

AES Wireless Keyboard

- Template Attack for Eavesdropping -

Kwonyoup Kim, Tae Hyun Kim, Taewon Kim, Sangryeol Ryu

SNT Works Inc., Rep. of Korea

<http://www.sntworks.kr>

March 22th, 2018

About us

- Kwonyoup Kim, Founder, CEO of SNT Works Inc.
- Tae Hyun Kim, PhD., CTO of SNT Works Inc.
- **SNT Works**
 - Specialized in Security Analysis on Embedded Systems
 - ✓ Reverse Engineering for Security Evaluation, Assessment
 - ✓ Side Channel Attack & Fault Attack
 - ✓ Patent infringement Investigation

Previous Related Works of Wireless Keyboard & Mouse

- KeyKeriki v1.0 and v2.0 by Dreamlab Technologies, 2010
- Promiscuity is the nRF24L01+'s Duty, Travis Goodspeed, 2011
- KeySweeper, Samy Kamkar, 2015
- MouseJack, Bastille Networks Internet Security, 2016
- Deeg & Klostermeier, Of Mice and Keyboards : On the Security of Modern Wireless Desktop Sets, Hacktivity 2016

Motivations

- Deeg & Klostermeier's Results, Hacktivity 2016

- Found Security Vulnerabilities

1. Insufficient protection of code (firmware) and data (cryptographic key)
2. Unencrypted and unauthenticated data communication
3. Missing protection against replay attacks
4. Insufficient protection against replay attacks
5. Cryptographic issues

- Manufacturer Feedback

"As you called out in your email, given each wireless desktop set has different cryptographic key which makes this attack not generic at all. It also requires physical access to the keyboard and sniffer to capture packets to decrypt with obtained key. If you can open keyboard and dump flash from it you can as well change the whole board. Hence, this doesn't meet security servicing bugbar. We have opened a bug in the next version of the product for the core team to evaluate."

(concerning insufficient protection of code and data)

Our Goals

- Eavesdropping of AES Wireless Keyboard without physical access
 - Reverse Engineering of AES Wireless Keyboard
 - ✓ Construction of SCA(Side Channel Attacks) environment
 - Side Channel Attacks of AES Wireless Keyboard
 - ✓ EM Side Channel Attacks
 - ✓ Recover secret 128 bits AES keys
 - Building templates using the recovered key
 - Attacking the other keyboard with different unknown key



AES Wireless Keyboard

- Reverse Engineering -

Reverse Engineering of AES Wireless Keyboard

- **Firmware & NVRAM Acquisition**
 - Keyboard (nRF24LE1)
- **Static Analysis**
 - Disassembling firmware (IDA Pro)
 - Finding Encryption functions
- **Dynamic Analysis**
 - Debugging Codes (nRFgo Studio)
 - Verifying Encryption functions (IV, Salt, Key, Enc-Mode, ...)
- **Firmware Modification**
 - Modifying Codes for Repeated execution
 - Reprogramming of Keyboard firmware



Reverse Engineering of AES Wireless Keyboard

- Firmware Acquisition – Keyboard

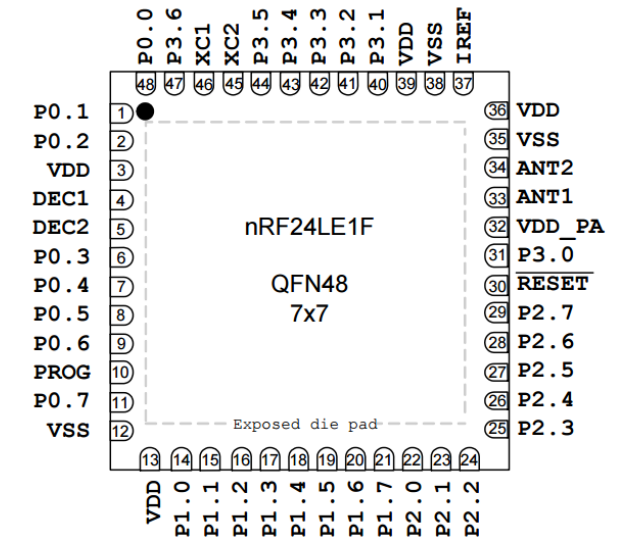
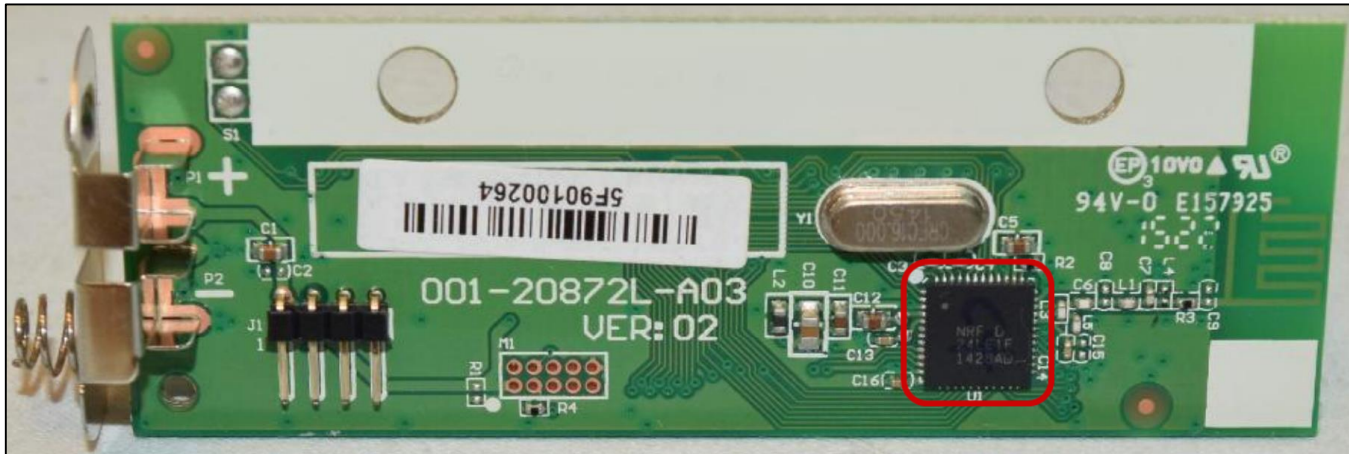


Figure 4. nRF24LE1F pin assignment (top view) for a QFN48 7x7 mm package

Reverse Engineering of AES Wireless Keyboard

■ Firmware Acquisition – nRF24LE1

• Flash memory : features

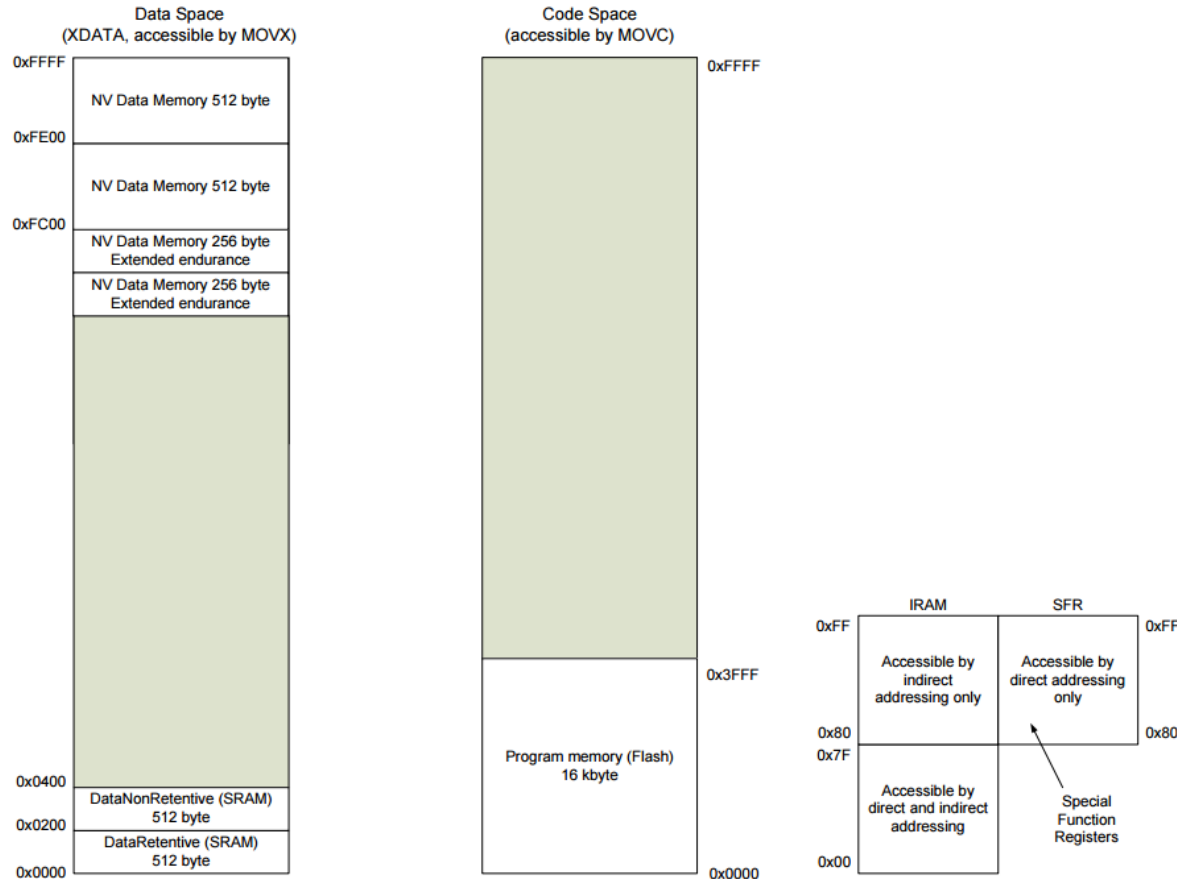


Figure 30. Memory map

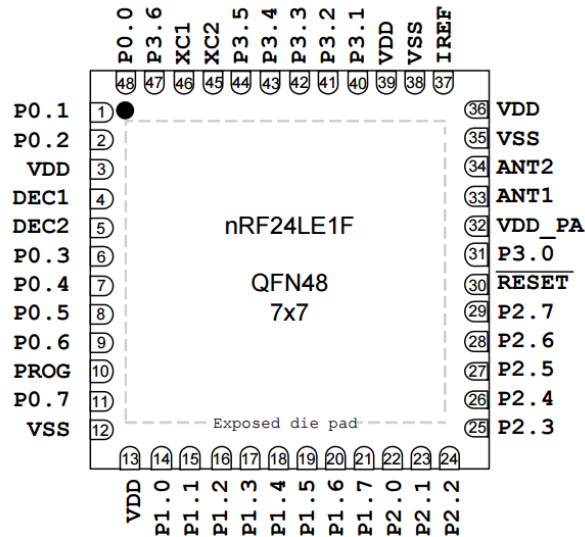
- 16 kB code memory
- 1k NV data memory
- Page size 512 bytes for NV data memory and program memory
- Two pages of 256 bytes each for extended endurance memory
- 32 pages of main block + 1 InfoPage
- Endurance minimum 1000 write/erase cycles
- Extended endurance memory, minimum 20000 write/erase cycles
- Direct SPI programmable
- Configurable MCU write protection
- Readback protection
- HW support for FW upgrades

Reverse Engineering of AES Wireless Keyboard

- Firmware Acquisition – Pinout



NA	FCSN	RESET	FSCK	GND
VCC	PROG	FMOSI	FMISO	NA



6.3.5 Flash programming through SPI

The on-chip flash is designed to interface a standard SPI device for programming. The interface uses an 8-bit instruction register and a set of instructions/commands to program and configure the flash memory.

6.3.5.1 SPI slave interface

To program the memory the SPI slave interface is used. SPI slave connection to the flash memory is activated by setting pin `PROG` = 1 while the reset pin is kept inactive. After the `PROG` pin is set to 1, apply a pulse on the `RESET` pin (Pull `RESET` pin low for a minimum of 0.2 μ s and return to high). Selected nRF24LE1 GPIO pins are automatically configured as a SPI slave as shown in [Table 33](#). Further information on SPI slave timing can be found in [chapter 18 on page 147](#).

	24pin-4x4	32pin-5x5	48pin-7x7
FCSN	P0.5	P1.1	P2.0
FMISO	P0.4	P1.0	P1.6
FMOSI	P0.3	P0.7	P1.5
FSCK	P0.2	P0.5	P1.2

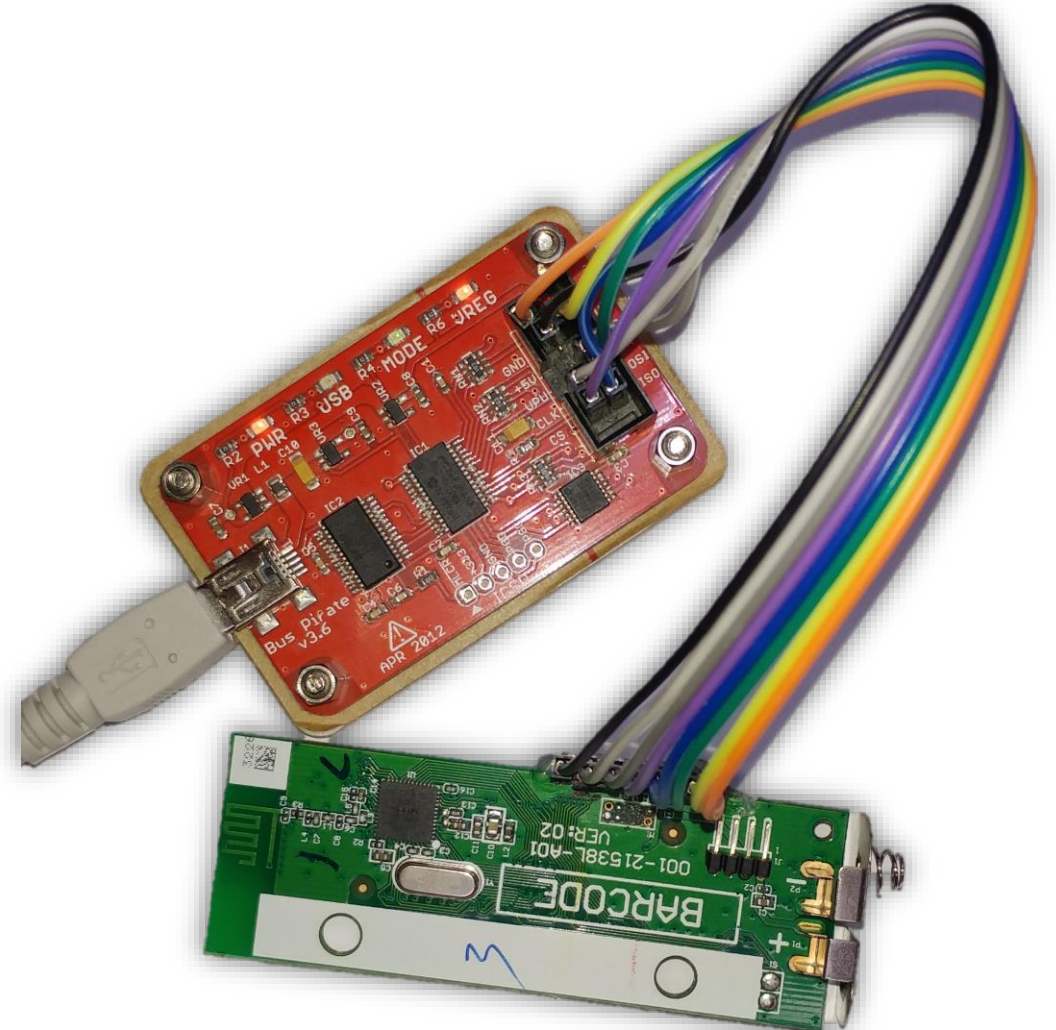
Table 33. Flash SPI slave physical interface for each nRF24LE1 package alternative

Reverse Engineering of AES Wireless Keyboard

■ Firmware Acquisition – nRF24LE1

- Flash memory Dump

```
$ ./nrfprog /dev/ttyUSB0 dump.bin
Opening the Bus Pirate UART
Setting the Bus Pirate to Binary Mode
Configuring SPI mode
Backing up the info page
Starting read operation
Reading from the device to dump.bin
Starting read operation
Putting the Bus Pirate back in normal operating mode
Closing the Bus Pirate
$ ls -alh dump.bin info_page.dat
-rwxrwxrwx 1 root root 16K April 28 19:32 dump.bin
-rwxrwxrwx 1 root root 256 April 28 19:32 info_page.dat
```



Reverse Engineering of AES Wireless Keyboard

■ Static Analysis – Disassembling

010 Editor - E:\Presentations\2018\BlackHatAsia\acquistion\dump.bin

File Edit Search View Format Scripts Templates Tools Window Help

dump.bin

Edit As: Hex Run Script Run Template

0 1 2 3 4 5 6 7 8 9 A B C D E F 0123456789ABCDEF

0000h: 02 28 56 02 00 11 C2 B9 D2 1A 32 02 00 12 D2 1F .(V...Ã°0.2...0.
0010h: 32 32 32 02 00 19 AF D7 22 32 32 02 2F 66 32 32 222...x"22./f22
0020h: 32 32 FF 02 00 1A FF FF FF FF FF 02 00 1E 7C 00 22ÿ...ÿÿÿÿ...|
0030h: 7D F9 90 00 F9 E4 F5 3C E0 62 3C A3 DF FA E5 3C }ù..ùãð<àb<£\$úã<
0040h: F4 F0 22 02 00 1F FF FF FF FF FF 02 00 06 FF FF ôð"...ÿÿÿÿ...ÿÿ
0050h: FF FF FF 02 00 20 FF FF FF FF FF 02 00 21 FF FF ÿÿÿ...ÿÿÿÿ...ÿÿ
0060h: FF FF FF 02 2F FF FF FF FF FF FF 02 00 0E 7F 00 ÿÿÿ./ÿÿÿÿÿÿ....
0070h: 90 01 84 74 01 F0 A3 74 00 F0 75 34 08 90 01 84 ...t.ð£t.ðu4....
0080h: E0 FE A3 E0 F5 82 8E 83 E4 F0 90 01 84 75 F0 01 àp£ãð,žfãð...uð.
0090h: 12 23 59 D5 34 E7 7E 01 90 01 84 74 01 F0 A3 74 .#YÔ4ç~...t.ð£t
00A0h: 01 F0 75 32 01 75 33 73 75 39 06 E4 F5 3A F5 37 .ðu2.u3su9.ãð:ð7
00B0h: 75 34 10 05 33 E5 33 70 02 05 32 F5 82 85 32 83 u4..3ã3p..2ð,...2f
00C0h: E0 FF 70 03 02 03 43 75 17 06 8F 22 20 12 06 20 àÿp...Cu..."..
00D0h: 13 03 02 03 37 E5 33 24 F0 F5 31 E5 32 34 FF F57ã3\$ðð1ã24ÿð
00E0h: 30 85 31 82 F5 83 E0 90 37 6A 93 F5 36 64 F0 70 0...1,ðfà.7j"ðððð
00F0h: 23 7F B0 7E FF 12 35 3B 8F 3B E5 3B B4 FF 15 20 #.°~ÿ.5;.:ã;ÿ.
0100h: 0C 12 85 31 82 85 30 83 E4 F0 85 33 82 85 32 83 ...1,...0fãð...3,...2f
0110h: F0 02 27 B5 E5 36 64 03 70 57 85 31 82 85 30 83 ð.'uãðd.pW...1,...0f
0120h: E0 F5 38 75 35 0F 74 64 25 35 F5 82 E4 34 01 F5 àððu5.tð\$ð,ã4.ð
0130h: 83 E4 F0 74 74 25 35 F5 82 E4 34 01 F5 83 E4 F0 fãðtt\$ðð,ã4.ðfãð
0140h: D5 35 E3 20 13 0C 90 01 64 E5 38 F0 90 01 74 74 Ô5ãdãðð...tt
0150h: 08 F0 90 01 02 74 4C F0 E4 A3 F0 A3 F0 A3 F0 A3 .ð...tLðãðððððððð
0160h: F0 A3 F0 90 01 00 74 05 F0 75 3A 03 F5 37 02 03 ð£ð...t.ðu:..ð7..
0170h: 43 E5 36 C3 94 04 50 12 30 13 03 02 03 37 85 33 Cã6ã".P.O...7...3
0180h: 82 85 32 83 74 08 F0 02 03 37 E5 36 C3 94 3A 50 ,...2ft.ð...7ã6ã":P
0190h: 32 E5 39 70 03 02 03 37 90 01 84 E4 75 F0 01 12 2ã9p...7...ãuð..
01A0h: 23 59 15 39 90 01 84 E0 FE A3 E0 F5 82 8E 83 E5 #Y.9...àp£ãð,žfã
01B0h: 36 F0 20 13 09 85 33 82 85 32 83 74 08 F0 05 3A 6ð3,...2ft.ð.:
01C0h: 02 03 37 E5 36 D3 94 45 50 72 30 13 2F 20 16 06 ..7ã6ó"EPr0./..
01D0h: 20 25 03 02 03 37 E5 39 70 03 02 03 37 90 01 84 %...7ã9p...7...
01E0h: E4 75 F0 01 12 23 59 15 39 90 01 84 E0 FE A3 E0 ãuð...#Y.9...àp£ã

Pos: 16384 [4000h] Size: 16384 ANSI LIT W OVR

IDA - E:\Presentations\2018\BlackHatAsia\disasm\dump.bin

File Edit Jump Search View Options Windows Help

Library function Data Regular function Unexplored Instruction External symbol

Functions window

Function name

- RESET
- IE0
- TF0
- IE1
- nullsub_1
- TF1
- nullsub_2
- RJTI
- TF2_EXF2
- code_2E
- code_C51
- code_F62
- code_1271
- code_1490
- code_169F
- code_182F
- code_19A8
- code_1B01
- code_1C3C
- code_1D77
- code_1EA0
- code_1FBC
- code_20D6
- code_21E7
- code_21EF
- code_22DE

code:00000000 ;
code:00000000 ; +-----+
code:00000000 ; | This file has been generated by The Interactive Disassembler
code:00000000 ; | Copyright (c) 2015 Hex-Rays, <support@hex-rays.com>
code:00000000 ; | License info: <http://www.hex-rays.com>
code:00000000 ; | Kwon-young Kim, SNT Works Inc.
code:00000000 ; +-----+
code:00000000 ;
code:00000000 ; Input SHA256 : CC5519D4FCC41729BCE751A3A24F3E88C3B55F19B514688FD0
code:00000000 ; Input MD5 : 660E822C449F859AFCBC9567D36D489D
code:00000000 ; Input CRC32 : D6C11742
code:00000000 ;
code:00000000 ; File Name : E:\Presentations\2018\BlackHatAsia\disasm\dump.bin
code:00000000 ; Format : Binary file
code:00000000 ; Base Address: 0000h Range: 0000h - 4000h Loaded length: 4000h
code:00000000 ;
code:00000000 ; Processor : 8032 [RAM=256 ROM=0 EPROM=0 EEPROM=0]
code:00000000 ; Target assembler: ASMI
code:00000000 ; Byte sex : Big endian
code:00000000 ;
code:00000000 ;
code:00000000 ; Segment type: Pure code
code:00000000 ; .segment code
code:00000000 ; [00000003 BYTES: COLLAPSED FUNCTION RESET. PRESS CTRL-NUMPAD+ TO I
code:00000003 ; [00000003 BYTES: COLLAPSED FUNCTION IE0. PRESS CTRL-NUMPAD+ TO EX
code:00000006 ;
00000000 00000000: RESET (Synchronized with Hex View-1)

Output window

The initial autoanalysis has been finished.

Python

AU: idle Down Disk: 602GB

Reverse Engineering of AES Wireless Keyboard

■ Static Analysis – Finding Encryption functions

hal_aes_crypt:

```
mov    DPTR, #0x194
lcall  save_r1_r2_r3
clr    A
mov    DPTR, #0x197
movx   @DPTR, A
```

code_28FB:

```
mov    DPTR, #0x194
lcall  load_r1_r2_r3
mov    DPTR, #0x197
movx   A, @DPTR
mov    R7, A
mov    DPL, A
mov    DPH, #0
lcall  code_22F8
mov    R6, A
mov    A, #0xB9 ; '
add    A, R7
mov    DPL, A
clr    A
addc   A, #0
mov    DPH, A
mov    A, R6
movx   @DPTR, A
mov    DPTR, #0x197
movx   A, @DPTR
inc    A
movx   @DPTR, A
movx   A, @DPTR
cjne   A, #0x10, code_28FB
clr    A
mov    R7, A
lcall  key_upgrade
mov    CCPDATIB, #2
mov    DPTR, #0x197
mov    A, #1
movx   @DPTR, A
```

```
void hal_aes_crypt(uint8_t * dest_buf, const uint8_t * src_buf)
{
#ifdef __CS1__
    uint8_t data aes_state[16];    //AES State
#endif

    uint8_t a;
    for(a=0;a<16;a++)
    {
        aes_state[a] = src_buf[a];
    }

    //Start
    CCPDATIB = 0x02; //Set co-processor the GF(2^8)*2 (used in mix-columns)
    for (a=0;a<9;a++)
    {
        add_sub_shift(aes_state);
        mix_columns_hw(aes_state);
        key_upgrade(a);
    }

    //FINAL round
    add_sub_shift(aes_state);
    key_upgrade(9);
    add_key(aes_state);

    //Clean up
    for(a=0;a<16;a++)
    {
        aes_round_key[a]=cipher_key[a]; //Write back cipher-key
        dest_buf[a] = aes_state[a];    //Write out encrypted result
    }
}
```

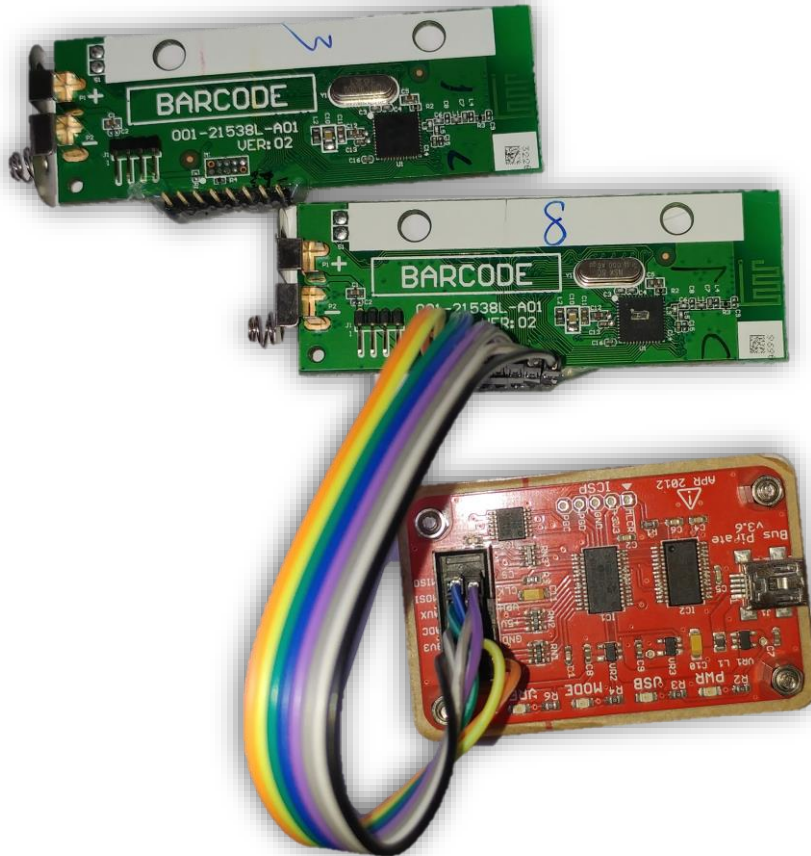
code_2933:

```
lcall  add_sub_shift
lcall  mix_columns_hw
mov    DPTR, #0x197
movx   A, @DPTR
mov    R7, A
lcall  key_upgrade
mov    DPTR, #0x197
movx   A, @DPTR
inc    A
movx   @DPTR, A
movx   A, @DPTR
clr    C
subb   A, #0xA
jc     code_2933
lcall  add_sub_shift
mov    R7, #0xA
lcall  key_upgrade
lcall  add_key
clr    A
mov    DPTR, #0x197
movx   @DPTR, A
```

nRFgo SDK source : hal_aes.c

Reverse Engineering of AES Wireless Keyboard

- Static Analysis – Finding AES key
 - Flash Dump from another keyboard



dumpA.bin

0000h

0010h

0020h

0030h

0040h

0050h

0060h

0070h

0080h

0090h

00A0h

00B0h

00C0h

00D0h

00E0h

00F0h

0100h

0110h

0120h

0130h

0140h

0150h

0160h

0170h

0180h

0190h

01A0h

01B0h

01C0h

01D0h

01E0h

01F0h

0200h

0210h

0220h

0230h

0240h

0250h

322856020011C2B9021A32020012021F

323232020019AF07223232022F663232

3232FF02001AFFFFFF02001E7C00

70F99000F9E4F53CE0623CA3DFFAE53C

F4F02202001FFFFFFF020006FFFF

FFFFFF020020FFFFFF020021FFFF

FFFFFF022FFFFFFF02000E7F00

9001847401F0A37400F0753408900184

E0FEA3E0F5828E83E4F090018475F001

1223590534E77E019001847401F0A374

01F0753201753373753906E4F53AF537

7534100533E53370020532F582853283

E0FF70030203437517068F2220120620

1303020337E53324F0F531E53234FFF5

30853182F583E090376A93F53664F070

237F807EFF1235388F38E538B4FF1520

0C12853182853083E4F0853382853283

F0022785E53664037057853182853083

E0F53875350F74642535F582E43401F5

83E4F074742535F582E43401F583E4F0

0535E320130C900164E538F090017474

08F0900102744CF0E4A3F0A3F0A3F0A3

F0A3F09001007405F0753A03F5370203

43E536C3940450123013030203378533

828532837408F0020337E536C3943A50

32E5397003020337900184E475F00112

23591539900184E0FEA3E0F5828E83E5

36F02013098533828532837408F0053A

020337E536039445507230132F201606

202503020337E5397003020337900184

E475F0011223591539900184E0FEA3E0

F5828E83E536F0053A02033720120302

0337200003302529E539601C900184E4

75F0011223591539900184E0FEA3E0F5

828E83E536F0053A8533828532837448

F0200F03020337020A020337E5360394

925032E5397003020337900184E475F0

011223591539900184E0FEA3E0F5828E

(V.....2.....

222.....22../122

22.....22.....

}.....<b<.....<

...../.....

...t...t..u4....

.....t.....u..

#V.4.....t...t..

..u2..u3su9.....7

u4...3..3p...2...2..

...p...Cu... ..

.....7.3\$...1.24...

0.1.....7j...6d.p

#.....5.....

...1...0...3..2..

.....6d.plw1..0..

..8u5..tdx5...4....

...ttx5...4.....

.5.....d.8...tt

.....tL.....

.....t...u...7...

C.6...P.0...7.3

..2.t...7.6...P

2.9p...7...u...

#V.9.....

6...3...2.t... ..

..7.6...EPr0../ ..

%...7.9p...7... ..

..u...#V.9.....

.....6...7... ..

.7...0x).9.....

u...#V.9.....

...6...3..2.tH

.....7...7.6... ..

.P2.9p...7...u... ..

#V.9.....

dumpB.bin

0000h

0010h

0020h

0030h

0040h

0050h

0060h

0070h

0080h

0090h

00A0h

00B0h

00C0h

00D0h

00E0h

00F0h

0100h

0110h

0120h

0130h

0140h

0150h

0160h

0170h

0180h

0190h

01A0h

01B0h

01C0h

01D0h

01E0h

01F0h

0200h

0210h

0220h

0230h

0240h

0250h

322856020011C2B9021A32020012021F

323232020019AF07223232022F663232

3232FF02001AFFFFFF02001E7C00

70F99000F9E4F53CE0623CA3DFFAE53C

F4F02202001FFFFFFF020006FFFF

FFFFFF020020FFFFFF020021FFFF

FFFFFF022FFFFFFF02000E7F00

9001847401F0A37400F0753408900184

E0FEA3E0F5828E83E4F090018475F001

1223590534E77E019001847401F0A374

01F0753201753373753906E4F53AF537

7534100533E53370020532F582853283

E0FF70030203437517068F2220120620

1303020337E53324F0F531E53234FFF5

30853182F583E090376A93F53664F070

237F807EFF1235388F38E538B4FF1520

0C12853182853083E4F0853382853283

F0022785E53664037057853182853083

E0F53875350F74642535F582E43401F5

83E4F074742535F582E43401F583E4F0

0535E320130C900164E538F090017474

08F0900102744CF0E4A3F0A3F0A3F0A3

F0A3F09001007405F0753A03F5370203

43E536C3940450123013030203378533

828532837408F0020337E536C3943A50

32E5397003020337900184E475F00112

23591539900184E0FEA3E0F5828E83E5

36F02013098533828532837408F0053A

020337E536039445507230132F201606

202503020337E5397003020337900184

E475F0011223591539900184E0FEA3E0

F5828E83E536F0053A02033720120302

0337200003302529E539601C900184E4

75F0011223591539900184E0FEA3E0F5

828E83E536F0053A8533828532837448

F0200F03020337020A020337E5360394

925032E5397003020337900184E475F0

011223591539900184E0FEA3E0F5828E

(V.....2.....

222.....22../122

22.....22.....

}.....<b<.....<

...../.....

...t...t..u4....

.....t.....u..

#V.4.....t...t..

..u2..u3su9.....7

u4...3..3p...2...2..

...p...Cu... ..

.....7.3\$...1.24...

0.1.....7j...6d.p

#.....5.....

...1...0...3..2..

.....6d.plw1..0..

..8u5..tdx5...4....

...ttx5...4.....

.5.....d.8...tt

.....tL.....

.....t...u...7...

C.6...P.0...7.3

..2.t...7.6...P

2.9p...7...u...

#V.9.....

6...3...2.t... ..

..7.6...EPr0../ ..

%...7.9p...7... ..

..u...#V.9.....

.....6...7... ..

.7...0x).9.....

u...#V.9.....

...6...3..2.tH

.....7...7.6... ..

.P2.9p...7...u... ..

#V.9.....

Output

Q:\W2017_MS_WKeyboard\Wnrprog-master\dumpA.bin

vs

Q:\W2017_MS_WKeyboard\Wnrprog-master\dumpB.bin

Result

Address A

Size A

Address B

Size B

☒ Match

0h

4000h

0h

4000h

Reverse Engineering of AES Wireless Keyboard

- Static Analysis – Finding AES key
 - NVRAM are not the same values
 - Whether or not the AES key is validated using debugging

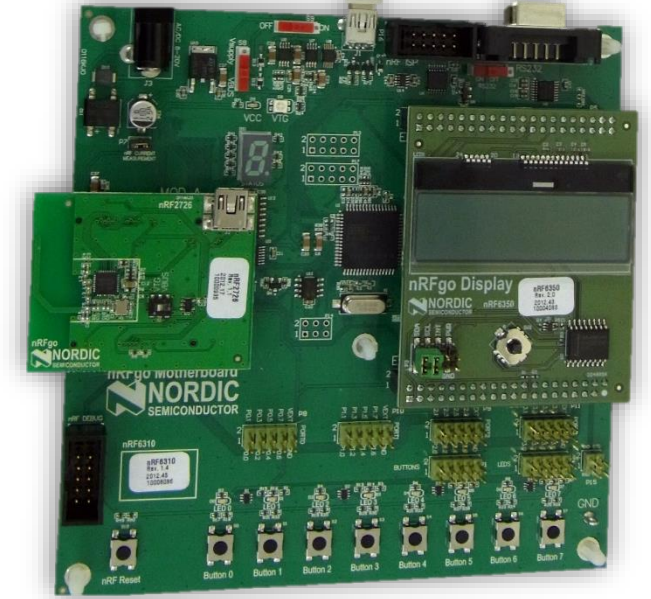
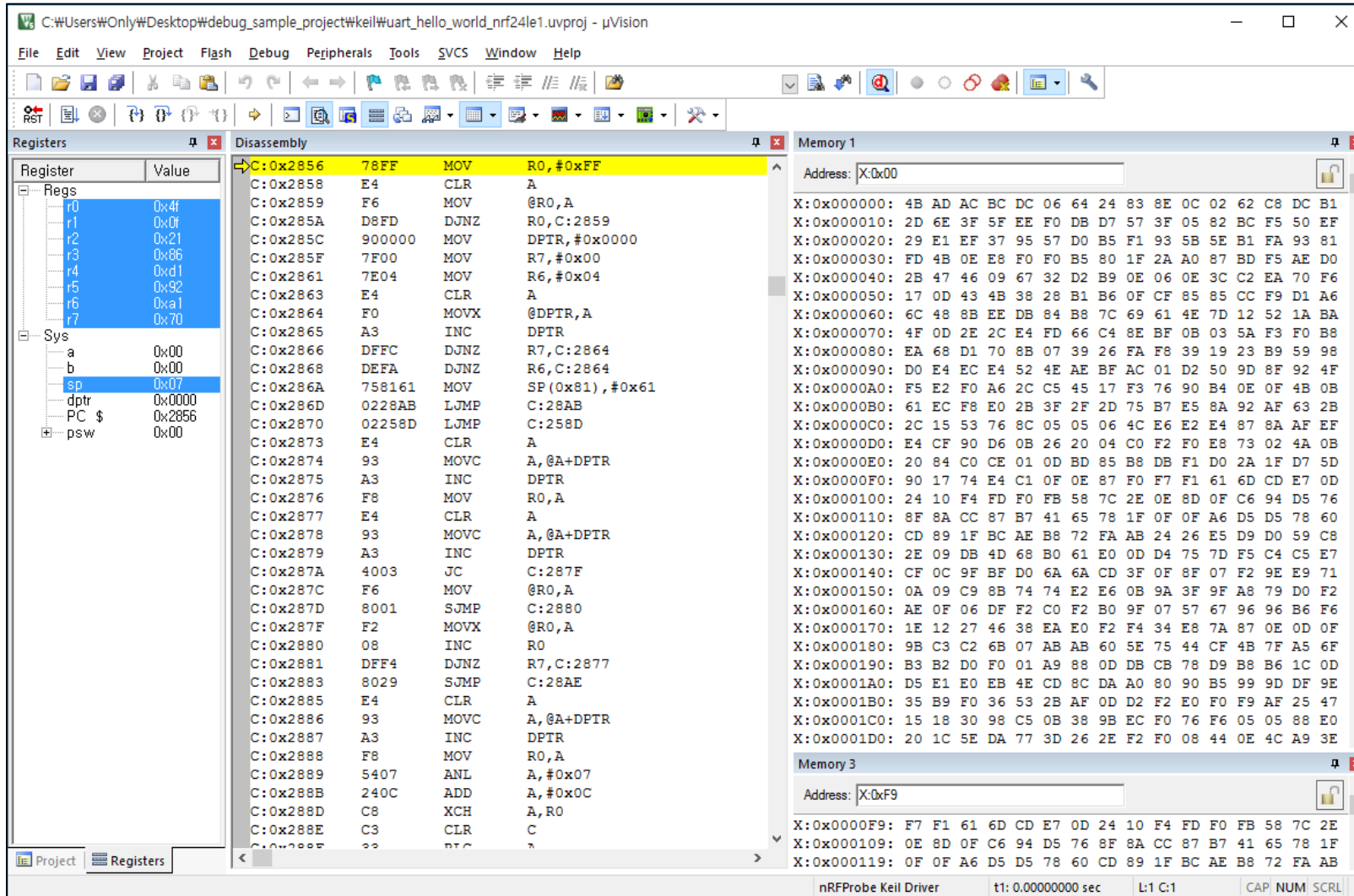
Address	Hex	ASCII
3F50h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3F60h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3F70h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3F80h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3F90h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3FA0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3FB0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3FC0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3FD0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3FE0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3FF0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
4000h	03 DF 61 F0 A9 92 C2 98 38 AC 5E 0F 90 26 E4 2F	..a.....8.^..&./
4010h	13 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4020h	02 00 C0 21 01 00 00 00 00 00 00 00 00 E4 00	...!.....
4030h	01 CD 30 D5 02 AA 1A 03 13 4E 44 FF 00 03 43 04	..0.....ND...C.
4040h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
4050h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
4060h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
4070h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
4080h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
4090h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
40A0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
40B0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
40C0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
40D0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
40E0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
40F0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

Address	Hex	ASCII
3F50h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3F60h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3F70h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3F80h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3F90h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3FA0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3FB0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3FC0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3FD0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3FE0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
3FF0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
4000h	03 7E 1E 19 4D 04 BC 99 B5 58 2B 31 78 66 BE A4	..M....[+1xf ..
4010h	06 00 00 00 00 00 00 00 00 00 00 00 00 E0 06
4020h	02 00 C0 21 01 00 00 00 00 00 00 00 00 E4 00	...!.....
4030h	01 CD C1 D3 02 AA 1A 1D 32 46 50 FF 00 03 0F 052FP.....
4040h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
4050h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
4060h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
4070h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
4080h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
4090h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
40A0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
40B0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
40C0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
40D0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
40E0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
40F0h	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

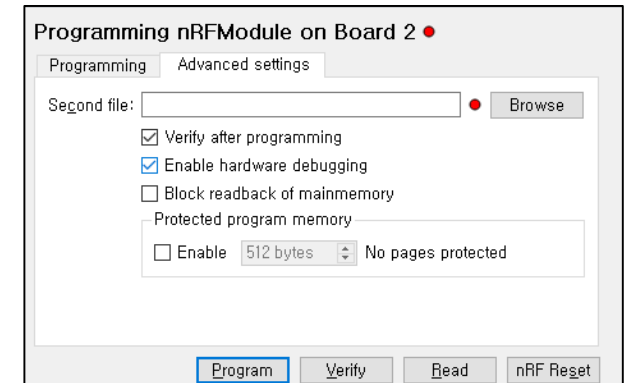
Pos: 17178 [431Ah] Val: 255 FFh 11111111b Size: 18432 ASCII LIT W OVR

Reverse Engineering of AES Wireless Keyboard

- Dynamic Analysis – Debugging codes

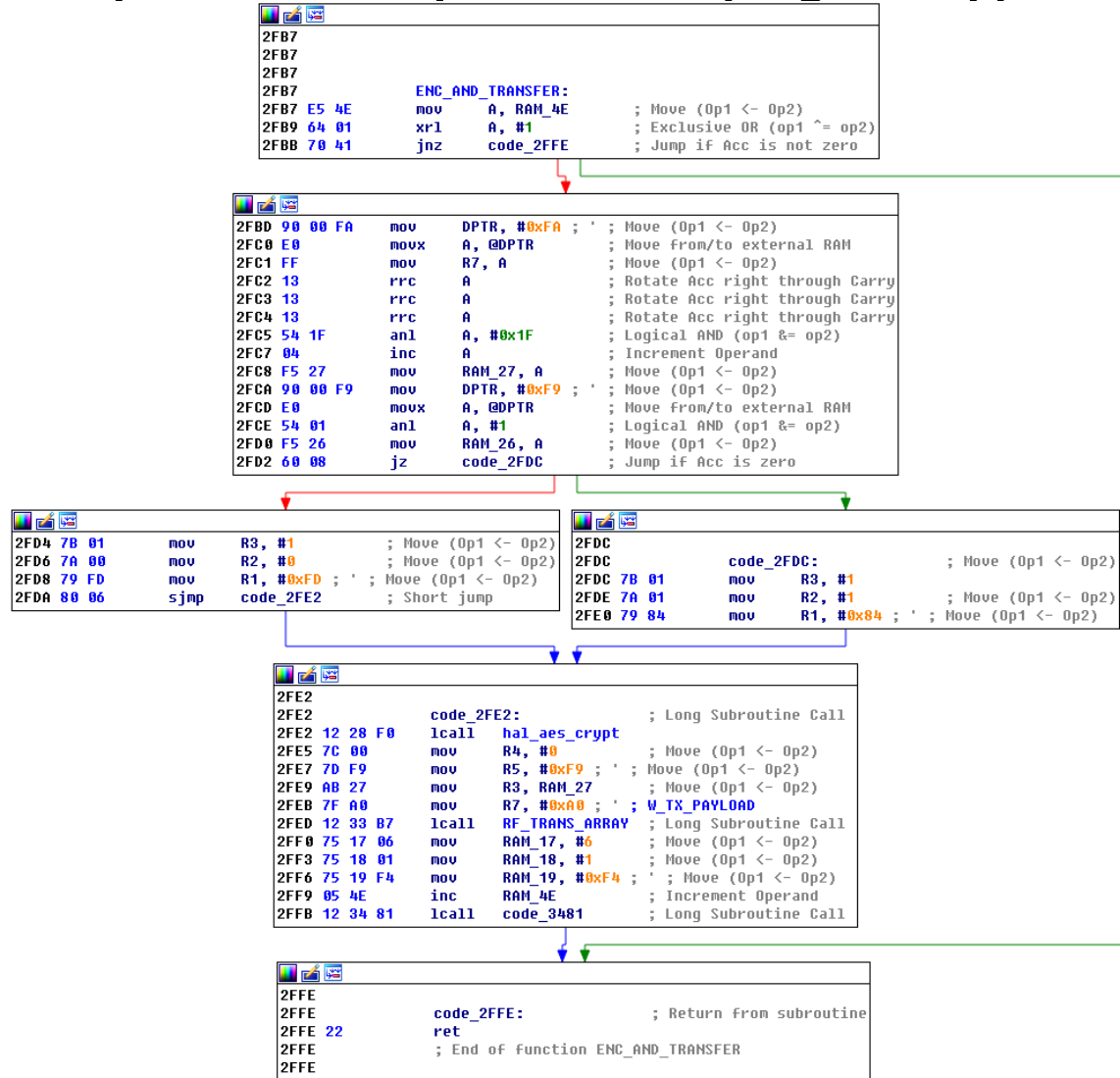


nRFGo Start Kit & nRF24LE1

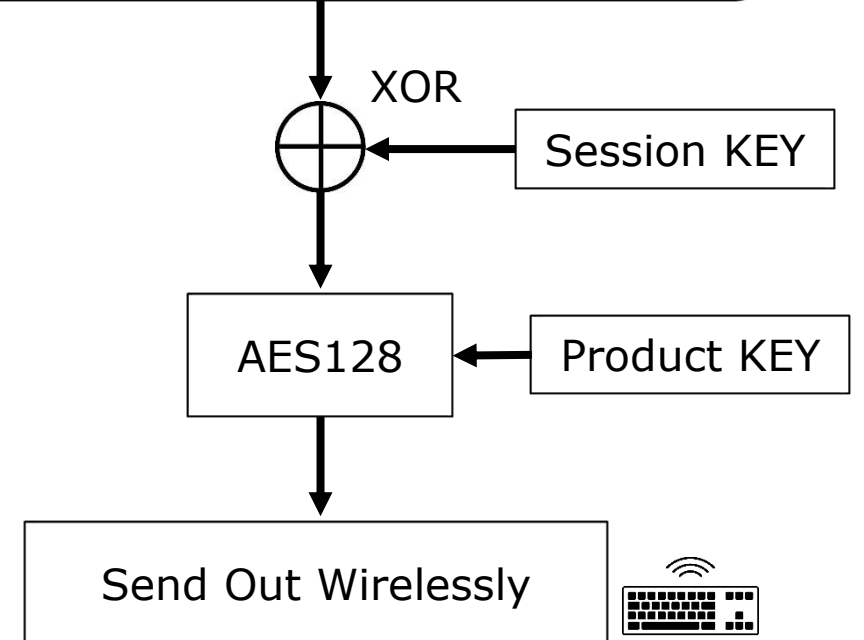


Reverse Engineering of AES Wireless Keyboard

Dynamic Analysis – Verifying Encryption functions



Plain Packet



Reverse Engineering of AES Wireless Keyboard

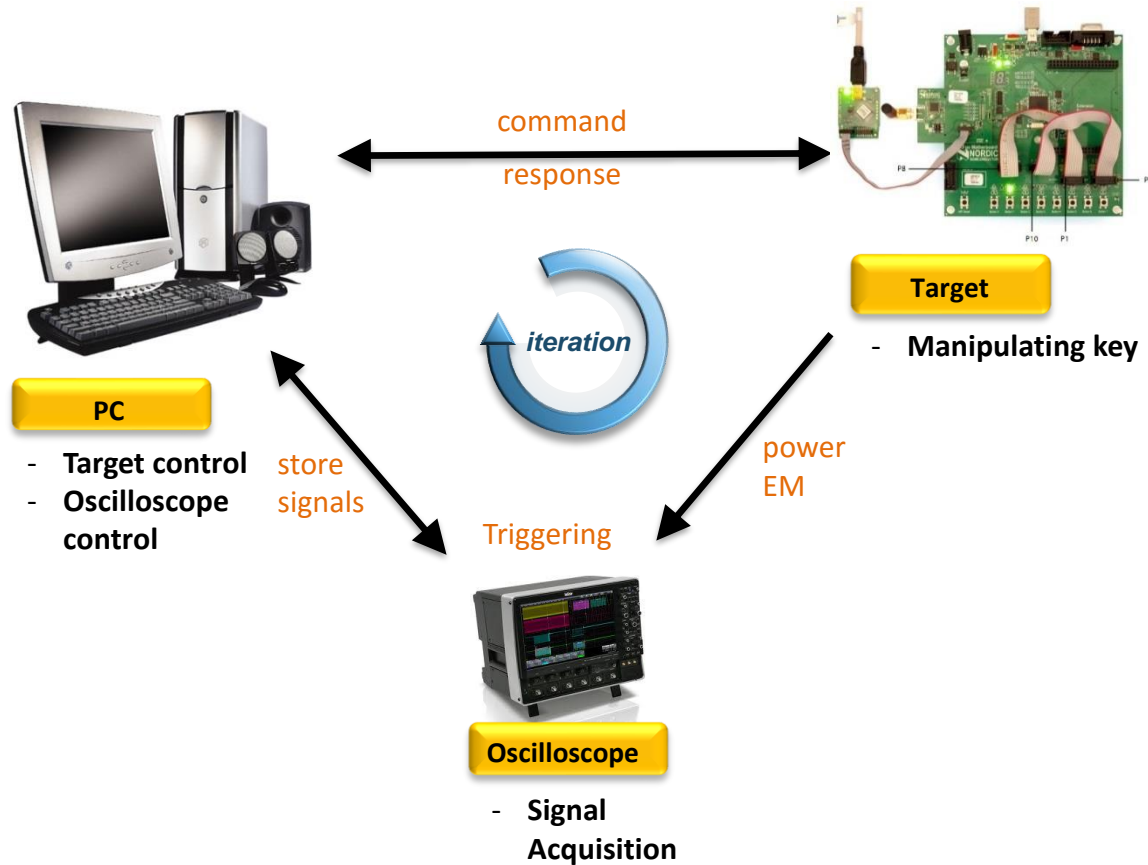
- **Firmware Modification – Modifying codes for Repeated execution**
 - Initial input of all zero bytes
 - Output → Next Encryption input
 - Inserting artificial trigger signal before each encryption
 - 100 ms delay
- **Firmware Modification – Reprogramming of Keyboard firmware**
 - Write protection is set for the target keyboard
 - So, We changed it to a reprogrammed chip for Side channel analysis

AES Wireless Keyboard

- Side Channel Attacks, Template Attacks -

Side Channel Attacks of AES Wireless Keyboard

■ Side Channel Attacks in Theory

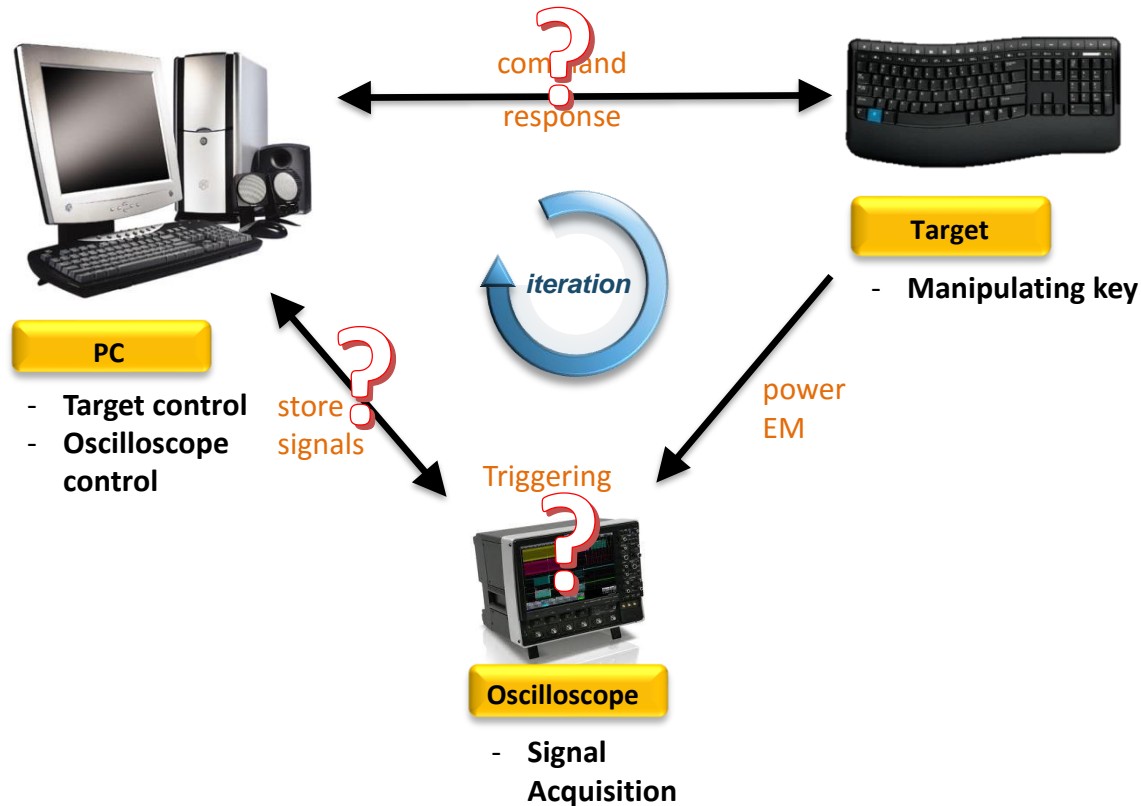


White-box Attack Model

- All information known and controllable
 - Known Specification
 - Known Interfaces
 - Known Design (by oneself)
 - Known Implementation details
 - Ideal Setup

Side Channel Attacks of AES Wireless Keyboard

■ Side Channel Attacks in Real World



Black-box Attack Model

- Most information unknown and controllable
 - Known Specification Only
 - **Unknown** Interfaces
 - **Unknown** SW/HW Design
 - **Unknown** Implementation details
 - Many **unknown** factors for setup

Side Channel Attacks of AES Wireless Keyboard

■ Side Channel Attack Environment

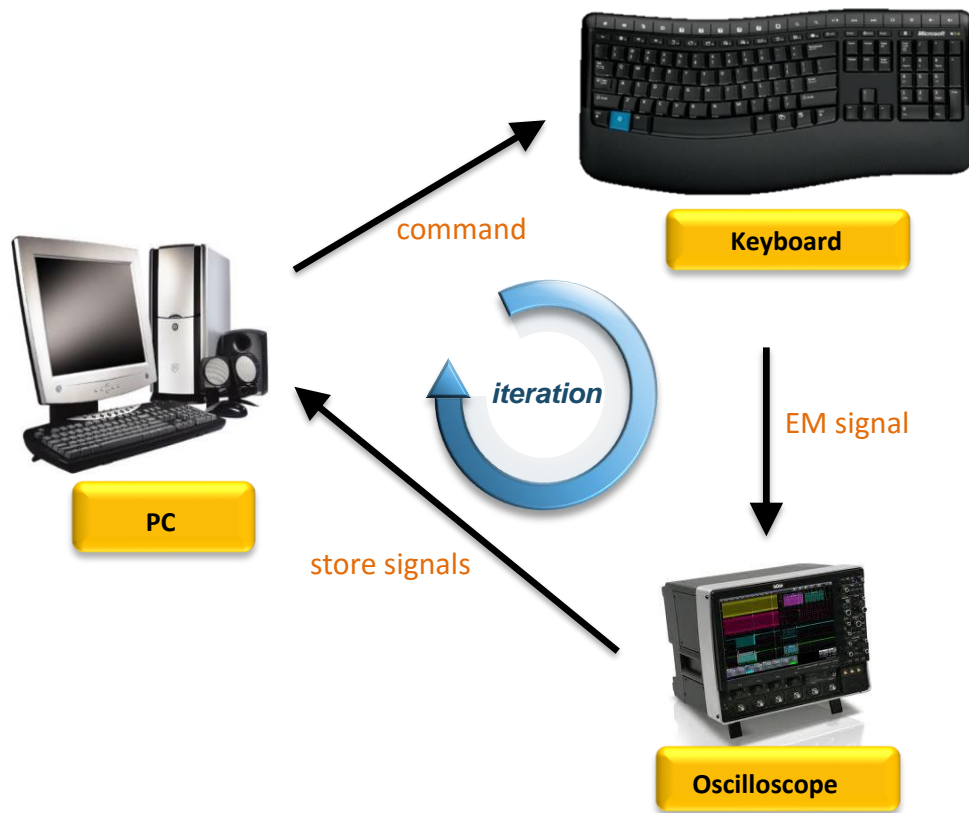


Setup

- ① Target Keyboard with AES key
- ② Langer LF-1 near-field probe set
- ③ Langer PA303 pre-amplifier
- ④ Digital Oscilloscope
- ⑤ Software to control a target and an oscilloscope

Side Channel Attacks of AES Wireless Keyboard

- Side Channel Attack



Attack Process

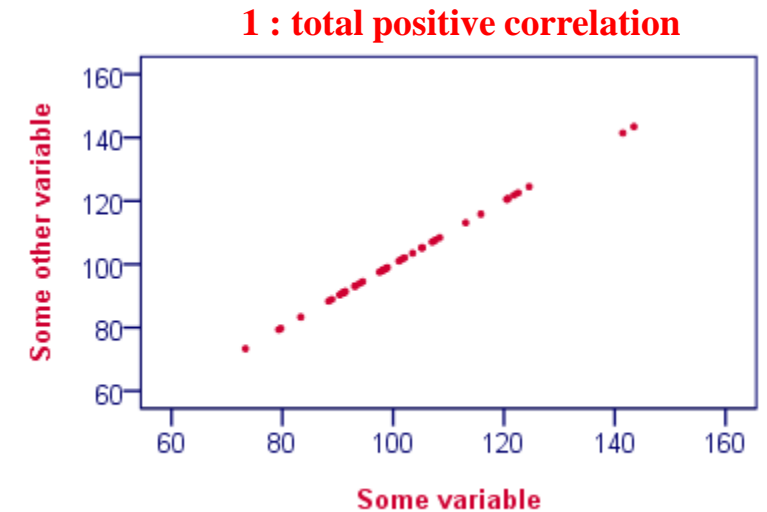
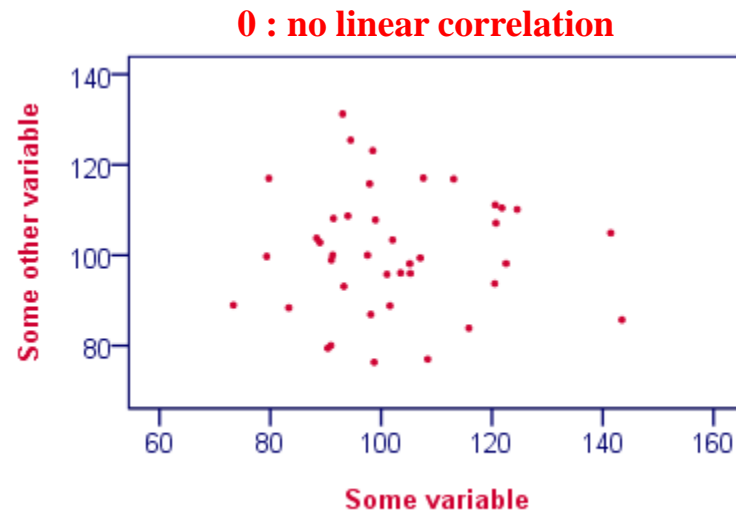
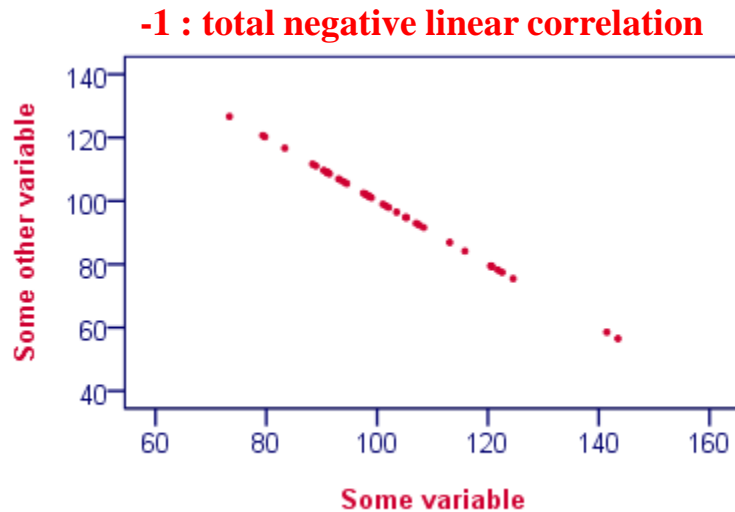
- Acquisition
- Alignment
- Correlation with input/output
- Profiling AES algorithm
- CPA
- Building templates
- Template attack

Side Channel Attacks of AES Wireless Keyboard

■ CPA (Correlation Power Analysis)

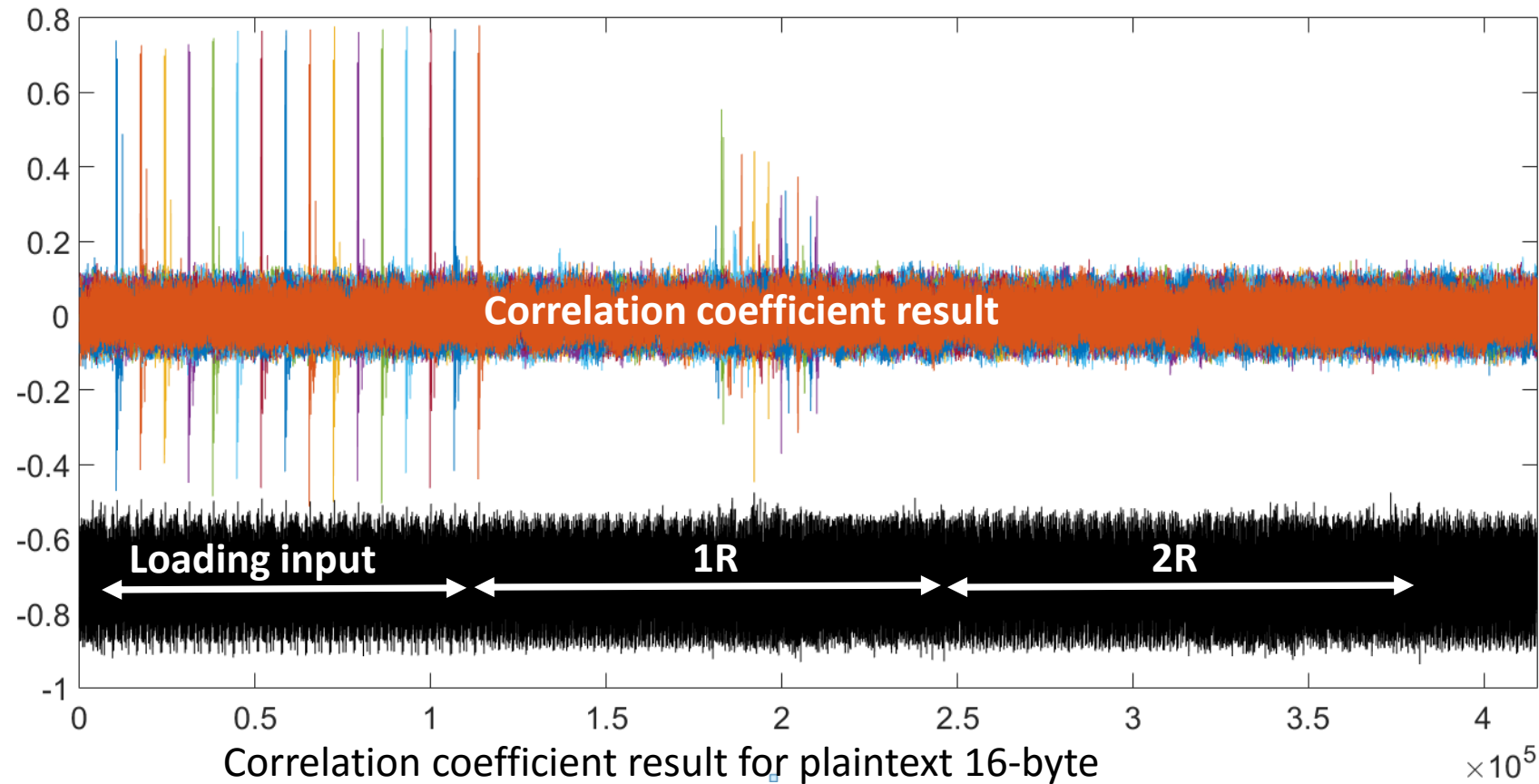
- Pearson Correlation Coefficient
- T : measurement, P : predictions
- Number between -1 and 1

$$\rho(T, P) = \frac{E(TP) - E(T)E(P)}{\sqrt{\text{Var}(T)\text{Var}(P)}}$$



Side Channel Attacks of AES Wireless Keyboard

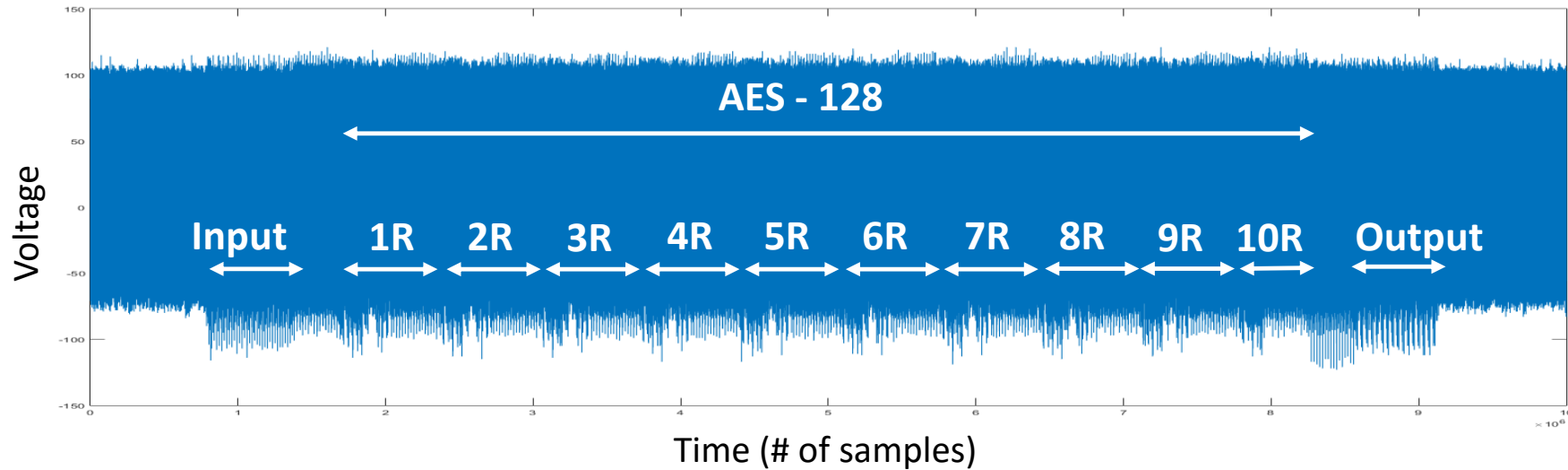
- CPA (Correlation Power Analysis) with input
 - Confirm plaintext loading operation



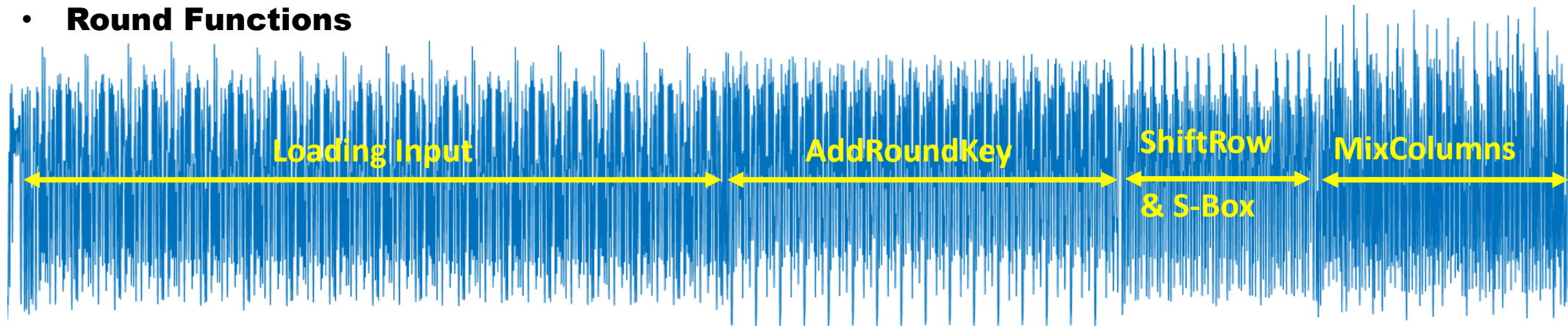
- HW model
- 1K traces
- 1M samples

Side Channel Attacks of AES Wireless Keyboard

- Profiling EM Trace



- **Round Functions**

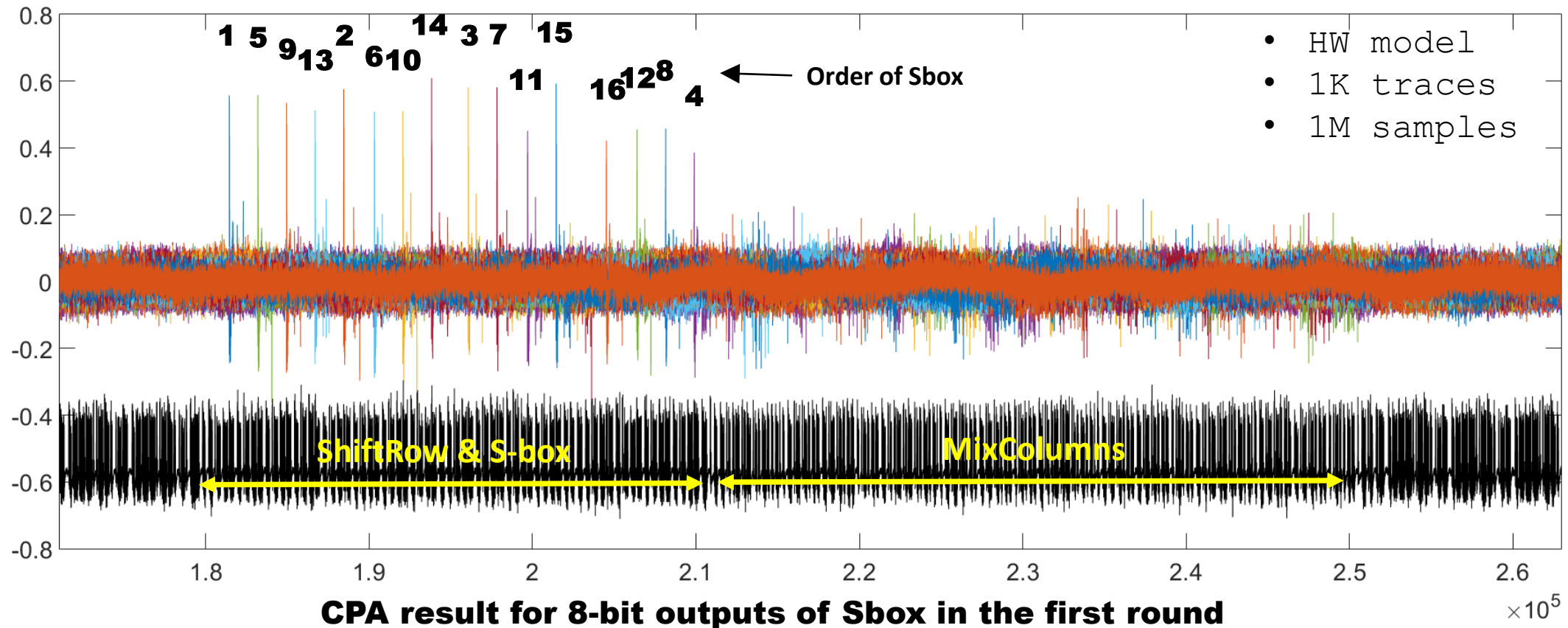


※ **Hardware Co-processor of MixColumns operation**

Side Channel Attacks of AES Wireless Keyboard

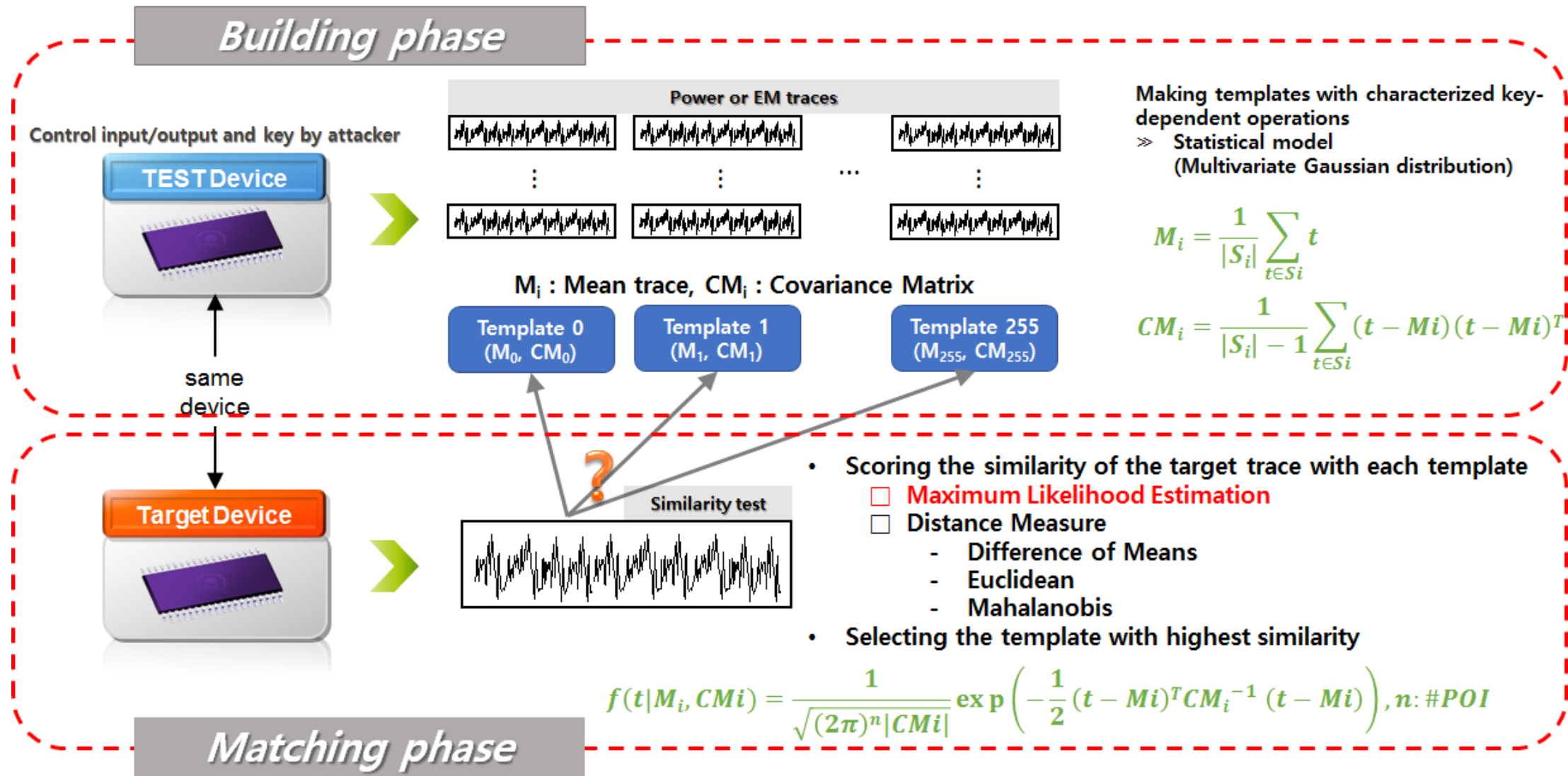
■ CPA (Correlation Power Analysis)

- Recover the secret key
- Target variable : S-Box output



Side Channel Attacks of AES Wireless Keyboard

Basic Template Attacks



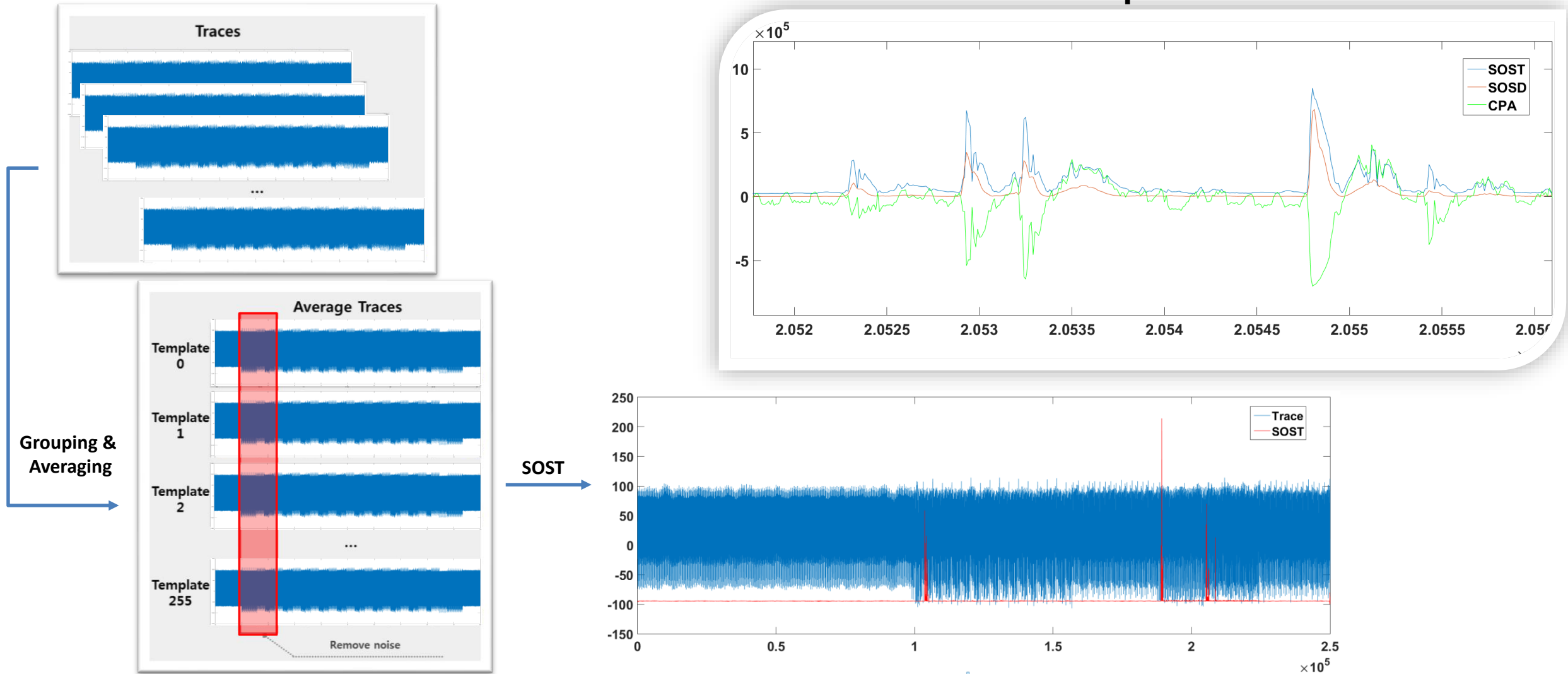
Side Channel Attacks of AES Wireless Keyboard

■ Selecting Interesting Points

- Computational or memory restrictions
- Reducing the number of samples and the size of templates
- Selection of some special samples that contain the most information about the characterized key-dependent operation
 - ✓ DOM (Sum of pairwise Differences)
 - ✓ SOSD (Sum of Squared pairwise Differences)
 - ✓ SOST (Sum of Squared pairwise T-differences)
 - ✓ NICV (Normalized Inter-Class Variance)
 - ✓ PCA (Principal Component Analysis)
 - ✓ LDA (Linear Discriminant Analysis)
 - ✓ ...

Side Channel Attacks of AES Wireless Keyboard

- Selecting Interesting Points – SOST



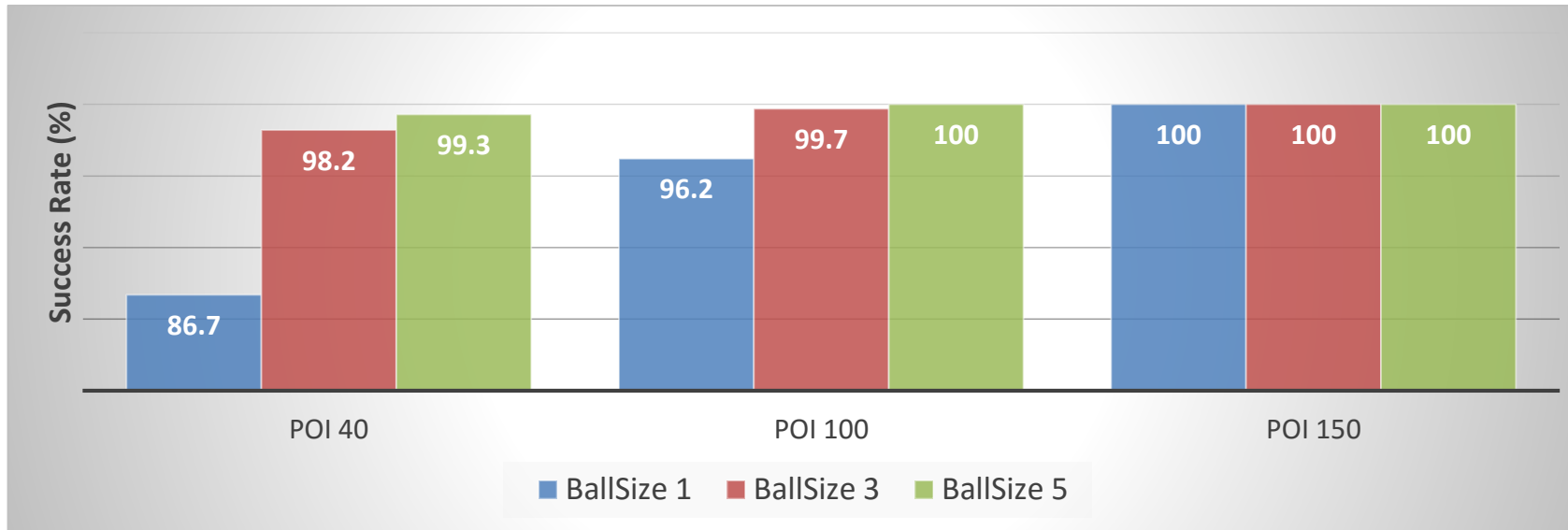
Side Channel Attacks of AES Wireless Keyboard

■ Building Templates

- 256 templates for a bytes AES key
- Training set : 1K traces for each template (total: 256K)
- Test set : 1K traces for each value (total: 256K)
- Point of Interest : 150 points
- Method : SOST, PCA, Pooled Covariance Matrix
 - ✓ Case 1: same device (profiling = matching device)
 - ✓ Case 2: different device (profiling \neq matching device)

Side Channel Attacks of AES Wireless Keyboard

- Template Attacks : Case 1 – Same device
 - Selecting POI : SOST
 - Template Matching : Maximum Likelihood

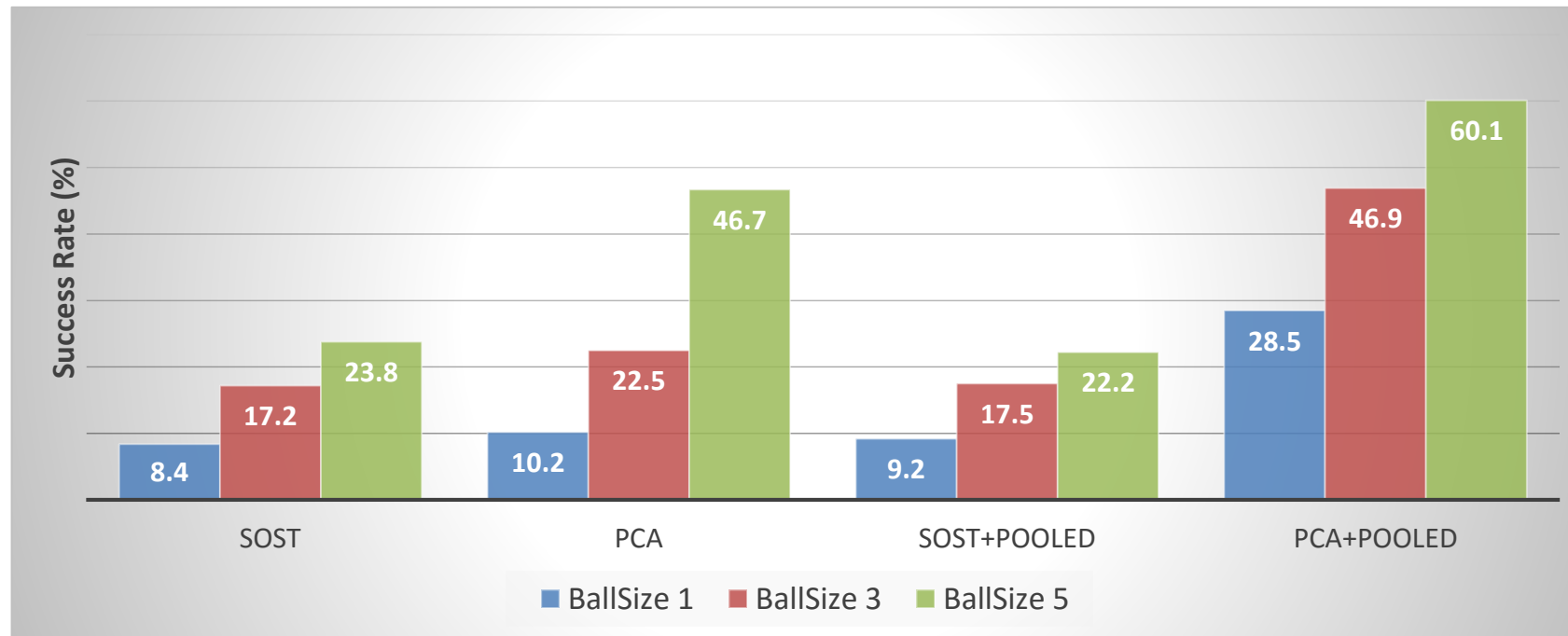


- **Success Rate** is the probability that the correct key is ranked among the first n candidates.
- **BallSize** n means the top n candidates ranked according to score, highest to lowest.

Side Channel Attacks of AES Wireless Keyboard

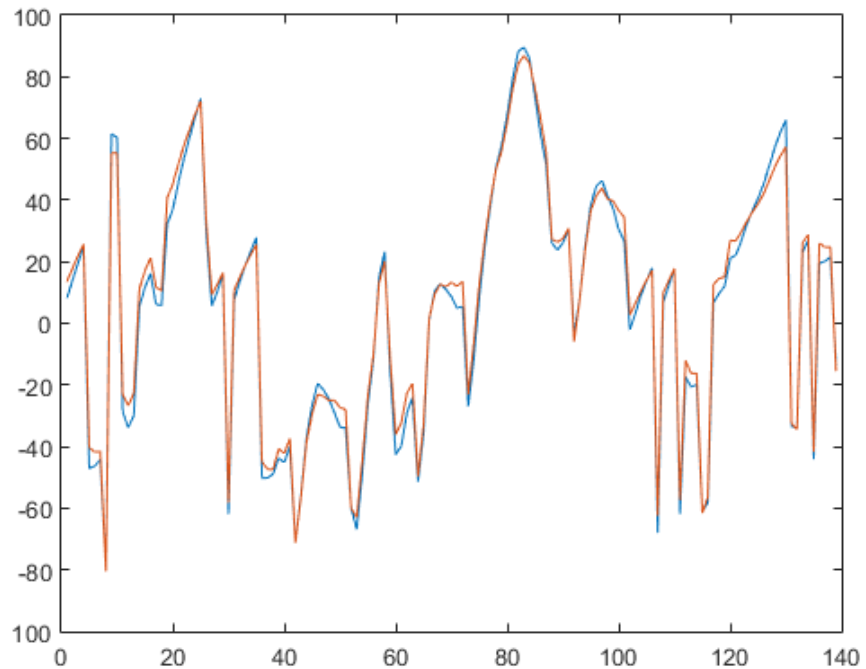
■ Template Attacks : Case 2 – Difference device

- Selecting POI : SOST, PCA, Pooled Covariance Matrix
 - ✓ Common POI of Device 1 (training) and POI of Device 2 (target)
 - ✓ 139 Points in 150 points
- Template Matching : Maximum Likelihood

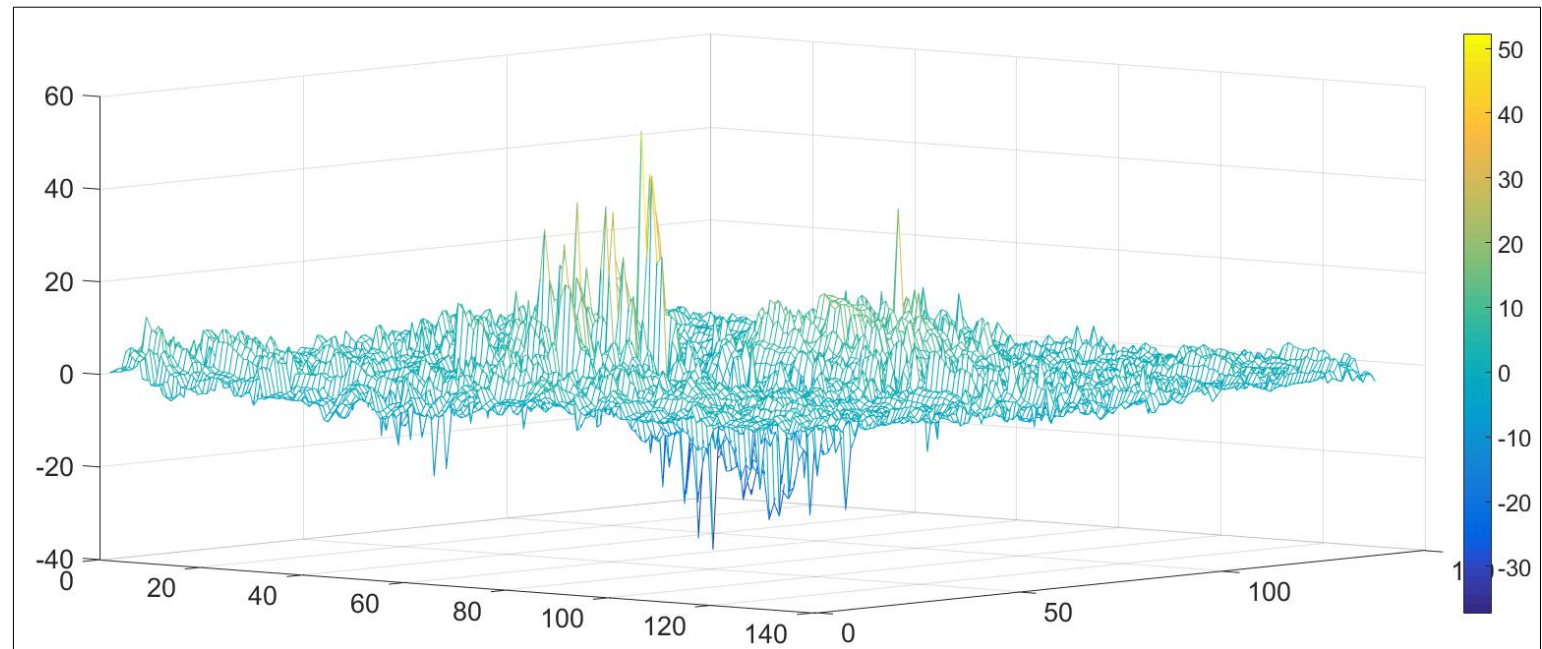


Side Channel Attacks of AES Wireless Keyboard

- **Template Attacks : Case 2 – Difference device**
 - Characteristics comparison between two devices



Comparison of mean traces

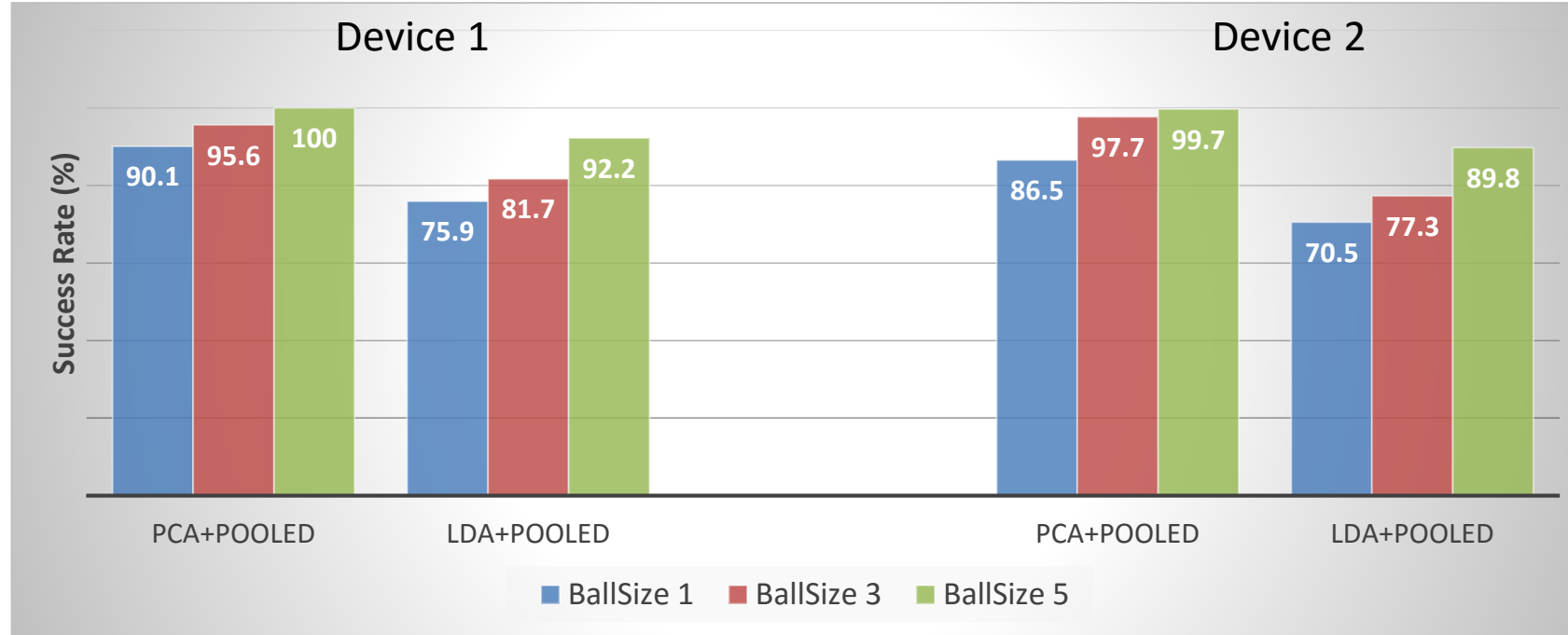


difference of covariance matrices

Side Channel Attacks of AES Wireless Keyboard

■ Enhanced Attacks : Case 2 – Difference device

- Rebuilding robust templates using both training device and target devices*
- Matching with robust templates to device 1, 2 respectively



*

O. Choudary and M. G. Kuhn, “Template Attacks on Different Devices”, in COSADE 2014

M. Renaud, F.-X. Standaert, N. Veyrat-Charvillon, D. Kamel, and D. Flandre, “A Formal Study of Power Variability Issues and Side-Channel Attacks for Nanoscale Devices”, in EUROCRYPT 2011

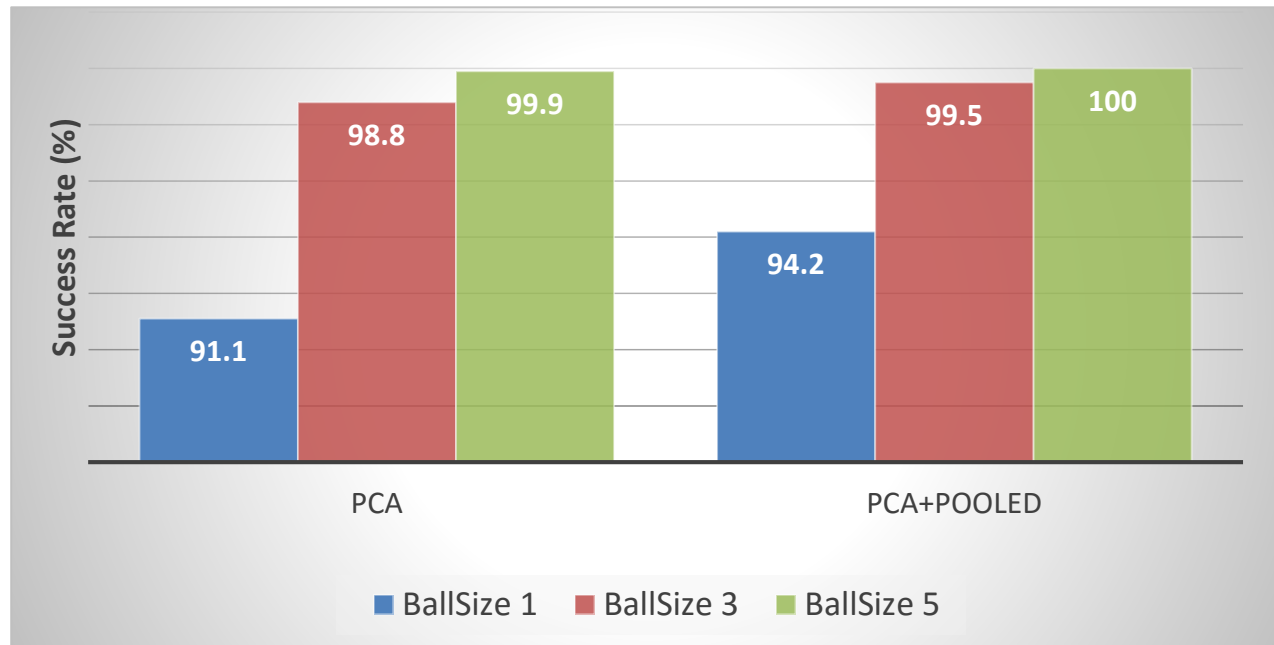
Side Channel Attacks of AES Wireless Keyboard

■ Proposed Attacks : Case 2 – Difference device

- Adjusting the difference between means of two devices by **correction vector**

$$\text{Correction vector} = \frac{\text{mean of target traces}}{\text{mean of template traces}}$$

- Rebuilding templates using corrected traces on training device
- Matching with corrected templates to target device



Side Channel Attacks of AES Wireless Keyboard

■ Result

- Success rate for case 1, 2

(%)

		the number of candidates			The used number of target trace
		1	3	5	
Same device		100	100	100	1
Difference device	Single	28.5	46.9	60.1	1
	Robust	88.3	96.65	99.85	1,000
	Corrected	94.2	99.5	100	256

※ Our attack requires the number of traces much less than the robust template attack

Summary

- Side-channel attack on secure wireless keyboard
- Setup for SCA environment using reverse engineering in Real World
- Performing EM analysis attacks to recover the AES key
- Template attack on the other keyboard

- Future research
 - We need more improved template attack
 - We need Far-field EM attack (long distance)
 - We need to apply it to other devices of similar types

Blackhat Sound Bytes

- Side-channel attack is a serious threats in real-world.
- Wireless keyboards with AES is not secure.
- Manufacturers need to defend against reverse engineering and side-channel attacks.

Thanks!

Any Questions?

Contact

Kwonyoup Kim, CEO/founder, kkyoup@sntworks.kr

Tae Hyun Kim, CTO, thkim@sntworks.kr

Taewon Kim, Senior Researcher, ktw@sntworks.kr

Sangryeol Ryu, Researcher, rsr@sntworks.kr

