Catégories

[Hardware]
[Outils]
[Redteam]
[Vulnerability]

Balises

[Hardware] [Outils] [Redteam] [Vulnerability]

Derniers articles

[EN] Cobbler <= 3.2.1 multiple vulnerabilities leading to RCE as root
20 Sep 2021

[EN] Responsible Disclosure - Gaining root access on Sonos Play (1st gen and 2nd gen 'One') Speakers
09 Aug 2021

[EN] Ligolo-ng : Tunneling like a VPN
28 Jul 2021

[EN] Responsible Disclosure - Gaining root access on Sonos Play (1st gen and 2nd gen 'One') Speakers

SONOS

By **Laboratoire TNP** / on **09 Aug, 2021**

Introduction

Sonos is a wireless home sound system, allowing multiroom music playing.

In 2020, we discovered a DMA (Direct Memory Access) vulnerability on Sonos Play speakers (1st gen and 2nd gen "One"). We worked with Sonos in order to address the security vulnerability and were credited on the Security Researcher Recognition page.

> We are aware that Synacktiv publicly disclosed the vulnerability on the Sonos One the 03/2021. After discussion with Sonos, they were not aware of this publication. We preferred to follow a responsible disclosure process.

Vulnerability

Attack on Sonos Play 1 (First gen)

We performed our first tests on the Sonos Play 1 speaker. Multiple papers are talking about some security issues on the web interface of Sonos.

However, we decided to take a look at the hardware part of the speakers to see if debugging interfaces were available.

After opening the Play 1 speaker, something immediately focused our attention: a *Mini PCI-Express (mPCIE)* Wireless card.

We decided to perform a DMA Attack on the Sonos speaker using PCILeech and the PLX USB3380 development board in order to read the memory content.

DMA Attack on Sonos Play Speaker

A web interface was available on TCP port 1400, it can be used to get some information about the system, and outputs some command outputs (from the *brctl* command) at the *status/showstp* page.

We searched for strings in the `pcileech` dump matching *brctl*, and found */usr/sbin/brctl showstp br0*. After some tests, we confirmed that this string is used by the *status/showstp* page.

We came up with the following PCILeech signature that will replace the *brctl* command with one of our choice:

```
*,2f7573722f7362696e2f627263746c2073686f7773747020627230,0,-,r0,2f62696e2f62757379626f7820756e616d65202d61000000000000
```

- `*` means to search into each memory pages.
- `2f7573722f7362696e2f627263746c2073686f7773747020627230` is the `/usr/sbin/brctl showstp br0` string hex-encoded.
- `0` and `-` means that we don't have to look for another chunk.
- `r0` is the relative offset where the patch should be applied.
- `2f62696e2f62757379626f7820756e616d65202d61000000000000` is the patch that will replace the `brctl` command with the command of our choice (here, `/bin/busybox uname -a`).

With the signature file at hand, we just needed to run the following command `pcileech patch -sig sonos_uname`, and browse the *status/showstp* page to gain arbitrary code execution.

```
<?xml version="1.0" ?>
<?xml-stylesheet type="text/xsl" href="/xml/review.xsl"?><ZPSupportInfo><Command cmdline='/bin/busybox ls -lahR /'>/:
-rw-rw-r--    1 root     root          11 Jan  1 00:00 LEGACYCOMPATVER
-rw-rw-r--    1 root     root          11 Jan  1 00:00 MINCOMPATVER
-rw-rw-r--    1 root     root          11 Jan  1 00:00 VERSION
drwxrwxr-x    1 root     root         916 Jan  1 00:00 bin
-rw-rw-r--    1 root     root         241 Jan  1 00:00 build.properties
drwxrwxr-x    1 root     root        2.6K Jan  1 00:00 dev
drwxrwxr-x    1 root     root         656 Jan  1 00:00 etc
drwxrwxr-x    1 root     root           0 Jan  1 00:00 home
drwxr-xr-x    7 root     root           0 Jan  1 00:00 jffs
drwxrwxr-x    1 root     root        2.0K Jan  1 00:00 lib
drwxrwxr-x    1 root     root           0 Jan  1 00:00 mnt
drwxrwxr-x    1 root     root          44 Jan  1 00:00 modules
drwxrwxr-x    1 root     root         184 Jan  1 00:00 opt
dr-xr-xr-x   42 root     root           0 Jan  1 00:00 proc
drwxr-xr-x    6 root     root           0 Jan  1 00:00 ramdisk
drwxrwxr-x    1 root     root         300 Jan  1 00:00 sbin
lrwxrwxrwx    1 root     root          12 Jan  1 00:00 tmp -<![CDATA[>]]> /ramdisk/tmp
drwxrwxr-x    1 root     root          32 Jan  1 00:00 usr
lrwxrwxrwx    1 root     root          12 Jan  1 00:00 var -<![CDATA[>]]> /ramdisk/var
drwxrwxr-x    1 root     root         236 Jan  1 00:00 wifi

/bin:
```

showstp page executing the command of our choice

After browsing the filesystem content, we found a way to enable a debugging feature that will execute a `telnetd` process (as root) at startup. We then gained unrestricted access to the Sonos Play 1 first generation speaker system.

Attack on Sonos Play One (Second gen)

After our research on Sonos Play 1, Sonos released a new speaker: the Sonos Play One.

It features Amazon Alexa integration (so a microphone), and a brand-new ARM Architecture.

We bought it, and immediately opened it: the miniPCIe card was still here.

We ran the same attack that the first generation speaker, but it seems that additional protections were applied.

DMA Attack on the Sonos One SL

PCILeech isn't able to dump the memory content. We tried multiple time, and we managed to dump only a few kilobyte of memory.

We analyzed the memory content, and discovered that it was a *U-Boot* memory dump. We found some logs checking for the vendor ID of the MiniPCIe card. Maybe a protection in order to not change the PCIe card with another vendor, maybe a security feature.

Spoofing the PCI Vendor ID

When you plug a PCILeech patched PLX USB3380 card on the target computer, the USB3380 is recognized as a Memory Card Reader (PCI ID: 16BC14E4).

We looked at how the PCILeech PLX USB3380 firmware is made, and we found the following content in the `usb3380_flash/linux/pcileech_flash.c` file:

```
static const unsigned char g_firmware_pcileech[] = {
   0x5a, 0x00, 0x2a, 0x00, 0x23, 0x10, 0x49, 0x38, 0x00, 0x00, 0x00, 0x00, 0xe4, 0x14, 0xbc, 0x16,
   0xc8, 0x10, 0x02, 0x06, 0x04, 0x00, 0xd0, 0x10, 0x84, 0x06, 0x04, 0x00, 0xd8, 0x10, 0x86, 0x06,
   0x04, 0x00, 0xe0, 0x10, 0x88, 0x06, 0x04, 0x00, 0x21, 0x10, 0xd1, 0x18, 0x01, 0x90, 0x00, 0x00 };
```

If you search for the PCI Vendor ID *16BC14E4*, you can find it in the firmware code *0xe4, 0x14, 0xbc, 0x16*.

We modified some parts of the PCILeech flash program to flash the USB3380 card with the vendor ID of the Wireless card.

We plugged our patch USB3380 card on another Linux computer, and confirmed that the USB33800 card as the same vendor ID as the Sonos PCIe Wi-Fi card.

We tested the `pcileech dump` command against the Sonos One speaker, and we were now able to dump the memory content. We then performed the same attack as the first generation speakers, and gained root access on the device.

Research Conclusion

The Play One system was hardened against some attacks: the root filesystem is mounted as read-only, all other filesystems are mounted with the *nodev* and *noexec* options.

If you tried to perform a remount of a filesystem (for example remount / as read-write), the kernel will enforce it as * nodev* and *noexec*.

We didn't have this behavior on the Sonos 1 (first gen) speaker.

Because our attack was performed on memory, we did not alter the filesystem content. We even managed to drop a *meterpreter shell* by patching the memory.

Fixing the vulnerabilities may be difficult because hardware-specific protections must be enabled (like [System Memory Management Unit - SMMU]() for ARM units).

We would like to thank the Sonos Team for the excellent work they have done handling this vulnerability.

Timeline (dd/mm/yyyy)

- 03/01/2020: Initial contact with security@sonos.com
- 03/01/2020: Advisory sent to Sonos
- 07/01/2020: Reply from Sonos that they are currently investigating the issue
- 28/01/2020: Sonos informs us that they are close to a solution for the ARM architecture (NXP iMX6)
- 11/02/2020: Video meeting with the Sonos Security Team
- 19/02/2020: CVE-2020-9285 assigned by MITRE
- 20/02/2020: Sonos informs us that they have a working solution, but need more testing
- Coronavirus Crisis
- 05/06/2020: Sonos send us a Sonos One SL for testing
- 05/08/2020: Exploit reproduced with Sonos One SL
- 06/08/2020: Sonos provide us custom tool and a custom firmware that should fix the security vulnerability
- 11/08/2020: The Sonos patch is working. However, we were able to dump memory during U-Boot initialization
- 12/08/2020: Sonos share technical patch details with us
- 14/08/2020: Sonos send us the source of the Kernel patch that fix the issue
- 19/08/2020: Sonos send us a new firmware fixing the U-Boot issue with the kernel patch
- 25/08/2020: We confirm that all the issues have been resolved
- 21/12/2020: Sonos inform us that the patch has a significant performance impact and can't be incorporated into products that use the NXP iMX6 SoC
- 04/02/2021: Sonos ask us about disclosure plans
- 23/02/2021: Expected publication date sent to Sonos
- 29/07/2021: Article sent to Sonos for validation
- 02/08/2021: Sonos authorizes the publication of the article
- 09/08/2021: Disclosure by TNP IT Security

Credits

- Nicolas Chatelain : nicolas.chatelain -at- tnpconsultants.com