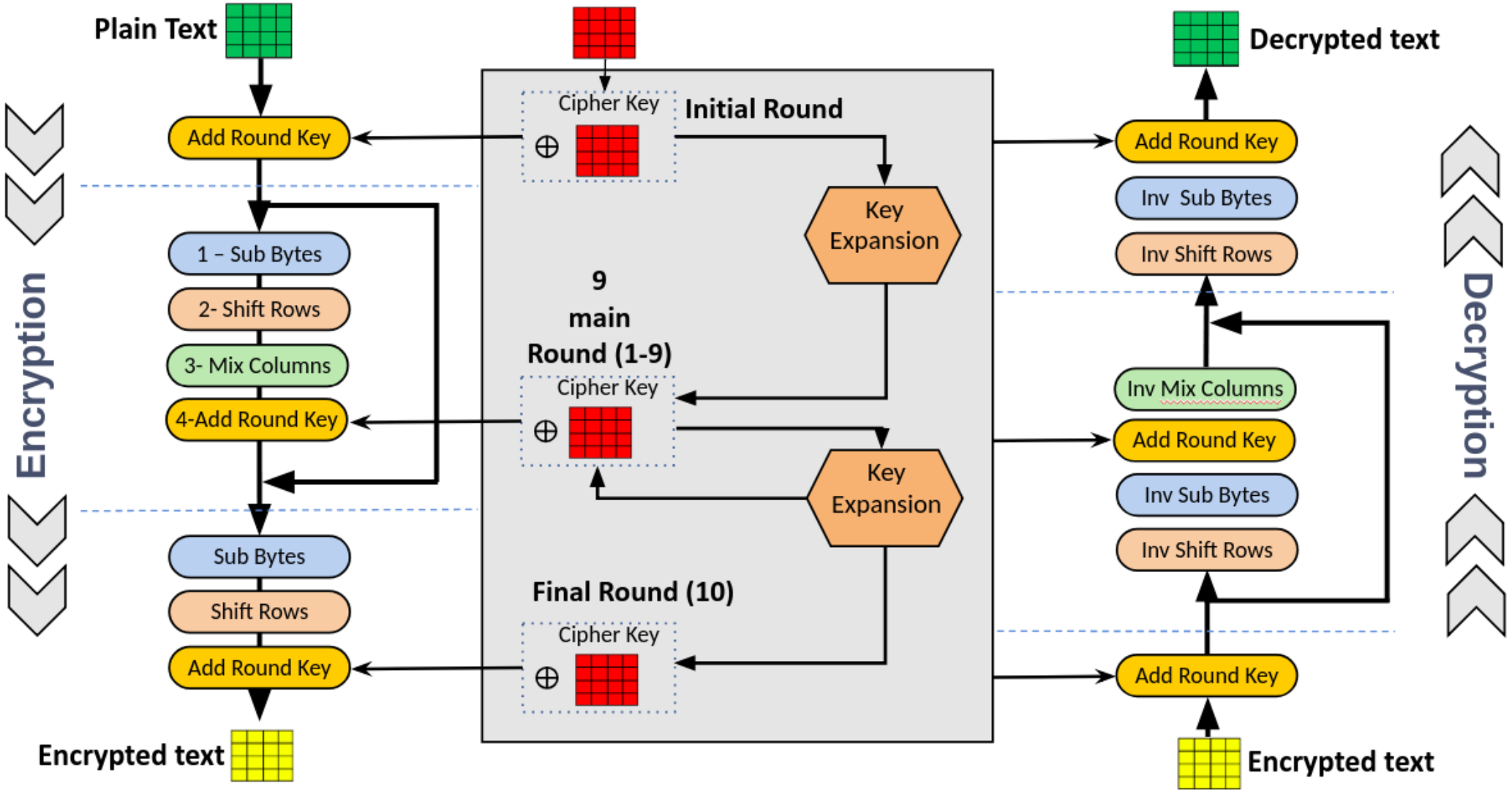
Steps for Decryption



Analysis of Steps

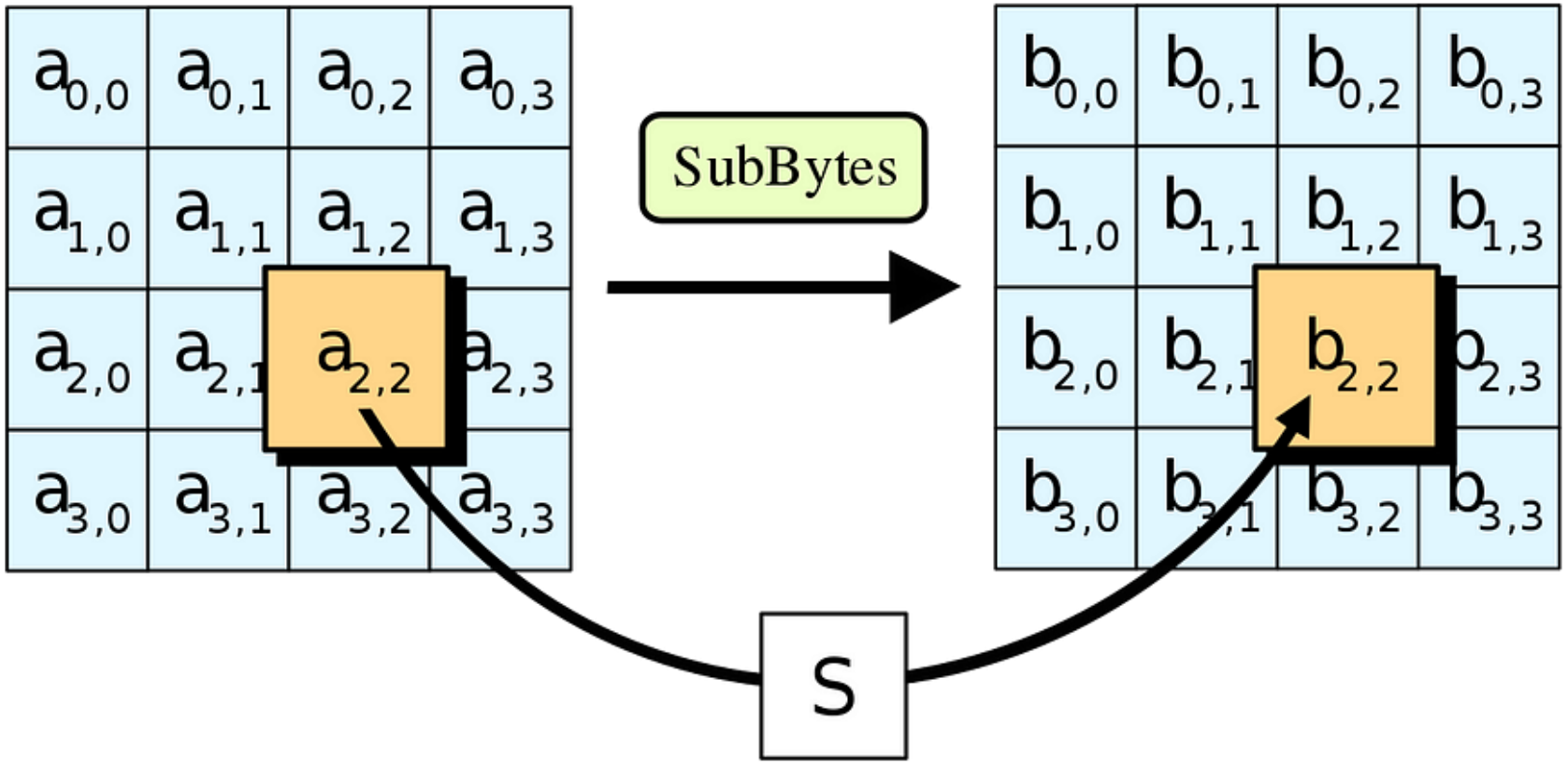
- Key Expansions – In the key expansion process the given 128 bits cipher key is stored in [4]x[4] bytes matrix (16*8=128 bits) and then the four column words of the key matrix is expanded into a schedule of 44 words (44*4=176) resulting in 11 round keys (176/11=16 bytes or 128 bits).
- Number of round keys = Nr + 1. Where Nr is the number of rounds (which is 10 in case of 128 bits key size) So here the round keys = 11.

Analysis of Steps



Sub Bytes

- Sub Bytes – Each element of the matrix is replaced by an element of s-box matrix.



Analysis of Steps

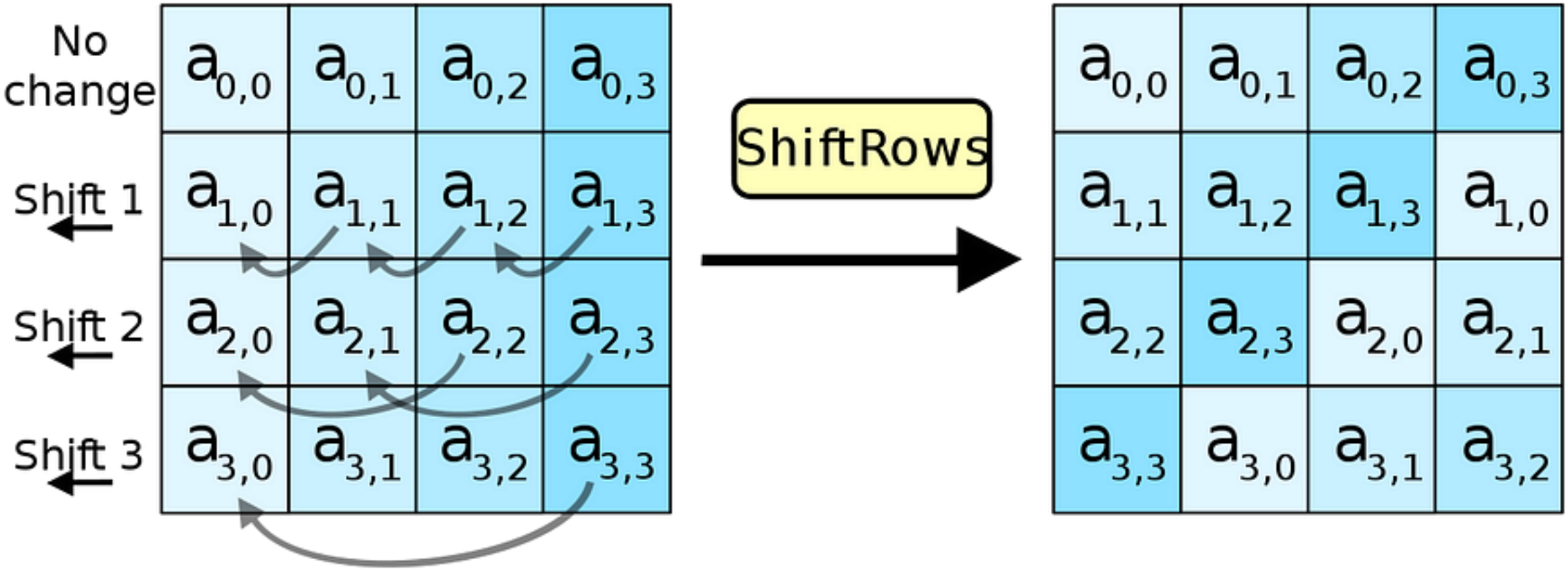
- **Sub Bytes**

For an element {d1} corresponding value is {3e}

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Shift Rows

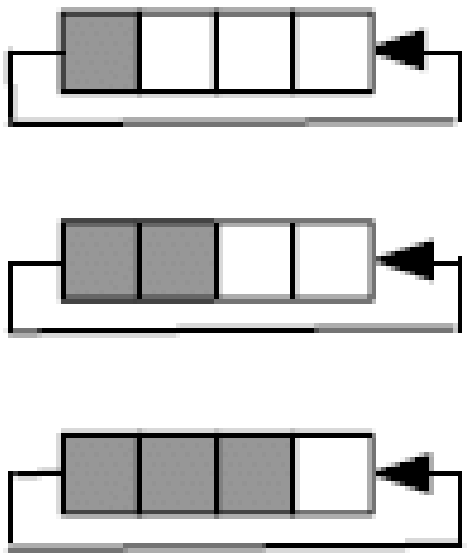
- In this step rows of the block are cylindrically shifted in left direction.
- The first row is untouched, the second by one shift, third by two, and fourth by three.



Shift Rows

S

$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$



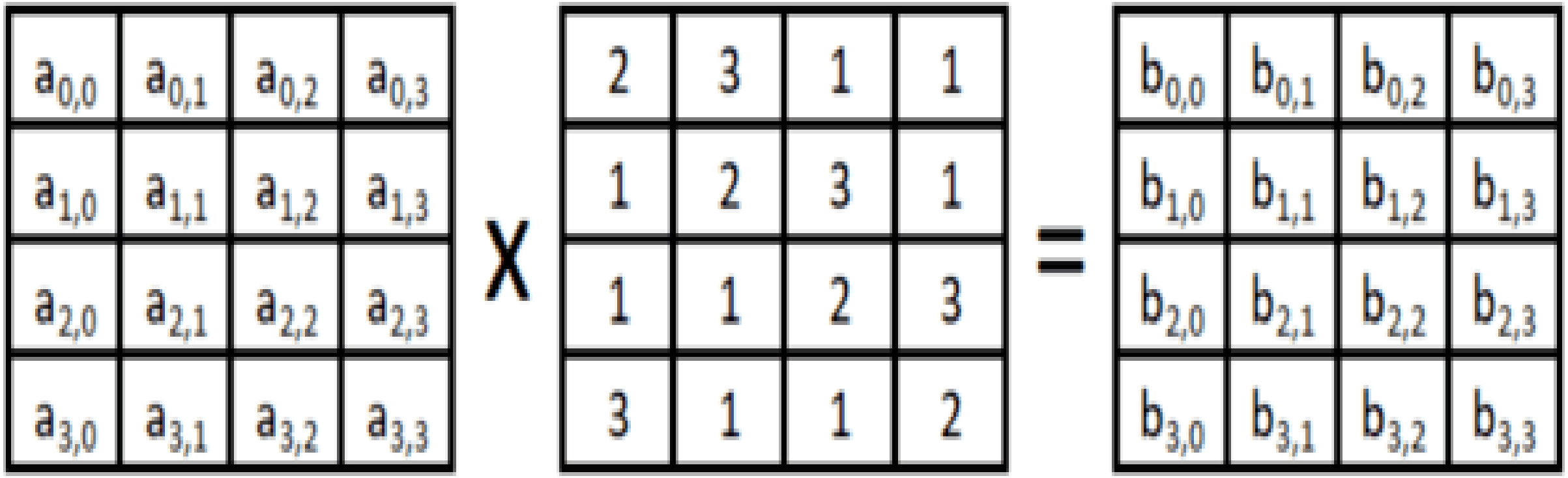
S'

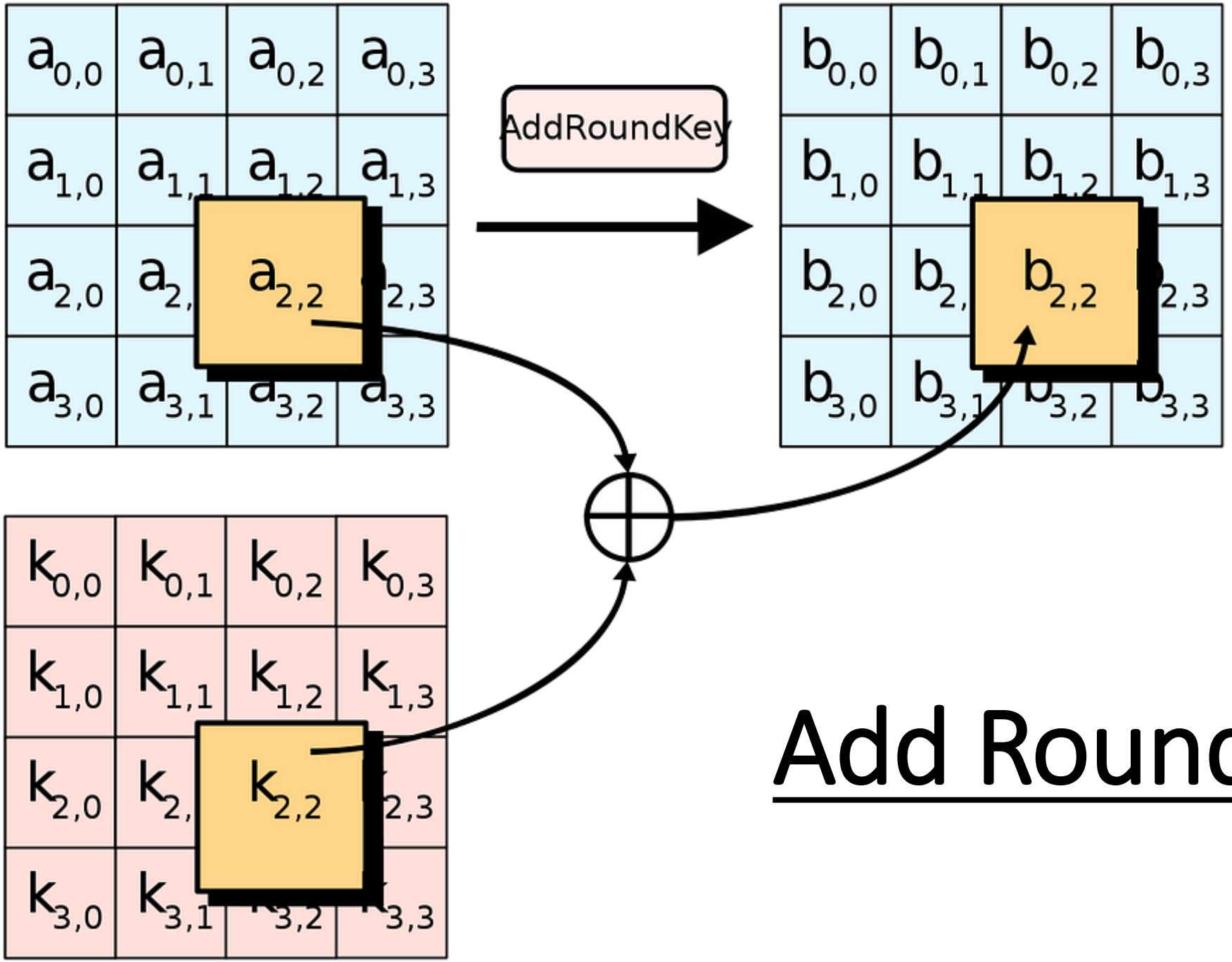
$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,1}$	$s_{1,2}$	$s_{1,3}$	$s_{1,0}$
$s_{2,2}$	$s_{2,3}$	$s_{2,0}$	$s_{2,1}$
$s_{3,3}$	$s_{3,0}$	$s_{3,1}$	$s_{3,2}$

Mix Columns

- This is the most important part of the algorithm.
- It causes the flip of bits to spread all over the block.
- In this step the block is multiplied with a fixed matrix.
- The multiplication is field multiplication in galois field.
- For each row there are 16 multiplication, 12 XORs and a 4 byte output.

Mix Columns


$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,0}$ $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
X
2 3 1 1
1 2 3 1
1 1 2 3
3 1 1 2
=
$b_{0,0}$ $b_{0,1}$ $b_{0,2}$ $b_{0,3}$
$b_{1,0}$ $b_{1,1}$ $b_{1,2}$ $b_{1,3}$
$b_{2,0}$ $b_{2,1}$ $b_{2,2}$ $b_{2,3}$
$b_{3,0}$ $b_{3,1}$ $b_{3,2}$ $b_{3,3}$



Add Round Key

Add Round Key

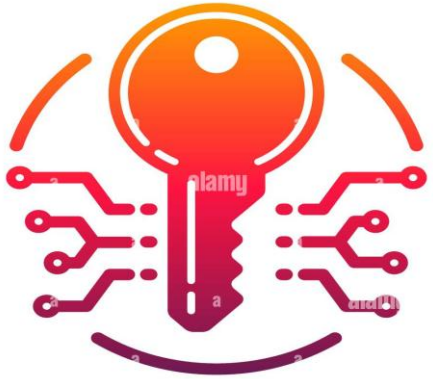
- In this step each byte is XOR-ed with corresponding element of key's matrix.
- Once this step is done the keys are no longer available for this step. Using the same key will weaken the algorithm.
- To overcome this problem keys are expanded.

Last Round

- In the last round the mix column step is skipped.
- It is not documented anywhere why this is done but recently a paper was published against this method highlighting the weakening of cipher text.

Attacks on AES

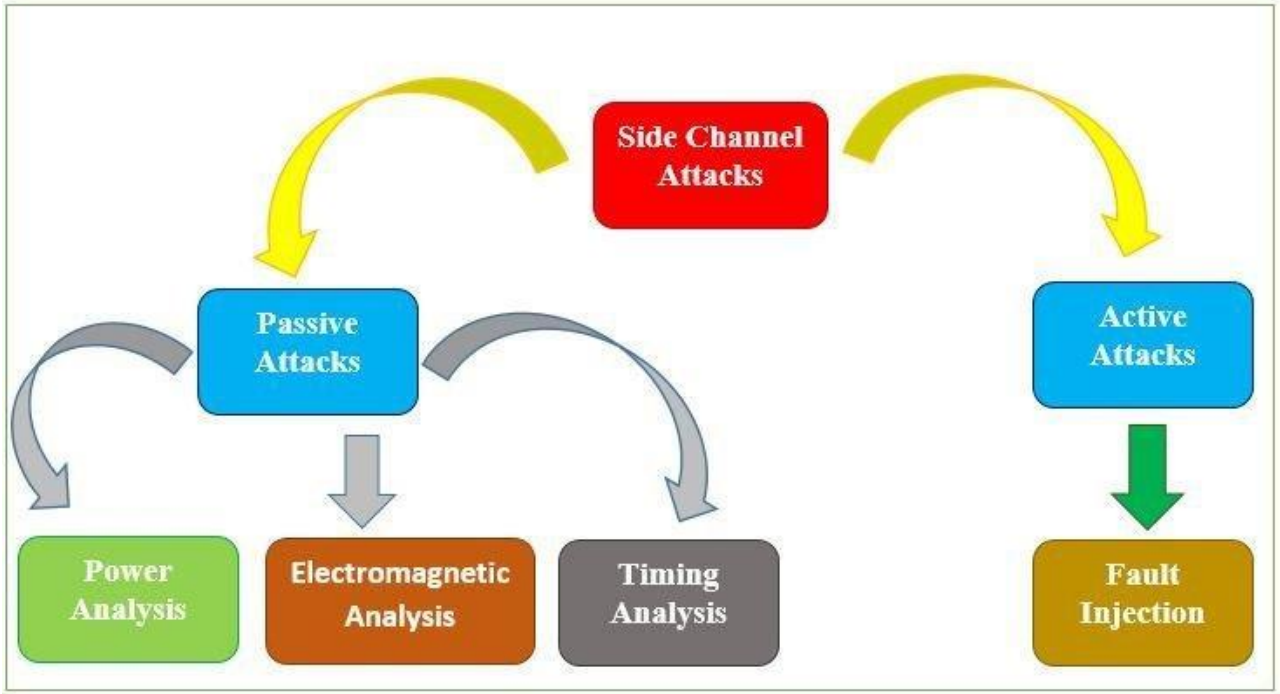
- Brute Force Attack
- Related-Key Attack
- Side-Channel Attack



Side Channel Attacks

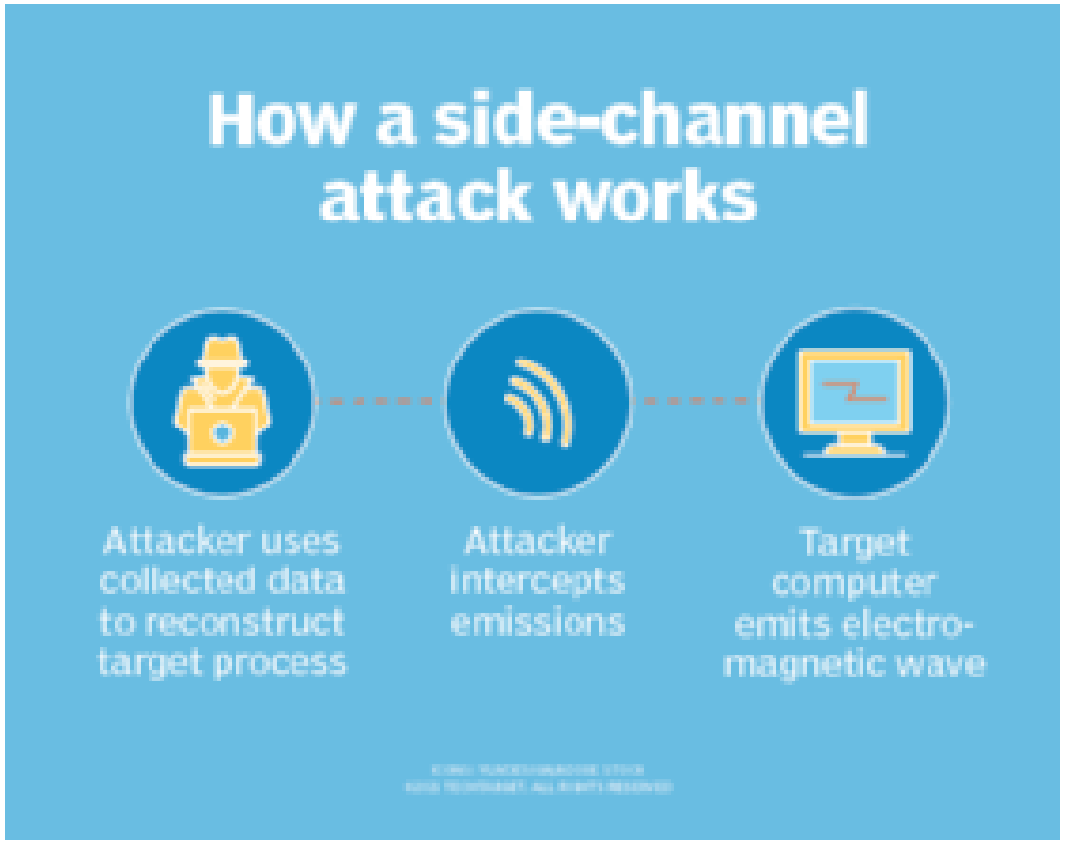
- Side channel attacks are a type of attack that focuses on the physical implementation of a cryptosystem, rather than the cryptographic algorithm itself.
- There are several types of side channel attacks, including :

1. **Timing Attacks**
2. **Power Analysis Attacks**
3. **Electromagnetic Attacks**
4. **Cache Attacks**
5. **Fault Injection Attacks**



Importance of Understanding Side Channel Attacks

- Understanding side channel attacks is crucial because they target the physical implementation of security measures.
- Ignoring these attacks can result in a system that is more vulnerable to them, despite using strong encryption or other security measures.



Power Analysis Attack

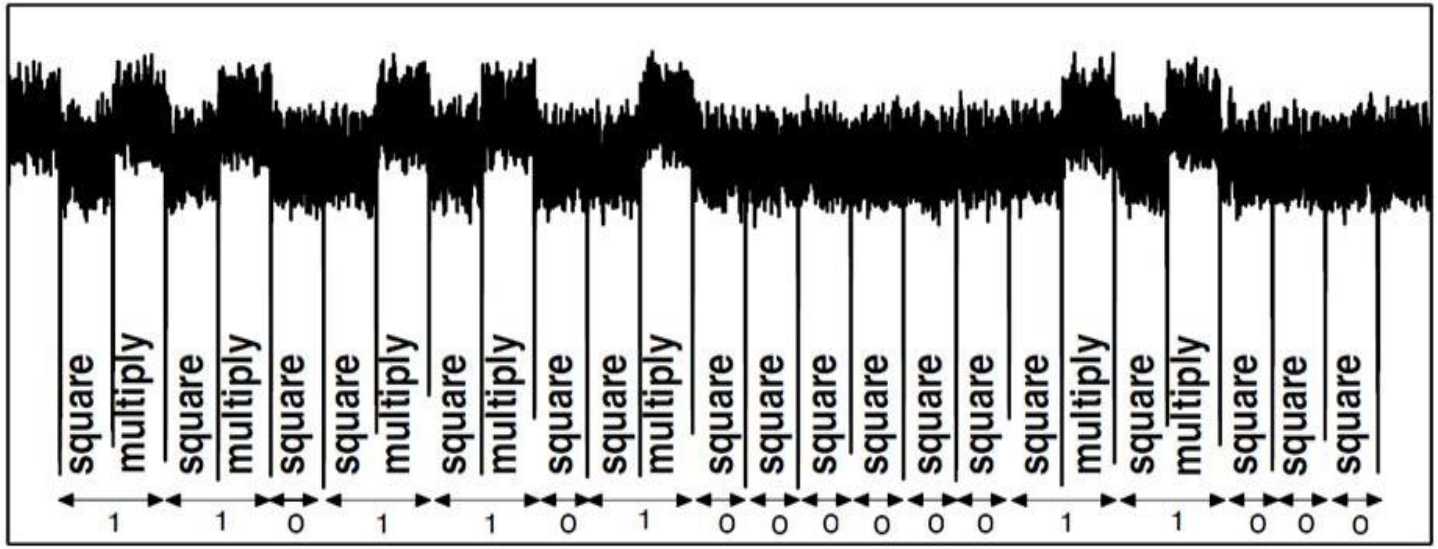
- These attacks measure the power consumption of a device while it is performing cryptographic operations. By analyzing the power consumption patterns, an attacker can infer information about the encryption key.
- Different types of Power Analysis Attacks include

1. **Simple Power Analysis (SPA) Attacks**
2. **Differential Power Analysis (DPA) Attacks**

Simple Power Analysis Attacks (SPA)

- Simple Power Analysis (SPA) attacks are a type of side channel attack that exploits the power consumption of a system during cryptographic operations. By analyzing the power consumption patterns of a system, attackers can deduce information about the system's state or internal data.

Example: square-and-multiply RSA exponentiation.



- Pros:
  - Single trace (or average of a few) may suffice
- Cons:
  - Detailed reverse engineering
  - Long manual part
  - Hard to handle bad signal-to-noise ratio, especially for small events, (e.g., AES)

How SPA Attacks work

- SPA attacks work by analyzing the power consumption patterns of a system during cryptographic operations. By comparing the power consumption patterns of different operations, attackers can deduce information about the system's state or internal data.
- For example, if an attacker knows the power consumption pattern of a system when it is processing a known plaintext, they can compare this pattern to the power consumption pattern of the system when it is processing an unknown plaintext to deduce information about the unknown plaintext.

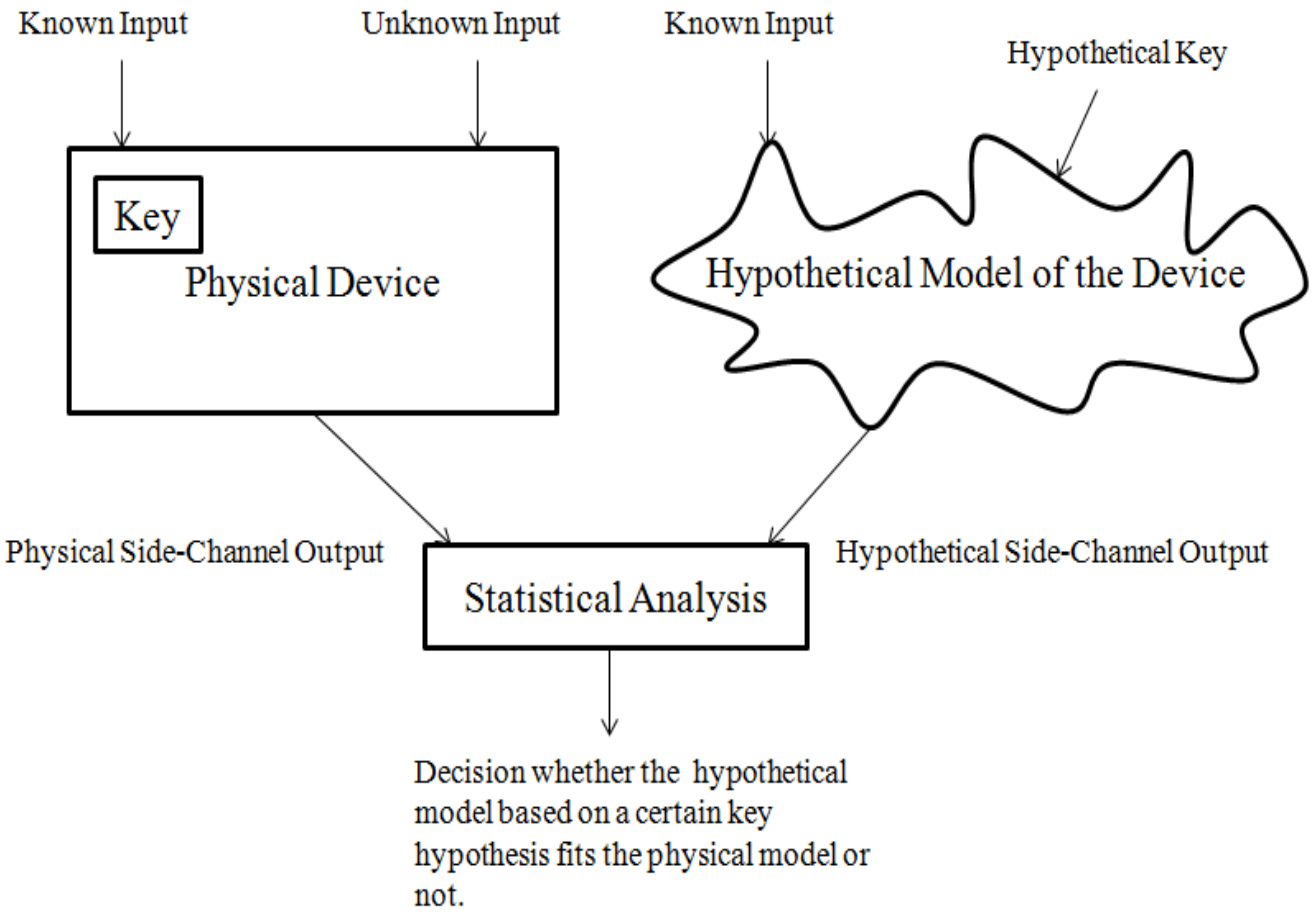
Examples of SPA Attacks

- Examples of SPA attacks include :

1. Power consumption analysis of a smart card during a cryptographic operation.
2. Power consumption analysis of a microcontroller during a cryptographic operation.
3. Power consumption analysis of a hardware security module during a cryptographic operation.

Differential Power Analysis Attacks (DPA)

- Differential Power Analysis (DPA) attacks are a type of side channel attack that exploits the power consumption patterns of a system during cryptographic operations.
- By analyzing the power consumption patterns of a system over multiple operations, attackers can deduce information about the system's internal data or cryptographic keys.

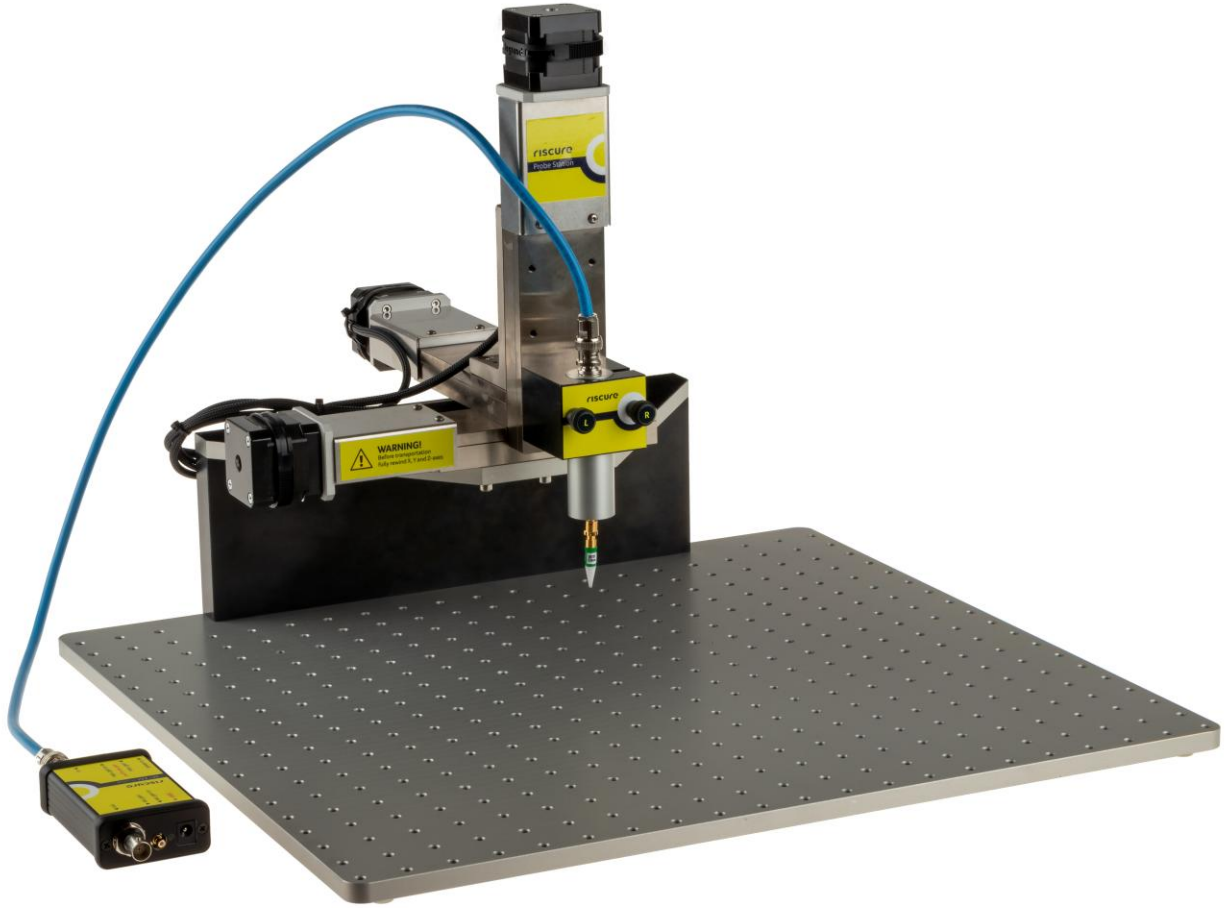


How DPA Attacks work

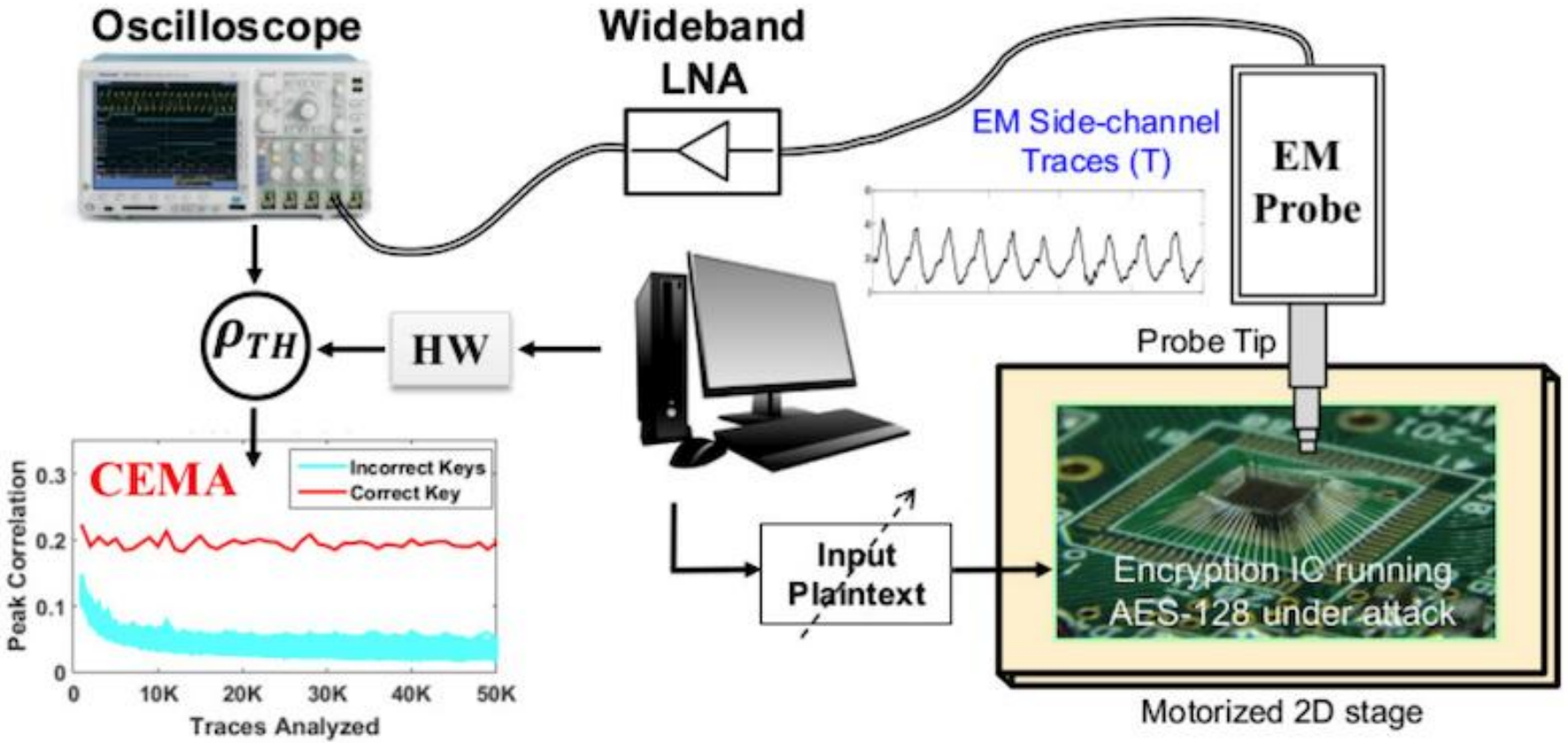
- DPA attacks work by analyzing the power consumption patterns of a system over multiple operations.
- By comparing the power consumption patterns of different operations, attackers can deduce information about the system's internal data or cryptographic keys.
- DPA attacks use statistical methods to analyze the power consumption patterns, making them more powerful than SPA attacks.

Electromagnetic (EM) Attack

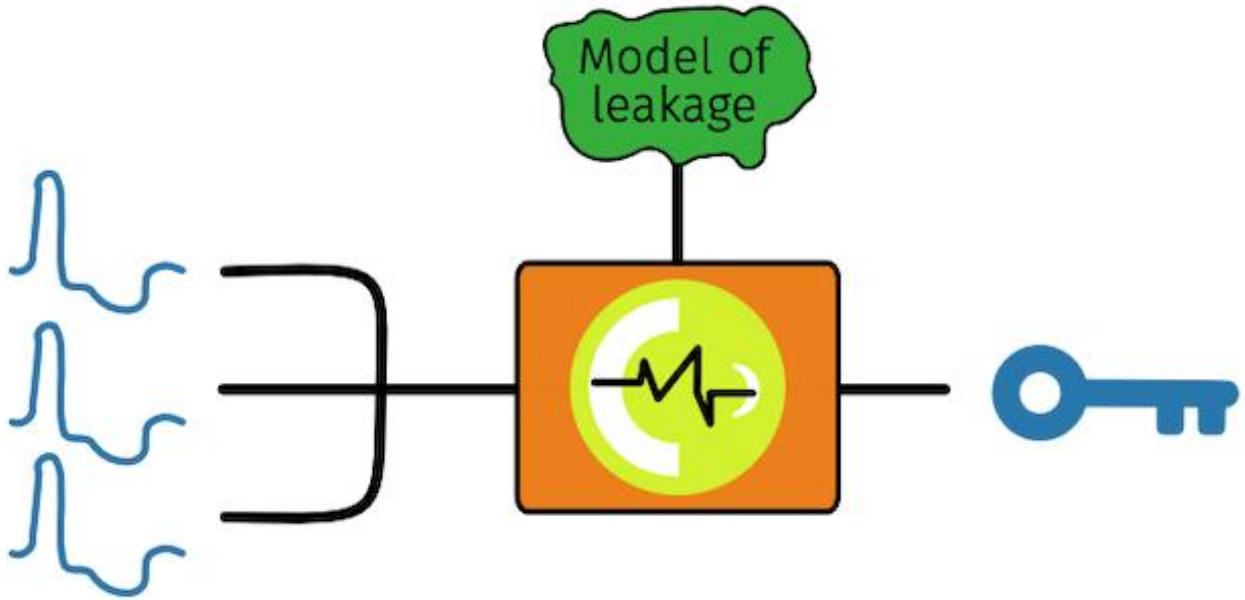
- These attacks measure the electromagnetic radiation emitted by a device while it is performing cryptographic operations.
- By analyzing the electromagnetic radiation patterns, an attacker can infer information about the encryption key.



Example Setup for an EM Attack



How EM Attacks work



EM-based attacks are used to extract information from the data.

Examples of Electromagnetic (EM) Attacks

- Examples of electromagnetic attacks include :

1. Near-field scanning of a smart card during a cryptographic operation.
2. Far-field scanning of a microcontroller during a cryptographic operation.
3. Near-field scanning of a hardware security module during a cryptographic operation.

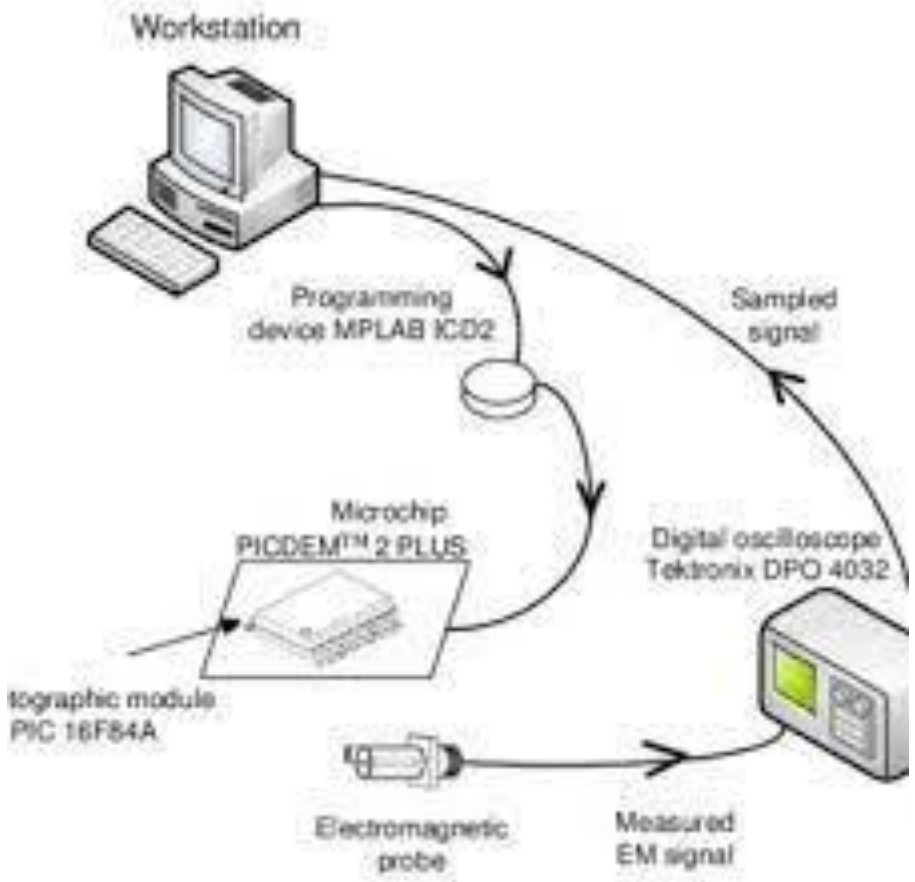
Types of EM Side-Channel Attacks

- Similar to power attacks, EM attacks can be divided into two main categories :

1. **Simple Electromagnetic Analysis (SEMA)**
2. **Differential Electromagnetic Analysis (DEMA)**

Simple EM Analysis

- SEMA is a type of side-channel attack that involves measuring the electromagnetic radiation emitted by a device during cryptographic operations.
- SEMA attacks attempt to interpret the data traces directly to extract sensitive information.

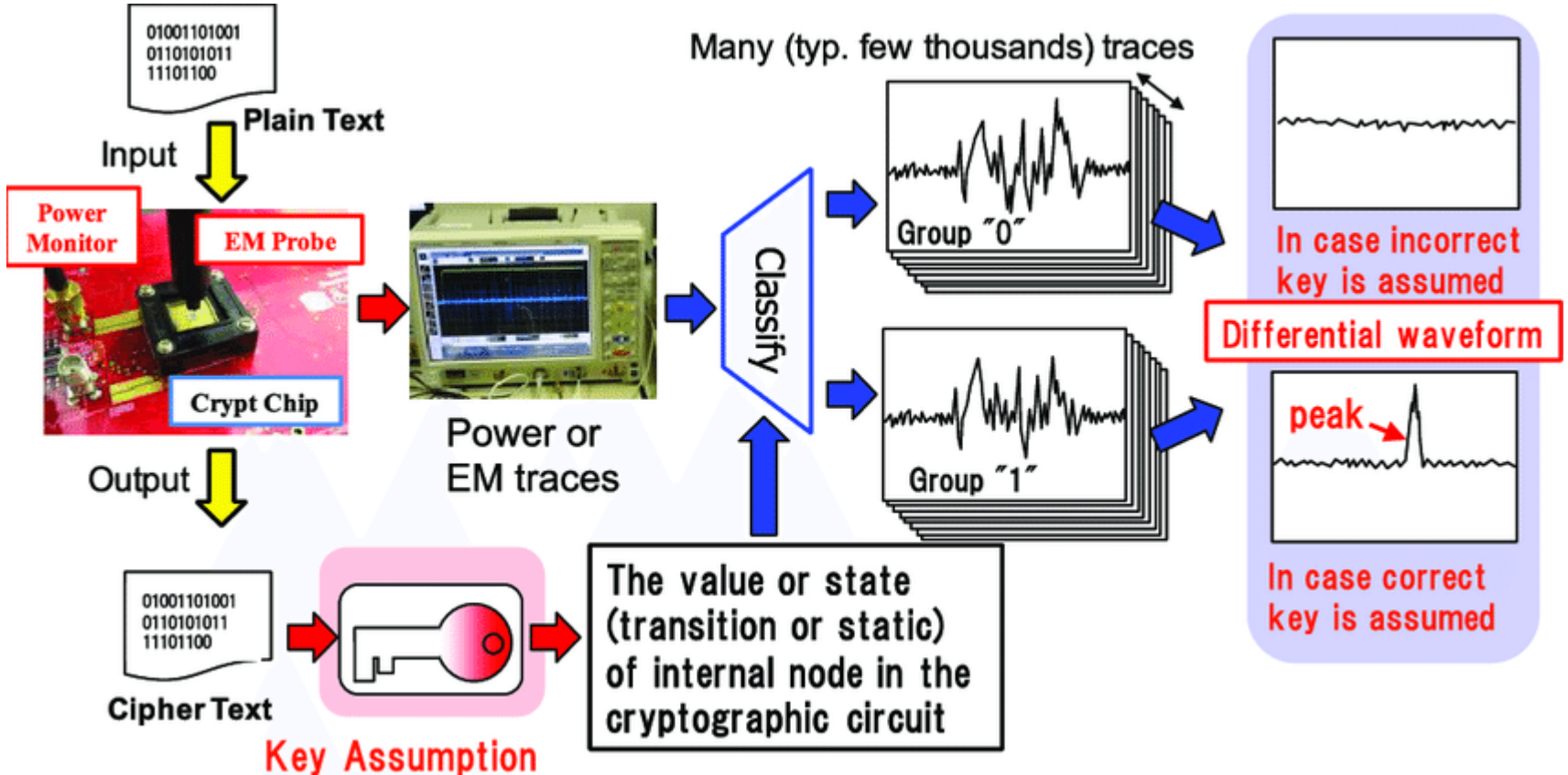


How SEMA Attack works

- SEMA attacks measure the electromagnetic radiation emitted by a device during cryptographic operations. By analyzing the data traces, attackers can extract sensitive information, such as encryption keys.
- SEMA attacks are typically non-invasive and passive, meaning that they do not require physical access to the device or modification of its hardware or software.

Differential EM Analysis

- DEMA is a more advanced type of side-channel attack that involves collecting large numbers of data traces and running differential statistical methods on them to identify data-dependent correlations.



How DEMA Attack works

- DEMA attacks involve collecting large numbers of data traces and analyzing them to identify correlations between the electromagnetic radiation emitted by a device and the data being processed. By analyzing the correlations, attackers can extract sensitive information, such as encryption keys.
- DEMA attacks are typically more robust and powerful than SEMA attacks, but they are also more complicated and time-consuming.

Electronic Devices Vulnerable to EM Attacks

- Some of the devices that have been shown to be susceptible to EM-based side-channel attacks include:

1. **Smart Cards**
2. **Mobile Payment Systems**
3. **Embedded Devices**
4. **Smartphones**
5. **IoT Devices**
6. **SoCs**

Bare Metal Implementation

- Bare metal programming is the practice of programming directly on the hardware, without the use of an operating system or middleware.
- In this approach, developers write code that interacts directly with the hardware, accessing registers and memory locations to control system behaviour.
- Bare metal programming is typically used for low-level tasks, such as device driver development and hardware initialization.
- **Pros of Bare Metal Programming :**

1. Performance
2. Efficiency
3. Simplicity

Choice of Hardware – Raspberry Pi 4B

- Raspberry Pi 4 Model B is a low-cost computer that runs on a 1.5 GHz 64-bit quad-core Arm Cortex-A72 processor.
- It comes in two variants, with 2GB, 4GB, or 8GB of LPDDR4 SDRAM.
- It has dual-band Wi-Fi and Bluetooth 5.0, making it easier to connect to the internet and other devices.
- It has two micro-HDMI ports, which support 4K resolution at 60Hz.
- It has a Gigabit Ethernet port, which provides faster wired internet connectivity.
- It has two USB 3.0 ports and two USB 2.0 ports, allowing for faster data transfer.
- It has a 40-pin GPIO header, which can be used to connect sensors, LEDs, and other components.
- It has a microSD card slot, which can be used for booting the operating system and storing data.
- It has a power management circuit, which allows for more efficient power consumption.
- It has a thermal management system, which helps to dissipate heat and prevent overheating.

Hardware Prerequisites

- An RPi4 with a dedicated power supply and HDMI lead.
- A monitor/TV connected to the RPi4 via HDMI.
- A micro-SD card to boot the RPi4 from.
- A USB keyboard and mouse.
- A computer to write your code on e.g. a Windows/Mac laptop (the dev machine).

Software Prerequisites

- A cross-compiler, such as GCC, that can compile code for the ARM architecture.
- A toolchain, such as the Raspberry Pi Tools, that includes the necessary libraries and utilities.
- A text editor or integrated development environment (IDE) for writing and editing code.
- A programmer's manual or reference guide for the Raspberry Pi 4, such as the Broadcom BCM2711 Peripherals datasheet.

Setting Up the Toolchain

- Download and install the Raspberry Pi Tools, which include the cross-compiler and other tools.
- Create a new project directory and initialize a new build system, such as CMake or Make.
- Write the code for the bare metal program, using the ARM assembly language or C language.
- Compile the code using the cross-compiler, generating a binary file that can be loaded onto the Raspberry Pi 4.

Equipment

- Raspberry Pi 4 board, with at least 1GB of RAM.
- A microSD card, with at least 8GB of storage.
- A USB keyboard and mouse.
- A monitor, with an HDMI cable.
- A power supply, with a micro-USB cable.
- An EM probe, such as the ELV Elektronik EM Probe Set.
- A digital oscilloscope, such as the Rigol DS1054Z.
- A computer, with Ubuntu VirtualBox installed, and the GCC-ARM compiler.

Data Collection and Processing

- Write the bare metal code for the Raspberry Pi 4, using the ARM assembly language or C language.
- Compile the code using the GCC-ARM compiler, generating a binary file that can be loaded onto the Raspberry Pi 4.
- Load the binary file onto the Raspberry Pi 4 and run the AES encryption algorithm.
- Use the EM probe to collect the electromagnetic radiation emitted by the Raspberry Pi 4.
- Use the digital oscilloscope to acquire and digitize the EM signals.
- Save the acquired data as a CSV file.
- Preprocess the CSV file to remove noise and normalize the data.
- Use the preprocessed data for the side-channel attack.

Attack Procedure

- Alignment: Align the traces to a common point in time, such as the start of the encryption algorithm.
- Correlation Power Analysis (CPA): Correlate the power consumption of the Raspberry Pi 4 with the Hamming weight of the AES key.
- Use the Pearson correlation coefficient to measure the similarity between the power consumption and the Hamming weight.
- Repeat the correlation for all possible key values.
- Select the key value with the highest correlation coefficient as the correct key.
- Verify the correctness of the key by decrypting a known plaintext using the extracted key.

Result and Conclusion

- **Result**

1. Present the results of the side-channel attack, such as the number of traces required to extract the key, the accuracy of the key extraction, and the time required for the attack.
2. Discuss the limitations of the attack, such as the need for physical access to the device, the dependence on the quality of the EM probe and oscilloscope, and the potential countermeasures to prevent the attack.

- **Conclusion**

1. Demonstrate the feasibility of a side-channel attack against AES on the Raspberry Pi 4 using an EM analysis attack.
2. Highlight the importance of physical security in embedded systems and the need for countermeasures against side-channel attacks.
3. Encourage further research in side-channel attacks and countermeasures for embedded systems.