I'M A NEWBIE YET I CAN HACK ZIGBEE

Take Unauthorized Control Over ZigBee Devices

LI Jun ,YANG Qing
Unicorn Team – Radio and Hardware Security Research
Qihoo 360 Technology Co. Ltd.

Who we are? Unicorn Team

- Qihoo360's UnicornTeam consists of a group of brilliant security researchers. We focus on the security of anything that uses radio technologies, from small things like RFID, NFC and WSN to big things like GPS, UAV, Smart Cars, Telecom and SATCOM.
- Our primary mission is to guarantee that Qihoo360 is not vulnerable to any wireless attack. In other words, Qihoo360 protects its users and we protect Qihoo360.
- During our research, we create and produce various devices and systems, for both attack and defense purposes.

LI Jun

Twitter: bravo_fighter
Weibo: GoRushing

- Hardware security intern in Unicorn Team of Qihoo360 ,China.
- Second year graduate student at Chengdu University of Information Technology, China.He received his bachelor's degree from University of Electronic Science and Technology of China
- Interested in the security of the internet of things and the security of automobile electronics

360UNICORNTEAM

YANG Qing

Weibo: Ir0nSmith

- YANG Qing is the team leader of Unicorn Team.
- He has rich experiences in wireless and hardware security area, including WiFi penetration testing, cellular network interception, IC card cracking etc. His interests also cover embedded system hacking, firmware reversing, automotive security, and software radio.
- He is the first one who reported the vulnerabilities of WiFi system and RF IC card system used in Beijing subway.

Why is this talk relevant to you ?

- Cause hackers might be able to control your ZigBee enabled appliances without authorization, this talk will teach you how to prevent it .

What will you learn from the talk ?

You will learn, step by step ,how to hack ZigBee enabled devices ,and you will also learn some techniques to protect your ZigBee appliance from being hacked.

So what is ZigBee ?

“ZigBee is the only open, global wireless standard to provide the foundation for the Internet of Things by enabling simple and smart objects to work together, improving comfort and efficiency in everyday life”

So what is ZigBee ?

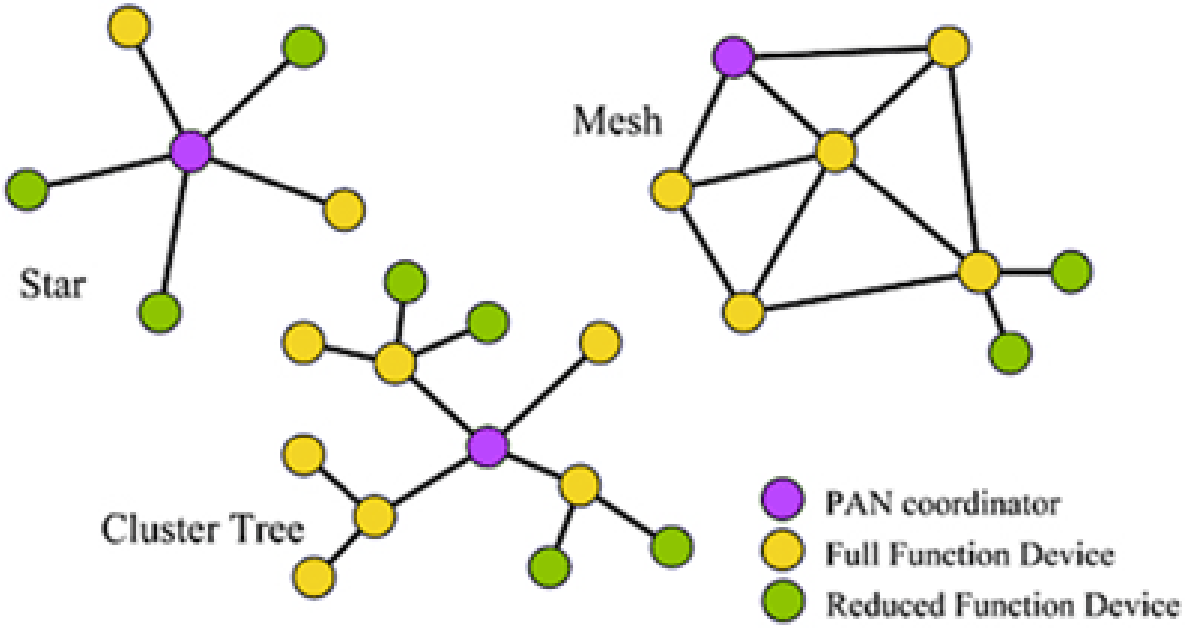
“ZigBee is the wireless language that everyday devices use to connect to one another. In fact, ZigBee could be at work in your home right now”

So what is ZigBee ?

- Technological Standard Created for Control and Sensor Networks
- Based on the IEEE 802.15.4 Standard
- Low-power, low data rate wireless protocol
- Widely used in the Internet of Things
- Widely adapted in applications that require low power consumption
- Flexible network topology

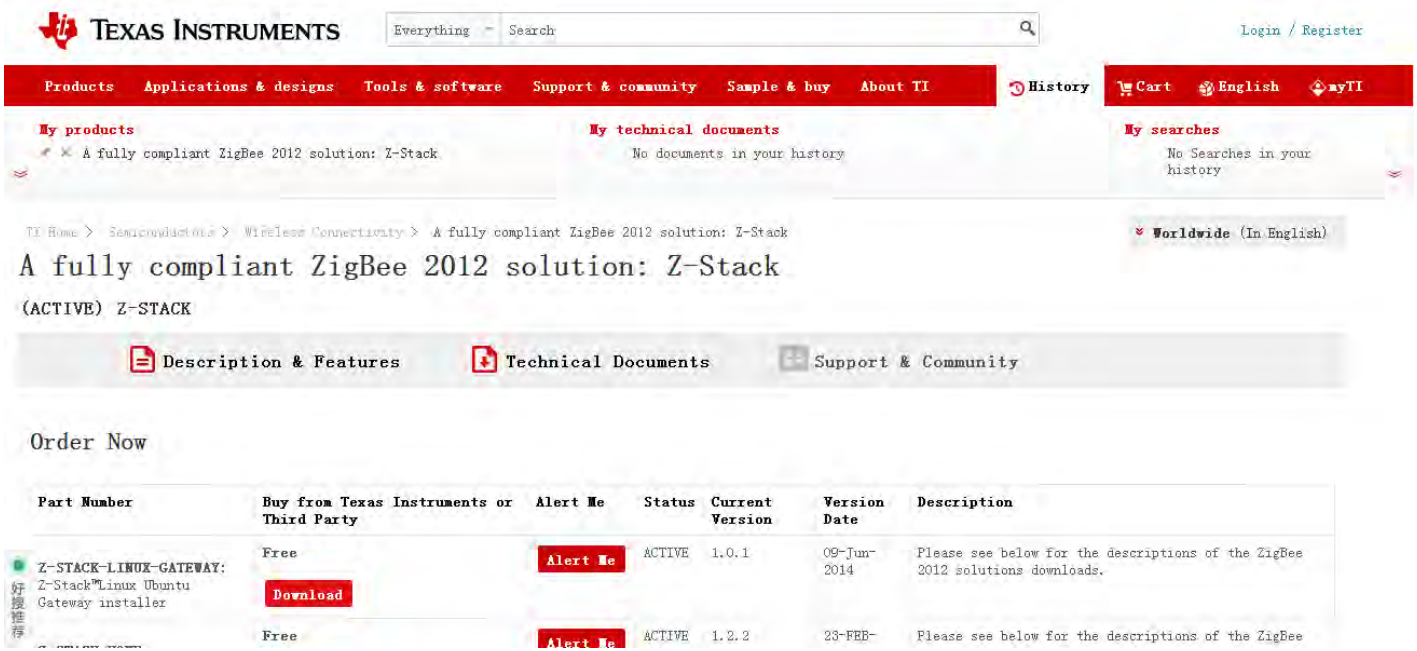
So what is ZigBee ?

ZigBee network topology



And then what is Zstack?

An specific implementation of ZigBee Stack from Texas Instruments based on its CC2530 (which is IEEE80.15.4 enabled)chip, in other words , ZigBee standard is written in plaintext while Zstack is written in code.



Security in ZigBee

ZigBee security is based on symmetric keys and both originator and recipient of a protected transaction need to share the same key.

Key distribution schemes

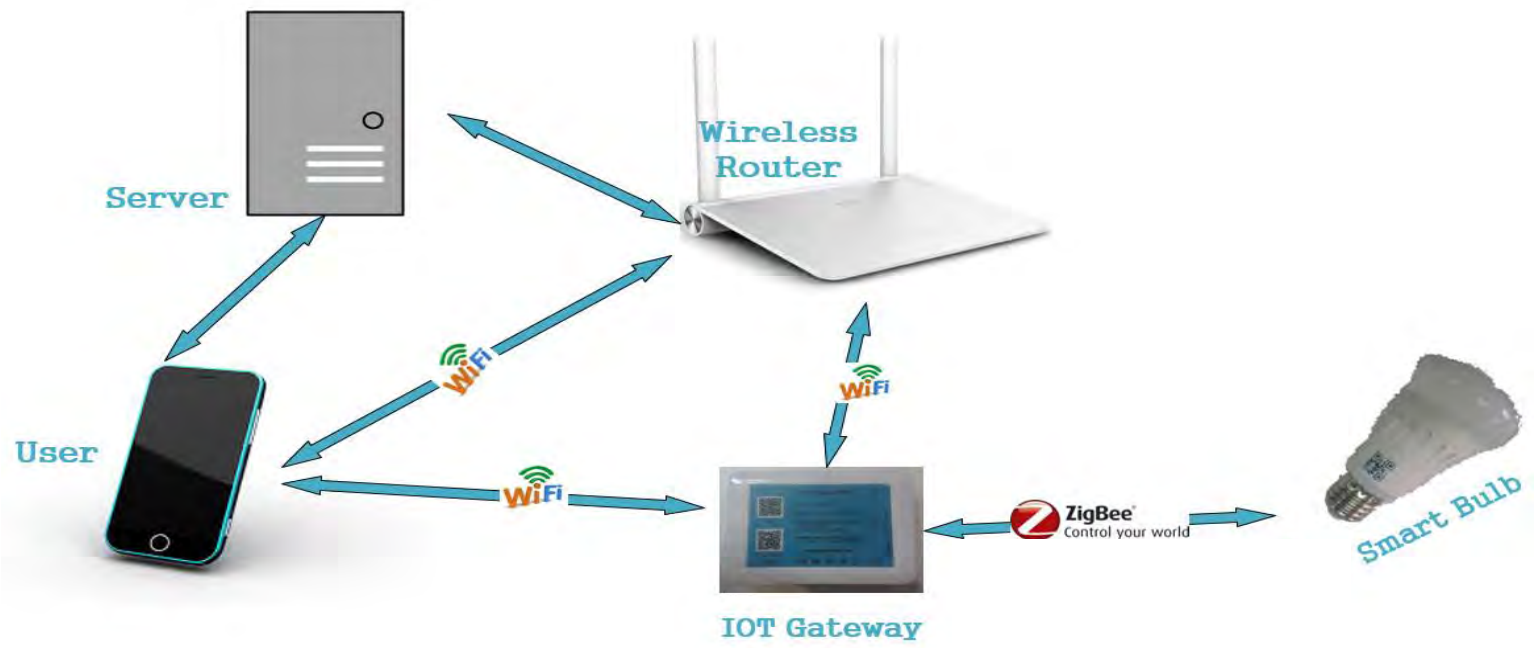
- Pre-installation
- Transport
- Establishment

Three key types

- Master key
- Link key
- Network key

Hacking ZigBee device step by step

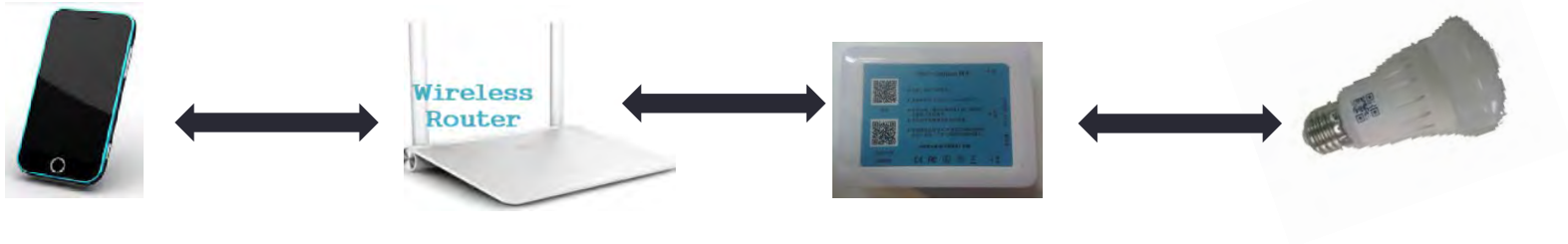
Following is a schematic diagram of a smart bulb system:

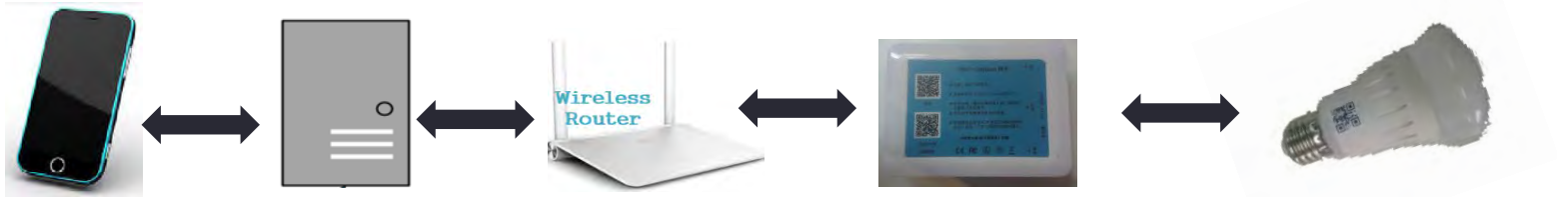


360UNICORNTEAM

Hacking ZigBee device step by step

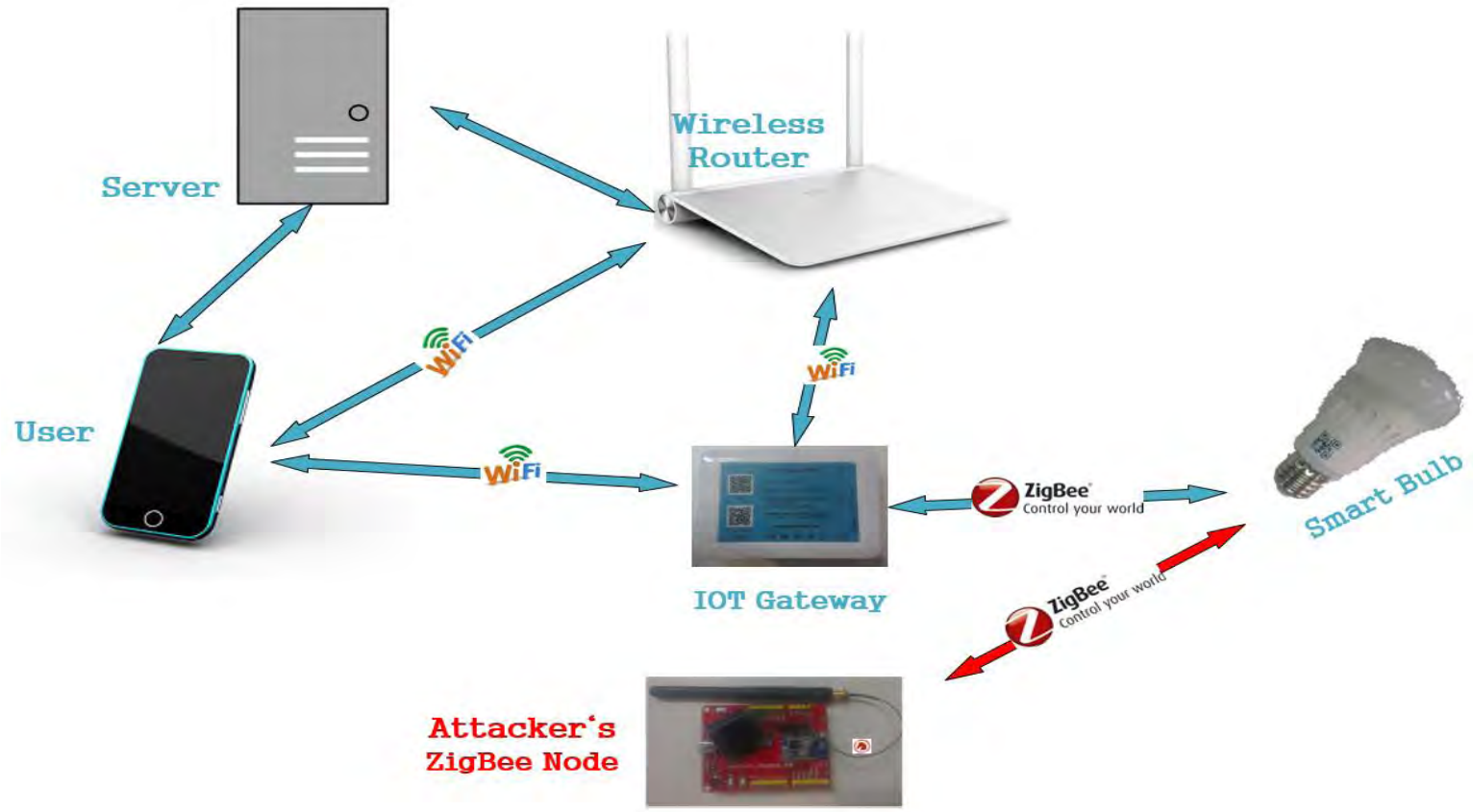
3 Normal control flows:





Hacking ZigBee device step by step

What we want is to directly control the bulb via our own ZigBee node:



Find the encryption key from firmware

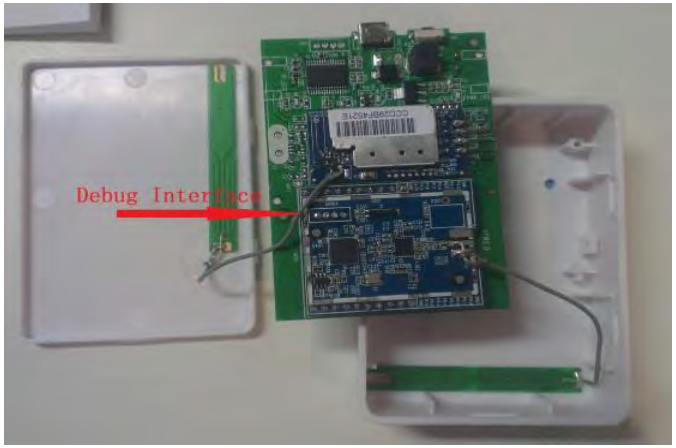
The keys are stored in every node in the network ,as the blub is harder to disassemble so we chose to extract the keys from the gateway.

Zigbee Bulb

Gateway

Find the encryption key from firmware

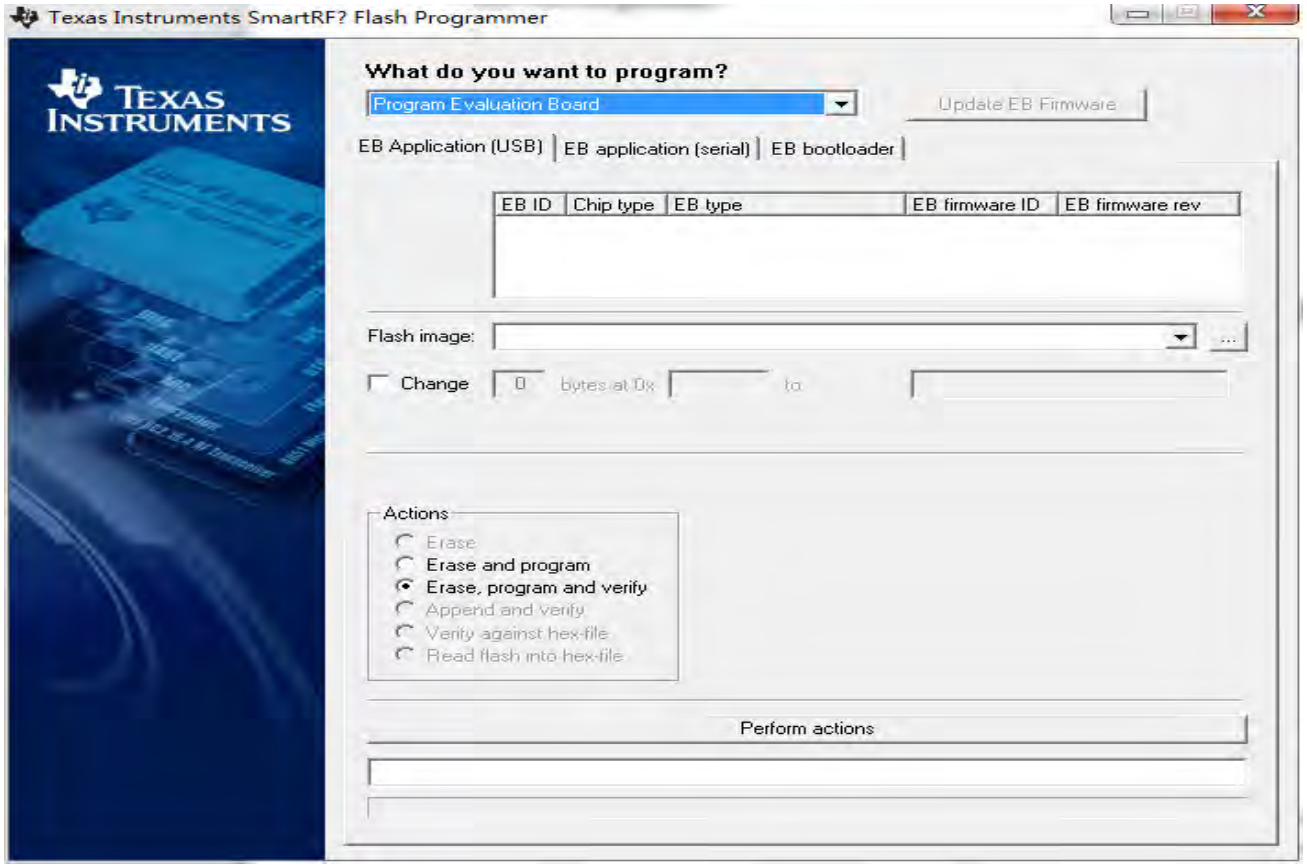
As the red arrow indicates ,the debug interface is right there，we solder on a few wires，connect it to a debugger ，and used TI's SmartRF Flash Programmer to dump the firmware.



Gateway being praised open

Debugger used to extract firmware

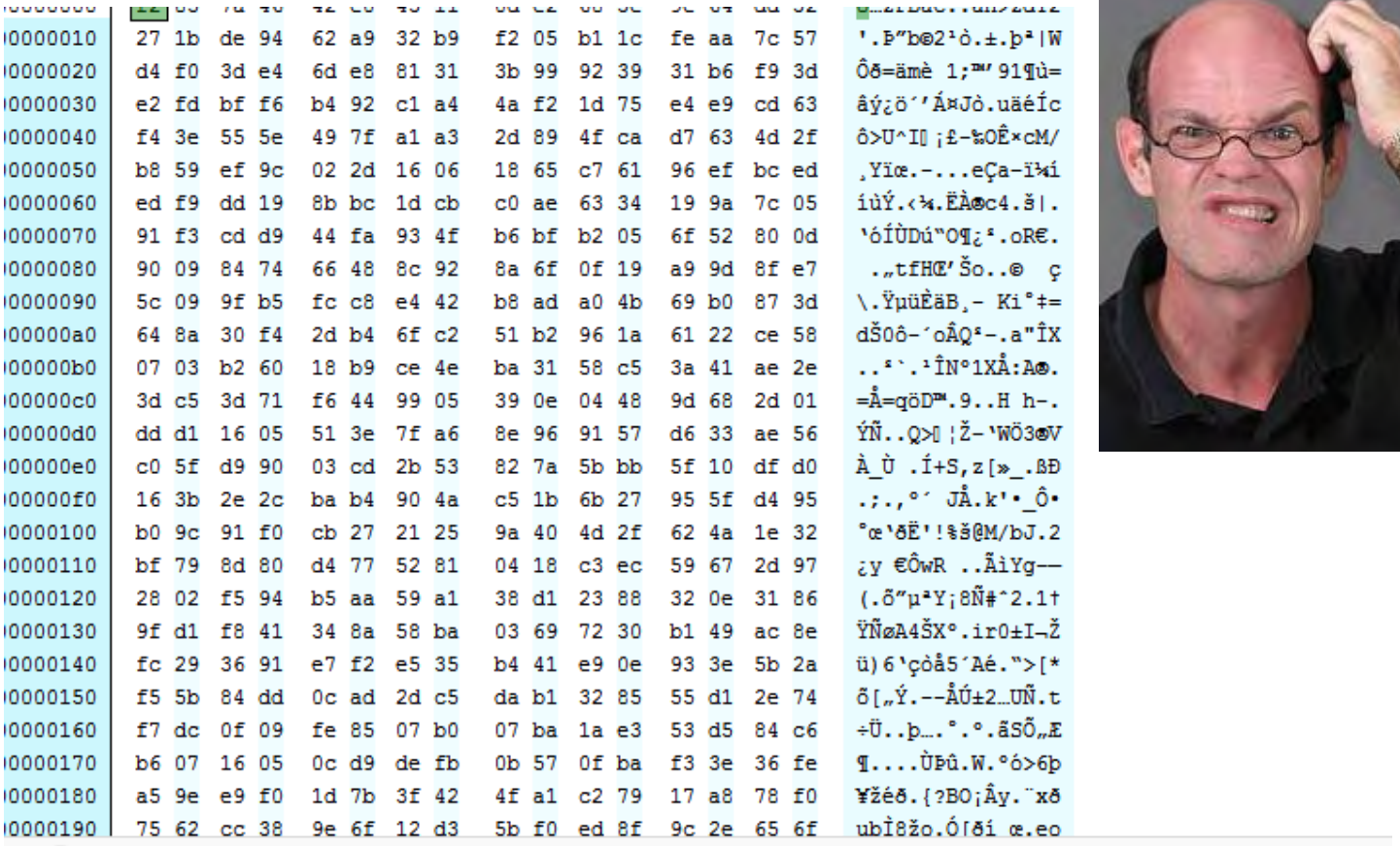
Find the encryption key from firmware



Screenshot of TI's SmartRF Flash Programmer

Find the encryption key from firmware

Now we got THE firmware, what is next ?



```
00000010  27 1b de 94  62 a9 32 b9  f2 05 b1 1c  fe aa 7c 57  '.Þ”b©2¹ò.±.þª|W
00000020  d4 f0 3d e4  6d e8 81 31  3b 99 92 39  31 b6 f9 3d  Ôð=ämè 1;™’91¶ù=
00000030  e2 fd bf f6  b4 92 c1 a4  4a f2 1d 75  e4 e9 cd 63  âý¿ö´’Á¤Jò.uäéÍc
00000040  f4 3e 55 5e  49 7f a1 a3  2d 89 4f ca  d7 63 4d 2f  ô>U^I¡£-‰OÊ×cM/
00000050  b8 59 ef 9c  02 2d 16 06  18 65 c7 61  96 ef bc ed  ¸Yïœ.-...eÇa-ï¼í
00000060  ed f9 dd 19  8b bc 1d cb  c0 ae 63 34  19 9a 7c 05  íùÝ.‹¼.ËÀ®c4.š|.
00000070  91 f3 cd d9  44 fa 93 4f  b6 bf b2 05  6f 52 80 0d  ‘óÍÙDú“O¶¿².oR€.
00000080  90 09 84 74  66 48 8c 92  8a 6f 0f 19  a9 9d 8f e7  .„tfHŒ’Šo..© ç
00000090  5c 09 9f b5  fc c8 e4 42  b8 ad a0 4b  69 b0 87 3d  \.Ÿµüȁ¸- Ki°‡=
000000a0  64 8a 30 f4  2d b4 6f c2  51 b2 96 1a  61 22 ce 58  dŠ0ô-´oÂQ²-.a"ÎX
000000b0  07 03 b2 60  18 b9 ce 4e  ba 31 58 c5  3a 41 ae 2e  ..²`.¹ÎN°1XÅ:A®.
000000c0  3d c5 3d 71  f6 44 99 05  39 0e 04 48  9d 68 2d 01  =Å=qöD™.9..H h-.
000000d0  dd d1 16 05  51 3e 7f a6  8e 96 91 57  d6 33 ae 56  ÝÑ..Q>¦Ž-‘WÖ3®V
000000e0  c0 5f d9 90  03 cd 2b 53  82 7a 5b bb  5f 10 df d0  À_Ù .Í+S‚z[»_.ßÐ
000000f0  16 3b 2e 2c  ba b4 90 4a  c5 1b 6b 27  95 5f d4 95  .;.,°´ JÅ.k'•_Ô•
00000100  b0 9c 91 f0  cb 27 21 25  9a 40 4d 2f  62 4a 1e 32  °œ‘ðË'!%š@M/bJ.2
00000110  bf 79 8d 80  d4 77 52 81  04 18 c3 ec  59 67 2d 97  ¿y €ÔwR ..ÃìYg-—
00000120  28 02 f5 94  b5 aa 59 a1  38 d1 23 88  32 0e 31 86  (.õ”µªY¡8Ñ#ˆ2.1†
00000130  9f d1 f8 41  34 8a 58 ba  03 69 72 30  b1 49 ac 8e  ŸÑøA4ŠX°.ir0±I¬Ž
00000140  fc 29 36 91  e7 f2 e5 35  b4 41 e9 0e  93 3e 5b 2a  ü)6‘çòå5´Aé.“>[*
00000150  f5 5b 84 dd  0c ad 2d c5  da b1 32 85  55 d1 2e 74  õ[„Ý.-ÅÚ±2…UÑ.t
00000160  f7 dc 0f 09  fe 85 07 b0  07 ba 1a e3  53 d5 84 c6  ÷Ü..þ….°.°.ãSÕ„Æ
00000170  b6 07 16 05  0c d9 de fb  0b 57 0f ba  f3 3e 36 fe  ¶....ÙÞû.W.°ó>6þ
00000180  a5 9e e9 f0  1d 7b 3f 42  4f a1 c2 79  17 a8 78 f0  ¥žéð.{?BO¡Ây.¨xð
00000190  75 62 cc 38  9e 6f 12 d3  5b f0 ed 8f  9c 2e 65 6f  ubÌ8žo.Ó[ðí œ.eo
```

Let's do "Firmware Diving"

Searching through firmware for keys

Find the encryption key from firmware

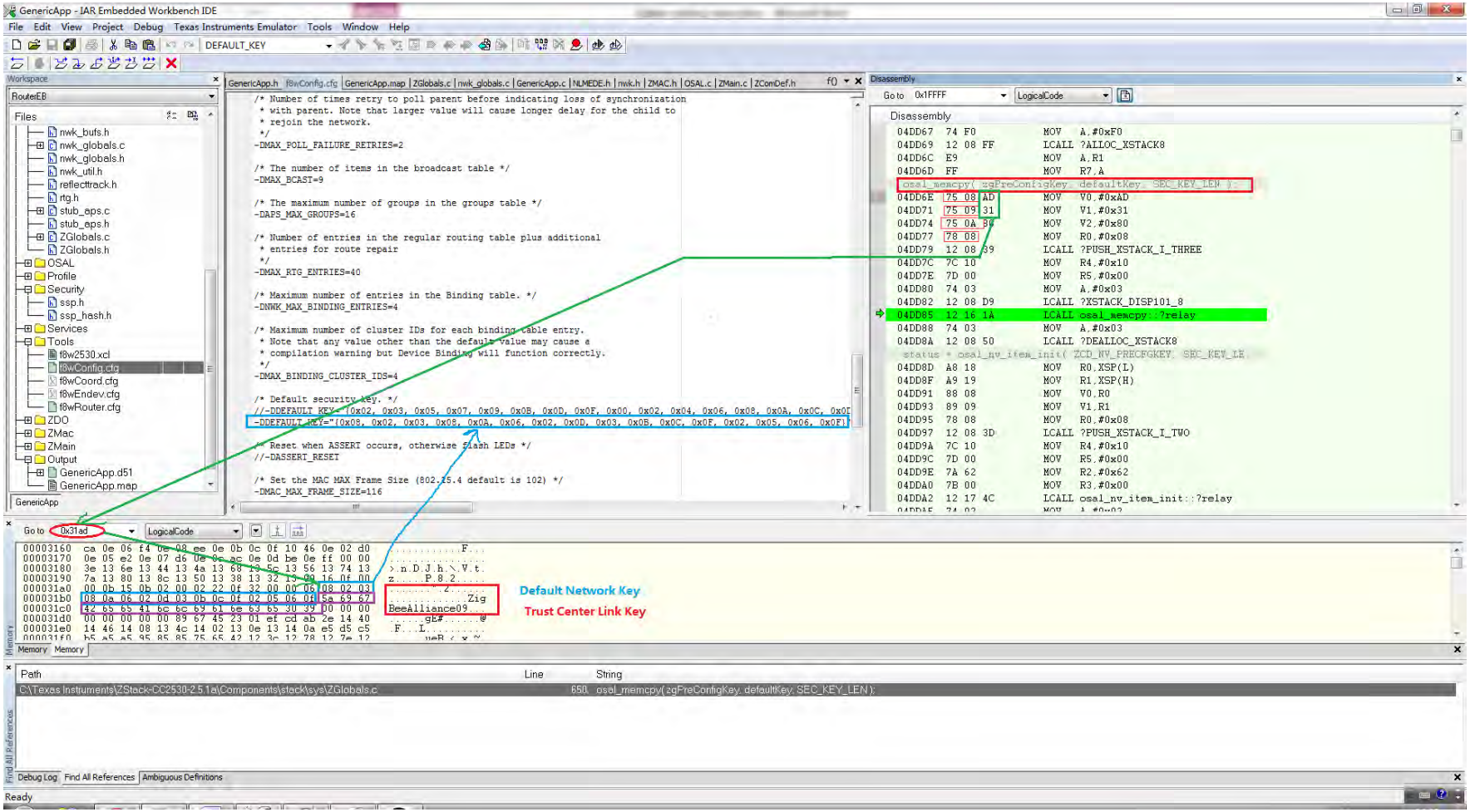
First set the keys to have distinct signature，then find it in the firmware and see if we could discover something interesting .

As the key is used to encrypt the packets ,why not try to find the instructions that manipulate the keys ?

Bingo ! We found that the instructions used to manipulate the keys have relatively fixed patern (shown in the next slide) and the four consecutive move instructions could be used as a filter (or signature) for the address of the keys

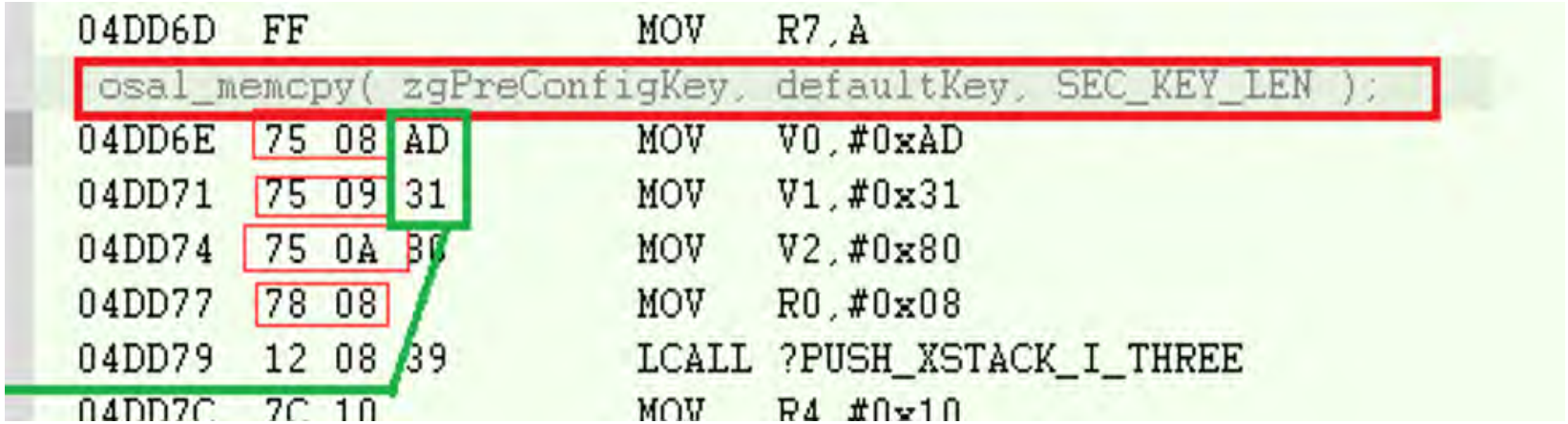
360UNICORNTEAM

Find the encryption key from firmware



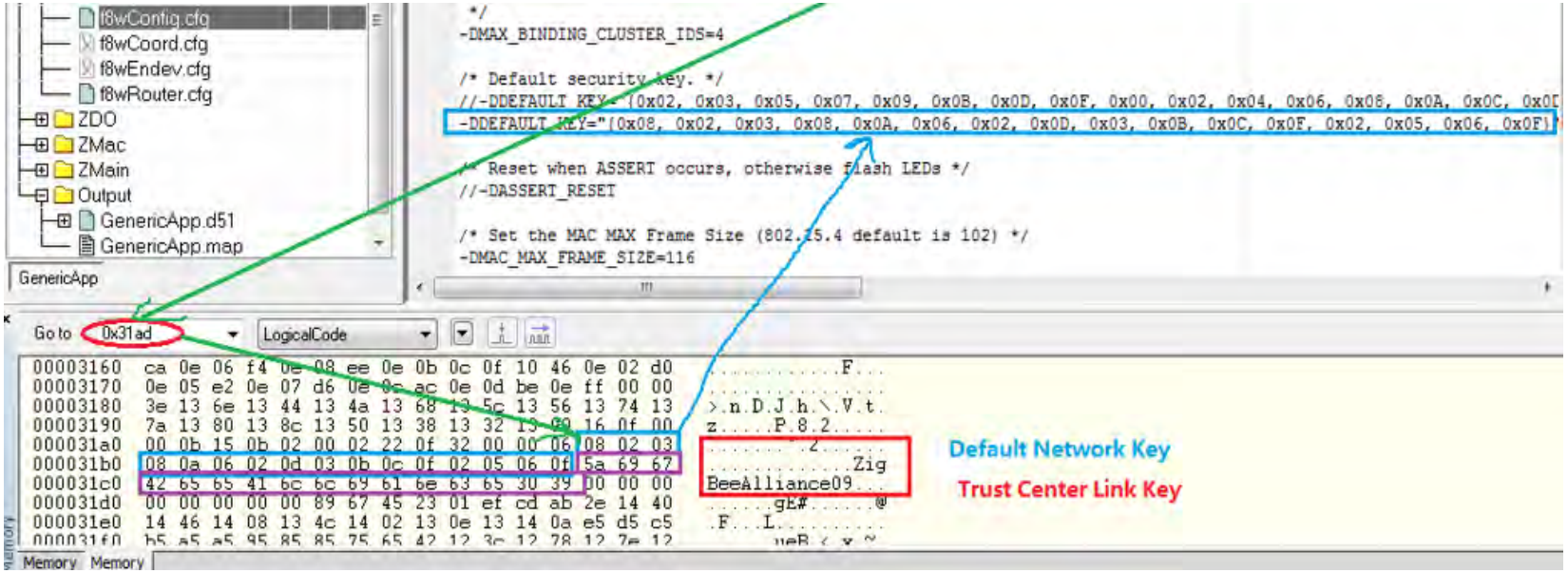
Find the encryption key from firmware

On the upper right corner is the instructions that manipulate the network key. The 0x31，0xAD is the memory address that stores the key（shown on the lower left corner）



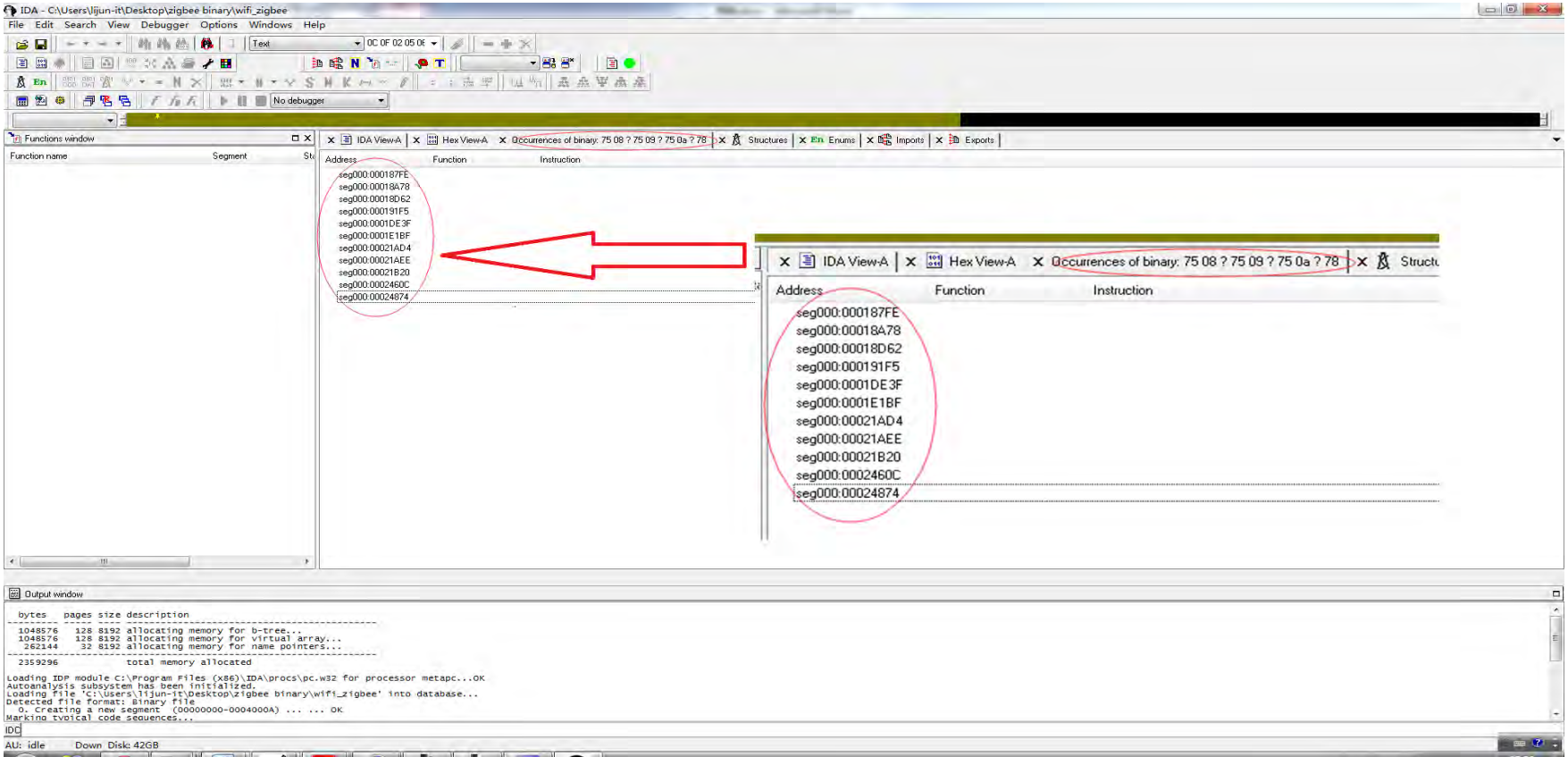
Find the encryption key from firmware

On the upper right corner is the instructions that manipulate the network key. The 0x31，0xAD is the memory address that stores the key（shown on the lower left corner）



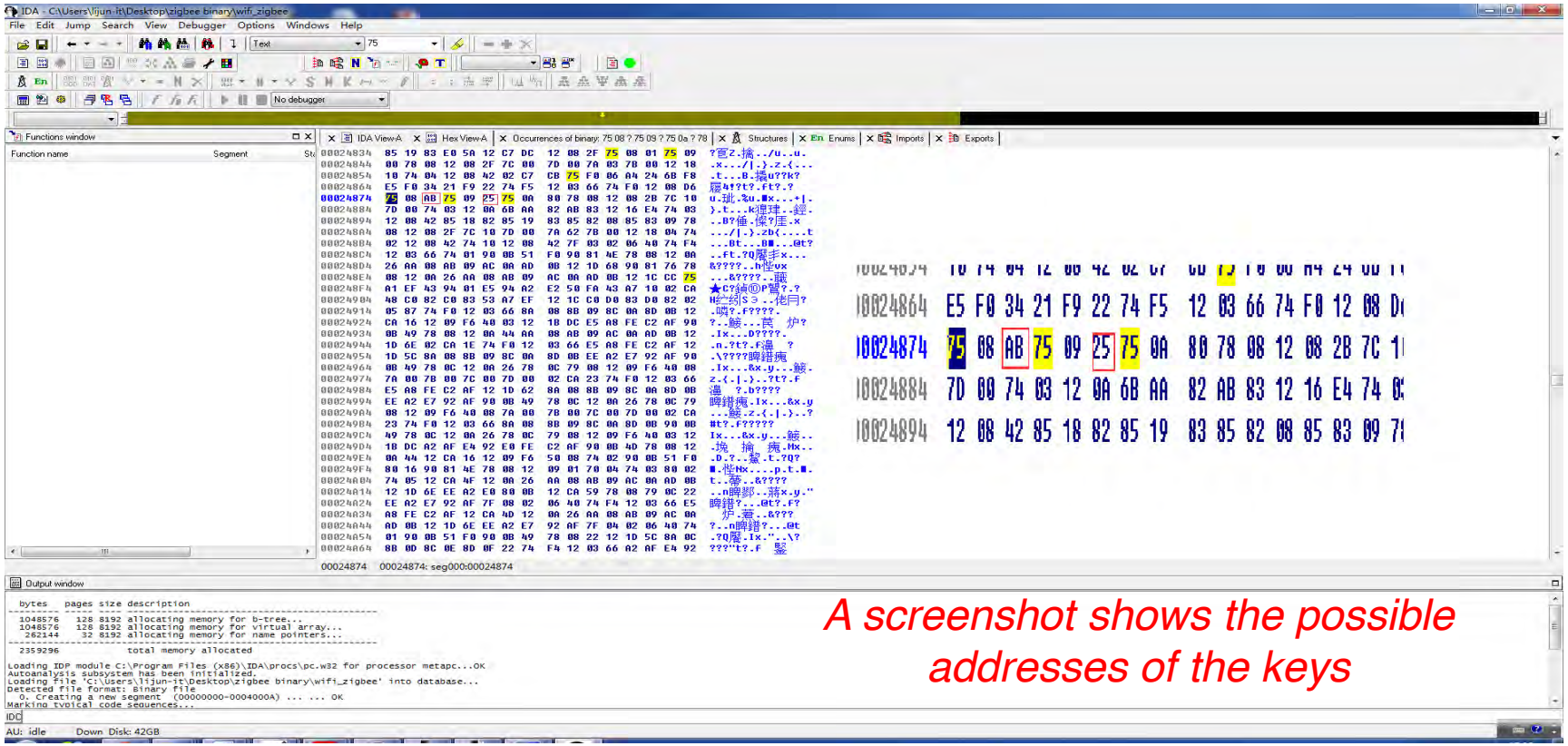
Find the encryption key from firmware

Then we use the four consecutive move instructions' corresponding machine code and operand（75 08 ? 75 09 ? 75 0A） as a filter to search through the firmware for the address of the keys.



Find the encryption key from firmware

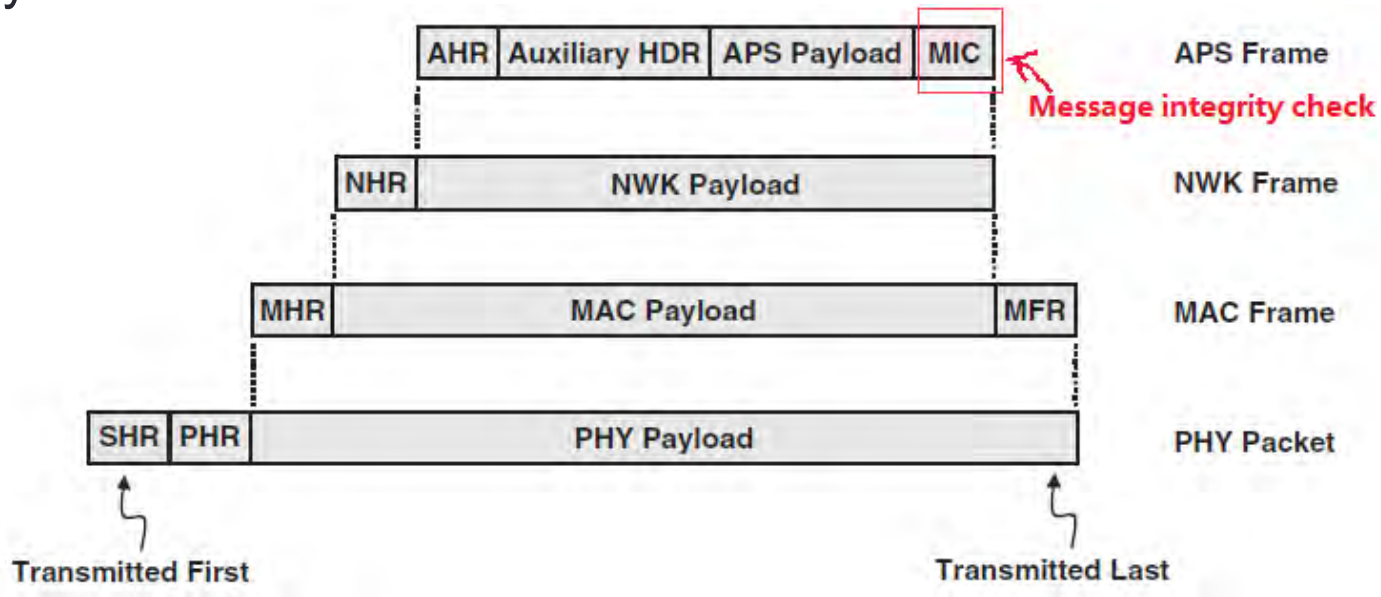
Then we use the four consecutive move instructions' corresponding machine code and operand（75 08 ? 75 09 ? 75 0A） as a filter to search through the firmware for the address of the keys.



A screenshot shows the possible addresses of the keys

Verify The Keys

In order to verify the keys， we utilized the MIC（message integrity check）contained in the packet， and if the deciphered packet can pass the MIC， we can assert that we find the right keys.



ZigBee Packet structure

Verify The Keys

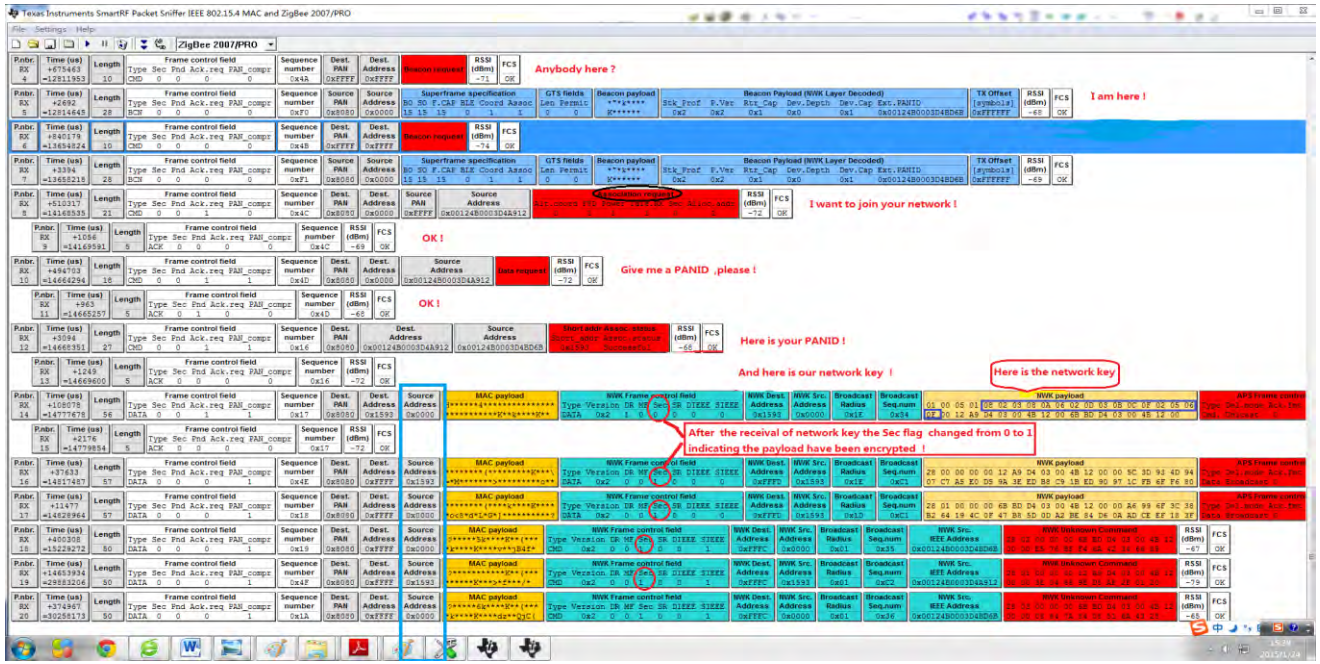
In order to verify the keys， we utilized the MIC（message integrity check）contained in the packet， and if the deciphered packet can pass the MIC， we can assert that we find the right keys.

A sniffer used to capture the packets

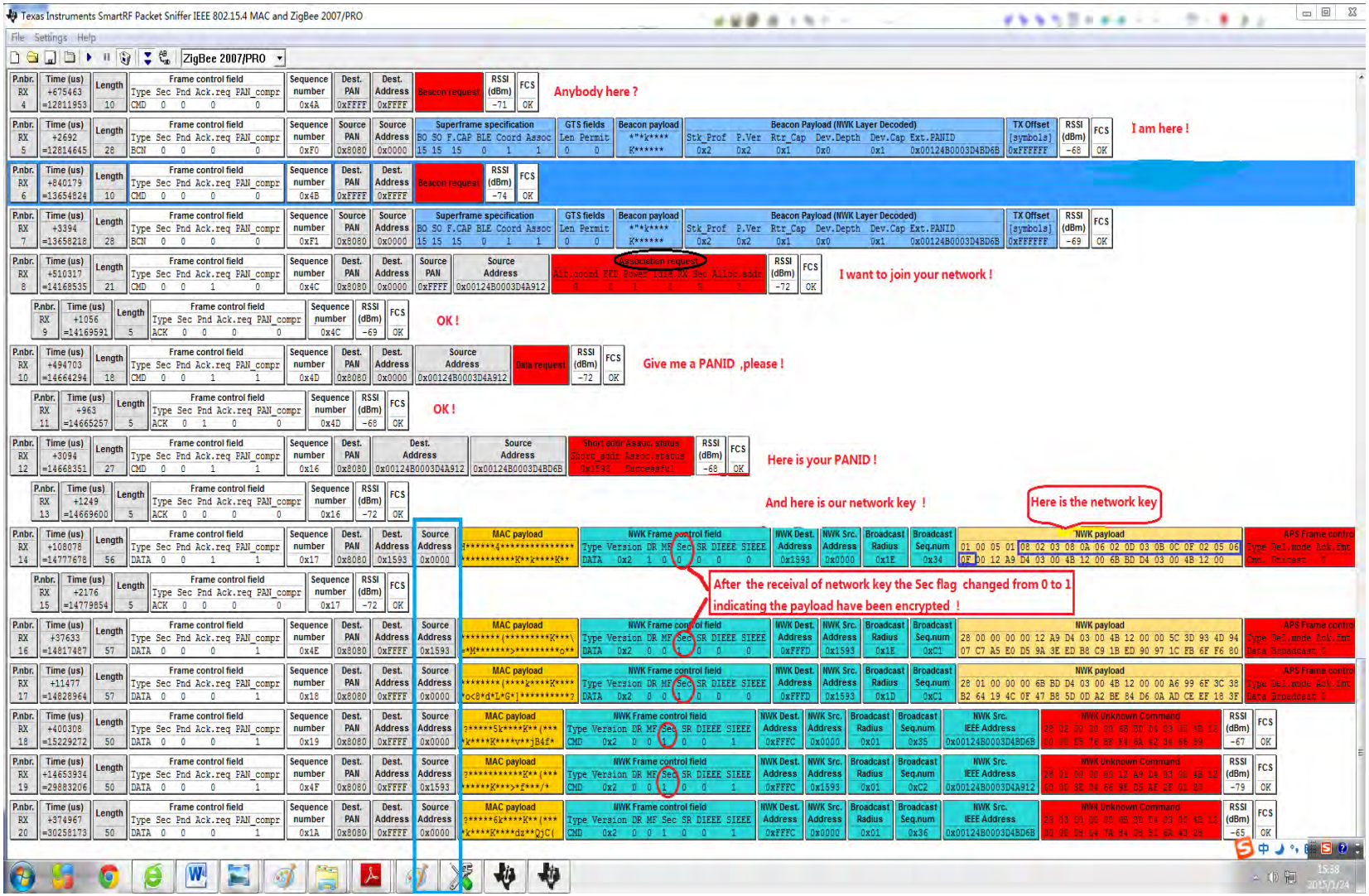
360UNICORNTEAM

Find the encryption key by sniffing

The following screenshot shows the process of a new node joining the network, and the figure is quite self-explanatory . The network key is sent from the coordinator to the joining device in plaintext， and after receiving the network key the communication is immediately encrypted.



Find the encryption key by sniffing



“Utilize” the keys found

I wouldn’t say that after we found the key we could do some data mining to find the users habit etc，cause that would be a little bit farfetched，but the following are some very practical attacks we can perform:

- Analysis of the deciphered data
- Replay& Spoof
- Intercept
- Disassociation attack

360UNICORNTEAM

Analysis of the deciphered data

After we deciphered the data ， in order to take control over the target device ,we have to analyze the application level data and the results are as following：

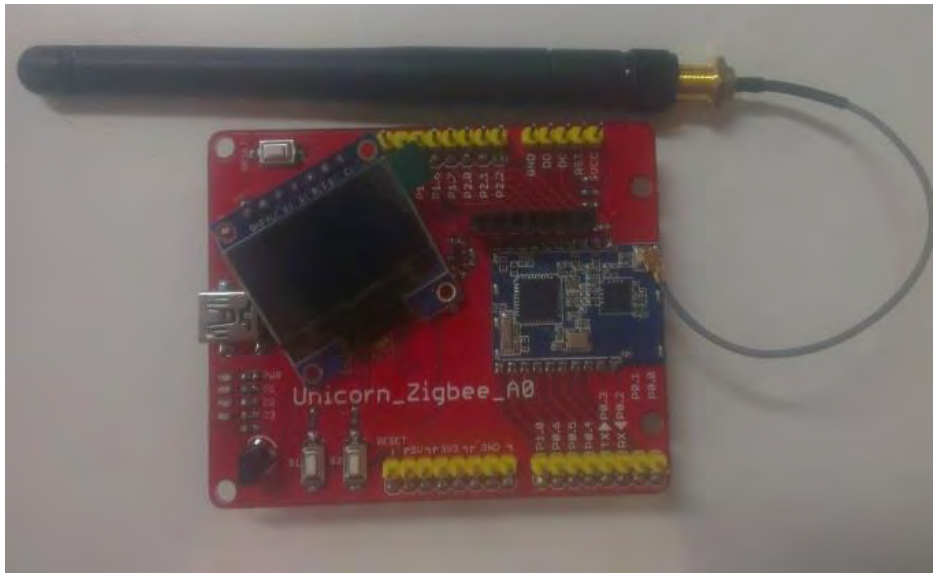
Byte0	0x04
Byte1	Target PANID
Byte2	
Byte3	Unknown
Byte4	Mode
Byte5	Red
Byte6	Green
Byte7	Blue
Byte8	Illuminance
Byte9	Checksum

Analysis of the deciphered data

The payload is 10 byte in length，with the last byte being the xor checksum of the foregoing bytes，the byte1 and byte2 is the PANID of the target device（the bulb in our case）. Now we can control the bulb with our own node .

Byte0	0x04
Byte1	Target PANID
Byte2	
Byte3	Unknown
Byte4	Mode
Byte5	Red
Byte6	Green
Byte7	Blue
Byte8	Illuminance
Byte9	Checksum

Take control

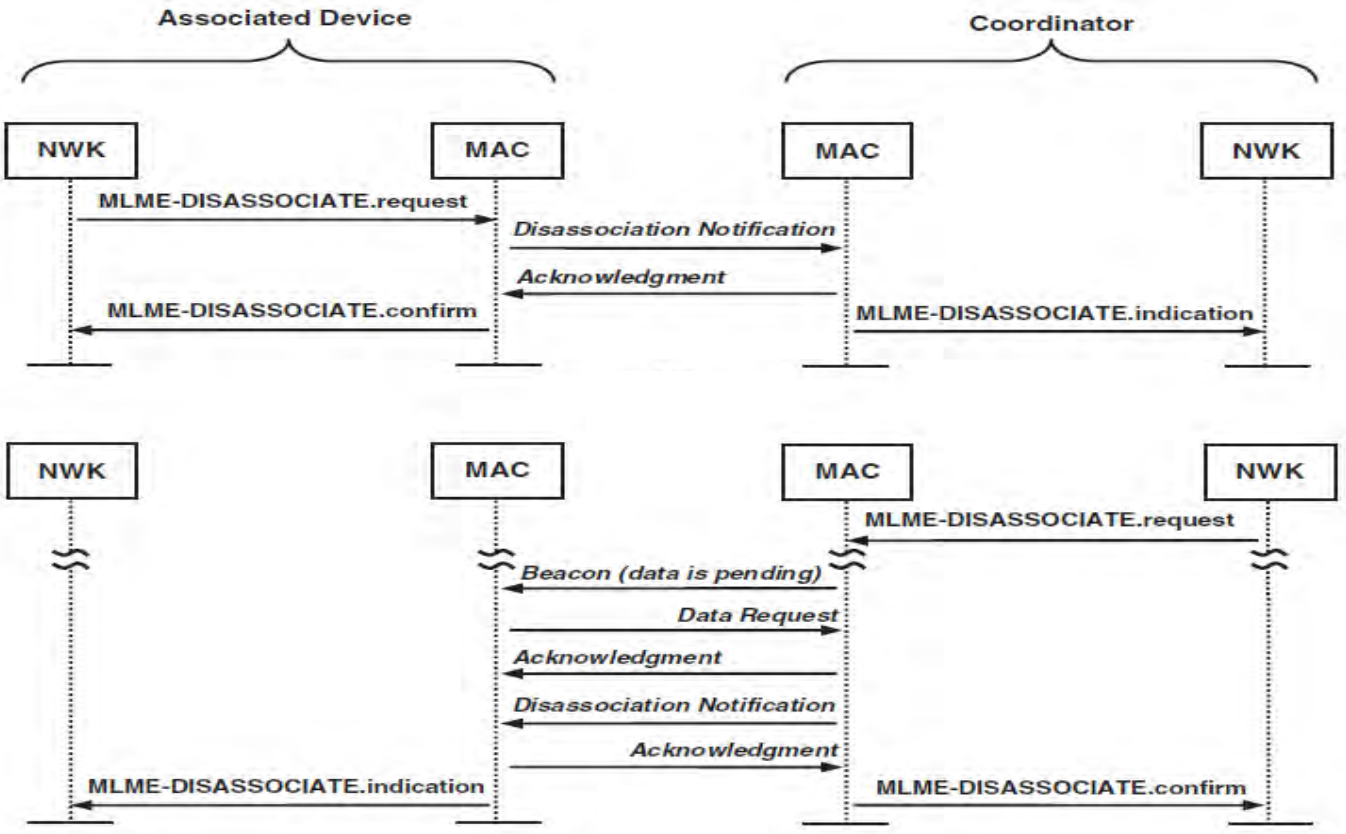


A Zigbee node we made and we used it to control the bulb

Replay& Spoof

360UNICORNTEAM

Disassociation attack



Disassociation Sequence

Hacking for Protection

Only talking about attacks and leaving protection aside goes against the ethos of Defcon and the whole hacker community I guess， so here come tips for prevention or , at least , mitigation:

- Store hash of the encryption key instead of plaintext.
- Don’t use OTA (over the air) key provisioning scheme, use preinstall or key negotiation instead.
- Blow the fuse to prevent the firmware from being dumped.
- Employ some light weight encryption on the application data to make the analysis of application data harder after key compromise.

References

- *Below are references related to the topics discussed .*
- *1)The ZigBee Alliance homepage:*
- http://www.ZigBee.org
- *2) The KillerBee framework :*
- https://github.com/riverloopsec/killerbee
- *3) Paperback book entitled "ZigBee Wireless Networks and Transceivers"*
- *2008, Elsevier Ltd . ISBN: 978-0-7506-8393-7*
- *4) Paper "Recommended Practices Guide For Securing ZigBee Wireless Networks in Process Control System Environments"*
- *April 2007.Author:Ken Masica. Lawrence Livermore National Laboratory*
- *5) Paper "ZigBee Security" ©2009 ZigBee Alliance.*
- *Author:Robert Cragie.Chair, ZigBee Alliance ZARC Security Task Group.Principal Engineer, Jennic Ltd.*
- *6) A webpage :* http://www.ciscopress.com/articles/article.asp?p=1823368&seqNum=4

Acknowledgment

ZHANG Kai

- Twitter: peekair_zhang

Former reversing engineer at Qihoo360.(protocol,fireware,binary format & crack).
He loves cats.

Nikita

- Twitter: Niki7a

Thank you!