

**IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE**

ANCORA TECHNOLOGIES, INC.,

Plaintiff,

v.

SONY MOBILE COMMUNICATIONS AB,
SONY MOBILE COMMUNICATIONS, INC.,
AND SONY MOBILE COMMUNICATIONS
(USA) INC.,

Defendants.

Civil Action No. 19-1703 (CFC)

JURY TRIAL DEMANDED

**FIRST AMENDED COMPLAINT FOR PATENT
INFRINGEMENT**

Plaintiff, ANCORA TECHNOLOGIES, INC. (“Ancora”), for its amended Complaint against Sony Mobile Communications (USA), Inc., Sony Mobile Communications, Inc., and Sony Mobile Communications AB (collectively “Sony” or “Defendants”) states the following:

I. THE PARTIES

1. Plaintiff Ancora Technologies, Inc. is a corporation organized and existing under the laws of the State of Delaware and having a place of business at 23977 S.E. 10th Street, Sammamish, Washington 98075.

2. Upon information and belief, Sony Mobile Communications, Inc. is a Japanese corporation that is a wholly-owned subsidiary of Sony Corporation with a principal place of business located at 4-12-3 Higashi-shinagawa, Shinagawa-ku, Tokyo, 140-0002, Japan. Upon information and belief, Sony Mobile Communications, Inc. offers for sale mobile devices, such as smartphones, through its website at <https://www.sonymobile.com/us/products/phones/>. Upon information and belief, Sony Mobile Communications, Inc. offers for sale mobile devices, such as smartphones, through an Amazon storefront website at <https://tinyurl.com/Sony-Amazon-Store>.

Upon information and belief, Sony Mobile Communications, Inc. directs sales of its mobile devices to Delaware residents through at least these websites.

3. Upon information and belief, Sony Mobile Communications AB is also a wholly-owned subsidiary of Sony Corporation and is incorporated under the laws of Sweden with its principal place of business at Nya Vattentornet SE-221, 88 Lund, Sweden. Upon information and belief, Sony Mobile Communications AB offers for sale mobile devices, such as smartphones, through its website at <https://www.sonymobile.com/us/products/phones/>. Upon information and belief, Sony Mobile Communications AB offers for sale mobile devices, such as smartphones, through an Amazon storefront website at <https://tinyurl.com/Sony-Amazon-Store>. Upon information and belief, Sony Mobile Communications AB directs sales of its mobile devices to Delaware residents through at least these websites.

4. Upon information and belief, Sony Mobile Communications (USA), Inc. is a wholly-owned subsidiary of Sony Mobile Communications AB and is incorporated under the laws of the State of Delaware with its principal place of business located at 2207 Bridgepointe Pkwy, San Mateo, CA 94404. Sony Mobile Communications (USA), Inc. may be served via its registered agent, Capitol Services, Inc., 1675 S. State St., Ste. B, Dover, DE, 19901.

5. Upon information and belief, Sony is in the business of supplying mobile devices, such as smartphones, to its customers in the United States, including within this District.

II. JURISDICTION

6. This is an action for patent infringement arising under the provisions of the Patent Laws of the United States of America, Title 35, United States Code, namely 35 U.S.C. §§ 271, 281, and 284-285, among others.

7. This Court has subject matter jurisdiction pursuant to 28 U.S.C. §§ 1331 and 1338(a).

8. Venue is proper in this District as to Sony Mobile Communications AB because they are foreign entities that may be sued in any judicial district under 28 U.S.C. § 1391(c).

9. Venue is proper in this District as to Sony Mobile Communications (USA) Inc. under 28 U.S.C. § 1400(b) because it was formed under the laws of the State of Delaware and, therefore, resides in this District.

10. Sony is subject to personal jurisdiction pursuant to due process due at least to its substantial business in this State, including: (A) at least part of its infringing activities alleged herein; and (B) regularly doing or soliciting business, engaging in other persistent conduct, and/or deriving substantial revenue from goods sold and services provided to Delaware residents. Sony has conducted and regularly conducts business within the United States and this District. Sony has purposefully availed itself of the privileges of conducting business in the United States, and more specifically in Delaware and this District. Sony has sought protection and benefit from the laws of the State of Delaware by placing infringing products into the stream of commerce through an established distribution channel with the awareness and/or intent that they will be purchased by consumers in this District.

11. On information and belief, Sony has significant ties to, and presence in, this District, making venue in this District both proper and convenient for this action.

III. BACKGROUND

12. On June 25, 2002, U.S. Patent No. 6,411,941 (“the ’941 patent”) entitled “Method Of Restricting Software Operation Within A License Limitation” was duly and legally issued. (*See* Exhibit A, U.S. Patent No. 6,411,941.) A reexamination certificate also issued to the ’941 patent on June 1, 2010 where the patentability of all claims was confirmed by the United States Patent Office. (Exhibit B, *Ex Parte* Reexamination Certificate Issued Under 35 U.S.C. § 307.)

13. The ’941 patent has been involved in litigation against Microsoft Corporation, Dell Incorporated, Hewlett Packard Incorporated, and Toshiba America Information Systems. (*See* 2009-cv-00270, Western District of Washington.)

14. The ’941 patent has also been involved in litigation against Apple Incorporated. (*See* 2015-cv-03659, Northern District of California.)

15. The '941 patent is currently involved in litigation against HTC America, Inc. and HTC Corporation. (*See* 2016-cv-01919, Western District of Washington.)

16. The '941 patent is currently involved in litigation against Samsung Electronics America, Inc. and Samsung Electronics Co., Ltd. (*See* 2019-cv-00385, Western District of Texas.)

17. The '941 patent is currently involved in litigation against LG Electronics USA, Inc. and LG Electronics, Inc. (*See* 2019-cv-00384, Western District of Texas.)

18. The '941 patent was involved in a Covered Business Method proceeding before the U.S. Patent and Trademark Office (*See* PTAB-CBM2017-00054). The U.S. Patent and Trademark Office denied institution of the petition filed by HTC and found the '941 patent recites a “technological improvement to problems arising in prior art software and hardware methods of restricting an unauthorized software program’s operation.” (*See* PTAB-CBM2017-00054, Paper No. 7 at pg. 9.)

19. The U.S. Court of Appeals for the Federal Circuit further issued an order on November 16, 2018 regarding the validity of the '941 patent. (*See* CAFC 18-1404, Dkt. # 39.) In this appeal, the U.S. Court of Appeals for the Federal Circuit held:

[T]he claimed invention moves a software-verification structure to a BIOS location not previously used for this computer-security purpose and alters how the function is performed (in that the BIOS memory used for verification now interacts with distinct computer memory to perform a software-verification function), yielding a tangible technological benefit (by making the claimed system less susceptible to hacking).

CAFC 18-1404, Dkt. # 39, pg. 13.

20. The U.S. Court of Appeals for the Federal Circuit further issued an order on March 3, 2014 regarding claim construction and invalidity of the '941 patent. (*See* CAFC 13-1378, Dkt. # 57.)

21. Ancora is the owner of all right, title and interest in the '941 patent.

IV. COUNT I – PATENT INFRINGEMENT

22. Ancora realleges the preceding paragraphs as though set forth fully herein.

23. Claim 1 of the '941 patent recites “a method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area; the method comprising the steps of: [1] selecting a program residing in the volatile memory, [2] using an agent to set up a verification structure in the erasable, non-volatile memory of the BIOS, the verification structure accommodating data that includes at least one license record, [3] verifying the program using at least the verification structure from the erasable non-volatile memory of the BIOS, and [4] acting on the program according to the verification.”

24. As explained in detail below, Sony directly infringed the '941 patent in violation of 35 U.S.C. § 271(a) by, prior to the expiration of the '941 patent, using within the United States, and without authorization, the method recited in at least Claim 1 of the '941 patent literally or under the doctrine of equivalents.

25. Sony designs at least the following smartphones to use the method recited in Claim 1: Xperia X, Xperia XZ, Xperia XA, Xperia XA1, Xperia XZ1, Xperia XZ2, Xperia E5, and Xperia XA Ultra. (*see e.g.*, <https://tinyurl.com/y9njhm24>.) (collectively, “Sony Android Devices”)

26. On information and belief, Sony has provided one or more OTA updates for the Sony Android Devices on or about December 16, 2016, April 25, 2017, and June 15, 2017. (*See e.g.*, <https://www.androidauthority.com/android-7-0-update-679175/>.)

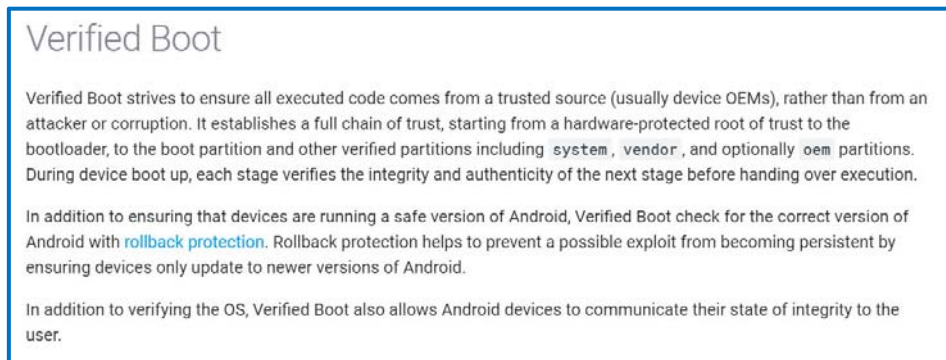
27. The Sony Android Devices include operating system software that is transmitted by Sony or received under Sony’s direction using over-the-air (“OTA”) servers and hardware (“the

OTA Products”) that cause the Sony Android Devices to perform the method of claim 1 prior to the expiration of the ‘941 Patent.

28. The following comparison between the limitations of Claim 1 of the ‘941 patent and Sony’s Over-the-Air update process (the “Accused Process”) used to update the Sony Android Devices establishes Sony’s infringement of the ‘941 patent.

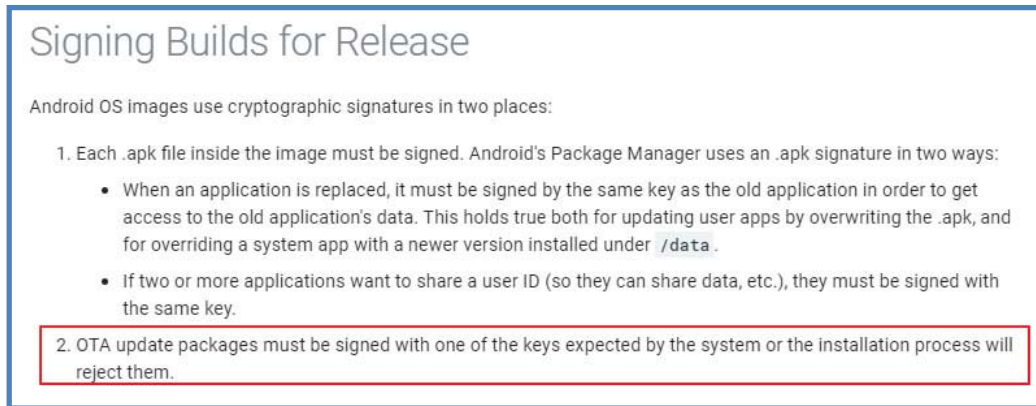
“A method of restricting software operation within a license for use with a computer including an erasable, non-volatile memory area of a BIOS of the computer, and a volatile memory area;”

29. The Accused Process is a method of restricting software operation within a license because, if the “Verified Boot” aspect of the Accused Process fails, the OTA update will not complete and the updated software will not execute.



<https://source.android.com/security/verifiedboot>

30. If the operating system update image is not cryptographically signed with the expected cryptographic keys, the update process Sony uses will reject the update:



The screenshot shows a document titled "Signing Builds for Release" with the following content:

Android OS images use cryptographic signatures in two places:

1. Each .apk file inside the image must be signed. Android's Package Manager uses an .apk signature in two ways:
 - When an application is replaced, it must be signed by the same key as the old application in order to get access to the old application's data. This holds true both for updating user apps by overwriting the .apk, and for overriding a system app with a newer version installed under /data .
 - If two or more applications want to share a user ID (so they can share data, etc.), they must be signed with the same key.
2. OTA update packages must be signed with one of the keys expected by the system or the installation process will reject them.

The second point is enclosed in a red rectangular box.

https://source.android.com/devices/tech/ota/sign_builds

31. Each Smartphone used with the Accused Process includes a computer having a non-volatile memory area of a BIOS (also referred to as Unified Extensible Firmware Interface (UEFI)) in the form of ROM or Flash memory (also described as “RAM disk”) and volatile memory in the form of RAM memory. The BIOS included within each Smartphone comprises data that is maintained when the power is removed and contains the set of essential startup operations that run when a computer is turned on, which tests hardware, starts the operating system, and supports the transfer of data among hardware devices of the computer.

“the method comprising the steps of: selecting a program residing in the volatile memory,”

32. The Accused Process loads at least a portion of the updated operating system program image into the Smartphone’s RAM (volatile memory) and selects the program for execution.

Life of an OTA update

A typical OTA update contains the following steps:

1. Device performs regular check in with OTA servers and is notified of the availability of an update, including the URL of the update package and a description string to show the user.
2. Update downloads to a cache or data partition, and its cryptographic signature is verified against the certificates in `/system/etc/security/otacerts.zip`. User is prompted to install the update.
3. Device reboots into recovery mode, in which the kernel and system in the recovery partition are booted instead of the kernel in the boot partition.
4. Recovery binary is started by init. It finds command-line arguments in `/cache/recovery/command` that point it to the downloaded package.
5. Recovery verifies the cryptographic signature of the package against the public keys in `/res/keys` (part of the RAM disk contained in the recovery partition).
6. Data is pulled from the package and used to update the boot, system, and/or vendor partitions as necessary. One of the new files left on the system partition contains the contents of the new recovery partition.
7. Device reboots normally.
 - a. The newly updated boot partition is loaded, and it mounts and starts executing binaries in the newly updated system partition.
 - b. As part of normal startup, the system checks the contents of the recovery partition against the desired contents (which were previously stored as a file in `/system`). They are different, so the recovery partition is reflashed with the desired contents. (On subsequent boots, the recovery partition already contains the new contents, so no reflash is necessary.)

The system update is complete! The update logs can be found in `/cache/recovery/last_log.#`.

<https://source.android.com/devices/tech/ota/nonab>

“using an agent to set up a verification structure in the erasable, non-volatile memory of the BIOS, the verification structure accommodating data that includes at least one license record,”

33. The Accused Process uses an agent to set up a verification structure in the erasable, non-volatile memory of the BIOS of the Sony Android Devices. For example, Sony implements an OTA Install program or subroutine that provides to the Sony Android Devices an OTA update containing a verification structure. The OTA Install program or subroutine also stores a verification structure within a partition (*e.g.*, the “cache” or “A/B” partitions) of the erasable, non-volatile memory of the Smartphone BIOS.

34. The verification structure includes data accommodating at least one license record. Examples of such a license record Sony uses in the Accused Process include a cryptographic signature or key:

Signing Builds for Release

Android OS images use cryptographic signatures in two places:

1. Each .apk file inside the image must be signed. Android's Package Manager uses an .apk signature in two ways:
 - When an application is replaced, it must be signed by the same key as the old application in order to get access to the old application's data. This holds true both for updating user apps by overwriting the .apk, and for overriding a system app with a newer version installed under /data .
 - If two or more applications want to share a user ID (so they can share data, etc.), they must be signed with the same key.
2. OTA update packages must be signed with one of the keys expected by the system or the installation process will reject them.

https://source.android.com/devices/tech/ota/sign_builds

Verifying Boot

Verified boot requires cryptographically verifying all executable code and data that is part of the Android version being booted before it is used. This includes the kernel (loaded from the `boot` partition), the device tree (loaded from the `dtbo` partition), `system` partition, `vendor` partition, and so on.

Small partitions, such as `boot` and `dtbo`, that are read only once are typically verified by loading the entire contents into memory and then calculating its hash. This calculated hash value is then compared to the *expected hash value*. If the value doesn't match, Android won't load. For more details, see [Boot Flow](#).

Larger partitions that won't fit into memory (such as, file systems) may use a hash tree where verification is a continuous process happening as data is loaded into memory. In this case, the root hash of the hash tree is calculated during run time and is checked against the *expected root hash value*. Android includes the `dm-verity driver` to verify larger partitions. If at some point the calculated root hash doesn't match the *expected root hash value*, the data is not used and Android enters an error state. For more details, see [dm-verity corruption](#).

The *expected hashes* are typically stored at either the end or beginning of each verified partition, in a dedicated partition, or both. Crucially, these hashes are signed (either directly or indirectly) by the root of trust. As an example, the AVB implementation supports both approaches, see [Android Verified Boot](#) for details.

<https://source.android.com/security/verifiedboot/verified-boot>

“verifying the program using at least the verification structure from the erasable non-volatile memory of the BIOS, and”

35. Sony uses the Accused Process to confirm whether the operating system update is licensed using at least the verification structure from the erasable, non-volatile memory of the

Smartphone BIOS. For instance, once the verification structure has been set up in the BIOS, Sony uses the Accused Process to reboot into recovery mode, load the OTA update into its volatile memory (e.g., RAM), and use the at least one license record from the BIOS to verify the OTA update (e.g., Step 5 below):

Life of an OTA update

A typical OTA update contains the following steps:

1. Device performs regular check in with OTA servers and is notified of the availability of an update, including the URL of the update package and a description string to show the user.
2. Update downloads to a cache or data partition, and its cryptographic signature is verified against the certificates in `/system/etc/security/otacerts.zip`. User is prompted to install the update.
3. Device reboots into recovery mode, in which the kernel and system in the recovery partition are booted instead of the kernel in the boot partition.
4. Recovery binary is started by init. It finds command-line arguments in `/cache/recovery/command` that point it to the downloaded package.
5. Recovery verifies the cryptographic signature of the package against the public keys in `/res/keys` (part of the RAM disk contained in the recovery partition).
6. Data is pulled from the package and used to update the boot, system, and/or vendor partitions as necessary. One of the new files left on the system partition contains the contents of the new recovery partition.
7. Device reboots normally.
 - a. The newly updated boot partition is loaded, and it mounts and starts executing binaries in the newly updated system partition.
 - b. As part of normal startup, the system checks the contents of the recovery partition against the desired contents (which were previously stored as a file in `/system`). They are different, so the recovery partition is reflashed with the desired contents. (On subsequent boots, the recovery partition already contains the new contents, so no reflash is necessary.)

The system update is complete! The update logs can be found in `/cache/recovery/last_log.#`.

<https://source.android.com/devices/tech/ota/nonab>

“acting on the program according to the verification.”

36. The Accused Process acts on the program (the operating system update) according to the verification. If the OTA update is verified, the Accused Process will load and execute the update (e.g., Steps 6 and 7 below):

Life of an OTA update

A typical OTA update contains the following steps:

1. Device performs regular check in with OTA servers and is notified of the availability of an update, including the URL of the update package and a description string to show the user.
2. Update downloads to a cache or data partition, and its cryptographic signature is verified against the certificates in `/system/etc/security/otacerts.zip`. User is prompted to install the update.
3. Device reboots into recovery mode, in which the kernel and system in the recovery partition are booted instead of the kernel in the boot partition.
4. Recovery binary is started by init. It finds command-line arguments in `/cache/recovery/command` that point it to the downloaded package.
5. Recovery verifies the cryptographic signature of the package against the public keys in `/res/keys` (part of the RAM disk contained in the recovery partition).
6. Data is pulled from the package and used to update the boot, system, and/or vendor partitions as necessary. One of the new files left on the system partition contains the contents of the new recovery partition.
7. Device reboots normally.
 - a. The newly updated boot partition is loaded, and it mounts and starts executing binaries in the newly updated system partition.
 - b. As part of normal startup, the system checks the contents of the recovery partition against the desired contents (which were previously stored as a file in `/system`). They are different, so the recovery partition is reflashed with the desired contents. (On subsequent boots, the recovery partition already contains the new contents, so no reflash is necessary.)

The system update is complete! The update logs can be found in `/cache/recovery/last_log.#`.

<https://source.android.com/devices/tech/ota/nonab>

37. If the verification fails, however, the update is rejected:

Signing Builds for Release

Android OS images use cryptographic signatures in two places:

1. Each `.apk` file inside the image must be signed. Android's Package Manager uses an `.apk` signature in two ways:
 - When an application is replaced, it must be signed by the same key as the old application in order to get access to the old application's data. This holds true both for updating user apps by overwriting the `.apk`, and for overriding a system app with a newer version installed under `/data`.
 - If two or more applications want to share a user ID (so they can share data, etc.), they must be signed with the same key.
2. OTA update packages must be signed with one of the keys expected by the system or the installation process will reject them.

https://source.android.com/devices/tech/ota/sign_builds

38. Sony uses and controls the use of the Accused Process to perform OTA software updates on the Sony Android Devices, practicing each limitation of Claim 1 as described above.

Sony directly infringed Claim 1 of the '941 Patent by using the Accused Process with Sony Android Devices by itself.

39. Once Sony has set up the verification structure by transmitting to a device an OTA update, the Accused Process is configured to automatically perform each of the remaining Claim 1 steps.

40. In addition to direct infringement by Sony, Ancora alternatively alleges that Sony jointly infringes the '941 Patent with Smartphone owners in the United States being responsible as a single entity as set forth below.

41. Sony conditions participation in the Accused Process and the receipt of the benefit of a software update on the performance of each of the above steps. For instance, Sony conditions participation by customers in using the Accused Process in order to gain access to new or upgraded Android operating systems.

42. Sony takes steps to ensure that the Accused Process cannot install an OTA update except by performing each of the above described steps.

43. Sony emphasizes the benefits associated with updating the software using the Accused Process. For instance, Sony has stated:

Sony Mobile Security Update Program is an on-going initiative to make regular protective patches available against the latest risks, vulnerabilities, and flaws, including: Android security (released by Google), those from relevant third-party vendors, and any Xperia-specific issues. Depending on timing and situation, security updates may be released separately, as part of firmware maintenance or within a full Android OS upgrade.

(<https://www.xperiainfo.com/us/software-security.html>.)

44. Sony controlled the manner of the performance of the Accused Process. As set forth above, Sony configured each Android Product such that, upon receiving an OTA update, it would automatically perform each remaining step of the Accused Process. For example, using the Accused Process Sony may require immediate installation of the OTA updates onto the Sony Android Devices. (<https://source.android.com/devices/tech/admin/ota-updates>.) Or using the Accused Process, Sony may allow the customer to postpone installation of the OTA update for a specified period.

45. Sony controlled the timing of the performance of the Accused Process by determining when to utilize the Accused Process to set up a verification structure in the Sony Android Devices.

46. Sony had the right and ability to stop or limit infringement by not using the Accused Process but failed to do so.

47. Sony's infringement has caused damage to Ancora, and Ancora is entitled to recover from Sony those damages Ancora has sustained as a result of Sony's infringement.

V. DEMAND FOR RELIEF

WHEREFORE, Plaintiff respectfully requests that this Court enter judgment against Sony as follows:

A. Declaring that Sony has infringed United States Patent No. 6,411,941 in violation of 35 U.S.C. § 271;

B. Awarding damages to Ancora arising out of this infringement, including enhanced damages pursuant to 35 U.S.C. § 284 and prejudgment and post-judgment interest, in an amount according to proof;

C. Awarding Ancora its costs and expenses in this action;

D. Declaring that this case is exceptional, and that Ancora is entitled to its reasonable attorneys' fees pursuant to 35 U.S.C. § 285; and

E. Awarding such other and further relief the Court deems just and proper, including any relief that the Court may deem appropriate under 35 U.S.C. § 285.

VI. DEMAND FOR JURY TRIAL

Ancora respectfully demands a trial by jury of any and all issues triable of right by a jury in this action.

Dated: May __, 2020

SMITH, KATZENSTEIN & JENKINS, LLP

OF COUNSEL:

BROOKS KUSHMAN P.C.

John P. Rondini

Mark A Cantor

John S. LeRoy

Marc Lorelli

1000 Town Center, 22nd Floor

Southfield, Michigan 48075-1238

(248) 358-4400

jrondini@brookskushman.com

mcantor@brookskushman.com

jderoy@brookskushman.com

mlorelli@brookskushman.com

/s/ Eve H. Ormerod

Neal C. Belgam (No. 2721)

Eve H. Ormerod (No. 5369)

1000 West Street, Suite 150

Wilmington, DE 19801

302.652.8400

nbelgam@skjlaw.com

eormerod@skjlaw.com

Attorneys for Plaintiff

Ancora Technologies, Inc.