A SMART LIGHT HACKING JOURNEY

ABOUT US

KHALED NASSAR

- Security Engineer
- Avid CTF player

notkmhn

@notkmhn

TOM CLEMENT

- Organiser
- Sr. embedded software
- Owner

tjclement

@Tom_Clement

A QUICK LOOK AHEAD

INTRODUCTION AND PRIOR WORK

ESP8266 - INITIAL INVESTIGATION

BK7231 - NEW CHIP, NEW VULNS

VULNERABILITY DETAILS AND EXPLOITATION

INSPIRATION

HACK42 FLASHING PARTY

OVER HACK42 | NIEUWS | AGENDA | MEEDOEN | TOUR | STATS | WIKI | CONTACT | SHOP

Home Automation, Domotica en Internet of Things/LSC Smart Connect Slimme Stekker

ACCOUNT AANVRAGEN | AANMELDEN

Zoeken

Doorzoek Hack42

OK | Zoeken

Actualiteiten
- Recente wijzigingen

Verhuizingdingen
- Maken Pand4.1

Activiteiten & projecten
- Activiteiten
- Projecten
- Kalender
- Activiteit Toevoegen
- Project Toevoegen

Faciliteiten
- Howto Hack42
- Het gebouw
- Gereedschap
- Museum

Diversiteiten
- Huisregels
- Deelnemers
- Wiki-nieuwkomers
- Wiki powerusers
- Pers & media
- Sponsoren
- Handige links

pagina | overleg | brontekst bekijken | geschiedenis

Flashen

De action slimme stekker is veel slimmer te maken en minder afhankelijk van tuya connect

De makkelijkste manier is github:tuya-convert op je laptop/rasberry pi/device met wifi te installeren. Volg de handleiding netjes en hark de software bij elkaar als je iets anders dan debian draait om het goed te laten werken.

Na het inpluggen van de stekker een paar seconde de push button ingedrukt houden en je hoort een klikje en de stekker begint te knipperen. Dan is die in flash modus, tik 'y' en/of ENTER om het flashen te starten.

Aan het einde van de flash het curl commando naar flash3 draaien en er staat tasmota op. (duurt ongeveer 17 seconden, heb wat geduld).

Nadat er tasmota op staan kun je met je telefoon / ... inloggen (telefoon is handig!) op tasmota-<nummer> en een wifi netwerk kiezen.

Als je op je DHCP server, via DNS (of met nmap) het device kunt vinden als tasmota-<nummer>.

Hint: meestal werkt http://tasmota-<nummer> wel in je browser.

Kies dan voor "console" en paste de volgende regel:

```
reset 5
```

Wacht even tot de module geherstart is en paste:

INSPIRATION

HACK42 FLASHING PARTY

OVER HACK42 | NIEUWS | AGENDA | MEEDOEN | TOUR | STATS | WIKI | CONTACT | SHOP

Home Automation, Domotica en Internet of Things/LSC Smart Connect Slimme Stekker

DUNT AANVRAGEN | AANMELDEN

pagina | overleg | brontekst bekijken | geschiedenis

Doorzoek Hack42

OK | Zoeken

Actualiteiten
- Recente wijzigingen

Verhuizingdingen
- Maken Pand4.1

Activiteiten & projecten
- Activiteiten
- Projecten
- Kalender
- Activiteit Toevoegen
- Project Toevoegen

Faciliteiten
- Howto Hack42
- Het gebouw
- Gereedschap
- Museum

Diversiteiten
- Huisregels
- Deelnemers
- Wiki-nieuwkomers
- Wiki powerusers
- Pers & media
- Sponsoren
- Handige links

Flashen

De action slimme stekker is veel slimmer te maken en minder afhankelijk van tuya connect

De makkelijkste manier is github:tuya-convert op je laptop/rasberry pi/device met wifi te installeren. Volg de handleiding netjes en hark de software bij elkaar als je iets anders dan debian draait om het goed te laten werken.

Na het inpluggen van de stekker een paar seconde de push button ingedrukt houden en je hoort een klikje en de stekker begint te knipperen. Dan is die in flash modus, tik 'y' en/of ENTER om het flashen te starten.

Aan het einde van de flash het curl commando naar flash3 draaien en er staat tasmota op. (duurt ongeveer 17 seconden, heb wat geduld).

Nadat er tasmota op staan kun je met je telefoon / ... inloggen (telefoon is handig!) op tasmota-<nummer> en een wifi netwerk kiezen.

Als je op je DHCP server, via DNS (of met nmap) het device kunt vinden als tasmota-<nummer>.

Hint: meestal werkt http://tasmota-<nummer> wel in je browser.

Kies dan voor "console" en paste de volgende regel:

```
reset 5
```

Wacht even tot de module geherstart is en paste:

PREVIOUS WORK - TUYA-CONVERT

ct-Open-Source / tuya-convert Public

A collection of scripts to flash Tuya IoT devices to alternative firmwares

MIT license

3.6k stars 425 forks

Star Watch

Code Issues 149 Pull requests 15 Discussions ...

master ...

kueblc Release v2.4.5 ... on 26 Jan 2021 400

View code

README.md

TUYA-CONVERT

Smart Home - Smart Hack

Wie der Weg ins digitale Zuhause zum Spaziergang wird

REFRESHING MEMORIES

Michael Steigerwald

BIG DATA (personenbezogene Daten)

smart home smart hack

Device activating information

Activation status	Yes
Time of activation	2018-11-22 07:38:12
Last device activity	2018-11-27 09:16:04
Last update	2018-11-27 09:16:25
Online now	No
Binding user	35c3@vtrust.de
Binding APP	涂鸦智能
Latitude and longitude	51.397840, 12.405506
Geographic position	Leipzig
Channel	
Time zone	Europe/Berlin GMT+01:00

00:00 | 51:21 1.00x

Playlists: '35c3' videos starting here / audio / related events

51 min 2018-12-28 2018-12-29 40610 Fahrplan

PREVIOUS WORK - TUYA-CONVERT

WHAT WAS THE ISSUE?

TLS-PSK CONNECTION

Smart device

Start TLS Connection →

← Server Hello + KX - PSK params

Client KX - PSK ID xxxx →

← Start session - Encrypted data

Vendor server

PREVIOUS WORK - TUYA-CONVERT

WHAT WAS THE ISSUE?

TLS-PSK CONNECTION

Smart device

Start TLS Connection →

← Server Hello + KX - PSK params

Client KX - PSK ID xxxx →

← Start session - Encrypted data

Vendor server

PREVIOUS WORK - TUYA-CONVERT

WHAT WAS THE ISSUE?

TLS-PSK CONNECTION

Smart device

Start TLS Connection →

← Server Hello + KX - PSK params

Client KX - PSK ID xxxx →

```
PSK = generate_psk(PSK_ID)
```

Vendor server

PREVIOUS WORK - TUYA-CONVERT

WHAT WAS THE ISSUE?

TLS-PSK CONNECTION

Smart device

Start TLS Connection →

← Server Hello + KX - PSK params

Client KX - PSK ID xxxx →

```
PSK = generate_psk(PSK_ID)
```

Vendor server

PREVIOUS WORK - TUYA-CONVERT

WHAT WAS THE ISSUE?

TLS-PSK CONNECTION

Smart device

Start TLS Connection →

← Server Hello + KX - PSK params

Client KX - PSK ID xxxx →

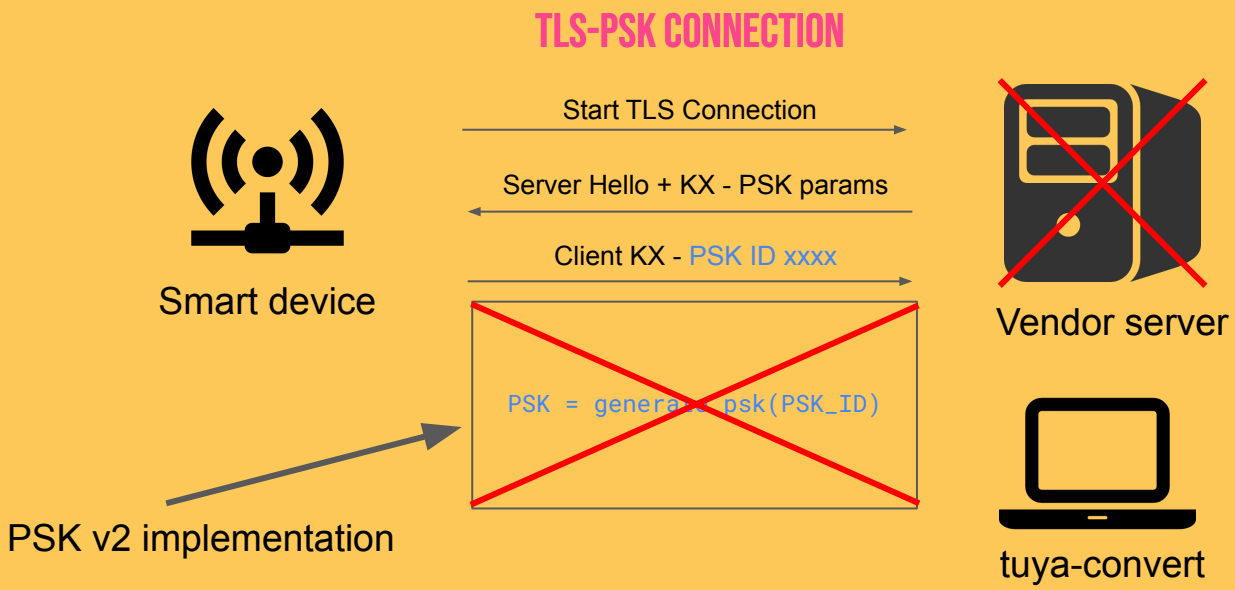
```
PSK = generate_psk(PSK_ID)
```

Vendor server

tuya-convert

PREVIOUS WORK - TUYA-CONVERT

WHAT WAS THE ISSUE?



GAMEPLAN


LSC
Smart
Connect
WIFI
2.4GHZ
RGB +
TUNABLE
WHITE
Smart
LED
806
lumen
E27
F
220-240V AC, 9.4 W, 50/60 Hz, RGB • 2700 K - 6500 K

GAMEPLAN

- Get the PSK
  - Overwrite
  - Leak

GAMEPLAN

- Get the PSK
  - Overwrite
  - Leak
- Downgrade to vulnerable protocol version

GAMEPLAN

- Get the PSK
  - Overwrite
  - Leak
- Downgrade to vulnerable protocol version
- Get code execution on the light bulb

ESP8266 - INITIAL INVESTIGATION

DUMPING FIRMWARE

ESPTOOL

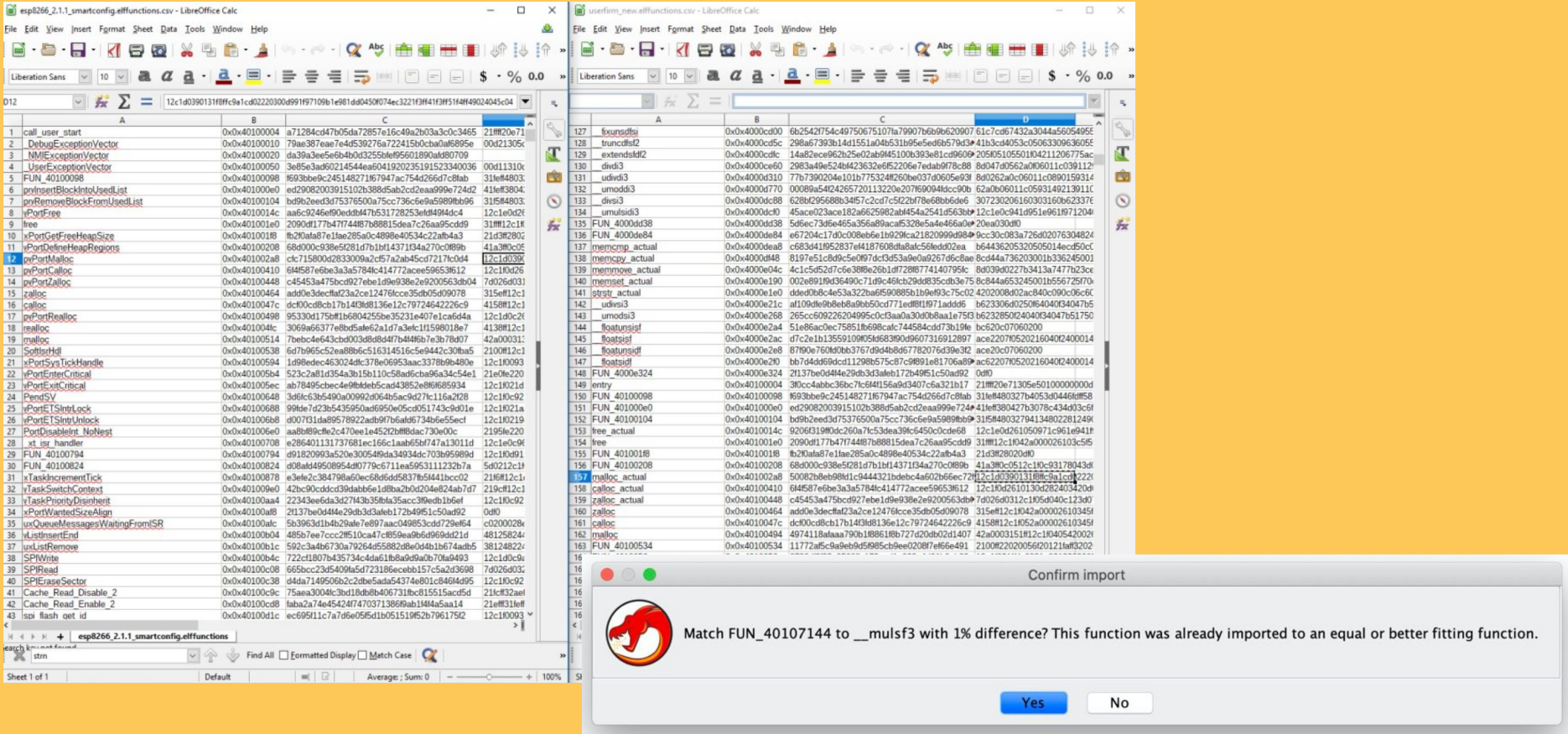
esptool.py [..] read_flash

Flash image

LOADING BINARY INTO GHIDRA

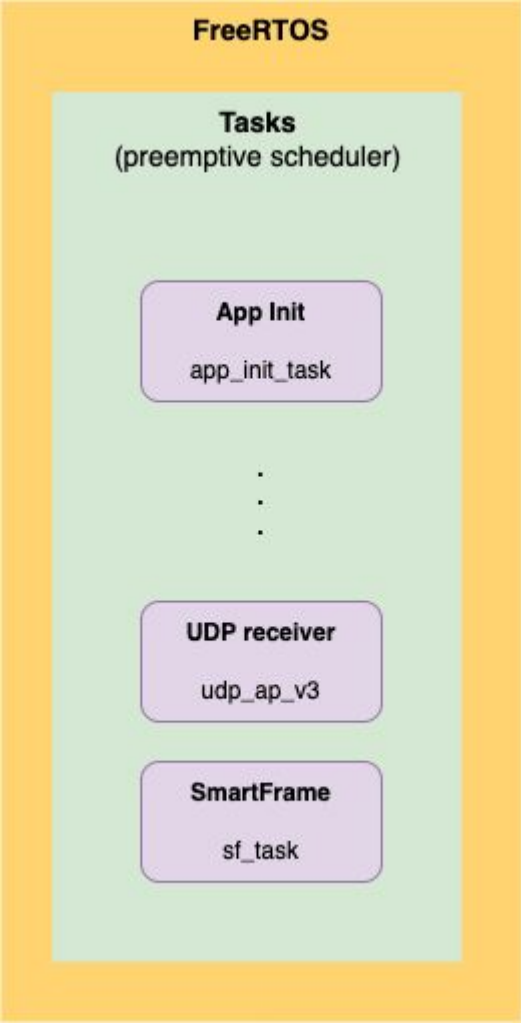
https://github.com/jeremygblake/esp-bin2elf-fork

IDENTIFYING KNOWN FUNCTIONS

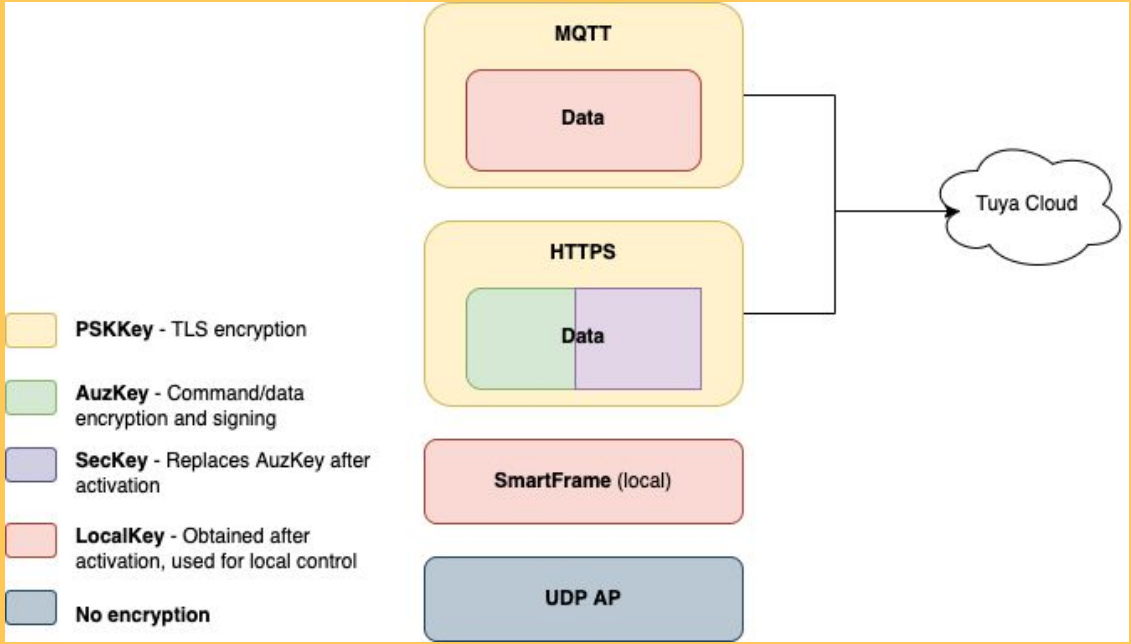


TUYA STACK

- FreeRTOS tasks
  - Initialization
  - "Smart config"
  - MQTT
  - Peripheral control
- HAL
  - Device configuration
  - WiFi
  - Bluetooth
  - Other utilities



GETTING ROOT



ESP8266 - VULNERABILITIES

BAD RANDOM GENERATION ™

```
undefined4 get_random_string(uint unused,char *dest,uint max_length)
{
  int clock_rand;
  int adc_rand;
  uint length;
  clock_rand = system_rtc_clock_cali_proc_inner();
  adc_rand = wrapper_r_rand_from_adc();
  clock_rand = __umodsi3(clock_rand + adc_rand,0x15f);
  length = 0x15eU - clock_rand;
  if (max_length < 0x15eU - clock_rand) {
    length = max_length;
  }
  memcpy_actual(dest,s_BAohbmd6aG91IFR1eVjaG5vbG9SBUwEw_3ffe9310,length);
  return 0;
}
```

BAD RANDOM GENERATION ™

```
undefined4 get_random_string(uint unused,char *dest,uint max_length)
{
  int clock_rand;
  int adc_rand;
  uint length;
  clock_rand = system_rtc_clock_cali_proc_inner();
  adc_rand = wrapper_r_rand_from_adc();
  clock_rand = __umodsi3(clock_rand + adc_rand,0x15f);
  length = 0x15eU - clock_rand;
  if (max_length < 0x15eU - clock_rand) {
    length = max_length;
  }
  memcpy_actual(dest,s_BAohbmd6aG91IFR1eVjaG5vbG9SBUwEw_3ffe9310,length);
  return 0;
}
```

BAD RANDOM GENERATION ™

```
undefined4 get_random_string(uint unuse
{
  int clock_rand;
  int adc_rand;
  uint length;
  clock_rand = system_rtc_clock_cali_pr
  adc_rand = wrapper_r_rand_from_adc();
  clock_rand = __umodsi3(clock_rand + a
  length = 0x15eU - clock_rand;
  if (max_length < 0x15eU - clock_rand)
    length = max_length;
  }
  memcpy_actual(dest,s_BAohbmd6aG91IFR1
  return 0;
}
```

```
Transport Layer Security
  TLSv1.2 Record Layer: Handshake Protocol: Client Hello
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 62
    Handshake Protocol: Client Hello
      Handshake Type: Client Hello (1)
      Length: 58
      Version: TLS 1.2 (0x0303)
      Random: 42416f6842416f68626d6436614739314946523165566a61…
      Session ID Length: 0
      Cipher Suites Length: 4
      Cipher Suites (2 suites)
      Compression Methods Length: 1
      Compression Methods (1 method)
      Extensions Length: 13
      Extension: max_fragment_length (len=1)
      Extension: encrypt_then_mac (len=0)
      Extension: extended_master_secret (len=0)

0000  6e 13 d1 9c 71 dd c4 4f  33 b8 88 8b 08 00 45 00   n···q··O 3·····E·
0010  00 6b 00 10 00 00 ff 06  53 12 0a 2a 2a 16 0a 2a   ·k····· S··**··*
0020  2a 01 bf 6b 01 bb 00 00  19 70 20 48 7a fb 50 18   *··k···· ·p Hz·P·
0030  11 1c 47 d9 00 00 16 03  03 00 3e 01 00 00 3a 03   ··G····· ··>···:·
0040  03 42 41 6f 68 42 41 6f  68 62 6d 64 36 61 47 39   ·BAohBAo hbmd6aG9
0050  31 49 46 52 31 65 56 6a  61 47 35 76 62 47 39 53   1IFR1eVj aG5vbG9S
0060  42 00 00 04 00 ae 00 ff  01 00 00 0d 00 01 00 01   B······ ........
0070  02 00 16 00 00 00 17 00  00                        ······ ··
```

BAD RANDOM GENERATION ™

```
undefined4 get_random_string(uint unuse
{
  int clock_rand;
  int adc_rand;
  uint length;
  clock_rand = system_rtc_clock_cali_pr
  adc_rand = wrapper_r_rand_from_adc();
```

```
Transport Layer Security
  TLSv1.2 Record Layer: Handshake Protocol: Client Hello
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 62
    Handshake Protocol: Client Hello
      Handshake Type: Client Hello (1)
      Length: 58
      Version: TLS 1.2 (0x0303)
      Random: 42416f6842416f68626d6436614739314946523165566a61…
      Session ID Length: 0
      Cipher Suites Length: 4
      Cipher Suites (2 suites)
      Compression Methods Length: 1
      Compression Methods (1 method)
      Extensions Length: 13
      Extension: max_fragment_length (len=1)
      Extension: encrypt_then_mac (len=0)
      Extension: extended_master_secret (len=0)
```

```
88 8b 08 00 45 00   n···q··O 3·····E·
0a 2a 2a 16 0a 2a   ·k····· S··**··*
20 48 7a fb 50 18   *··k···· ·p Hz·P·
3e 01 00 00 3a 03   ··G····· ··>···:·
6d 64 36 61 47 39   ·BAohBAo hbmd6aG9
35 76 62 47 39 53   1IFR1eVj aG5vbG9S
00 0d 00 01 00 01   B····· ·····
                    ····· ··
```

This function which returns a prefix of a long hardcoded string is provided as the RNG function to be used by mbedTLS through an invocation of mbedtls_ssl_conf_rng. It is then used for all operations which require randomness for TLS purposes. Said operations include parameter generation for key exchange algorithms. For example, the ClientRandom in the TLS Client Hello message as shown in traffic, as well as ClientSecret for DH key exchange. Meaning that TLS MITM is possible by leveraging the predicted client-side secrets (with high frequency of success) to derive the TLS Pre-master Secret.

It is not exploitable since KX-based suites are not used at all, only TLS-PSK is.

STACK + HEAP BUFFER OVERFLOW

```
[N]mqtt_client.c:603 gw wifi stat is:3
Fatal exception (9):
epc1=0x401000e3
epc2=0x00000000
epc3=0x4025eeb0
epcvaddr=0x40006c6e
depc=0x00000000
rtn_add=0x401003d4

Free Heap Size: 27112
Stack Point: 3fff5900
3fff5900: 40226890 80000000 00006a20 3ffefe44
3fff5910: 00000023 3fff5940 3fff1b2c 00000000
3fff5920: 00000000 40108800 00000023 401004a6
3fff5930: 00000000 40108800 00000023 4025eb65
3fff5940: 00000045 40107ec8 00000001 00000000
3fff5950: 40108800 00000000 00000020 4025f260
3fff5960: 40107f80 40107ec8 00000022 00000008
3fff5970: 00000000 00000024 00000024 00000000
```

STACK + HEAP BUFFER OVERFLOW

```
[N]mqtt_client.c:603 gw wifi stat is:3
Fatal exception (9):
epc1=0x401000e3
epc2=0x00000000
epc3=0x4025eeb0
epcvaddr=0x40006c6e
depc=0x00000000
rtn_add=0x401003d4

Free Heap Size: 27112
```

Stack buffer overflow in the smart_config task, which is implemented as an infinite loop and never returns, hence no IP control. No interesting and usable data on the stack either.

Heap buffer overflow vulnerability in the same smart_config task. Manipulating heap blocks is possible, but could not exploit it further due to input constraints (no null bytes allowed in trigger payload).

"DYNAMIC ANALYSIS" - RAM DUMP

- Reset to bootrom
- esptool to dump RAM

"DYNAMIC ANALYSIS" - RAM DUMP

- Reset to bootrom
- esptool to dump RAM

```
                    BlockLink_40107720                          XREF[2]:     3ffefe2c(*), 3fffaab0(*)
40107720 50 77 10        BlockLink
         40 30 00
         00 80 90 ...
  40107720 50 77 10 40    BlockLin...BlockLink_40107750      pxNextFreeBl... =                    XREF[2]:     3ffefe2c(*), 3fffaab0(*)
  40107724 30 00 00 80    size_t     80000030h               xBlockSize
  40107728 90 68 22 40    char *     s_user_app_40226890     file           = "user_app"
  4010772c 00 00 00 00    uint32_t   0h                      line
40107730 30              ??         30h    0                                   ?  ->  40109d30
40107731 9d              ??         9Dh
40107732 10              ??         10h
40107733 40              ??         40h    @
```

"DYNAMIC ANALYSIS" - RAM DUMP

- Reset to bootrom
- esptool to dump RAM
- Debugging over JTAG with gdb

```
                    BlockLink_40107720                          XREF[2]:     3ffefe2c(*), 3fffaab0(*)
40107720 50 77 10       BlockLink
         40 30 00
         00 80 90 ...
  40107720 50 77 10 40    BlockLin...BlockLink_40107750     pxNextFreeBl... =                XREF[2]:     3ffefe2c(*), 3fffaab0(*)
  40107724 30 00 00 80    size_t     80000030h              xBlockSize
  40107728 90 68 22 40    char *     s_user_app_40226890    file          = "user_app"
  4010772c 00 00 00 00    uint32_t   0h                     line
40107730 30             ??         30h    0                                  ?  ->  40109d30
40107731 9d             ??         9Dh
40107732 10             ??         10h
40107733 40             ??         40h    @
```

notkmhn / **binutils-gdb-xtensa** Public

forked from jcmvbkbc/binutils-gdb-xtensa

binutils and GDB for xtensa

GPL-2.0 and 3 other licenses found

0 stars **7** forks

Star | Notifications

Code | Pull requests | Actions | Projects | Wiki | Security | ...

master | Go to file

This branch is 1 commit ahead, 4653 commits behind jcmvbkbc:master. Contribute

notkmhn Changes to use gdb with esp8266-openocd ... on Apr 8, 2021 **101,385**

SUMMARY

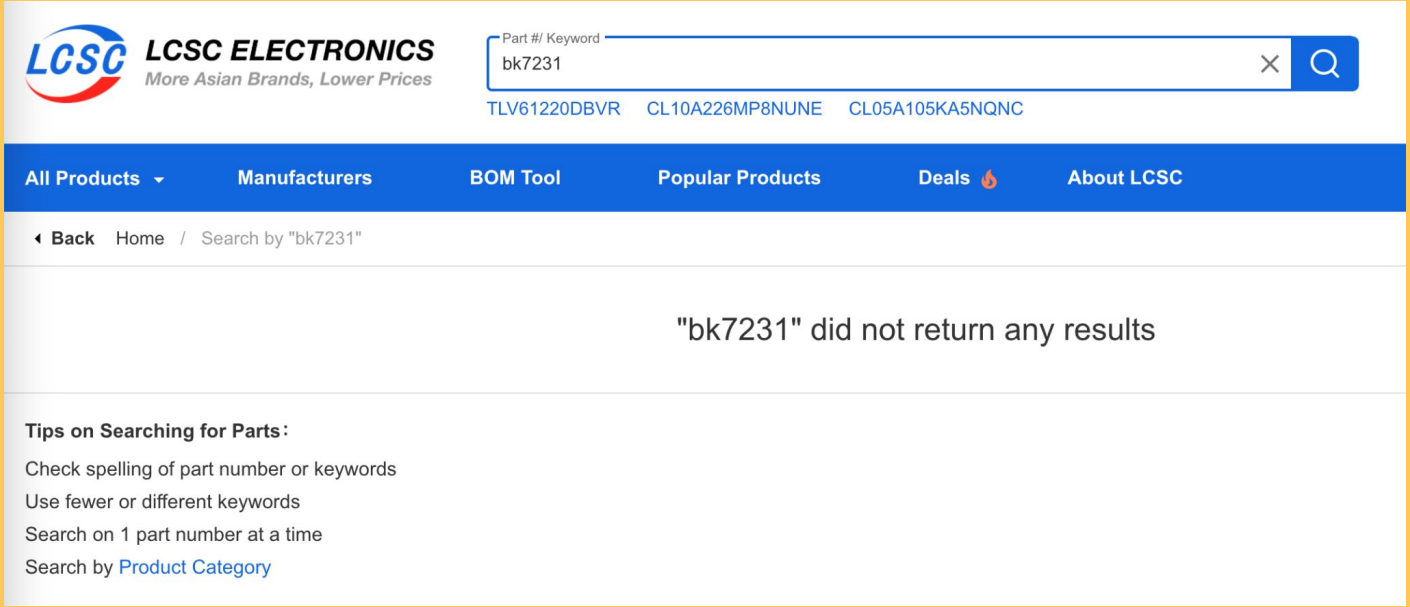
- Found a few bugs that weren't easily exploitable
- RAM dumping using esptool for static analysis with Ghidra
- Debugging worked somewhat
- Learned a lot about Tuya's stack


A FEW
MOMENTS LATER

NEW CHIP, WHO DIS?

BK7231 ARCHITECTURE

FINDING DATASHEETS



BK7231 ARCHITECTURE

BAIDU TO THE RESCUE

BK7231 ARCHITECTURE

DATASHEET

BEKEN

BK7231 Data Sheet QN40V0.6

BK7231 Data Sheet

BK7231 ARCHITECTURE

968E-S (ARM9 W/ ARMV5TE) ARCH

ty of network protocols.

double I2C

nel ADC An input voltage

e Support

F switches, using QFN Package,

stand by Wi-Fi STA , AP , Direct , Repeater mode

stand by SGI , Green-Field Preamble and A-MPDU

stand by WPA , WPA2 with WAPI Security Protocol

stand by 802.11e as well as WMM-PS protocol

ARM968E-S MCU The highest frequency 120 MHz

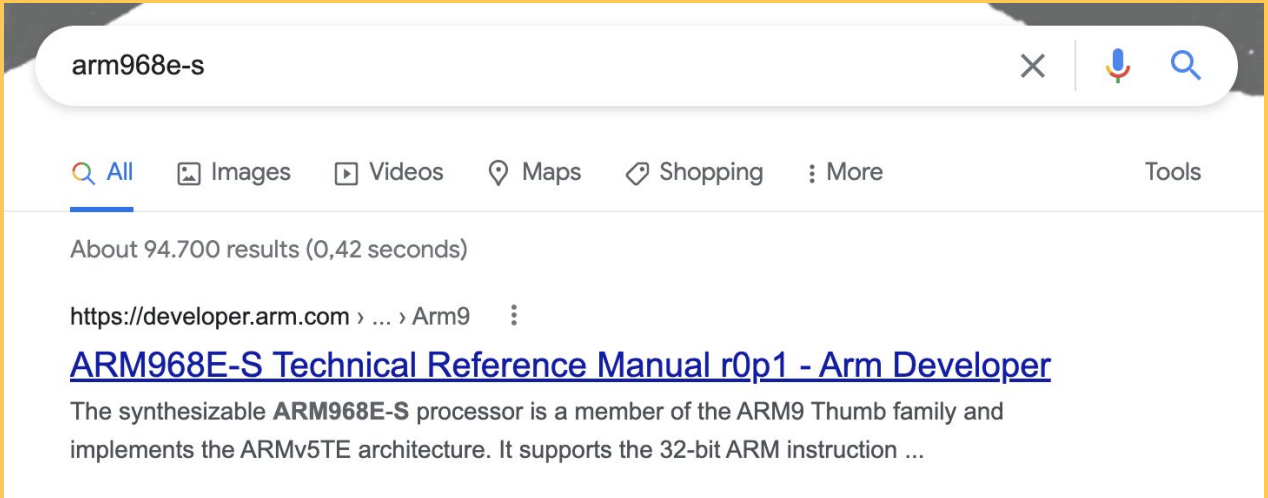
Chip FLASH , Support for transparent download

Chip 256 Kbyte data RAM 50 MHz SDIO Interface and

singlet SPI interface

BK7231 ARCHITECTURE

968E-S (ARM9 W/ ARMV5TE) ARCH



BK7231 ARCHITECTURE

968E-S (ARM9 W/ ARMV5TE) ARCH

Contents

ARM968E-S Technical Reference Manual

Google

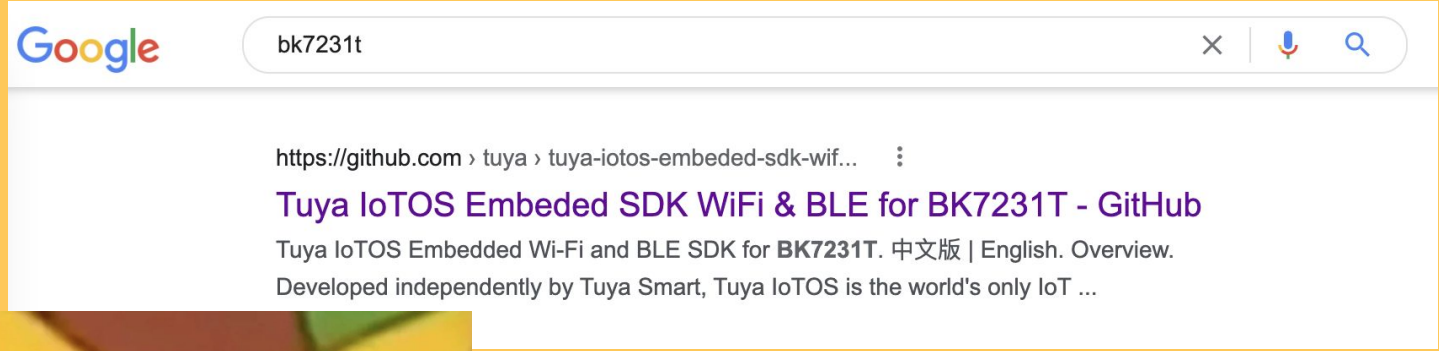
bk7231t

https://github.com › tuya › tuya-iotos-embeded-sdk-wif...

Tuya IoTOS Embeded SDK WiFi & BLE for BK7231T - GitHub

Tuya IoTOS Embedded Wi-Fi and BLE SDK for **BK7231T**. 中文版 | English. Overview. Developed independently by Tuya Smart, Tuya IoTOS is the world's only IoT ...

OPEN SOURCE!


Google
bk7231t
https://github.com › tuya › tuya-iotos-embeded-sdk-wif...
Tuya IoTOS Embeded SDK WiFi & BLE for BK7231T - GitHub
Tuya IoTOS Embedded Wi-Fi and BLE SDK for BK7231T. 中文版 | English. Overview.
Developed independently by Tuya Smart, Tuya IoTOS is the world's only IoT ...

OPEN SOURCE!

WITH SOME BLOBS

tuya / tuya-iotos-embeded-sdk-wifi-ble-bk7231t Public

Code | Issues 2 | Pull requests | Actions | Projects | Wiki | Security | Insights

master | 1 branch | 1 tag | Go to file | Add file | Code

chenyisong update package.exe — 4c04a6c on 23 Sep 2021 — 9 commits

apps	update template_demo PRODECT_ID	13 months ago
platforms/bk7231t	update package.exe	6 months ago
sdk	tuya iotos embeded wifi&ble sdk for bk7231t version 1.0.2	14 months ago
.gitignore	tuya iotos embeded wifi&ble sdk for bk7231t version 1.0.2	14 months ago
README.md	Update README.md	13 months ago
README_zh.md	fix README.md link address	14 months ago
build_app.sh	tuya iotos embeded wifi&ble sdk for bk7231t version 1.0.2	14 months ago

IOTOS

README.md

Tuya IoTOS Embedded Wi-Fi and BLE SDK for BK7231T

中文版 | English

Overview

Developed independently by Tuya Smart, Tuya IoTOS is the world's only IoT operating system covering all levels of IoT sensing, interruption, network, platform, and application. Benefiting from Tuya Smart's accumulation in the IoT industry, TuyaIoTOS provides solutions for a full range of products from product design, R&D, to post-operation.

Tuya IoTOS embedded SDK is an important part of Tuya IoTOS. By virtue of dedicated design, it provides customers with unified APIs, rich SDKs, and DIY functions, enhancing the integrality of the IoT industry. It can be applied to industrial IoT, vehicle networking, security monitoring, outing, and smart home development.

IOTOS

ACTUALLY FREERTOS

master | tuya-iotos-embeded-sdk-wifi-ble-bk7231t / platforms / bk7231t / bk7231t_os / beken378 / os / | Go to file | Add file | ···

chenyisong tuya iotos embeded wifi&ble sdk for bk7231t version 1.0.2 — dd9c585 on 27 Jan 2021 — History

..		
FreeRTOSv9.0.0	tuya iotos embeded wifi&ble sdk for bk7231t version 1.0.2	14 months ago
include	tuya iotos embeded wifi&ble sdk for bk7231t version 1.0.2	14 months ago
mem_arch.c	tuya iotos embeded wifi&ble sdk for bk7231t version 1.0.2	14 months ago
str_arch.c	tuya iotos embeded wifi&ble sdk for bk7231t version 1.0.2	14 months ago

A NEW CHIP, A NEW BUG

```c
struct LAN_AP_NW_CFG_S {
    // ...
    char ap_cfg_token[64];
    int fd;
    short log_ack_timer;
    short send_log_mid;
    int (*finish_cb)(PTR_SSID_PASSWORD_TOKEN, int);
    SSID_PASSWORD_TOKEN spt;
    // ...
};

struct LAN_AP_NW_CFG_S lan_ap_nw_cfg;

//...

void __udp_ap_v3_task()
{
    // ...
    cJSON *json_object = parse_json_payload();
    cJSON *ssid = cJSON_GetObjectItem("ssid", json_object);
    cJSON *password = cJSON_GetObjectItem("passwd", json_object);
    cJSON *token = cJSON_GetObjectItem("token", json_object);
    // ...
    strncpy(lan_ap_nw_cfg->spt.ssid,ssid->valuestring, 32);
    strncpy(lan_ap_nw_cfg->spt.passwd,password->valuestring, 64);

    int token_length = strlen(token);
    memcpy(lan_ap_nw_cfg.ap_cfg_token, token, token_length);

    int result = lan_ap_nw_cfg->finish_cb(lan_ap_nw_cfg->spt, 0x10002);
    cJSON_Delete(json_object);
    // ...
}
```

A NEW CHIP, A NEW BUG

```c
struct LAN_AP_NW_CFG_S {
    // ...
    char ap_cfg_token[64];
    int fd;
    short log_ack_timer;
    short send_log_mid;
    int (*finish_cb)(PTR_SSID_PASSWORD_TOKEN, int);
    SSID_PASSWORD_TOKEN spt;
    // ...
};

struct LAN_AP_NW_CFG_S lan_ap_nw_cfg;

//...

void __udp_ap_v3_task()
{
    // ...
    cJSON *json_object = parse_json_payload();
    cJSON *ssid = cJSON_GetObjectItem("ssid", json_object);
    cJSON *password = cJSON_GetObjectItem("passwd", json_object);
    cJSON *token = cJSON_GetObjectItem("token", json_object);
    // ...
    strncpy(lan_ap_nw_cfg->spt.ssid,ssid->valuestring, 32);
    strncpy(lan_ap_nw_cfg->spt.passwd,password->valuestring, 64);

    int token_length = strlen(token);
    memcpy(lan_ap_nw_cfg.ap_cfg_token, token, token_length);

    int result = lan_ap_nw_cfg->finish_cb(lan_ap_nw_cfg->spt, 0x10002);
    cJSON_Delete(json_object);
    // ...
}
```

```json
{
    "ssid": "AP SSID",
    "passwd": "AP passphrase",
    "token": "AP configuration token"
}
```

A NEW CHIP, A NEW BUG

```c
struct LAN_AP_NW_CFG_S {
    // ...
    char ap_cfg_token[64];
    int fd;
    short log_ack_timer;
    short send_log_mid;
    int (*finish_cb)(PTR_SSID_PASSWORD_TOKEN, int);
    SSID_PASSWORD_TOKEN spt;
    // ...
};

struct LAN_AP_NW_CFG_S lan_ap_nw_cfg;

//...

void __udp_ap_v3_task()
{
    // ...
    cJSON *json_object = parse_json_payload();
    cJSON *ssid = cJSON_GetObjectItem("ssid", json_object);
    cJSON *password = cJSON_GetObjectItem("passwd", json_object);
    cJSON *token = cJSON_GetObjectItem("token", json_object);
    // ...
    strncpy(lan_ap_nw_cfg->spt.ssid,ssid->valuestring, 32);
    strncpy(lan_ap_nw_cfg->spt.passwd,password->valuestring, 64);

    int token_length = strlen(token);
    memcpy(lan_ap_nw_cfg.ap_cfg_token, token, token_length);

    int result = lan_ap_nw_cfg->finish_cb(lan_ap_nw_cfg->spt, 0x10002);
    cJSON_Delete(json_object);
    // ...
}
```

```json
{
    "ssid": "AP SSID",
    "passwd": "AP passphrase",
    "token": "AP configuration token"
}
```

A NEW CHIP, A NEW BUG

```c
struct LAN_AP_NW_CFG_S {
    // ...
    char ap_cfg_token[64];
    int fd;
    short log_ack_timer;
    short send_log_mid;
    int (*finish_cb)(PTR_SSID_PASSWORD_TOKEN, int);
    SSID_PASSWORD_TOKEN spt;
    // ...
};

struct LAN_AP_NW_CFG_S lan_ap_nw_cfg;

//...

void __udp_ap_v3_task()
{
    // ...
    cJSON *json_object = parse_json_payload();
    cJSON *ssid = cJSON_GetObjectItem("ssid", json_object);
    cJSON *password = cJSON_GetObjectItem("passwd", json_object);
    cJSON *token = cJSON_GetObjectItem("token", json_object);
    // ...
    strncpy(lan_ap_nw_cfg->spt.ssid,ssid->valuestring, 32);
    strncpy(lan_ap_nw_cfg->spt.passwd,password->valuestring, 64);

    int token_length = strlen(token);
    memcpy(lan_ap_nw_cfg.ap_cfg_token, token, token_length);

    int result = lan_ap_nw_cfg->finish_cb(lan_ap_nw_cfg->spt, 0x10002);
    cJSON_Delete(json_object);
    // ...
}
```

```json
{
    "ssid": "AP SSID",
    "passwd": "AP passphrase",
    "token": "AP configuration token"
}
```

A NEW CHIP, A NEW BUG

```c
struct LAN_AP_NW_CFG_S {
    // ...
    char ap_cfg_token[64];
    int fd;
    short log_ack_timer;
    short send_log_mid;
    int (*finish_cb)(PTR_SSID_PASSWORD_TOKEN, int);
    SSID_PASSWORD_TOKEN spt;
    // ...
};

struct LAN_AP_NW_CFG_S lan_ap_nw_cfg;

//...

void __udp_ap_v3_task()
{
    // ...
    cJSON *json_object = parse_json_payload();
    cJSON *ssid = cJSON_GetObjectItem("ssid", json_object);
    cJSON *password = cJSON_GetObjectItem("passwd", json_object);
    cJSON *token = cJSON_GetObjectItem("token", json_object);
    // ...
    strncpy(lan_ap_nw_cfg->spt.ssid,ssid->valuestring, 32);
    strncpy(lan_ap_nw_cfg->spt.passwd,password->valuestring, 64);

    int token_length = strlen(token);
    memcpy(lan_ap_nw_cfg.ap_cfg_token, token, token_length);

    int result = lan_ap_nw_cfg->finish_cb(lan_ap_nw_cfg->spt, 0x10002);
    cJSON_Delete(json_object);
    // ...
}
```

```json
{
    "ssid": "AP SSID",
    "passwd": "AP passphrase",
    "token": "AP configuration token"
}
```

DUMPING FIRMWARE

DUMPING FIRMWARE

SERIAL PROTOCOL

- FLASH READING/WRITING
- DEVICE METADATA
- SPECIFICATION IS .. UNKNOWN

DUMPING FIRMWARE

SERIAL PROTOCOL

- FLASH READING/WRITING
- DEVICE METADATA
- SPECIFICATION IS .. ~~UNKNOWN~~ SORT OF UNKNOWN

Module pin	Serial pin
RXD1	TX
TXD1	RX
VCC	VCC3.3V
GND	GND

Prepare software

Download and open the BK7231T chip flashing tool. The following figure shows the window of the chip flashing tool. The following table describes the required parameters boxed in red in the figure.

Parameter	Description
Flashing Target	The chip platform is BK7231T. Therefore, **BK7231 is used.**
Start address	Set the value to `0x00011000` in most cases.
Operation length	Select `0x001EF000 (0x00200000-0x00011000)`
Baud rate	Select `921600`

DUMPING FIRMWARE

MOAR OPEN SOURCE!

tiancj / **hid_download_py** Public

4 stars 11 forks

Star Watch

Code Issues Pull requests Actions Projects Wiki ...

master ...

tiancj change crc timeout ... on 17 Feb 58

View code

README.md

Install for Debian/Ubuntu/Linux Mint

DUMPING FIRMWARE

MOAR OPEN SOURCE!

... BUT IT ISN'T RELIABLE.

tiancj / **hid_download_py** Public

4 stars 11 forks

Star Watch

Code Issues Pull requests Actions Projects Wiki ...

master ...

tiancj change crc timeout ... on 17 Feb 58

View code

README.md

Install for Debian/Ubuntu/Linux Mint

DUMPING FIRMWARE

REVERSING THE BOOTLOADER AND COMPARING WITH THE OPEN SOURCE TOOL

```
case 8:
  zero_var = *(ushort *)(maybe_parse_state + 2);
  cmd_type_var = *request_bytes;
  one_var_2byte = zero_var + 1 & 0xfffeffff;
  *(short *)(maybe_parse_state + 2) = (short)one_var_2byte;
  puVar3[zero_var] = cmd_type_var;
  cmd_type_again = *puVar3;
  if ((cmd_type_again == '\x09') && (one_var_2byte == *(ushort *)(maybe_parse_state + 8))) {
                /* if command type == 0x9
                   and command length lsb == 0x1

                   This seems to be flash read 4k */
    addr_for_cmd = *(uint *)(puVar3 + 1);
    *(uint *)(maybe_parse_state + 0x1c) = addr_for_cmd;
                /* if addr < 0x10000 */
    if (addr_for_cmd < 0x10000) {
      puVar3[5] = (char)*(ushort *)(maybe_parse_state + 8) + -5;
      maybe_build_cmd_response(9,6,7,puVar3 + 1);
    }
```

```
def BuildCmd_FlashRead4K(addr: int):
    length=1+(4+0)
    buf = bytearray(4096)
    buf[0]=0x01
    buf[1]=0xe0
    buf[2]=0xfc
    buf[3]=0xff
    buf[4]=0xf4
    buf[5]=(length&0xff)
    buf[6]=((length>>8)&0xff)
    buf[7]=CMD_FlashRead4K
```

DUMPING FIRMWARE

CREATING A NEW TOOL FOR THE BOOTLOADER SERIAL PROTOCOL

README.md

bk7231tools

This is a collection of tools to interact with and analyze artifacts for BK7231 MCUs.

Contributors

- Kuba Szczodrzyński - @kuba2k2

Packages

No packages published
Publish your first package

Contributors 3

notkmhn Khaled Nassar

tjclement Tom Clement

kuba2k2 Kuba Szczodrzyński

ANALYZING FIRMWARE DUMPS

ANALYZING FIRMWARE DUMPS

Flash layout contains two code partitions

- Bootloader
- User app

Other configuration partitions too, defined by user app

ANALYZING FIRMWARE DUMPS

Flash layout contains two code partitions

- Bootloader
- User app

Other configuration partitions too, defined by user app

AKA OBFUSCATION

ANALYZING FIRMWARE DUMPS

Flash layout contains two code partitions

- Bootloader
- User app

Other configuration partitions too, defined by user app

BK7231TOOLS TO THE RESCUE

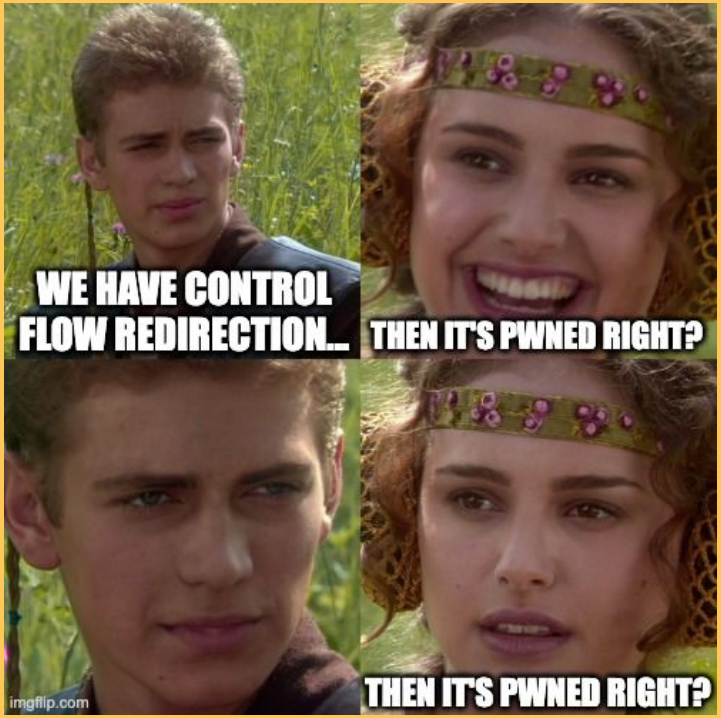
BUG TO JAILBREAK

BUG TO JAILBREAK

```c
struct LAN_AP_NW_CFG_S {
    // ...
    char ap_cfg_token[64];
    int fd;
    short log_ack_timer;
    short send_log_mid;
    int (*finish_cb)(PTR_SSID_PASSWORD_TOKEN, int);
    SSID_PASSWORD_TOKEN spt;
    // ...
};

struct LAN_AP_NW_CFG_S lan_ap_nw_cfg;

//...

void __udp_ap_v3_task()
{
    // ...
    cJSON *json_object = parse_json_payload();
    cJSON *ssid = cJSON_GetObjectItem("ssid", json_object);
    cJSON *password = cJSON_GetObjectItem("passwd", json_object);
    cJSON *token = cJSON_GetObjectItem("token", json_object);
    // ...
    strncpy(lan_ap_nw_cfg->spt.ssid,ssid->valuestring, 32);
    strncpy(lan_ap_nw_cfg->spt.passwd,password->valuestring, 64);

    int token_length = strlen(token);
    memcpy(lan_ap_nw_cfg.ap_cfg_token, token, token_length);

    int result = lan_ap_nw_cfg->finish_cb(lan_ap_nw_cfg->spt, 0x10002);
    cJSON_Delete(json_object);
    // ...
}
```

```json
{
    "ssid": "AP SSID",
    "passwd": "AP passphrase",
    "token": "AP configuration token"
}
```

DONE! OR NOT..?


WE HAVE CONTROL FLOW REDIRECTION...
THEN IT'S PWNED RIGHT?
THEN IT'S PWNED RIGHT?
imgflip.com

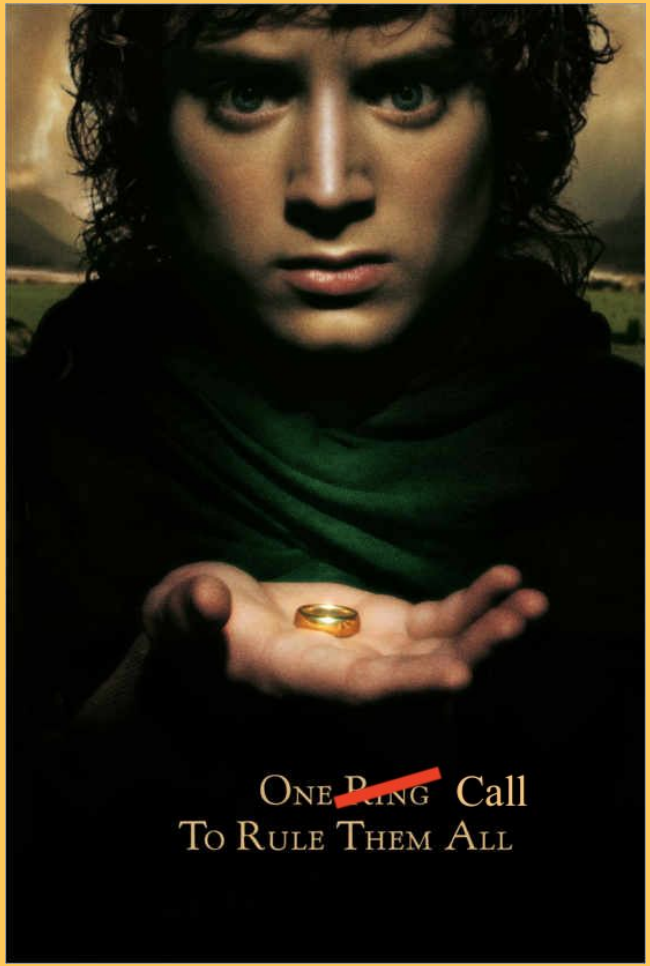
OVERWRITING THE PSK

```
void __tmm_gw_pskkey_get_cb(s_tm_msg *tm_msg)

{
  ...
  OVar1 = http_pskkey_get(&result);
  if (OVar1 == 0) {
        ...
    }
    else {
      ptVar2 = ty_cJSON_GetObjectItem(result,"pskKey");
      __src = ptVar2->valuestring;
      PrintLog(...);
      strncpy(gw_cntl.gw_base.psk_key,__src,0x29);
      OVar1 = wd_gw_base_if_write(&gw_cntl.gw_base);
      if (OVar1 == 0) {
        tuya_tls_register_constant
                  (gw_cntl.gw_base.uuid,gw_cntl.gw_base.auth_key,gw_cntl.gw_base.psk_key);
      }
      else {
        PrintLog(...);
      }
      ty_cJSON_Delete(result);
    }
  }
  ...
}
```

REASSESSING WIN CONDITIONS

- Get the PSK
  - Overwrite
  - ~~Leak~~
- ~~Downgrade to vulnerable protocol version~~
- Get code execution on the device itself
- Overwrite security keys

OVERWRITING KEYS



GADGET HUNTING

- Function that sets the security keys to jump into
- A "fixup" gadget to set up the registers as needed

GADGET HUNTING

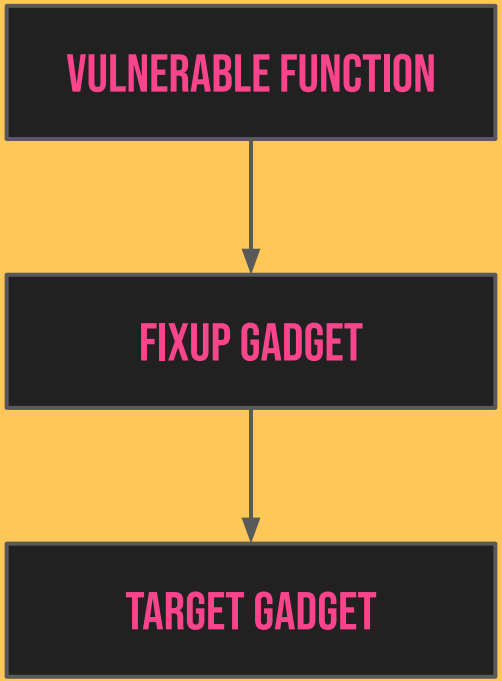
TARGET GADGET

```
void __mf_cmd_process(MF_PRO_HEAD_S *hd) {
    ...
    ptVar17 = ty_cJSON_GetObjectItem(ptVar9,"uuid");
    strcpy((char *)pGVar18,ptVar17->valuestring);
    ptVar17 = ty_cJSON_GetObjectItem(ptVar9,"pskKey");
    strcpy(pGVar18->psk_key,ptVar17->valuestring);
    ptVar17 = ty_cJSON_GetObjectItem(ptVar9,"auzkey");
    strcpy(pGVar18->auth_key,ptVar17->valuestring);
    ...
    OVar10 = wd_gw_base_if_write(pGVar18);
    ...
}
```

TRAMPOLINE GADGET

```
adds r0, r7, #0
ldr r1, [sp, #8]
ldr r3, [r5, #0x20]
blx r3
```

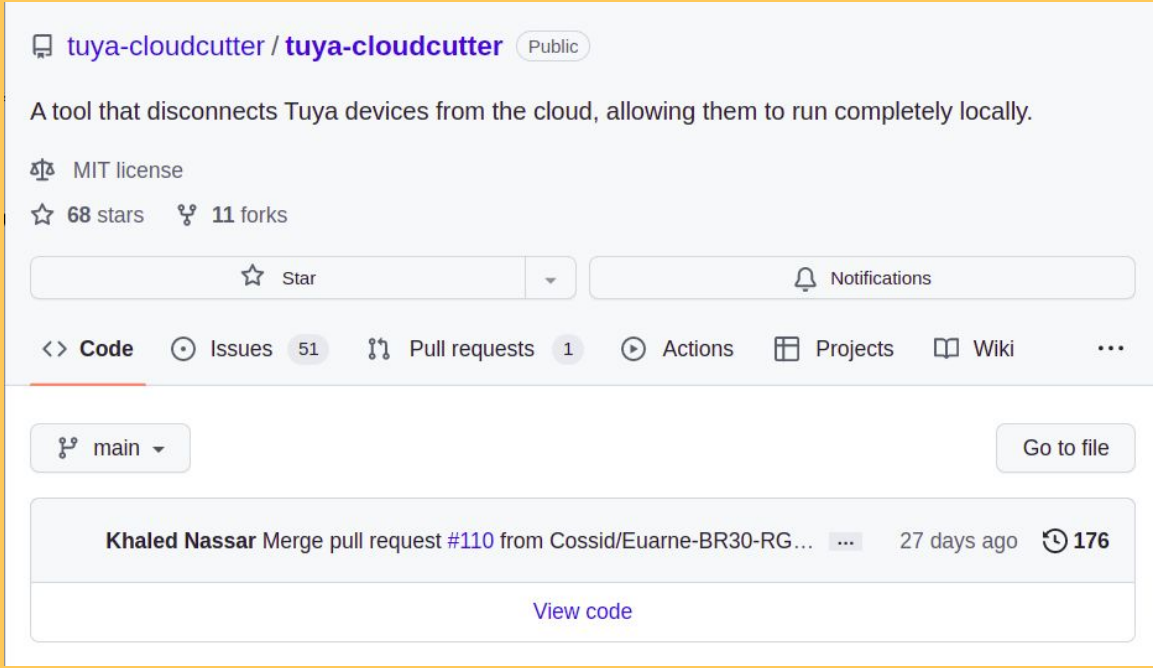
PUTTING EVERYTHING TOGETHER



RELEASING INTO THE WILD

RELEASING INTO THE WILD

TUYA-CLOUDCUTTER



https://github.com/tuya-cloudcutter/tuya-cloudcutter

DEMO TIME

IN CLOSING

- Found a vulnerability affecting almost all BK7231 devices to date
- Tuya was really cool about it -> we sent bug bounty to charity
- Embedded security is catching up -> still interesting target

GETTING IN TOUCH

KHALED NASSAR

notkmhn

@notkmhn

TOM CLEMENT

tjclement

@Tom_Clement

ACKNOWLEDGMENTS

- ius - ESP8266 JTAG debugging
- blasty (@bl4sty) - ESP8266 vulnerability analysis support
- Jilles Groenendijk (@jilles_com) - BK7231 firmware gathering and support
- V-TRUST for disclosure tips