MWR LABS

Security Advisory

Bluetooth Pairing Authentication Bypass

12/04/2016

Software	Android Open Source Project (AOSP)
Affected Versions	4.2.0 - 4.4.4, 5.0.0 - 5.1.1, 6.0.0 – 6.0.1
CVE Reference	CVE-2016-0850
Author	Romain Trouvé
Severity	High
Vendor	Google
Vendor Response	Patch Released

Description:

A vulnerability in Bluetooth Security Manager could enable an untrusted device to pair with a phone during an initial pairing process. This could lead to unauthorized access of the device resources.

Impact:

An attacker would have access to a range of Bluetooth Profiles [1] compatible with the device such as the HID Profile for the support of mice, keyboards or GAVDP Profile for relaying video/audio stream; some require additional authorization. As proof of concept, an untrusted device was paired with the victim's phone and was then able to use the Bluetooth tethering feature to access the Internet connection.

Before the initial pairing authentication process times out, multiple devices can be paired in a row without the user's knowledge. The Bluetooth User Interface does not reveal the successful pairing(s) in the paired devices list.

Cause:

An untrusted device could abuse the Porsche car-kit pairing workaround to generate a reply to a legacy pin code request during an initial pairing process.

Solution:

Google have released a security update through an over-the-air (OTA) update as part of its Android Security Bulletin Monthly Release process. Please refer to the Nexus Security Bulletin – April 2016 [2].

The Porsche car-kit pairing workaround has been removed. (Change-Id: I14c5e3fcda0849874c8a94e48aeb7d09585617e1)

Technical details:

Since Android 4.2, Android Open Source Project has switched its Bluetooth stack from BlueZ to BlueDroid. To work around a Porsche car-kit pairing conflict, some conditional compilation code had been added in the Bluetooth Security Manager.

Defined within PORCHE_PAIRING_CONFLICT, the workaround affected the btm_sec_pin_code_request() function which is called when the controller requests a legacy PIN code.

1. **Android 4.2 – 5.0.0 [3]**

```
void btm_sec_pin_code_request (UINT8 *p_bda)
{
    tBTM_SEC_DEV_REC *p_dev_rec;
    tBTM_CB          *p_cb = &btm_cb;
    ...
    if (btm_cb.pairing_state != BTM_PAIR_STATE_IDLE)
    {
        if ( (memcmp (p_bda, btm_cb.pairing_bda, BD_ADDR_LEN) == 0)  &&
             (btm_cb.pairing_state == BTM_PAIR_STATE_WAIT_AUTH_COMPLETE) )
        {
            ...
        } else if ((btm_cb.pairing_state != BTM_PAIR_STATE_WAIT_PIN_REQ)
                || memcmp (p_bda, btm_cb.pairing_bda, BD_ADDR_LEN) != 0)
```

```
        {
            ...
#ifdef PORCHE_PAIRING_CONFLICT
            ...
            if(! btm_cb.pin_code_len_saved)
            {
                btsnd_hcic_pin_code_neg_reply (p_bda);
            }
            else
            {
                btm_sec_change_pairing_state (BTM_PAIR_STATE_WAIT_AUTH_COMPLETE);
                btsnd_hcic_pin_code_req_reply (p_bda, btm_cb.pin_code_len_saved, p_cb-
>pin_code);
            }
#else
            btsnd_hcic_pin_code_neg_reply (p_bda);
#endif
            return;
        }
    ...
```

The PORCHE_PAIRING_CONFLICT workaround is accessible if the device is in a non-idle pairing state and a mismatch occurs between p_bda and btm_cb.pairing_bda (both of which can be controlled by an attacker).

- p_bda is the device MAC address currently pairing, responsible of the legacy pin code request event.
- btm_cb.pairing_bda is the peer device MAC address, part of the changing pairing event.

From an attacker's perspective, opening a first pairing process and then initiating a legacy pin code pairing with a different MAC address will give access to the PORCHE_PAIRING_CONFLICT code.

The workaround included a branch to a legacy pin code reply. A saved pin code from a previously successful legacy pairing should exist to trigger the reply. In this scenario, instead of asking the GUI plugin agent for a pin, the saved pin_code is re-used in response to a legacy pin code request. Performing an online pin code guessing/bruteforce attack would lead to a complete and unauthorized pairing.

2. **Android 5.0.0 and later**

From KitKat to Lollipop (Android 5.0.0), the following patch has been merged with the AOSP Bluetooth stack:

- Change-Id: I67b2e689cfcc52c93fdda62dd742812698baa0e6 [4] committed on Apr 17, 2013 and merged on Mar 25, 2014

```
diff --git a/stack/btm/btm_sec.c b/stack/btm/btm_sec.c
index 8bcf435..1e15d78 100644
--- a/stack/btm/btm_sec.c
+++ b/stack/btm/btm_sec.c

@@ -4774,6 +4774,10 @@
     tBTM_SEC_DEV_REC *p_dev_rec;
     tBTM_CB          *p_cb = &btm_cb;

+#ifdef PORCHE_PAIRING_CONFLICT
+    UINT8 default_pin_code_len = 4;
+    PIN_CODE default_pin_code = {0x30, 0x30, 0x30, 0x30};
+#endif
     BTM_TRACE_EVENT3 ("btm_sec_pin_code_request()  State: %s, BDA:%04x%08x",
                       btm_pair_state_descr(btm_cb.pairing_state),
                       (p_bda[0]<<8)+p_bda[1],
(p_bda[2]<<24)+(p_bda[3]<<16)+(p_bda[4]<<8)+p_bda[5] );
@@ -4807,7 +4811,8 @@
             BTM_TRACE_EVENT0 ("btm_sec_pin_code_request from remote dev. for local
initiated pairing");
             if(! btm_cb.pin_code_len_saved)
             {
-                btsnd_hcic_pin_code_neg_reply (p_bda);
+                btm_sec_change_pairing_state (BTM_PAIR_STATE_WAIT_AUTH_COMPLETE);
+                btsnd_hcic_pin_code_req_reply (p_bda, default_pin_code_len,
default_pin_code);
             }
             else
             {
```

This patch introduced a hard-coded pin code {0x30, 0x30, 0x30, 0x30} i.e. 0000. In the event of no existing saved pin_code, the device would use the hard-coded pin to generate a reply to a legacy pin code request. This would lead to an unauthorized pairing in response to a legacy pin code 0000.

Detailed Timeline

Date	Summary
2016-01-13	Reported to Android Open Source Project (AOSP) Issue Tracker
2016-01-13	Report acknowledged by Google
2016-01-21	Technical details reviewed by The Android Security Team and Severity set
2016-02-24	Google informed to release a patch in an upcoming bulletin
2016-04-04	Nexus Security Bulletin (April 2016) Published

[1] https://developer.bluetooth.org/TechnologyOverview/Pages/Profiles.aspx

[2] https://source.android.com/security/bulletin/2016-04-02.html

[3] https://android.googlesource.com/platform/external/bluetooth/bluedroid/+/android-4.2_r1

[4] https://android-review.googlesource.com/#/c/88928/