THEODORE J. BOUTROUS JR., SBN 132099
tboutrous@gibsondunn.com
NICOLA T. HANNA, SBN 130694
nhanna@gibsondunn.com
ERIC D. VANDEVELDE, SBN 240699
evandevelde@gibsondunn.com
GIBSON, DUNN & CRUTCHER LLP
333 South Grand Avenue
Los Angeles, CA 90071-3197
Telephone: 213.229.7000
Facsimile: 213.229.7520

THEODORE B. OLSON, SBN 38137
tolson@gibsondunn.com
1050 Connecticut Avenue, N.W.
Washington, DC 20036-5306
Telephone: 202.955.8500
Facsimile: 202.467.0539

MARC J. ZWILLINGER*
marc@zwillgen.com
JEFFREY G. LANDIS*
jeff@zwillgen.com
ZWILLGEN PLLC
1900 M Street N.W., Suite 250
Washington, D.C. 20036
Telephone: 202.706.5202
Facsimile: 202.706.5298
*Admitted *Pro Hac Vice*

UNITED STATES DISTRICT COURT

CENTRAL DISTRICT OF CALIFORNIA

EASTERN DIVISION

IN THE MATTER OF THE SEARCH OF AN APPLE IPHONE SEIZED DURING THE EXECUTION OF A SEARCH WARRANT ON A BLACK LEXUS IS300, CALIFORNIA LICENSE PLATE 35KGD203

ED No. CM 16-10 (SP)

SUPPLEMENTAL DECLARATION OF ERIK NEUENSCHWANDER IN SUPPORT OF APPLE INC.'S REPLY IN SUPPORT OF MOTION TO VACATE ORDER COMPELLING APPLE INC. TO ASSIST AGENTS IN SEARCH

Hearing:

Date:	March 22, 2016
Time:	1:00 p.m.
Place:	Courtroom 3 or 4
Judge:	Hon. Sheri Pym

I, Erik Neuenschwander, declare:

1. I have personal knowledge of the facts set forth below. If called as a witness, I would and could testify to the statements and facts contained herein, all of which are true and accurate to the best of my knowledge and belief.

2. I have reviewed the Government's Reply in Support of Motion to Compel and Opposition to Apple Inc.'s Motion to Vacate Order, as well as the Declaration of Stacey Perino ("Perino Declaration") and Supplemental Declaration of Christopher Pluhar ("Supplemental Pluhar Declaration") submitted therewith.

3. In this declaration I offer responses to certain statements and assertions made in those materials.

4. Paragraphs 13 through 17 of the Perino Declaration purport to describe Apple's use of key encryption on its devices, relying primarily on language from Apple's iOS Security White Paper. This includes Apple's "Chain of Trust," a process Apple uses to make sure that when a device is powered on, each step of the boot process is checked for any changes that could indicate that the device was tampered with.

5. Mr. Perino notes that as part of this "Chain of Trust" process Apple has created its own certificate authority and public/private key pair used on its devices, and that because only Apple possesses the private key, only Apple can sign system software that can be loaded on its devices during the secure boot process.

6. The fundamental basis of the process Mr. Perino describes is a well-accepted security best-practice. It is sometimes referred to as "Root of Trusts," or "RoTs." The National Institute of Standards and Technology ("NIST") endorsed RoTs as a best practice in its October 2012 Guidelines on Hardware Rooted Security in Mobile Devices, NIST SP 800-164 (Draft) (the "October 2012 NIST Report"). NIST is the entity responsible for developing information security standards and guidelines, including minimum requirements for Federal information systems.

7. The October 2012 NIST Report defined RoTs as "security components" that "provide a set of trusted, security-critical functions," and identified them as "the

foundation of assurance of the trustworthiness of a mobile device." NIST further noted that it "expect[ed] mobile operating systems to utilize the capabilities provided by the RoTs to create and protect device integrity reports, verify and measure firmware and software, and protected locally stored cryptographic keys, authentication credentials, and other sensitive data."

8. The October 2012 NIST Report also cautioned that "[m]any mobile devices are not capable of providing strong security assurances" because they "lack the hardware-based roots of trust that are increasingly built into laptops and other types of hosts."

9. Similarly, the SANS Institute, a major provider of information security and cybersecurity training, noted in its June 2013 Whitepaper "Implementing Hardware Roots of Trust: The Trusted Platform Module Comes of Age," that this hardware-based process better "protect[s] secrets and data that are worth money to cybercriminals (for example, intellectual property and personal financial information)," compared to software-based security, which "is regularly defeated." SANS also wrote in its 2013 Whitepaper that the use of Trusted Platform Modules was "indicative of a strong push coming from defense and intelligence agencies."

10. Many other companies have followed these best practices and recommendations and rely on "chains of trust," "roots of trust," or similar hardware-based programs to provide enhanced security on their devices. Apple is by no means unique in that regard.

11. For example, the organization that develops the Trusted Platform Module ("TPM")—a specific type of hardware-based RoTs—has noted that there are more than a billion PCs, servers, embedded systems, network devices and other devices with TPM or similar functionality embedded in them. ("Trusted Platform Module: A Delayed Reaction?" SC Magazine, Feb. 20, 2013, http://www.scmagazineuk.com/trusted-platform-module-a-delayed-reaction/article/281085/.) Neil Kittelson of the National Security Agency (which has invested heavily in using TPM on its high-

assurance platform), stated that "TPM capabilities represent a shift against today's attackers who are embedding rootkits beneath the notice of software-based security solutions." (*Id.*)

12. Similarly, Microsoft is now including a TPM chip in all of its handheld devices. ("Secure is the New Black: The Evolution of Secure Mobile Technology for Government Agencies," Federal Technology Insider, Jun. 5, 2014, http://www.federaltechnologyinsider.com/secure-new-black-evolution-secure-mobile-technology-government-agencies/.) Even aerospace and defense contractor Boeing has announced an Android-based, high-security mobile device specifically for government agencies, which incorporates "trusted computing architecture," "a TPM chip for securely storing encryption keys," "Secure Boot to maintain the device image integrity," "Hardware Root of Trust [to] ensure[] software authenticity," and a "Hardware Crypto Engine to protect both stored and transmitted data." (*Id.*) While Apple does not use TPM specifically, the Apple security measures discussed in the Perino Declaration provide similar functionality as TPM.

13. The current Protection Profile for Mobile Device Fundamentals ("MDFPP")—a set of security requirements for mobile devices published by the US National Information Assurance Partnership ("NIAP") with the involvement of multiple U.S. government agencies, industry participants, and other organizations as part of the Common Criteria certification program—also encourages hardware secure key storage for a device's Root Encryption Key ("REK"), and protecting sensitive data using a key derived from the REK and a passcode. (*See* "Protection Profile for Mobile Device Fundamentals" at 55, 57, NIAP, Sept. 17, 2014, https://www.niap-ccevs.org /pp/pp_md_v2.0.pdf.) Both of these have been implemented for iOS devices, resulting in certification of iOS 9.2 as MDFPP-compliant. (*See* "Compliant Product – Apple iOS 9," NIAP, https://www.niap-ccevs.org/Product/Compliant.cfm?pid=10695.)

14. Digitally signed software, another key component of Apple's iOS chain of trust anchored by the RoTs described by NIST, are similarly common. As a recent

example, car manufacturer Tesla said that when building a secure connected car, "[t]he first precaution is to ensure that any software updates to the vehicle are authorized by the manufacturer. This can be achieved by using industry standard cryptography technology called 'signing'. Tesla employs this technology. This technology ensures that only Tesla authorized software is applied to the vehicles, even if someone is trying to tamper with the software inappropriately as the software signal transits the network." (*See* "Tesla Motors 4-Point Plan to Build Secure Connected Cars," Evannex, Nov. 19, 2015, http://evannex.com/blogs/news/68988613-tesla-motors-4-point-plan-to-build-secure-connected-cars?rfsn=3664.9c8.)

15. The same practice is common among software developers generally. For instance, Microsoft notes that software "downloaded from the Internet to users' computers can contain programs such as viruses and Trojan horses that are designed to cause malicious damage or provide clandestine network access to intruders," and thus advises Windows software developers to "counter this growing threat" by "digitally sign[ing] the software that you distribute on your intranets or the Internet to ensure its integrity and to assure others that the software can be trusted." (Microsoft TechNet: Digitally Signed Software, https://technet.microsoft.com/en-us/library/cc962053.aspx). Digital signature-based authentication also has a long legacy. For instance, code signing capability for software written in the Java language was added to the official JDK development platform in early 1997. See Gary McGraw & Edward W. Felten, Securing Java (2d ed., 1999) (available at http://www.securingjava.com/chapter-three/).

16. Paragraphs 18 through 24 of the Perino Declaration purport to describe the process by which Apple signs its operating systems. In describing that process, Mr. Perino claims that Apple creates operating systems that "will work only on one specific Apple device." Mr. Perino's inference appears to be that creating GovtOS (which Mr. Perino refers to as the "SIF") would therefore not pose any security risk because it can only be used on the subject device.

17. Mr. Perino's characterization of Apple's process, however, is inaccurate. Apple does not create hundreds of millions of operating systems each tailored to an individual device. Each time Apple releases a new operating system, that operating system is the same for every device of a given model. The operating system then gets a personalized signature specific to each device. This personalization occurs as part of the installation process after the iOS is created.

18. Once GovtOS is created, personalizing it to a new device becomes a simple process. If Apple were forced to create GovtOS for installation on the device at issue in this case, it would likely take only minutes for Apple, or a malicious actor with sufficient access, to perform the necessary engineering work to install it on another device of the same model.

19. Thus, as noted in my initial declaration (ECF No. 16-33), the initial creation of GovtOS itself creates serious ongoing burdens and risks. This includes the risk that if the ability to install GovtOS got into the wrong hands, it would open a significant new avenue of attack, undermining the security protections that Apple has spent years developing to protect its customers.

20. There would also be a burden on the Apple employees responsible for designing and implementing GovtOS. Those employees, if identified, could themselves become targets of retaliation, coercion, or similar threats by bad actors seeking to obtain and use GovtOS for nefarious purposes. I understand that such risks are why intelligence agencies often classify the names and employment of individuals with access to highly sensitive data and information, like GovtOS. The government's dismissive view of the burdens on Apple and its employees seems to ignore these and other practical implications of creating GovtOS.

21. Paragraphs 25 through 28 of the Perino Declaration describe supposedly already existing software that Mr. Perino suggests Apple use as a starting point to create GovtOS. For example, Mr. Perino points to a security exploit that supposedly allowed an iPhone to load a minimal operating system in RAM that had not been

signed by Apple, which is what the government is requesting here. Similarly, Mr. Perino points to a hacking tool the FBI created that supposedly allowed it to brute force the device passcode on older iPhones.

22. These descriptions show that the FBI, along with its partners, currently have, and have had in the past, the capability to develop the types of code that Apple is being asked to create.

23. Mr. Perino is incorrect, however, in his suggestion that Apple can use these third-party items, add Apple's signature, and load the finished product on to the subject device to accomplish the result that the government seeks with less effort than what I described in my initial declaration.

24. Using the allegedly already existing software code that Mr. Perino identifies would not be an appropriate way to accomplish what the government wants. Setting aside the legal question of whether Apple can incorporate a software tool created by some other party (such as the Cellebrite UFED tool Mr. Perino identifies) for this purpose, Apple would not save time and effort by incorporating unfamiliar third-party code that has never been used and deployed by Apple before, and it would introduce a host of new issues and potential risks that would need to be addressed.

25. Before Apple utilized any unknown third-party created code, Apple would need to fully audit and inspect that code to understand how it functions (including to ensure it is not malware), how it would need to be modified, and how it would need to interact with the Apple-created code necessary to accomplish the task. Apple would also need to modify each separate component piece of software to combine it into a single operating system (the new GovtOS).

26. Once the operating system is created it would still need to go through Apple's quality assurance and security testing process as described in paragraphs 30-34 of my initial declaration. Indeed, this process would be even more critical if Apple were relying on software created by third parties that Apple had never deployed on its devices. Once the new GovtOS is quality assured and security tested, it will then need

to be deployed on the subject device as described in paragraphs 35-38 of my initial declaration. This endeavor would save neither time nor effort, even if possible.

27. The engineering efforts involved in these development, quality assurance and security testing processes can only be performed by a limited set of Apple employees with the appropriate expertise, who will necessarily be diverted from contributing to their normal work of developing and securing iOS. The overwhelming majority of Apple's employees could not perform this task.

28. More importantly, the historical security vulnerabilities and jailbreak incidents Mr. Perino identifies underscore the constant battle Apple is engaged in to identify and close off security vulnerabilities. I believe that Apple's iOS platform is the most-attacked software platform in existence. Each time Apple closes one vulnerability, attackers work to find another. This is a constant and never-ending battle. Mr. Perino's description of third-party efforts to circumvent Apple's security demonstrates this point. And the protections that the government now asks Apple to compromise are the most security-critical software component of the iPhone—any vulnerability or back door, whether introduced intentionally or unintentionally, can represent a risk to all users of Apple devices simultaneously.

29. This evolution of attack technology described in Mr. Perino's declaration is a vivid illustration of why Apple is always striving to increase the security of its devices. Mr. Perino makes clear that third parties have already come close to developing a tool that would defeat part of iOS's present security capabilities.

30. Mr. Perino also asserts in Paragraph 28(d) of his declaration that recent publicly available jailbreaks of Apple phones have been applied from within the iPhone user interface, after a device has been unlocked. Mr. Perino's inference is that an iPhone cannot be jail broken from the lock screen. However, particularly given the past exploits that have bypassed the lock screen and the present-day reality of innumerable security firms, malicious actors, cybercriminals and potential adversaries of the United States constantly seeking vulnerabilities to exploit in a dominant

software platform, it is not reasonable to draw such a conclusion based solely on publicly revealed exploits. Additionally, new jailbreaks for iOS versions after 9.0.2 continue to be created. (*See* "Pangu Releases a Jailbreak for iOS 9.1," 9To5Mac, Mar. 11, 2016, http://9to5mac.com/2016/03/11/pangu-ios-9-1-jailbreak-released/.)

31. Paragraphs 30 through 35 of the Perino Declaration discuss the role that the Unique ID ("UID") plays in the data protection process. Mr. Perino calls the UID "unknowable" and because of this concludes that any encrypted data on the subject device must be decrypted on the subject device itself (as opposed to being extracted in encrypted form and decrypted elsewhere). I would not characterize the UID as "unknowable." While it is designed not to be known, it is certainly not impossible for someone to determine the UID.

32. Paragraphs 37 through 39 of the Perino Declaration discuss the potential for the government to have obtained more recent data from the subject device through an iCloud backup had the FBI not instructed the San Bernardino County Public Health Department ("SBCPHD") to change the iCloud password associated with the account. Mr. Perino asserts that even if the device did perform an iCloud backup "the user data would still be encrypted with the encryption key formed from the 256 bit UID and the user's passcode."

33. The statement that even if the device did perform an iCloud backup "the user data would still be encrypted with the encryption key formed from the 256 bit UID and the user's passcode" is incorrect. Data backed up to iCloud is not encrypted with a user's passcode.

34. As noted above, I also reviewed the Supplemental Pluhar Declaration. I believe that declaration contains several mistakes. For example, in paragraph 10(a), Agent Pluhar claims that the device's keyboard cache would not backup to iCloud and that such keyboard cache "contains a list of keystrokes typed by the user on the touchscreen." This is false. The keyboard cache in iOS 9 does not contain a list of keystrokes typed by the user, or anything similar.

35. Agent Pluhar also makes incorrect claims in paragraph 10(b). Agent Pluhar claims that exemplar iPhones that were used as restore targets for the iCloud backups on the subject device "showed that . . . iCloud back-ups for 'Mail,' 'Photos,' and 'Notes' were all turned off on the subject device." This is false because it is not possible. Agent Pluhar was likely looking at the wrong screen on the device. Specifically, he was not looking at the settings that govern the iCloud backups. It is the iCloud backup screen that governs what is backed up to iCloud. That screen has no "on" and "off" options for "Mail," "Photos," or "Notes."[1]

36. In fact, users cannot exclude individual Apple apps on a one-by-one basis from backing up to iCloud, except that a user can choose to have their photos stored in their iCloud Photo Library instead of in their iCloud backup, or not stored at all. Once iCloud backup is enabled, all other Apple apps will backup with no configurable settings for the user. Thus, contacts, calendar events, reminders, notes, device settings, call history, home screen and app organization, iMessage, text (SMS), and MMS messages all would have been available from Apple had iCloud backup been enabled.

I declare under penalty of perjury under the laws of the United States of America that the foregoing is true and correct.

Executed this 15th day of March 2016 in Washington, D.C.

By:________________________
Erik Neuenschwander
Manager of User Privacy
Apple Inc.

[1] The screen Agent Pluhar may have been referring to pertains to functions that allow certain App data to be synchronized across multiples devices connected to the same iCloud account.