February 6, 2020
by Jan Ruge

Critical Bluetooth Vulnerability in Android (CVE-2020-0022) – BlueFrag

On November 3rd, 2019, we have reported a critical vulnerability affecting the Android Bluetooth subsystem. This vulnerability has been assigned CVE-2020-0022 and was now patched in the latest security patch from February 2020. The security impact is as follows:

- On Android 8.0 to 9.0, a remote attacker within proximity can silently execute arbitrary code with the privileges of the Bluetooth daemon as long as Bluetooth is enabled. No user interaction is required and only the Bluetooth MAC address of the target devices has to be known. For some devices, the Bluetooth MAC address can be deduced from the WiFi MAC address. This vulnerability can lead to theft of personal data and could potentially be used to spread malware (Short-Distance Worm).
- On Android 10, this vulnerability is not exploitable for technical reasons and only results in a crash of the Bluetooth daemon.
- Android versions even older than 8.0 might also be affected but we have not evaluated the impact.

Users are strongly advised to install the latest available security patch from February 2020. If you have no patch available yet or your device is not supported anymore, you can try to mitigate the impact by some generic behavior rules:

- Only enable Bluetooth if strictly necessary. Keep in mind that most Bluetooth enabled headphones also support wired analog audio.
- Keep your device non-discoverable. Most are only discoverable if you enter the Bluetooth scanning menu. Nevertheless, some older phones might be discoverable permanently.

As soon as we are confident that patches have reached the end users, we will publish a technical report on this vulnerability including a description of the exploit as well as Proof of Concept code.

Android Bluetooth exploit

Comments

Wade Mealing says: FEBRUARY 7, 2020 AT 3:39 AM

Are you confident that this doesn't affect generic Linux ?

Reply

jruge says: FEBRUARY 7, 2020 AT 10:09 AM against an Ubuntu system and could not observe any crashes.

Reply

shinigami says: FEBRUARY 7, 2020 AT 12:58 PM

nevermind. bluetooth itself will not usually work on linux.

Reply

Farmer Bob says: FEBRUARY 7, 2020 AT 1:28 PM

🤣🤣🤣

jay says: FEBRUARY 7, 2020 AT 2:14 PM

Bluetooth works just fine on all of my Linux systems.

Jan Ciger (@janoc200) says: FEBRUARY 7, 2020 AT 2:47 PM

Bluetooth works in Linux just fine, pretty much on par with MacOS and far better than in Windows. Sounds more like confusing operator’s incompetence with an OS problem to me.

face_it says: FEBRUARY 7, 2020 AT 2:51 PM

You mean, the linux *you* set up doesn’t work. Don’t worry, you can do better buddy

Linus T. says: FEBRUARY 7, 2020 AT 5:06 PM

Sarcasm doesn’t run well in most Linux geeks but I’m fairly sure Linus is working in that.

Search ...

N says: FEBRUARY 8, 2020 AT 1:01 AM

That was hilarious 🤣🤣🤣🤣🤣

yayaya says: FEBRUARY 10, 2020 AT 4:29 PM

LoL

Patrick says: FEBRUARY 7, 2020 AT 2:29 PM exploit as well as Proof of Concept code."

Given the generally terrible rate of patching on android handsets what criteria will you use to judge that 'patches have reached end users'.

Reply

jruge says: FEBRUARY 7, 2020 AT 3:22 PM

That most major Vendors (Samsung, Huawei, Xiaomi) have distributed a patch plus a couple of days for the users to react, what should take approx 2-3 Weeks in total. Therefore even after the 90+14 days used by Google Project Zero [1]

Note that as the patch was now published, the exploit can be reproduced as well. We needed three weeks for this and there are far more competent people out there.

[1] https://googleprojectzero.blogspot.com/2020/01/policy-and-disclosure-2020-edition.html

Reply

John says: APRIL 10, 2020 AT 6:44 AM

So they should be able to publish this now.... The date has passed.

Reply

jioundai says: FEBRUARY 7, 2020 AT 2:38 PM

Have you implemented the full exploit on Android 8/9 using the vulnerability?

Reply

jruge says: FEBRUARY 7, 2020 AT 3:27 PM

Yes. Ultimately a reverse shell over network will be opened.

Reply

Fuzzdamental Bystander says: FEBRUARY 7, 2020 AT 3:31 PM

Did you implement your own fuzzer, or an existing one?

Reply

jruge says: FEBRUARY 7, 2020 AT 3:53 PM

During my thesis at SEEMOO we have implemented our own fuzzer for the Broadcom Bluetooth firmware. In this case we used a modified firmware and have broken Android instead of the firmware by accident ¯_(ツ)_/¯

Reply

Fuzzdamental Bystander says: FEBRUARY 7, 2020 AT 5:06 PM

Are you going to opensource it? I'd love to take a look at this specific-case fuzzer.

disclosure. The code belongs to a larger project called "Frankenstein" that Jiska mentioned briefly at 36c3 [1]. A description of the technique used will be included in the technical post.

[1] https://media.ccc.de/v/36c3-10531-all_wireless_communication_stacks_are_equally_broken

Fuzzdamental Bystander says: FEBRUARY 7, 2020 AT 5:37 PM

Great!

hafis says: FEBRUARY 7, 2020 AT 6:01 PM

hi Ruge, im an digital journalism for Cyberthreat.id, can i know, what is Bluetooth daemon or BluetoothD definition? based on Google, i didn't find the answer… please answer my question thanks

Reply

jruge says: FEBRUARY 7, 2020 AT 8:01 PM

Hi, the Bluetooth daemon is a process on the Android system that runs in the background (daemon) that is responsible for managing the Bluetooth controller and handling of various Bluetooth related protocols, such as HCI, L2CAP and GATT. As it has to process attacker-controlled input it is susceptible to attacks. In addition, it has to run with high privileges (not as 'root' like on Linux) to support features like:
– file transfer => read files
– share Internet connection => configure network and VPN
– Human Interaction Devices => emulate keyboard and mouse

Reply

Ben Siron says: FEBRUARY 8, 2020 AT 2:51 PM

Ruge,

To pinpoint the vulnerability, it looks like the single line code change that fixes the problem, is this one?

/hci/src/packet_fragmenter.cc
line 224

```
– packet->len = partial_packet->len – partial_packet->offset;
+ packet->len =
+ (partial_packet->len – partial_packet->offset) + packet->offset;
```

GAP: Correct the continuous pkt length in l2cap

L2cap continuous pkt length wrongly calculated in
reassembly logic when remote sends more data
than expected.

l2cap reassembly logic.

Bug: 135239489
Bug: 143894715
CRs-Fixed: 2434229
Test: make and internal testing
Change-Id: I758d9e31465b99e436b9b1841320000f08186c97
Merged-In: I758d9e31465b99e436b9b1841320000f08186c97
(cherry picked from commit 337bd4579453bd6bf98ff519de3ac1019cd30d28)
(cherry picked from commit 602f4b44fe30ec8b225e1cee5f96817607d93e5a)

And if so, it looks like this flaw had been present in the code for at least the last 5 years

Reply

jruge says: FEBRUARY 8, 2020 AT 3:18 PM

Yes, this is the fix for the vulnerability. But as you can see from the bulletin, this bug is not exploitable on all Android versions and there is a reason for that. But I'm sorry, no spoilers yet 😉

Reply

Rich says: FEBRUARY 8, 2020 AT 3:55 PM

Do you think you're the first group to find this? This is a pretty serious vulnerability. Especially for devices with sequential NIC MAC IDs

jruge says: FEBRUARY 8, 2020 AT 5:35 PM

We are for sure not the first ones who found this bug. After we have tracked down and understood our crashes we realized, that the bug was already fixed in the master branch by the commit Ben Siron has referenced. The commit was publicly visible at that time but was not applied to any release for some reason. It is hard to blame anyone for this as the severity is not really visible in the code. We were probably the first ones to realize, that this bug is exploitable because we found it in a different way.

Arick says: FEBRUARY 8, 2020 AT 8:32 PM

Rip everyone that is counting on ota updates from cellphone providers.

Reply

Ben Siron says: FEBRUARY 9, 2020 AT 12:53 AM

I did a bit of searching to see what permissions the Bluetooth process has, since if an attacker is allowed to execute code running as this process, this would define what a potential attacker is able to do.

application-permissions-mean

```
android.permission.ACCESS_BLUETOOTH_SHARE
android.permission.ACCESS_COARSE_LOCATION
android.permission.ACCESS_NETWORK_STATE
android.permission.BLUETOOTH
android.permission.BLUETOOTH_ADMIN
android.permission.BLUETOOTH_MAP
android.permission.BLUETOOTH_PRIVILEGED
android.permission.BLUETOOTH_STACK
android.permission.CALL_PRIVILEGED
android.permission.CHANGE_NETWORK_STATE
android.permission.CONNECTIVITY_INTERNAL
android.permission.DEVICE_POWER
android.permission.DUMP
android.permission.GET_ACCOUNTS
android.permission.INTERACT_ACROSS_USERS_FULL
android.permission.INTERACT_ACROSS_USERS
android.permission.INTERNET
android.permission.LISTEN_ALWAYS_REPORTED_SIGNAL_STRENGTH
android.permission.MANAGE_APP_OPS_MODES
android.permission.MANAGE_USERS
android.permission.MEDIA_CONTENT_CONTROL
android.permission.MODIFY_AUDIO_ROUTING
android.permission.MODIFY_AUDIO_SETTINGS
android.permission.MODIFY_PHONE_STATE
android.permission.NET_ADMIN
android.permission.NET_TUNNELING
android.permission.NETWORK_FACTORY
android.permission.NFC_HANDOVER_STATUS
android.permission.PACKAGE_USAGE_STATS
android.permission.READ_CALL_LOG
android.permission.READ_CONTACTS
android.permission.READ_EXTERNAL_STORAGE
android.permission.READ_PRIVILEGED_PHONE_STATE
android.permission.READ_PROFILE
android.permission.READ_SMS
android.permission.REAL_GET_TASKS
android.permission.RECEIVE_BOOT_COMPLETED
android.permission.RECEIVE_SMS
android.permission.SEND_SMS
android.permission.TETHER_PRIVILEGED
android.permission.UPDATE_APP_OPS_STATS
android.permission.UPDATE_DEVICE_STATS
android.permission.VIBRATE
android.permission.WAKE_LOCK
android.permission.WHITELIST_BLUETOOTH_DEVICE
android.permission.WRITE_APN_SETTINGS
android.permission.WRITE_CALL_LOG
android.permission.WRITE_CONTACTS
android.permission.WRITE_EXTERNAL_STORAGE
android.permission.WRITE_SECURE_SETTINGS
```

Some of these permissions (while not root level access) are fairly powerful. Definitely not the kinds of things I would want to allow an attacker to do on my phone. And definitely not the kinds of things that I would want to see many phones doing simultaneously due to a wormable infection.

android.permission.DEVICE_POWER
Allows low-level access to power management

android.permission.GET_ACCOUNTS
Allows access to the list of accounts in the Accounts Service.

android.permission.MANAGE_USERS
Allows an application to call APIs that allow it to query and manage users on the device. Allows apps to manage users on the device, including query, creation and deletion. This permission is not available to third party applications. On Android, most apps/packages run under their own separate 'user' account, identified by a userId / UID / appId / AID. Therefore, this privilege allows management of other packages / applications.

android.permission.RECEIVE_SMS
android.permission.SEND_SMS
android.permission.WRITE_SMS
Allows an application to read, send, or modify SMS messages. Malicious apps may delete your messages.

android.permission.READ_CALL_LOG
android.permission.WRITE_CALL_LOG
Allows an application to read, write the user's call log data.

android.permission.REAL_GET_TASKS
https://stackoverflow.com/questions/27974583/get-tasks-permission-deprecated
allows querying for all running tasks on the system

android.permission.WAKE_LOCK
Allows using PowerManager WakeLocks to keep processor from sleeping or screen from dimming.

android.permission.CALL_PRIVILEGED
Allows an application to call any phone number, including emergency numbers, without going through the Dialer user interface for the user to confirm the call being placed.

Hopefully android phone manufacturers will issue security updates to include this fix, but the track record so far is not encouraging. I'm turning off android bluetooth for the meantime.

Reply

Jan Ruge says: FEBRUARY 9, 2020 AT 1:41 PM

Thanks for the detailed Information, great job!

Reply

Andrew says: FEBRUARY 9, 2020 AT 3:34 AM

So is it possible to gain temporary root access in any way?

Reply

required.

Reply

Filip Piasecki says: FEBRUARY 10, 2020 AT 9:00 PM

Like, most people won't ever see any patches from their Vendors. It just wont happen. No body cares. My V30+ from M1 or some other operator has the patches from May 2019. And nothing I could do.

Reply

Jan Ruge says: FEBRUARY 11, 2020 AT 10:35 AM

This is definitely a problem. I saw a great discussion on this topic [1] at the weekend. Sadly, besides flashing a custom ROM, there is not much more you can do. Hopefully customer complains will change the situation some day. .As far I can see the V30 is still sold?!

[1] https://news.ycombinator.com/item?id=22264553

P.S. This work was done on a perfectly fine laptop from 2013

Reply

Android User says: FEBRUARY 11, 2020 AT 10:51 PM

Do you and the attacker need to be on the same internet connection or is this just strictly Bluetooth connection?

Reply

Jan Ruge says: FEBRUARY 12, 2020 AT 1:29 PM

For convenience we choose a reverse-shell over network, so you can connect back to any server on the internet. It should also be possible to bring the daemon back to an operational state and create a reverse shell over Bluetooth.

Reply

Ivan says: FEBRUARY 16, 2020 AT 5:45 PM

How to check old phone stopped on Google patch Jan 2019?

Reply

Junming Chen says: FEBRUARY 20, 2020 AT 9:36 AM

Hello, we are the OEM of the automobiles. We have products in the range of affected Android version but we are unclear if out product is actually affected. Is there a way to get a copy of poc so we can start verify and fix this vulnerability before the poc release to the public? Thanks

Reply

Chris Amendola says: MARCH 7, 2020 AT 9:58 PM people use that Bluetooth interface for hands-free which means they'd be enabled and active on a lot of vehicles. Wondering as I'm trying to get a copy of the firmware for my vehicle.

Reply

Jan Ruge says: MARCH 11, 2020 AT 11:10 AM

From my point of view, the tablet/phone patch process is already horrific. Do you have more information or references on that?

Reply

Jacob C says: MARCH 9, 2020 AT 10:59 PM

Can you provide a method for testing this vuln? I am trying to verify it across my devices, but I am not able to reproduce.

Reply

Jan Ruge says: MARCH 11, 2020 AT 11:08 AM

In general, if you are running a Android device Android 9 and earlier and with a patch Level prior Feb 2020, you are probably vulnerable. We do not want to release PoC code to a limited set of people and keep the disclosure process transparent.

Reply

Jacob C says: MARCH 19, 2020 AT 3:19 AM

Anyone recommend any good proof of concept code/scripts? I need to verify we fixed it in our older android build we are supporting.

Reply

AndroidPie User says: APRIL 24, 2020 AT 10:22 AM

How does this problem relate to the heavily discussed Corona tracing apps, that are all based on BT? What you say above is, do not active BT without having installed patch level February 2020, but most phones will never see that (e.g. my phone, where Sony has stopped updates in Sep 2019). That limits the amount of Android phones, that safely can use the tracing apps (not talking about privacy issues here!) to maybe 10-15% (the phones that still get updates).

Reply

Jan Ruge says: APRIL 24, 2020 AT 9:21 PM

First of all, luckily this Bug has been fixed before the Corona thing. We have verified the patch using the PoC and our fuzzer. We can not observe any more crashes. Therefore for updated devices, the risk is mitigated.

Protocol interaction always comes with the risk of vulnerabilities. This does not only apply to Bluetooth but also WiFi, NFC, Baseband, web browsers, etc. Therefore having an

If you do not receive any patches and want to stay secure in the long run, you might want to install an up to date custom ROM. Even though, this solution is not for everyone possible.

I did not look into any corona tracking app in detail. We provided our PoC, so the people designing those apps can estimate the risk, which depends on the implementation. From a theoretical point of view, you do not need protocol interaction (e.g. active connection) as you are only interested in the presence of nearby devices. Theoretical this can be securely implemented with this vulnerability in mind, but I'm not sure of the current support of those features by the Operating System.

Gus says: MARCH 16, 2020 AT 12:50 PM

On Android 8-9, there is a parameter for high accuracy location using Bluetooth even when Bluetooth is off. Should this be disabled also? Thanks in advance for advice.

Reply

Jacob says: MARCH 22, 2020 AT 5:31 AM

Are you able to see data even if the devices is locked? Like on the lockscreen?

Reply

John says: APRIL 10, 2020 AT 6:42 AM

Well, you should be able to show the PiC its been past the accrued time you mentioned required before publishing…

Reply

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Website

POST COMMENT