black hat ASIA 2018 #BHASIA

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

SNTWORKS

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 -

- **Firmware & NVRAM Acquisition**
  - Keyboard (nRF24LE1)

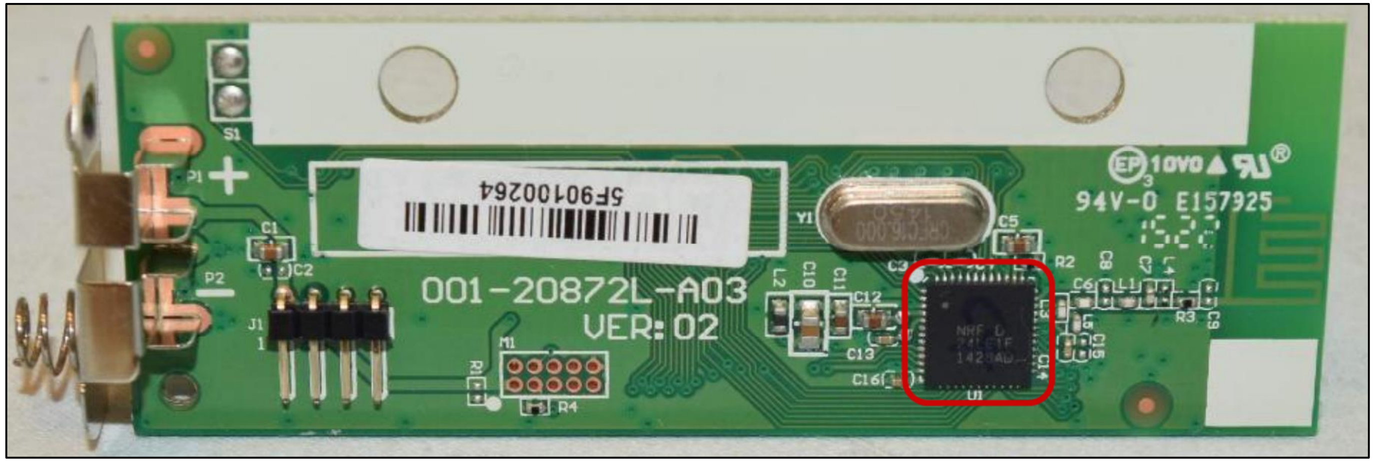
- **Static Analysis**
  - Disassembling firmware (IDA Pro)
  - Finding Encryption functions

- **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**



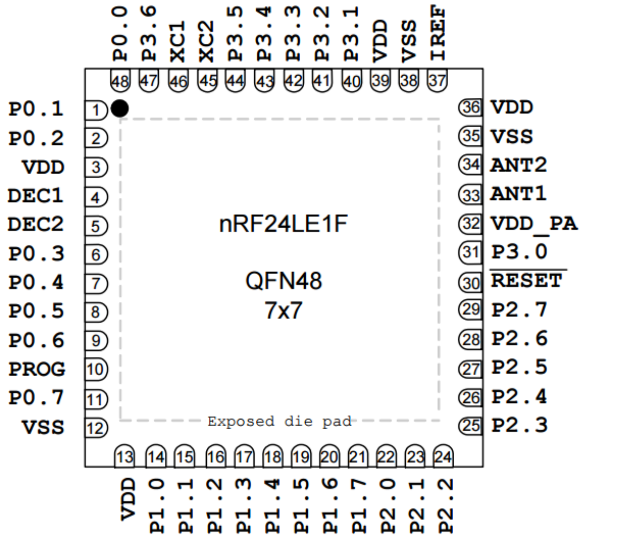


Figure 4. nRF24LE1F pin assignment (top view) for a QFN48 7×7 mm package

- **Firmware Acquisition – nRF24LE1**
  - Flash memory : features

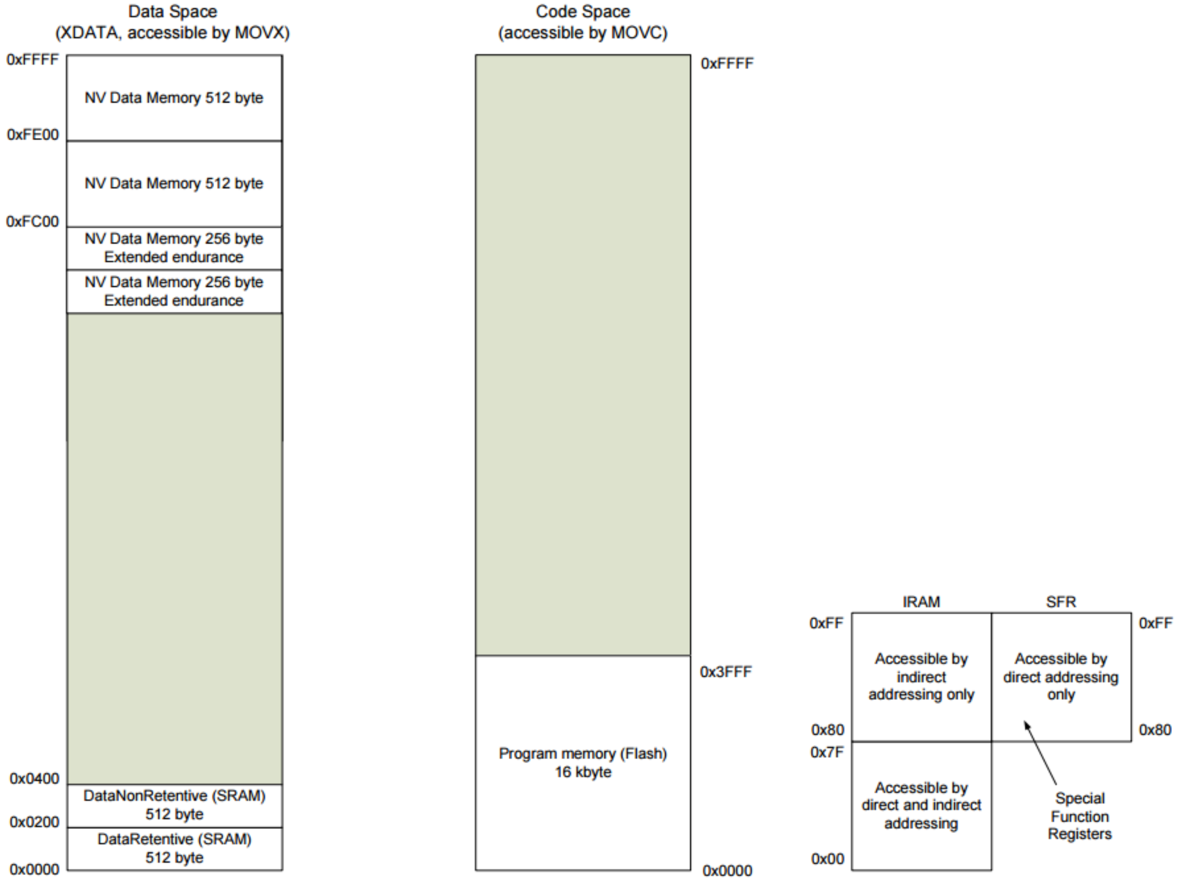
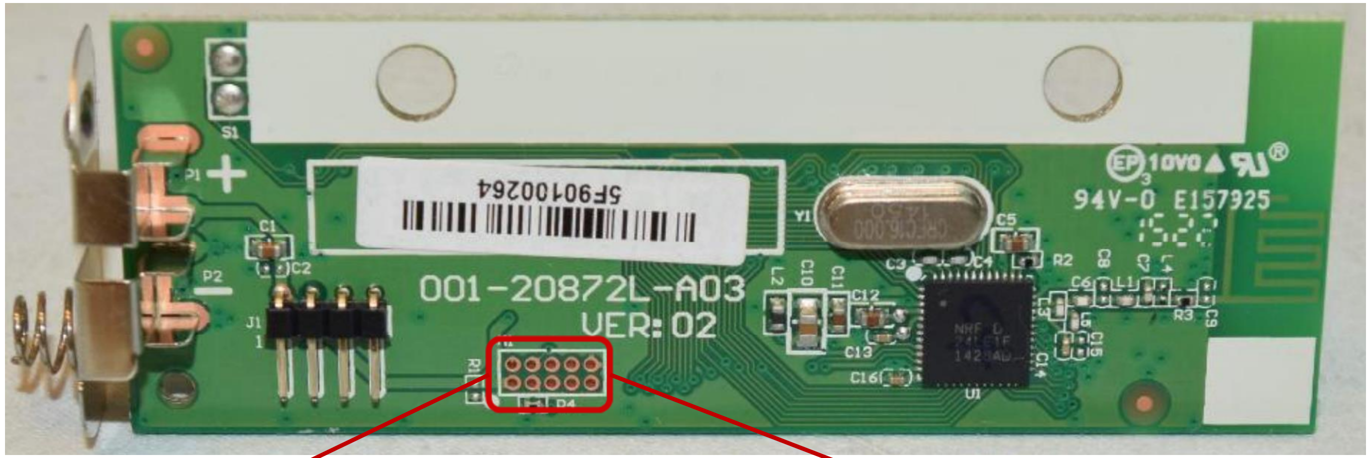


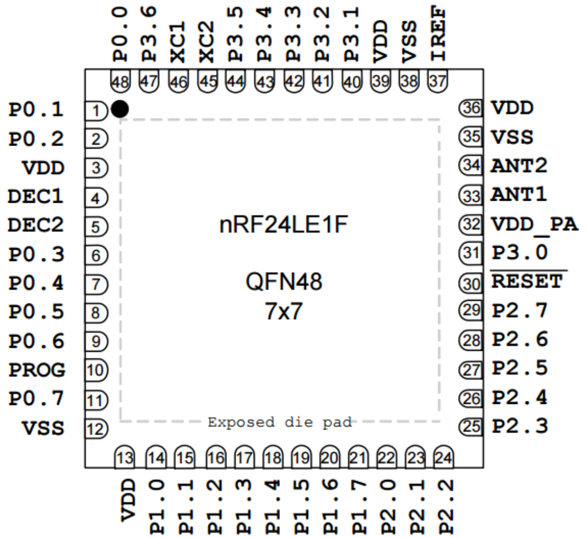
Figure 30. Memory map

- 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

- 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 µs 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.

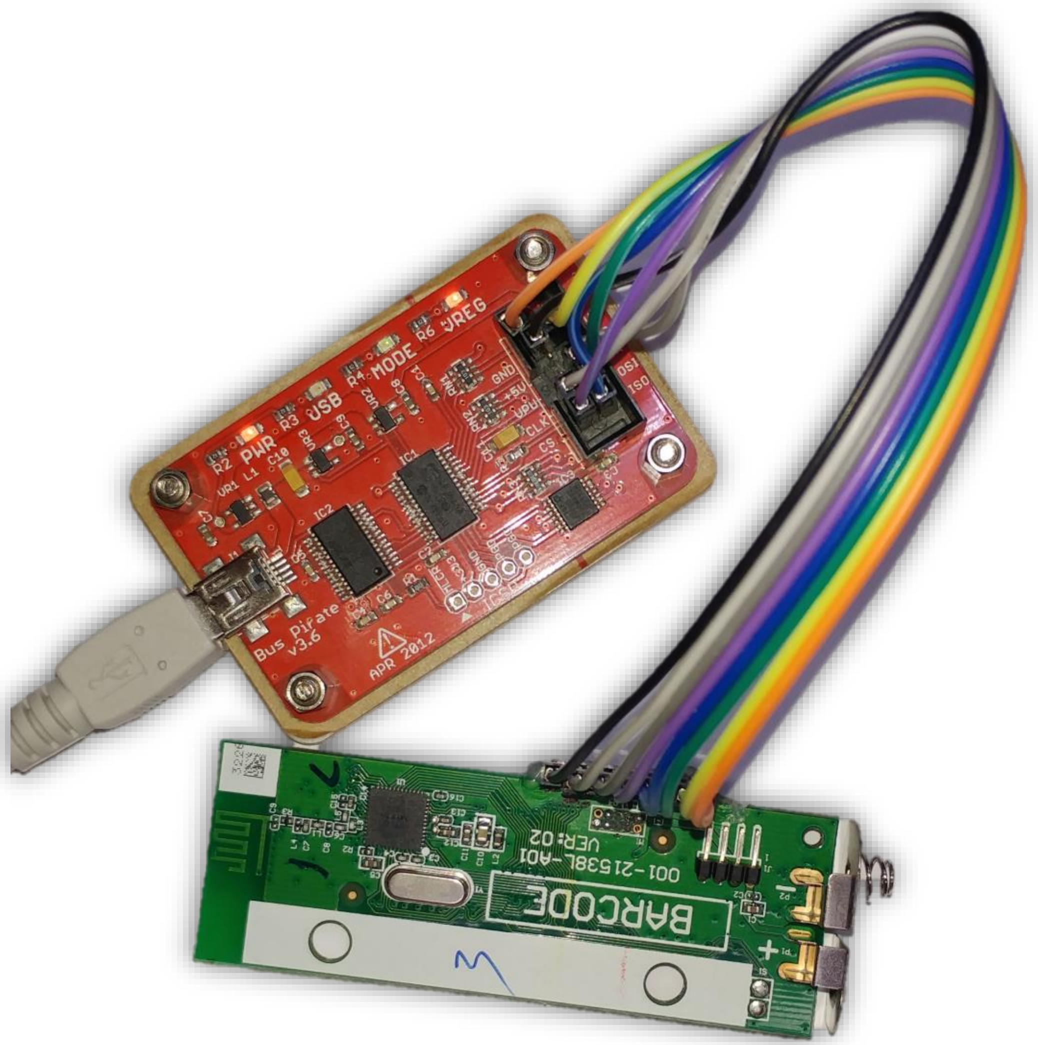
	24pin-4×4	32pin-5×5	48pin-7×7
FCSN	P0.5	P1.1	P2.0
FMISO	P0.4	P1.0	P1.6
FMOSI	P0.3	P0.7	P1.5
FSCK	P0.2	P0.5	P1.2

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

Reverse Engineering of AES Wireless Keyboard

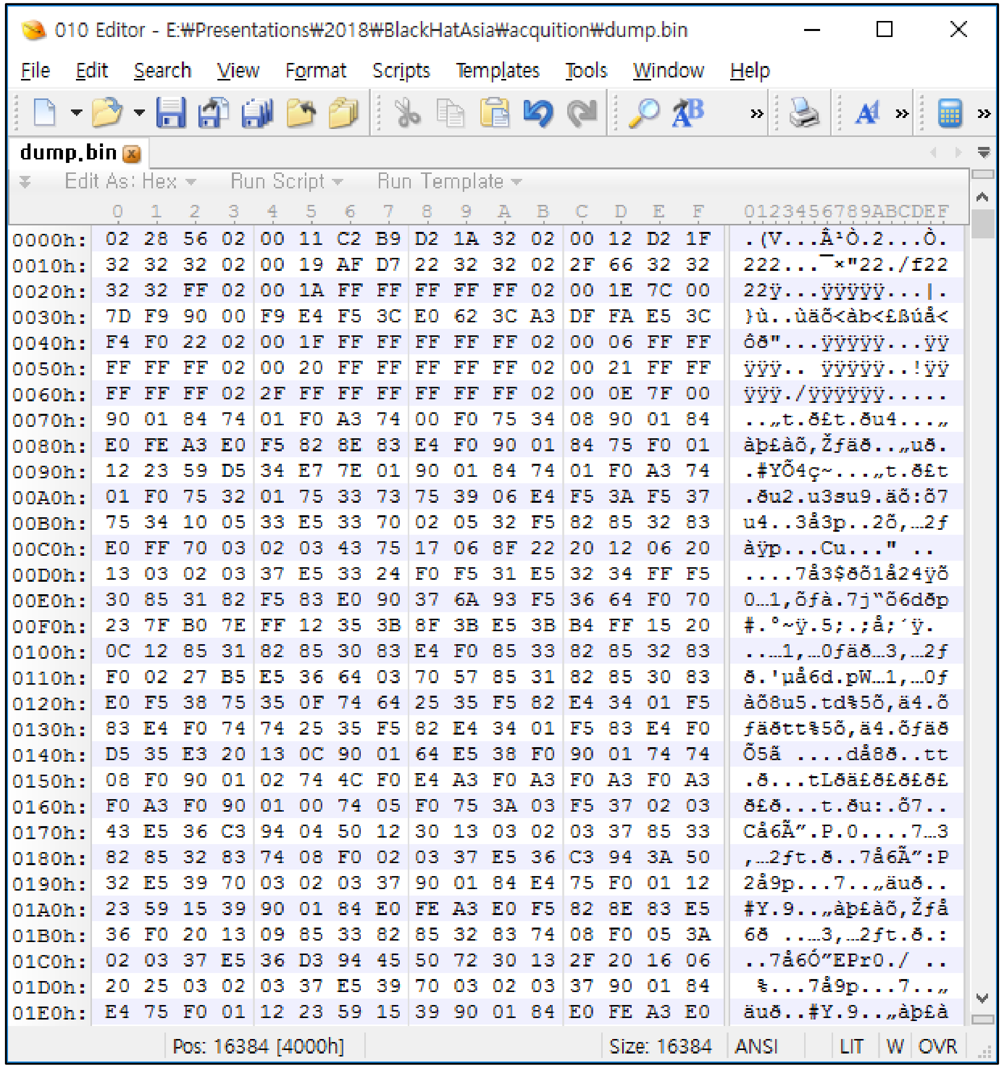
- **Firmware Acquisition – nRF24LE1**
  - Flash memory Dump

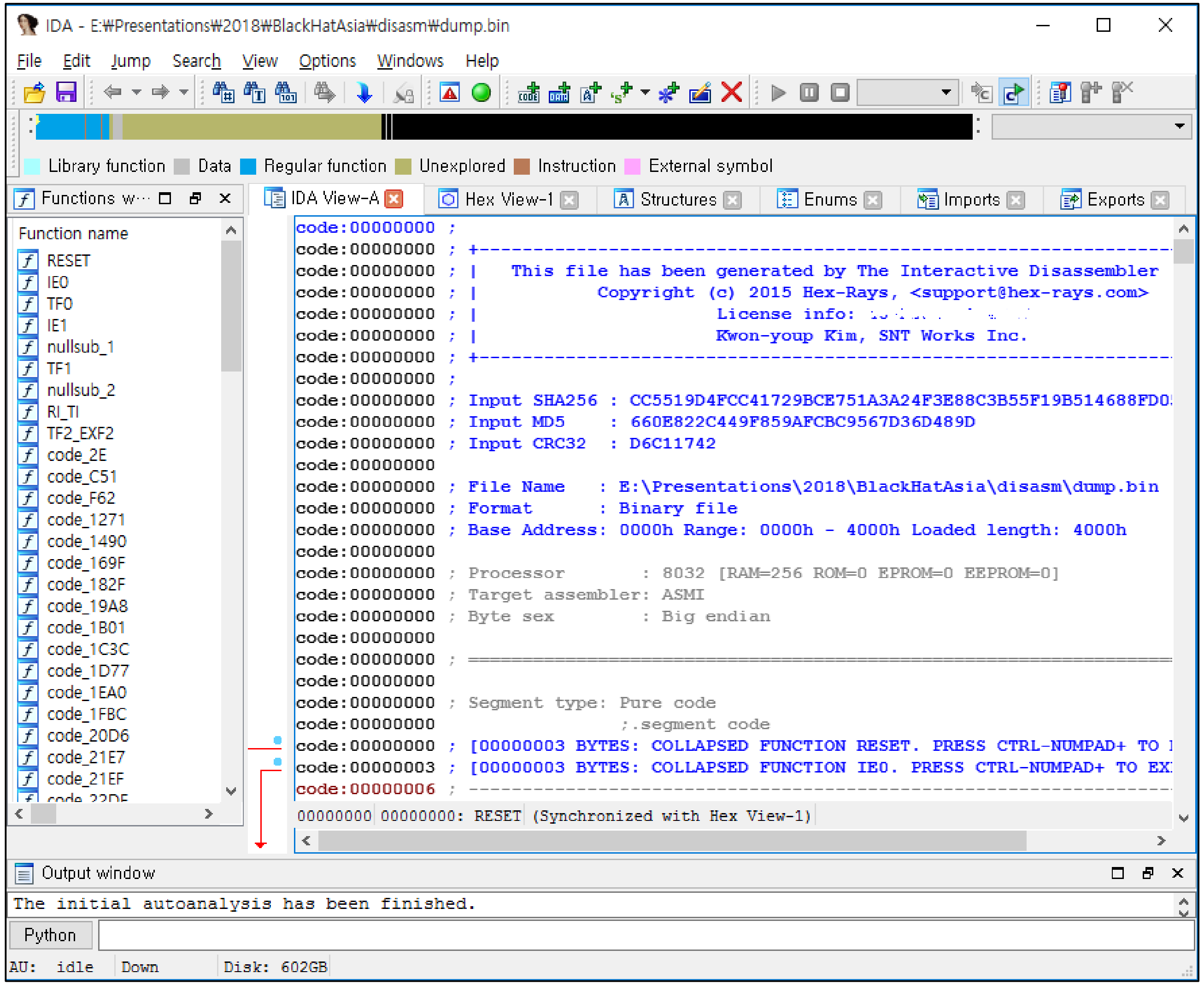
```
$ ./nrfprog /dev/ttyUSB0 dump.bin
Opening the Bus Pirate UART
Setting the Bus Pirate to Binary Mode
Configuring SPI mode
Backing up the info page
Starting read operation
Reading from the device to dump.bin
Starting read operation
Putting the Bus Pirate back in normal operating mode
Closing the Bus 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





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
```

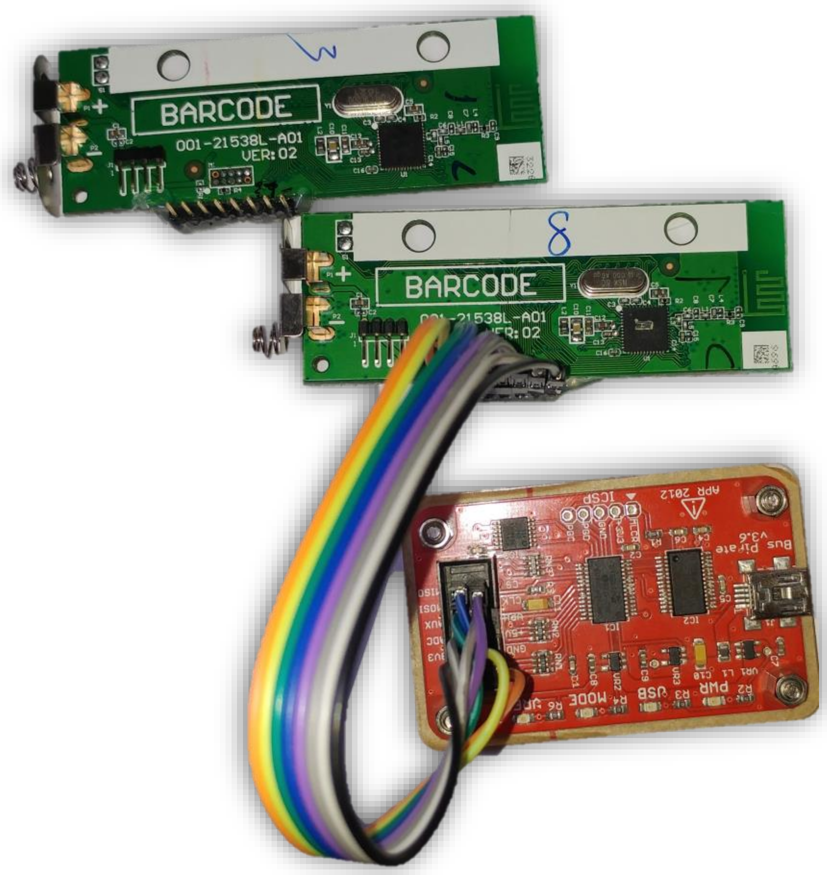
```c
void hal_aes_crypt(uint8_t * dest_buf, const uint8_t * src_buf)
{
#ifdef __C51__
  uint8_t data aes_state[16];      //AES State
#endif

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

  //Start
  CCPDATIB = 0x02;  //Set co-prosessor the GF(2^8)*2 (used in mix-colums)
  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
    }
}
```

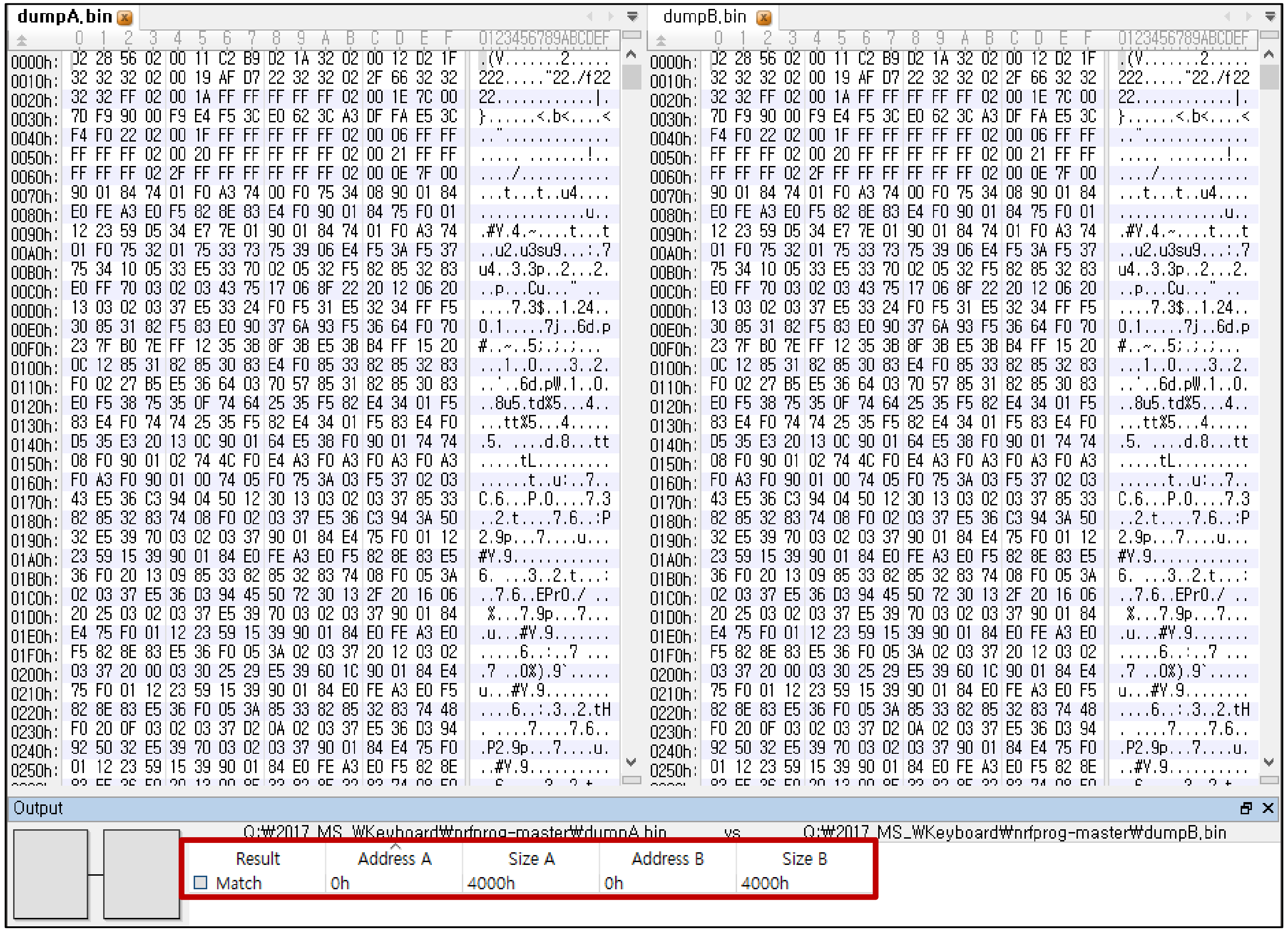
nRFgo SDK source : hal_aes.c

```
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
```

Reverse Engineering of AES Wireless Keyboard

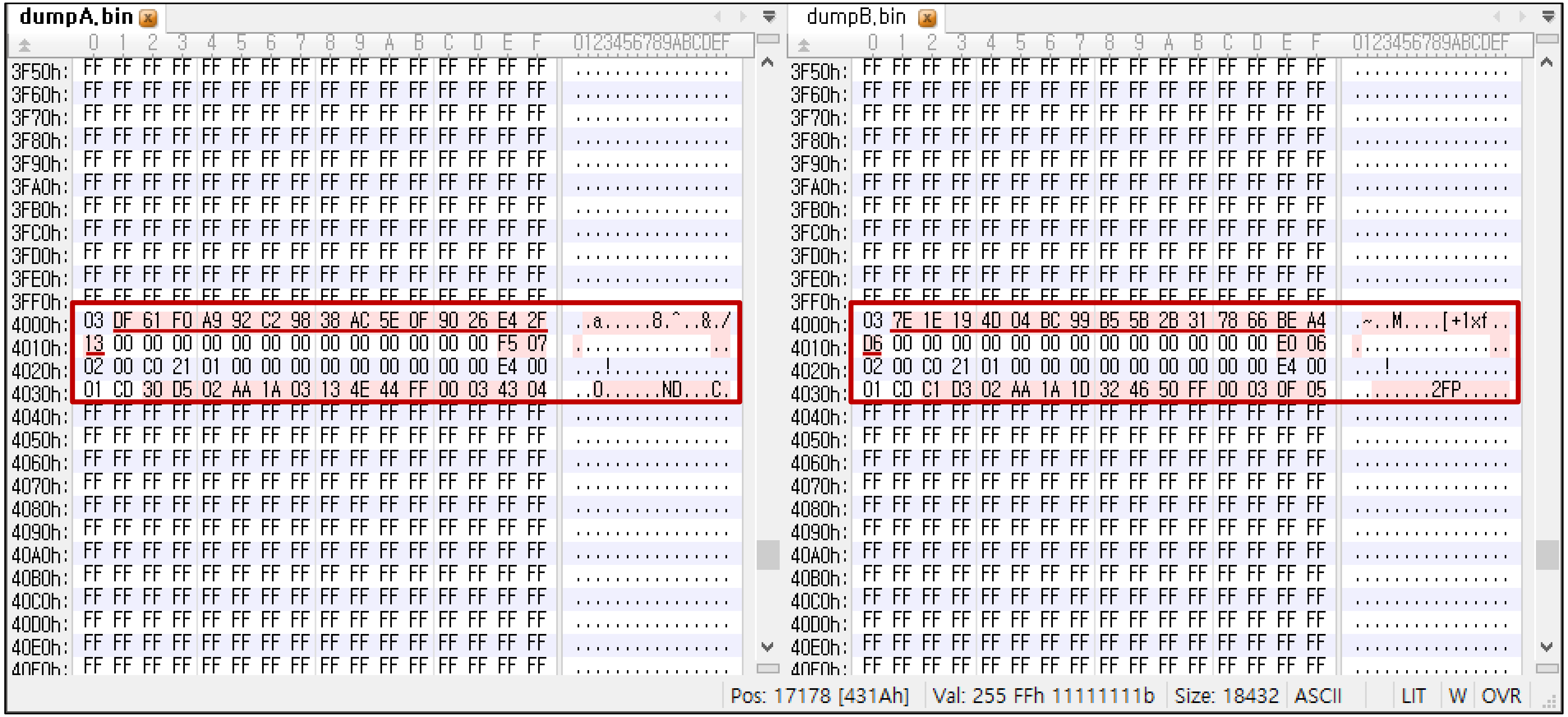
- **Static Analysis – Finding AES key**
  - Flash Dump from another keyboard





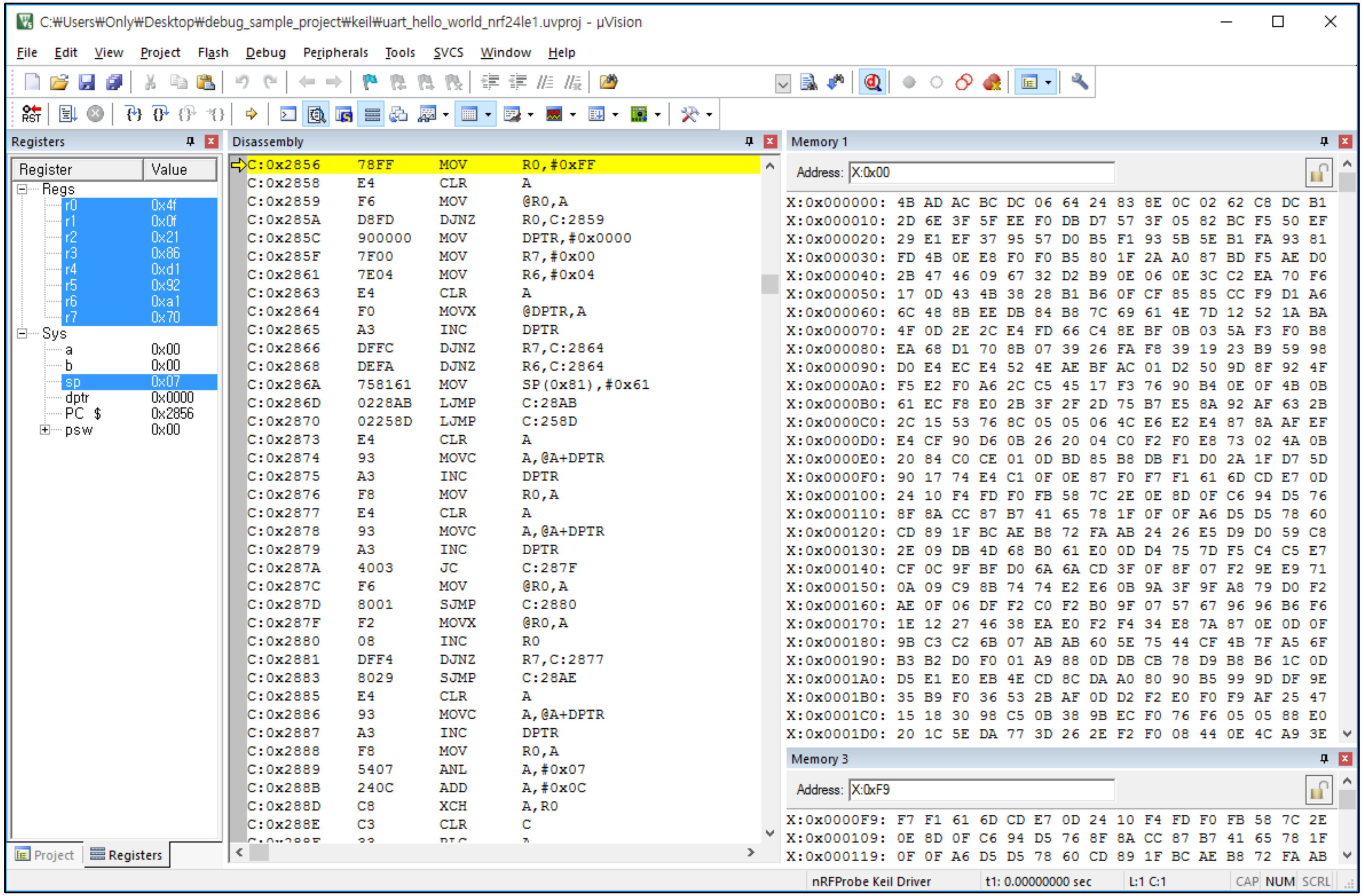
Reverse Engineering of AES Wireless Keyboard

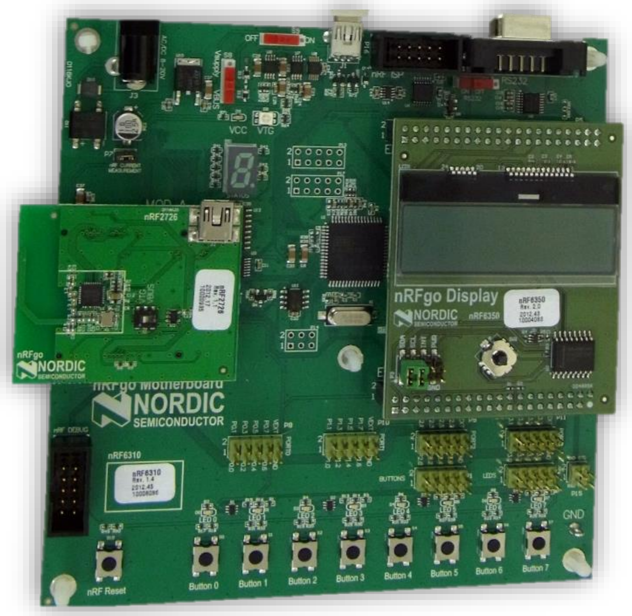
- **Static Analysis – Finding AES key**
  - NVRAM are not the same values
  - Whether or not the AES key is validated using debugging



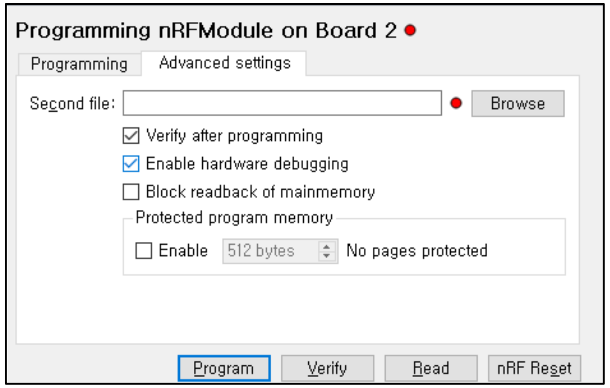
Reverse Engineering of AES Wireless Keyboard

- ## Dynamic Analysis – Debugging codes



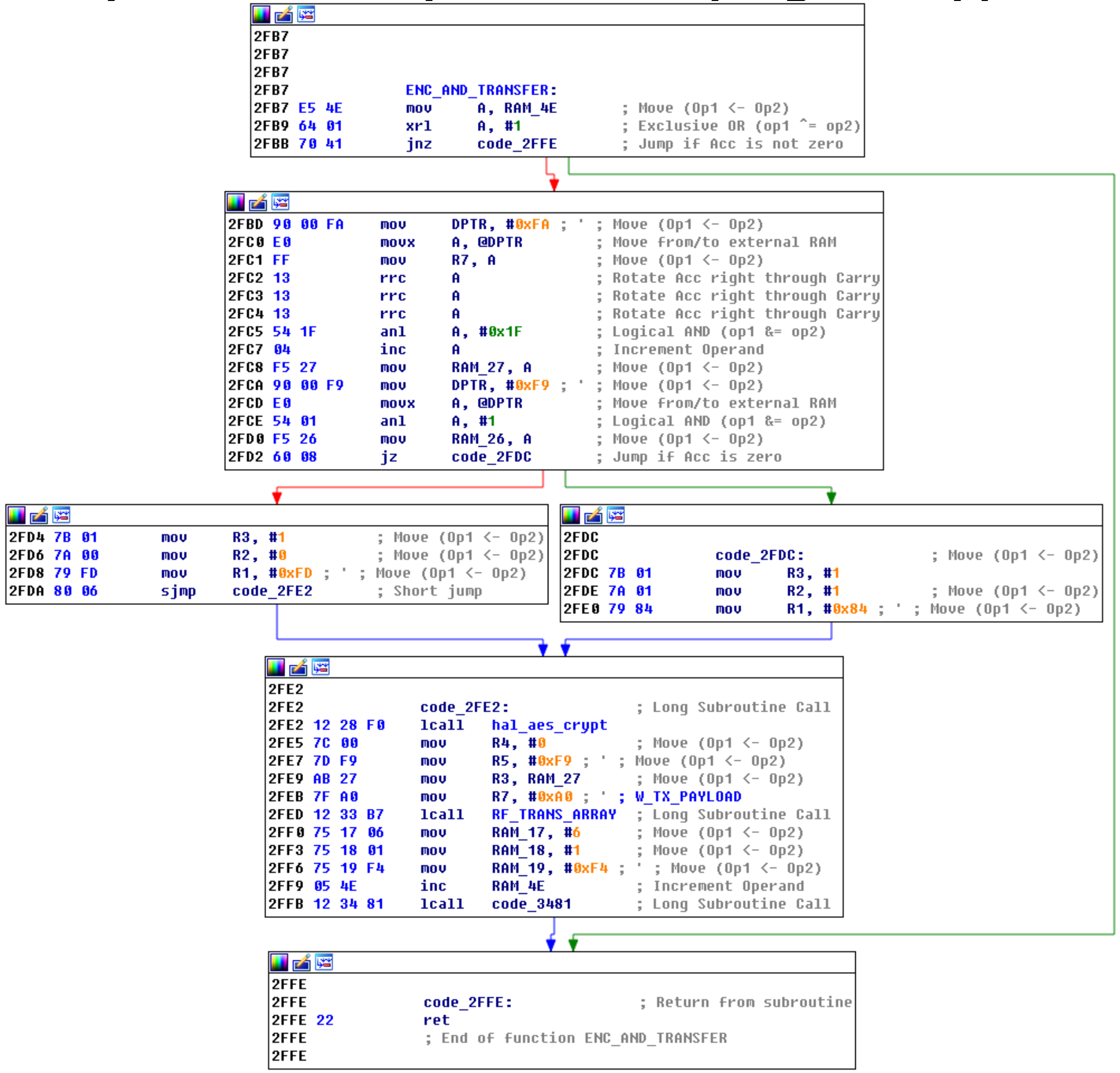


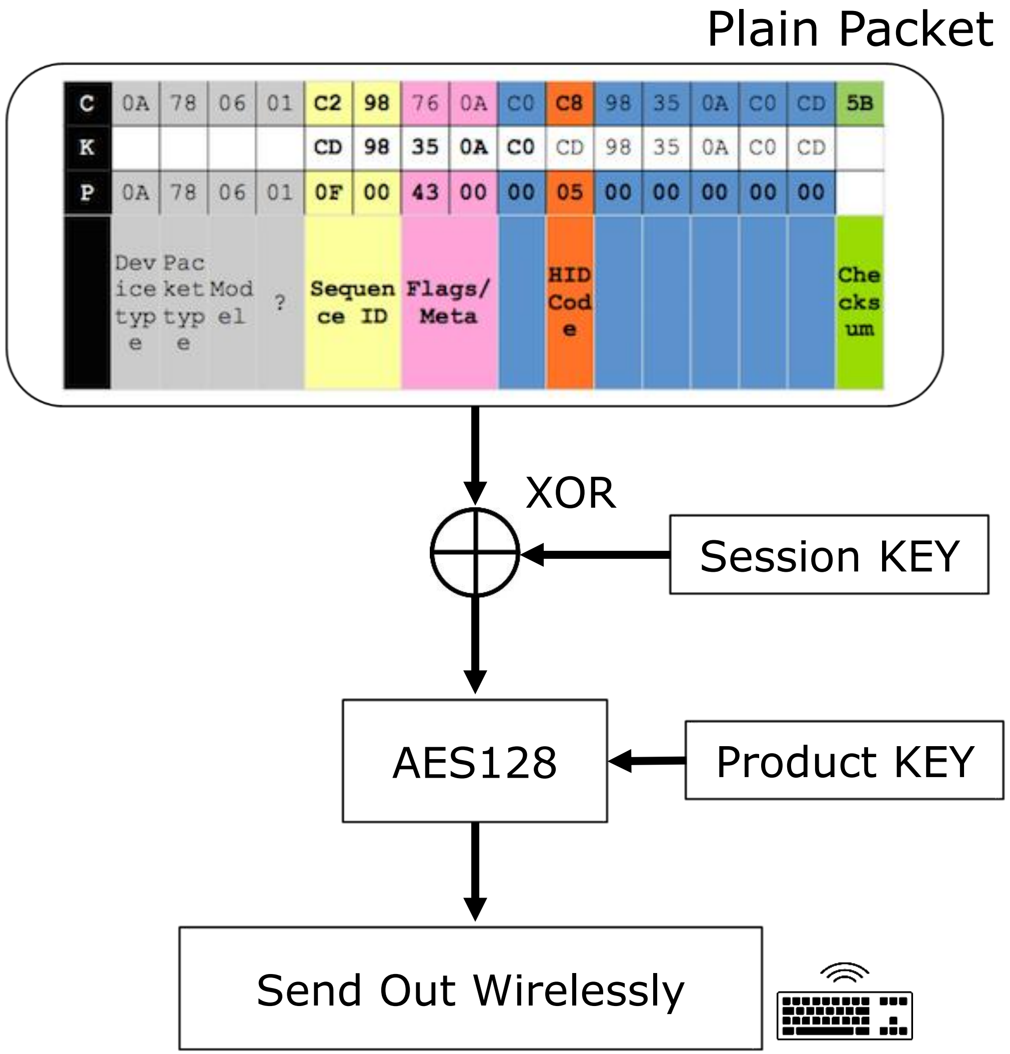
nRFgo Start Kit & nRF24LE1



Reverse Engineering of AES Wireless Keyboard

- **Dynamic Analysis – Verifying Encryption functions**





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

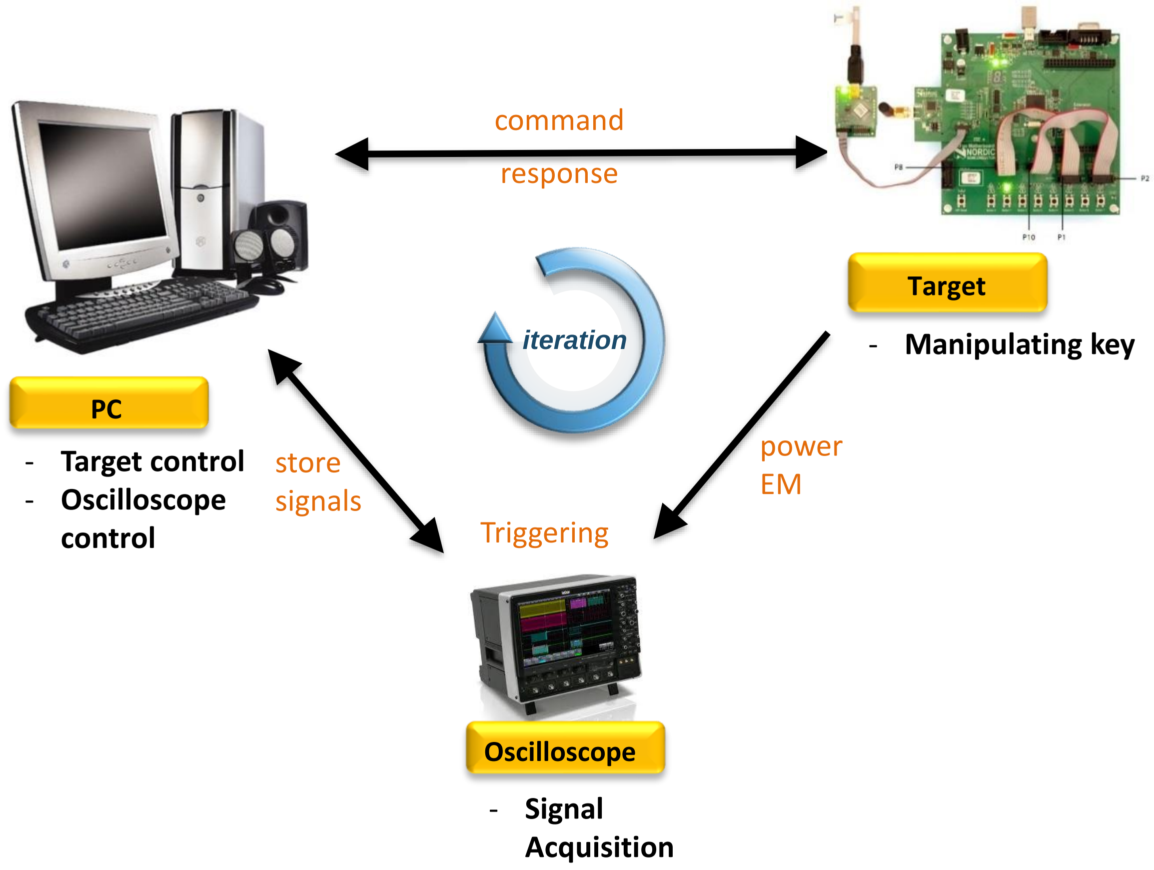
- **Firmware Modification – Reprogramming of Keyboard firmware**
  - Write protection is set for the target keyboard
  - So, We changed it to a reprogrammed chip for Side channel analysis

AES Wireless Keyboard

- Side Channel Attacks, Template Attacks -

Side Channel Attacks of AES Wireless Keyboard

- Side Channel Attacks in Theory

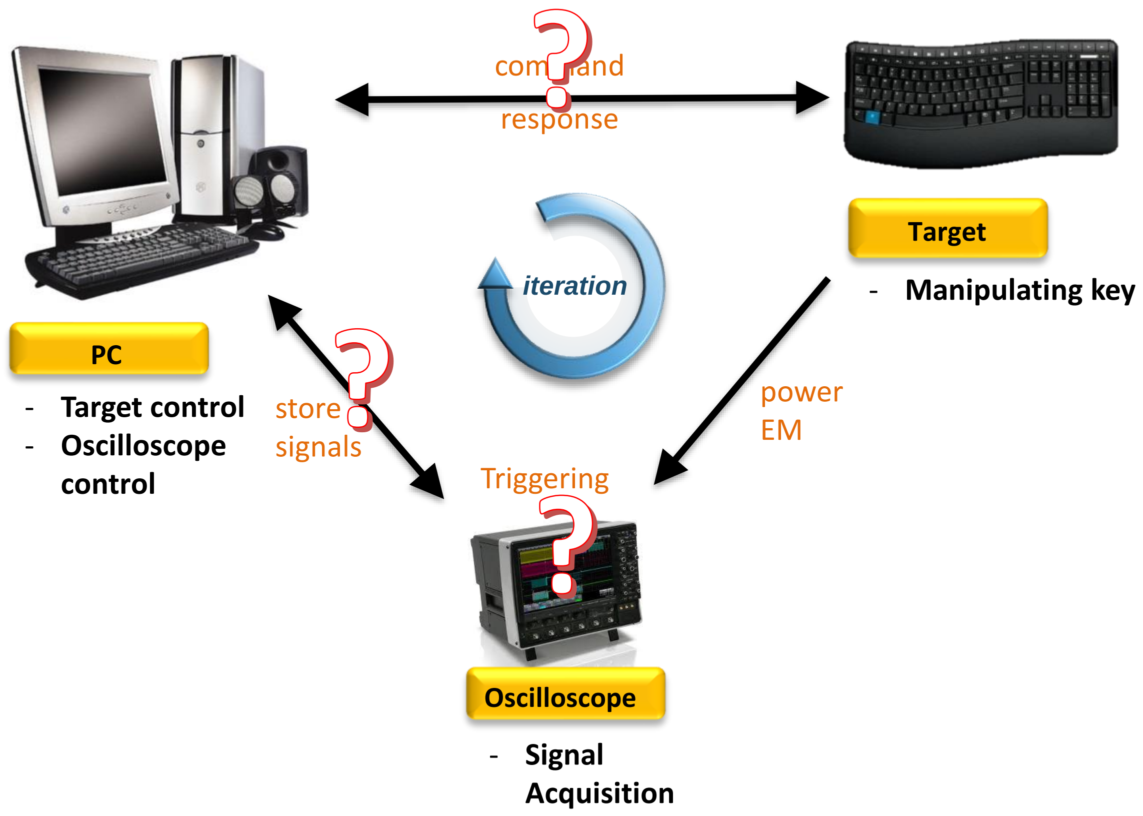


White-box Attack Model

- 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

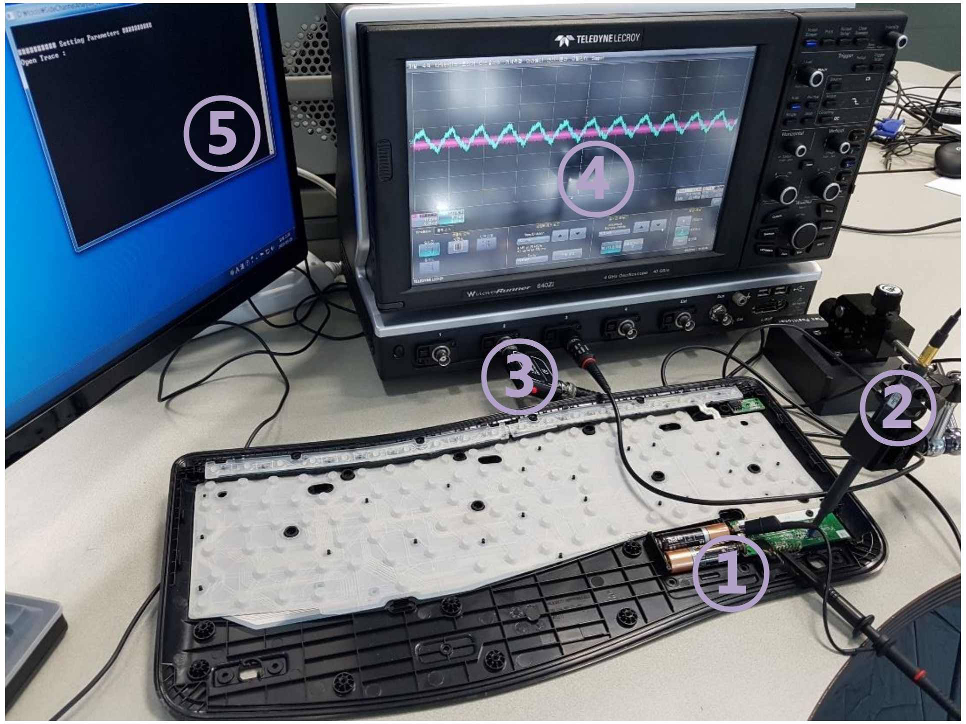


<u>Black-box Attack Model</u>

- 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

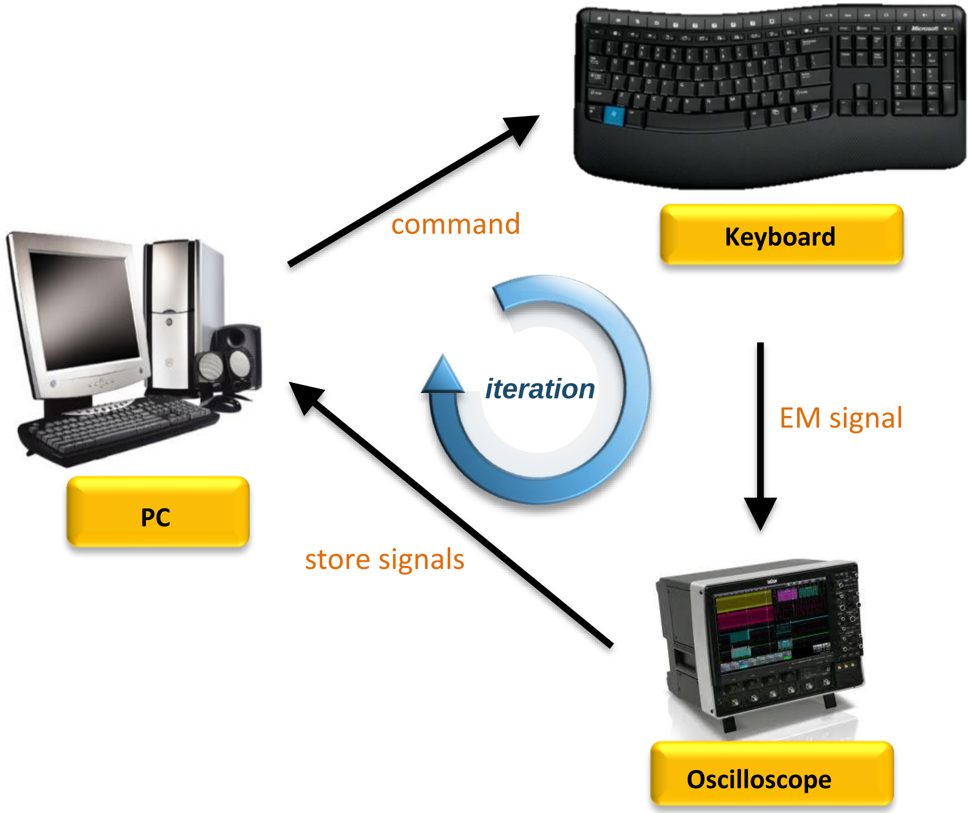
③ Langer PA303 pre-amplifier

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

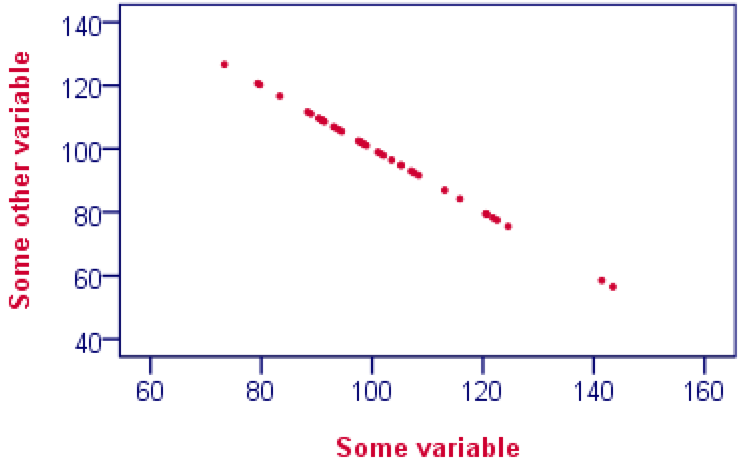
Side Channel Attacks of AES Wireless Keyboard

- CPA (Correlation Power Analysis)

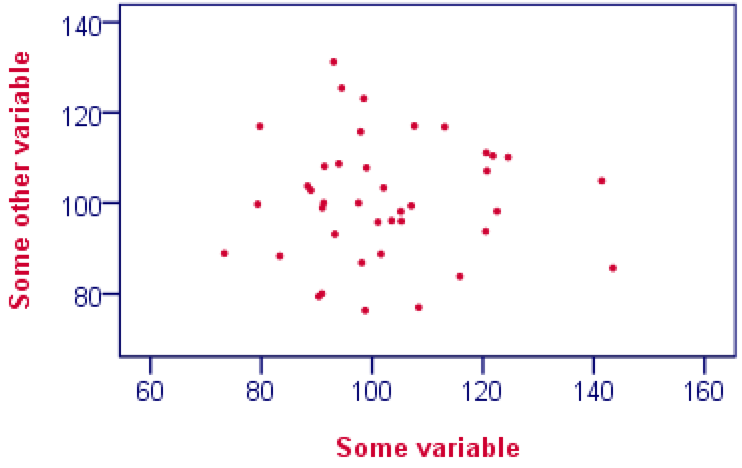
  - Pearson Correlation Coefficient
  - T : measurement, P : predictions
  - Number between -1 and 1

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

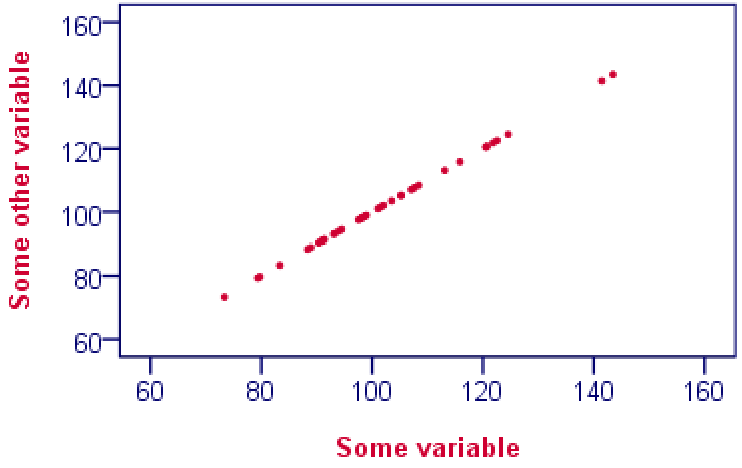
-1 : total negative linear correlation



0 : no linear correlation

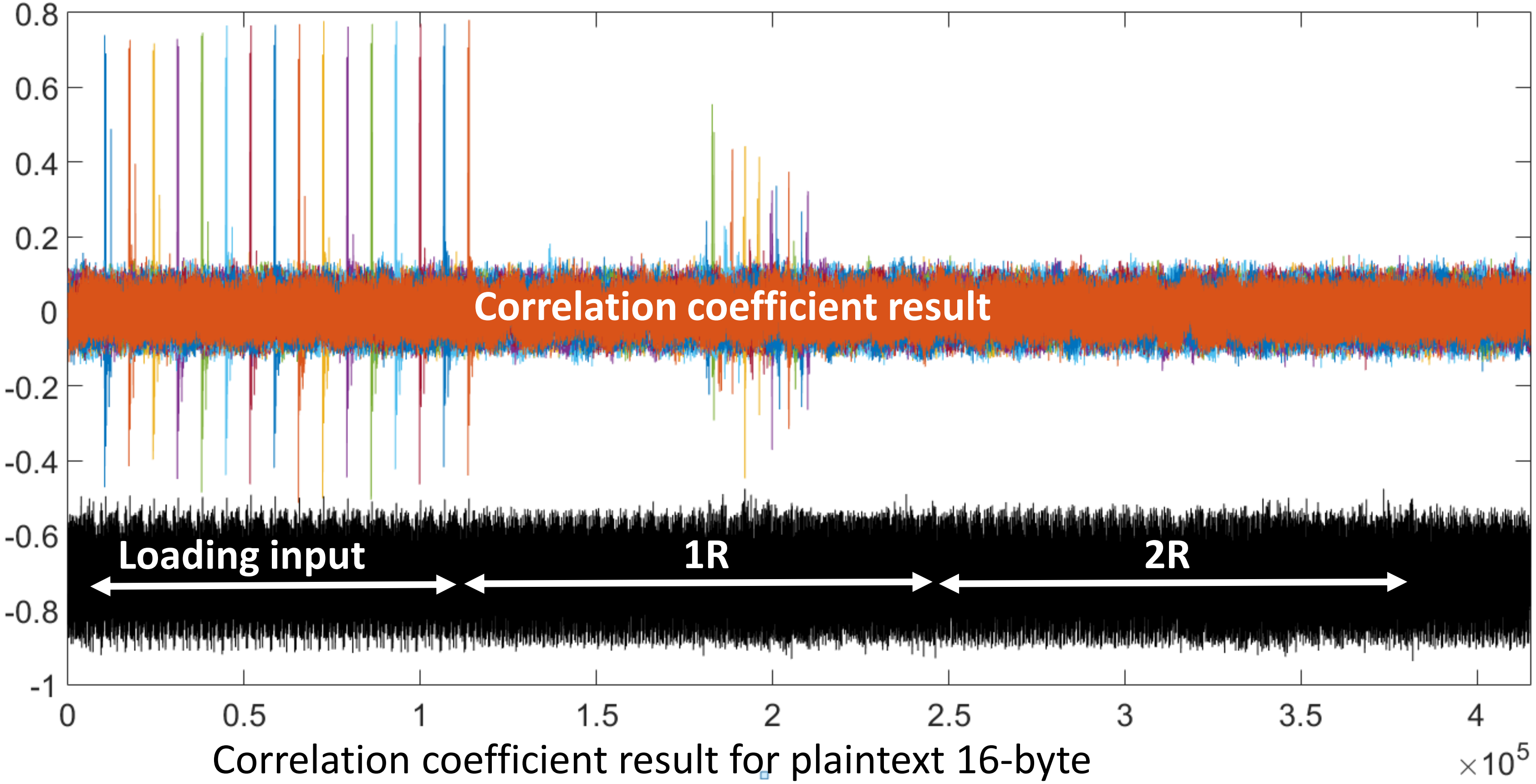


1 : total positive correlation



Side Channel Attacks of AES Wireless Keyboard

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

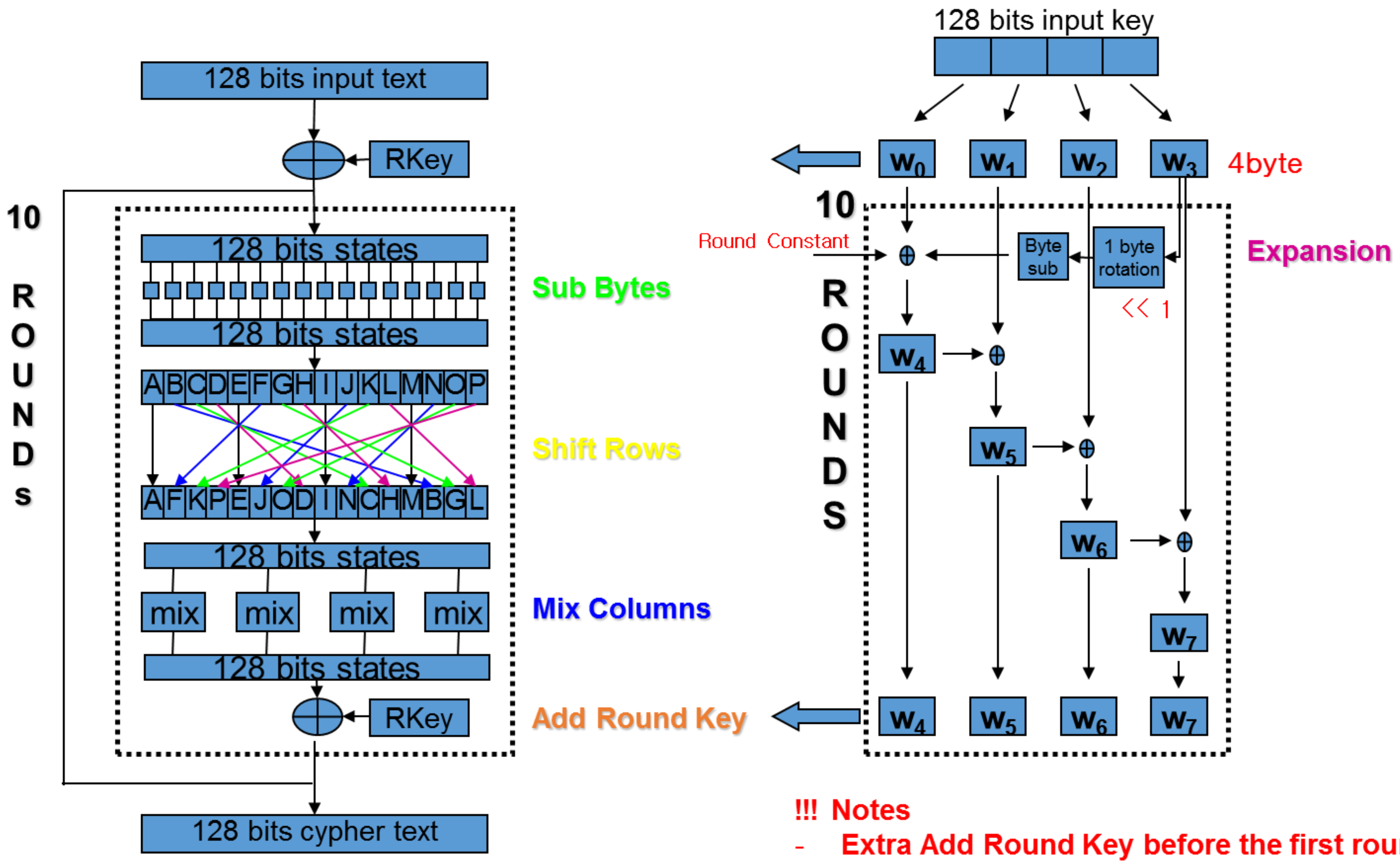


Correlation coefficient result for plaintext 16-byte

- HW model
- 1K traces
- 1M samples

Side Channel Attacks of AES Wireless Keyboard

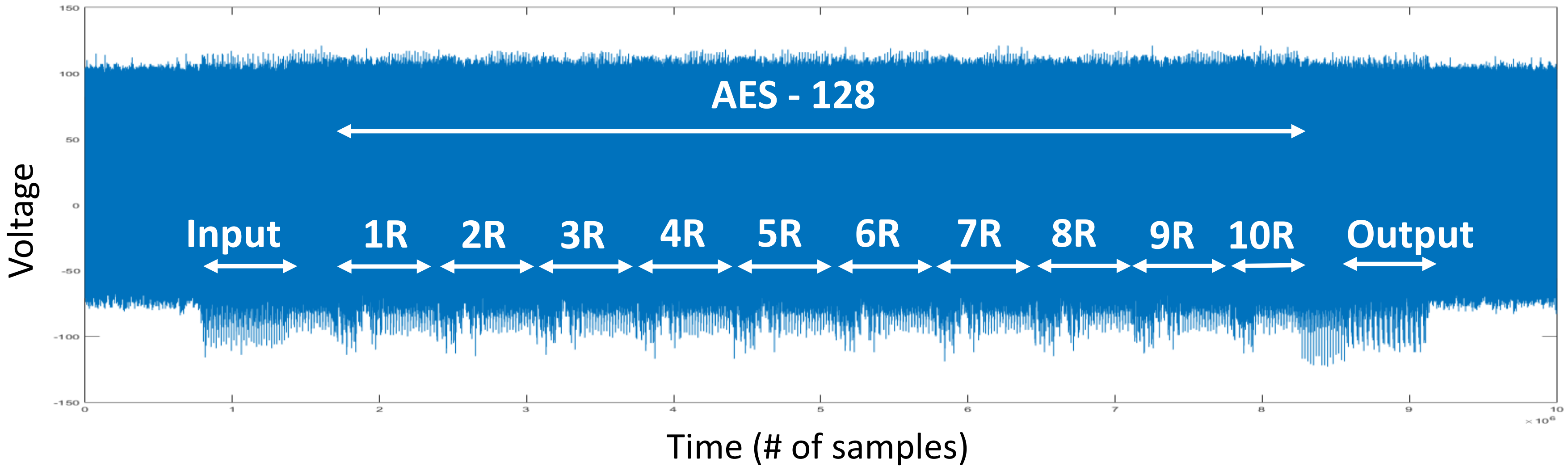
- AES (Advanced Encryption Standard)



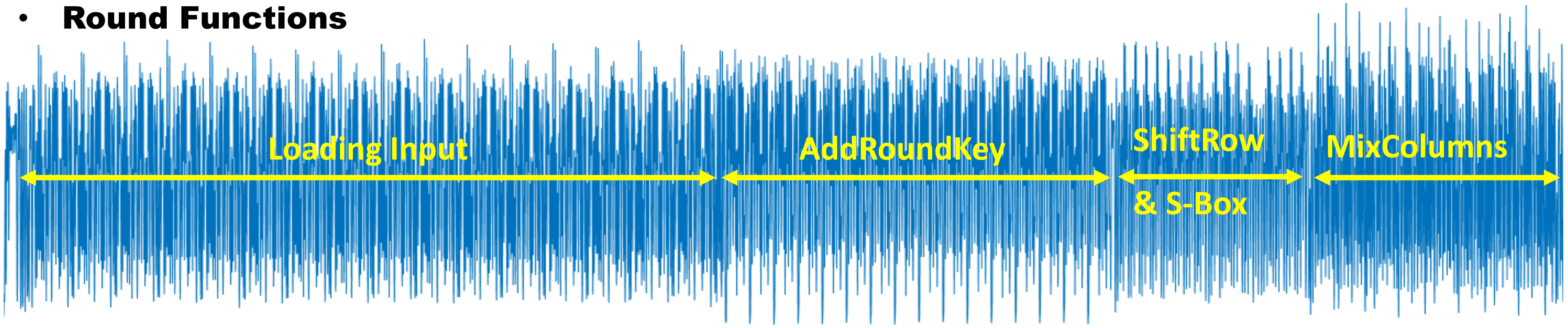
!!! Notes
- Extra Add Round Key before the first round
- No Mix Columns in the last round

Side Channel Attacks of AES Wireless Keyboard

- Profiling EM Trace



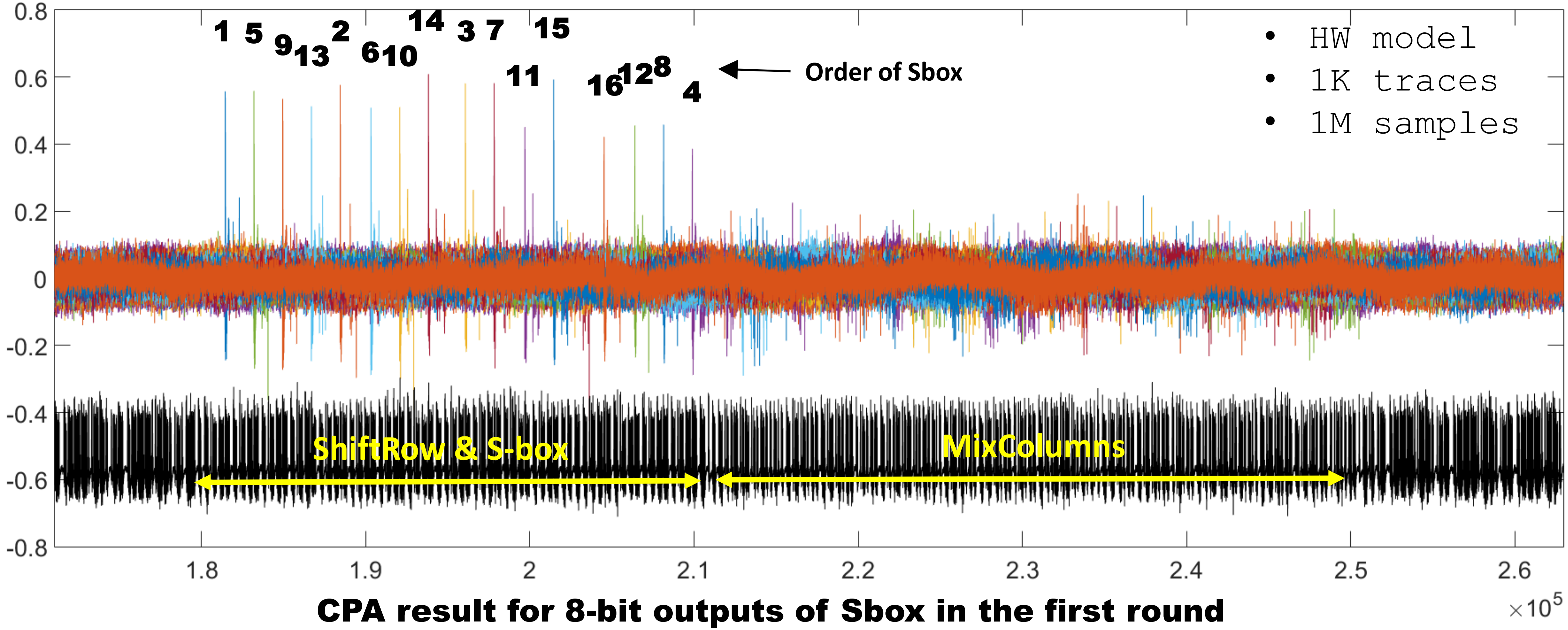
- **Round Functions**



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

Side Channel Attacks of AES Wireless Keyboard

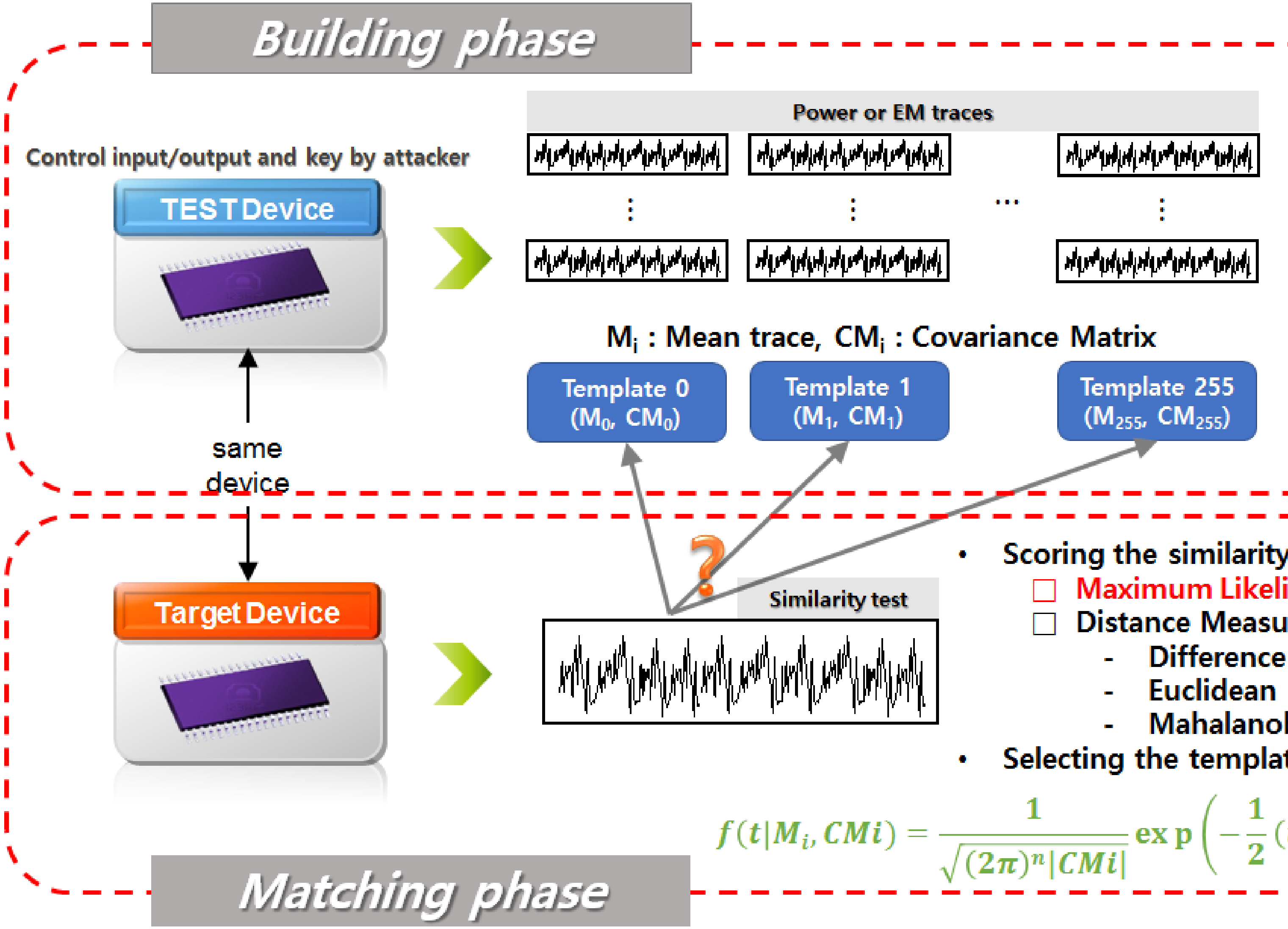
- CPA (Correlation Power Analysis)
  - Recover the secret key
  - Target variable : S-Box output



CPA result for 8-bit outputs of Sbox in the first round

Side Channel Attacks of AES Wireless Keyboard

- Basic Template Attacks



Making templates with characterized key-dependent operations

» Statistical model (Multivariate Gaussian distribution)

$$M_i = \frac{1}{|S_i|}\sum_{t\in Si} t$$

$$CM_i = \frac{1}{|S_i|-1}\sum_{t\in Si}(t-Mi)(t-Mi)^T$$

- Scoring the similarity of the target trace with each template
  - ☐ Maximum Likelihood Estimation
  - ☐ Distance Measure
    - Difference of Means
    - Euclidean
    - Mahalanobis
- Selecting the template with highest similarity

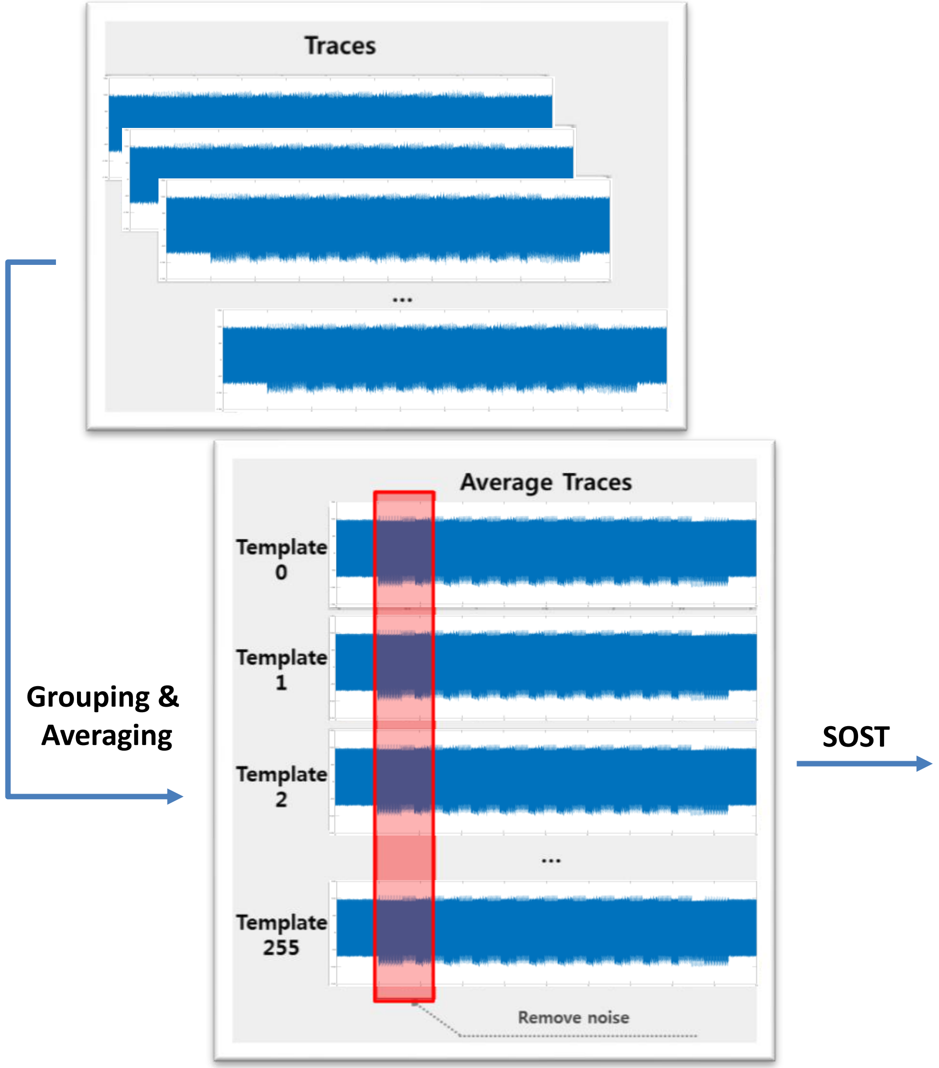
$$f(t|M_i, CMi) = \frac{1}{\sqrt{(2\pi)^n|CMi|}}\exp\left(-\frac{1}{2}(t-Mi)^T CM_i^{-1}(t-Mi)\right), n: \#POI$$

Side Channel Attacks of AES Wireless Keyboard

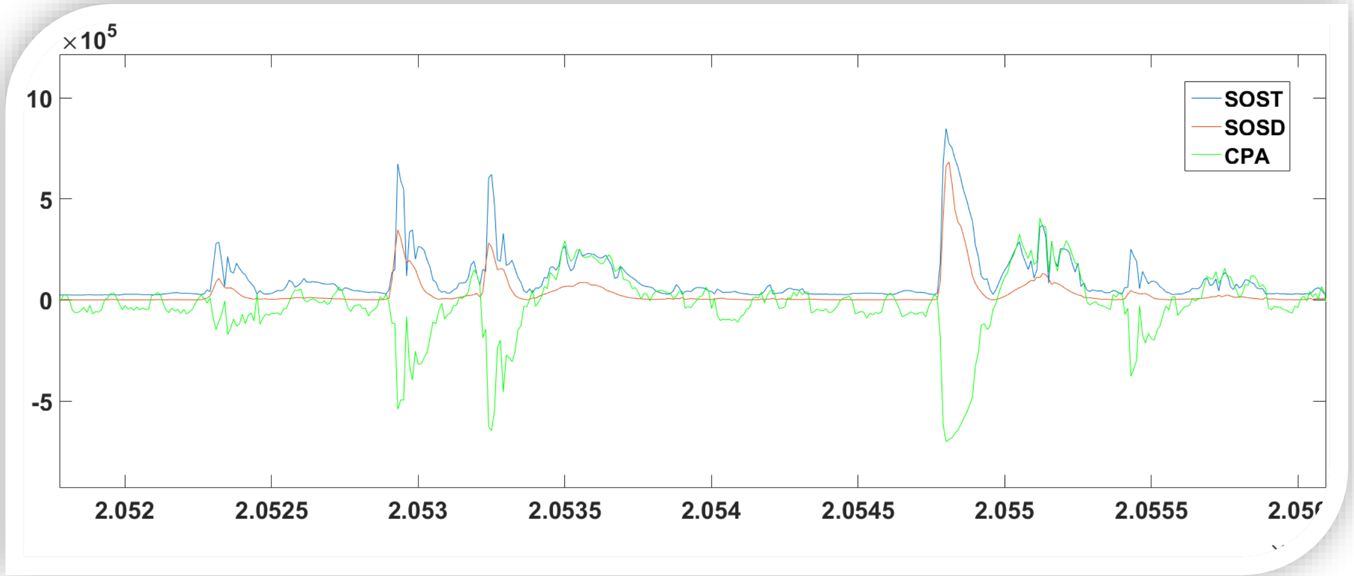
- **Selecting Interesting Points**
  - Computational or memory restrictions
  - Reducing the number of samples and the size of templates
  - Selection of some special samples that contain the most information about the characterized key-dependent operation
    - ✓ DOM (Sum of pairwise Differences)
    - ✓ 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)
    - ✓ …

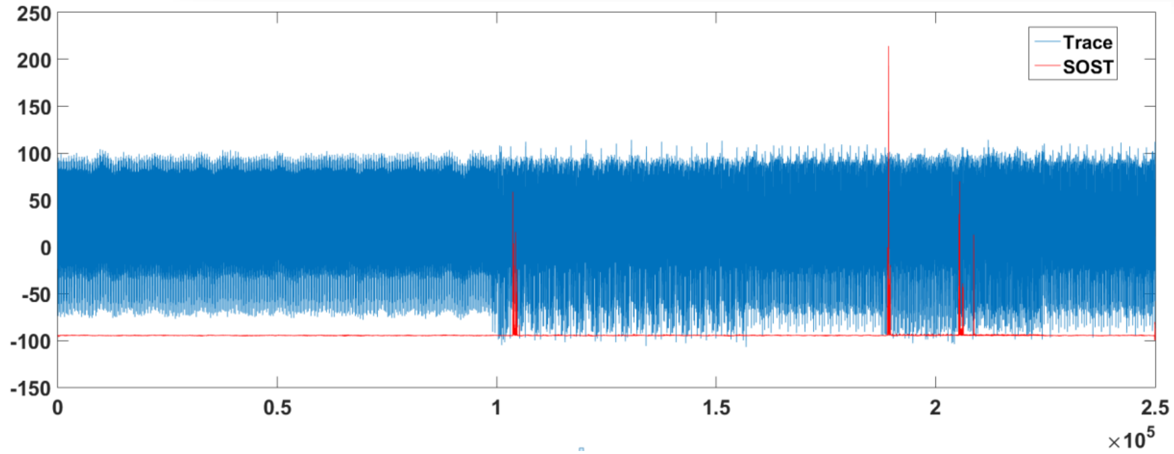
Side Channel Attacks of AES Wireless Keyboard

- Selecting Interesting Points – SOST



Comparison





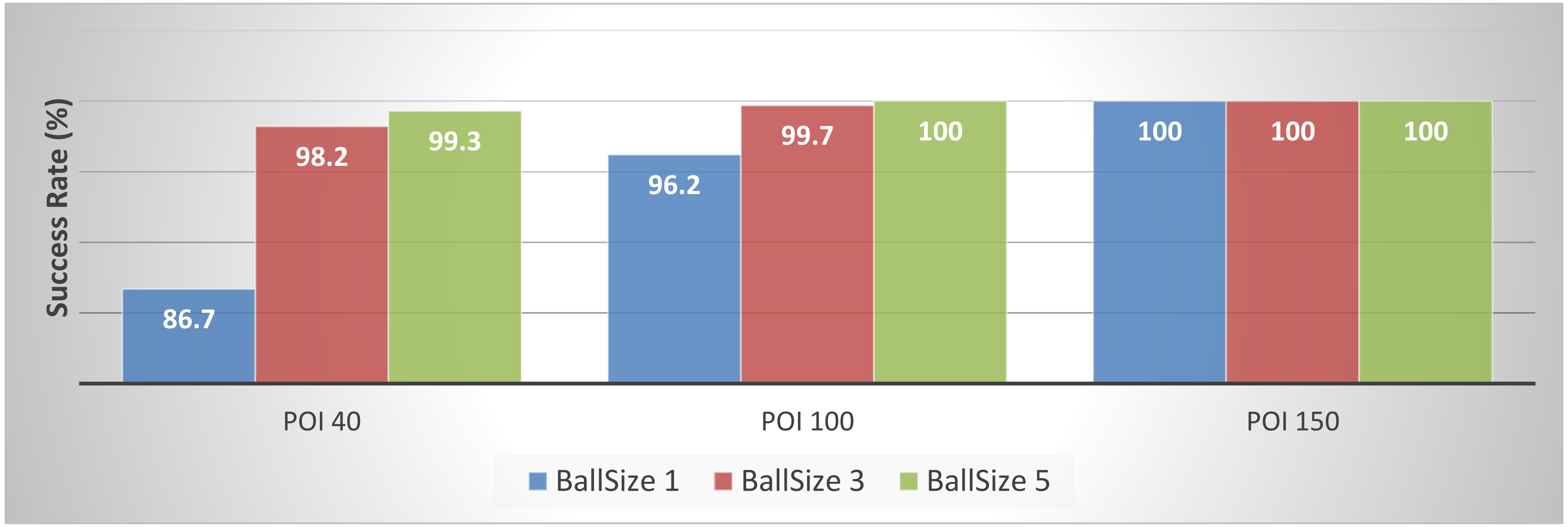
Side Channel Attacks of AES Wireless Keyboard

- **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 ≠ matching device)

Side Channel Attacks of AES Wireless Keyboard

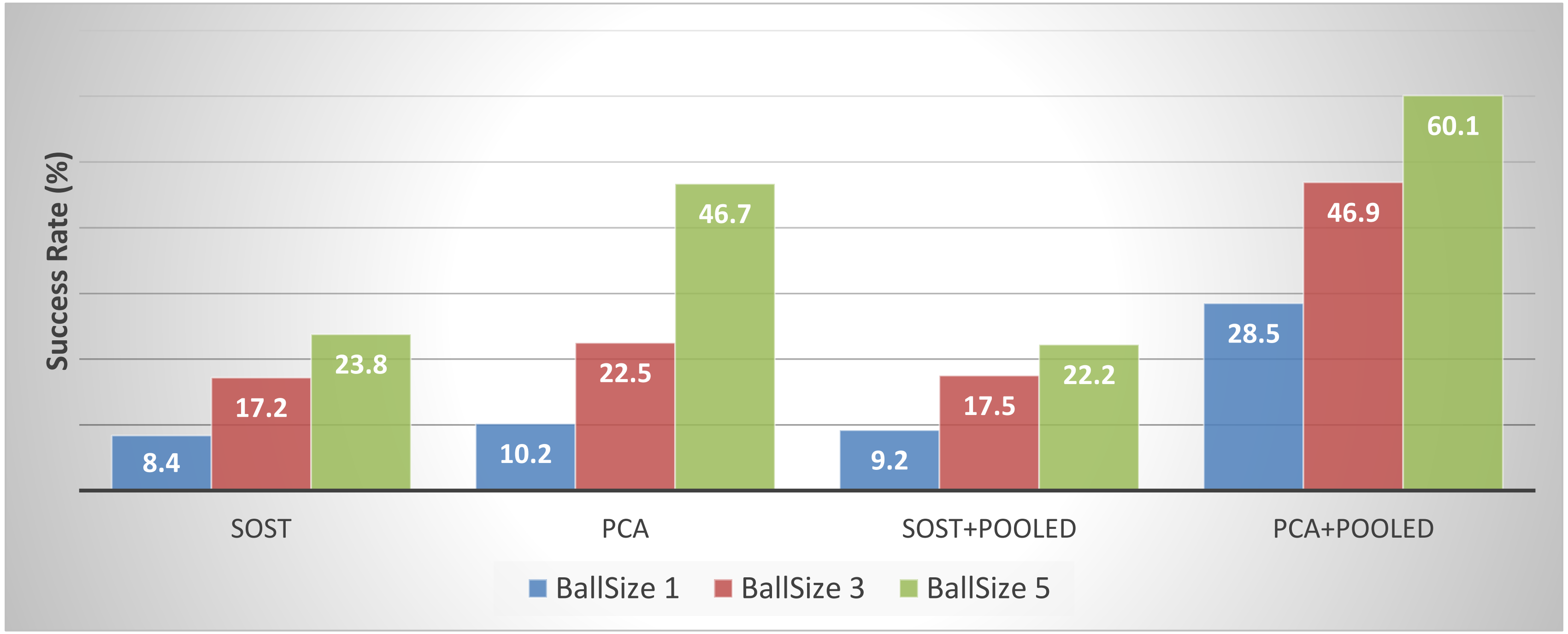
- **Template Attacks : Case 1 – Same device**
  - Selecting POI : SOST
  - Template Matching : Maximum Likelihood



- **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.

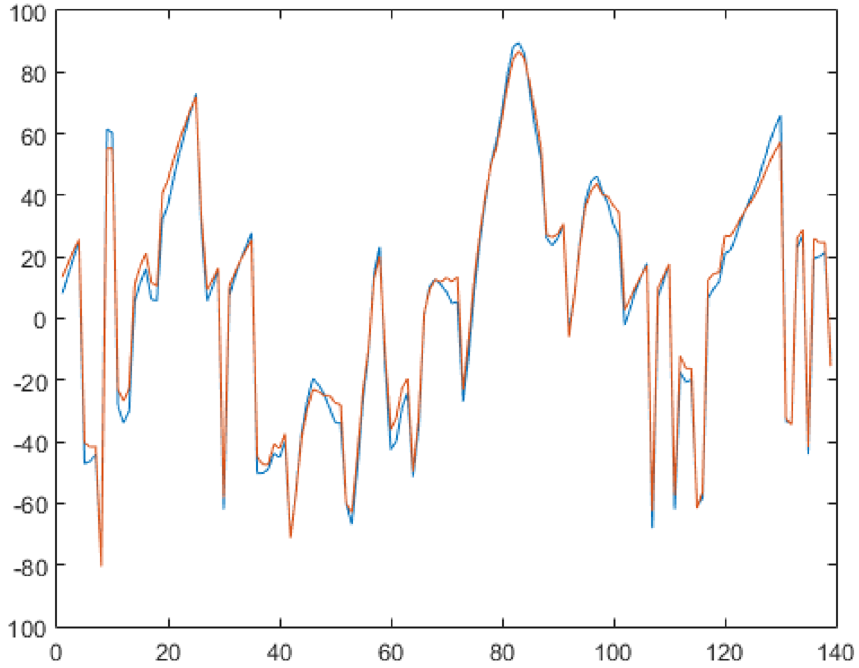
Side Channel Attacks of AES Wireless Keyboard

- **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

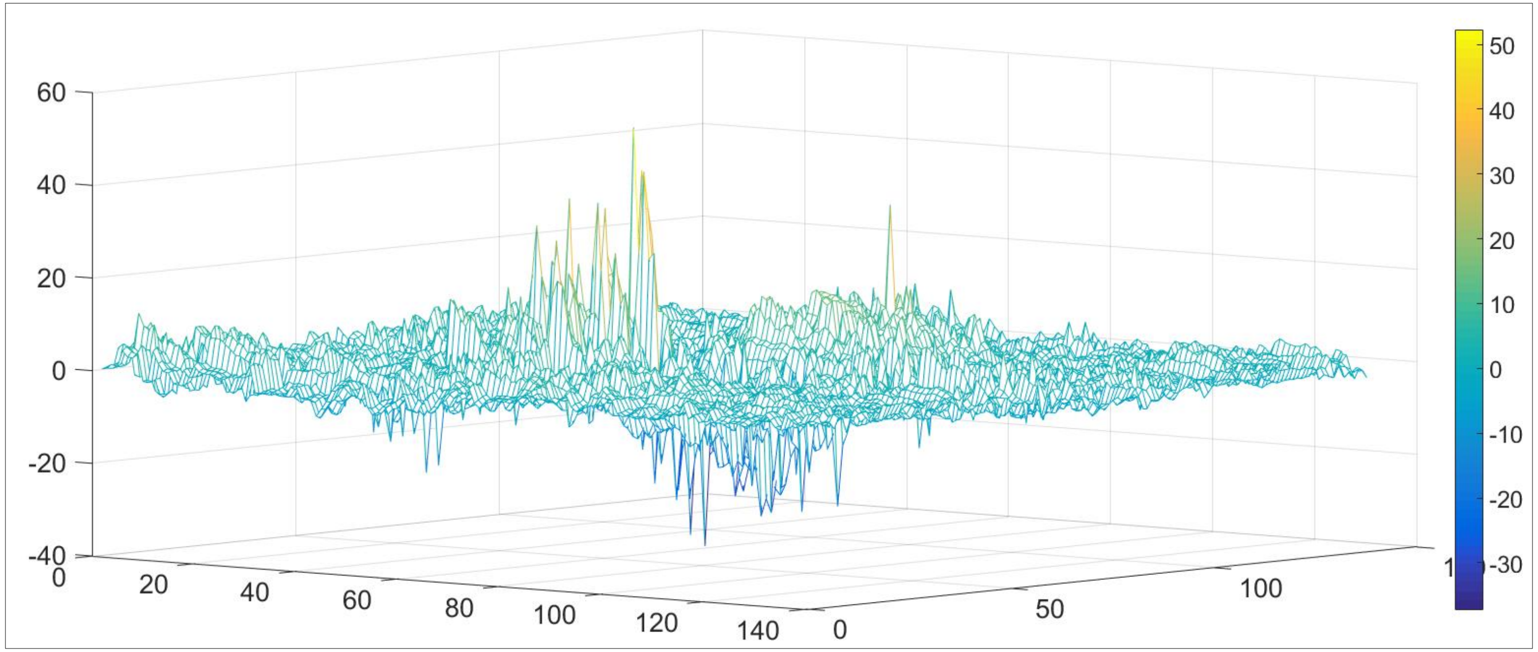


Side Channel Attacks of AES Wireless Keyboard

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



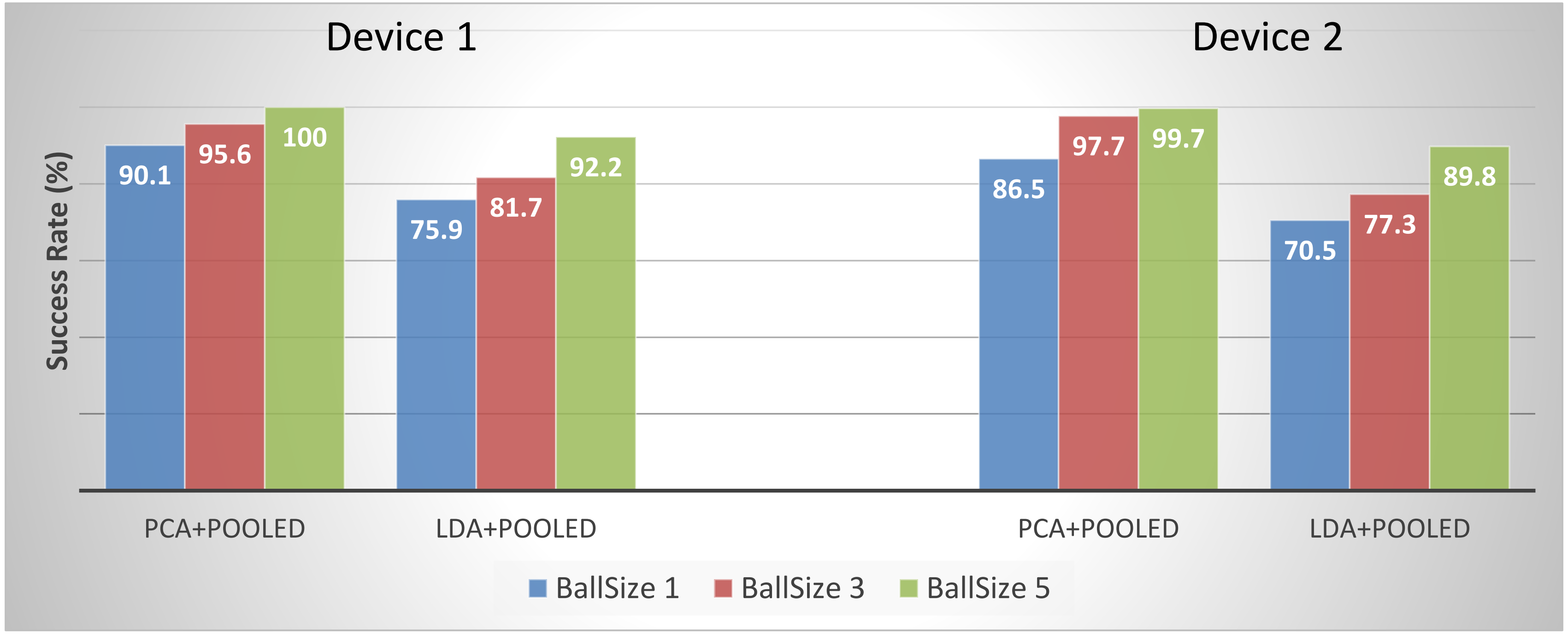
Comparison of mean traces



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. Renauld, 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

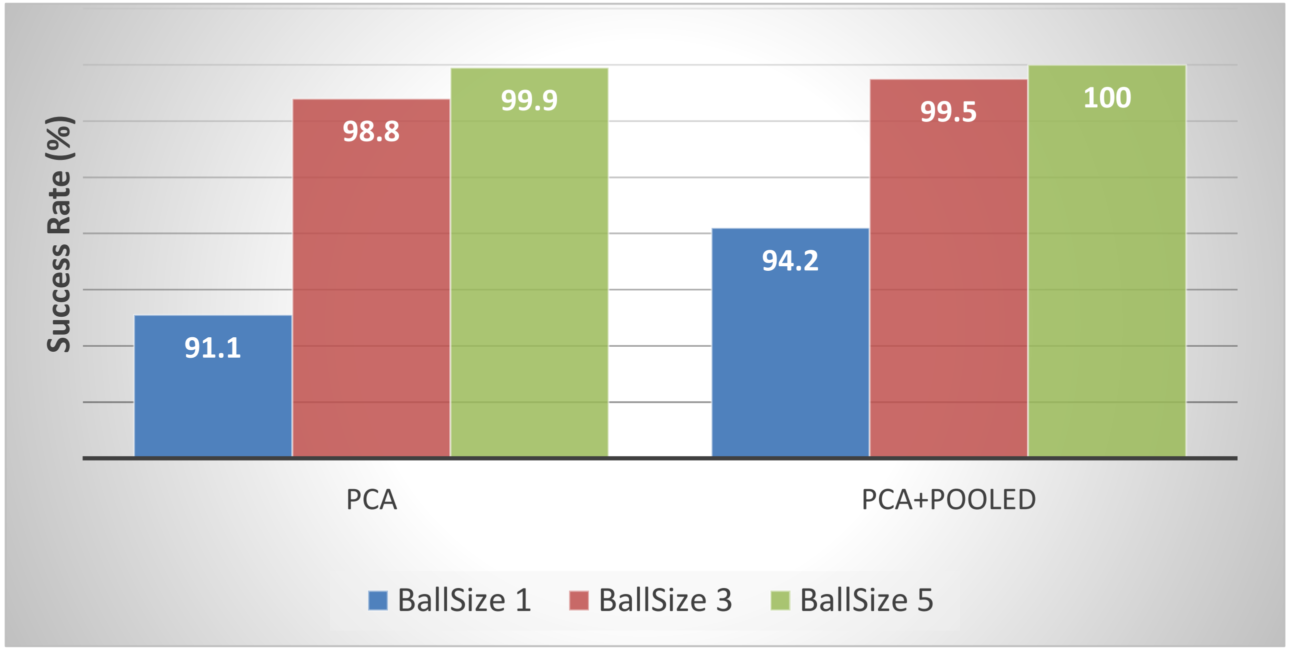
Side Channel Attacks of AES Wireless Keyboard

▪ Proposed Attacks : Case 2 – Difference device

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

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

- 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*

SNTWORKS